**MS231E**

# C-867 PILine® Controller

## User Manual

Version: 3.1.0 Date: 10/24/2024

**This document describes the following product:**

- **C-867.2U2**
  Piezo motor controller for PILine® systems, 2 axes, USB, RS-232, TCP/IP, SPI, I/O, analog or digital joystick, networkable via daisy chain

Physik Instrumente (PI) SE & Co. KG, Auf der Römerstraße 1, 76228 Karlsruhe, Germany
Phone +49 721 4846-0, fax +49 721 4846-1019, e-mail info@pi.ws, www.pi.ws

PI

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®

Notes on brand names and third-party trademarks:
Microsoft® and Windows® are registered trademarks or trademarks of Microsoft Corporation in the USA and/or other countries.
EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
TwinCAT® is a registered trademark of and licensed by Beckhoff Automation GmbH.
LabVIEW, National Instruments and NI are trademarks of National Instruments. Neither the driver software nor the software programs offered by PI or other goods and services are connected to or sponsored by National Instruments.
Python® is a registered trademark of Python Software Foundation.
BiSS is a registered trademark of iC-Haus GmbH.

The following designations are protected company names, trademarks or registered trademarks of other owners:
Linux, MATLAB, MathWorks, FTDI
These designations are used for identification purposes only.

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 (https://www.physikinstrumente.com/fileadmin/user_upload/physik_instrumente/files/legal/General-Software-License-Agreement-Physik-Instrumente.pdf) and in the Third-Party Software Notes (https://www.physikinstrumente.com/fileadmin/user_upload/physik_instrumente/files/legal/Third-Party-Software-Note-Physik-Instrumente.pdf) on our website.



Original instructions
First printing: 10/24/2024
Document number: MS231E, ASt, Version 3.1.0

Subject to change. This manual is superseded by any new release. The latest respective release is available for download on our website (https://www.physikinstrumente.com/en/).

# Contents

# 1 About this Document

## 1.1 Objective and Target Audience of this User Manual

This user manual contains the information required for using the C-867 as intended.

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

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

## 1.2 Symbols and Typographic Conventions

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

***CAUTION***

**Dangerous situation**
Failure to comply could result in minor injuries.

- Precautions to avoid the risk.

***NOTICE***

**Dangerous situation**
Failure to comply could result in damage to the equipment.

- Precautions to avoid the risk.

***INFORMATION***

Information for easier handling, tricks, tips, etc.

| Symbol/Label | Meaning |
|---|---|
| **RS-232** | Label on the product indicating an operating element (example: RS-232 interface socket) |
|  | Warning sign on the product referring to detailed information in this manual. |
| ***Start > Settings*** | Menu path in the PC software (example: to open the menu, the ***Start*** and ***Settings*** menu items must be selected successively) |

| Symbol/Label | Meaning |
|---|---|
| `POS?` | Command line or a command from PI's General Command Set (GCS) (example: command to get the axis position) |
| ***Device S/N*** | Parameter name (example: parameter where the serial number is stored) |
| *5* | Value that must be entered or selected via the PC software |

## 1.3 Definition of Terms

| Term | Explanation |
|---|---|
| Axis | Also referred to as "logical axis". The logical axis represents the motion of the mechanics in the firmware of the C-867. For mechanics that allow motion in several directions (e.g., in X, Y, and Z), each direction of motion corresponds to a logical axis. |
| Positioner | Mechanics connected to the C-867. In the case of positioners with just one motion axis, the designation "axis" is synonymous with "positioner". Positioners that allow motion in several axes are also designated as "multi-axis positioners". For these positioners, a distinction must be made between the individual axes. |
| Control value | The control value is converted by the D/A converter of the C-867 into an analog control voltage. The control voltage is the input variable for the PILine® drive electronics of the C-867. The PILine® drive electronics converts the control voltage into the piezo voltage for the axis of the positioner. |
| Two-phase control | A PILine® piezo motor has a separate piezo segment for the positive and the negative direction of motion. Depending on the direction of motion, only the corresponding segment is normally driven by the piezo voltage. The two-phase control allows the second segment to be driven parallel to the first segment at specified time intervals, in order to interrupt the feed motion of the piezo motor for the duration of the interval. Depending on the application, interruption of the feed motion at intervals can improve the settling behavior of the axis. |
| Absolute measuring position sensor | Sensor (encoder) for capturing changes of position or changes of angle. Signals from the absolute-measuring position sensor are used for axis position feedback. After the controller is switched on, absolute target positions can be commanded and reached immediately. Referencing is not necessary. |
| Incremental position sensor | Sensor (encoder) for detecting changes of position or changes of angle. Signals from the incremental position sensor are used for axis position feedback. After the controller is switched on, referencing must be done before absolute target positions can be commanded and reached. |

| Term | Explanation |
|---|---|
| Dynamics profile | Comprises the target position, velocity, and acceleration of the axis calculated by the profile generator of the C-867 for each point in time of a point-to-point motion. The calculated values are called "commanded values". |
| Trajectory | A trajectory is a motion along a path made up of points that were externally calculated and loaded to the C-867 (target positions) and that are travelled according to a specified chronological interval. |
| Daisy chain | Wiring diagram by which one controller is connected to the next in sequence (series connection principle). Here 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. |
| HID | HID (Human Interface Device) refers to an input or output device connected to the controller and is intended for manual operation. Depending on the controller, the connection can be made via USB, analog or digital interfaces. Joysticks and gamepads are typical HIDs. |
| HID control | Control of the motion variables of the axes of the C-867 by displacing the axes of HIDs. |
| Volatile memory | RAM module where the parameters are saved when the controller is switched on (working memory). The parameter values in the volatile memory determine the current behavior of the system.<br>The parameter values in the volatile memory are also referred to as "Active Values" in the PC software from PI. |
| Nonvolatile memory | Memory module (read-only memory, e.g., EEPROM or flash memory) from which the default values of the parameters are loaded into the volatile memory when the controller is started.<br>In the PC software from PI, the parameter values in the nonvolatile memory are also referred to as "startup values". |
| Default settings | Parameter values and parameter-independent settings to which the volatile and nonvolatile memories of the C-867 can be reset by the user if necessary. After being reset to default settings, the parameter values must be adapted before motions of the connected positioner can be started.<br><br>Default settings **cannot** be changed by the user. |
| Firmware | Software that is installed on the controller. |
| PC software | Software installed on the PC. |
| GCS | PI General Command Set: command set for PI controllers |

## 1.4 Figures

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

## 1.5 Other Applicable Documents

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

| Product | Document |
|---|---|
| Short instructions for the installation and startup of the C-867.2U2 | MS242EK Short instructions for digital motor controllers |
| PI GCS driver library for use with NI LabVIEW software | SM158E Software Manual |
| PI MATLAB Driver GCS 2.0 | SM155E Software Manual |
| PI GCS 2.0 DLL | SM151E Software Manual |
| GCS array<br>data format description | SM146E Software Manual |
| PIMikroMove | SM148E Software Manual |
| PIStages3Editor<br>Software for managing the positioner database | SM156E Software Manual |
| PIUpdateFinder: Updating PI software | A000T0028 User Manual |
| PIFirmwareManager: Updating controller firmware | SM164E User Manual |
| Downloading manuals from PI: PDF file with links to the manuals for digital electronics and software from PI. Supplied with the PI software. | A000T0081 Technical Note |

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

## 1.6 Downloading Manuals

*INFORMATION*

If a manual is missing or problems occur with downloading:

➢ Contact our customer service department (p. 309).

### Downloading manuals

1. Open the website **www.pi.ws**.
2. Search the website for the product number (e.g., C-867).
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 Safety

## 2.1 Intended Use

The C-867 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 C-867 is intended for the operation of positioners with PILine® ultrasonic piezo motors and Sub-D 15 (m) connectors.

The C-867 is intended for closed-loop operation with incremental or absolute-measuring position sensors. In addition, it can read and process the reference point and limit switch signals from the positioner connected.

The C-867 may only be used in compliance with the technical specifications and instructions in this user manual. The user is responsible for process validation.

The C-867 must not be used for purposes other than those stated in this user manual. In particular, the C-867 must not be used to drive ohmic or inductive loads.

## 2.2 General Safety Instructions

The C-867 is built according to state-of-the-art technology and recognized safety standards. Improper use of the C-867 may result in personal injury and/or damage to the C-867.

- Use the C-867 for its intended purpose only, and only when it is in perfect condition.
- Read the user manual.
- Immediately eliminate any faults and malfunctions that are likely to affect safety.

The operator is responsible for installing and operating the C-867 correctly.

- Install the C-867 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Use the supplied components (power supply, adapter, power cord) to connect the C-867 to the power source.
- If one of the supplied components for connecting to the power source has to be replaced, use a sufficiently dimensioned component.

## 2.3 Organizational Measures

### User manual

- Always keep this user manual together with the C-867. The latest versions of the user manuals are available for download on our website (p. 4).
- Add all information from the manufacturer such as supplements or technical notes to the user manual.
- If you give the C-867 to other users, include this user manual as well as all other relevant information provided by the manufacturer.
- Do the work only if the user manual is complete. Missing information due to an incomplete user manual can result in minor injury and damage to equipment.
- Install and operate the C-867 only after you have read and understood this user manual.

### Personnel qualification

The C-867 may only be installed, started, operated, maintained, and cleaned by authorized and appropriately qualified personnel.

## 2.4 European Declarations of Conformity

For the C-867, 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

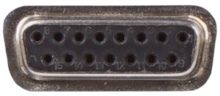
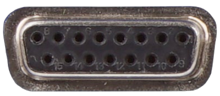
RoHS: EN IEC 63000

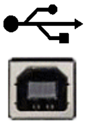
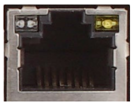
# 3 Product Description

## 3.1 Product View

### 3.1.1 Front Panel


Figure 1: Front panel of the C-867.2U2

| Element | Type | Function |
|---|---|---|
|  | Toggle switch | On/off switch:<br>O: C-867 switched off<br>—: C-867 switched on |
| **24 V DC**<br>**4.5 A**<br> | M8 connector, 4-pole (m) (p. 318) | Connector for the supply voltage<br>A protective cap is screwed on the connector before dispatching:<br> |
|  | Threaded bolt for protective earth conductor | Protective earth connector (p. 58)<br>The threaded bolt must be connected to a protective earth conductor because the C-867 is not grounded via the power adapter connector. |
| **Axis 1**<br>**71 V rms**<br> | D-sub 15 (f) (p. 314) | Connections for positioners<br>Only for PILine® ultrasonic piezo motors!<br>Per axis:<br>▪ Outputs for piezo voltage<br>▪ Input of the signals of the position sensor<br>▪ Signal input from the limit switches and reference switch<br>▪ Output of the supply voltage for position sensor, reference and limit switches<br>▪ Input for signals of the ID chip |
| **Axis 2**<br>**71 V rms**<br> | | |

| Element | Type | Function |
|---|---|---|
|  **RS-232 In** | D-sub 9 (m) (p. 314) | Serial interface for connecting to the PC or preceding controller in a daisy chain network |
|  **RS-232 Out** | D-sub 9 (f) (p. 314) | Serial interface for connecting to the subsequent controller in a daisy chain network |
|  | USB type B | Universal serial bus for connecting to the PC |
| **SPI**  | Display port | Serial connection to a serial peripheral interface (SPI) master unit<br>If you wish to use the SPI connection, contact our customer service department (p. 309). |
| **Ethernet**  | RJ45 socket | Ethernet interface for communication via TCP/IP |
|  **Dig. Joystick** | USB type A | Connector for a digital HID (e. g. joystick or gamepad) |
| **I/O**  | Mini-DIN, 9-pole (f) (p. 316) | Digital inputs/outputs:<br>▪ Outputs: Controlling external devices<br>▪ Inputs: Use in macros as switch signals or for HID control<br>Analog inputs:<br>▪ Use in macros or for scanning processes |
| **Analog Joystick**  | Mini-DIN, 6-pole (f) | Connector for an analog HID<br>▪ Inputs for signals from the axes and buttons of the joystick<br>▪ Output for the supply voltage of the joystick |
|  **STA** | LED, green | Controller state:<br>▪ Lights up continuously: C-867 is ready for normal operation<br>▪ Flashing: C-867 is in firmware update mode<br>▪ Off: C-867 is not connected to the supply voltage |

| Element | Type | Function |
|---|---|---|
|  ERR | LED red | Error indicator:<br>▪ On: Error (error code ≠ 0)<br>▪ Off: No error (error code = 0)<br>The error code can be queried with the `ERR?` command. The query resets the error code to zero and the LED is switched off. |
| Analog In  | Socket for a TRS jack plug (p. 317) | Analog input socket (-10 V to +10 V) for connecting an HID (joystick) |

### 3.1.2 Type Plate

| Labeling | Function |
|---|---|
| | Data matrix code (example; contains the serial number) |
| **C-867.2U2** | Product name |
| **PI** | Manufacturer's logo |
| **116056789** | Serial number (example), individual for each C-867<br>Meaning of each position (from the left): 1 = internal information, 2 and 3 = year of manufacture, 4 to 9 = consecutive number |
| Country of origin: Germany | Country of origin |
| ⚠ | Warning sign "Pay attention to the manual!" |
| | Old equipment disposal (p. 319) |
| CE | CE conformity mark |
| WWW.PI.WS | Manufacturer's address (website) |

## 3.2 Scope of Delivery

| Article | Component |
|---|---|
| C-867.2U2 | PILine® Motion Controller |
| C-501.24120M8 | Wide-range-input power supply 24 V 120 W, M8 connector |
| 3763 | Power cord |
| 000011448 | USB cable (USB-A/USB-B) for connection to the PC |
| 000084853 | 4 adhesive feet for C-867.2U2 |
| C-990.CD1 | Data storage device with PC software from PI |

| Article | Component |
|---|---|
| MS242EK | Short instructions for digital motor controllers |

## 3.3 Optional Accessories

| Item | Component |
|---|---|
| C-862.CN1 | Cable for daisy chain network, 1 m |
| C-862.CN2 | Cable for daisy chain network, 3 m |
| C-815.553 | FTP patch cable 1:1<br>Straight-through network cable for connecting to the PC via a TCP/IP network |
| C-815.563 | FTP patch cable crossed<br>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; see "Available HIDs" (p. 124) for details |
| C-819.30 | Analog joystick for 3 axes; see "Available HIDs" (p. 125) for details |
| C-819.JD | Digital joystick for 2 axes, 3 programmable buttons, USB; details see "Available HIDs" (p. 126) |
| C-170.PB | Pushbutton box with 4 buttons and 4 LEDs<br><br>Connection to the **I/O** socket of the C-867, sends 4 TTL input signals and displays the state of the 4 digital outputs via the LEDs. |
| C-170.IO | I/O cable, 2 m, open end (p. 316) |
| C-867.L01 | Firmware feature "Slow Motion Drive" for PILine® controllers |

To order, contact our customer service department (p. 309).

## 3.4 Overview of PC Software

### 3.4.1 PI Software Suite

A data storage device with the PI Software Suite is included in the C-867's scope of delivery (p. 11). 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 | Short description | Recommended use |
|---|---|---|
| Dynamic program library for GCS | Allows software programming for the C-867 with programming languages such as C++. The functions in the dynamic program library are based on the PI General Command Set (GCS). | For users who would like to use a dynamic program library for their application.<br>Is required for PIMikroMove.<br>Is required for NI LabVIEW drivers. |
| Drivers for use with NI LabVIEW software | NI LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments).The driver library is a collection of virtual instrument drivers for PI controllers.<br>The drivers support the PI GCS. | For users who want to use NI LabVIEW to program their application. |
| MATLAB drivers | MATLAB is a development environment and programming language for numerical calculations (must be ordered separately from MathWorks).<br>The PI MATLAB driver consists of a MATLAB class that can be included in any MATLAB script. This class supports the PI GCS.<br>The PI MATLAB driver does not require any additional MATLAB toolboxes. | For users who want to use MATLAB to program their application. |
| USB driver | Driver for the USB interface | For users who want to connect the controller to the PC via the USB interface. |

### User software

| PC software | Short description | Recommended use |
|---|---|---|
| PIMikroMove | Graphic user interface for Windows with which the C-867 and other controllers from PI can be used.<br>▪ The system can be started without programming effort<br>▪ Graph of motions in open-loop and closed-loop operation<br>▪ Macro functionality for storing command sequences on the PC (host macros)<br>▪ Support of HID devices<br>▪ Complete environment for command entry, for trying out different commands<br>PIMikroMove uses the dynamic program library to supply commands to the controller. | For users who want to do simple automation tasks or test their equipment before or instead of programming an application. A log window showing the commands sent makes it possible to learn how to use the commands. |
| PITerminal | Terminal program that can be used for nearly all PI controllers. | For users who want to send GCS commands directly to the controller. |
| PIStages3Editor | Program for opening and editing positioner databases in .db format. | For users who want to deal with the contents of positioner databases more intensively. |
| PIUpdateFinder | Checks the PI software installed on the PC. If more current versions of the PC software are available on the PI server, downloading is offered. | For users who want to update the PC software. |
| PIFirmwareManager | Program for user support when updating firmware of the C-867. | For users who want to update the firmware. |

## 3.5 Positioner Databases

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

| Database file name | 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<br>New parameter sets can be created, edited, and saved. |
| X1000.db<br>e.g.:<br>M-xxxxxxx.db | Includes the parameter set for a custom positioner. In order for the parameter set to be selected in the PC software, it must be added to the PISTAGES3.DB first, see "Installing Custom Positioner Databases" (p. 66). |

The positioner database only contains some of the information that is required to operate a positioner with the <product name>. Further information is loaded as parameter values to the volatile memory of the <product name> from the ID chip (p. 15) of the positioner when the C-867 is switched on or rebooted.

Parameters that are loaded from the positioner database or the ID chip, are described in the parameter overview (p. 282).

For more information on the positioner database, see the manuals for the PIStages3Editor and the PI GCS program library.

***INFORMATION***

If the pistages2.dat and pimicosstages2.dat positioner databases are on your PC:

Positioner databases in .dat format are only installed for compatibility reasons and **not** used for the C-867 described in this manual.

## 3.6 ID Chip Detection

Positioners with PILine® ultrasonic piezo motors and D-sub 15 connectors have an ID chip in the connector on which the following data is saved as parameters:

- Information on the positioner: Type, serial number, date of manufacture, hardware version
- Signal type output by the position sensor
- When the position sensor outputs sine/cosine signals that are interpolated in the C-867: Settings for interpolation rate, as well as hysteresis, phase and offset corrections, and gain values

The data of the connected positioner is loaded from the ID chip into the volatile memory of the C-867 when the C-867 is switched on (p. 69) or rebooted.

The parameter values in the C-867's volatile memory can be queried and written to the nonvolatile memory, see "Adapting Settings" (p. 273).

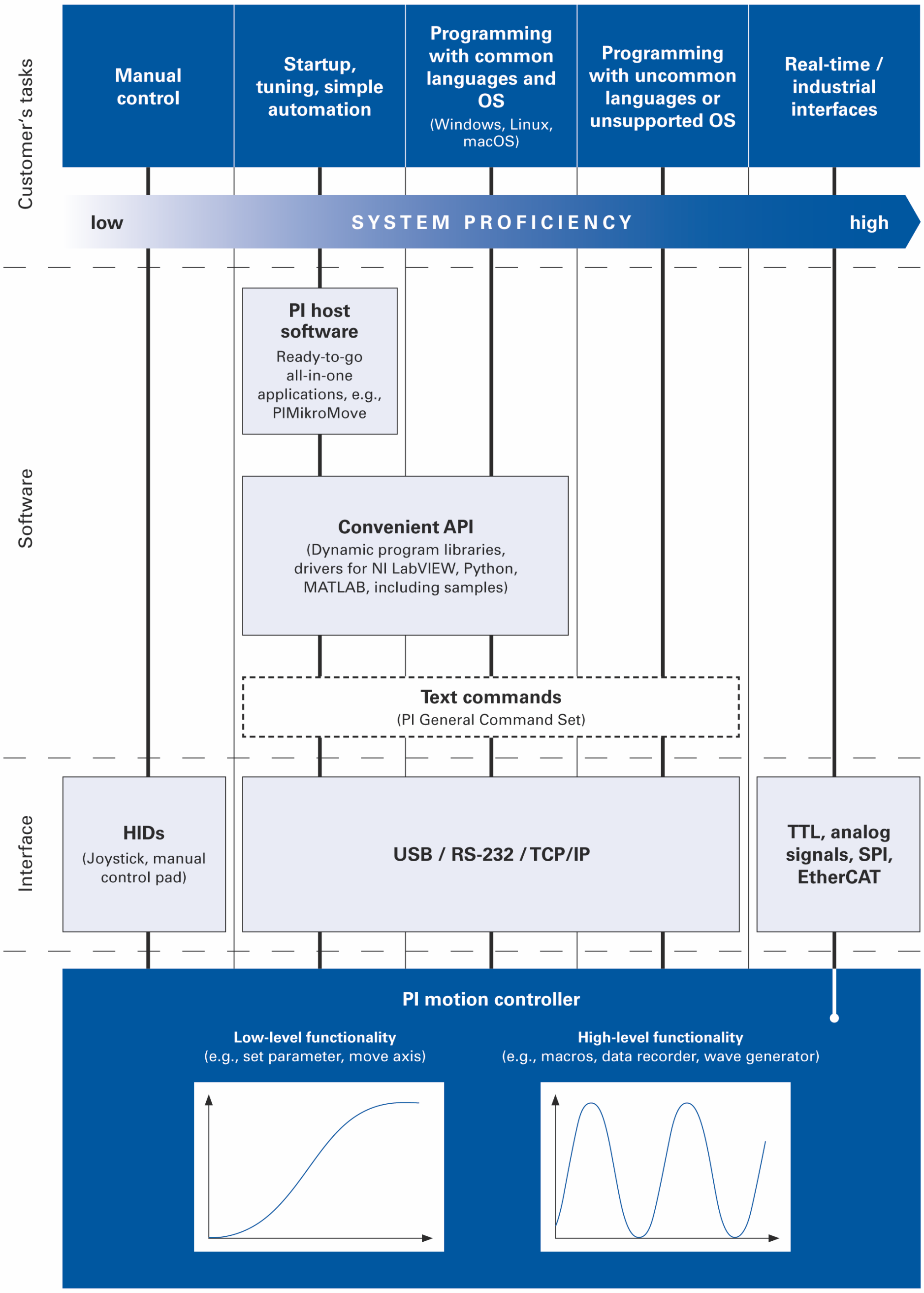
***INFORMATION***

The ID chip only contains some of the information that is required to operate the positioner with the C-867. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 14) into the volatile memory of the C-867.

# 3.7 Communication Interfaces

## 3.7.1 Control of PI Systems

Basically, systems from PI can be controlled as follows:



### Communication interfaces available

The C-867 can be controlled from a PC. The following interfaces of the C-867 can be used for connection to the PC:

- USB connection
- Serial RS-232 connection
- TCP/IP connection

The C-867 must not be connected to the PC via the USB and the serial RS-232 interface at the same time.

### Daisy chain network

With a daisy chain network, up to 16 controllers can be connected to the PC via a single interface connection. Interlinking occurs in series.

### Default communication settings

The communication settings of the C-867 can be queried with the IFS? (p. 204) command and changed with IFC (p. 200). The default values for the communication interfaces are:

| Interface | Property | Default value |
|---|---|---|
| | DEVADR | 1<br>Controller address; can be changed for use in a daisy chain network |
| TCP/IP | IPSTART | Startup behavior: 0<br>The IP address defined by IPADR is used. |
| | IPADR | 192.168.0.75:50000<br>IP address: 192.168.0.75, port: 50000 |
| | IPMASK | Bit mask for subnet: 255.255.255.255 |
| | IPGTWAY | 192.168.0.1<br>Standard gateway for TCP/IP communication |
| | MACADR | MAC level address: 00:04:A3:3A:1F:ED |

## 3.8 Functional Principles

### 3.8.1 Block Diagram

The C-867 can control up to two logical positioner axes. The following block diagram shows how the C-867 generates the piezo voltage for the axes connected.

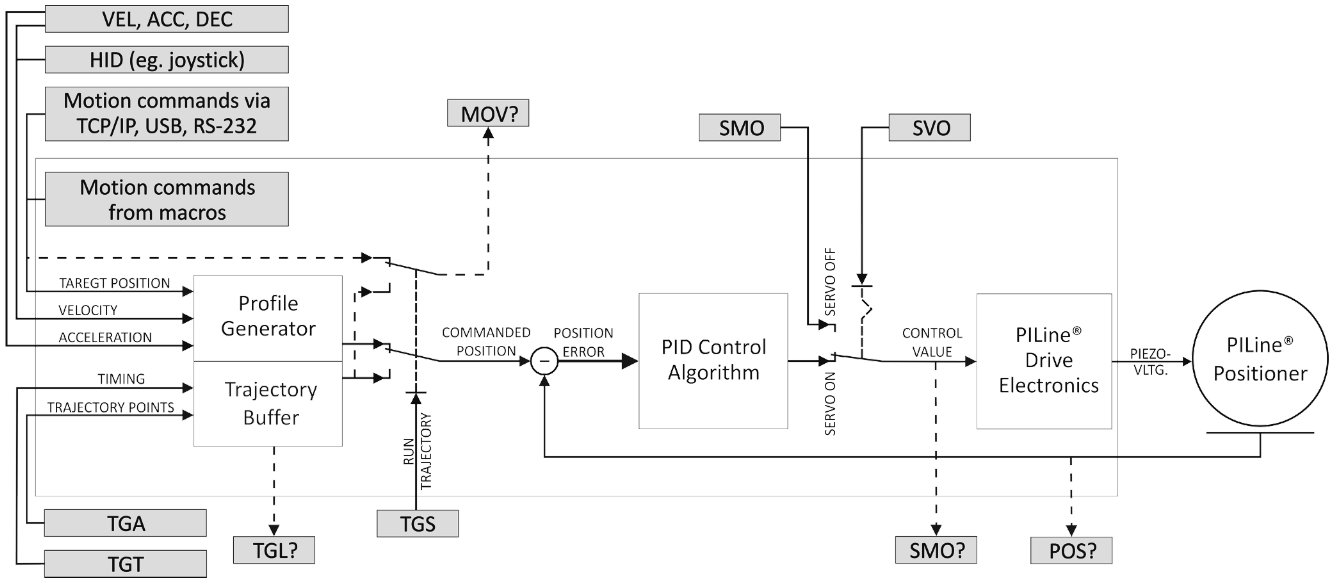



Figure 2: C-867: Control value generation

The C-867 supports positioners with PILine® ultrasonic piezo motor and incremental or absolute-measuring position sensor.

### 3.8.2 Commandable Elements

The following table contains the elements that can be commanded with GCS commands (p. 151).

| Element | Number | Identifier | Description |
|---|---|---|---|
| Logical axes | 2 | 1, 2 (modifiable) | A logical axis represents the motion of the positioner in the firmware of the C-867. It corresponds to an axis of a linear coordinate system.<br>Motion for logical axes is commanded in the C-867's firmware (i.e., for the positioner's direction of motion). The motion commands `MOV` and `MVR`, for example, are available in closed-loop operation. Motion in open-loop operation is triggered by `SMO`.<br>The axis identifier can be queried with the `SAI?` command and modified with the `SAI` command. It can |

| Element | Number | Identifier | Description |
|---|---|---|---|
| | | | consist of up to 8 characters; valid characters are 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ-_<br>The new axis identifier is saved automatically to nonvolatile memory and is therefore still available even after a reboot or after the next switch-on. When the `DPA` command is used, the axis identifier is reset to the default setting in the volatile and nonvolatile memory.<br>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`. |
| Trajectories | 2 | 1, 2 | The number of trajectories corresponds to the number of logical axes. Each trajectory is permanently allocated to a logical axis.<br>Trajectories are commanded with TG* commands.<br>For further information, refer to "Trajectories for Motion Paths" (p. 95). |
| 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 (p. 316). 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).<br>Further analog input lines are found at the **Analog Joystick** and **Analog In** (p. 317) sockets. These lines are not output via `TAC?` and `TAV?`.<br>The values of all inputs can be recorded via record option 81 of the `DRC` command. |
| Digital inputs | 4 | 1 to 4 | 1 to 4 identify digital input lines 1 to 4 of the **I/O** socket (p. 316), which can also be used as analog inputs (see above).<br>Refer to "Digital Input Signals" (p. 109) for further information. |
| Digital outputs | 4 | 1 to 4 | 1 to 4 identify digital output lines 1 to 4 of the **I/O** socket (p. 316).<br>Refer to "Digital Output Signals" (p. 100) for further information. |
| HIDs | 6 | 1 to 6 | An analog HID (p. 2) can be connected to the C-867.2U2: Command as HID 1.<br>Up to five digital human interface devices can be connected to the **Dig. Joystick** socket of the C-867 via a USB hub: Command as HID 2 to 6.<br>Refer to "Controlling with an HID" (p. 114) for further information. |

| Element | Number | Identifier | Description |
|---|---|---|---|
| Axes and buttons of the HIDs | x | 1 to x | The axes and buttons of HIDs can be connected to the C-867 as follows:<br>Two axes and two buttons of an analog HID can be connected to the **Analog Joystick** socket:<br>▪ Pin 4 (0 to 3.3 V): Command as axis 1 of the HID 1<br>▪ Pin 2 (0 to 3.3 V): Command as axis 2 of the HID 1<br>▪ Pin 5 (0 or 3.3 V): Command as button 1 of the HID 1<br>▪ Pin 6 (0 or 3.3 V): Command as button 2 of the HID 1<br>Two axes of an analog HID can be connected to the **Analog In** (p. 317) socket:<br>▪ Tip (-10 to 10 V): Command as axis 3 of HID 1<br>▪ Ring (-10 to 10 V): Command as axis 4 of HID 1<br>Two axes of an analog HID can be connected to the **I/O** (p. 316) socket:<br>▪ Pins 1 and 2 (0 to +5 V): Command as axis 5 of HID 1<br>▪ Pins 3 and 4 (0 to +5 V): Command as axis 6 of HID 1<br>The number of commandable axes and buttons connected to the **Dig. Joystick** socket depends on HID connected to the HID. Information on the commandable axes and buttons of the connected HIDs can be queried with the `HIS?` command. |
| Data recorder tables | 4 | 1 to 4 | The C-867 has 4 data recorder tables (query with `TNR?`) with 8192 data points per table. |
| Controller address | 1 | 1 to 16 | The controller address can be set in the range from 1 to 16 with the IFS (p. 202) command. In a daisy chain (p. 76), each controller must have a unique address (p. 145). |
| Overall system | 1 | 1 | C-867 as an overall system |

### 3.8.3 Important Components of the Firmware

The firmware of the C-867 provides the following functional units:

| Component | Description |
|---|---|
| Parameters | Parameters reflect the properties of the positioner connected (e.g., travel range) and specify the behavior of the C-867 (e.g., settings for the servo algorithm).<br>The parameters can be divided into the following categories:<br>▪ Protected parameters whose default settings cannot be changed<br>▪ Parameters that must be set by the user to adapt to the application<br>For further information, see "Adapting Settings" (p. 273).<br>In the case of positioners with ID chip, the values of some parameters are |

| Component | Description |
|---|---|
| | stored on the ID chip. They are loaded to the volatile memory when switching on or rebooting the C-867. |
| Command levels | The command levels determine the write permission for the parameters.<br>The current command level can be changed with the `CCL` command. This may require entering a password. |
| ASCII commands (GCS) | Communication with the C-867 can be managed using the commands of the PI General Command Set (GCS; version 2.0). The GCS is independent of the hardware (controller, positioners connected).<br>Examples of the use of GCS:<br>▪ Configuring the C-867<br>▪ Setting the operating mode<br>▪ Starting motion of the positioner<br>▪ Getting system and position values<br>You can find a list of the available commands in the "Command Overview" section (p. 147). |
| Profile generator | During point-to-point motion in closed-loop operation, the profile generator performs calculations to specify the target position, velocity, and acceleration of an axis for each point in time during a motion. The result is the dynamics profile.<br>For further information, see "Generation of Dynamics Profile" (p. 27). |
| Trajectory buffer | For motion along freely definable paths, externally calculated trajectory points (target points) are loaded to the trajectory buffer of the C-867. The trajectory points are travelled according to a specified chronological interval in closed-loop operation.<br>For further information, see "Trajectories for Motion Paths" (p. 95). |
| 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 PID servo algorithm.<br>For further information, see "Servo Algorithm and Other Control Value Corrections" (p. 30). |
| Data recorder | The C-867 contains a real-time data recorder (p. 98). The data recorder can record various signals (e. g., position, control value) from different data sources (e. g., logical axes). |
| Macros | The C-867 can save macros (p. 126). 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 C-867 is switched on or rebooted. The startup macro simplifies stand-alone operation (operation without a connection to the PC).<br>Further information can be found in the "Controller Macros" section (p. 126). |

The firmware can be updated with a tool (p. 299).

### 3.8.4 Operating Modes

The C-867 supports the following operating modes:

| Operating mode | Description |
|---|---|
| Closed-Loop Operation<br>Servo mode on | The commanded position for the axes comes from one of the two following sources:<br>▪ Dynamics profile (p. 27):<br>A profile generator calculates the dynamics profile from the values specified for target position, velocity, acceleration, and deceleration.<br>▪ Trajectory buffer (p. 95):<br>The motion follows a path made up of points that were externally calculated and loaded to the C-867 (target positions) and that are travelled according to a specified chronological interval.<br>The position error resulting from the difference between the commanded target position and the actual position (sensor feedback) runs through a PID servo algorithm (**p**roportional **i**ntegral **d**erivative). Additional corrections can be made as well.<br>The result is the control value that is converted into the control voltage for the PILine® drive electronics integrated in the C-867.<br>Refer to "Servo Algorithm and Other Control Value Corrections" (p. 30) for further information. |
| Open-loop operation<br>Servo mode off | The C-867 does not calculate a dynamics profile in open-loop operation and a trajectory cannot be followed.<br>The C-867 does **not** evaluate the signals of the position sensor. As a result, the positioner can move unbraked to the end of the travel range and, despite the limit switch function, strike the hard stop. |

***INFORMATION***

The C-867 is intended for closed-loop operation with 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.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a startup macro that starts the C-867 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 134).
- Avoid motion in open-loop operation.

### 3.8.5 Physical Units

The C-867 supports various units of length for positions. The adaptation is made via a factor with which the counts of the encoder are converted into the physical unit of length required. The conversion factor is set with the following parameters:

<table>
<tr><th>Parameters</th><th>Description and Possible Values</th></tr>
<tr><td>Numerator Of The Counts-Per-Physical-Unit Factor<br>0xE</td><td rowspan="2">Numerator and denominator of the factor for counts per physical length unit<br>1 to 1,000,000 for each parameter.<br>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.<br>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.<br>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.</td></tr>
<tr><td>Denominator Of The Counts-Per-Physical-Unit Factor<br>0xF</td></tr>
</table>

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

| Parameters | Description and Possible Values |
|---|---|
| ***Axis Unit***<br>0x07000601 | Unit symbol<br>Maximum of 20 characters.<br>For example, the unit symbol is "MM", if the factor for the counts per physical unit of length is set with the 0xE and 0xF parameters so that the encoder counts are converted into millimeters. The unit symbol for rotation stages is normally "deg".<br>The value of the parameter 0x07000601 is not evaluated by the C-867 but is used by the PC software for display purposes.<br>Examples:<br>1 encoder count = 100 nm<br>Counts per physical length unit: 10000:1<br>→ Unit symbol: mm<br>1 encoder count = 0.254 mm<br>Counts per physical length unit: 100:1<br>→ Unit symbol: inch |

### 3.8.6 Supported Motor Types

The C-867 supports all types of PILine® positioners and drives integrated in PILine® ultrasonic piezo motors currently offered by PI. The adaptation to the connected motor type is done using the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Maximum Motor Output***<br>0x9 | Maximum permissible absolute measure of the control value (dimensionless)<br>0 to 32767<br>This parameter also limits the output piezo voltage. Relationship between control value and piezo voltage:<br>▪ The control value 32767 corresponds to the maximum permissible amplitude of the piezo voltage (given by the value of the parameter 0x7C).<br>▪ The control value 0 corresponds to 0 $V_{rms}$ piezo voltage.<br>Example:<br>Maximum permissible piezo voltage (0x7C) = 57 $V_{rms}$<br>Maximum permissible absolute measure of the control value (0x9) = 20000<br>→ By limiting the maximum permissible absolute measure of the control value to 20000, the C-867 outputs a maximum of 35 $V_{rms}$ (rounded) piezo voltage. |
| ***Maximum Motor Output (V)***<br>0x7C | Maximum permissible piezo voltage<br>0 to 71 $V_{rms}$<br>This parameter determines the maximum permissible amplitude of the output piezo voltage. The actual output piezo voltage is limited by the maximum permissible absolute measure of the control value; for details see parameter 0x9. |
| ***Output Frequency (kHz)***<br>0x51 | Frequency of the piezo voltage<br>0 to 500 kHz<br>This parameter determines the frequency with which the output piezo voltage oscillates in order to excite the piezo actuator in the PILine® ultrasonic piezo motor. Sources for the value of the parameter:<br>▪ When the frequency control (p. 25) is switched on and active: Determination by the frequency control<br>▪ Direct modification, e.g., with the SPA command (simultaneously switches off the frequency control) |

***INFORMATION***

Further information can be found in the user manual of your PILine® positioner.

### 3.8.7 Automatic Frequency Control

The C-867 is equipped with a frequency control that optimizes the frequency of the output piezo voltage. At the optimum operating point, the frequency of the piezo voltage is as close as possible to the resonant frequency of the connected motor. The resonant frequency of the motor is influenced by various factors:

- Motor type
- Installation conditions of the motor
- Execution of the run-in procedure
- Temperature

The frequency control operates with 1 kHz.

The frequency control can be configured via the parameters listed below:

| Parameter | Description and Possible Values |
|---|---|
| ***Frequency Control***<br>0x52 | State of the frequency control<br>0 = Frequency control switched off<br>1 = Frequency control switched on (default setting)<br>When the frequency control is switched on **and** active, it sets the value of the ***Output Frequency (kHz)*** parameter (0x51, see "Supported Motor Types" (p. 23)). The criterion for activating the frequency control is given by parameter 0x55.<br>Direct modification of the parameter 0x51 (e.g., with the SPA command) simultaneously switches off the frequency control. |
| ***Minimum Output Frequency (kHz)***<br>0x53 | Minimum frequency of the piezo voltage (kHz)<br>0 to 500 kHz<br>This parameter gives the smallest possible value for parameter 0x51 when the frequency control is switched on and active. |
| ***Maximum Output Frequency (kHz)***<br>0x54 | Maximum frequency of the piezo voltage (kHz)<br>0 to 500 kHz<br>This parameter gives the largest possible value for parameter 0x51 when the frequency control is switched on and active. |
| ***Minimum Motor Output For Frequency Control***<br>0x55 | Minimum control value for activating the frequency control<br>0 to 32767<br>When the modulus of the current control value is at least as large as the value of this parameter, the switched-on frequency control becomes active and sets the value of the parameter 0x51. |

## 3.8.8 Motion Triggering

### Motion in closed-loop operation

Motion is triggered in closed-loop operation either by commands (p. 147) or an HID, e.g., a joystick.

Motion commands and following trajectories are not permitted when HID control (p. 114) is activated for the axis.

<table>
<tr><th>Triggering motion</th><th>Commands</th><th>Description</th></tr>
<tr><td rowspan="5">Commands for point-to-point motion sent from the command line or via the PC software</td><td>MOV, MVR</td><td>Motion to absolute or relative target position</td></tr>
<tr><td>GOH</td><td>Motion to zero position</td></tr>
<tr><td>STE</td><td>Initiates a jump over a specified distance and records the response</td></tr>
<tr><td>FRF</td><td>Starts reference moves</td></tr>
<tr><td>FED</td><td>Starts moves to signal edges</td></tr>
<tr><td rowspan="2">Controller macros with commands for point-to-point motion</td><td>MAC</td><td>Calls a macro function. Permits recording, deleting, and running macros on the controller.<br>All 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.</td></tr>
<tr><td colspan="2">Additional macro commands and information see "Controller Macros" (p. 126).</td></tr>
<tr><td rowspan="2">Following trajectories</td><td>TGS</td><td>Starts the execution of an externally calculated trajectory</td></tr>
<tr><td colspan="2">For further commands and information, refer to "Trajectories for Motion Paths" (p. 95).</td></tr>
<tr><td rowspan="4">HID control</td><td>HIN</td><td>Activates or deactivates control of the C-867's axes via the HID's axes.</td></tr>
<tr><td>HIA</td><td>Configures HID control for the axes of the C-867. The following motion parameters of the C-867's axes can be controlled via the HID's axes:<br>▪ Absolute target position<br>▪ Relative target position (specifies how many times motion is to be executed over the same distance)<br>▪ Velocity<br>▪ Maximum velocity</td></tr>
<tr><td>SST</td><td>Sets the distance to be travelled for relative motion triggered by HIDs.</td></tr>
<tr><td colspan="2">For further commands, see "Commands and Parameters for HIDs" (p. 115).</td></tr>
</table>

### INFORMATION

Absolute target positions can only be commanded for positioners with incremental positions sensor when the axis has been previously referenced, refer to "Referencing" (p. 50).

#### Motion in open-loop operation

Following trajectories and HID control are not possible in open-loop operation.

Motion is triggered by the following command:

| Command | Description |
|---|---|
| SMO | Directly defines the control value for the PILine® drive electronics in the C-867. |

#### Stopping motion

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

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

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

HLT also triggers an abrupt stop when trajectories are being followed

## 3.8.9 Generation of the Dynamics Profile

The profile generator of the C-867 is used for point-to-point motion in closed-loop operation. The profile generator does calculations to specify the target position, velocity, and acceleration of the axis for each point in time during motion (dynamics profile). The values calculated are called commanded values.

### INFORMATION

The profile generator does **not** calculate a dynamics profile when following trajectories (p. 95).

The dynamics profile generated by the profile generator of the C-867 depends on the motion variables that are specified by commands, parameters (p. 282) and/or HID.

| Motion variable | Com-mands | Parameters | Remarks |
|---|---|---|---|
| Acceleration (A) | ACC (p. 155)<br>ACC? (p. 155) | 0xB<br>Closed-Loop Acceleration (Phys. Unit/$s^2$)<br>Change with the | Limited by parameter 0x4A (maximum acceleration in closed-loop operation).<br>The maximum acceleration during HID control is specified by parameter 0x75. |

| Motion variable | Com-mands | Parameters | Remarks |
|---|---|---|---|
| | | ACC command or with SPA (p. 226) / SEP (p. 222); query with ACC? | |
| Deceleration (D) | DEC (p. 165)<br>DEC? (p. 166) | 0xC<br>Closed-Loop Deceleration (Phys. Unit/s²)<br>Change with the DEC command or with SPA / SEP; query with DEC? | Limited by parameter 0x4B (maximum deceleration in closed-loop operation).<br>The maximum deceleration during HID control is specified by parameter 0x76. |
| Velocity (V) | VEL (p. 246)<br>VEL? (p. 247) | 0x49<br>Closed-Loop Velocity (Phys. Unit/s)<br>Change with the VEL command or with SPA / SEP; query with VEL? | Limited by parameter 0xA (maximum velocity in closed-loop operation).<br>The maximum velocity during HID control is specified by parameter 0x74. Refer to "Controlling with an HID" (p. 114) and the description of the HIA command (p. 185) for further information. |
| Target position at the end of the movement | MOV (p. 213)<br>MVR (p. 215)<br>GOH (p. 181)<br>STE (p. 230) | - | The target position can be directly specified by HID control. The soft limits are set as the respective target position during HID control of the velocity. Refer to "Controlling with an HID" (p. 114) for further information.<br>The C-867 sets the target position to the current position of the axis in the following cases:<br>▪ Disabling HID control for the axis<br>▪ Switching servo mode on with the SVO (p. 232) command<br>▪ Stopping the motion with the #24 (p. 154), STP (p. 231), or HLT (p. 197) commands |

The profile generator of the C-867 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.

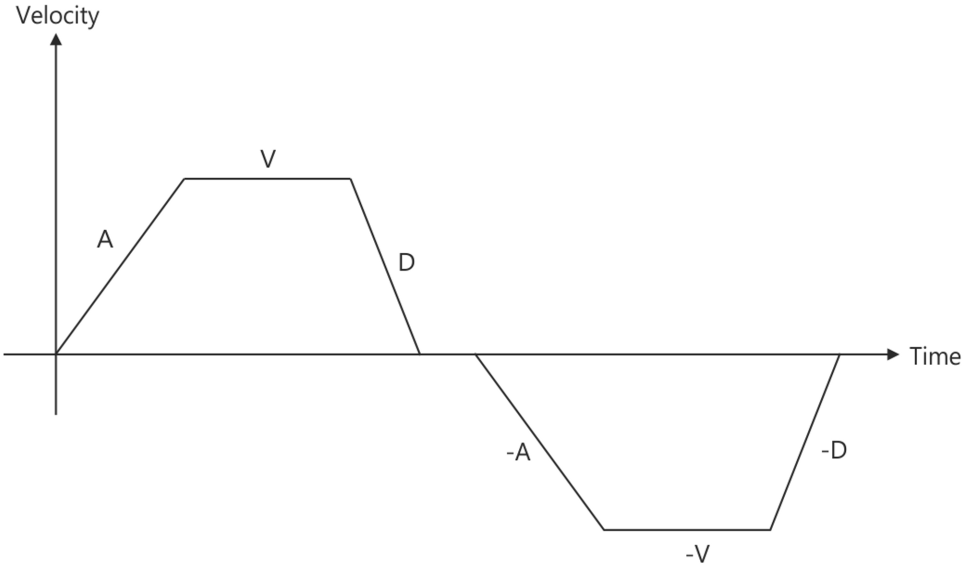



Figure 3: Basic trapezoidal velocity profile; A = acceleration, D = deceleration, V = velocity

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.

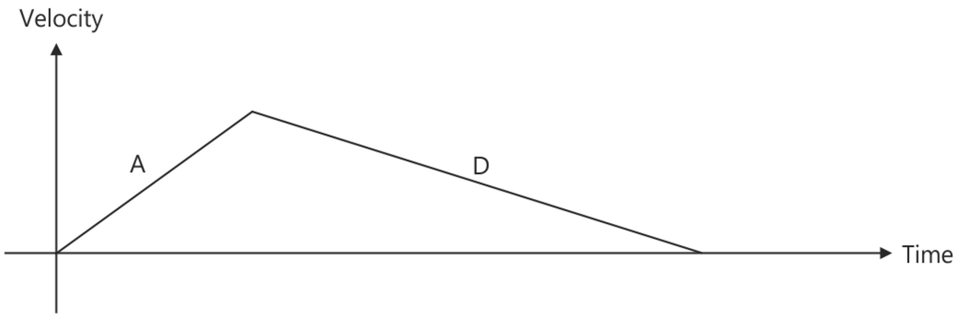



Figure 4: Basic trapezoidal velocity profile; A = acceleration, D = deceleration, no constant velocity

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.

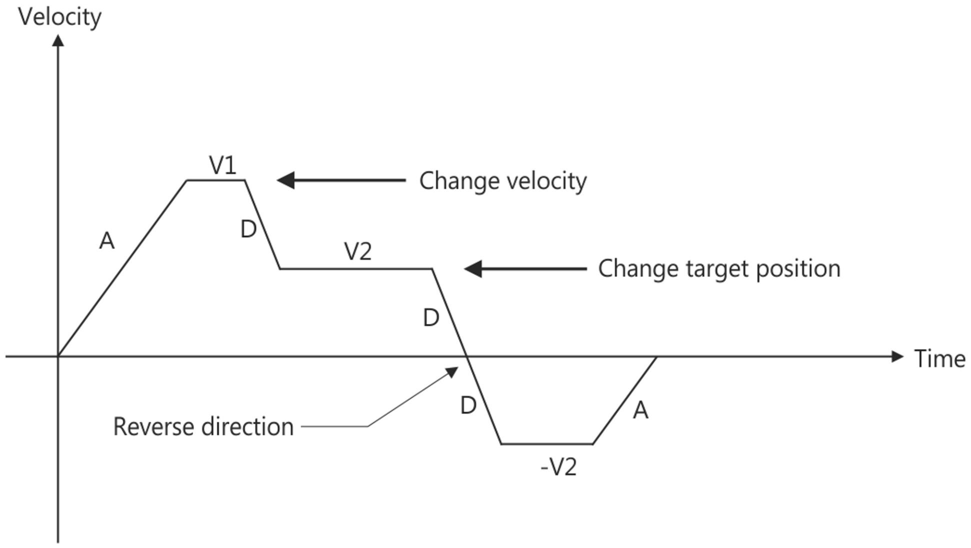



Figure 5: Complex trapezoidal profile with parameter changes; A = acceleration; D = deceleration; V1, V2, -V2 = velocities

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.

## 3.8.10 Servo Algorithm and Other Control Value Corrections

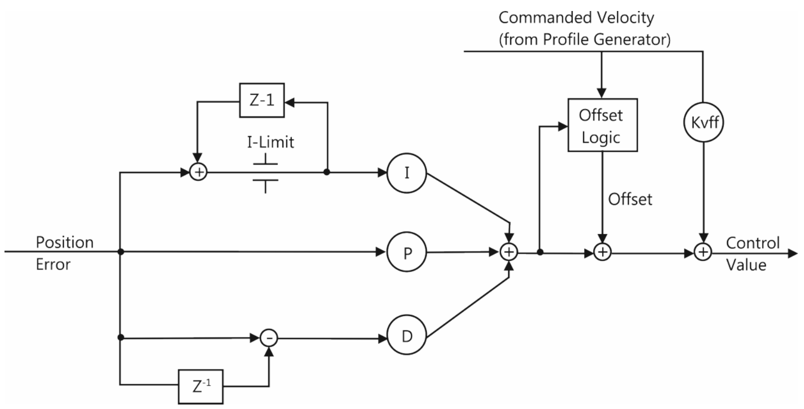



Figure 6: PID algorithm, offset compensation, and feed-forward control of the velocity (KVff)

In closed-loop operation, the control value for the control voltage of the PILine® driver electronics integrated in the C-867 is optimized along with the settling behavior of the system via the following corrections:

- Servo algorithm: The position error that results from the difference between the commanded position (from the dynamics profile (p. 27) or trajectory (p. 95)) and the actual position (sensor feedback) runs through a PID servo algorithm (**p**roportional **i**ntegral **d**erivative).
- Corrections of the control value: The dynamics profile or the trajectory can be subjected to an offset correction and a feed-forward control of the velocity.

For finer corrections, the C-867 switches between parameter groups 0 to 4 during the axis motion in closed-loop operation. The switching is done on the basis of configurable position windows.

Parameter groups 0 to 4 each contain the following settings:

- P, I, D terms and I limit for the servo algorithm
- Kvff term for feedforward control of the velocity (is only evaluated for the dynamics profile)
- Window limits for entry and exit

### Servo algorithm

The servo algorithm uses the following servo control parameters. The optimum servo control parameter setting depends on your application and your requirements; see "Optimizing Servo Control Parameters" (p. 88).

| Parameters | Description and Possible Values |
|---|---|
| ***D Term Delay (No. Of Servo Cycles)***<br>0x71 | D term delay<br>The D term can be calculated as a floating average over several servo cycles. The parameter specifies how many values (i.e., servo cycles) are to be used for averaging. |
| ***P term 0*** 0x401<br>***P term 1*** 0x411<br>***P term 2*** 0x421<br>***P term 3*** 0x431<br>***P term 4*** 0x441 | Proportional constants (dimensionless) of parameter groups 0 to 4<br>0 to 65535<br>Aim: Rapid correction of the position error |
| ***I term 0*** 0x402<br>***I term 1*** 0x412<br>***I term 2*** 0x422<br>***I term 3*** 0x432<br>***I term 4*** 0x442 | Integral constants (dimensionless) of parameter groups 0 to 4<br>0 to 65535<br>Aim: Reduction of the static position error |
| ***D term 0*** 0x403<br>***D term 1*** 0x413<br>***D term 2*** 0x423<br>***D term 3*** 0x433<br>***D term 4*** 0x443 | Differential constants (dimensionless) of parameter groups 0 to 4<br>0 to 65535<br>Aim: Damping of rapid control oscillation |

| Parameters | Description and Possible Values |
|---|---|
| ***I limit 0*** 0x404<br>***I limit 1*** 0x414<br>***I limit 2*** 0x424<br>***I limit 3*** 0x434<br>***I limit 4*** 0x444 | Limitation of the integral constants (dimensionless) of parameter groups 0 to 4<br>0 to 65535 |

## *INFORMATION*

To prevent a servo jitter of the axis after the target position is reached, the I term of the parameter group used for step-and-settle (group 0 by default) should be deactivated or minimized.

➢ Use the corresponding I limit to deactivate or minimize an I term. Example: I term 0 (0x402) is deactivated when I limit 0 (0x404) has the value zero (default setting)

The input of the servo algorithm can be configured for the C-867 with the following parameters:

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

### Corrections of the control value

The control value corrections for closed-loop operation can be configured via the parameters listed below:

| Parameters | Description and Possible Values |
|---|---|
| ***Motor Offset Positive***<br>0x33 | Offset for the positive direction of motion (dimensionless)<br>0 to 32767<br>The range of values corresponds to 0 to 10 V control voltage. Compensates the internal preload of the piezo motor. |
| ***Motor Offset Negative***<br>0x34 | Offset for the negative direction of motion (dimensionless)<br>0 to 32767<br>The range of values corresponds to 0 to -10 V control voltage. Compensates the internal preload of the piezo motor. |

| Parameters | Description and Possible Values |
|---|---|
| ***Motor Drive Offset***<br>0x48 | Velocity-dependent offset (dimensionless)<br>Is used if the commanded velocity does not equal zero (i.e., if the end of the motion has not been reached yet).<br>0 to 32767<br>The range of values corresponds to 0 to 10 V control voltage. Depending on the current direction of motion, the offset value has a positive or negative sign. |
| ***Kvff 0*** 0x405<br>***Kvff 1*** 0x415<br>***Kvff 2*** 0x425<br>***Kvff 3*** 0x435<br>***Kvff 4*** 0x445 | Feed-forward control of the commanded velocity for parameter groups 0 to 4<br>0 to 65535<br>Aim: Minimization of the position error |

***INFORMATION***

To start motion, PILine® ultrasonic piezo motors require a particular piezo voltage that is not equal to zero. For this reason, offset values (parameters 0x33, 0x34, 0x48) are added to the control value and therefore to the control voltage. The offset values for the positive and negative direction of motion (0x33 and 0x34) are to be kept as low as possible with a velocity-dependent offset (0x48). The optimum offset values for the positive and negative direction of motion can strongly deviate from each other especially in the case of a vertically aligned motion axis.

### Switching between parameter groups 0 to 4

Switching between parameter groups 0 to 4 for servo algorithm and feed-forward control of the velocity can be configured with the parameters listed in the following.

| Parameters | Description and Possible Values |
|---|---|
| ***Servo Window Mode***<br>0x4D | Reference variable for the position windows<br>0 = Target position<br>1 = Commanded position (default setting)<br>This parameter specifies the reference variable for the position windows that are used to switch between parameter groups 0 to 4 for servo algorithm and feed-forward control. The switching is done based on the difference between the current position and the selected reference variable.<br>When trajectories are executed, the 0 setting is implemented as follows: The target position is the position that would be reached next if the controller were to keep its current velocity. |
| ***Window 0 Delay (s)***<br>0x62 | Delay time for activating parameter group 0<br>0 to 1.000 s<br>The delay time starts with the entry of the current position in the entrance window of parameter group 0. The settings of parameter group 0 for P, I, and D term, I limit, and feed-forward control of the velocity are only activated after the end of the delay time (when the |

| Parameters | Description and Possible Values |
|---|---|
| | current position is still in the exit window of parameter group 0). |
| ***Number Of Servo Parameter Groups***<br>0x400 | Maximum number of parameter groups used<br>1 to 5<br>This parameter specifies the maximum number of parameter groups, between which, axis motion is switched. |
| ***Window Enter 0***<br>0x406<br>***Window Enter 1***<br>0x416<br>***Window Enter 2***<br>0x426<br>***Window Enter 3***<br>0x436<br>***Window Enter 4***<br>0x446 | Position windows for activating parameter groups 0 to 4<br>0 to $2^{31}$ counts of the encoder<br>The parameters specify the entrance windows for the parameter groups. The windows are centered around the reference variable selected with the parameter 0x4D. When the current position enters the entrance window of a parameter group, this parameter group is activated.<br>The parameter values each correspond to half of the window width. They can only be changed if the servo mode is switched off. |
| ***Window Exit 0***<br>0x407<br>***Window Exit 1***<br>0x417<br>***Window Exit 2***<br>0x427<br>***Window Exit 3***<br>0x437<br>***Window Exit 4***<br>0x447 | Position window for deactivating parameter groups 0 to 4<br>0 to $2^{31}$ counts of the encoder<br>The parameters specify the exit windows for the parameter groups. The windows are centered around the reference variable selected with the parameter 0x4D. When the current position leaves the exit window of a parameter group, this parameter group is deactivated, and the next-highest parameter group is activated.<br>The parameter values each correspond to half of the window width. They can only be changed if the servo mode is switched off. |

### *INFORMATION*

Parameter groups 0 to 4 are used as follows:

- Optimization of the motion using parameter groups 1 to 4, depending on the setting of the ***Number Of Servo Parameter Groups*** parameter (0x400)
- Optimization of the settling behavior of the system at the end of the motion by:
  – Parameter group 0, when parameter 0x400 has a value 2 to 5
  – Parameter group 1, when parameter 0x400 has the value 1 (the entrance and exit windows of parameter group 0 are still used as settling windows for determining the on-target state).

### *INFORMATION*

The following applies to the position windows:

- The entrance window for parameter group n must be smaller than the entrance window for parameter group n+1.
- The exit window for parameter group n must be smaller than the exit window for parameter group n+1.
- The position windows of the "outermost" parameter group used are ignored. Which

parameter group is the outermost group used depends on the setting of the parameter ***Number Of Servo Parameter Groups*** (0x400). Example: Parameter 0x400 has the value 3. The switching is then done between parameter groups 0, 1, and 2. Parameter group 2 is the outermost parameter group used. Because the position windows of parameter group 2 are ignored, it remains activated even when the current position is outside of its exit window (0x427).

- The entrance and exit windows for parameter group 0 are also used as settling windows for determining the on-target state (p. 44).
- For a stable switching behavior, the exit window of a parameter group should be larger than its entrance window.

The following two figures show the switching between parameter groups during axis motion. Settings in the examples:

- Reference variable of the switching: Target position (upper figure) or commanded position (lower figure)
- The entrance windows of the parameter groups are smaller than their exit windows.
- The maximum number of parameter groups used is 3 (0 to 2).
- When the current position enters the entrance window for parameter group 0, parameter group 0 is activated without delay (parameter 0x62 has the value 0).

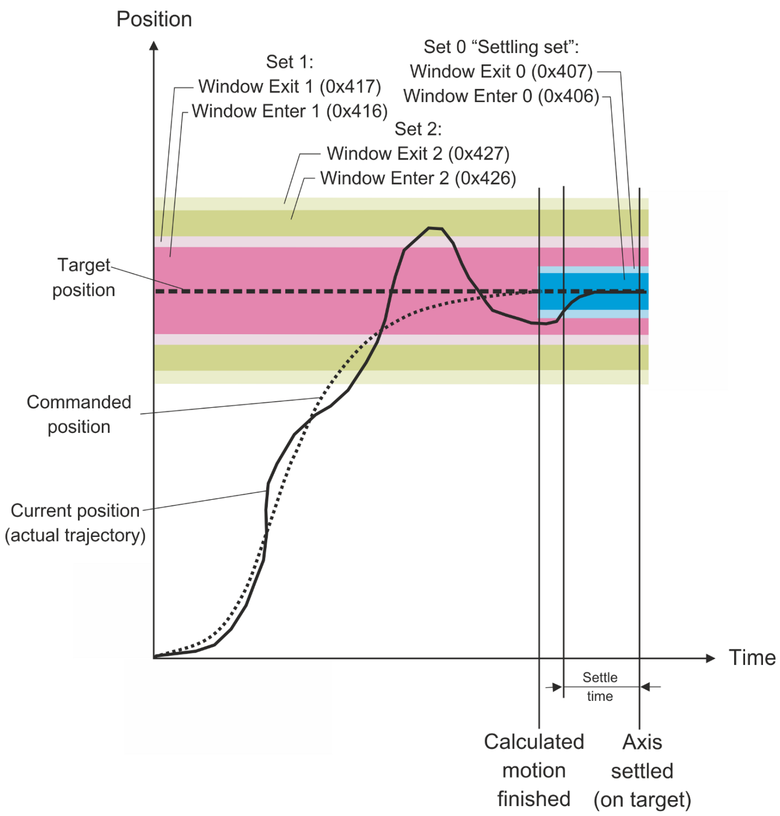



Figure 7: Switching between parameter groups 0 to 2 based on the difference between the current position and the target position

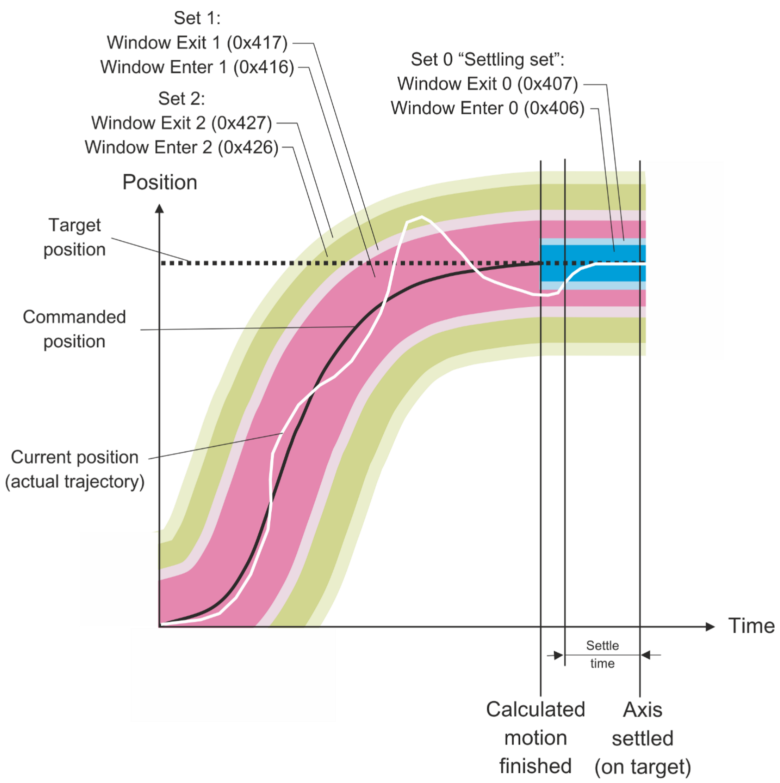



Figure 8: Switching between parameter groups 0 to 2 based on the difference between the current position and the commanded position

## 3.8.11 Adaptive Control: Adapted PID control

The C-867 with the firmware feature "Adaptive Control" offers an adapted PID servo algorithm for motion control in large dynamic ranges (from very fast to very slow). This control makes adjustments for different phases (states) of axis motion. To adjust, a separate parameter group is activated for each motion state.

The "Adaptive Control" firmware feature must be activated in order to use it, refer to Activating Additional Features (p. 85). The parameters for setting the adapted PID servo control are only visible after activating the feature in the controller.

### Operating principle

Axis motion can assume the following states:

| State | Description |
|---|---|
| Motion | **Moving the axis to the target position**<br>This state is active during trajectory travel. The velocity is checked afterwards. The *End Position* state is set for axis motion when the current velocity reaches the value of the ***Velocity to detect end of Motion (Phys. Unit/s) (Motion)*** parameter (0x47A).<br>A velocity-dependent adaptation of the control can be set for this state. This is done via the ***Velocity adaptive PID-Terms?*** parameter (0x476). |

| State | Description |
|---|---|
| End Position | **Final axis motion for reaching the target position**<br>This state is active until the on-target state is true. The axis motion is then set to the *Target* state.<br>The on-target state is set to true as soon as<br>▪ the axis has reached the entrance window around the target position (***Window Enter 0 (Target)*** parameter (0x455))<br>▪ and the delay time for setting the on-target state has expired (***Settling Time (s)*** parameter (0x3F)).<br>The delay time is reset when the axis leaves the entrance window around the target position (***Window Exit 0 (Target)*** parameter (0x456)). |
| Target | **Holding the axis at the target position**<br>This state is active as long as the current axis position is inside the exit window around the target position (***Window Exit 0 (Target)*** parameter (0x456)). The *End Position* state is reset in order to move the axis to the target position again as soon as the current position leaves this window. |
| Global Stable | **Avoiding oscillation in the *End Position state***<br>This state is set as soon as the velocity exceeds the value defined in the ***Velocity to detect vibration (Phys. Unit/s) (End Position)*** parameter (0x485). If the velocity drops below half this value, the previously active *End Position* state is set again. |

### Configuring the motion states

Parameter groups for configuring the states:

- Parameter group 2 (Motion)
- Parameter group 3 (End Position)
- Parameter group 0 (Target)
- Parameter group 1 (Global Stable)

| Parameters | Description and Possible Values |
|---|---|
| ***P-Term 2 (Motion)***<br>0x471 | Proportional constant (dimensionless) for *Motion* state of axis motion<br>The value defined her is used as minimum for the P term when the ***Velocity adaptive PID-Terms?*** parameter (0x476) for velocity-dependent adaptation of the PID values is activated.<br>0 to 65535<br>Objective: Fast correction of the position error |
| ***I-Term 2 (Motion)***<br>0x472 | Integral constant (dimensionless) for *Motion* state of axis motion<br>The value defined her is used as minimum for the I term when the ***Velocity adaptive PID-Terms?*** parameter (0x476) for velocity-dependent adaptation of the PID values is activated. |

| Parameters | Description and Possible Values |
|---|---|
| | 0 to 65535<br>Objective: Reduction of static position error |
| ***D-Term 2 (Motion)***<br>0x473 | Differential constant (dimensionless) for *Motion* state of axis motion<br>The value defined her is used as minimum for the D term when the ***Velocity adaptive PID-Terms?*** parameter (0x476) for velocity-dependent adaptation of the PID values is activated.<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***I-Limit 2 (Motion)***<br>0x474 | Limitation of the integral constant (dimensionless) for *Motion* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***KVFF 2 (Motion)***<br>0x475 | Feed-forward control of the commanded velocity in the *Motion* state of axis motion<br>0 to 65535<br>Objective: Minimization of the position error |
| ***Velocity adaptive PID-Terms?***<br>0x476 | Adapt the PID values for the *Motion* state depending on velocity?<br>0: Do not adapt PID values<br>1: Adapt PID values<br>The velocity range where adjusting is done is defined via parameters 0x478 and 0x479.<br>Decreasing speed within the range leads to an increase of the PID values. The values of parameters 0x471, 0x472, and 0x473 are used as minimum values for the P, I, and D terms. The maximum values for the P, I, and D terms are specified via parameters 0x477, 0x47B, and 0x47C. The target velocity defined in parameter 0x49 is decisive for the adjustment. |
| ***P-Term Max at min. Velocity (Motion)***<br>0x477 | P term for the *Motion* state at minimum velocity |
| ***Max. Velocity for adaptive PID-Term (Motion)***<br>0x478<br>***Min. Velocity for adaptive PID-Term (Motion)***<br>0x479 | Maximum and minimum value of the velocity for adapting the PID values in the *Motion* state<br>The PID values are adjusted according to the velocity only if the current velocity is between the defined values. If the current velocity is outside of this range, the minimum values (for high velocities) or the maximum values (for low velocities) are used for the P, I, and D terms.<br>PID values can only be adjusted if parameter 0x476 is also set to 1. |
| ***Velocity to detect end of Motion (Phys. Unit/s) (Motion)***<br>0x47A | Threshold value of the velocity for switching to the *End Position* state |

| Parameters | Description and Possible Values |
|---|---|
| ***Max I-Term at min. Velocity (Motion)***<br>0x47B | I term for the *Motion* state at minimum velocity |
| ***Max D-Term at min. Velocity (Motion)***<br>0x47C | D term for the *Motion* state at minimum velocity |

| Parameters | Description and Possible Values |
|---|---|
| ***P-Term 3 (End Position)***<br>0x481 | Proportional constant (dimensionless) for *End Position* state of axis motion<br>0 to 65535<br>Objective: Fast correction of the position error |
| ***I-Term 3 (End Position)***<br>0x482 | Integral constant (dimensionless) for *End Position* state of axis motion<br>0 to 65535<br>Objective: Reduction of static position error |
| ***D-Term 3 (End Position)***<br>0x483 | Differential constant (dimensionless) for *End Position* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***I-Limit 3 (End Position)***<br>0x484 | Limitation of the integral constant (dimensionless) for *End Position* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***Velocity to detect vibration (Phys. Unit/s) (End Position)***<br>0x485 | Threshold value of the velocity for switching to the *Global Stable* state<br>If the current velocity exceeds this value the *Global Stable* state is activated, which is intended to avoid oscillation.<br>If the velocity drops below half this value, the previously active *End Position* state is set again. |

| Parameters | Description and Possible Values |
|---|---|
| ***P-Term 0 (Target)***<br>0x451 | Proportional constant (dimensionless) for *Target* state of axis motion |
| ***I-Term 0 (Target)***<br>0x452 | Integral constant (dimensionless) for *Target* state of axis motion<br>0 to 65535<br>Objective: Reduction of static position error |
| ***D-Term 0 (Target)***<br>0x453 | Differential constant (dimensionless) for *Target* state of axis motion<br>0 to 65535 |

| Parameters | Description and Possible Values |
| --- | --- |
| | Aim: Damping of rapid control oscillations |
| ***I-Limit 0 (Target)***<br>0x454 | Limitation of the integral constant (dimensionless) for *Target* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***Window enter 0 (Target)***<br>0x455 | Position window for activating the parameter group 0 (*Target*)<br>0 to $2^{31}$ encoder counts<br>The parameter group is activated when the axis has reached the entry window around the target position and the delay for setting the on-target state has expired (Parameter ***Settling Time (s)*** (0x3F)).<br>The parameter value corresponds to half the width of the window. It can be changed only if servo mode is switched off. |
| ***Window exit 0 (Target)***<br>0x456 | Position window for deactivating parameter group 0 (*Target*)<br>0 to $2^{31}$ encoder counts<br>When the current position leaves the exit window around the target position, the parameter group is deactivated and parameter group 3 (*End Position*) is reactivated.<br>The parameter value corresponds to half the width of the window. It can be changed only if servo mode is switched off. |

| Parameters | Description and Possible Values |
| --- | --- |
| ***P-Term 1 (Global Stable)***<br>0x461 | Proportional constant (dimensionless) for *Global Stable* state of axis motion<br>0 to 65535<br>Objective: Fast correction of the position error |
| ***I-Term 1 (Global Stable)***<br>0x462 | Integral constant (dimensionless) for *Global Stable* state of axis motion<br>0 to 65535<br>Objective: Reduction of static position error |
| ***D-Term 1 (Global Stable)***<br>0x463 | Differential constant (dimensionless) for *Global Stable* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |
| ***I-Limit 1 (Global Stable)***<br>0x464 | Limitation of the integral constant (dimensionless) for *Global Stable* state of axis motion<br>0 to 65535<br>Aim: Damping of rapid control oscillations |

### Further Settings for Adaptive Control

The following parameters are used for further settings:

| Parameters | Description and Possible Values |
|---|---|
| ***Motor Output - Phase's Magnitude Dependencies***<br>0x35 | Behavior of both motor phases:<br>0 = constant<br>Amplitude of the second phase is constant<br>1 = correlation (default setting)<br>The amplitude of the first phase depends on the second phase<br><br>Additionally with activated "Adaptive Control" feature<br>2 = adaptive: Amplitude of the second phase depends on the velocity<br>If the current velocity is within a defined range, the amplitude of the second phase is adapted according to the velocity.<br>The velocity range for adapting is defined via the ***Max. Velocity for adaptive PID-Term (Motion)*** parameter (0x478) and ***Min. Velocity for adaptive PID-Term (Motion)*** parameter (0x479)<br>Decreasing speed leads to an increase of the amplitude. The value of the ***Motor Output Two-Phase Magnitude*** parameter (0x6F) is used as minimal value for the amplitude. The maximum value is specified via the ***Phase Two Motor Output - Max. Magnitude at min. Velocity*** parameter (0x47D). |
| ***Phase Two Motor Output - Max. Magnitude at min. Velocity***<br>0x47D | Maximum amplitude of the second motor phase<br>0 to 32767<br>Maximum value for the velocity-dependent adaptation of the second phase (0x35 = 2) |

### Switching the control

Switching between the servo algorithms is done via the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Control Algorithm Mode***<br>0x1000 | Type of servo algorithm<br>0 = Position-dependent PID control<br>1 = Adapted PID control (Adaptive Control) |

## 3.8.12 Electronic Camming

The axes of the C-867 can be coupled via the firmware (electronic camming).

If electronic camming is activated, the 2nd axis moves automatically when the 1st axis is moved.

To use the "Camming" firmware feature, it must be activated (see Activating Additional Features (p. 85)). The parameters for configuring the camming are only visible after activating the feature in the controller.

### Configuring camming

The transmission ratio between axis 1 and axis 2 is defined by the following polynomial:

$y = c_0 \times x^0 + c_1 \times x^1 + c_2 \times x^2 + c_3 \times x^3 + c_4 \times x^4 + c_5 \times x^5 + c_6 \times x^6$

x = target position axis 1 [counter]

y = target position axis 2 [counter]

The coefficients of the polynomial can be adapted via parameters. The parameters are only evaluated if the "Camming" feature is activated (parameter 0xE002722: ***Enable Camming Feature*** = 1).

| Parameter | Description and possible values |
|---|---|
| ***Electronic Camming Mode***<br>0x4F | Couple axes?<br>0: Do not couple axes<br>1: Couple axes<br>This parameter can be used to switch off the camming without completely deactivating the feature in the controller. |
| ***Camming Polynomial - Coefficient 0***<br>0x6010B60 | Value for coefficient 0 for adapting the camming<br>Coefficient 0: $c_0 \times x^0$ |
| ***Camming Polynomial - Coefficient 1***<br>0x6010B61 | Value for coefficient 1 for adapting the camming<br>Coefficient 1: $c_1 \times x^1$ |
| ***Camming Polynomial - Coefficient 2***<br>0x6010B62 | Value for coefficient 2 for adapting the camming<br>Coefficient 2: $c_2 \times x^2$ |
| ***Camming Polynomial - Coefficient 3***<br>0x6010B63 | Value for coefficient 3 for adapting the camming<br>Coefficient 3: $c_3 \times x^3$ |
| ***Camming Polynomial - Coefficient 4***<br>0x6010B64 | Value for coefficient 4 for adapting the camming<br>Coefficient 4: $c_4 \times x^4$ |
| ***Camming Polynomial - Coefficient 5***<br>0x6010B65 | Value for coefficient 5 for adapting the camming<br>Coefficient 5: $c_5 \times x^5$ |
| ***Camming Polynomial -*** | Value for coefficient 6 for adapting the camming |

| Parameter | Description and possible values |
|---|---|
| ***Coefficient 6***<br>0x6010B66 | Coefficient 6: $c_6 \times x^6$ |

### 3.8.13 Optional Two-Phase Control

A PILine® piezo motor has a separate piezo segment for the positive and the negative direction of motion. Depending on the direction of motion, only the corresponding segment is normally driven by the piezo voltage. The two-phase control allows the second segment (2nd phase) to be driven parallel to the first segment (1st phase), in order to diminish the feed motion of the piezo motor. The two-phase control makes very gentle and slow axis motion possible.

The default two-phase control of the C-867 is only performed via the piezo voltage amplitude of the 2nd phase of the motor (the assignment of the piezo segments as the 1st and 2nd phase changes according to the current direction of motion). The offset of the phases cannot be changed; it is always 180.

The default two-phase control of the C-867 is configured via the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Motor Output - Phase's Magnitude Dependencies***<br>0x35 | Behavior of both motor phases:<br>0 = constant<br>Amplitude of the second phase is constant<br>The value set by the ***Motor Output Two-Phase Magnitude*** parameter (0x6F) is used constantly.<br>1 = correlation (default setting)<br>The amplitude of the first phase depends on the second phase<br>The ***Motor Output Two-Phase Increment*** (0x6A) and ***Motor Output Two-Phase Decrement*** parameter (0x6C) are used.<br>2 = adaptive<br>Amplitude of the second phase depends on the velocity<br>Selection only possible when the "Adaptive Control" feature has been activated, refer to: Adaptive Control: Adapted PID Control (p. 36) |
| ***Motor Output Two-Phase Increment***<br>0x6A | Increase in the amplitude of the second phase per servo cycle<br>The amplitude of the second phase of the motor is increased by this value each servo cycle until the maximum value set by 0x6F is reached.<br>This parameter allows a gentle increase in the amplitude of the second phase. This prevents jerky and therefore uncontrolled axis motion.<br>0 to 32767<br>The preset value should not be changed. |
| ***Motor Output Always On***<br>0x6B | Maintain output piezo voltage?<br>0 = No, set piezo voltage to 0<br>The piezo voltage is set to 0 as soon as the axis has reached the |

| Parameters | Description and Possible Values |
|---|---|
| | target position.<br>1 = Yes, maintain piezo voltage<br>The piezo voltage ≠ 0 is maintained, even after the axis has reached the target position. It is only set to 0 when servo mode is switched off. |
| ***Motor Output Two-Phase Decrement***<br>0x6C | Decrease in the amplitude of the second phase per servo cycle<br>The amplitude of the second phase of the motor is decreased by this value each servo cycle.<br>This parameter allows a gentle decrease in the amplitude of the second phase. This prevents jerky and therefore uncontrolled axis motion.<br>0 to 32767<br>The preset value should not be changed. |
| ***Motor Output Phase 2 - Frequency Dependency***<br>0x6E | Frequency behavior of the second motor phase<br>0 = synchronous<br>Phase 2 is synchronized with phase 1, irrespective of the frequency set<br>1 = free run<br>Phase 2 runs independently from phase 1 with the set frequency<br>2 = follow<br>Phase 2 follows the frequency changes of the first phase. If the frequency of the first phase is changed by the value dx, the frequency of the second phase is also changed by the value dx.<br><br>Note: The values 1 and 2 take effect only when the C-867.L01 - "Slow Motion Drive" feature is activated, refer to Optional Accessories (p. 12). |
| ***Motor Output Two-Phase Magnitude***<br>0x6F | Amplitude of the second motor phase<br>The lower the velocity, the higher the amplitude of the second motor phase.<br>0 to 32767<br>Note that the current consumption and therefore the heat development as well is independent of the velocity. This means that the heat development is just as high at low velocities as at high velocities. |

### 3.8.14 On-Target State

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

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

The C-867 determines the on-target state on the basis of the following criteria:

- Settling window around the target position, is given by the entrance and exit windows for parameter group 0 (parameter 0x406 and 0x407)
- Delay time for setting the on-target state (parameter 0x3F)

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

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

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

In the *On Target* trigger mode (p. 104), the on-target state of the selected axis is output at the selected trigger output.

| Parameters | Description and Possible Values |
|---|---|
| ***Settling Time (s)***<br>0x3F | Delay time for setting the on-target state<br>0 to 1.000 s |
| ***Window Enter 0 (encoder counts)***<br>0x406<br>***Window Exit 0 (encoder counts)***<br>0x407 | Settling window around the target position<br>0 to $2^{31}$ counts of the encoder<br>The parameters give the window limits for entry and exit. If the current position exits the settling window, the target position is no longer considered as reached.<br>The parameter values each correspond to half of the window width. They can only be changed if the servo mode is switched off.<br>The limits of the settling window are also used to activate and deactivate parameter group 0 (p. 30). |

## 3.8.15 Reference Switch Detection

The C-867 receives reference switch signals of an axis at pin 13 of the panel plug **Axis 1** and **Axis 2** respectively (motor connector D-sub 15 (f) (p. 314)).

The following parameters can be used to configure how the C-867 detects the reference switch:

| Parameters | Description and Possible Values |
|---|---|
| ***Invert Reference?***<br>0x31 | Should the reference signal be inverted?<br>0 = Reference signal not inverted<br>1 = Reference signal inverted<br>This parameter is used for inverting the reference signal whose source can be either the reference switch or a digital input which is used instead of the reference switch (p. 111). |
| ***Has Reference?***<br>0x14 | Does the positioner have a reference switch?<br>0 = Reference switch not installed<br>1 = Reference switch (signal input at the axis connections)<br>This parameter activates or deactivates reference moves to the installed reference switch. |

| Parameters | Description and Possible Values |
|---|---|
| ***Reference Signal Type***<br>0x70 | Reference signal type<br>0 = Direction-sensing reference switch. The signal level changes when passing the reference switch.<br>1 = Pulse signal with a pulse width of several nanoseconds (parameter 0x47 must be set correctly).<br>2 = Index pulse. The reference switch is approached via the negative limit of the travel range.<br>3 = Index pulse. The reference switch is approached via the positive limit of the travel range.<br>4 = No reference signal<br>5 = The reference signal is output at the negative limit switch.<br>6 = The reference signal is output at the positive limit switch. |

The signal from the reference switch of the positioner can be used for reference moves. After a reference move to the reference switch, the controller knows the absolute axis position; see "Reference Point Definition" (p. 50).

### 3.8.16 Limit Switch Detection

The C-867 receives limit switch signals of an axis at the following connections of the **Axis 1** and **Axis 2** panel plugs (motor connector D-sub 15 (f) (p. 314)):

- Pin 5: Positive limit switch
- Pin 12: Negative limit switch

Limit switch detection by the C-867 can be configured with the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Limit Mode***<br>0x18 | Signal logic of the limit switches<br>0 = pos-HI, neg-HI<br>1 = pos-LO, neg-HI<br>2 = pos-HI, neg-LO<br>3 = pos-LO, neg-LO<br>posHI/pos-LO - positive limit switch active high/active low<br>neg-HI/neg-LO - negative limit switch active high/active low |
| ***Has No Limit Switches?***<br>0x32 | Does the positioner have limit switches?<br>0 = Positioner has limit switches (signal inputs at the axis connections)<br>1 = Positioner does not have limit switches<br>This parameter activates or deactivates a stop of the motion at the limit switches installed. |
| ***Use Limit Switches Only For Reference Moves?***<br>0x77 | Should the limit switches only be used for reference moves?<br>0 = Use limit switches for stopping at the end of the travel range and for reference moves (default)<br>1 = Use limit switches only for reference moves<br>This parameter is intended for use with rotation stages.<br>This parameter is only evaluated when the parameter 0x32 has the value 0. |

The signals from the limit switches (also end-of-travel sensors) of a linear positioner are used to stop motion in front of the hard stop at both ends of the travel range. Because the set deceleration is not taken into account here, there is a risk at high velocities that the positioner will hit the hard stop anyway. To prevent this, soft limits (p. 47) can be set via parameters of the C-867.

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 to a limit switch; see "Referencing" (p. 50).

### 3.8.17 Travel Range and Soft Limits

The following parameters of the C-867 reflect the physical travel range of the positioner and define soft limits:

| Parameters | Description and Possible Values |
|---|---|
| ***Maximum Travel In Positive Direction (Phys. Unit)***<br>0x15 | Soft limit in positive direction (physical unit)<br>Based on the zero position. If this value is smaller than the position value for the positive limit switch (which results from the sum of the parameters 0x16 and 0x2F), the positive limit switch cannot be used for reference moves.<br>The value can be negative. |
| ***Value At Reference Position (Phys. Unit)***<br>0x16 | Position value at the reference switch (physical unit)<br>The current position is set to this value if the axis has performed a reference move to the reference switch.<br>The parameter value is used in addition for calculating the position values which are set after reference moves to the limit switches; this also applies when the mechanics does not have a reference switch. |
| ***Distance From Negative Limit To Reference Position (Phys. Unit)***<br>0x17 | Distance between reference switch and negative limit switch (physical unit)<br>If the axis has performed a reference move to the negative limit switch, the current position is set to the difference between the values of parameters 0x16 and 0x17. |
| ***Distance From Reference Position To Positive Limit (Phys. Unit)***<br>0x2F | Distance between reference switch and positive limit switch (physical unit)<br>If the axis has performed a reference move to the positive limit switch, the current position is set to the sum of the values of parameters 0x16 and 0x2F. |
| ***Maximum Travel In Negative Direction (Phys. Unit)***<br>0x30 | Soft limit in negative direction (physical unit)<br>Based on the zero position. If this value is larger than the position value for the negative limit switch (which results from the difference between the parameters 0x16 and 0x17), the negative limit switch cannot be used for reference moves.<br>The value can be negative. |
| ***Range Limit Min***<br>0x07000000 | Additional soft limit for the negative direction of motion (physical unit)<br>If the current position reaches this value in either closed-loop or open-loop operation, the control value is set to zero and the motion |

| Parameters | Description and Possible Values |
|---|---|
| | is stopped as a result. The axis can move again as soon as the value for the soft limit has been decreased. |
| ***Range Limit Max***<br>0x07000001 | Additional soft limit for the positive direction of motion (physical unit)<br>If the current position reaches this value in either closed-loop or open-loop operation, the control value is set to zero and the motion is stopped as a result. The axis can move again as soon as the value for the soft limit has been increased. |

### *INFORMATION*

The C-867 supports two parameter pairs for establishing soft limits. They are intended for different applications:

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

### Examples

The following examples refer to an axis of a positioner with incremental sensor, reference switch and limit switches.

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

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

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

### *INFORMATION*

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

**Example 1: Maximum travel range available**

After reference moves (p. 50), the current position is to have the following values:

- Move to the negative limit switch: Current position = 0
- Move to the reference switch: Current position = 8
- Move to the positive limit switch: Current position = 20

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

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

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

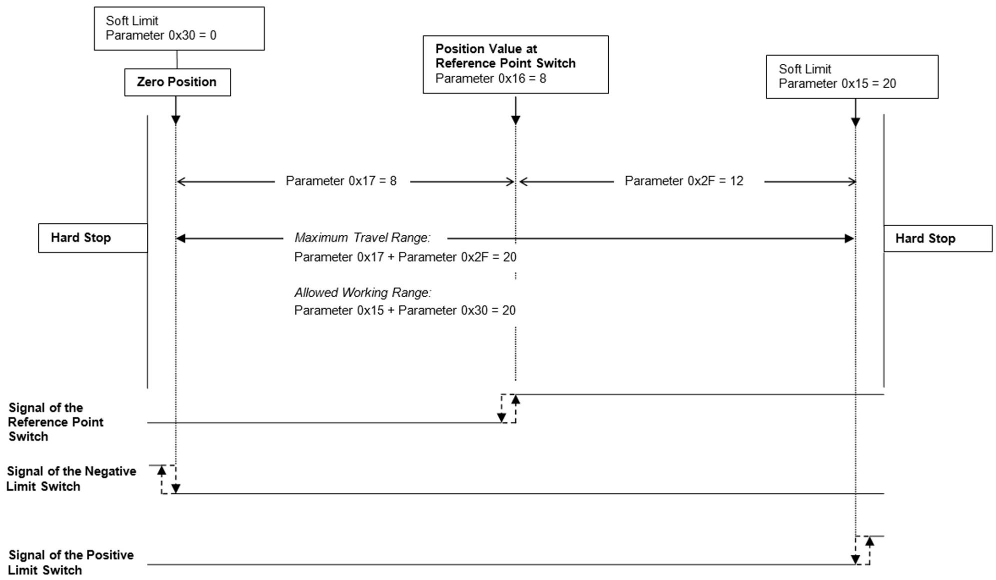



Figure 9: The travel range of the axis is not limited by soft limits.

After a reference move of the axis to the reference switch, query commands return the following responses:

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

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

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

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

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

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

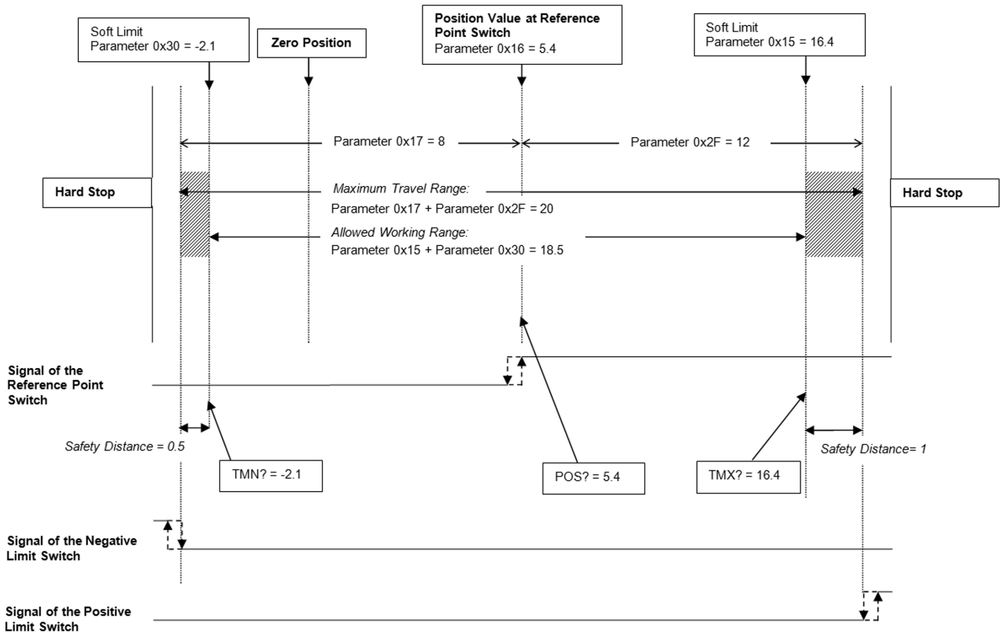



Figure 10: The travel range of the axis is limited by soft limits.

After a reference move of the axis to the reference switch, query commands return the following responses:

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

## 3.8.18 Referencing

***INFORMATION***

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.

The information on the signal type of the position sensor is loaded from the ID chip (p. 15) and given by the value of the ***Sensor Signal Type*** parameter (ID 0x3003320) (see "Parameter Overview" (p. 282)).

Incremental sensors only supply relative motion information. When the positioner is equipped with an incremental position sensor, the controller does not therefore know the absolute position of the axis during switch-on or reboot. Before absolute target positions can be commanded and reached, referencing must be done for the axis.

Referencing can be done in different ways:

- **Reference move** (default): A reference move moves the axis to a defined point, e.g., to the reference switch or to a limit switch. At this point, the current position is set to a defined value. The controller now knows the absolute axis position.
- **Setting the absolute position manually**: If this referencing method was activated by the `RON` command (p. 218), you can set the current position of the axis to an arbitrary value at an arbitrary point using the `POS` command (p. 217). The axis is not moved here. The controller knows the absolute axis position afterwards.

### *INFORMATION*

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

### *INFORMATION*

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

1. First move to the switch selected. The maximum velocity is specified via parameter 0x49 (***Closed-Loop Velocity (Phys. Unit/s)***, equivalent to setting with the VEL command).
2. Stop on reaching the switch edge. 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 switch selected. The maximum velocity is specified via parameter 0x50 (***Velocity For Reference Moves (Phys. Unit/s)***, specific velocity for reference moves only).
5. Stop when 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.

- Parameter 0x49 or 0x50
- Parameter 0x63 (***Distance Between Limit And Hard Stop (Phys. Unit)***)
- Parameter 0xC (***Closed-Loop Deceleration (Phys. Unit/s$^2$)***)

### Commands

The following commands are available for referencing:

| Com-mand | Syntax | Function |
|---|---|---|
| `RON` | RON {<AxisID> <ReferenceOn>} | Sets referencing method:<br>▪ <ReferenceOn> = 0: To define the reference point of the axis, an absolute position value can be assigned with POS or a reference move can be started with FRF.<br><ReferenceOn> = 1 (default): A reference move must be started with FRF to reference the axis. Using POS is not allowed. |
| `RON?` | RON? [{<AxisID>}] | Gets referencing method. |
| `FRF` | FRF [{<AxisID>}] | Starts a reference move to the reference switch. The approach depends on the value of the ***Reference Signal Type*** parameter (0x70):<br>▪ 0 or 1: The approach always takes place from the same side irrespective of the axis position when the command is sent.<br>▪ 2: The approach takes place via the negative limit switch.<br>▪ 3: The approach takes place via the positive limit switch.<br>▪ 4: No reference signal<br>▪ 5: The reference move is made to the negative limit switch; this is set as reference position.<br>▪ 6: The reference move is made to the positive limit switch; this is set as reference position. |
| `FRF?` | FRF? [{<AxisID>}] | Queries whether the reference point for an axis has already been defined.<br>1 = Reference point has been defined<br>0 = Reference point has not been defined |
| `POS` | POS {<AxisID> <Position>} | Sets the current position (does not trigger a motion) and therefore defines the reference point. |

### Parameters

Reference moves can be configured with the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Closed-Loop Deceleration (Phys. Unit/s2)***<br>0xC | Deceleration in closed-loop operation<br>For details, see "Generation of the Dynamics Profile (p. 27)". |
| ***Reference Travel Direction***<br>0x47 | Default direction for the reference move<br>0 = automatic detection<br>1 = negative direction<br>2 = positive direction |
| ***Closed-Loop Velocity (Phys. Unit/s)***<br>0x49 | Velocity in closed-loop operation<br>For details, see "Generation of the Dynamics Profile (p. 27)". |
| ***Velocity For Reference Moves (Phys. Unit/s)***<br>0x50 | Velocity for reference move<br>Specifies the maximum velocity during a reference move for the second approach of the switch selected. For high repeatability during referencing, the maximum of this value should be as large as the value of parameter 0x49. If the value of parameter 0x50 is set to 0, reference moves are not possible. |
| ***Distance Between Limit And Hard Stop (Phys. Unit)***<br>0x63 | Distance between the built-in limit switch and the hard stop<br>Determines the maximum stopping distance during reference moves. The actual velocities during a reference move are calculated on the basis of this value, the deceleration set (0xC) and the velocities set (0x49 and 0x50). |
| ***Distance From Limit To Start Of Ref. Search (Phys. Unit)***<br>0x78 | Distance between limit switch and the starting position for the motion to the index pulse<br>For details, see explanation below the table. |
| ***Distance For Reference Search (Phys. Unit)***<br>0x79 | Maximum distance for the motion to the index pulse<br>For details, see explanation below the table. |

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

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

Sequence of the reference move:

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

***INFORMATION***

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

***INFORMATION***

The limit switches can be used for reference moves only if the travel range is not limited by soft limits (p. 47).

***INFORMATION***

For reference moves, you can also use the digital inputs of the C-867 as the source of the reference signal, the negative limit switch signal or the positive limit switch signal. See "Using Digital Input Signals as Switch Signals" (p. 111) for more information.

***INFORMATION***

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

➢ Set the absolute position of the axis manually only if referencing is not otherwise possible.

***INFORMATION***

If the current parameter settings of the C-867 are written to the nonvolatile memory in PIMikroMove or by entering the WPA command using the password 100 or 101, the axis will no longer be considered "referenced" (the response to FRF? is 0).

# 4 Unpacking

1. Unpack the C-867 with care.
2. If the C-867 was delivered with protective caps on the connectors: Do **not** remove the 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. 309) immediately.
5. Keep all packaging materials in case the product needs to be returned.

# 5 Installing

## 5.1 General Notes on Installation

- Install the C-867 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Only use cables and connectors that meet local safety regulations.

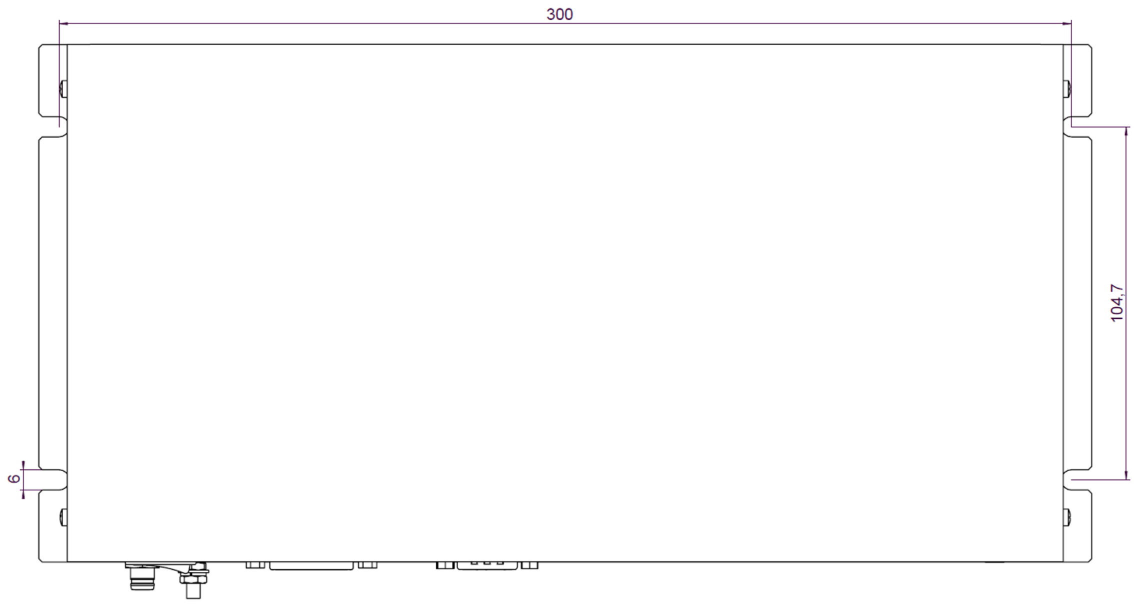
## 5.2 Ensuring Ventilation

High temperatures can overheat the C-867.

- Set up the C-867 with a distance of at least 10 cm to the top and rear panels and at least 5 cm to the sides. If this is not possible, make sure that the environment is cooled sufficiently.
- Ensure sufficient ventilation at the place of installation.
- Keep the ambient temperature to a noncritical level (<40 °C).

## 5.3 Mounting the C-867

The C-867 can be used as benchtop device or mounted in any orientation on a surface.




### Tools and accessories

- Suitable screws

- Suitable screwdriver

### Mounting the C-867

1. Make the necessary holes in the surface.

   The arrangement of the recesses in the mounting rails of the C-867 can be found in the figure.

2. Use two screws on each side to affix the C-867 to the recesses in the mounting rails.

## 5.4 Connecting the C-867 to the Protective Earth Conductor

***INFORMATION***

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

The C-867 is equipped with a threaded bolt (p. 9) to which the protective earth conductor is to be connected. For information on the placement and labeling of the protective earth connector on the C-867 see "Product View" (p. 9).

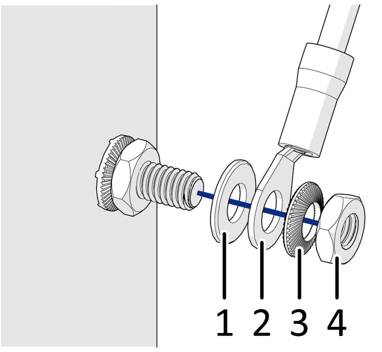



Figure 11: Connecting the protective earth conductor to the C-867

| | |
|---|---|
| 1 | Flat washer |
| 2 | Cable lug with protective earth conductor |
| 3 | Lock washer |
| 4 | Nut |

### Requirements

✓ The C-867 is switched off, i.e., the power adapter is **not** connected to the power socket with the power cord.

### Tools and accessories

- Suitable protective earth conductor:
  - Cable cross section ≥ 0.75 $mm^2$
  - Contact resistance < 0.1 ohm at 25 A at all connection points relevant for mounting the protective earth conductor

- Mounting hardware for the protective earth conductor; is on the protective earth connector on delivery of the C-867
- Suitable wrench

### Connecting the C-867 to the protective earth conductor

1. If necessary, attach a suitable cable lug to the protective earth conductor.
2. Remove the nut and lock washer from the protective earth connector of the C-867.
3. In the order shown above, push the cable lug of the protective earth conductor and the lock washer onto the threaded bolt.
4. Screw the nut onto the threaded bolt. In this way, the cable lug attached to the protective earth conductor is wedged between the flat washer and the lock washer.
5. Tighten the nut with at least three turns and a torque of 1.2 Nm to 1.5 Nm.

## 5.5 Connecting the Power Adapter to the C-867

### Requirements

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

### Tools and accessories

- 24-V wide input range power supply included (for line voltages between 100 and 240 V alternating current at 50 or 60 Hz)
  Alternative: Suitable power supply that supplies 24 V direct current and a maximum output current of at least 4.5 amperes
- Included power cord
  Alternatively: Sufficiently sized power cord

### Connecting the power adapter to the C-867

1. If necessary: Remove the protective cap from the C-867's voltage connector.
2. Connect the barrel connector on the power adapter to the 24 V connection (**24 V DC 4.5 A**) of the C-867.
3. Connect the power cord to the power adapter.

## 5.6 Connecting the Positioner

*NOTICE*

**Damage if a wrong motor is connected!**
Connecting a positioner with DC motor, stepper motor, or voice coil drive to the C-867 can cause irreparable damage to the positioner or controller.

➢ Only connect a positioner with PILine® ultrasonic piezo motors to the C-867.

***NOTICE***

**Unsuitable cables!**
Unsuitable cables can cause damage to the controller and can affect the performance of the positioner.

- ➢ Only use genuine PI parts to connect the positioner to the C-867.
- ➢ If you need longer cables, use extension cables from PI (p. 12).

**Requirements**

- ✓ The C-867 is switched off, i.e., the toggle switch on the front panel is in the **O** position.
- ✓ You have read and understood the user manual for the positioner(s).

**Tools and accessories**

- Positioner with PILine® ultrasonic piezo motor(s) and D-sub 15 (m) connector
- Optional: Suitable extension cable from PI

**Connect the positioner with the PILine® ultrasonic piezo motor**

1. Connect the positioner to either the **Axis 1** or **Axis 2** socket of the C-867.
2. Use the integrated screws to secure the connections against accidental disconnection.

## 5.7 Connecting an HID

### 5.7.1 Connecting an Analog HID

***INFORMATION***

A total of 6 axes of an analog HID can be connected to the **Analog Joystick**, **Analog In** (p. 317), and **I/O** (p. 316) sockets of the C-867.
The axes of the analog HID are suitable for controlling the following motion variables of the positioner axis connected to the C-867:

- Axes 1 to 4: Absolute target position, velocity, maximum velocity
- Axes 5 and 6: Relative target position

Connection options at the **Analog Joystick** socket:

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

Connection options at the **Analog In** socket:

- Axis 3: Tip (-10 to 10 V)
- Axis 4: Ring (-10 to 10 V)

Connection options at the **I/O** socket:

- Axis 5: Pins 1 and 2 (0 to +5 V)
- Axis 6: Pins 3 and 4 (0 to +5 V)

The two buttons of the HID can be connected to the **Analog Joystick** socket. Connection options:

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

#### Tools and Accessories

If the absolute target position, the velocity, or the maximum velocity of the axis of the C-867 is to be controlled with the HID:

- Analog joystick for operation with 0 to 3.3 V, available as an accessory (p. 12):
- Alternative: Analog signal source that supplies -10 to 10 V

If the relative target position of the C-867's axis is to be controlled with the HID:

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

#### Connecting an HID

➢ If you want to use axis 1 and/or 2 of the HID, connect an HID to the **Analog Joystick** socket of the C-867.

➢ If you want to use axis 3 and/or 4 of the HID, connect an analog signal source that supplies -10 to 10 V to the **Analog In** socket.

➢ If you want to use axis 5 and/or 6 of the HID, connect a suitable rotary encoder or pulse generator to the following pins of the **I/O** socket of the C-867:

  – For axis 5 of the human interface device: Pins 1 and 2
  – For axis 6 of the human interface device: Pins 3 and 4

### 5.7.2 Connecting a Digital HID

***INFORMATION***

Up to five digital human interface devices can be connected to the **Dig. Joystick** socket of the C-867.

The C-867 assigns identifiers to connected HIDs as follows:

- When an HID is connected directly to the C-867, it always receives the identifier 2.
- When human interface devices are connected to a USB hub connected to the C-867 when the C-867 is *switched off*, the C-867 assigns the identifiers of the human interface devices according to the numbering of the physical slots when it is switched on.
- When HIDs are connected to a USB hub connected to the C-867 when the C-867 is *switched on*, the C-867 assigns the identifiers in the order of connection. The identifiers of the HIDs can therefore change after the C-867 is rebooted.

➢ For a consistent assignment of the identifiers, always connect the HIDs in the same chronological order and in the same slots on the hub when the C-867 is switched on.

#### Tools and Accessories

- Suitable HID with type A USB connector such as: Joystick or gamepad
- If several HIDs are to be connected: Suitable USB hub

### Connecting an HID

- Connect a single HID directly to the USB connector (type A) **Dig. Joystick** on the C-867.

If you are using a USB hub to connect several HIDs:

- Connect the HIDs to the USB hub.
- Connect the USB hub to the USB connector (type A) **Dig. Joystick** on the C-867.

# 5.8 Connecting Digital Inputs and Outputs

The digital inputs and outputs on the **I/O** socket of the C-867 can be used as follows:

- Outputs: Triggering of external devices; see "Digital Output Signals" (p. 100).
- Inputs: Use in macros (p. 111) and/or as a source for the reference switch and limit switch signals of the axis (p. 111) and/or for HID control (p. 114)

## 5.8.1 Connecting the Digital Outputs

***INFORMATION***

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

***INFORMATION***

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

### Tools and accessories

- Suitable cable, e. g. C-170.IO IO cable with open end, available as an optional accessory (p. 12)
- Device to be triggered having digital input for TTL signals

### Connecting a device to be triggered

- Connect an appropriate device to one of pins 5, 6, 7, and 8 of the **I/O** socket of the C-867.

## 5.8.2 Connecting the Digital Inputs

***INFORMATION***

Digital input signals can be fed to the C-867 via pins 1, 2, 3, and 4 of the **I/O** socket.

**INFORMATION**

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

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

#### Tools and accessories

- Suitable signal source:
  - If the digital inputs are to be used in macros, it is possible for example, to connect the C-170.PB pushbutton box, which is available as an optional accessory (p. 12).
  - If the digital inputs are to be used as the source for the reference and limit switch signals of the axis, the signal level may only change once across the entire travel range.
- If necessary: Suitable cable, e. g. C-170.IO IO cable with exposed end, available as an optional accessory (p. 12).

#### Connecting a digital signal source

➢ If you want to use the digital inputs in macros or as switch signals: Connect a suitable signal source to one of the pins 1, 2, 3, or 4 of the **I/O** socket of the C-867.

➢ If you want to use the digital inputs for HID control, follow the instructions in "Connecting an HID" (p. 60).

## 5.9 Connecting Analog Signal Sources

The analog inputs on the **I/O** socket of the C-867 can be used as follows:

- Use in macros (p. 114): Details and examples of macros are found in "Controller Macros" (p. 126).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

**INFORMATION**

Analog input signals can be fed via pins 1, 2, 3, and 4 of the **I/O** socket into the C-867.

**INFORMATION**

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

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

#### Tools and accessories

- Suitable signal source

- If necessary: Suitable cable, e. g. C-170.IO IO cable with open end, available as an optional accessory (p. 12).

**Connecting an analog signal source**

➢ Connect an appropriate signal source to one of pins 1, 2, 3 or 4 of the **I/O** socket of the C-867.

## 5.10 Installing the PC Software

Communication between the C-867 and a PC is required to configure the C-867 and to command motion using the GCS commands. Various PC software applications are available for this purpose.

### 5.10.1 Doing Initial Installation

**Accessories**

- PC with Windows or Linux operating system and at least 30 MB free storage space
- Data storage device with PI Software Suite (included in the scope of delivery)

  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.

**Installing the PC software on Windows**

1. Start the installation wizard by double-clicking ***PISoftwareSuite.exe*** in the installation directory (root directory of the data storage device).

   The ***InstallShield Wizard*** window opens for installing the PI Software Suite.
2. Follow the instructions on the screen.

   The PI Software Suite includes the following components:
   - Drivers for use with NI LabVIEW software
   - Dynamic program library for GCS
   - PIMikroMove
   - PC software for updating the firmware of the C-867
   - PIUpdateFinder for updating the PI Software Suite
   - USB driver

**Installing the PC software on Linux**

1. Unpack the tar archive from the /Linux directory of the data storage device to a directory on your PC.
2. Open a terminal and go to the directory to which you have unpacked the tar archive.
3. Log in as a superuser (root privileges).

4. To start the installation, enter ./INSTALL
   Pay attention to capitalization while entering the command.
5. Follow the instructions on the screen.

You can select individual components for installation.

## 5.10.2 Installing Updates

PI is constantly improving the PI Software Suite.

➢ Always install the latest version of PI Software Suite and the positioner database.

### Requirements

✓ Active connection to the Internet

✓ If your PC uses a Windows operating system:

   – You have downloaded the PIUpdateFinder manual (A000T0028) from the PI website. The link is in the "A000T0081-Downloading Manuals from PI.pdf" file in the \Manuals folder on the data storage device with the PI Software Suite.

### Updating the PC software and PISTAGES3.DB in Windows

➢ Use the PIUpdateFinder:

   – Follow the instructions in the manual for the PIUpdateFinder (A000T0028).

### Updating the PC software on Linux

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

   The download link will be sent to the email address entered in the form.
6. Unpack the archive file on your PC to a separate installation directory.
7. In the directory with the unpacked files, go to the ***linux*** subdirectory.
8. Unpack the archive file in the ***linux*** directory by entering the command tar -xvpf <name of the archive file> on the console.
9. Log into the PC as superuser (root privileges).
10. Install the update.

### *INFORMATION*

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

➢ Contact our customer service department (p. 309).

### Updating PISTAGES3.DB in Linux

1. Contact the customer service department (p. 309) to get the latest version of the PISTAGES3.DB positioner database.
2. Log into the PC as superuser (root privileges).
3. Install the update that you received from our customer service department on your PC.

## 5.10.3 Installing Custom Positioner Databases

PI provides a data carrier with a custom positioner that has the following contents:

- Program Import PI CustomStage
- Custom positioner database with the parameter set for the positioner

In order for the parameter set to be selected in the PC software, it must first be inserted into the PIStages3 positioner database by the Import PI Custom Stage program.

➢ Install the custom positioner database by double-clicking the file ***Import_PI_CustomStage.exe*** in the root directory of the data carrier.

The parameter set from the custom positioner database is inserted into PIStages3.

If a message appears that installation of the custom positioner database failed:

a) Update the PIStages3 database on your PC, see "Installing Updates" (p. 65).

b) Repeat the installation of the custom positioner database.

# 5.11 Connecting the PC

The C-867 can be controlled from a PC with ASCII commands. Connecting to the PC can be done via a direct connection or via a daisy chain network. The following interfaces of the C-867 can be used for direct connection to the PC:

- RS-232 interface
- USB interface
- TCP/IP interface

***NOTICE***

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

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

In this section, you learn how to establish proper cable connections between the C-867 and a PC as well as in a TCP/IP network.

The steps for establishing communication between C-867 and PC are described in the section "Startup":

- Establishing communication via the TCP/IP interface (p. 73)
- Establishing communication via the USB interface (p. 72)
- Establishing communication via the RS-232 interface (p. 71)

Setting up a daisy chain network as well as establishing communication between the PC and a networked C-867 controller are also described in the "Startup" chapter:

- Setting up a daisy chain network (p. 76)

## 5.11.1 Connecting the C-867 via the RS-232 Interface

### Requirements

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

### Tools and accessories

- RS-232 null-modem cable (available as an accessory (p. 12): C-815.34)

### Connecting the C-867 to the PC

➢ Connect the **RS-232 In** socket on the front panel of the C-867 and the RS-232 interface of the PC via the null-modem cable.

## 5.11.2 Connecting the C-867 via the USB interface

### Requirements

✓ The PC has a free USB interface.

### Tools and accessories

- USB A/USB B cable (000011448 in scope of delivery)

### Connecting the C-867 to the PC

➢ Connect the USB socket on the front panel of the C-867 and the USB interface of the PC with the USB cable.

## 5.11.3 Connecting the C-867 via the TCP/IP Interface

### Requirements

✓ If the C-867 is to be directly connected to the PC:
The PC has a free RJ45 Ethernet connection socket.

- ✓ If the C-867 and a PC are to be operated together in a network:
  A free access point to the network is available for the C-867; a suitable hub or switch is connected to the network for this purpose if necessary.

### Tools and accessories

- If the C-867 is to be directly connected to the PC:
  Crossover network cable (available as an accessory (p. 12): C-815.563)
- If the C-867 is to be connected to a network access point:
  Straight-through network cable (available as an accessory: C-815.553)

### Connecting the C-867 directly to the PC

- ➢ Connect the RJ45 socket on the front panel of the C-867 to the RJ45 Ethernet connection socket of the PC via the crossover network cable.

### Connecting the C-867 to the network in which the PC is also located

- ➢ Connect the RJ45 socket on the front panel of the C-867 with the network access point via the straight-through network cable.

# 6 Startup

## 6.1 General Notes on Startup

***CAUTION***

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

If the protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the C-867 in the event of a malfunction or failure of the system. If there are touch voltages, touching the C-867 can result in minor injuries from electric shock.

- Connect the C-867 to a protective earth conductor (p. 58) before starting.
- Do **not** remove the protective earth conductor during operation.
- If the protective earth conductor has to be removed temporarily (e.g., in the case of modifications), reconnect the C-867 to the protective earth conductor before restarting.

***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 acceleration, can cause damage to or considerable wear on the mechanics.

- Avoid motion in open-loop operation.
- If motion in open-loop operation is necessary:
  - Set the control value with the SMO command so that the axis moves with low velocity.
  - Stop the axis 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 disable the evaluation of the limit switches by the C-867 via parameter setting.
- 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.

## 6.2 Switching the C-867 On

***INFORMATION***

The C-867 is intended for closed-loop operation with 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.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a startup macro that starts the C-867 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 134).

➢ Avoid motion in open-loop operation.

***INFORMATION***

The ID chip is not read when you connect the positioner while the C-867 is switched on.

➢ After connecting a positioner, reboot the C-867 with the `RBT` (p. 218) command or with the corresponding PC software functions in order to read the data from the ID chip.

**Requirements**

✓ You have read and understood the General Notes on Startup (p. 69).

✓ The C-867 has been installed properly (p. 57).

**Switching the C-867 on**

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

   The C-867 loads information to the volatile memory in the following order:

   a) Parameter values from the nonvolatile memory

   b) Parameter values from the ID chip of the positioner
3. Wait until the status LED lights up green.

   The **Status** LED on the front panel of the device indicates the status of the C-867:

   – green: C-867 is ready for normal operation

   – off: The C-867 is not connected to the power supply or could be defective

➢ If the C-867 is connected to the power adapter (p. 59) properly and the **Status** LED does not light up after switching on, contact our customer service department (p. 309).

## 6.3 Establishing Communication

The procedure for PIMikroMove is described in the following.

***INFORMATION***

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

***INFORMATION***

A non-networked controller must have the address 1, if it is to be used in PIMikroMove.

### 6.3.1 Establishing Communication via the RS-232 Interface

#### Requirements

✓ You have read and understood the General Notes on Startup (p. 69).

✓ The C-867 is connected to the RS-232 interface of the PC (p. 67).

✓ The C-867 is switched on (p. 69).

✓ The PC is switched on.

✓ The required software is installed on the PC (p. 64).

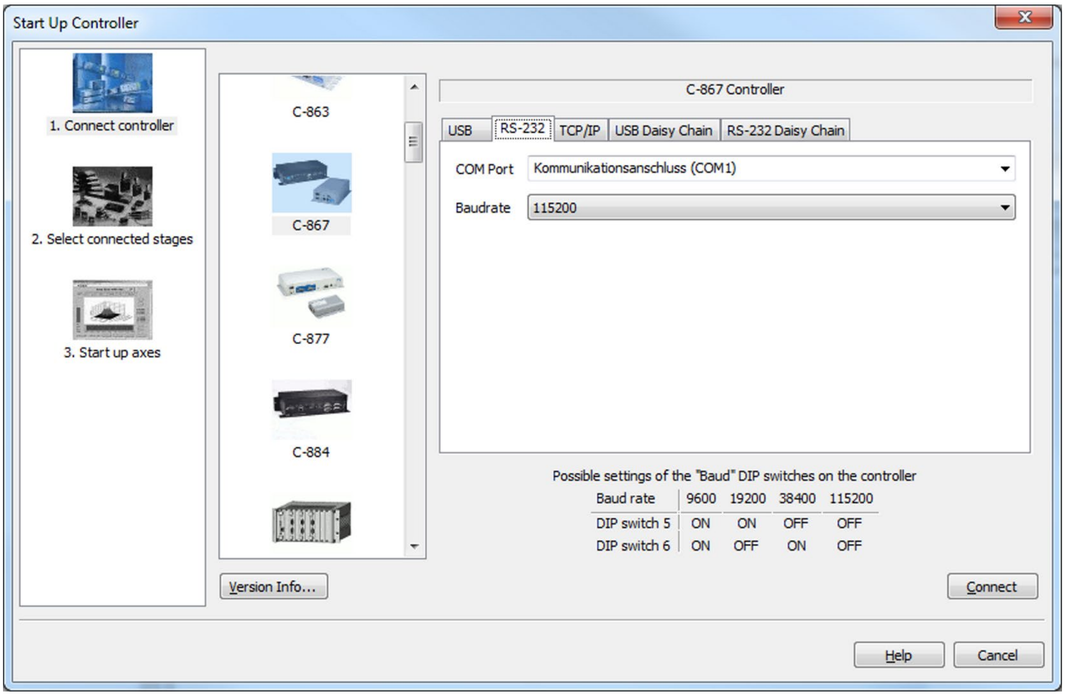
✓ You have read and understood the manual of the PC software used.The links to the software manuals are in the A000T0081 file on the PI software data storage medium.

#### Establishing communication via RS-232

1. Start PIMikroMove.

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

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




2. Select ***C-867*** in the field for controller selection.
3. Select the ***RS-232*** tab on the right-hand side of the window.
4. In the ***COM Port*** field, select the COM port of the PC to which you have connected the C-867.
5. In the ***Baudrate*** field, set the value that is set for the C-867.

   This adapts the baud rate of the PC to the baud rate of the C-867.
6. Click ***Connect*** to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the C-867 for the connected positioner; see "Starting Motion" (p. 80).

## 6.3.2 Establishing Communication via the USB Interface

***INFORMATION***

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

### Requirements

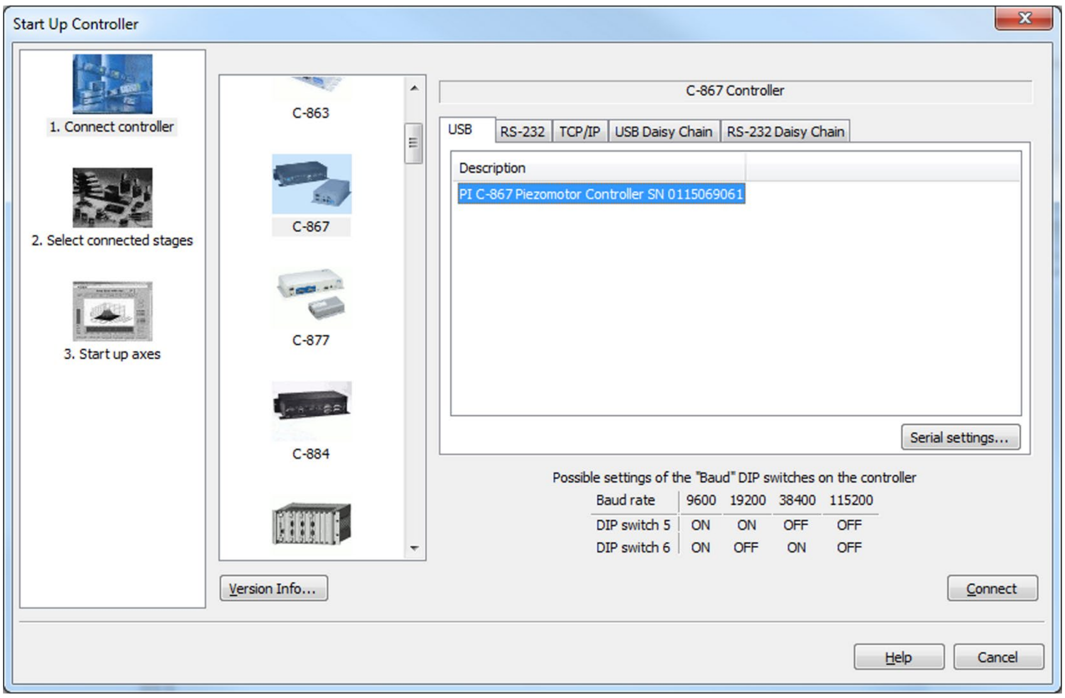
- ✓ You have read and understood the General Notes on Startup (p. 69).
- ✓ The C-867 is connected to the USB interface of the PC (p. 67).
- ✓ The C-867 is switched on (p. 69).
- ✓ The PC is switched on.
- ✓ The required software and USB drivers are installed on the PC (p. 64).
- ✓ You have read and understood the manual of the PC software used. The links to the software manuals are in the A000T0081 file on the PI software data storage medium.

### Establishing communication via USB

1. Start PIMikroMove.

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

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



2. Select ***C-867*** in the field for controller selection.
3. Select the ***USB*** tab on the right-hand side of the window.
4. On the ***USB*** tab, select the ***C-867*** connected.
5. Click ***Connect*** to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the C-867 for the connected positioner; see "Starting Motion" (p. 80).

➢ If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 303).

### 6.3.3 Establishing Communication via the TCP/IP Interface

Before communication is established, it can be necessary to adapt the interface parameters (p. 16) once, depending on the type of networking:

- **Network with DHCP server**: No adjustment of the factory settings of the C-867's interface parameters is required. You can begin setting up the communication.
- **Network without DHCP server or direct connection** of the C-867 to the PC's Ethernet socket: It is necessary to adapt the C-867's interface parameters. Make the necessary adaptations before establishing communication.

#### Requirements

- ✓ The C-867 is connected (p. 67) to the network or directly to the PC via the RJ45 Ethernet socket.
- ✓ If several C-867 are connected to the same network via their TCP/IP interfaces: You have the serial number of the C-867 ready, which you want to establish the communication with. The serial number can be found on the type plate of the C-867 (p. 11).
- ✓ The C-867 is switched on.
- ✓ The PC is switched on.
- ✓ The required software is installed 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 storage medium.

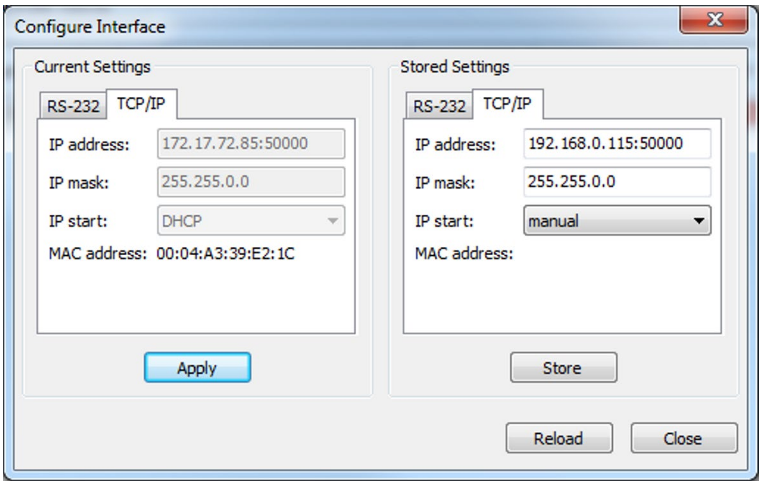
#### Adapting the TCP/IP Interface Parameters of the C-867

If you need to adapt the interface parameters of the C-867 to use the C-867 in a network, proceed as follows:

1. Establish communication between the C-867 and the PC via a different interface: USB (p. 72) or RS-232 (p. 71).
2. Select the ***C-867 > 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.




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

4. Make the necessary adaptations in the ***TCP/IP*** tab in the ***Stored Settings*** area:
   - ***IP address*** field: C-867's IP address in format xxx.xxx.xxx.xxx:50000
   - ***IP mask*** mask: Network's subnet mask
   - ***IP start*** field: C-867'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 C-867 by clicking on ***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 C-867 .
7. Close the ***Configure Interface*** window.
8. Close the connection with the C-867 by selecting the ***Connections > Close > C-867*** menu item in the main window of PIMikroMove.
9. Switch the C-867 off and on again.

### Establishing Communication via TCP/IP in the PC Software

***INFORMATION***

Communication via TCP/IP can fail if the network cable was connected to the Ethernet socket of the C-867 while the C-867 was switched on.

➢ If communication cannot be established, switch the C-867 off. Now reconnect the network cable and switch the C-867 on again.

## INFORMATION

➢ For communication via TCP/IP, the C-867 only has one unchangeable port (50000) available that cannot be used for more than one connection at a time.

### Establishing communication via TCP/IP

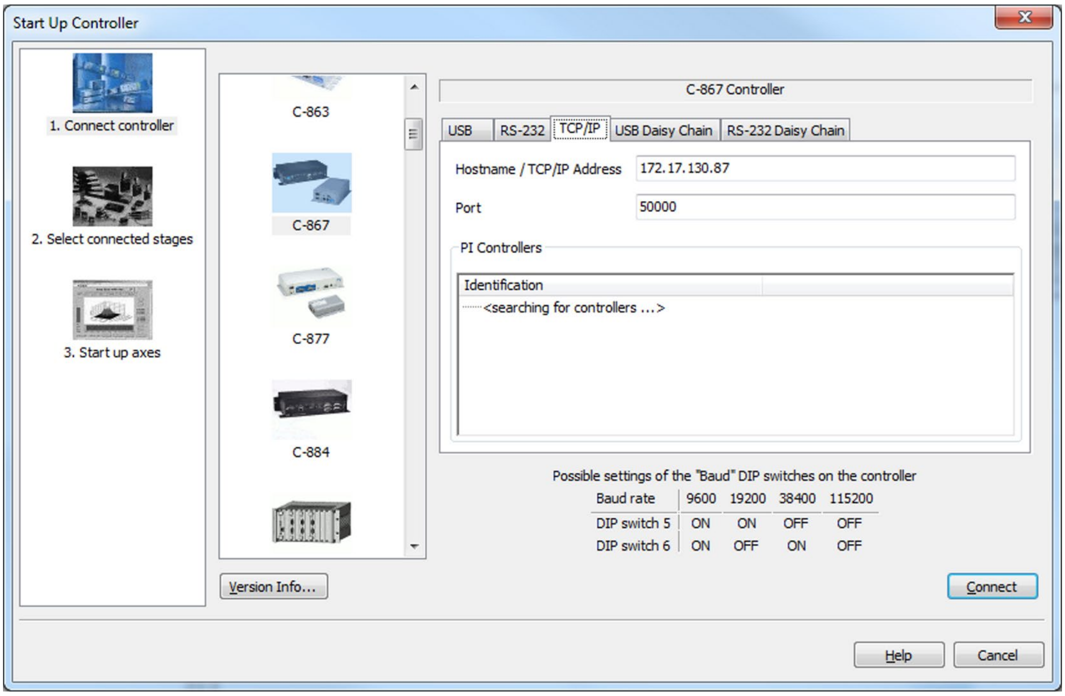
1. Start PIMikroMove.

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

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

2. Select ***C-867*** in the controller selection field.

3. Select the ***TCP/IP*** tab on the right-hand side of the window.




   The software now searches the network for all controllers of the C-867 type. After a successful search, all controllers that are in the same network are shown in the ***PI Controllers*** field.

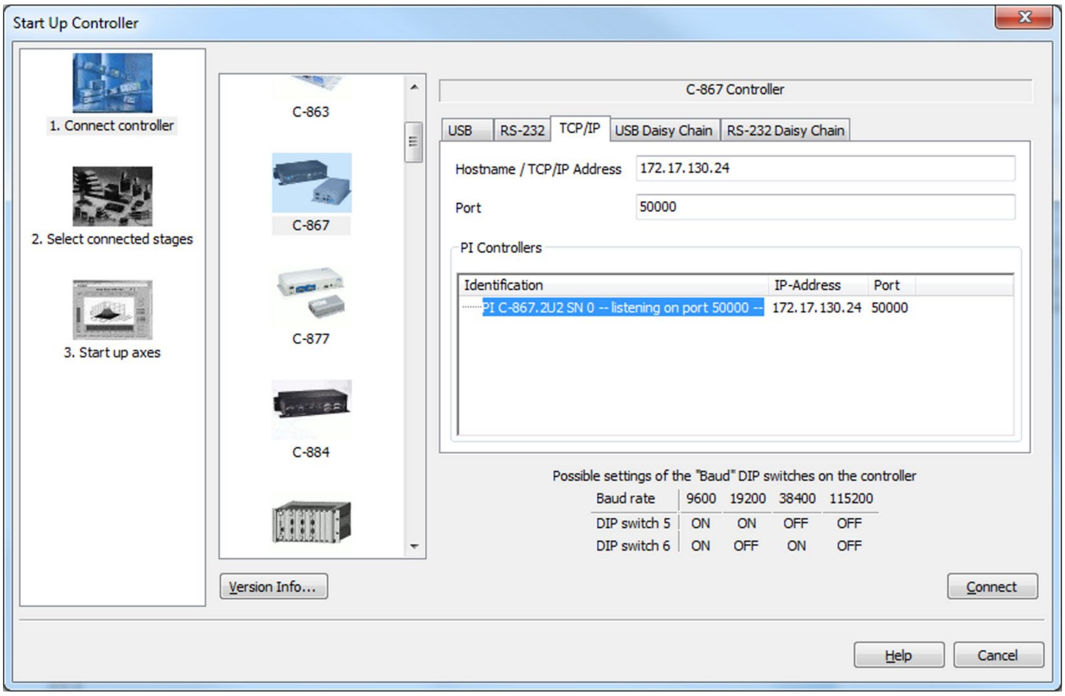
4. Click the entry of your C-867 model found in the list of controllers. This must show the status "listening on port 50000".

   – If several entries with the same name are shown, identify your C-867 on the basis of its nine-digit serial number.

   – If the C-867 is not displayed in the list of the controllers found, check the network settings (p. 303). Consult your network administrator if necessary.

   – Do **not** select a controller that is already connected via TCP/IP (status "connected to ..."). Otherwise, an error message will be displayed as soon as you try to establish communication with this controller.

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




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

If communication was established successfully, PIMikroMove guides you through configuring the C-867 for the connected positioner; refer to "Starting Motion" (p. 80).

➢ If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 303).

# 6.4 Building a Daisy Chain Network

## 6.4.1 Setting the Controller Address

### *INFORMATION*

With a daisy chain network, up to 16 controllers can be networked and operated via the same computer interface. Interlinking occurs in series. The first controller is connected directly to the PC.

Each controller in a daisy chain network has to have its own unique address. Controller addresses from 1 to 16 are possible and are set via DIP switches or the `IFS` command, depending on the device. In a daisy chain network, one of the controllers must have the address 1. This controller does not have to be the one directly connected to the PC.

➢ Set a unique address for each controller in a daisy chain network.

➢ Set the same baud rate for every controller in a daisy chain network.

If a C-867 is to be used in a daisy chain network, the controller address must be set via the `IFS` command. To do this, proceed as follows:

1. Connect the C-867 to the PC via USB (p. 64, p. 67), RS-232 (p. 67), or TCP/IP (p. 67).
2. Start PIMikroMove or PITerminal on the PC, and establish communication with the C-867 via the connected interface (USB (p. 72), RS-232 (p. 71), or TCP/IP (p. 73)).
3. If you use PIMikroMove, open the window for sending commands:
   – In the main window select the ***Tools > Command entry*** menu item or press the F4 key on the keyboard.

   In PITerminal the main window from which commands can be sent is opened automatically after establishing communication.
4. Send the `IFS?` command to display the interface parameters of the C-867.

   The default address of a controller is 1. For this reason, the answer to the query contains the following string:
   *<<DEVADR=1*
5. To change the controller address of a C-867, send the following command: `IFS 100 DEVADR <Controlleradresse>`

   100: Password for changing the parameter value in the nonvolatile memory of the controller
   <Controlleradresse>: e.g.,: 2
6. To save the changed controller address in the C-867, reboot the C-867. You have the following options:
   – Send the `RBT` command.
   – Switch the C-867 off and on again.

   The controller is rebooted. After the reboot, the changed controller address of the C-867 goes into effect.
7. End the communication with the C-867:
   – PIMikroMove: First close the window for sending commands. Then disable the communication connection to the controller via ***Connections > Close***.
   – PITerminal: Disable the communication connection to the controller via the ***Close*** button.
8. Disconnect the controller from the PC by removing the corresponding interface cable from the connection socket of the controller.

Set an individual controller address in this way for all C-867 controllers that are to be used in a daisy chain network. In the next step, the daisy chain network can be set up.

### 6.4.2 Building a Daisy Chain

***INFORMATION***

A C-867 can be operated in a common daisy chain network with the following controllers:

- Mercury DC motor controller of the C-863 series
- Mercury Step stepper motor controller of the C-663 series
- PILine® piezo motor controller of the C-867 series
- E-861 NEXACT® controller
- Q-Motion® controller of the E-873 series

**Requirements**

✓ You have set an individual unique controller address for each controller that is to be networked via a daisy chain:

- For controllers with DIP switches: Prior to switching on the device, you have set the controller address via the DIP switches.
- For controllers without DIP switches: You have set the controller address with the `IFS` command and then rebooted the controller, e. g., with the `RBT` command.

**Tools and Accessories**

- A network cable for every controller to be connected to the network. The following are available (as accessories):
  - C-862.CN1, 1 m
  - C-862.CN2, 3 m

**Networking the controllers**

➢ Set up the controller chain. Connect **RS-232 Out** on the previous controller to **RS-232 In** on the subsequent controller with the network cable.

➢ Connect the first controller of the chain to the PC. In the case of a C-867.2U2, use the RS-232 interface (p. 71).

The controller connected to the PC serves as master for the daisy chain network. This may not be the controller with controller address 1.

### 6.4.3 Establishing Communication for Networked Controllers

The following describes the procedure for PIMikroMove and for PITerminal with the C-867.2U2 as daisy chain network master.

***INFORMATION***

If you are establishing communication with a networked controller via PITerminal, the address of the controller to be addressed is required in every command line. See "Target and Sender Address" (p. 145) for details.

➢ Use PITerminal to test communication with networked controllers.

### Requirements

- ✓ You have read and understood the General Notes on Startup (p. 69).
- ✓ You have set up a daisy chain network .
- ✓ You have set a unique controller address and the same baud rate for all networked controllers (either via the DIP switches or the `IFS` command).
- ✓ All controllers in the daisy chain network are switched on.
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 64).
- ✓ 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 storage medium.

### Establishing communication with PIMikroMove

1. Start PIMikroMove.

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

   – If the ***Start up controller*** window does not open automatically, select the ***Connections > New...*** menu item in the main window.
2. Select the appropriate controller type in the controller selection field: Select ***C-867***.
3. Select the ***RS-232 Daisy Chain*** tab on the right-hand side of the window:
4. Make the settings for the interface in the tab:

   – In the ***COM Port*** field, select the COM port of the PC to which you have connected the C-867.

   – Set the value in the ***Baudrate*** field that is set for the C-867 (default setting: 115200).
5. Click the ***Scan*** button in the bottom section of the tab to list every controller in the daisy chain network.
6. Select the C-867.2U2 controller from the list.

   The selection must match the controller type that you selected in step 2.
7. Click ***Connect*** to establish communication with the controller selected.

   When communication has been successfully established, PIMikroMove guides you through configuring the C-867 for the connected positioner.

   – Proceed further as described in "Starting Motion" (p. 80).
8. If you wish to connect an additional controller of the daisy chain network, select the ***Connections > New...*** menu item in the main window.
9. Perform steps 2, 6 and 7 once again in the specified order.
10. Repeat steps 8, 2, 6 and 7 for every additional controller of the daisy chain network, which you want to connect.

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

➢ Select the ***Connections > Close*** menu item for the corresponding controller in the main window.

### Establishing communication with PITerminal

**_INFORMATION_**

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

➢ Make sure that the ***Mercury*** button is **not** activated in PITerminal.

1. Start PITerminal.
2. Click the ***Connect...*** button.

   The ***Connect*** window opens.
3. Select the ***RS-232*** window in the ***Connect*** window.
4. Make the settings for the interface in the tab or select the C-867.2U2 controller that you want to connect:
   - In the ***COM Port*** field, select the COM port of the PC to which you have connected the C-867.
   - Set the value in the ***Baudrate*** field that is set for the C-867 (default setting: 115200).
5. Click ***OK*** to establish communication.
6. Send the `*IDN?` command for every controller in the daisy chain network to check communication:
   - `*IDN?` to query the device identification string of the controller with address 1; the controller address is not required (because = 1)
   - `2 *IDN?` to query the device identification string of the controller with the address 2.

## 6.5 Starting Motion

The PIMikroMove is used in the following to move the positioner. The program guides you through the following steps so that you do not have to deal with the respective GCS commands:

- Configuration of the C-867 for the connected positioner
- Switching on the servo mode (closed-loop operation)
- Performing a reference move; details see "Referencing" (p. 50).

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

### NOTICE

**Selecting an incorrect positioner type**

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

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

### NOTICE

**Oscillation!**

Unsuitable settings of the notch filter and the C-867's servo control parameters can cause the positioner to oscillate. Oscillation can damage the positioner and/or the load affixed to it.

➢ Secure the positioner and all loads adequately.

➢ If the mechanics is oscillating (unusual operating noise), immediately switch off the servo mode or the C-867.

➢ Only switch on the servo mode after you have modified the settings of the notch filter and the servo control parameters of the C-867; see "Setting the Notch Filter" and "Optimizing Servo Control Parameters" (p. 88).

➢ If, due to a very high load, oscillation occurs during the reference move, follow the instructions for the reference move in "Troubleshooting" (p. 303).

### INFORMATION

After communication has been established between the C-867 and the PC, PIMikroMove guides you through the configuration of the C-867 for the connected positioner. Selection of the configuration steps offered by PIMikroMove is based on evaluation of the following parameter values in the volatile memory of the C-867:

- ***Stage Name*** (ID 0x3C): The value is used by PIMikroMove as a criterion for finding a suitable parameter set in the positioner databases.
- ***Stage Type*** (ID 0x0F000100): The value was loaded from the ID-Chip (p. 15) of the connected positioner when the C-867 is switched on.

Possible configuration steps:

- When the values of the parameters 0x3C and 0x0F000100 are identical, PIMikroMove assumes that all parameters of the C-867 have already been adapted to the connected positioner. The ***Start up controller*** window goes directly to the ***Start up axes*** step, where the reference move can be started.
- If the values of the parameters 0x3C and 0x0F000100 are not identical, the ***Stage Type Configuration*** window opens. The ***Yes, configure for ...*** button can be used to load a suitable parameter set from a positioner database to the C-867. After the parameter set has been loaded, the ***Start up controller*** window goes to the ***Start up axes step***. If a matching parameter set is not in the positioner databases, a corresponding notice will appear in the ***Stage Type Configuration*** window.
- If the value of the parameter 0x0F000100 is empty because the positioner does not have an ID chip, for example, the ***Start up controller*** window will go to the ***Select connected stages*** step.

### Requirements

- ✓ You have read and understood the General Notes on Startup (p. 69).
- ✓ PIMikroMove is installed on the PC (p. 64).
- ✓ You have read and understood the PIMikroMove manual. The links to the software manuals are in the file A000T0081 on the data storage device with the PI software.
- ✓ You have installed the latest version of the PISTAGES3.DB database onto your PC (p. 64).
- ✓ If PI provided a custom positioner database for your positioner, the dataset was imported into PIStages3 (p. 66).
- ✓ 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 C-867 to the positioner (p. 66).
- ✓ You have established communication with PIMikroMove between the C-867 and the PC (p. 70).

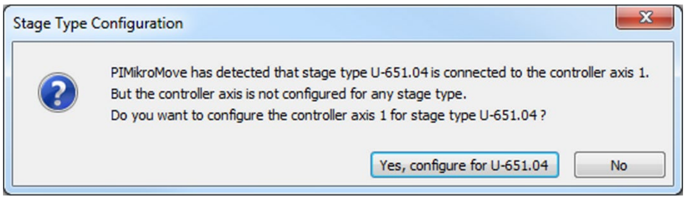
### Starting motion with PIMikroMove

1. If one of the two following points applies, configure the C-867 for the connected positioner:

   – The ***Stage Type Configuration*** dialog has opened.

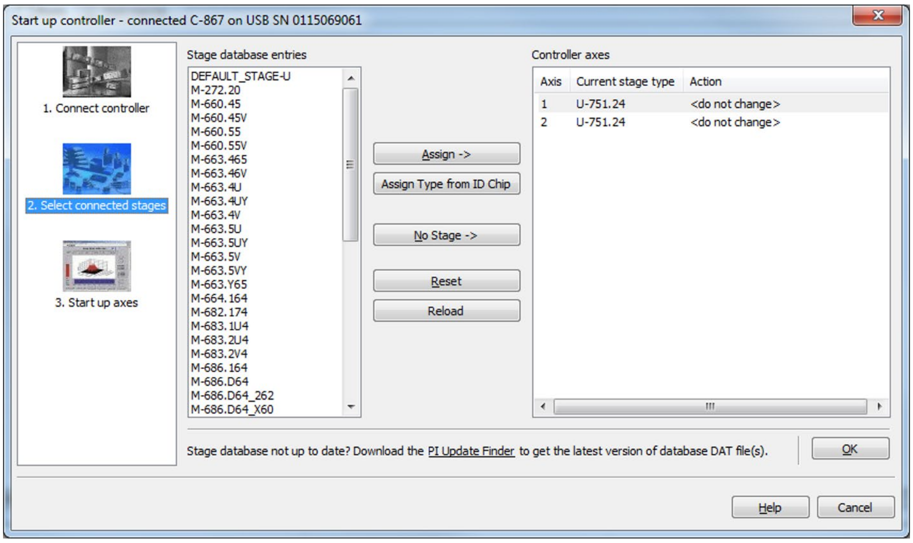
   – The ***Select connected stages*** step is displayed In the ***Start up controller*** window.

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

   – Click the ***Yes, configure for ...*** button to load the appropriate parameter set from the positioner database into the C-867. This opens the ***Save all changes permanently?*** dialog.

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




a) Select the matching positioner type. You have two options:
   - Click ***Assign Type from ID Chip***.
   - Mark the appropriate positioner type in the ***Stage database entries*** list and click ***Assign***.

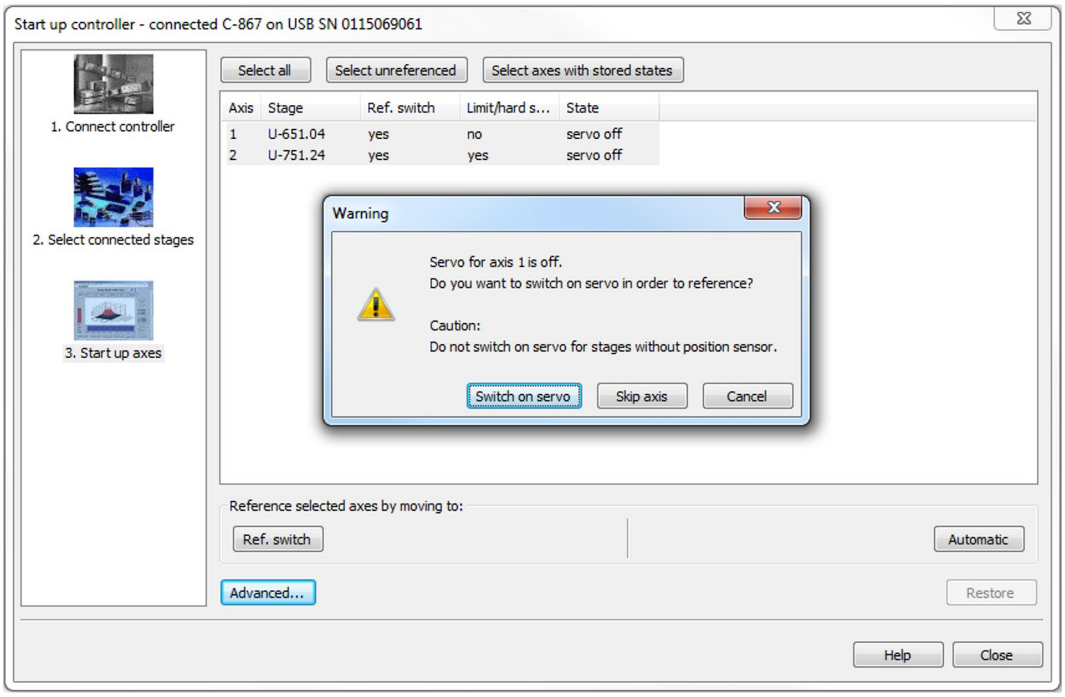
b) Confirm selection with ***OK*** to load the parameter settings for the selected positioner type from the positioner database into the C-867. This opens the ***Save all changes permanently?*** dialog.

2. Specify how you want to load the parameter settings into the C-867 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 C-867. The settings are lost when the C-867 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 C-867. The settings are available immediately after switching on or rebooting the C-867 and do not need to be reloaded.

   The ***Start up controller*** window changes to the ***Start up axes*** step.

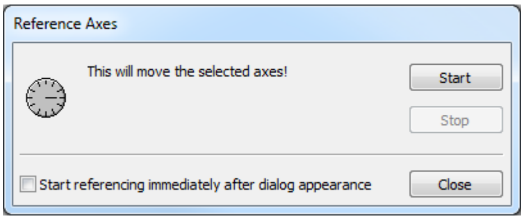
3. During the ***Start up axes*** step, do a reference move for the axis so that the controller knows the absolute axis position: You have the following options (options not supported by the positioner/controller either do not exist or cannot be activated):
   – If you want to start the reference move to the reference switch, click ***Ref. switch***.
   – If you want to start the reference move to the negative limit switch, click ***Neg. limit***.
   – If you want to start the reference move to the positive limit switch, click ***Pos. limit***.

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



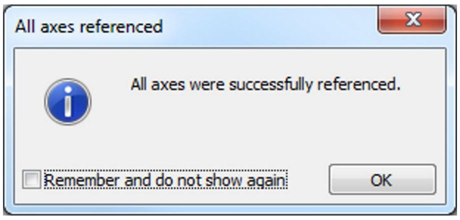

a) Switch on the servo mode by clicking on the ***Switch on servo*** button.

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




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

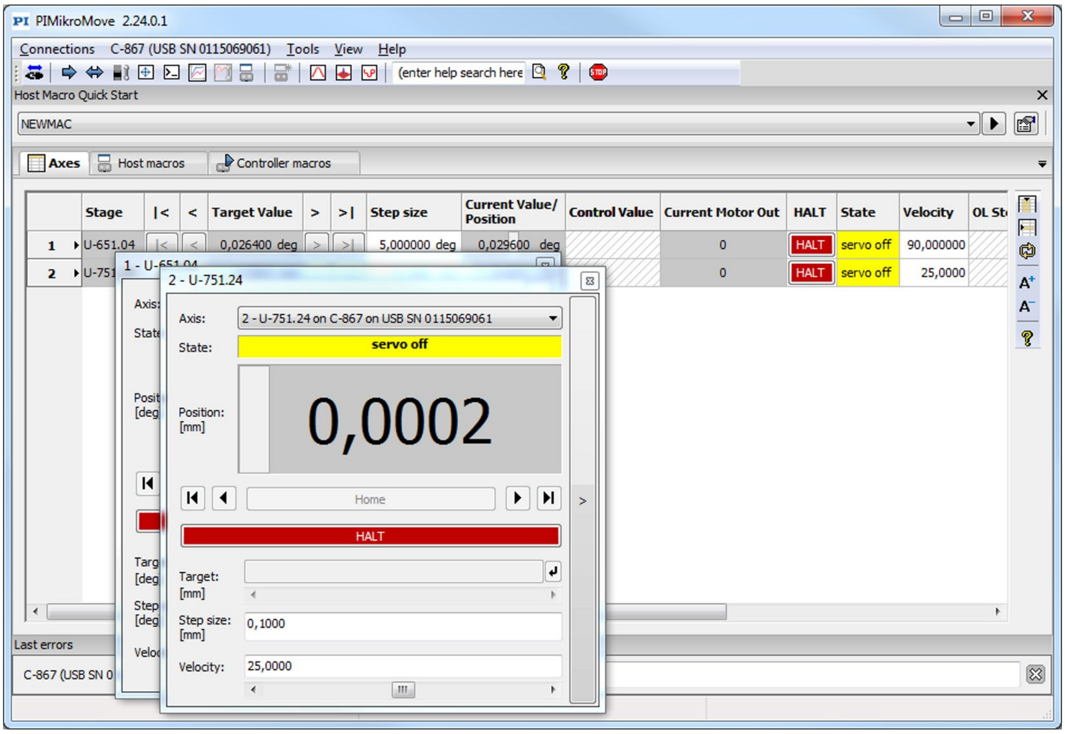
If the corresponding message is displayed after a successful reference move:




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

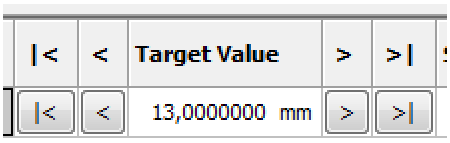
4. After a successful reference move, close the ***Start up controller*** window by clicking ***Close***.

The main window of PIMikroMove opens.




5. Test the motion of the axis several times.

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




## 6.6 Activating Additional Features

Additional features are represented in the controller in the form of parameters. An additional feature must be activated in order to use it. The corresponding parameters are only visible when a feature is activated.

Active features are displayed in the controller's response to the `VER?` query (p. 247). They are shown as system components there.

A description of how to activate an additional feature in the C-867 is show here. If it is a licensed feature, this is done by unlocking the feature.

The procedure for PIMikroMove is described in the following.

### Requirements

✓ You have established communication between the C-867 and the PC (p. 70) with PIMikroMove.

## 6.6.1 Unlocking Licensed Features

The license number for the feature must be entered into the controller in order to unlock the licensed feature in the controller. The feature is then active and can be used after entering the license number and rebooting the controller.

A document with a license number is issued when you purchase an additional feature.

The license number for a feature is entered into the corresponding parameter of the controller:

| Feature | Parameter name | Parameter ID |
|---|---|---|
| Slow Motion Drive | Slow Motion Feature License Number | 0xE002700 |

### Unlocking an Additional Feature

1. Open the system parameter window in the main window of PIMikroMove via the ***C-867 > Show system parameters*** menu item.

   The system parameter window is shown with the following values:

   – ***Active Values***: Values in the controller's volatile memory

   – ***Startup Values***: Values in the controller's nonvolatile memory

   a) If the ***Startup Values*** column is not displayed, mark the ***Show startup values*** option in the***Configure View*** menu.

   b) If the required parameter is not shown in the list, select the ***Show all*** function in the ***Configure View*** menu.

2. Click the parameter.

   A dialog for entering the password is opened because the parameter is write-protected (indicated by the lock symbol).

3. Enter `advanced` into the input field of the password dialog and click ***OK***.

   The dialog is closed and the parameter value can be changed.

4. Enter the license number for the feature into the corresponding parameter's ***Startup Value*** field.

5. Press the Enter key or click outside the input field with the mouse to transfer the parameter value to the nonvolatile memory of the C-867.

6. Close the system parameter window.

7. Switch the C-867 off and on again or reboot the C-867 with the `RBT` command.

After rebooting, the controller checks the entered license number for validity. If the license is valid, the feature is now unlocked and activated. The corresponsing parameters are now visible in the controller and the feature can be configured and used.

### 6.6.2 Activating Features

The C-867 supports the following additional features:

- Slow Motion: Fine-tuning of the controller for extremely slow motion (see C-867.L01 user manual)
- Adaptive Control: Optimization of the PID control for large dynamic ranges (p. 36)
- Camming: Electronic camming (p. 41)

Activating and deactivating of the additional features in the controller is done via parameters:

| Feature | Parameter name | Parameter ID |
|---|---|---|
| Slow Motion | Enable Slow Motion Feature | 0xE002702 |
| Adaptive Control | Enable Adaptive Control Feature | 0xE002712 |
| Camming | Enable Camming Feature | 0xE002722 |

off (0) = feature deactivated

on (1) = feature activated

#### Activating the feature

1. Open the system parameter window in the main window of PIMikroMove via the ***C-867 > Show system parameters*** menu item.

   The system parameter window is shown with the following values:

   – ***Active Values***: Values in the controller's volatile memory

   – ***Startup Values***: Values in the controller's nonvolatile memory

   a) If the ***Startup Values*** column is not displayed, mark the ***Show startup values*** option in the***Configure View*** menu.

   b) If the required parameter is not shown in the list, select the ***Show all*** function in the ***Configure View*** menu.
2. Click the parameter.

   A dialog for entering the password is opened because the parameter is write-protected (indicated by the lock symbol).
3. Enter `advanced` into the input field of the password dialog and click ***OK***.

   The dialog is closed and the parameter value can be changed.
4. Select the value "on" in the ***Startup Value*** field of the corresponding parameter.
5. Press the Enter key or click outside the input field with the mouse to transfer the parameter value to the nonvolatile memory of the C-867.
6. Close the system parameter window.
7. Switch the C-867 off and on again or reboot the C-867 with the `RBT` command.

The feature in the controller is activated after rebooting the C-867. The corresponding parameters are now visible in the C-867 and the feature can be configured and used.

## 6.7 Optimizing the Servo Control Parameters

The dynamic characteristics of the system (e.g., starting motion, oscillation behavior during motion, overshoot, and settling time) are optimized by adjusting the parameters for the servo algorithm and control value corrections. The optimal settings depend on your application and your wishes.

Typically, optimization is determined empirically, i.e., the behavior of the positioner is monitored with different values in closed-loop operation. Optimization is done via the following parameters (refer to "Servo Algorithm and Other Control Value Corrections" (p. 30) for details and information on further relevant parameters):

- P, I, D terms and I limit of parameter groups 0 to 4 (IDs 0x4n1, 0x4n2, 0x4n3, 0x4n4; n takes on a value of 0 to 4 depending on the parameter group)
- Parameters for switching between parameter groups 0 to 4, e.g., window limits (IDs 0x4n6, 0x4n7; n takes on a value from 0 to 4 depending on the parameter group)
- Velocity-dependent offset and offsets for the positive and negative direction of motion (IDs 0x48, 0x33, 0x34)

### *INFORMATION*

Parameter groups 0 to 4 are used as follows:

- Optimization of the motion using parameter groups 1 to 4, depending on the setting of the ***Number Of Servo Parameter Groups*** parameter (0x400)
- Optimization of the settling behavior of the system at the end of the motion by:
  - Parameter group 0, when parameter 0x400 has a value 2 to 5
  - Parameter group 1, when parameter 0x400 has the value 1 (the entrance and exit windows of parameter group 0 are still used as settling windows for determining the on-target state).

### *INFORMATION*

Experienced users can also enable the two-phase control (p. 30) to improve the settling behavior.

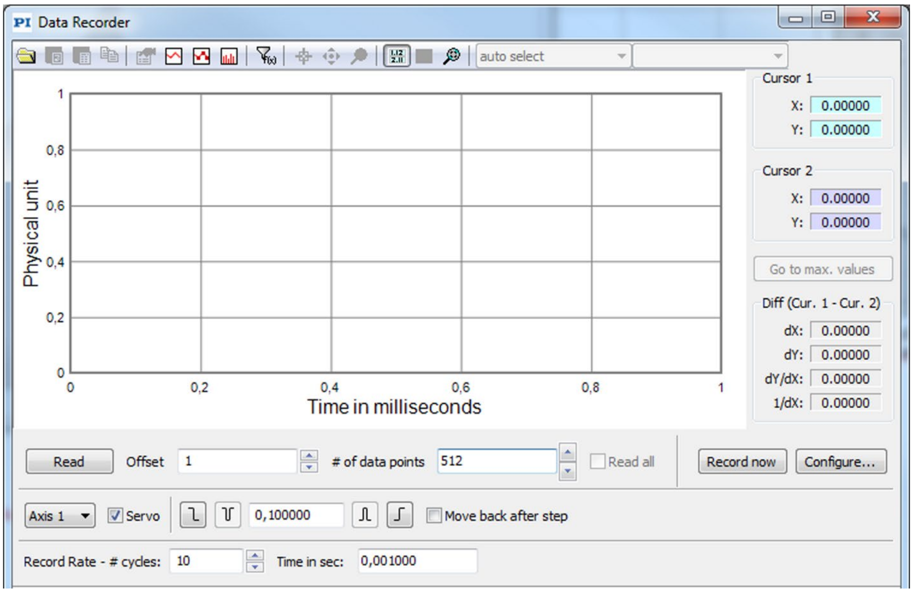
In the following, PIMikroMove is used for optimizing the dynamic characteristics of the system. The two-phase control is **not** taken into account here.

#### Requirements

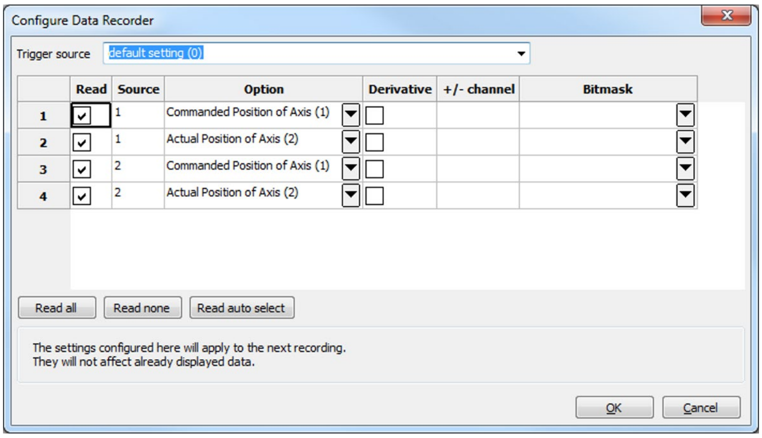
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and fixing).
- ✓ You started initial motion (p. 80) with PIMikroMove.
- ✓ All devices are still ready for operation.

### Checking the servo control parameters: Measuring the step response

1. Open the ***Data Recorder*** window in the main window of PIMikroMove via the ***C-867 > Show data recorder*** menu item.
2. Select the axis to be checked via the ***Axis #*** button and switch the servo mode on for the axis by clicking the ***Servo*** checkbox (tick in the checkbox).
3. Configure the data recorder.
   a) Set the size of the step to be made to a value that is typical for your application, e.g., 0.100000 (specified in physical units).
   b) Set the value 10 for the record table rate in the ***Record Rate - # cycles*** field.
   c) Set the value 8192 (or less) for the number of data points to be read for the graphic display in the field ***# of data points***.




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




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

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

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

Examples for step responses:

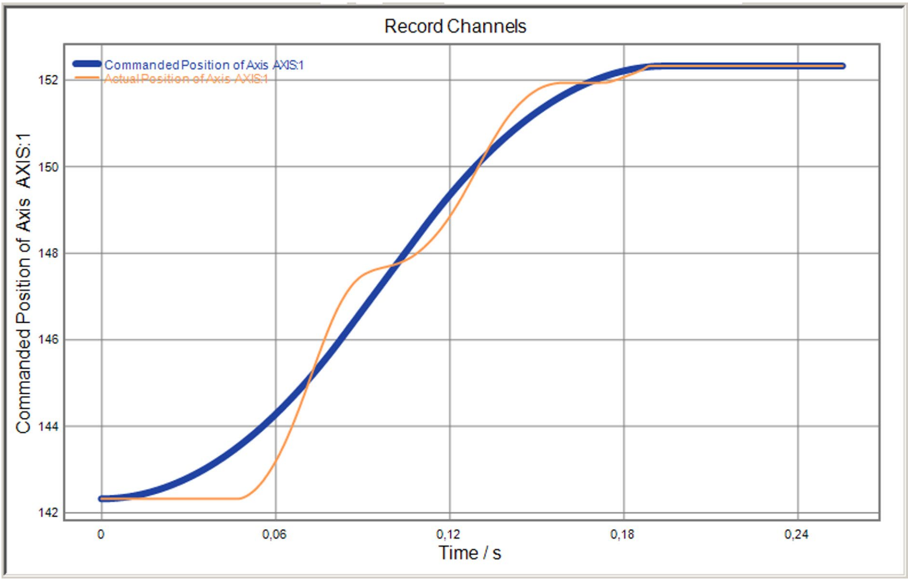



Figure 12: Improper settings, causing oscillation and unacceptable settling behavior

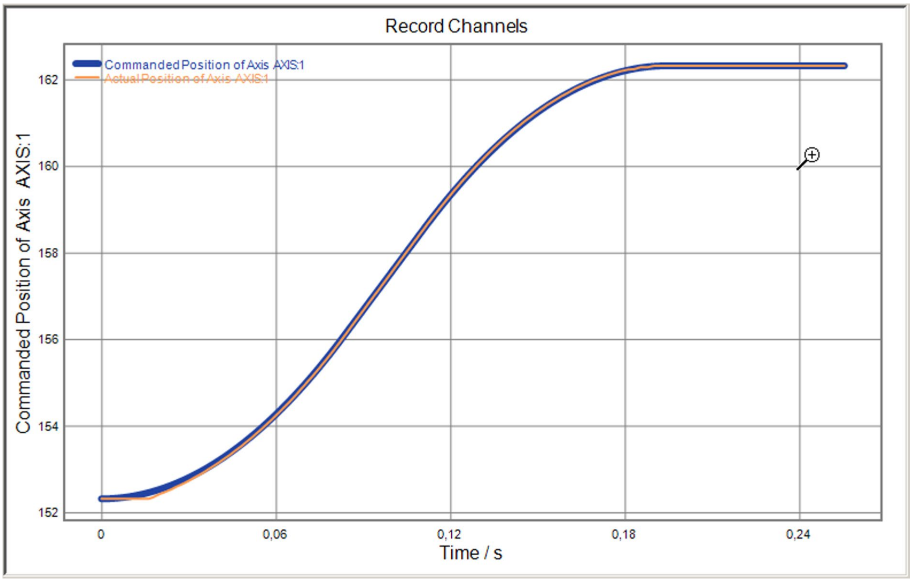



Figure 13: Settling behavior already almost optimum, but behavior at start of the motion not yet satisfactory (offset settings still have to be optimized)

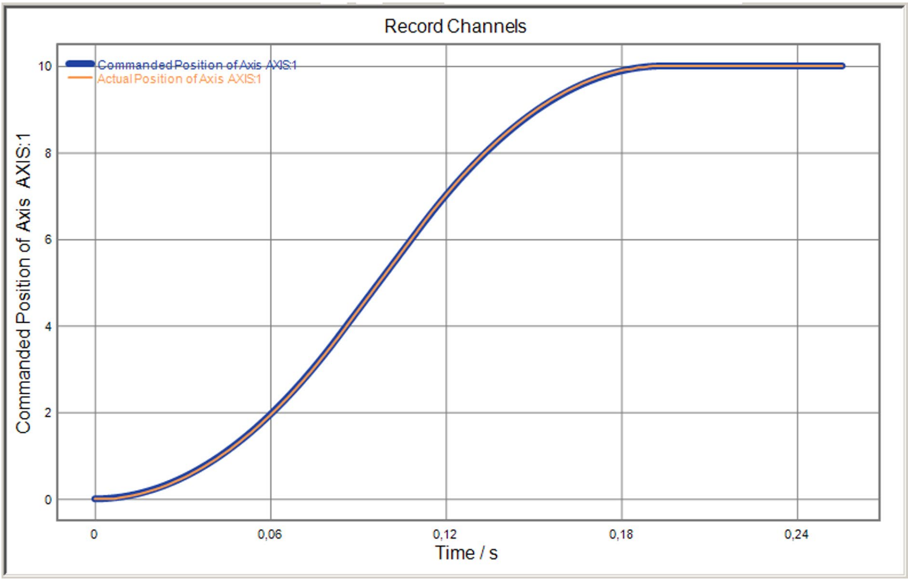



Figure 14: Optimum dynamic characteristics of the system, no adjustments necessary

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

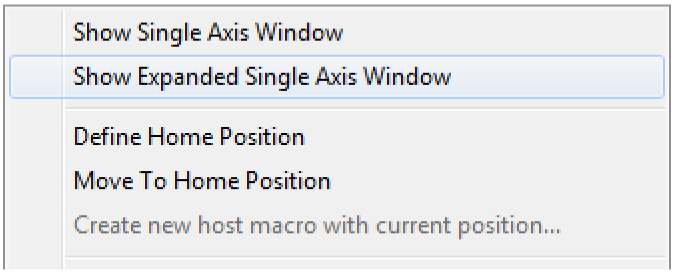
– You already have optimal parameter settings and do not have to do anything further.

If the result is not satisfactory:

– Optimize the parameters for the dynamic characteristics of the system; see below.

### Optimizing the servo control parameters

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




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

   If the parameters to be changed are not included in the list on the right-hand side of the window, click ***Configure View -> Select parameters...*** and add them to the list.

   a) Type the new parameter value into the corresponding input field in the ***Active Value*** column of the list.

   b) Press the Enter key on the PC keyboard or click outside the input field with the mouse to transfer the parameter value to the volatile memory of the controller. Note: If a parameter value in the volatile memory (***Active Value*** column) is different to the parameter value in the nonvolatile memory (***Startup Value*** column), the line in the list is highlighted in color.

3. Rerecord the positioner's step response in the ***Data Recorder*** window.

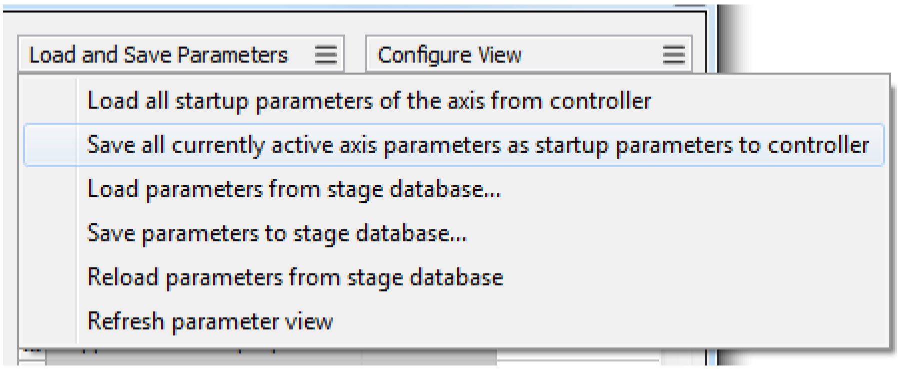
   If the result is not satisfactory:

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

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

   – Save a parameter set in the positioner database on the PC by clicking ***Load and Save Parameters -> Save parameters to stage database...***, see "Creating or Modifying a Positioner Type" (p. 278).

   – Transfer the current values of the listed parameters from the volatile memory to the nonvolatile memory of the C-867 by clicking ***Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller***.



# 7 Operation

## 7.1 Protective Functions of the C-867

### 7.1.1 Protection Against Overheating

When a high control value remains set over a long period of time, the connected positioner can heat up. Overheating can result in damage to the positioner.

The ***PID Maximum Output Time (s)*** parameter (ID 0x7B) specifies the maximum time period for which a high control value may be set in closed-loop operation. A high control value is present when the following applies:

Current absolute measure of the control value of the control value ≥ 95 % of ***Maximum Motor Output*** (ID 0x9).

If the high control value is still set after the maximum time period has expired, the C-867 will react as follows to protect the system against damage:

- The control value is set to the value zero for the axis in question.
- The servo mode is switched off for the axis in question.

### 7.1.2 Behavior with Motion Errors

Motion errors can be caused for example, by malfunctions of the drive or the position sensor of the positioner.

A motion error occurs, when the position error (i.e., the absolute value of the difference between the current position and the commanded position) exceeds the specified maximum value in closed-loop operation. The range in which the deviation may lie is specified by the ***Maximum Position Error (Phys. Unit)*** parameter (ID 0x8).

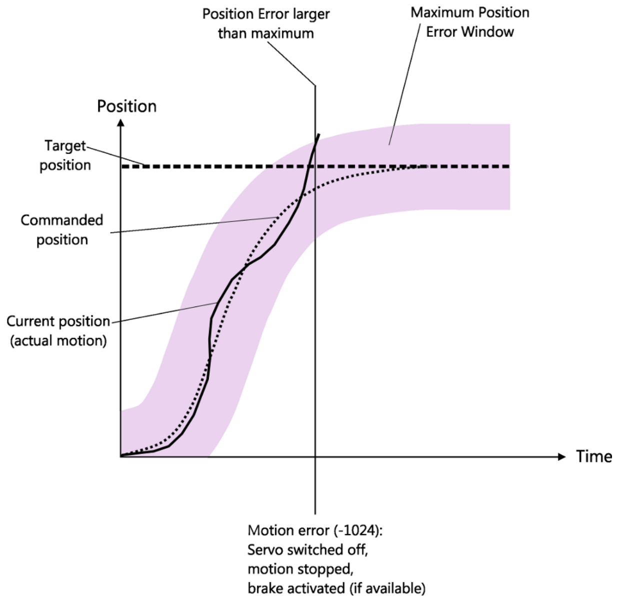
Motion errors can have the following causes, for example:

- Malfunction of the drive
- Malfunction of the position sensor
- Positioner malfunction

If a motion error occurs, the C-867 reacts as follows to protect the system against damage:

- The servo mode is switched off for the axis in question.
- If applicable, the brake is activated for the axis in question.
- All motion is stopped.
- Error code 1024 is set.

Then restore the operational readiness (p. 94) for the C-867.




### *INFORMATION*

With the `CTO` (p. 161) and `TRO` (p. 242) commands, you can program the digital output lines of the C-867 so that they are activated in the case of motion errors. The programmed output lines remain activated until the error code is reset to 0. Refer to "Configuring the "Motion Error" Trigger Mode" (p. 104) for details.

## 7.1.3 Re-establishing Readiness for Operation

1. Send the `ERR?` (p. 178) 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 C-867 has **not** overheated (internal temperature is maximum 65 °C).
3. If the servo mode was switched off after an error or overheating:
   – Switch on the servo mode for the axis with the `SVO` command.

   When the servo mode is switched on, the target position is set to the current axis position.
4. When the error has occurred during the trajectory execution:
   – Delete the trajectory points still present in the buffer with the `TGC` (p. 235) command.

### *INFORMATION*

With the `CTO` (p. 161) and `TRO` (p. 242) commands, you can program the digital output lines of the C-867 so that they are activated in the case of motion errors. The programmed output lines remain activated until the error code is reset to 0. Refer to "Configuring the "Motion Error" Trigger Mode" (p. 104).

## 7.2 Trajectories for Motion Paths

### 7.2.1 Operating Principle of the Trajectory Buffer

In closed-loop operation, the C-867 can process externally calculated one- or two-dimensional motion paths (e.g., circles, sine curves) as trajectories (p. 2).

The individual target positons of the motion path must be loaded into the trajectory buffer of the C-867 as trajectory points. During the execution of the trajectory, the buffer outputs the points with a fixed chronological interval. The points are output in the order they were loaded to the trajectory buffer (FIFO principle: **F**irst **I**n **F**irst **O**ut).

The content of a trajectory buffer is only present in the volatile memory of the C-867 and cannot be saved to the permanent memory.

The trajectories are permanently assigned to the axes of the C-867: Trajectory 1 to axis 1, trajectory 2 to axis 2.

### 7.2.2 Commands and Parameters for Trajectories

The trajectory buffer of the C-867 can be configured via the following parameters:

| Parameters | Description and Possible Values |
|---|---|
| ***Maximum Buffer Size***<br>0x22000020 | Maximum number of trajectory points in the trajectory buffer<br>This parameter indicates the maximum number of points that can be loaded to the trajectory buffer for a trajectory.<br>During the execution of motion paths that require more than this number of points, trajectory points must be reloaded. |

The following commands are available for trajectories:

| Command | Arguments | Function |
|---|---|---|
| `TGT` | <NoOfServoCycles> | Set timing for trajectories |
| `TGT?` | | Query timing for trajectories |
| `TGA` | {<Trajectory> <Point>} | Load trajectory point to the trajectory buffer |
| `TGC` | [{<Trajectory>}] | Delete all points of a trajectory |
| `TGS` | [{<Trajectory>}] | Start execution of a trajectory |
| `TGF` | [{<Trajectory>}] | Complete execution of a trajectory |
| `TGL?` | [{<Trajectory>}] | Query the number of points of a trajectory present in the trajectory buffer |

### 7.2.3 Working with Trajectories

*NOTICE*

**Execution of trajectories!**

The C-867 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

- Values of the trajectory points
- Timing for the trajectories
- Sufficiently fast refilling of the trajectory buffer

Following an unsuitable trajectory can cause the positioner to oscillate or stop motion abruptly. Oscillation or stopping abruptly can damage the positioner and/or the load fixed to it.

➢ Therefore, pay attention to the following when working with trajectories:
  - The path that is specified by the trajectory points must be continuously differentiable at least twice.
  - During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
  - When following the trajectory, an abrupt stop may **not** damage the load on the positioner.
  - To generate the trajectory points and continuously transfer them to the C-867 during the trajectory execution, it is recommended to use a suitable program.

*INFORMATION*

For working with trajectories, it is recommended to use the ***Trajectory Assistant*** in PIMikroMove (call via the menu of the C-867). This allows you to define and execute trajectories conveniently.

The timing for trajectories is set with the TGT (p. 238) command.

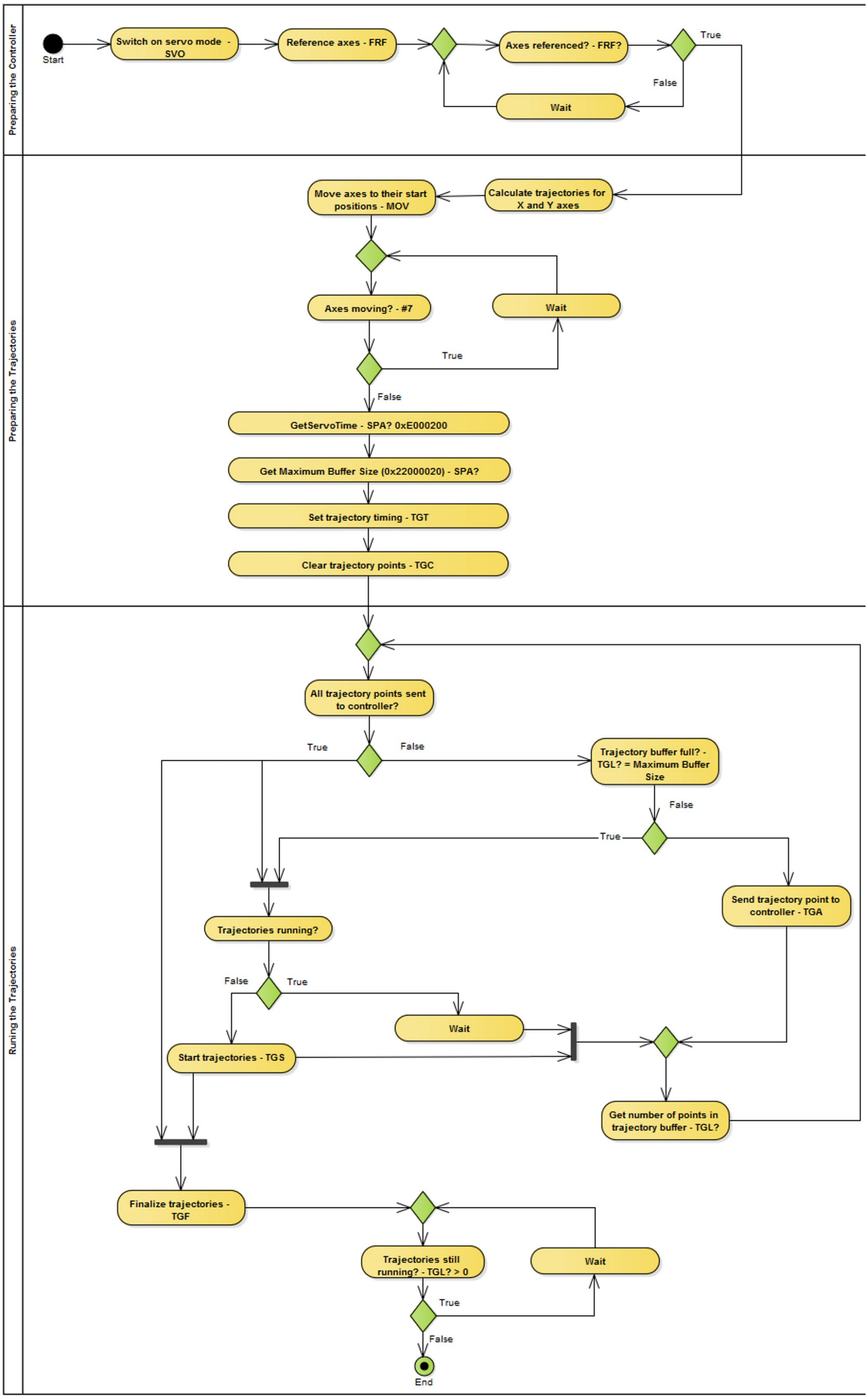
Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with the TGA (p. 234) command. The maximum number of points in the trajectory buffer is determined by the ***Maximum Buffer Size*** parameter (0x22000020).

The TGS (p. 237) command starts the execution of a trajectory. During the execution of a trajectory, the buffer must be refilled fast enough.

The TGF (p. 236) command properly completes the execution of a trajectory.

If the execution of a trajectory is cancelled after an error or stopped with `STP`, `#24`, or `HLT`, the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading or executing a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 237), deletion with TGC (p. 235)).

The following diagram shows an example of the sequence of a trajectory execution. Corresponding example programs are found on the PC after the PC software for the C-867 has been installed.



Preparing the Controller
Start
Switch on servo mode - SVO
Reference axes - FRF
Axes referenced? - FRF?
True
False
Wait
Preparing the Trajectories
Move axes to their start positions - MOV
Calculate trajectories for X and Y axes
Axes moving? - #7
Wait
True
False
GetServoTime - SPA? 0xE000200
Get Maximum Buffer Size (0x22000020) - SPA?
Set trajectory timing - TGT
Clear trajectory points - TGC
Runing the Trajectories
All trajectory points sent to controller?
True
False
Trajectory buffer full? - TGL? = Maximum Buffer Size
False
True
Send trajectory point to controller - TGA
Trajectories running?
False
True
Wait
Start trajectories - TGS
Get number of points in trajectory buffer - TGL?
Finalize trajectories - TGF
Trajectories still running? - TGL? > 0
Wait
True
False
End

## 7.3 Data Recorder

### 7.3.1 Configuring the Data Recorder

The C-867 contains a real-time data recorder. The data recorder can record different variables for the axes (e.g., current position).

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

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

***INFORMATION***

The following settings of the data recorder can only be changed in the volatile memory of the C-867:

- Data to be recorded
- Trigger option for triggering the recording
- Record table rate

After the C-867 has been switched on or rebooted, factory settings will be active unless a configuration takes place with a startup macro.

**Reading general information from the data recorder**

- Send the `HDR?` command (p. 182).

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

**Configuring data to be recorded**

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

- Send the `DRC?` command (p. 173) to read out the current configuration. Data recorder tables with the record option 0 are deactivated, i.e., nothing is recorded. By default, the data recorder tables of the C-867 record the following:
  - Data recorder table 1: Axis 1 - record option 1: Commanded position of axis 1
  - Data recorder table 2: Axis 1 - Record option 2: Current position of axis 1
  - Data recorder table 3: Axis 2 - record option 1: Commanded position of axis 2
  - Data recorder table 4: Axis 2 - record option 2: Current position of axis 2
- Configure the data recorder with the `DRC` command (p. 172).

***INFORMATION***

The time reference of the recorded data points can be easily created by recording the current value of the timer as well (record option 44; get with `TIM?`).

### Configuring the recording trigger

You can specify how the recording is to be triggered.

- Get the current trigger option with `DRT?` (p. 177)
- Change the trigger option with the `DRT` command (p. 176). The trigger option applies to all data recorder tables with a record option not set to 0.

### Setting the record table rate

- Send the `RTR?` command (p. 221) to read out the record table rate of the data recorder.

  The parameter indicates the number of servo cycles required for recording each data point. The default value is 10 servo cycles. The servo cycle time of the C-867 is 50 µs.

- Change the record table rate with the `RTR` command (p. 220).

  As the record table rate increases, the maximum duration of the data recording is increased.

### Configuring data processing

You can configure recorded data processing with the parameters in the following list.

| Parameters | Description and Possible Values |
|---|---|
| ***Recorded Points Per Trigger***<br>0x16000001 | Number of data points to be recorded per trigger.<br>0 = Unlimited number of points (default setting)<br>*n* = *n* points; *n* is a whole-number value, lowest possible value is 1<br>The behavior of the C-867 when the data recorder tables are full depends on the value of the ***Data Recorder Buffer Mode*** parameter. |
| ***Clearing Of RecTable On Trigger***<br>0x16000002 | Write mode during the recording<br>Determines how the data points are written to the data recorder tables when the recording is started by a trigger.<br>0 = Recorded points are added to the already exisiting contents of the data recorder tables (default setting)<br>1 = The trigger deletes the data recorder tables, i. e., the recording always starts with the first point of the data recorder tables. |
| ***Data Recorder Buffer Mode***<br>0x16000003 | Behavior with full data recorder tables<br>0 = Recording ends (default setting)<br>1 = Recording is continued with the first point of the data recorder table and overwrites the existing contents. The value of the ***Data Recorder Buffer Overflow*** parameter is increased by 1. |
| ***Data Recorder Buffer Overflow***<br>0x16000004 | Buffer overflow counter of the data recorder<br>Counts how often the recording starts again with the first point of the data tables when the ***Data Recorder Buffer Mode*** parameter has the value 1.<br>Reading out the recorded data with `DRR?` resets the value of the buffer overflow counter to zero. The parameter is write-protected. |

### 7.3.2 Starting the Recording

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

Irrespective of the trigger option set, the data recording is always triggered when a step response measurement is started with `STE` (p. 230).

The data recording always takes place for all data recorder tables whose record option is not set to 0. The behavior of the C-867 when the data recorder tables are full depends on the value of the ***Data Recorder Buffer Mode*** parameter; see "Configuring data processing" (p. 98).

### 7.3.3 Reading Recorded Data

***INFORMATION***

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

- Read out the last recorded data with the `DRR?` command (p. 174).

  The data is output in the GCS array format (see the SM146E user manual (p. 4)). Depending on the value of the ***Data Recorder Buffer Mode*** parameter, reading out the recorded data with `DRR?` resets the value of the buffer overflow counter to zero; see "Configuring data processing" (p. 98).
- Get the number of points contained in the last recording with the `DRL?` command (p. 174).

## 7.4 Digital Output Signals

The digital outputs of the C-867 are available at the **I/O** socket (p. 316).

- Get the number of the output lines available on the C-867 with the `TIO?` command (p. 241).

External devices can be triggered via the digital outputs of the C-867. Potential applications:

- Linking the trigger output to the motion of the axis. Details and examples can be found in this section.
- Direct switching of output lines, e. g., in macros. Details and examples of macros can be found in "Controller Macros" (p. 126).

### 7.4.1 Commands for Digital Outputs

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

| Command | Syntax | Function |
|---|---|---|
| `CTO` | CTO {<TrigOutID> <CTOPam> <Value>} | Configures the conditions for the trigger output. Couples the trigger output to the axis motion. |

| Command | Syntax | Function |
|---|---|---|
| `DIO` | DIO {<DIOID> <OutputOn>} | Switches digital output lines directly to the low or high state, either separately or all lines at once. Should not be used for output lines where the trigger output is activated with `TRO`. |
| `TRO` | TRO {<TrigOutID> <TrigMode>} | Activates or deactivates the trigger output conditions set with `CTO`. Default: Trigger output deactivated. |

One configuration setting can be made per `CTO` command:

`CTO <TrigOutID> <CTOPam> <Value>`

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

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

| <Value> | Trigger mode | Short description |
|---|---|---|
| 0 (default) | Position Distance | Once the axis has moved a specified distance, a trigger pulse is output (p. 102).<br>Optionally, start and stop values can be defined to limit triggering to one position range and one particular direction of motion (negative or positive). |
| 2 | On Target | The on-target state of the axis selected is output at the selected trigger output (p. 104). |
| 5 | Motion Error | The selected digital output line becomes active when a motion error occurs (p. 104). The line stays active until the error code is reset to 0 (by a query with `ERR?`). |
| 6 | In Motion | The selected digital output line is active as long as the selected axis is in motion (p. 105). |
| 7 | Position+Offset | The first trigger pulse is output when the axis has reached a specified position. The next trigger pulses are output respectively when the axis position equals the sum of the last valid trigger position and a specified distance. The trigger output is stopped when a stop value is reached. The polarity sign of the distance value determines the direction of motion in which trigger pulses are to be output. Refer to "Configuring the "Position + Offset" Trigger Mode" (p. 105) for details. |
| 8 | Single Position | The selected digital output line is active when the axis position has reached or exceeded a given position (p. 106). |

| <Value> | Trigger mode | Short description |
|---|---|---|
| 9 | HardwareTrigger | Basically corresponds to the Position+Offset trigger mode but is executed by the FPGA circuit of the C-867 (shorter processing time). Refer to "Configuring the "HardwareTrigger" Trigger Mode" (p. 107) for further details. |

In addition, the polarity (active high / active low) of the signal at the digital output can be set (p. 109).

***INFORMATION***

The settings for the configuration of the digital output lines can only be modified in the volatile memory of the C-867. After the C-867 has been switched on or rebooted, factory default settings are enabled, provided a configuration has not already been carried out with a startup macro.

## 7.4.2 Configuring the "Position Distance" Trigger Mode

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

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

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

**Example:**

A pulse on digital output line 1 is output every time the axis 1 of the positioner has covered a distance of 0.1 µm.

➢ Send:

```
CTO 1 2 1
CTO 1 3 0
CTO 1 1 0.0001
TRO 1 1
```

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

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

**INFORMATION**

If start and stop values have the same value, they are ignored.
If the direction of motion is reversed before the axis position has reached the stop value, trigger pulses continue to be output.

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

**Example**

A pulse on digital output line 1 is output every time the axis 1 of the positioner has covered a distance of 0.1 µm, as long as axis 1 is moving in positive direction of motion within the range of 0.2 µm to 0.55 µm (start value < stop value).

➢ Send:

```
CTO 1 2 1
CTO 1 3 0
CTO 1 1 0.0001
CTO 1 8 0.0002
CTO 1 9 0.00055
TRO 1 1
```

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

The above example is presented with interchanged start and stop values in the following. Triggering occurs in negative motion direction of the axis (stop value < start value) in the range between 0.55 µm and 0.2 µm.

**Example:**

➢ Send:

```
CTO 1 2 1
```

```
CTO 1 3 0
CTO 1 1 0.0001
CTO 1 8 0.00055
CTO 1 9 0.0002
TRO 1 1
```

### 7.4.3 Configuring the "On Target" Trigger Mode

The on-target state of the axis selected (p. 44) is output at the selected trigger output in *On Target* trigger mode.

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

**Example:**

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

- Send:

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

### 7.4.4 Configuring the "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 one of the connected axes. The line stays active until the error code is reset to 0 (by an ERR? query).

***INFORMATION***

A motion error occurs when the current position differs too much from the commanded position during motion.
For further information, see "Motion Error" (p. 93).

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

### 7.4.5 Configuring the "In Motion" Trigger Mode

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

The motion state can also be read with the `#5` (p. 152), `#4` (p. 151), and `SRG?` (p. 228) commands.

***INFORMATION***

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

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

**Example:**

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

➢ Send:

`CTO 1 2 1`

`CTO 1 3 6`

`TRO 1 1`

### 7.4.6 Configuring the "Position + Offset" Trigger Mode

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

The pulse width is one servo cycle.

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

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
   - Send `CTO <TrigOutID> 2 A`, where *A* indicates the axis to be moved.
   - Send `CTO <TrigOutID> 3 7`, where 7 specifies the *Position+Offset* trigger mode.
   - Send `CTO <TrigOutID> 1 S`, where *S* indicates the distance.

   - Send `CTO <TrigOutID> 10 TriPos`, where *TriPos* indicates the position for the output of the first trigger pulse.
   - Send `CTO <TrigOutID> 9 Stop`, where *Stop* indicates the stop value.
2. If you want to activate the conditions for trigger output, send `TRO <TrigOutID> 1`.

**Example 1:**

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

### 7.4.7 Configuring the "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 of the parameters 0xE and 0xF. Default is mm.

1. Configure the digital output line (<TrigOutID>) to be used as the trigger output:
   - Send `CTO <TrigOutID> 2 A`, where *A* indicates the axis to be moved.

- Send `CTO <TrigOutID> 3 8`, where 8 specifies the *Single Position* trigger mode.
- Send `CTO <TrigOutID> 10 TriPos`, where *TriPos* indicates the position at which the output line is to become active.

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

**Example:**

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

➢ Send:

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

### 7.4.8 Setting up the "HardwareTrigger" Trigger Mode

The *HardwareTrigger* mode basically corresponds to the *Position+Offset* trigger mode (p. 105) but is executed by the FPGA circuit of the C-867 (shorter processing time).

***INFORMATION***

The *HardwareTrigger* mode can only be used for the C-867 when the encoder of the connected mechanics supplies A/B signals.
The *HardwareTrigger* mode does not function with the C-867 in conjunction with other signal types.

The first trigger pulse is output when the axis has reached a specified position (TriggerPosition). The next trigger pulses are output respectively when the axis position equals the sum of the last valid trigger position and a specified distance (TriggerStep). The trigger output is stopped when a stop value is reached. The polarity sign of the distance value determines the direction of motion in which trigger pulses are to be output. A specified factor n (PulseWidth) determines the pulse width as follows:

Pulse width = n * 33.3 ns

For the *HardwareTrigger* trigger mode, there is a fixed assignment of the axes to the digital output lines: axis 1 to line 1, axis 2 to line 2.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
   - Send `CTO <TrigOutID> 2 A`, where *A* indicates the axis to be moved.
   - Send `CTO <TrigOutID> 3 9`, where *9* indicates the HardwareTrigger trigger mode.
   - Send `CTO <TrigOutID> 1 S`, where *S* indicates the distance.

- Send `CTO <TrigOutID> 10 TriPos`, where *TriPos* indicates the position for the output of the first trigger pulse.
- Send `CTO <TrigOutID> 9 Stop`, where *Stop* indicates the stop value.
- Send `CTO <TrigOutID> 11 n`, where *n* indicates the factor for calculating the pulse width.

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 0.5 µm in the positive direction. The last trigger pulse is to be output if the absolute axis position is 2.5 mm. The pulse width should be approximately 0.8 µs.

➢ Send:

```
CTO 1 2 1
CTO 1 3 9
CTO 1 1 0.0005
CTO 1 10 1.5
CTO 1 9 2.5
CTO 1 11 24
TRO 1 1
```

**Example 2:**

The first trigger pulse is to be output on digital output line 1 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. The pulse width should be approximately 0.166 µs.

➢ Send:

```
CTO 1 2 B
CTO 1 3 9
CTO 1 1 -0.001
CTO 1 10 0.4
CTO 1 9 0.1
CTO 1 11 5
TRO 1 1
```

### *INFORMATION*

The velocity setting of the axis must be appropriate for the distance setting (TriggerStep) commanded by the `CTO` command. Recommended value:

Maximum velocity = distance * 20 kHz / 2

where 20 kHz is the servo cycle frequency of the C-867.

### 7.4.9 Setting Signal Polarity

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

- active high = 1 (default setting)
- active low = 0
- Configure the digital output line (<TrigOutID>) to be used as the trigger output:
  - Send `CTO <TrigOutID> 7 P`, where *P* indicates the polarity.

Example:

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

- Send:

  `CTO 1 7 0`

## 7.5 Digital Input Signals

The digital inputs of the C-867 are available on the **I/O** socket (p. 316).

- Get the number of the input lines available on the C-867 with the `TIO?` command (p. 241).
- Get the state of the input lines with the `DIO?` command (p. 170).

Potential applications:

- Use in macros (p. 111). Details and examples of macros can be found in "Controller Macros" (p. 126).
- Use as switch signals (p. 111)

***INFORMATION***

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

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

### 7.5.1 Commands and Parameters for Digital Inputs

#### Commands

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

| Command | Syntax | Function |
|---|---|---|
| `CPY` | CPY <Variable> <CMD?> | Copies the state of a digital input line to a variable when used in conjunction with the DIO? query command. Use in macros to set local variables (p. 146). |

| Command | Syntax | Function |
|---|---|---|
| DIO? | DIO? [{<DIOID>}] | Gets the state of the digital input lines. |
| FED | FED {<AxisID> <EdgeID> <Param>} | Starts a move to a signal edge. The signal source can be a digital input line. |
| FRF | FRF [{<AxisID>}] | Starts a reference move to the reference switch. A digital input line can be used as the source of the reference switch signal instead of the reference switch. |
| JRC | JRC <Jump> <CMD?> <OP> <Value> | Can only be used in macros. Triggers a relative jump of the macro run pointer depending on the state of a digital input line when used in conjunction with the DIO? query command. |
| MEX | MEX <CMD?> <OP> <Value> | Can only be used in macros. Stops running of the macro depending on the state of a digital input line when used in conjunction with the DIO? query command. |
| WAC | WAC <CMD?> <OP> <Value> | Can only be used in macros. Waits until a digital input line reaches a certain state when used in conjunction with the DIO? query command. |

### Parameters

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

| Parameters | Description and Possible Values |
|---|---|
| ***Source Of Reference Signal***<br>0x5C | Specifies the source of the reference signal for the FRF and FED commands:<br>0 = Reference switch<br>1 = Digital input 1<br>2 = Digital input 2<br>3 = Digital input 3<br>4 = Digital input 4 |
| ***Source Of Negative Limit Signal***<br>0x5D | Specifies the source(s) of the negative limit switch signal for the FRF (with parameter 0x70 = 5) and FED commands via a bitmask:<br>0 = Negative limit switch (default setting)<br>1 = Digital input 1 (bit 0)<br>2 = Digital input 2 (bit 1)<br>4 = Digital input 3 (bit 2)<br>8 = Digital input 4 (bit 3) |
| ***Source Of Positive Limit Signal***<br>0x5E | Specifies the source(s) of the positive limit switch signal for the FRF (with parameter 0x70 = 6) and FED commands via a bitmask:<br>0 = Positive limit switch (default setting)<br>1 = Digital input 1 (bit 0)<br>2 = Digital input 2 (bit 1)<br>4 = Digital input 3 (bit 2)<br>8 = Digital input 4 (bit 3) |

| Parameters | Description and Possible Values |
|---|---|
| ***Invert Digital Input Used For Negative Limit***<br>0x5F | Inverts the polarity of the digital inputs, which are used for the source of the negative limit switch signal, via a bitmask:<br>0 = No digital input inverted (default setting).<br>1 = Digital input 1 inverted (bit 0)<br>2 = Digital input 2 inverted (bit 1)<br>4 = Digital input 3 inverted (bit 2)<br>8 = Digital input 4 inverted (bit 3) |
| ***Invert Digital Input Used For Positive Limit***<br>0x60 | Inverts the polarity of the digital inputs, which are used for the source of the positive limit switch signal, via a bitmask:<br>0 = No digital input inverted (default setting).<br>1 = Digital input 1 inverted (bit 0)<br>2 = Digital input 2 inverted (bit 1)<br>4 = Digital input 3 inverted (bit 2)<br>8 = Digital input 4 inverted (bit 3) |

## 7.5.2 Using Digital Input Signals in Macros

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

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

Further information and examples can be found in "Controller Macros" (p. 126).

### *INFORMATION*

You can connect the C-170.PB pushbutton box from PI to the **I/O** socket (p. 316) to generate the digital input signals for use in macros. It also displays the state of the digital output lines via LEDs.

## 7.5.3 Using Digital Input Signals as Switch Signals

The digital inputs on the **I/O** socket can be used as the source of reference point and limit switch signals (e.g., for reference moves (p. 50)) for an axis.

### Using digital input as reference signal

#### *INFORMATION*

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

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

#### *INFORMATION*

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

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

Detailed information on changing parameters can be found in "Adapting Settings" (p. 273).

### Using digital inputs as source of the limit switch signals

#### *INFORMATION*

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

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

#### *INFORMATION*

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

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

#### *INFORMATION*

The ***Has No Limit Switches?*** parameter (ID 0x32) determines whether the C-867 evaluates the signals from the internal limit switches of the positioner. This parameter has no influence on the use of digital input lines as the source of the limit switch signal.

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

Detailed information on changing parameters can be found in "Adapting Settings" (p. 273).

**Example:**

Digital input lines 1, 3, and 4 are to be used for axis 1 as the sources of the positive limit switch signal. In addition, the signal polarity of lines 1 and 3 is to be inverted for axis 1. All adaptations are made in the volatile memory of the C-867 only.

- Send:

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

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

## 7.6 Analog Input Signals

The analog inputs of the C-867 are available on the **I/O** socket (p. 316).

- Get the number of the analog input lines available on the C-867 with the `TAC?` command (p. 233).
- Query the voltage on the analog inputs with the `TAV?` command (p. 233).
- Use the data recorder (p. 98) to record the analog input signals.

Potential applications:

- Use in macros (p. 114): Details and examples of macros are found in "Controller Macros" (p. 126).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

***INFORMATION***

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

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

### 7.6.1 Commands for Analog Inputs

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

| Command | Syntax | Function |
|---|---|---|
| `CPY` | CPY <Variable> <CMD?> | Copies the voltage value of an analog input line to a variable when used in combination with the TAV? query command. Use in macros to set local variables (p. 146). |
| `DRC` | DRC {<RecTableID> <Source> <RecOption>} | Configures the data recorder. Analog input values can be recorded using record option 81. |

| Command | Syntax | Function |
|---|---|---|
| JRC | JRC <Jump> <CMD?> <OP> <Value> | Can only be used in macros. Triggers a relative jump of the pointer when running the macro depending on the voltage at an analog input line when used in conjunction with the TAV? query command. |
| MEX | MEX <CMD?> <OP> <Value> | Can only be used in macros. Stops running of the macro depending on the voltage at an analog input line when used in conjunction with the TAV? query command. |
| TAC? | TAC? | Get the number of installed analog lines. |
| TAV? | TAV? [{<AnalogInputID>}] | Get voltage at analog input. |
| WAC | WAC <CMD?> <OP> <Value> | Can only be used in macros. Waits until an analog input line reaches a certain voltage when used in conjunction with the TAV? query command. |

### 7.6.2 Using Analog Input Signals in Macros

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

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

Further information and examples can be found in "Controller Macros" (p. 126).

## 7.7 Controlling with HID

### 7.7.1 Functionality of HID Control

Axes of HIDs can control the following motion variables of the positioner axes connected to the C-867:

- **Absolute target position**

  The relationship between the displacement of the axis of the HID and the motion variable of the positioner axis is created by the C-867 using a lookup table. The values in the lookup table are factors that are applied to the motion parameter to be controlled during HID control. The value range is between -1.0000 and 1.0000.
  The firmware of the controller provides two predefined lookup table types to choose from (linear and parabolic) and allows customized lookup tables to be filled with individual values. For details, see the descriptions of the HDT (p. 183) and HIT (p. 193) commands.

- **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 value that corresponds to the current displacement of the axis of the HID and the currently valid maximum velocity of the controller axis.

- **Maximum velocity**

  Product of the lookup table value that corresponds to the current displacement of the axis of the HID and the value of the ***Closed-Loop Velocity For HI Control*** parameter (0x74).

For further details, see the description of the HIA (p. 185) command.

During HID control, the target position of the controlled axis of the C-867 is set to the soft limit that is specified by the parameter 0x15 or 0x30. Details on the parameters can be found in "Travel Range and Soft Limits (p. 47)". When HID control is deactivated, the target position is set to the current position of the controlled axis.

***INFORMATION***

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

### Programming HID control

HID buttons and LEDs (i.e., output units) can for example, be used in controller macros (p. 126) to program HID control.

In this manual, you will find an example macro for HID control with relative motion.

## 7.7.2 Commands and parameters for HIDs

### Commands

The following commands are available for using HIDs:

| Command | Syntax | Function |
|---|---|---|
| `HDT` | HDT {<HIDeviceID> <HIDeviceAxis> <HIDTableID>} | Assigns a lookup table to an HID axis.<br>The assignment can be saved in the nonvolatile memory with WPA. |
| `HDT?` | HDT? [{<HIDeviceID> <HIDeviceAxis>}] | Queries the current assignment of lookup tables to HID axes. |
| `HIA` | HIA {<AxisID> <MotionParam> <HIDeviceID> <HIDeviceAxis>} | Configures control of the C-867's axes by HID axes ("HID Control").<br>The configuration can be saved in the nonvolatile memory with WPA. |

| Command | Syntax | Function |
|---|---|---|
| `HIA?` | HIA? [{<AxisID> <MotionParam>}] | Queries the current configuration of HID control. |
| `HIB?` | HIB? [{<HIDeviceID> <HIDeviceButton>}] | Queries the current state of HID buttons. |
| `HIE?` | HIE? [{<HIDeviceID> <HIDeviceAxis>}] | Queries the current displacement HID axes. |
| `HIN` | HIN {<AxisID> <HIDControlState>} | Activates or deactivates the HID control for the axes of the C-867. |
| `HIN?` | HIN? [{<AxisID>}] | Queries the HID control activation status. |
| `HIS` | HIS {<HIDeviceID> <HIDItemID> <HIDPropID> <HIDPropValue>} | Configures the specified HID.<br>For the C-867, the functionality of HIS corresponds to that of the HIL command. |
| `HIS?` | HIS? [{<HIDeviceID> <HIDItemID> <HIDPropID>}] | Queries the properties of HID operating elements. |
| `HIT` | HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>} | Fills lookup tables with values.<br>The table contents can be saved in the nonvolatile memory with WPA. |
| `HIT?` | HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]] | Queries the values of the points in the lookup tables. |
| `SST` | SST {<AxisID> <StepSize>} | Is only used when the relative target position is set as the motion variable to be controlled.<br>Sets the distance to be covered per impulse received. |
| `SST?` | SST? [{<AxisID>}] | Queries the distance set with SST. |

***INFORMATION***

The DPA command resets the settings that were made with HDT, HIA, and HIT to default settings in the volatile memory **and** the nonvolatile memory.

### Parameters

The following parameters are available for the HID control:

| Parameters | Description and Possible Values |
|---|---|
| ***Invert Direction Of Motion For Joystick-Controlled Axis?***<br>0x61 | Specifies the direction of motion for HID-controlled axes of the C-867.<br>0 = direction of motion not inverted (default setting)<br>1 = direction of motion inverted |

| Parameters | Description and Possible Values |
|---|---|
| ***Closed-Loop Velocity For HI Control (Phys. Unit/s)***<br>0x74 | Maximum velocity during HID control<br>Limited by parameter 0xA.<br>If parameter 0x74 has the value zero, the value of the parameter 0x49 is used during the HID control. |
| ***Closed-Loop Acceleration For HI Control (Phys. Unit/s2)***<br>0x75 | Maximum acceleration during HID control<br>Limited by parameter 0x4A. |
| ***Closed-Loop Deceleration For HI Control (Phys. Unit/s2)***<br>0x76 | Maximum deceleration during HID control<br>Limited by parameter 0x4B. |

### 7.7.3 Testing the HID

We recommend testing the HID operating elements in PIMikroMove after connecting it to the C-867 .

***INFORMATION***

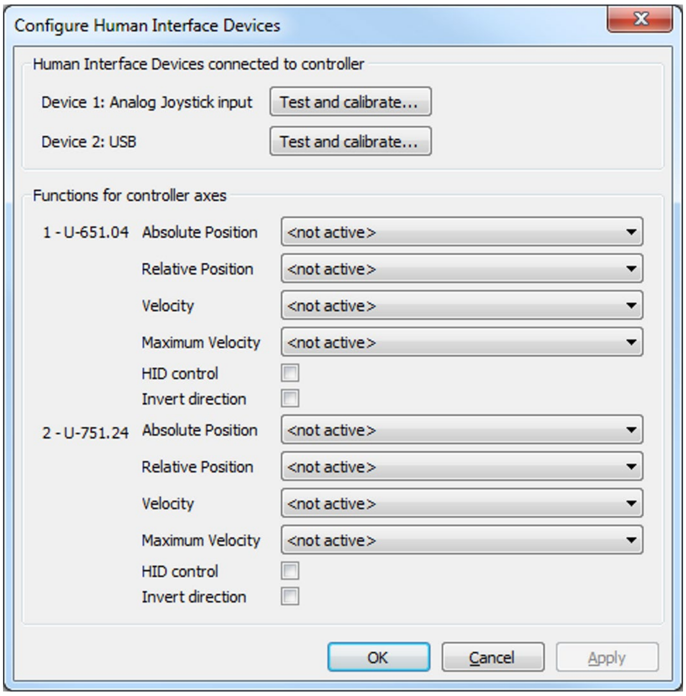
A positioner does not have to be connected to the C-867 to test HID operating elements in PIMikroMove.

**Requirements**

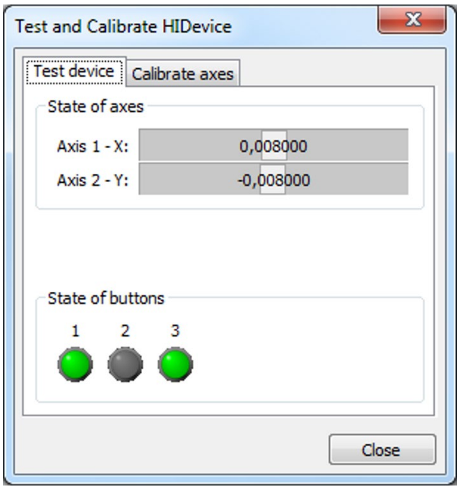
- ✓ You have read and understood the General Notes on Startup (p. 69).
- ✓ PIMikroMove is installed on the PC (p. 64).
- ✓ PIMikroMove has established communication between the C-867 and the PC (p. 70).
- ✓ You have connected (p. 60) the HID to the C-867.

**Testing the HID in PIMikroMove**

1. Open the ***Configure Human Interface Devices*** window in the main window of PIMikroMove via the ***C-867 > Configure controller HIDevice(s)…*** menu item.




2. Open the ***Test and Calibrate HIDevice*** window by clicking the ***Test and calibrate...*** button.
   - for ***Device 1: Analog Joystick Input*** for configuring the analog HID connected to the C-867;
   - for ***Device 2: USB*** for configuring the digital HID connected to the C-867.
   - for ***Device x: USB*** for configuring a digital HID connected to the C-867 via a USB hub.
3. Select the ***Test device*** tab in the ***Test and Calibrate HIDevice*** window.
4. Test the HID's operating elements:
   - Move the HID axis and watch the status indicators in the ***State of axes*** area.
   - Press the HID's buttons and at the same time, watch the status indicators in the ***State of buttons*** area.



In this example, a digital joystick with 2 axes and 3 buttons is connected to a C-867. The C-867 supports both axes of the joystick. The identifier of the X axis is 1, and that of the Y axis is 2. The three buttons of the joystick are available via the identifiers 1, 2, and 3. Current status in the figure: The X axis of the joystick is displaced in the positive direction, the Y axis is displaced in the negative direction, and buttons 1 and 3 are pressed.

### 7.7.4 Configuring and Enabling HID Control

It is recommended to use PIMikroMove for setting up and enabling HID control. We recommend testing the connected HID (p. 117) before activating HID control.

***INFORMATION***

It is not possible to simultaneously control the absolute and relative target position of an axis of the C-867 via the axes of the HID.

➢ Configure the HID control for an axis **either** for the absolute **or** the relative target position.

**Requirements**

✓ You have carried out a successful reference move for each axis of the C-867 with PIMikroMove; see "Starting Motion" (p. 80).

✓ You have connected (p. 60) the HID to the C-867.

✓ All devices are still ready for operation.

**Setting up and activating HID control in PIMikroMove**

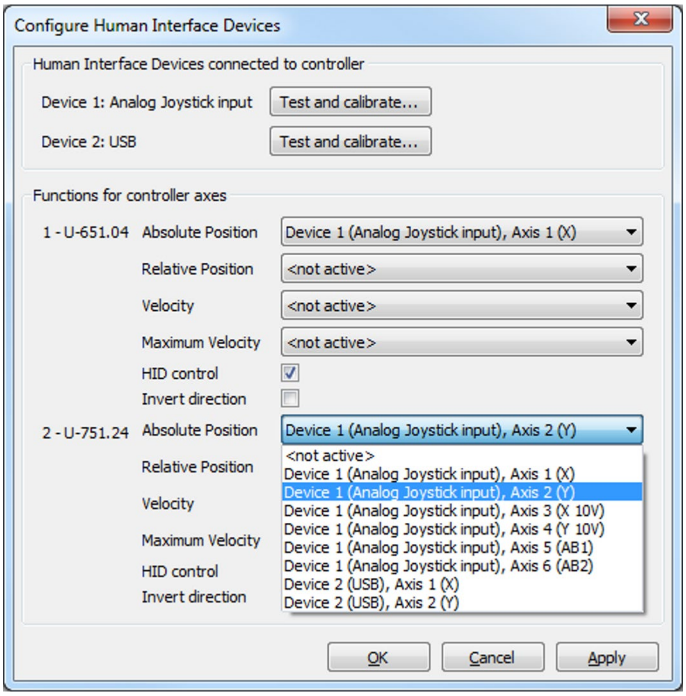
1. Open the ***Configure Human Interface Devices*** window in the main window of PIMikroMove via the ***C-867 > Configure controller HIDevice(s)…*** menu item.
2. Set up the HID control for the axes of the C-867 in the ***Functions for controller axes*** area:

   a) Select the HID axis in the corresponding field to be used for the motion variable to be controlled. The following motion variables 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*** - velocity for axis motion
   - ***Maximum Velocity*** - maximum velocity for axis motion

   b) Activate HID control by clicking in 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 two axes of the controller are controlled via the X and Y axes of HID 1 (analog joystick).

3. Send the settings for setting up HID control to the C-867 by clicking the ***OK*** button.

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

4. In PIMikroMove, make sure that servo mode is switched on for the axes of the C-867 (e.g., by clicking the ***Servo*** checkbox in the ***Axes*** tab in PIMikroMove's main window).

   The axes of the C-867 can now be controlled by the HID according to the settings made in step 2.

If HID control of the absolute target position does not work satisfactorily:

➢ Follow the instructions in "Calibrating HID Axes" (p. 120).

If you want to save the assignment of the axis of the HID to a motion variable in the nonvolatile memory of the C-867:

➢ Follow the instructions in "Saving the Configuration of HID Control Permanently" (p. 123).

## 7.7.5 Calibrating HID Axes

Calibration involves the following steps:

1. If there are corresponding operating elements on the HID: Mechanical adjustment of the axis.
2. Calibrating the HID axis in PIMikroMove

When the axes of HIDs are calibrated in PIMikroMove, the lookup table to be used is selected and is filled with custom values if necessary. To do this, a positioner does not need to be connected to the C-867.

### *INFORMATION*

The parabolic lookup table allows for greater sensitivity when moving slowly.

### Requirements

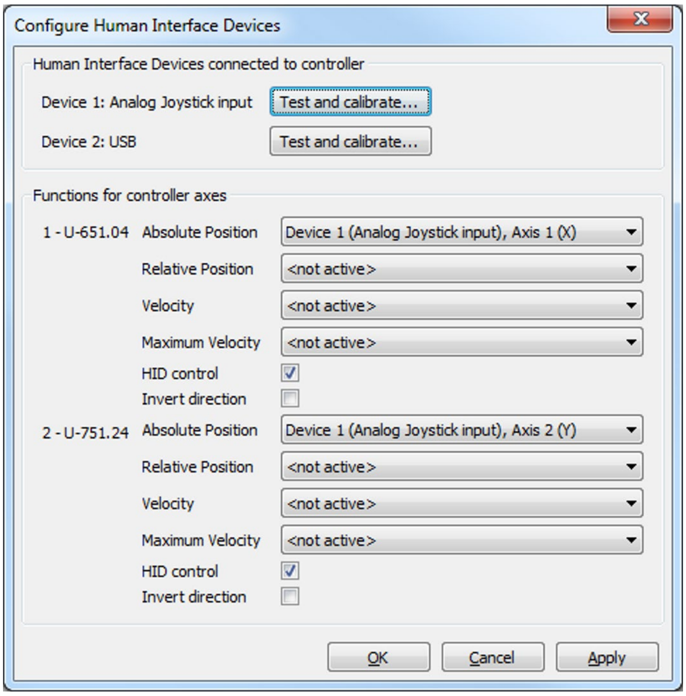
- ✓ You have set up and activated HID control in PIMikroMove .
- ✓ All devices are still ready for operation.

### Adjusting an HID axis manually

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

### Calibrating the axis of an HID in PIMikroMove

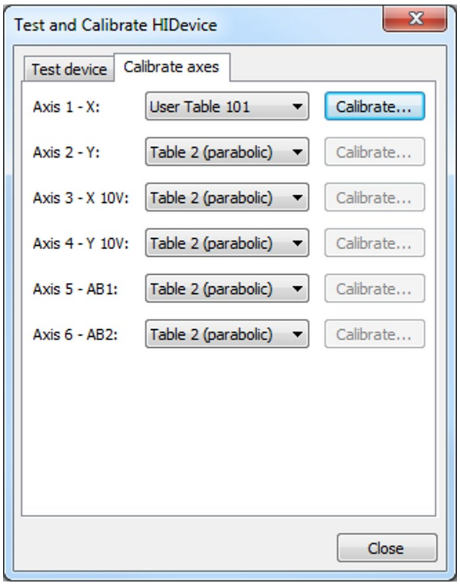
1. Open the ***Configure Human Interface Devices*** window in the main window of PIMikroMove via the ***C-867 > Configure controller HIDevice(s)…*** menu item.




   The figure shows an example in which an analog joystick is connected to the C-867. Both axes of the C-867 are controlled by axis 1 (X) and 2 (Y) of the joystick; the respective absolute target position is set as the variable to be controlled.

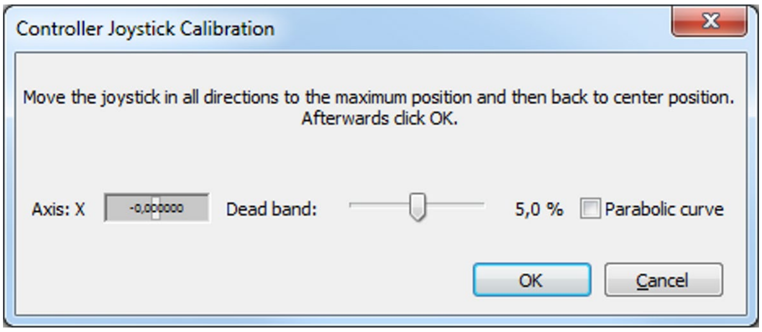
2. Open the ***Test and Calibrate HIDevice*** window by clicking the ***Test and calibrate...*** button.

3. Select the ***Calibrate axes*** tab in the ***Test and Calibrate HIDevice*** window.
4. Assign a lookup table with the ***User Table*** designation to the HID axis to be calibrated in the corresponding selection box.




In the example in the figure, a user-defined lookup table was assigned to axis 1 of the connected joystick.

5. Calibrate the HID axis by filling the assigned user-defined lookup table with values:
   a) Click the corresponding ***Calibrate...*** button to open the ***Controller Joystick Calibration*** window.
   b) Move the 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 in the ***Controller Joystick Calibration*** window accordingly.
   e) If the values in the user-defined lookup table are to describe a parabolic waveform, check the ***Parabolic curve*** box in the ***Controller Joystick Calibration*** box.
   f) Click ***OK*** in the ***Controller Joystick Calibration*** window to write the lookup table values to the volatile memory of the C-867. You can watch the writing process in a separate window.



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

6. If you want to save the assignment of the lookup tables to the HID axes and the contents of user-defined lookup tables in the C-867's nonvolatile memory:
   a) Close the ***Test and Calibrate HIDevice*** window.
   b) If necessary, adapt the settings to your application in the ***Configure Human Interface Devices*** window; refer to "Configuring and Activating HID Control" (p. 119).
   c) If necessary, click the ***Apply*** button to activate the settings in the ***Configure Human Interface Devices*** window.
   d) Close the ***Configure Human Interface Devices*** window.
   e) Follow the instructions in "Saving the Configuration of HID Control Permanently" (p. 123).

### 7.7.6 Saving the Configuration of HID Control Permanently

The following settings for the configuration of HID control can be saved in the nonvolatile memory of the C-867:

- Assignment of lookup tables to the axes of the human interface device; see "Calibrating HID Axes" (p. 120)
- For contents of user-defined lookup tables, see "Calibrating HID Axes" (p. 120)
- To assign the HID axes to the motion variables to be controlled for the C-867's axis, see "Setting up and Activating HID Control" (p. 119)

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

***INFORMATION***

The values in the nonvolatile memory are loaded to the volatile memory when switching on or rebooting the C-867 and take effect immediately.

#### Requirements

- ✓ You have read and understood the General Notes on Startup (p. 69).
- ✓ PIMikroMove is installed on the PC (p. 64).
- ✓ You have read and understood the PIMikroMove manual. The manual is on the data storage device for the product.
- ✓ PIMikroMove has established communication between the C-867 and the PC (p. 70).

#### Saving the configuration of the HID control permanently in PIMikroMove

If you want to write the current settings for the configuration of HID control to the nonvolatile memory of the C-867:

1. Select the ***C-867 > Save parameters to nonvolatile memory*** menu item in the main window of PIMikroMove. The ***Save Parameters to Non-Volatile Memory*** dialog opens.

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

***INFORMATION***

The settings for the configuration of the HID control are also written to the nonvolatile memory of the C-867 if you select the *All Parameters, Settings of HDT, HIA, HIT (100)* or enter the password *100*. However, the entry or the password 100 also saves the current values of all parameters of the C-867, see the description for the WPA (p. 248) command and "Adapting Settings" (p. 273).

***INFORMATION***

The DPA command resets the settings that were made with HDT, HIA, and HIT to default settings in the volatile memory **and** the nonvolatile memory.

## 7.7.7 Available HIDs

PI offers the HIDs described in the following as optional accessories (p. 12).

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

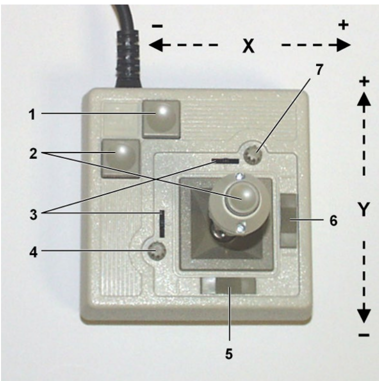



Figure 15: C-819.20 joystick

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

## C-819.30 analog joystick, 3 axes

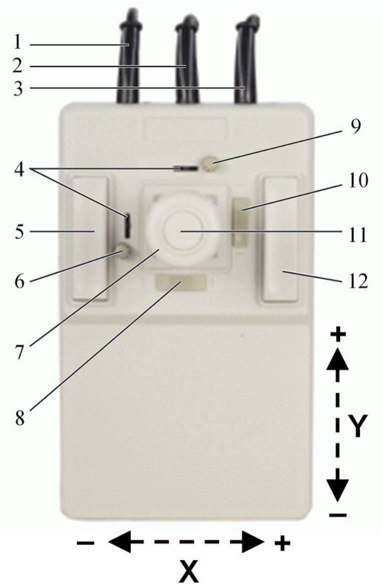



Figure 16: C-819.30 joystick

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

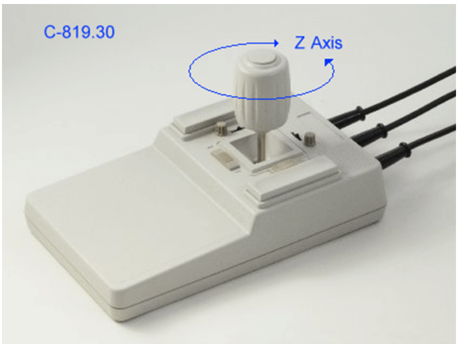



Figure 17: C-819.30 joystick, rotary knob for the Z axis

**C-819.JD: Digital joystick for 2 axes**

Operating elements:

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

# 7.8 Controller Macros

## 7.8.1 Overview: Macro Functionality and Example Macros

The C-867 can save and process command sequences as macros.

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

- Several macros can be saved at the same time.
- Any macro can be defined as the startup macro. The startup macro runs each time the C-867 is switched on or rebooted.
- Processing and stopping a macro can be linked to conditions. This makes loops possible.
- Macros can call up themselves or other macros.
- Variables (p. 146) can be set for the macro and in the macro itself and used in different operations.
- Input signals can be evaluated for conditions and variables.

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

- Moving an axis back and forth (p. 130)
- Recording a macro for a controller whose address is different from 1 (p. 131)
- Moving an axis with a variable travel back and forth (p. 132)
- Implementing multiple calls of a macro via a loop (p. 133)
- Preparing an axis via startup macro for closed-loop operation (p. 135)
- Synchronization of two controllers (p. 136)
- Stopping motion by pushbutton (p. 137)
- HID control with storage of positions

## 7.8.2 Commands and Parameters for Macros

### Commands

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

| Command | Syntax | Function |
|---|---|---|
| ADD (p. 156) | ADD <Variable> <FLOAT1> <FLOAT2> | Adds two values and saves the result to a variable (p. 146). Can only be used for local variables in macros. |
| CPY (p. 159) | CPY <Variable> <CMD?> | Copies a command response to a variable (p. 146). Can only be used for local variables in macros. |
| DEL (p. 166) | DEL <uint> | Can only be used in macros. Delays <uint> milliseconds. |
| JRC (p. 205) | JRC <Jump> <CMD?> <OP> <Value> | Can only be used in macros. Triggers a relative jump of the macro execution pointer depending on a condition. |
| MAC (p. 207) | MAC BEG <macro name> | Starts the recording of a macro with the name *macro name* on the controller. *macro name* can consist of up to 8 characters. |
| | MAC DEF <macro name> | Defines the specified macro as the startup macro. |
| | MAC DEF? | Gets the startup macro. |
| | MAC DEL <macro name> | Deletes the specified macro. |
| | MAC END | Ends the macro recording. |
| | MAC ERR? | Reports the last error that occurred while the macro was running. |
| | MAC FREE? | Gets the free memory space for macro recording. |
| | MAC NSTART <macro name> <uint> [<String1> [<String2>]] | Starts the specified macro n times in succession (n = number of executions). The values of local variables can be set for the macro with <String1> and <String2>. |
| | MAC START <macro name> [<String1> [<String2>]] | Runs the specified macro. The values of local variables can be set for the macro with <String1> and <String2>. |
| MAC? (p. 209) | MAC? [<macro name>] | Lists all macros or the content of a specified macro. |
| MAT (p. 211) | MAT <Variable> "=" <Float1> <OP> <Float2> | Carries out a mathematical operation or bit operation and saves the result as a variable (p. 146). Can only be used for local variables in macros. |
| MEX (p. 212) | MEX <CMD?> <OP> <Value> | Can only be used in macros. Stops the macro execution depending on a condition. |
| RMC? (p. 218) | RMC? | Lists macros which are currently running. |

| Command | Syntax | Function |
|---|---|---|
| VAR (p. 244) | VAR <Variable> <String> | Sets a variable (p. 146) to a certain value or deletes it. Can only be used for local variables in macros. |
| VAR? (p. 245) | VAR? [{<Variable>}] | Gets variable values. |
| WAC (p. 247) | WAC <CMD?> <OP> <Value> | Can only be used in macros. Waits until a condition is met. |
| #8 (p. 153) | - | Tests if a macro is running on the controller. |

#### Parameters

The following parameter is available for working with macros:

| Parameters | Description and Possible Values |
|---|---|
| ***Ignore Macro Error?*** 0x72 | Determines whether the controller macro is stopped if an error occurs when it is running.<br>▪ 0 = Stop macro when error occurs (default)<br>▪ 1 = Ignore error |

### 7.8.3 Working with Macros

Work with macros comprises the following:

- Recording macros (p. 128)
- Starting macros (p. 131)
- Stopping macros (p. 134)
- Configuring a startup macro (p. 134)
- Deleting macros (p. 135)

***INFORMATION***

It is recommended to use the ***Controller macros*** tab in PIMikroMove when working with controller macros. There you can record, start, and manage controller macros easily. Refer to the PIMikroMove manual for details.

#### Recording a macro

***INFORMATION***

The C-867 can save up to 32 macros simultaneously. A maximum of 5 nesting levels are possible in macros.

### *INFORMATION*

Basically all GCS commands (p. 143) can be included in a macro. Exceptions:

- `RBT` for rebooting the C-867
- `MAC BEG` and `MAC END` for macro recording
- `MAC DEL` for deleting a macro

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

### *INFORMATION*

If you record a macro on a C-867 whose controller address is different to 1, note the following when entering the commands that are to be an element of the macro:

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

### *INFORMATION*

To make the use of macros more flexible, you can use local and global variables in macros. For further information, see "Variables" (p. 146).

### *INFORMATION*

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

- ➢ Only record macros if it is necessary.
- ➢ Use variables (p. 146) in macros to make macros more flexible, and give the corresponding variable values when starting macro execution.
- ➢ Contact our customer service department (p. 309) if the C-867 shows unexpected behavior.

### *INFORMATION*

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

1. Start the macro recording.
   - If you are working with PITerminal or in the ***Command entry*** window of PIMikroMove: Send the `MAC BEG macroname` command, where *macro name* indicates the name of the macro.

- If you are working in PIMikroMove on the ***Controller macros*** tab: Click the ***Create new empty macro*** icon to create a tab for entering a new macro. Do **not** enter the `MAC BEG macroname` command.

2. Enter the commands to be included in the *macro name* macro line by line, using the normal command syntax.

   Macros can call up themselves or other macros in several nesting levels.

3. End the macro recording.

   - If you are working with PITerminal or in the ***Command entry*** window of PIMikroMove: Send the `MAC END` command.
   - If you are working in PIMikroMove on the ***Controller macros*** tab: Do **not** enter the `MAC END` command. Click the ***Send macro to controller*** icon and enter the name of the macro in a separate dialog window.

   The macro has been stored in the nonvolatile memory of the C-867.

4. If you want to check whether the macro has been correctly recorded:

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

   - Get which macros are saved in the C-867 by sending the `MAC?` command.
   - Get the contents of the *macro name* macro by sending the `MAC? macroname` command.

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

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

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

***INFORMATION***

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

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

➢ Record the macros by sending:

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

```
MAC BEG macro3
MAC START macro1
MAC START macro2
MAC END
```

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

***INFORMATION***

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

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

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

1. Record the macro by sending:

   ```
   2 MAC BEG ref
   2 SVO 1 1
   2 DEL 1000
   2 FRF 1
   2 MAC END
   ```

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

   ```
   2 MAC? ref
   ```

   The response reads:

   ```
   0 2 SVO 1 1
   DEL 1000
   FRF 1
   ```

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

### Starting a macro

***INFORMATION***

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

***INFORMATION***

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

***INFORMATION***

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

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

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

   Further information on changing parameters can be found in "Adapting Settings" (p. 273).
2. Start the macro:
   - If the macro is to be run once, send the `MAC START macroname string` command, whereby *macroname* indicates the name of the macro.
   - If the macro is to be run n times, send the `MAC NSTART macroname n string` command, whereby *macroname* indicates the name of the macro and *n* indicates the number of runs.

   *string* stands for the values of local variables. The values only have 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 local variable 1. The individual values must be separated from each other by spaces.
3. If you want to check whether the macro is running:
   - Query whether a macro is running on the controller by sending the `#8` command.
   - Query the name of the macro that is currently running on the controller by sending the `RMC?` command.

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

***INFORMATION***

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

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

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

```
VAR LEFT 5
VAR RIGHT 15
```

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

– Create the global variables again each time the C-867 is switched on or rebooted, since they are only written to the volatile memory of the C-867.

2. Record the MOVLR macro by sending:

```
MAC BEG movlr
MAC START movwai ${LEFT}
MAC START movwai ${RIGHT}
MAC END
```

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

3. Record the MOVWAI macro by sending:

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

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

4. Run the MOVLR macro by sending:

```
MAC NSTART movlr 5
```

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

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

***INFORMATION***

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

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

➢ Record the LOOPDION macro by sending:

```
MAC BEG loopdion
```

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

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

### Stopping a macro

***INFORMATION***

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

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

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

### Configuring a startup macro

Any macro can be defined as the startup macro. The startup macro is executed each time the C-867 is switched on or rebooted.

***INFORMATION***

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

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

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

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

**_INFORMATION_**

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

The STARTCL macro switches the HID control off and the servo mode on for axis 1 and starts a reference move. 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 1
MAC END
MAC DEF startcl
```

**_INFORMATION_**

When using this macro, the parameter settings of the C-867 should be adapted in the nonvolatile memory to the connected positioner. Alternatively, the parameter settings can also be configured in the volatile memory via the startup macro. For further information, see "Adapting Settings" (p. 273).

### Deleting a macro

**_INFORMATION_**

A macro cannot be deleted while it is running.

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

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

## 7.8.4 Making Backups and Loading Controller Macros

For example, making backups of controller macros on the PC can be useful before updating the firmware (p. 299).

***INFORMATION***

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

### Backing up controller macros onto the PC with PIMikroMove

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

   By selecting one or more macros, the  (*Save selected macros to PC*) button becomes active.
3. Save the selected macros on the PC:
   a) Click the  button to open a directory selection window.
   b) Select the directory on the PC where you want to save the macros.
   c) Click ***Save***.

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

### Loading controller macros from the PC to the C-867 with PIMikroMove

1. Select the ***Controller macros*** tab in the PIMikroMove main window.
2. Load macros from the PC to the C-867:
   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 C-867.
   c) Click ***Open***.

   For each selected text file (<macro name>.txt), the content is loaded as a macro <macro name> into the C-867.

### 7.8.5 Macro Example: Synchronization of Two Controllers

***INFORMATION***

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

| Action | Command | Result |
|---|---|---|
| Connect the digital output line 1 on the **I/O** socket of the master controller to digital input line 1 on the **I/O** socket of the slave controller. | -<br>Use a suitable cable. Pin assignment see "I/O" (p. 316). | The digital output signal of the master controller can be used as the trigger for the motion of the axis connected to the slave controller. |
| Set up the motion on the master controller and on the slave controller. | `SVO 1 1`<br>`FRF 1 1`<br>`VEL 1 0`<br>`MOV 1 5.5` | For both controllers: The servo mode is switched on and the axis has executed a reference move – here to the reference switch. The velocity is set to zero. The axis does not move for now as a result, even though the motion command for the move to absolute position 5.5 has already been sent. |
| Record the MASTER macro on the master controller. | `MAC BEG master`<br>`DIO 1 1`<br>`VEL 1 100`<br>`MAC END` | The macro has the following tasks:<br>▪ Switch the digital output line 1 of the master controller to high state to trigger the slave controller<br>▪ Set velocity to 100 to start the motion |
| Record the SLAVE macro on the slave controller. | `MAC BEG slave`<br>`WAC DIO? 1 = 1`<br>`VEL 1 100`<br>`MAC END` | The macro has the following tasks:<br>▪ Set condition: The macro continues only if digital input line 1 has the high state (i.e., if the master controller outputs the trigger signal).<br>▪ Set velocity to 100 to start the motion |
| Start the SLAVE macro on the slave controller. | `MAC START slave` | The axis on the slave controller is still not moving because the condition for further macro execution has not yet been met. |
| Start the MASTER macro on the master controller. | `MAC START master` | Both axes are moving because their velocity is now each different from zero. The motion occurs synchronously. |

### 7.8.6 Macro Example: Stopping Motion by Pushbutton

***INFORMATION***

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

***INFORMATION***

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

| Action | Command | Result |
|---|---|---|
| Connect digital input line 1 on the **I/O** socket to an appropriate signal source. | -<br>Pin assignment see "I/O" (p. 316). | For example,the digital input signal can be used for a conditional jump of the macro pointer. |
| Record the HALT macro on the controller. | `MAC BEG halt`<br>`MVR 1 5`<br>`JRC 2 DIO? 1 = 1`<br>`JRC -1 ONT? 1 = 0`<br>`HLT 1`<br>`MAC END` | The macro has the following tasks:<br>▪ Start relative motion of axis 1<br>▪ Set a condition: If digital input line 1 has the high state (when using the pushbutton box: button 1 is pressed), the macro execution pointer jumps two lines forward. This stops the axis. Otherwise macro execution is continued with the next line.<br>▪ Set condition: The macro execution pointer jumps back one line as long as axis 1 has not yet reached the target position. A loop is established as a result. |
| Run 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). Irrespective of whether the axis has reached the target position or was halted previously, the error code is set to 10 via the `HLT` command. |
| If error code 10 interferes: Record alternative HALTVAR macro which uses a variable. Details see "Variables" (p. 146). | `MAC BEG haltvar`<br>`MVR 1 5`<br>`JRC 2 DIO? 1 = 1`<br>`JRC -1 ONT? 1 = 0`<br>`CPY TARGET POS? 1`<br>`MOV 1 ${TARGET}`<br>`VAR TARGET`<br>`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. |
| 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. |

### 7.8.7 Macro Example: HID Control with Storage of Positions

**Task:**
The velocity of axis 1 is to be controlled with a HID. HID control should only be active when a certain button on the HID is pressed at the same time. By using the buttons of a connected pushbutton box, in addition up to four positions are to be stored in the controller or approached by the axis. The LEDs of the pushbutton box should indicate whether the controller is ready to save the current position and whether it has been saved.

**Approach:**
The STARTUP, MAINLOOP, TESTJOYB, TESTDION, and MVAX2ST macros are recorded on the controller. They use the global variables STORE1, STORE2, STORE3, STORE4, COUNTER, and the local variables 1 and 2.

***INFORMATION***

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

| **Action** | Command | **Result** |
|---|---|---|
| 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. The states of digital output lines 1 to 4 are indicated by the LEDs which are integrated in the buttons. |
| Connect the joystick to the **Analog Joystick** socket. | - | The connected joystick is accessible for commands as axis 1 of HID 1. The joystick button to be used is accessible as button 1 of HID 1. |
| Switch on servo mode for axis 1. | `SVO 1 1` | The servo mode must be switched on, so that axis 1 can be controlled via an axis of the HID. |
| Start reference move for axis 1 (stage has an incremental position sensor). | `FRF 1` | The axis starts a reference move – here to the reference switch. After this, absolute axis positions can be commanded. |
| Specify which motion variable of the axis is to be controlled via an axis of the HID. | `HIA 1 0 0 0`<br>`HIA 1 3 1 1` | Delete the current configuration of the HID control. Reconfigure afterwards: The velocity of axis 1 is controlled via axis 1 of HID 1. The HID control is not yet enabled. |
| Record the STARTUP macro on the controller. | `MAC BEG startup`<br>`CPY STORE1 POS? 1`<br>`CPY STORE2 POS? 1`<br>`CPY STORE3 POS? 1`<br>`CPY STORE4 POS? 1`<br>`MAC START MAINLOOP` | The macro has the following tasks:<br>▪ Initialize variables for storing the position<br>▪ Start MAINLOOP macro for the main loop |

| Action | Command | Result |
|---|---|---|
| | `MAC END` | |
| Record the MAINLOOP macro on the controller. | `MAC BEG mainloop`<br>`MAC START TESTJOYB`<br>`VAR COUNTER 1`<br>`MAC START TESTDION ${COUNTER}`<br>`ADD COUNTER ${COUNTER} 1`<br>`JRC -2 VAR? COUNTER < 5`<br>`MAC START MAINLOOP`<br>`MAC END` | The macro has the following tasks:<br>▪ Start TESTJOYB macro for HID control<br>▪ Start TESTDION macro successively for all digital inputs (i.e., every pushbutton box button), using a loop<br>▪ Call itself to set up the main loop |
| Record the TESTJOYB macro on the controller. | `MAC BEG testjoyb` | The macro has the following tasks: |
| | `MEX HIB? 1 1 = 0` | ▪ Stop macro if button 1 on HID 1 is no longer pressed |
| | `HIN 1 1` | ▪ Activate control for axis 1 via the HID |
| | `DIO 0 15` | ▪ Switch all LEDs on the pushbutton box on |
| | `JRC 6 HIB? 1 1 = 0` | ▪ Jump forward 6 lines (to HIN 1 0) if button 1 on HID 1 is no longer pressed |
| | `DEL 50` | ▪ Wait 50 ms |
| | `DIO 0 0` | ▪ Switch all LEDs on the pushbutton box off |
| | `JRC 3 HIB? 1 1 = 0` | ▪ Jump forward 3 lines (to HIN 1 0) if button 1 on HID 1 is no longer pressed |
| | `DEL 50` | ▪ Wait for an additional 50 ms |
| | `JRC -6 HIB? 1 1 = 1` | ▪ Jump back 6 lines (to DIO 0 15) if button 1 of HID 1 is still pressed |
| | `HIN 1 0` | ▪ Disable control via the HID for axis 1 |
| | `DIO 0 0` | ▪ Switch all LEDs on the pushbutton box off |
| | `MAC END` | |
| Record the TESTDION macro on the controller. | `MAC BEG testdion` | The macro has the following tasks: |
| | `MEX VAR? 0 != 1` | ▪ Stop running the macro if the number of local variables specified is not 1 when starting TESTDION |
| | `MEX DIO? $1 = 0` | ▪ Stop running the macro if the pushbutton box button specified via local variable 1 is no longer pressed (corresponding input line has the low |

| Action | Command | Result |
|---|---|---|
| | | state) |
| | `DEL 300` | ▪ Wait 300 ms |
| | `JRC 3 DIO? $1 = 1` | ▪ If the button is still pressed, jump 3 lines forward (to `DEL 400`) |
| | `MAC START MVAX2ST $1` | ▪ Start the MVAX2ST macro because the button was only briefly pressed. The value of the local variable 1 is also used for local variable 1 in MVAX2ST. MVAX2ST moves axis 1 to the position assigned for the button. |
| | `MEX DIO? $1 = 0` | ▪ Stop macro if button is no longer pressed |
| | `DEL 400` | ▪ Wait 400 ms |
| | `MEX DIO? $1 = 0` | ▪ Stop macro if button is no longer pressed |
| | `DIO $1 1` | ▪ Switch the pushbutton box LED on that is associated with the button pressed to indicate storing of the current position |
| | `WAC DIO? $1 = 0` | ▪ The macro continues to run only if the button is no longer pressed |
| | `DIO $1 0` | ▪ Switch LED off |
| | `CPY STORE$1 POS? 1` | ▪ Save the current position of axis 1 in the global variable designated via local variable 1 |
| | `MAC END` | |
| Record the MVAX2ST macro on the controller. | `MAC BEG MVAX2ST` | The macro has the following tasks: |
| | `CPY 2 VAR? STORE$1` | ▪ Queries the storage variable designated via local variable 1 and copies its value to local variable 2 |
| | `MOV 1 $2` | ▪ Move axis 1 to the target position specified via local variable 2 |
| | `MAC END` | |
| Run the STARTUP macro on the controller.<br>Alternative:<br>If the variables for storing positions are not to be initialized, start the MAINLOOP | `MAC START startup` | HID control is activated by pressing the button of the HID. When HID control is activated, the pushbutton box LEDs flash rapidly and therefore indicate that the box buttons should not be pressed. After releasing the button on the HID is released, HID control is deactivated and the LEDs switch off. The pushbutton box can now be used for moving to the saved |

| Action | Command | Result |
|---|---|---|
| macro on the controller instead. | | positions or for saving the current position.<br>The respective button on the pushbutton box is pressed briefly to move the positioner to a stored position.<br>To store the current position of the positioner, a button is pressed on the pushbutton box until the button LED lights up. |

# 8 GCS Commands

## 8.1 Notation

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

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

## 8.2 GCS Syntax for Syntax Version 2.0

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

Command mnemonic:

CMD ::= character1 character2 character3 [?]

Exceptions:

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

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

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

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

Exception:

- Single-character commands are not followed by a termination character. However, the response to a single-character command is followed by a termination character.

The argument <AxisID> is used for the logical axes of the controller. Depending on the controller, an axis identifier can consist of up to 16 characters. All alphanumeric characters and the underscore are allowed. Refer to "Commandable Elements" (p. 18) for the identifiers supported by the C-867.

Example 1:

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

Send: `MOV`SP`1`SP`10.0`LF

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

Example 2:

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

Send: `MOV`SP`1`SP`17.3`SP`2`SP`2.05`LF

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

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

Example 3:

All parameters in the volatile memory are to be reset.

Send: `RPA`LF

Example 4:

The position of all axes is to be queried.

Send: `POS?`LF

The response syntax is as follows:

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

With multi-line replies, the space preceding the termination character is left out of the last line:

{[<Argument>[{SP<Argument>}]"="]<Value>SPLF}

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

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

Query command:

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

Response to this command:

<Arg3>"="<Val3>SPLF

<Arg1>"="<Val1>SPLF

<Arg2>"="<Val2>LF

Example 5:

Send: `TSP?`SP`2`SP`1`LF

Receive: `2=-1158.4405`SPLF

`1=+0000.0000`LF

***INFORMATION***

The following restrictions apply to the C-867:

- Up to two items (e. g. axis, channel, or parameter) can be addressed per command line if the command supports this.
- Maximum length of a command line: 512 bytes
- Maximum length of an argument: 31 characters

## 8.3 Target and Sender Address

In principle, the addresses of the target controller and the sender are required in every command line. This applies even to single-character commands like #4 or to macro recording. Because only the PC may send command lines to the controller, its address (0) can be omitted. However, both the target and the sender addresses are part of the controller reply. Multiline responses include the target and sender address only in the first line.

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

Example: Query the device identification string of the C-867.2U2 with address 1

Send: `*IDN?`

The controller replies:
`(c)2024 Physik Instrumente(PI) Karlsruhe, C-867.2U2, 0, 1.2.0.0`

Send: `1 *IDN?`

The same controller replies:
`0 1 (c)2024 Physik Instrumente(PI) Karlsruhe, C-867.2U2, 0, 1.2.0.0`

## 8.4 Variables

For more flexible programming, the C-867 supports variables. 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 present in volatile memory (RAM) only. The variable values are of the STRING data type.

The following conventions apply to variable names:

- Variable names may not contain special characters (especially not "$").
- The maximum number of characters is 8.
- Names of global variables can consist of characters A to Z and 0 to 9. They must start with an alphabetic character.
- Names of local variables may not contain alphabetic characters. Possible characters are 0 to 9.
- The variable name can also be 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, the braces can be left out.

Note that if the braces are left out of variable names consisting of multiple characters, the first character after the "$" is interpreted as the variable name.

**Local variables:**

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

  The command formats are:

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

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

  <STRING1> and <STRING2> indicate the values for the local variables 1 and 2 used in the macro. <STRING1> and <STRING2> can be specified directly or via the values of variables. <uint> defines the number of times the macro is to be run. See the `MAC` command (p. 207) description for more information.
- The local variable 0 is read-only. Its value gives the number of arguments (i.e., values of local variables) set when starting the macro.
- Inside a macro, the values of local variables can be modified using `ADD` (p. 156), `CPY` (p. 159) or `VAR` (p. 244), and can be deleted with `VAR` (except for the local variable 0).
- As long as the macro is running, the values of the local variables can be queried with

  `VAR? 0`

`VAR? 1`

`VAR? 2`

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

**Global variables:**

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

## 8.5 Command Overview

| Com-mand | Arguments | Description |
|---|---|---|
| #4 | | Request Status Register (p. 151) |
| #5 | | Request Motion Status (p. 152) |
| #7 | | Request Controller Ready Status (p. 153) |
| #8 | | Query If Macro Is Running (p. 153) |
| #24 | | Stop All Axes (p. 154) |
| *IDN? | | Get Device Identification (p. 154) |
| ACC | {<AxisID> <Acceleration>} | Set Closed-Loop Acceleration (p. 155) |
| ACC? | [{<AxisID>}] | Get Closed-Loop Acceleration (p. 155) |
| ADD | <Variable> <FLOAT1> <FLOAT2> | Add and Save To Variable (p. 156) |
| CCL | <Level> [<PSWD>] | Set Command Level (p. 158) |
| CCL? | | Get Command Level (p. 159) |
| CPY | <Variable> <CMD?> | Copy Into Variable (p. 159) |
| CST? | [{<AxisID>}] | Get Assignment Of Stages To Axes (p. 160) |
| CSV? | | Get Current Syntax Version (p. 160) |
| CTO | {<TrigOutID> <CTOPam> <Value>} | Set Configuration Of Trigger Output (p. 161) |
| CTO? | [{<TrigOutID> <CTOPam>}] | Get Configuration Of Trigger Output (p. 164) |
| DEC | {<AxisID> <Deceleration>} | Set Closed-Loop Deceleration (p. 165) |
| DEC? | [{<AxisID>}] | Get Closed-Loop Deceleration (p. 166) |
| DEL | <uint> | Delay The Command Interpreter (p. 166) |

| Com-mand | Arguments | Description |
|---|---|---|
| DFH | [{<AxisID>}] | Define Home Position (p. 166) |
| DFH? | [{<AxisID>}] | Get Home Position Definition (p. 168) |
| DIA? | [{<MeasureID>}] | Get Diagnosis Information (p. 168) |
| DIO | {<DIOID> <OutputOn>} | Set Digital Output Lines (p. 169) |
| DIO? | [{<DIOID>}] | Get Digital Input Lines (p. 170) |
| DPA | <Pswd> [{<ItemID> <PamID>}] | Reset Settings to Default (p. 171) |
| DRC | {<RecTableID> <Source> <RecOption>} | Set Data Recorder Configuration (p. 172) |
| DRC? | [{<RecTableID>}] | Get Data Recorder Configuration (p. 173) |
| DRL? | [{<RecTableID>}] | Get Number Of Recorded Points (p. 174) |
| DRR? | [<StartPoint> <NumberOfPoints> [{<RecTableID>}]] | Get Recorded Data Values (p. 174) |
| DRT | {<RecTableID> <TriggerSource> <Value>} | Set Data Recorder Trigger Source (p. 176) |
| DRT? | [{<RecTableID>}] | Get Data Recorder Trigger Source (p. 177) |
| ERR? | | Get Error Number (p. 178) |
| FED | {<AxisID> <EdgeID> <Param>} | Find Edge (p. 178) |
| FRF | [{<AxisID>}] | Fast Reference Move To Reference Switch (p. 180) |
| FRF? | [{<AxisID>}] | Get Referencing Result (p. 181) |
| GOH | [{<AxisID>}] | Go To Home Position (p. 181) |
| HDI? | | Get Help For Interpretation Of DIA? (p. 182) |
| HDR? | | Get All Data Recorder Options (p. 182) |
| HDT | {<HIDeviceID> <HIDeviceAxis> <HIDTableID>} | Set HID Default Lookup Table (p. 183) |
| HDT? | [{<HIDeviceID> <HIDeviceAxis>}] | Get HID Default Lookup Table (p. 184) |
| HIA | {<AxisID> <MotionParam> <HIDeviceID> <HIDeviceAxis>} | Configure Control Done By HID Axis (p. 185) |
| HIA? | [{<AxisID> <MotionParam>}] | Get Configuration Of Control Done By HID Axis (p. 187) |
| HIB? | [{<HIDeviceID> <HIDeviceButton>}] | Get State Of HID Button (p. 187) |
| HIE? | [{<HIDeviceID> <HIDeviceAxis>}] | Get Deflection Of HID Axis (p. 188) |
| HIN | {<AxisID> <HIDControlState>} | Set Activation State For HID Control (p. 189) |
| HIN? | [{<AxisID>}] | Get Activation State Of HID Control (p. 190) |
| HIS? | [{<HIDeviceID> <HIDItemID> <HIDPropID>}] | Get Configuration Of HI Device (p. 190) |

| Com-mand | Arguments | Description |
|---|---|---|
| HIT | {<HIDTableID> <HIDTableAddr> <HIDTableValue>} | Fill HID Lookup Table (p. 193) |
| HIT? | [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]] | Get HID Lookup Table Values (p. 194) |
| HLP? | | Get List of Available Commands (p. 196) |
| HLT | [{<AxisID>}] | Halt Motion Smoothly (p. 197) |
| HPA? | | Get List Of Available Parameters (p. 198) |
| HPV? | | Get List Of Possible Parameter Values (p. 199) |
| IFC | {<InterfacePam> <PamValue>} | Set Interface Parameters Temporarily (p. 200) |
| IFC? | [{<InterfacePam>}] | Get Current Interface Parameters (p. 201) |
| IFS | <Pswd> {<InterfacePam> <PamValue>} | Set Interface Parameters As Default Values (p. 202) |
| IFS? | [{<InterfacePam>}] | Get Interface Parameters As Default Values (p. 204) |
| JRC | <Jump> <CMD?> <OP> <Value> | Jump Relatively Depending On Condition (p. 205) |
| LIM? | [{<AxisID>}] | Indicate Limit Switches (p. 206) |
| MAC | <keyword> {<parameter>}<br>BEG <macro><br>DEF <macro><br>DEF?<br>DEL <macro><br>END<br>ERR?<br>NSTART <macro> <uint> [<String1> [<String2>]]<br>START <macro> [<String1> [<String2>]] | Call Macro Function (p. 207) |
| MAC? | [<macro name>] | List Macros (p. 209) |
| MAN? | <CMD> | Get Help String For Command (p. 210) |
| MAT | <Variable> = <FLOAT1> <OP> <FLOAT2> | Calculate And Save To Variable (p. 211) |
| MEX | <CMD?> <OP> <Value> | Stop Macro Execution Due To Condition (p. 212) |
| MOV | {<AxisID> <Position>} | Set Target Position (p. 213) |
| MOV? | [{<AxisID>}] | Get Target Position (p. 214) |
| MVR | {<AxisID> <Distance>} | Set Target Relative To Current Position (p. 215) |
| ONT? | [{<AxisID>}] | Get On-Target State (p. 216) |
| POS | {<AxisID> <Position>} | Set Real Position (p. 217) |
| POS? | [{<AxisID>}] | Get Real Position (p. 217) |
| RBT | | Reboot System (p. 218) |
| RMC? | | List Running Macros (p. 218) |

| Command | Arguments | Description |
|---|---|---|
| RON | {<AxisID> <ReferenceOn>} | Set Reference Mode (p. 218) |
| RON? | [{<AxisID>}] | Get Reference Mode (p. 219) |
| RPA | [{<ItemID> <PamID>}] | Reset Volatile Memory Parameters (p. 219) |
| RTR | <RecordTableRate> | Set Record Table Rate (p. 220) |
| RTR? | | Get Record Table Rate (p. 221) |
| SAI | {<AxisID> <NewIdentifier>} | Set Current Axis Identifiers (p. 221) |
| SAI? | [ALL] | Get List Of Current Axis Identifiers (p. 222) |
| SEP | <Pswd> {<ItemID> <PamID> <PamValue>} | Set Nonvolatile Memory Parameters (p. 222) |
| SEP? | [{<ItemID> <PamID>}] | Get Nonvolatile Memory Parameters (p. 223) |
| SMO | {<AxisID> <ControlValue>} | Set Open-Loop Control Value (p. 224) |
| SMO? | [{<AxisID>}] | Get Control Value (p. 225) |
| SPA | {<ItemID> <PamID> <PamValue>} | Set Volatile Memory Parameters (p. 226) |
| SPA? | [{<ItemID> <PamID>}] | Get Volatile Memory Parameters (p. 227) |
| SRG? | {<AxisID> <RegisterID>} | Query Status Register Value (p. 228) |
| SST | {<AxisID> <StepSize>} | Set Step Size (for HID control) (p. 229) |
| SST? | [{<AxisID>}] | Get Step Size (of HID control) (p. 230) |
| STE | <AxisID> <Amplitude> | Start Step And Response Measurement (p. 230) |
| STP | | Stop All Axes (p. 231) |
| SVO | {<AxisID> <ServoState>} | Set Servo Mode (p. 232) |
| SVO? | [{<AxisID>}] | Get Servo Mode (p. 232) |
| TAC? | | Tell Number Of Analog Input Lines (p. 233) |
| TAV? | [{<AnalogInputID>}] | Get Analog Input Voltage (p. 233) |
| TCV? | [{AxisID}] | Get Commanded Closed-Loop Velocity (p. 234) |
| TGA | {<Trajectory> <Point>} | Append Value To Trajectory (p. 234) |
| TGC | [{<Trajectory>}] | Clear All Values In Trajectory (p. 235) |
| TGF | [{<Trajectory>}] | Finalize Trajectory (p. 236) |
| TGL? | [{<Trajectory>}] | Get Number Of Values In Trajectory (p. 237) |
| TGS | [{<Trajectory>}] | Start Trajectory (p. 237) |
| TGT | <NoOfServoCycles> | Set Trajectory Timing (p. 238) |
| TGT? | | Get Trajectory Timing (p. 240) |
| TIM | [<Float>] | Set Timer Value (p. 240) |

| Com-mand | Arguments | Description |
|---|---|---|
| TIM? | | Get Timer Value (p. 240) |
| TIO? | | Tell Number Of Digital I/O Lines (p. 241) |
| TMN? | [{<AxisID>}] | Get Minimum Commandable Position (p. 241) |
| TMX? | [{<AxisID>}] | Get Maximum Commandable Position (p. 241) |
| TNR? | | Get Number Of Record Tables (p. 242) |
| TRO | {<TrigOutID> <TrigMode>} | Set Trigger Output State (p. 242) |
| TRO? | [{<TrigOutID>}] | Get Trigger Output State (p. 243) |
| TRS? | [{<AxisID>}] | Indicate Reference Switch (p. 243) |
| TVI? | | Tell Valid Character Set For Axis Identifiers (p. 244) |
| VAR | <Variable> <String> | Set Variable Value (p. 244) |
| VAR? | [{<Variable>}] | Get Variable Value (p. 245) |
| VEL | {<AxisID> <Velocity>} | Set Closed-Loop Velocity (p. 246) |
| VEL? | [{<AxisID>}] | Get Closed-Loop Velocity (p. 247) |
| VER? | | Get Versions Of Firmware And Drivers (p. 247) |
| WAC | <CMD?> <OP> <Value> | Wait For Condition (p. 247) |
| WPA | <Pswd> [{<ItemID> <PamID>}] | Save Parameters To Non-Volatile Memory (p. 248) |

## 8.6 Command Descriptions for GCS 2.0

### #4 (Request Status Register)

Description: Queries system status information.

Format: #4

Arguments: none

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

Notes: This command is identical in function to SRG? (p. 228), but only one character is sent via the interface. Therefore #4 can also be used while the controller is performing time-consuming tasks.

For multi-axis controllers, the response has the following format:

0x<status Axis1><status Axis2> etc.

Deactivated axes are not included in the response.
<Status Axis#> is the sum of the following codes in hexadecimal format:

| Bit | Description |
|---|---|
| 15 | On-target state |
| 14 | Determines the reference value |
| 13 | In motion |
| 12 | Servo mode on |
| 11 | - |
| 10 | - |
| 9 | - |
| 8 | Error flag |
| 7 | Digital Input 4 |
| 6 | Digital Input 3 |
| 5 | Digital Input 2 |
| 4 | Digital Input 1 |
| 3 | - |
| 2 | Positive limit switch |
| 1 | Reference switch |
| 0 | Negative limit switch |

Example:

Response of a controller with two active axes:
`0x90021102`
The response means:

- For axis 1: `9002`
The axis is on target (on-target state = true), servo mode is ON, no error has occurred, the states of digital input lines 1 to 4 are low, and the positioner axis is on the positive side of the reference switch.
- For axis 2: `1102`
The axis is not on target (on-target state = false), servo mode is ON, an error has occurred, the states of digital input lines 1 to 4 are low, and the positioner axis is on the positive side of the reference switch.

### #5 (Request Motion Status)

Description: Queries the motion status of the axes.

Format: #5

Arguments: None

Response: The response <uint> is bit-mapped and returned as the hexadecimal sum of the following codes:

1=First axis in motion
2=Second axis in motion
4=Third axis in motion
...
0 indicates that all axes have finished moving.

### #7 (Request Controller Ready Status)

Description: Queries the controller for ready state (tests if controller is ready to do a new command).

Note: Use #5 (p. 152) instead of #7 to verify if motion has finished.

Format: #7

Arguments: None

Response: B1h (ASCII character 177 = "±" in Windows) if controller is ready

B0h (ASCII character 176 = "°" in Windows) if controller is not ready
(e.g., executing a referencing move)

Troubleshooting: The response characters may be displayed differently in non-Western character sets or other operating systems.

### #8 (Query if Macro Is Running)

Description: Tests if a macro is running on the controller.

Format: #8

Arguments: None

Response: <uint>=0 no macro is running
<uint>=1 a macro is currently running

**#24 (Stop All Axes)**

Description: Stops all axes abruptly. See the notes below for further details.

Sets error code to 10.

This command is identical in function to STP (p. 231), but only one character is sent via the interface.

Format: #24

Arguments: None

Response: None

Notes: #24 stops all motion caused by motion commands (e.g., MOV (p. 213), MVR (p. 215), GOH (p. 181), STE (p. 230), SMO (p. 224)), follow trajectory (TGS (p. 237)), the command for referencing (FRF (p. 180)), and macros (MAC (p. 207)). Also stops macro running.

After the axes are stopped, their target positions are set to their current positions.

HLT (p. 197) in contrast to #24 stops motion with specified deceleration regarding system inertia. Does not apply to trajectories.

Setting the ***Controller Disable Error 10*** parameter (0xE000301) prevents the error code 10 from being output when an axis motion is stopped with #24.

***IDN? (Get Device Identification)**

Description: Reports the device identity number.

Format: *IDN?

Arguments: None

Response: Single-line text terminated with a termination character (line feed) with controller name, serial number, and firmware version

Notes: With C-867, *IDN? responds something like:

```
(c)2024 Physik Instrumente (PI) GmbH &
Co. KG, C-867, 113059603, 1.005
```

### ACC (Set Closed-Loop Acceleration)

Description: Sets acceleration of specified axes.

ACC can be changed while the axis is moving.

Format: ACC {<AxisID> <Acceleration>}

Arguments: <AxisID> is one axis of the controller

<Acceleration> is the acceleration value in physical units/s².

Response: None

Troubleshooting: Illegal axis identifiers

Notes: The ACC setting only takes effect when the specified axis is in closed-loop operation (servo mode ON).

The lowest possible value for <Acceleration> is 0.

ACC changes the value of the ***Closed-Loop Acceleration (Phys. Unit/s²)*** parameter (ID 0xB) in the volatile memory of the C-867. The parameter value can be stored as default with WPA (p. 248), for details see "Adapting Settings" (p. 273).

The maximum value that can be set with the ACC command is specified by the ***Maximum Closed-Loop Acceleration (Phys. Unit/s²)*** parameter (ID 0x4A).

### ACC? (Get Closed-Loop Acceleration)

Description: Queries the acceleration value set with ACC (p. 155).

If all arguments are left out, gets the value of all axes set with ACC.

Format: ACC? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the acceleration value set with ACC, in physical units/$s^2$.

**ADD (Add and Save to Variable)**

Description: Adds two values and saves the result to a variable (p. 146).

The variable is present in volatile memory (RAM) only.

Format: ADD <Variable> <FLOAT1> <FLOAT2>

Arguments: <Variable> is the name of the variable to which the result is to be saved.

<FLOAT1> is the first summand.

<FLOAT2> is the second summand.

Floating point numbers are expected for the summands. They can be specified directly or via the value of a variable.

Response: None

Notes: Local variables can be set using ADD in macros only.

Example 1: Value $B is added to value $A, and the result is saved to variable C:

`ADD C $A $B`

Example 2: The name of the variable where the result is to be copied is specified via the value of another variable:

Send: `VAR?`
Receive:
`A=468`
`B=123`
`3Z=WORKS`

Send: `ADD A${3Z} $A $B`
Send: `VAR?`
Receive:
`A=468`
`B=123`
`AWORKS=591`

`3Z=WORKS`

Send: `ADD ${3Z} $A $B`
Send: `VAR?`
Receive:
```
A=468
B=123
AWORKS=591
WORKS=591
3Z=WORKS
```

Example 3:

The macros below can be used to create a "flashing light" with LEDs that are connected to the digital output lines of the controller. $1 and $2 are values of local variables and must be specified as arguments of MAC START or MAC NSTART command when starting the macros (see below).

DIO 0 <bitmask>: Sets the output channels according to <bitmask>. For example, "DIO 0 5" activates channels 1 and 3 and deactivates all other channels (5 is 0000 0101 in binary notation).

To implement the "flashing light", do the following steps:

1. Write the "STEPS" macro:
```
MAC BEG STEPS
DIO 0 $1
ADD 1 $1 1
DEL $2
JRC -3 VAR? 1 <= 15
ADD 1 $1 -1
DIO 0 $1
DEL $2
JRC -3 VAR? 1 > 0
MAC END
```
2. Write the "TEST" macro:
```
MAC BEG TEST
MAC START STEPS 0 $1
ADD 1 $1 10
JRC -2 VAR? 1 < 110
VAR 1 10
ADD 2 $2 -1
JRC -5 VAR? 2 > 0
MAC END
```
3. Start the TEST macro with arguments that define the variable values $1 and $2:
```
MAC START Test 10 50
```

Meaning of the variables here:

$1: Delay in ms between each step in the STEPS macro. The value is incremented by 10 by the TEST macro until it reaches 110.

$2: Number of repetitions of the whole "flashing light" procedure.

**CCL (Set Command Level)**

| | |
|---|---|
| Description: | Changes the active "command level" and therefore determines the availability of commands and write access to system parameters. |
| Format: | CCL <Level> [<PSWD>] |
| Arguments: | <Level> is a command level of the controller<br><br><PSWD> is the password required for changing to the appropriate command level<br><br>The following command levels and passwords apply:<br><br>Level = 0 is the default setting, all commands provided for "normal" users are available, read access to all parameters, no password required.<br><br>Level = 1 adds additional commands and write access to level-1 parameters (commands and parameters from level 0 are included). The password required is "advanced".<br><br>Level > 1 is only intended for PI service personnel. Users cannot change to a level > 1. Contact the customer service department (p. 309) if you have problems with the parameters for command level 2 or higher. |
| Response: | none |
| Troubleshooting: | Invalid password |
| Notes: | With the C-867, the command levels only determine the write permission for the parameters. The availability of the commands of the C-867 is independent of the active command level.<br><br>HPA? (p. 198) lists the parameters including the information on which command level allows write access to them. For further information on using parameters, see "Adapting Settings" (p. 273).<br><br>After controller switch-on or reboot, the active command level is always level 0. |

### CCL? (Get Command Level)

Description: Get the active "command level".

Format: CCL?

Arguments: none

Response: <Level> is the currently active command level; uint.

Notes: <Level> should be 0 or 1.

<Level> = 0 is the default setting, write access is specified for level 0 parameters, read access is specified for all parameters

<Level> = 1 allows write access for level 1 parameters (parameters from level 0 are included).

### CPY (Copy Into Variable)

Description: Copies a command response to a variable (p. 146).

The variable is present in volatile memory (RAM) only.

Format: CPY <Variable> <CMD?>

Arguments: <Variable> is the name of the variable to which the command response is to be copied.

<CMD?> is one query command in its usual notation. The response has to be a single value and not more.

Response: None

Notes: Local variables can be set using CPY in macros only.

Example 1: Using the following macro, it is possible to connect through the digital input and output lines of the controller. 1 is a local variable whose value must be specified as argument of the MAC START or MAC NSTART command when starting the macro.

Write the "connect" macro:

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

| | |
|---|---|
| | `MAC END` |
| Example 2: | It is possible to copy the value of one variable (e.g., SOURCE) to another variable (e.g., TARGET):<br><br>`CPY TARGET VAR? SOURCE` |

### CST? (Get Assignment Of Stages To Axes)

| | |
|---|---|
| Description: | Returns the name of the connected positioner type for the queried axis. |
| Format: | CST? [{<AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller |
| Response: | {<AxisID>"="<string> LF}<br><br>where<br><br><string> is the name of the positioner type assigned to the axis. |
| Notes: | The positioner name is read from the ***Stage Name*** parameter (ID 0x3C). If the parameter has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries). The identifier of a deactivated axis can only be queried with `SAI? ALL`.<br><br>You can set the value of the 0x3C parameter specifically to the name of your positioner with SPA (p. 226) or SEP (p. 222). Because the PC software from PI uses the parameter value to configure the C-867 for the connected positioner (p. 80), it is not recommended to change manually with SPA or SEP. |

### CSV? (Get Current Syntax Version)

| | |
|---|---|
| Description: | Queries the GCS syntax version used in the firmware. |
| Format: | CSV? |
| Arguments: | None |
| Response: | The current GCS syntax version |
| Notes: | 1.0 (for GCS 1.0) or 2.0 (for GCS 2.0) are possible responses. |

### CTO (Set Configuration Of Trigger Output)

| | |
|---|---|
| Description: | Configures the trigger output conditions for the specified digital output line. |
| Format: | CTO {<TrigOutID> <CTOPam> <Value>} |
| Arguments: | <TrigOutID> is one digital output line of the controller, see below for details.<br><br><CTOPam> is the CTO parameter ID in decimal format, see below for the available IDs.<br><br><Value> is the value that the CTO parameter is set to, see below. |
| Response: | None |
| Notes: | The trigger output conditions will become active when enabled with TRO (p. 242). Do not use DIO (p. 169) on digital output lines for which the trigger output is enabled with TRO.<br><br>The CTO settings are lost when you power down or reboot the C-867. An easy way to keep them is to save them to a macro. |
| Output lines and trigger conditions available: | <TrigOutID> corresponds to digital output lines 1 to 4, IDs = 1 to 4; see "I/O" (p. 316).<br><br><CTOPam> parameter IDs available for C-867:<br><br>1 = TriggerStep<br>2 = Axis<br>3 = TriggerMode<br>7 = Polarity<br>8 = StartThreshold<br>9 = StopThreshold<br>10 = TriggerPosition<br>11 = PulseWidth<br><br><Value> available for the appropriate <CTOPam> ID:<br><br>for TriggerStep: Distance<br><br>for Axis: The identifier of the axis to be connected to the digital output line. Irrelevant for the MotionError trigger mode.<br><br>for TriggerMode (default value is 0): |

- 0 = PositionDistance;
a trigger pulse is written whenever the axis has covered the TriggerStep distance (<CTOPam> ID 1). Optionally, values for StartThreshold and StopThreshold (<CTOPam> IDs 8 and 9) can be defined to activate the trigger output for a limited position range and a certain direction of motion only (negative or positive; Note: If the motion direction is reversed before the axis position has reached the stop threshold, trigger pulses will continue to be generated). When StartThreshold and StopThreshold are set to the same value, they will not be used.
- 2 = OnTarget;
the on-target state of the selected axis is transferred to the selected digital output line (this state can also be read with the ONT? command).
- 5 = MotionError;
the selected digital output line becomes active when a motion error occurs. The line will stay active until the error code is reset to 0 (by a query).
- 6 = InMotion;
the selected digital output line is active as long as the selected axis is in motion (the motion state can also be read with commands, e.g. SRG? or #5).
- 7 = Position+Offset;
the first trigger pulse is written when the axis has reached the position specified by TriggerPosition (<CTOPam> ID 10). The next trigger pulses are written each time the axis position equals the sum of the last valid trigger position and the distance specified by TriggerStep (<CTOPam> ID 1). Trigger output ends when the axis position exceeds the value specified by StopThreshold (<CTOPam> ID 9). The sign of the TriggerStep value determines the direction of motion in which trigger pulses are to be output. Trigger processing is done by the DSP of the C-867.
- 8 = SinglePosition;
the selected digital output line is active when the axis position has reached or exceeded the position specified by TriggerPosition (<CTOPam> ID 10).
- 9 = HardwareTrigger;
basically corresponds to the Position+Offset trigger mode but is executed by the FPGA circuit of the C-867 (shorter processing time). Further differences to Position+Offset: Assignment of the axes to the digital output lines is fast (axis 1 to line 1, axis 2 to line 2); HardwareTrigger functions only with A/B signals. The pulse width of the trigger pulses is determined by the PulseWidth factor (<CTOPam> ID 11).

for Polarity (default value is 1): sets the signal polarity for the digital output line
0 = Active Low
1 = Active High

for StartThreshold/StopThreshold: Position value;
if used for the PositionDistance trigger mode, both thresholds must be set in order to determine the position range and the direction of motion for the trigger output;
StopThreshold is used as the stop condition for Position+Offset and HardwareTrigger trigger modes

for TriggerPosition: Position value;
if used in the Position+Offset and HardwareTrigger trigger modes, the first trigger pulse is output at this position;
if used in SinglePosition trigger mode, the output line is active when this position has been attained or exceeded

for PulseWidth: Factor "n", which determines the pulse width for the HardwareTrigger trigger mode as follows:
Pulse width = n * 33.3 ns

For application examples and further details see "Digital Output Signals" (p. 100) and the lines below.

Example 1: A pulse is to be generated on digital output line 1 (ID 1) whenever axis 1 has covered a distance of 0.05 µm. The following parameters must be set:

TrigOutID = 1
Axis = 1
TriggerMode = 0
TriggerStep = 0.05
Send: `CTO 1 2 1`
Send: `CTO 1 3 0`
Send: `CTO 1 1 0.00005`

Example 2: In this example, digital output line 1 is to be set from low to high when axis A starts to move. The following parameters must be set:
TrigOutID = 1
Axis = A (axis identifier was changed with `SAI`)
TriggerMode = 6
Polarity = Active High
So you have to send:
```
CTO 1 2 A
CTO 1 3 6
CTO 1 7 1
```

Example 3: U-521.23 is connected to axis 1. The reference position of U-521.23 is 9 mm. Starting from its reference position, the axis is to be moved forwards and backwards alternately; trigger pulses are to be output for both directions of motion in a range of 1 mm using the Position+Offset trigger mode. For that purpose, two macros are written to the controller. Macro TRIGREF initializes the controller and could also be defined as startup macro, while macro TRIGGER starts motion and therefore trigger output. Write the macros as shown below. For further information on macros, see "Controller Macros" (p. 126).

Make sure that the velocity for the axis matches the CTO setting for the distance. Recommended value:
maximum velocity = distance * 20 kHz / 2
where 20 kHz is the frequency of the C-867 servo cycle.

In this example, the distance is set to 0.02 mm, so the axis velocity should be a maximum of 200 mm/s.

➢ Record a macro named TRIGREF with the following contents:

```
CTO 1 3 7
SVO 1 1
FRF
TRO 1 1
MAC START TRIGGER
```

➢ Record a macro named TRIGGER with the following contents:

```
CTO 1 1 0.02
CTO 1 9 11
CTO 1 10 10
DEL 1000
MOV 1 12
WAC POS? 1 > 11.8
MEX CTO? 1 10 < 10.9
CTO 1 1 -0.02
CTO 1 9 10
CTO 1 10 11
DEL 1000
MOV 1 9
WAC POS? 1 < 9.2
MEX CTO? 1 10 > 10.1
MAC START TRIGGER
```

### CTO? (Get Configuration Of Trigger Output)

Description: Queries the values set for specified trigger output lines and parameters.

Format: CTO? [{<TrigOutID> <CTOPam>}]

Arguments: <TrigOutID>: is a digital output line of the controller; see CTO.

<CTOPam>: parameter ID; see CTO.

If all arguments are left out, the response contains the values for all parameters and all output lines.

Response: {<TrigOutID> <CTOPam>"="<Value> LF}

For <Value> see CTO.

### DEC (Set Closed-Loop Deceleration)

Description: Sets deceleration of specified axes.

DEC can be changed while the axis is in motion.

Format: DEC {<AxisID> <Deceleration>}

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

<Deceleration> is the deceleration value in physical units/$s^2$.

Response: None

Troubleshooting: Illegal axis identifiers

Notes: The DEC setting only takes effect when the specified axis is in closed-loop operation (servo mode ON).

The lowest possible value for <Deceleration> is 0.

DEC changes the value of the ***Closed Loop Deceleration (Phys. Unit/s²)*** parameter (ID 0xC) in the volatile memory of the C-867. The parameter value can be stored as default with WPA (p. 248), for details see "Adapting Settings" (p. 273).

The maximum value that can be set with the DEC command is specified by the ***Maximum Closed-Loop Deceleration (Phys. Unit/s$^2$)*** parameter (ID 0x4B).

### DEC? (Get Closed-Loop Deceleration)

Description: Queries the deceleration value set with DEC (p. 165).

If no arguments are specified, queries the value of all axes set with DEC.

Format: DEC? [{<AxisID>}]

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

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

where

<float> is the deceleration value set with DEC, in physical units/s$^2$.

### DEL (Delay the Command Interpreter)

Description: Delays <uint> milliseconds.

Format: DEL <uint>

Arguments: <uint> is the delay value in milliseconds.

Response: None

Notes: DEL can only be used in macros. Do not mistake MAC DEL (deletes macros) for DEL (delays).
Further information can be found in the description of the MAC command (p. 207) and in the "Controller Macros" (p. 126) section.

### DFH (Define Home Position)

Description: Redefines the zero position of the specified axis by setting the position value to zero at the current position.

If no arguments are specified, DFH defines the zero position of all axes.

Format: DFH [{<AxisID>}]

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

Response: none

Troubleshooting: Illegal axis identifier

Notes: DFH sets the current position of the axis to zero and saves the position value which was valid when the command was called as offset in the volatile memory. By adding this offset to the response, the output values of the following commands are adapted to the new zero position:

- POS? (p. 217) (Get the current position)
- TMN? (p. 241) (Get the minimum commandable position)
- TMX? (p. 241) (Get the maximum commandable position)

DFH does **not** change the values of the parameters for the definition of travel range and soft limits (p. 47).

The offset is reset to zero in the following cases:

- When switching on and rebooting the C-867: For all axes
- During referencing: For the affected axis

Example:

Send: `MOV 1 9.87`
Send: `POS? 1`
Receive: `1=9.8700005`
Send: `DFH? 1`
Receive: `1=0.0000000`
Send: `TMN? 1`
Receive: `1=0.0000000`
Send: `TMX? 1`
Receive: `1=14.9999982`

Note: Axis 1 is moved to absolute position 9.87 mm. Finally, the current axis position (with POS?), the current offset value (with DFH?), and the minimum and maximum commandable position (with TMN? and TMX?) are queried.

Send: `DFH 1`
Send: `POS? 1`
Receive: `1=0.0000000`
Send: `DFH? 1`
Receive: `1=9.8700005`
Send: `TMN? 1`
Receive: `1=-9.8700005`

Send: `TMX? 1`

Receive: `1=5.1299978`

Note: The axis has not moved. The current axis position was defined as new zero position using DFH. Therefore, the offset value of axis 1 is 9.87 mm. The values for the minimum and maximum commandable position were adapted to the new zero position by adding the offset.

### DFH? (Get Home Position Definition)

Description: Queries the position value that is currently used as the offset for the specified axis to move the zero position.

If no arguments are specified, queries the position value of all axes.

Format: DFH? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<PositionOffset> is the axis position that was valid at the time the last DFH command was processed. This position value is used internally as offset for the calculation of the current axis position.

Troubleshooting: Illegal axis identifier

Notes: The axis position that was valid when the last DFH command was processed, is available in the volatile memory as an offset. The offset is reset to zero in the following cases:

- When switching on and rebooting the C-867: For all axes
- During referencing: For the affected axis

See DFH for an example.

### DIA? (Get Diagnosis Information)

Description: Gets the current value of the given measurand.

If all arguments are omitted, the current values of all measurands are queried.

Format: DIA? [{<MeasureID>}]

Arguments: <MeasureID> is the identifier of one measurand; see below for details.

Response: {<MeasureID>"="<MeasuredValue> LF}

where

<MeasuredValue> gives the current value of the measurand; see below for details.

Note: The C-867 supports the following measurands:

1: Position Error Axis 1

2: Position Error Axis 2

Current position error (i.e., the absolute value of the difference between the current position and the commanded position), in physical units

3: Motor Output Axis 1

4: Motor Output Axis 2

Current absolute control value in % of 32767

5: Motor frequency 1 Axis 1

6: Motor frequency 2 Axis 1

Axis 1: Current frequency of the 1st and 2nd motor phase, in kHz

7: Motor frequency 1 Axis 2

8: Motor frequency 2 Axis 2

Axis 2: Current frequency of the 1st and 2nd motor phase, in kHz

9: Ref Capture Position Axis 1

10: Ref Capture Position Axis 2

Referencing: Encoder value at the reference signal, in physical units

11: Ref Diff Position Axis 1

12: Ref Diff Position Axis 2

Referencing: Deviation of the current position from the reference position, in physical units

Use the response to the HDI? command (p. 182) to obtain descriptions and physical units of the supported measurands.

### DIO (Set Digital Output Line)

| | |
|---|---|
| Description: | Switches the specified digital output line(s) to specified state(s).<br><br>Use TIO? (p. 241) to get the number of installed digital I/O lines. |
| Format: | DIO {<DIOID> <OutputOn>} |
| Arguments: | <DIOID> is one digital output line of the controller, see below for details.<br><br><OutputOn> is the state of the digital output line, see below for details. |
| Response: | none |
| Notes: | You can use the DIO command to activate/deactivate the Output 1 to Output 4 lines on the I/O socket (p. 316). The C-867 allows you to either set a single line per DIO command, or all lines at once.<br><br>The <DIOID> identifiers to use for the lines are 1 to 4. With the identifier 0, all lines are set according to a bit pattern specified by <OutputOn>.<br><br>If <OutputOn>=1 the line is set to HIGH/ON, if <OutputOn>=0 it is set to LOW/OFF.<br>Do not use DIO on output lines for which the trigger output is activated with TRO (p. 242). |

### DIO? (Get Digital Input Lines)

| | |
|---|---|
| Description: | Queries the states of the specified digital input lines.<br><br>Use TIO? (p. 241) to query the number of available digital I/O lines. |
| Format: | DIO? [{<DIOID>}] |
| Arguments: | <DIOID> is the identifier of the digital input line, see below for details. |
| Response: | {<DIOID>"="<InputOn> LF}<br><br>where |

| | |
|---|---|
| | <InputOn> specifies the state of the digital input line, see below for details. |
| Notes: | You can use the DIO? command to read digital input lines 1 to 4 on the **I/O** socket directly (p. 316).<br><br>The <DIOID> identifiers to use for the lines are 1 to 4. If the identifier is left out or 0, all lines are queried.<br><br>If <InputOn>=0, the digital input is LOW/OFF; if <InputOn>=1, the digital input is HIGH/ON. If <DIOID> is 0, <InputOn> is a bit pattern which gives the states of all lines in hexadecimal format. |

### DPA (Reset Settings to Default)

| | |
|---|---|
| Description: | Resets parameter values and parameter-independent settings to the factory settings. |
| Format: | DPA <Pswd> [{<ItemID> <PamID>}] |
| Arguments: | <Pswd> is the password for resetting the memory. See below for details.<br><br><ItemID> is the element for which a parameter is to be reset. See below for details.<br><br><PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details. |
| Response: | None |
| Troubleshooting: | Illegal element identifier, wrong parameter ID, invalid password |
| Notes: | With the C-867, DPA resets the following settings to default settings in the volatile memory **and** in the nonvolatile memory:<br>▪ Values of all parameters<br>▪ Identifier of the axis<br>▪ Settings for the HID control: HDT, HIA, HIT<br><br>Valid password: 100<br><br>With the C-867, the specification of <ItemID> and <PamID> is not necessary.<br><br>DPA should only be executed when new parameters have been introduced with a firmware update; for details see "Updating Firmware" (p. 299).<br><br>Before executing the DPA command, create a backup copy |

of the current parameter values on the PC (see "Saving Parameter Values in a Text File" (p. 275)). You can then restore the overwritten settings at any time.

In the default setting, the ***Stage Name*** parameter (ID 0x3C) has the value NOSTAGE, so that the axis of the C-867 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.

The saving of settings with WPA (p. 248) has no effect on the default settings that are loaded with DPA.

**DRC (Set Data Recorder Configuration)**

Description: Determines the data source to be used and the type of data to be recorded (record option) for the data recorder table specified.

Format: DRC {<RecTableID> <Source> <RecOption>}

Arguments: <RecTableID> is one data recorder table of the controller, see below.

<Source> is the ID of the data source, for example, an axis or channel of the controller. The required source depends on the selected record option.

<RecOption> is the type of data to be recorded (record option).

Refer to the following list of available record options and the corresponding data sources for details

Response: None

Notes: The C-867 has 4 data recorder tables with 8192 points per table.

With HDR? (p. 182), you will obtain a list of all available record and trigger options and additional information on the data recording. The number of available data recorder tables can be read with TNR? (p. 242).

Refer to "Data Recorder" (p. 98) for further information.

Available recording options:

0=Nothing is recorded

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
81=Analog input (channel = 1 - 8)
86=Number of fifo values for axis
87=Interpolation data for axis
90=Active parameter set of axis
91=Actual frequency of axis
92=p0
93=DIA?

Note: The input channels for the record option 81 can be the following channels:

- 1 to 4: Input lines 1 to 4 of the **I/O** (p. 316) socket
- 5 to 8: Input lines 5 to 8 of HID 1 (**Analog Joystick** and **Analog In** (p. 317)) sockets

**DRC? (Get Data Recorder Configuration)**

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

Format: DRC? [{<RecTableID>}]

Arguments: <RecTableID>: is a data recorder table of the controller; if this entry is not specified, the response will contain the settings for all tables.

Response: The current DRC settings:

{<RecTableID>"="<Source> <RecOption> LF}

where

<Source>: is the data source, for example, an axis or a channel of the controller. The source type depends on the record option.

<RecOption>: is the type of data to be recorded (record option).

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

182).

**DRL? (Get Number of Recorded Points)**

Description: Reads the number of points comprised by the last recording.

Format: DRL? [{<RecTableID>}]

Arguments: <RecTableID> is one data recorder table of the controller

Response: {<RecTableID>"="<uint> LF}

where

<uint> specifies the number of points recorded with the last recording

Notes: The number of points is reset to zero for a data recorder table when changing its configuration with DRC (p. 172).

When the ***Data Recorder Buffer Mode*** parameter (ID 0x16000003) has the value 1, the recording does **not** end when the data recorder tables are full but is continued with the first point of the data recorder tables. In this case, the contents of the data recorder tables might be overwritten one or more times. For this reason, the data should already be read out with DRR? while the recording is still in progress. With the response to DRL? and the value of the ***Data Recorder Buffer Overflow*** parameter (ID 0x16000004), you can calculate the number of points that were recorded since the last DRR? query:

Number of recorded points = response to DRL? + max. number of points per table * value of ***Data Recorder Buffer Overflow***

**DRR? (Get Recorded Data Values)**

Description: Queries the last recorded data.

Querying can take some time depending on the number of points to be read!

It is possible to read the data while recording is still in progress.

Format: DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]

Arguments: <StartPoint> is the first point to be read from the data recorder table, starts with index 1.

<NumberOfPoints> is the number of points to be read per table.

<RecTableID> is one data recorder table of the controller.

Response: For the recorded data in GCS array format, refer to the separate manual for the GCS array, SM146E, and the example below.

Notes: If <RecTableID> is not specified, the data is read from all tables with a record option not equal to zero.

When the recording starts again with the first point of the data tables because the data tables are full (***Data Recorder Buffer Mode*** parameter has the value 1), the value of the buffer overflow counter (***Data Recorder Buffer Overflow*** parameter, ID 0x16000004) is increased by 1 each time. Reading out the recorded data with DRR? resets the value of the buffer overflow counter to zero.

With HDR? (p. 182), you will obtain a list of all available recording and triggering options as well as additional information on data recording.

Refer to the description of the DRC (p. 172) and DRL? (p. 174) commands as well as "Data Recorder" (p. 98) for further information.

Example:

```
rtr?
10
drr? 1 20
# REM C-867
#
# VERSION = 1
# TYPE = 1
# SEPARATOR = 32
# DIM = 2
# SAMPLE_TIME = 0.00050
# NDATA = 20
#
# NAME0 = Actual Position of Axis
AXIS:1
# NAME1 = Position Error of Axis  AXIS:1
#
# END_HEADER
5.00000 0.00000
```

```
4.99998 0.00002
5.00000 0.00000
5.00000 0.00000
5.00000 0.00000
5.00000 0.00000
5.00000 0.00000
4.99998 0.00002
5.00000 0.00000
4.99998 0.00002
5.00000 0.00000
5.00000 0.00000
5.00000 0.00000
5.00000 0.00000
4.99998 0.00002
5.00000 0.00000
4.99998 0.00002
4.99998 0.00002
5.00000 0.00002
4.99998 0.00004
```

**DRT (Set Data Recorder Trigger Source)**

| | |
|---|---|
| Description: | Defines a trigger source for the specified data recorder table. |
| Format: | DRT <RecTableID> <TriggerSource> <Value> |
| Arguments: | <RecTableID> is one data recorder table of the controller. See below for details.<br><br><TriggerSource> ID of the trigger source, see below for a list of available options.<br><br><Value> depends on the trigger source, can be a dummy, see below. |
| Response: | none |
| Notes: | Currently, only 0 is valid for <RecTableID>; this means that the specified trigger source is set for all data recorder tables with a record option that is not zero.<br><br>Irrespective of the set trigger option, data recording is always triggered when step response measuring is done with STE (p. 230).<br><br>With HDR? (p. 182), you will obtain a list of all available recording and triggering options as well as additional |

| | |
|---|---|
| | information on data recording.<br><br>Refer to the description of the DRC command (p. 172) as well as "Data Recorder" (p. 98) for further information. |
| Available trigger options: | 0 = default setting; data recording is triggered by STE; <Value> must be a dummy.<br><br>1 = any command changing target position (e.g., MVR (p. 215), MOV (p. 213)); <Value> must be a dummy.<br><br>2 = next command, resets trigger after execution; <Value> must be a dummy.<br><br>3 = external trigger; data recording is started with the digital input line whose ID is specified by <Value> (see "I/O" (p. 316) for available input lines).<br><br>6 = any command changing target position (e.g., MVR, MOV), resets trigger after execution; <Value> must be a dummy.<br><br>7 = SMO command, resets trigger after execution; <Value> must be a dummy. |
| Example: | The recording is to be triggered for all data recorder tables via digital input line 1. Send:<br>`DRT 0 3 1`<br>The servo cycle time of the C-867 is 50 µs. For this reason, the trigger signal should have a frequency ≤ 10 kHz. With RTR you can set the record table rate, i.e., the number of servo cycles to be used in data recording operations. |

### DRT? (Get Data Recorder Trigger Source)

| | |
|---|---|
| Description: | Queries the trigger source for the data recorder tables. |
| Format: | DRT? [{<RecTableID>}] |
| Arguments: | <RecTableID> is one data recorder table of the controller. |
| Response: | {<RecTableID>"="<TriggerSource> <Value> LF}<br><br>where<br><br><TriggerSource> is the identifier of the trigger source.<br><br><Value> depends on the trigger source. |

| | |
|---|---|
| | Further information can be found in the description of the DRT command (p. 176). |
| Notes: | Because all data record tables of the C-867 have the same trigger source, the DRT? response is specified as a single line of the form<br><br>0=<TriggerSource> <Value> |

**ERR? (Get Error Number)**

| | |
|---|---|
| Description: | Get error code <int> of the last occurred error and reset the error to 0.<br><br>Only the last error is buffered. You should therefore call ERR? after each command.<br><br>The error codes and their descriptions are listed in "Error Codes" (p. 250). |
| Format: | ERR? |
| Arguments: | None |
| Response: | The error code of the last error that occurred (integer). |
| Troubleshooting: | Communication breakdown |
| Notes: | In the case of simultaneous access to the controller by several instances, the error code is only returned to the first instance that sent the ERR? command. Because the error is reset to 0 by the query, the error is not visible for any further querying instance.<br>➢ If possible, access the controller with one instance only.<br>➢ If incorrect system behavior does not cause the controller to send an error code, check whether the error code is queried regularly in the background by a macro, script or the PC software (e.g., PIMikroMove).<br><br>If the cause of an error continues, the corresponding error code is immediately set again after a query with ERR?. |

**FED (Find Edge)**

| | |
|---|---|
| Description: | Moves the specified axis to a specified signal edge.<br>FED does not set a specific position value at the selected edge (in contrast to the FRF (p. 180) command for |

| | |
|---|---|
| | referencing), i.e., the axis is not referenced after FED is used.<br>If multiple axes are specified in the command, they are moved synchronously. |
| Format: | FED {<AxisID> <EdgeID> <Param>} |
| Arguments: | <AxisID> is one axis of the controller.<br><EdgeID> is the type of edge the axis has to move to. See below for available edge types.<br><Param> depends on the selected edge and determines it more precisely. See below for details. |
| Response: | None |
| Troubleshooting: | ▪ Illegal axis identifier<br>▪ Limit switches and/or reference switches are disabled<br>▪ Servo mode is switched off. |
| Notes: | Servo mode must be switched on with SVO (p. 232) for the commanded axis prior to using this command (closed-loop operation).<br><br>The firmware of the C-867 determines the following based on parameters:<br>▪ Is a reference switch present (parameter 0x14)?<br>▪ Are limit switches present (parameter 0x32)?<br>▪ If the reference switch is represented by an index pulse: How should the move to the index pulse take place (parameters 0x70, 0x78, 0x79)?<br>According to the values of those parameters, the C-867 activates or deactivates FED motion to the corresponding signal edges. Adapt the parameter values to your hardware using SPA (p. 226) or SEP (p. 222). Refer to "Adapting Settings" for further information.<br><br>You can use the digital input lines instead of the switches as source of the switch signals for FED. Refer to "Digital Input Signals" (p. 109) for further information.<br><br>FED can be used to measure the physical travel range of new mechanics and therefore determine the values for the corresponding parameters:<br>▪ Distance from the negative to the positive limit switch<br>▪ Gap between the negative limit switch and the reference switch (parameter ID 0x17)<br>▪ Gap between the reference switch and the positive limit switch (parameter ID 0x2F).<br>Refer to "Travel Range and Soft Limits" (p. 47) for further information. |

| | |
|---|---|
| | The motion can be stopped by #24 (p. 154), STP (p. 231) and HLT (p. 197).<br><br>Motion commands such as FED are not permitted when HID control is activated for the axis. Refer to "Controlling with an HID" (p. 114) for further information. |
| Available edge types and parameters: | The following edge types with their parameter settings are available:<br><br>1 = negative limit switch, <Param> must be 0<br>2 = positive limit switch, <Param> must be 0<br>3 = reference switch, <Param> must be 0 |

### FRF (Fast Reference Move To Reference Switch)

| | |
|---|---|
| Description: | Starts a referencing move.<br><br>Moves the specified axis to the reference switch and sets the current position to a defined value. See below for details.<br><br>If multiple axes are specified in the command, they are started simultaneously. |
| Format: | FRF [{<AxisID>}] |
| Arguments: | <AxisID> is a controller's axis, all axes are affected if not specified. |
| Response: | None |
| Troubleshooting: | Illegal axis identifier |
| Notes: | Servo mode must be switched on with SVO (p. 232) for the commanded axis prior to using this command (closed-loop operation).<br>If the reference move was successful, absolute motion will then be possible in closed-loop operation.<br>The ***Reference Signal Type*** parameter (0x70) is evaluated for the reference move. Further information, see "Referencing" (p. 50).<br>The value of the ***Value At Reference Position*** parameter (0x16) is set as the current position when the axis is at the reference position.<br>You can use a digital input instead of the reference switch as source of the reference signal for the FRF command. For further information, see "Digital Input Signals" (p. 109).<br>The motion can be stopped by #24 (p. 154), STP (p. 231), |

and HLT (p. 197).
Use FRF? (p. 181) to check whether the reference move was successful.

### FRF? (Get Referencing Result)

Description: Queries whether the specified axis is referenced or not.

Format: FRF? [{<AxisID>}]

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

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

where

<uint> indicates whether the axis has been successfully referenced (=1) or not (=0).

Troubleshooting: Illegal axis identifier

### GOH (Go To Home Position)

Description: Moves the specified axis to the zero position.

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

The motion can be stopped by #24 (p. 154), STP (p. 231), and HLT (p. 197).

Format: GOH [{<AxisID>}]

Arguments: <AxisID>: Is one axis of the controller; if not specified, all axes are affected.

Response: None

Troubleshooting: Illegal axis identifier

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

The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 27).

### HDI? (Get Help For Interpretation Of DIA? Response)

Description: Shows descriptions and physical units for the measurands that can be queried with the DIA? command (p. 168).

Format: HDI?

Arguments: None

Response {<MeasureID>"="<Description>TAB<PhysUnit> LF}

where

<MeasureID> is the identifier of the measurand

<Description> is the name of the measurand

<PhysUnit> is the physical unit of the measurand.

### HDR? (Get All Data Recorder Options)

Description: Lists a help string which contains all information available on data recording (record options and trigger options, information on additional parameters and commands concerning data recording).

Format: HDR?

Arguments: None

Response #RecordOptions
{<RecOption>"="<DescriptionString>[ of <Channel>]}

#TriggerOptions
[{<TriggerOption>"="<DescriptionString>}]

#Parameters to be set with SPA
[{<ParameterID>"="<DescriptionString>}]

#Additional information
[{<Command description>"("<Command>")"}]

#Sources for Record Options
[{<RecOption>"="<Source>}]

end of help

Example: For the C-867, the response to HDR? reads as follows:

```
#RecordOptions
```

```
0=Nothing is recorded
1=Commanded Position of Axis
2=Actual Position of Axis
3=Position Error of Axis
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
81=Analog input (Channel = 1 - 8)
86=Number of fifo values
87=Interpolation data
90=active parameterset
91=actual frequency
92=p0
93=dia?
#TriggerOptions
0=default setting
1=any command changing position (e.g.,
MOV)
2=next command
3=external trigger
6=any command changing position (e.g.,
MOV), reset trigger after execution
7=with SMO command, reset trigger after
execution
#Additional information
4 record tables
8192 datapoints per table
end of help
```

Note: TriggerOptions = 0 (default) means that recording is triggered by the STE command (p. 230).

### HDT (Set HID Default Lookup Table)

Description: Assigns a lookup table to the specified axis of the specified HID.

Lookup tables are used while HID is controlling several motion variables of the C-867' axes, see HIA (p. 185) for details. A lookup table maps the displacement of an HID axis to the controlled motion variable (see HIE? (p. 188) for

| | |
|---|---|
| | further details). |
| Format: | HDT {<HIDeviceID> <HIDeviceAxis> <HIDTableID>} |
| Arguments: | <HIDeviceID> is an HID connected to the controller; see below for more details.<br><br><HIDeviceAxis> is an HID axis; see below for more details.<br><br><HIDTableID> is a controller's lookup table; see below for more details. |
| Response: | None |
| Notes: | Lookup tables are assigned with HDT only in the volatile memory (RAM) of the C-867. With the WPA command (p. 248), the currently valid assignment can be saved in the nonvolatile memory of the C-867. The DPA command (p. 171) can be used to reset the allocation of lookup tables in the volatile memory **and** in the nonvolatile memory to the default settings.<br>The C-867 supports one analog human interface device (identifier: 1) and up to 5 digital HIDs (identifiers: 2 to 6). Information on the supported HID operating elements can be queried with the HIS? command (p. 190).<br>Refer to "Commandable Elements" (p. 18) and "Connecting an HID " (p. 60) for further details. |
| Available lookup tables: | The C-867 provides the following lookup tables with 256 points each: |

| Identifier | Type |
|---|---|
| 1 | linear |
| 2 | parabolic (default) |
| 101 | user-defined |
| 102 | user-defined |

With the HIT command (p. 193), user-defined lookup tables can be filled with values.

### HDT? (Get HID Default Lookup Table)

| | |
|---|---|
| Description: | Queries the currently assigned lookup table for the specified axis of the specified HID. |
| Format: | HDT? [{<HIDeviceID> <HIDeviceAxis>}] |
| Arguments: | <HIDeviceID> is an HID connected to the controller; see HDT for more details.<br><br><HIDeviceAxis> is an HID axis; see HDT for more details. |

| | |
|---|---|
| Response: | {<HIDeviceID> <HIDeviceAxis>"="<HIDTableID>LF}<br><br>where<br><br><HIDTableID> is a controller's lookup table; see HDT for more details. |

**HIA (Configure Control Done By HID Axis)**

| | |
|---|---|
| Description: | Configures control of the C-867's axis by HID axes ("HID Control"):<br><br>Assigns a specified motion variable of the specified C-867 axis to an HID axis.<br><br>The HID control is activated or deactivated with the HIN command (p. 189). HIA can only be used when HID control is deactivated for the affected axis of the C-867. |
| Format: | HIA {<AxisID> <MotionParam> <HIDeviceID> <HIDeviceAxis>} |
| Arguments: | <AxisID> is one axis of the controller.<br><br><MotionParam> is a motion parameter of the controller's axis; see below for further information.<br><br><HIDeviceID> is an HID connected to the controller, see below for more details.<br><br><HIDeviceAxis> is an HID axis; see below for more details. |
| Response: | None |
| Values for <MotionParam>: | <MotionParam> specifies the motion parameter to be controlled and can take on the following values:<br>**0 - delete configuration**<br>Deletes the current configuration of the HID control. Can be sent without specifying the <HIDeviceID> and <HIDeviceAxis> in abbreviated notation as:<br>HIA <AxisID> 0<br>**1 - absolute target position**<br>The lookup table value corresponding to the current displacement of the HID axis is mapped onto the travel range of the C-867 axis to be controlled. The travel range limits are specified by the values of parameters 0x30 and 0x15 and can be queried with TMN? and TMX?<br>**2 - relative target position:**<br>Intended for use with AB rotary encoders or pulse generators (p. 60). Each pulse received (if applicable: each |

mechanical detent) triggers relative motion over the distance set with the SST command (p. 229).
Lookup tables are **not** used for controlling the relative target position.

**3 - velocity of the controller 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. The currently valid maximum velocity is specified by one of the following sources:

- Parameter 0x74
- Parameter 0x49, when parameter 0x74 is zero
- Displacement of an HID axis, see below.

**4 - maximum velocity of the controller axis**
Product from the lookup table value corresponding to the current displacement of the HID axis and the value of the ***Closed-Loop Velocity For HI Control*** parameter (ID 0x74). If the parameter 0x74 has the value zero, the lookup table value is multiplied by the value of the parameter 0x49.

Notes: If an axis of the C-867 is configured for HID control of the absolute or relative target position: The current configuration must be deleted by sending HIA with the value zero for <MotionParam> before a new configuration can be set.

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

- <MotionParam> has the value zero, i.e., a function to be controlled has not been selected for the C-867's axis
- <HIDeviceID> has the value zero, i.e., an HID is not selected for HID control
- <HIDeviceAxis> has the value zero, i.e., HID's axis has not been selected for HID control

HID control is only configured with HIA in the volatile memory (RAM) of the C-867. With the WPA command (p. 248), the currently valid configuration can be saved in the nonvolatile memory of the C-867. The DPA command (p. 171) can be used to reset the configuration of the HID control in the volatile memory **and** in the nonvolatile memory to the default settings.

The C-867 supports one analog human interface device (identifier: 1) and up to 5 digital HIDs (identifiers: 2 to 6). Information on the supported HID operating elements can be queried with the HIS? command (p. 190).

Refer to "Commandable Elements" (p. 18) and "Connecting an HID " (p. 60) for further details.

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

Description: Queries the current motion variable for the specified motion parameter of the specified C-867 axis, i.e., the currently assigned HID axis.

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

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

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

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

where

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

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

### HIB? (Get State Of HID Button)

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

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

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

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

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

where

<HIDButtonState> specifies the status of the button as integer value:
The possible values depend on the button type. The value range for the individual buttons can be queried with the HIS? command (p. 190). When only the values 0 and 1 are permitted, they have the following meaning:
0 = Button not pressed, 1 = Button pressed

| | |
|---|---|
| | The meaning of values > 1 depends on the HID. |
| Notes: | The C-867 supports one analog human interface device (identifier: 1) and up to 5 digital HIDs (identifiers: 2 to 6). Information on the supported HID operating elements can be queried with the HIS? command (p. 190).<br>Refer to "Commandable Elements" (p. 18) and "Connecting an HID " (p. 60) for further details. |

**HIE? (Get Deflection Of HID Axis)**

| | |
|---|---|
| Description: | Queries the current displacement of the specified axis of the specified HID. |
| Format: | HIE? [{<HIDeviceID> <HIDeviceAxis>}] |
| Arguments: | <HIDeviceID> is an HID connected to the controller; see below for more details.<br><br><HIDeviceAxis> is an HID axis; see below for more details. |
| Response: | {<HIDeviceID> <HIDeviceAxis> "="<HIDDeflection>}<br><br>where<br><br><HIDDeflection> specifies the current HID axis displacement, see below for more details. |
| Notes: | The C-867 supports one analog human interface device (identifier: 1) and up to 5 digital HIDs (identifiers: 2 to 6). Information on the supported HID operating elements can be queried with the HIS? command (p. 190).<br>Refer to "Commandable Elements" (p. 18) and "Connecting an HID " (p. 60) for further details.<br><br><HIDDeflection> specifies the current HID axis displacement as floating point number in a range from -1.0 to 1.0 an.<br><br>For HID axes with hard stops, the value -1.0 corresponds to the maximum displacement in the negative direction and the value 1.0 corresponds to the maximum displacement in the positive direction.<br><br>The C-867 prepares the information received by the HID so that 256 different displacement values can be shown. When HID control for a motion variable is based on lookup tables, each of the displacement values is assigned to exactly one point in the currently assigned lookup table |

(see HDT (p. 183) and HIT (p. 193) for more details).

Example:

| | |
|---|---|
| Send: | `HIE? 1 1 1 2` |
| Receive: | `1 1=0.02`<br>`1 2=-0.7` |
| Note: | Displacement of axes 1 and 2 of HID 1:<br>Axis 1 has the value 0.02, which corresponds to approximately the center position.<br>Axis 2 has the value -0.7, i.e., it is displaced in the negative direction by around 2/3. |

### HIN (Set Activation State For HID Control)

Description: Activates or deactivates control of the specified C-867 axis by HIDs ("HID control") connected to the controller.

The HID control is configured with the HIA command (p. 185).

Format: HIN {<AxisID> <HIDControlState>}

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

<HIDControlState> is the HID control activation status:

0 = control by HID is deactivated

1 = control by HID is activated

Response: None

Notes: The C-867 supports up to 6 HIDs (identifiers: 1 for an analog human interface device, 2 to 6 for digital HIDs).

Refer to"Connecting an HID" (p. 60) for information on the connection options.

The enabled HID control will not have any effect if it has not been configured suitably with HIA.

During HID control of the velocity or the maximum velocity, the target position of the controlled axis of the C-867 is set to the soft limit that is specified by the parameter 0x15 or 0x30. Details on the parameters can be found in "Travel Range and Soft Limits" (p. 47).

When HID control is deactivated, the target position is set to the current position of the controlled axis.

No HID control is possible in open-loop operation (servo mode switched off).

Motion commands such as MOV (p. 213) and the execution of trajectories (TGS (p. 237)) are not permitted when HID control is activated for the axis. Refer to "Controlling with an HID" (p. 114) for further information.

**HIN? (Get Activation State Of HID Control)**

Description: Queries the activation status of the control by HIDs ("HID control") connected to the controller for the specified axis of the C-867.

Format: HIN? [{<AxisID>}]

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

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

where

<HIDControlState> is the HID control activation status:
0 = control by HID is deactivated
1 = control by HID is activated

**HIS? (Get Configuration Of HI Device)**

Description: Queries the specified property for the specified operating element of an HID.

Format: HIS? [{<HIDeviceID> <HIDItemID> <HIDPropID>}]

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

<HIDItemID> is an HID operating element, see below for more details.

<HIDPropID> is a property of an HID operating element; see below for more details.

Response: {<HIDeviceID> <HIDItemID> <HIDPropID>"="<HIDPropValue>LF}

where

<HIDPropValue> is a string with the value that the property of the operating element is set to; see below for more details.

HIDItemID: Supported operating elements:
The C-867 supports up to 6 HIDs (identifiers: 1 for an analog human interface device, 2 to 6 for digital HIDs). Refer to"Connecting an HID" (p. 60) for information on the

connection options.

All supported HID operating elements are numbered consecutively for the <HIDItemID> starting with 1 irrespective of their type.

HIDPropValue: Properties of the operating elements:

For <HIDPropID> = 1:

<HIDPropValue> indicates the type and identifier of the operating element. Possible types:

- "Axis" = HID axis, can be a joystick axis or an AB rotary encoder or pulse generator for example
- "Button" = HID button, can be a pushbutton for example

The type is followed by the identifier, separated by an underscore. The identifier must be used in all relevant commands, in order to specifically address the operating element.

For <HIDPropID> = 2:

<HIDPropValue> is the value for the current status of the operating element. The meaning of the value depends on the type of operating element:

- "Axis": Current displacement of the axis
- "Button": Current status of the button:
  - 0: Inactive
  - 1: Active

For <HIDPropID> = 3: For HID axes only

<HIDPropValue> is the name of an HID axis

For <HIDPropID> = 4:

<HIDPropValue> the of an HID

For <HIDPropID> = 5: For HID buttons only

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

For <HIDPropID> = 6: For HID buttons only

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

Example:

```
>>his?
<<1 1 1=Axis_1
<<1 1 2=0.117
<<1 1 3=X
<<1 1 4=Analog Joystick input
<<1 2 1=Axis_2
<<1 2 2=0.109
```

```
<<1 2 3=Y
<<1 2 4=Analog Joystick input
<<1 3 1=Axis_3
<<1 3 2=-0.016
<<1 3 3=X 10V
<<1 3 4=Analog Joystick input
<<1 4 1=Axis_4
<<1 4 2=-0.016
<<1 4 3=Y 10V
<<1 4 4=Analog Joystick input
<<1 5 1=Axis_5
<<1 5 2=0.000
<<1 5 3=AB1
<<1 5 4=Analog Joystick input
<<1 6 1=Axis_6
<<1 6 2=0.000
<<1 6 3=AB2
<<1 6 4=Analog Joystick input
<<1 7 1=Button_1
<<1 7 2=0
<<1 7 4=Analog Joystick input
<<1 7 5=0
<<1 7 6=1
<<1 8 1=Button_2
<<1 8 2=0
<<1 8 4=Analog Joystick input
<<1 8 5=0
<<1 8 6=1
<<2 1 1=Axis_1
<<2 1 2=0.000
<<2 1 3=X
<<2 1 4=USB
<<2 2 1=Axis_2
<<2 2 2=0.000
<<2 2 3=Y
<<2 2 4=USB
<<2 3 1=Button_1
<<2 3 2=0
<<2 3 4=USB
<<2 3 5=0
<<2 3 6=1
<<2 4 1=Button_2
<<2 4 2=0
<<2 4 4=USB
<<2 4 5=0
```

```
<<2 4 6=1
<<2 5 1=Button_3
<<2 5 2=0
<<2 5 4=USB
<<2 5 5=0
<<2 5 6=1
```

Note: Although the analog HID supported by C-867 is called "Analog Joystick input", HID axes 5 and 6 require digital input signals (AB, TTL), refer to "Connecting an HID" (p. 60).

**HIT (Fill HID Lookup Table)**

Description: Fills the specified lookup table with values.

Lookup tables are used while HID is controlling several motion variables of the C-867' axes, see HIA (p. 185) for details. A lookup table maps the displacement of an HID axis to the controlled motion variable (see HIE? (p. 188)) for further details.

The HDT (p. 183) command assigns the lookup tables to HID axes.

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

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

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

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

Response: None

Notes: HIT only fills the lookup tables in the volatile memory (RAM) of the C-867. With the WPA command (p. 248), the currently valid table contents can be saved in the nonvolatile memory of the C-867. The DPA command (p. 171) can be used to reset the table contents in the volatile memory **and** in the nonvolatile memory to the default settings.

Per HIT command, the value of one point can be sent to the C-867.

Available lookup tables: The C-867 provides the following lookup tables with 256 points each:

| Identifier | Type |
|---|---|
| 1 | linear |
| 2 | parabolic (default) |
| 101 | user-defined |
| 102 | user-defined |

HIT can only be used to fill user-defined tables. Tables with identifier ≤ 100 identifier are predefined and write-protected.

The first point of a lookup table corresponds to the maximum axis displacement of the HID in the negative direction; the 256th point corresponds to the maximum displacement in the positive direction. The values for points 1 to maximally 127 have a negative sign by default, while the remaining values have a positive sign.

When HID control is activated, the direction of motion for the C-867's axis can be reversed with the ***Invert Direction Of Motion For Joystick-Controlled Axis?*** parameter (ID 0x61).

### HIT? (Get HID Lookup Table Values)

Description: Queries the values of the specified points in the specified lookup table.

Format: HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]

Arguments: <StartPoint> is the index of the first point to be queried in the lookup table, smallest possible value is 1.

<NumberOfPoints> specifies the number of the points to be queried per lookup table; see HIT for more details.

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

Response: The lookup table content in GCS array format, see the separate manual for GCS array, SM 146E, and the example below.

Example:

```
hit?
# TYPE = 1
#
# SEPARATOR = 32
# DIM = 4
```

```
# NDATA = 256
# NAME0 = Table 1
# NAME1 = Table 2
# NAME2 = Table 101
# NAME3 = Table 102
# END_HEADER
-1.0000 -1.0000 -1.0000 -1.0000
-0.9922 -0.9834 -0.9834 -0.9834
-0.9834 -0.9678 -0.9678 -0.9678
-0.9756 -0.9521 -0.9521 -0.9521
-0.9678 -0.9355 -0.9355 -0.9355
...
-0.7314 -0.5352 -0.5352 -0.5352
-0.7236 -0.5234 -0.5234 -0.5234
-0.7158 -0.5117 -0.5117 -0.5117
-0.7070 -0.5000 -0.5000 -0.5000
-0.6992 -0.4893 -0.4893 -0.4893
...
-0.5605 -0.3145 -0.3145 -0.3145
-0.5527 -0.3057 -0.3057 -0.3057
-0.5449 -0.2969 -0.2969 -0.2969
-0.5361 -0.2881 -0.2881 -0.2881
-0.5283 -0.2793 -0.2793 -0.2793
-0.5205 -0.2705 -0.2705 -0.2705
...
-0.3496 -0.1221 -0.1221 -0.1221
-0.3418 -0.1162 -0.1162 -0.1162
-0.3330 -0.1113 -0.1113 -0.1113
-0.3252 -0.1055 -0.1055 -0.1055
-0.3174 -0.1006 -0.1006 -0.1006
...
-0.1465 -0.0215 -0.0215 -0.0215
-0.1387 -0.0195 -0.0195 -0.0195
-0.1299 -0.0166 -0.0166 -0.0166
-0.1221 -0.0146 -0.0146 -0.0146
-0.1143 -0.0127 -0.0127 -0.0127
...
-0.0244 -0.0010 -0.0010 -0.0010
-0.0166 0.0000 0.0000 0.0000
-0.0078 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
```

```
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0078 0.0000 0.0000 0.0000
0.0166 0.0000 0.0000 0.0000
0.0244 0.0010 0.0010 0.0010
0.0322 0.0010 0.0010 0.0010
0.0410 0.0020 0.0020 0.0020
...
0.1299 0.0166 0.0166 0.0166
0.1387 0.0195 0.0195 0.0195
0.1465 0.0215 0.0215 0.0215
0.1543 0.0234 0.0234 0.0234
0.1631 0.0264 0.0264 0.0264
...
0.2764 0.0762 0.0762 0.0762
0.2842 0.0811 0.0811 0.0811
0.2930 0.0859 0.0859 0.0859
0.3008 0.0908 0.0908 0.0908
0.3086 0.0957 0.0957 0.0957
...
0.4883 0.2383 0.2383 0.2383
0.4961 0.2461 0.2461 0.2461
0.5039 0.2539 0.2539 0.2539
0.5117 0.2627 0.2627 0.2627
0.5205 0.2705 0.2705 0.2705
...
0.6914 0.4775 0.4775 0.4775
0.6992 0.4893 0.4893 0.4893
0.7070 0.5000 0.5000 0.5000
0.7158 0.5117 0.5117 0.5117
0.7236 0.5234 0.5234 0.5234
...
0.9678 0.9355 0.9355 0.9355
0.9756 0.9521 0.9521 0.9521
0.9834 0.9678 0.9678 0.9678
0.9922 0.9834 0.9834 0.9834
1.0000 1.0000 1.0000 1.0000
```

### HLP? (Get List Of Available Commands)

| | |
|---|---|
| Description: | Lists a help string which contains all commands available. |
| Format: | HLP? |
| Arguments: | none |
| Response: | List of commands available |
| Troubleshooting: | Communication breakdown |

### HLT (Halt Motion Smoothly)

| | |
|---|---|
| Description: | Stops the motion of specified axes smoothly. See the notes below for further details.<br><br>Error code 10 is set.<br><br>#24 (p. 154) and STP (p. 231) in contrast abort current motion as fast as possible for the controller without taking care of maximum velocity and acceleration. |
| Format: | HLT [{<AxisID>}] |
| Arguments: | <AxisID>: is one axis of the controller, if left out, all axes are stopped |
| Response: | none |
| Troubleshooting: | Illegal axis identifier |
| Notes: | HLT stops motion with specified system deceleration regarding system inertia. Does not apply to trajectories.<br><br>HLT stops all motion caused by motion commands (e.g., MOV (p. 213), MVR (p. 215), GOH (p. 181), STE (p. 230), SMO (p. 224)), the command for referencing (FRF (p. 180)), and macros (MAC (p. 207)).<br><br>After the axes are stopped, their target positions are set to their current positions.<br><br>Setting the ***Controller Disable Error 10*** parameter (0xE000301) prevents the error code 10 from being output when an axis motion is stopped with HLT. |

### HPA? (Get List Of Available Parameters)

Description: Responds with a help string that contains all available parameters with short descriptions. Refer to "Parameter Overview" (p. 282) for further information.

Format: HPA?

Arguments: None

Response {<PamID>"="<string> LF}

where

<PamID> is the ID of one parameter, hexadecimal format

<string> is a string which describes the corresponding parameter.

The string has following format:

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

where

<CmdLevel> is the command level which allows write access to the parameter value.

<MaxItem> is the maximum number of items of the same type which are affected by the parameter. With the C-867, an "item" is an axis or the entire system.

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

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

<ParameterDescription> is the parameter name.

<PossibleValue> is one value from the allowed data range.

<ValueDescription> is the meaning of the corresponding value.

The parameters listed with HPA? can be changed and/or saved using the following commands:

SPA (p. 226) influences the parameter settings in volatile memory (RAM).

WPA (p. 248) copies parameter settings from volatile to nonvolatile memory.

SEP (p. 222) writes parameter settings directly into nonvolatile memory (without changing settings in volatile memory).

RPA (p. 219) resets volatile memory to the values from nonvolatile memory.

DPA (p. 171) resets parameter values and parameter-independent settings to the default settings.

### HPV? (Get Parameter Value Description)

Description: Responds with a help string that contains possible parameter values. Use HPA? instead to get a help string that contains all available parameters with short descriptions.

Format: HPV?

Arguments: None

Response: <string> has the following format:

"#Possible parameter values are:
{<PamID> <ItemID> "=" <ListType>
[ {TAB <PossibleValue> "=" <ValueDescription>} ] }
#CCL levels are:
{<PamID> <ItemID> "="<CmdLevel> }
#HPA_Category enabled
end of help"

where

<PamID> is the ID of one parameter, hexadecimal format

<ItemID> is one item (axis, channel, whole system) of the controller, if item=0 the description applies to all items

<ListType> determines how the possible parameter values listed in the string have to be interpreted:
- 0 = parameter not applicable for this item
- 1 = enumeration
- 2 = min/max

<PossibleValue> is a value from the permissible data range

<ValueDescription> is the meaning of the corresponding

value

Some parameters are write protected (by a command level > 1) for certain items. These parameters are listed below the “#CCL levels are” line.

<CmdLevel> is the command level that allows write access to the parameter value.

The "#HPA_Category enabled" line is evaluated by the PC software for display purposes.

**IFC (Set Interface Parameters Temporarily)**

Description: Configures interface parameters.

The new settings are enabled after IFC is sent. The PC interface configuration may have to be modified (closing of the current connection and re-opening with the new settings necessary).

Settings made with IFC are lost when the C-867 is switched off or rebooted. The defaults can be changed in the nonvolatile memory with the IFS command (p. 202).

Format: IFC {<InterfacePam> <PamValue>}

Arguments: <InterfacePam> is the interface parameter to be changed, see below

<PamValue> specifies the value of the interface parameter, see below

Interface parameters and values: <InterfacePam>: Interface parameters

DEVADR: Controller address

RSBAUD: Baud rate for the RS-232 communication

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

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

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

IPMASK: Subnet mask for TCP/IP communication
Formats:

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

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

<PamValue>: Possible and default values for the interface parameters are:
DEVADR:
Possible values: 1 to 16
default: 1

RSBAUD:
Possible values: 9600, 19200, 38400, 57600, and 115200
Default: 115200.

IPADR
Default port: 50000, cannot be changed
Note: The C-867 will use the address specified by IPADR only if IPSTART = 0.

IPMASK:
Default: 24 (= subnet mask 255.255.255.0)

Response: None

Notes: The interface parameters can be changed in the nonvolatile memory with the IFS command (p. 202).

**IFC? (Get Current Interface Parameters)**

Description: Queries the values of the interface parameters for communication from volatile memory.

Format: IFC? [{<InterfacePam>}]

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

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

where

<PamValue> gives the value of the interface parameter

| | |
|---|---|
| | from volatile memory. |
| Interface parameters | DEVADR: Controller address<br><br>RSBAUD: Baud rate for the RS-232 communication<br><br>IPSTART: Startup behavior for configuring the IP address for TCP/IP communication:<br>▪ 0 = The address is used that is defined by IPADR.<br>▪ 1 = DHCP is used to get the IP address.<br><br>IPADR: IP address with port for TCP/IP communication<br>Format: <IP address>:<port number><br><br>IPMASK: Subnet mask for TCP/IP communication<br>Formats:<br>▪ uint.uint.uint.uint<br>e.g.: 255.255.0.0<br>▪ <decimal number><br>= Number of bits of the subnet mask (specifies how many bits make up the beginning of the IP address in the network component)<br>e.g.: 16<br><br>IPGTWAY: Standard gateway for TCP/IP communication<br>Format: uint.uint.uint.uint |
| Notes: | For communication via TCP/IP, the current used IP address and the startup behavior partially depend on the network type. For that reason, the settings for IPADR and IPSTART may be ignored:<br><br>If a DHCP server is in the network, the IPSTART setting is ignored, and the IP address is always obtained from the DHCP server.<br><br>If the C-867 is directly connected to the Ethernet card in the PC (no DHCP server is present), the current IP address of the C-867 will be as follows:<br>for IPSTART = 0, the IPADR setting will be used<br>for IPSTART = 1, the default value 192.168.0.1 will be used.<br>Note that when the C-867 is part of a network with DHCP server, the IPADR indicates the static IP address of the C-867 in the response to IFC? but not the currently used IP address that was obtained from the DHCP server. |

### IFS (Set Interface Parameters as Default Values

| | |
|---|---|
| Description: | Stores interface parameters.<br><br>The default parameters for the interface are changed in the nonvolatile memory, but the currently active parameters are not. Settings made with IFS become active after the next switch-on or reboot. |
| Format: | IFS <Pswd> {<InterfacePam> <PamValue>} |
| Arguments: | <Pswd> is the password for writing to the nonvolatile memory, default value is "100"<br><br><InterfacePam> is the interface parameter to be changed, see below<br><br><PamValue> specifies the value of the interface parameter, see below |
| Response: | none |
| Interface parameters and values: | <InterfacePam>: Interface parameters<br><br>DEVADR: Controller address<br><br>RSBAUD: Baud rate for the RS-232 communication<br><br>IPSTART: Startup behavior for configuring the IP address for TCP/IP communication:<br>▪ 0 = The address is used that is defined by IPADR.<br>▪ 1 = DHCP is used to get the IP address.<br><br>IPADR: IP address with port for TCP/IP communication<br>Format: <IP address>:<port number><br><br>IPMASK: Subnet mask for TCP/IP communication<br>Formats:<br>▪ uint.uint.uint.uint<br>e.g.: 255.255.0.0<br>▪ <decimal number><br>= Number of bits of the subnet mask (specifies how many bits make up the beginning of the IP address in the network component)<br>e.g.: 16<br><br>IPGTWAY: Standard gateway for TCP/IP communication<br>Format: uint.uint.uint.uint<br><br><PamValue>: Possible and default values for the interface |

parameters are:

DEVADR:
Possible values: 1 to 16
default: 1

RSBAUD:
Possible values: 9600, 19200, 38400, 57600, and 115200
Default: 115200.

IPADR
Default port: 50000, cannot be changed
Note: The C-867 will use the address specified by IPADR only if IPSTART = 0.

IPMASK:
Default: 24 (= subnet mask 255.255.255.0)

| | |
|---|---|
| Response: | None |
| Notes: | The interface parameters can be changed in the volatile memory with the IFC command (p. 200). Changes in the volatile memory become effective immediately. |

### IFS? (Get Interface Parameters as Default Values)

| | |
|---|---|
| Description: | Queries the parameter values of the interface configuration stored in the nonvolatile memory (i.e. default settings) |
| Format: | IFS? [{<InterfacePam>}] |
| Arguments: | <InterfacePam> is the interface parameter to be queried. See below for possible values. |
| Response: | {<InterfacePam>"="<PamValue> LF}<br><br>where<br><br><PamValue> is the value of the interface parameter in nonvolatile memory. |
| Interface parameters | DEVADR: Controller address<br><br>RSBAUD: Baud rate for the RS-232 communication<br><br>IPSTART: Startup behavior for configuring the IP address for TCP/IP communication:<br>▪ 0 = The address is used that is defined by IPADR.<br>▪ 1 = DHCP is used to get the IP address. |

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

IPMASK: Subnet mask for TCP/IP communication
Formats:

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

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

MACADR: Ethernet address (= unchangeable, unique address of the network hardware) in the C-867

**JRC (Jump Relatively Depending On Condition)**

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

Can only be used in macros.

Format: JRC <Jump> <CMD?> <OP> <Value>

Arguments: <Jump> is the size of the relative jump. -1 means that the macro execution pointer jumps back to the previous line; 0 means that the command is executed again, which is the same behavior as with WAC (p. 247). 1 jumps to the next line, making the command unnecessary, and 2 jumps over the next command. Jumps are only permitted in the current macro.

<CMD?> is one query command in its usual notation. The response must be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:
= <= < > >= !=
Important: There must be a space before and after the operator!

<Value> is the value to be compared with the response to

<CMD?>.

Response: none

Troubleshooting: Check proper jump target

Example: Using the following macro, you can stop motion of axis "1" using a stop button connected to a digital input. The stop button is checked until the axis has reached the target position (query ONT?). When the stop button is pressed before the target position has been reached: The response to the POS? query is copied into the TARGET variable. This variable is then used as second argument for the MOV command. Therefore, the positioner just stays where it was. To clean up, TARGET is defined as empty with the VAR command which deletes the variable.

Write the "stop" macro:

```
MAC BEG stop
MOV 1 20
JRC 2 DIO? 1 = 1
JRC -1 ONT? 1 = 0
CPY TARGET POS? 1
MOV 1 ${TARGET}
VAR TARGET
MAC END
```

**LIM? (Indicate Limit Switches)**

Description: Queries whether axes have limit switches.

Format: LIM? [{<AxisID>}]

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

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

where

<uint> indicates whether the axis has limit switches (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The C-867 firmware detects the presence or absence of limit switches using a parameter (ID 0x32). According to the value of this parameter, the C-867 enables or disables the stopping of the motion at the limit switches.

Adapt the parameter value to your hardware using SPA (p. 226) or SEP (p. 222). For further information, see "Limit Switch Detection" (p. 46).

You can use the digital input lines instead of the limit switches as source of the negative or positive limit switch signal. Further information see "Digital Input Signals" (p. 109).

### MAC (Call Macro Function)

Description: Calls a macro function. Permits recording, deleting and running macros on the controller.

Format: MAC <keyword> {<parameter>}

in particular:

MAC BEG <macro name>
MAC DEF <macro name>
MAC DEF?
MAC DEL <macro name>
MAC END
MAC ERR?
MAC FREE?
MAC NSTART <macro name> <uint> [<String1> [<String2>]]
MAC START <macro name> [<String1> [<String2>]]

Arguments: <keyword> determines which macro function is called. The following keywords and parameters are used:

MAC BEG <macroname>
Starts recording a macro to be named *macroname* on the controller; may not be used in a macro; the commands that follow become the macro. End the recording with MAC END. Note that erroneous macro content cannot be detected by sending the ERR? command.

MAC END
Stops macro recording (cannot become part of a macro)

MAC ERR?
RepoReports the last error that occurred while the macro was running.
Response: <macroname> <uint1>”=”<uint2> <”<”CMD”>”>
where <macroname> is the name of the macro, <uint1> is the line in the macro, <uint2> is the error code and <”<”CMD”>”> is the erroneous command which was sent to the parser.

MAC DEF <macroname>
Sets specified macro as startup macro. This macro will be run automatically after the next switch-on or reboot of the controller. If <macroname> is not specified, the current startup macro selection is canceled.

MAC DEF?

Asks for the startup macro
Response: <macroname>
If a startup macro is not defined, the response is an empty string with the terminating character.

MAC DEL <macroname>

Deletes specified macro.

MAC FREE?

Gets the free memory space for macro recording

Response: <uint> is the number of characters in bytes for which free memory is still available

MAC NSTART <macro name> <uint> [<String1> [<String2>]]

Repeats the specified macro <uint> times. The macro is re-run each time.

<String1> and <String2> are optional arguments which specify the values for local variables 1 and 2 used in the specified macro. <String1> and <String2> can be specified directly or via the values of variables. Macro will not run if the macro contains local variables but <String1> and <String2> are not specified in the MAC NSTART command. Refer to “Variables” (p. 146) for further details.

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

Runs the specified macro once. <String1> and <String2> have the same function as with MAC NSTART.

Response: None

Troubleshooting: Macro recording is active (keywords BEG, DEL) or inactive (END)

Macro contains a disallowed MAC command

Notes: Running a macro is not allowed when a macro is being recorded.

When a macro is recorded for a controller whose address is different from 1, the target address must be part of each command line, but will not become part of the macro content. PIMikroMove automatically sends the target address during the macro recording so that it does not have to be entered there. You will find further information in "Working with Macros" (p. 128) and "Target and Sender Address" (p. 145).

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

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

Macros can contain local and global variables. Refer to

"Variables" (p. 146) for further information.

A running macro sends no responses to any interface.

Depending on the value of parameter 0x72 (***Ignore Macro Error?***), the following options exist when an error is caused by a running macro:

0 = Macro running is aborted (default).
1 = The error is ignored and the macro continues to run.

MAC ERR? always reports the last error that occurred while the a macro was running irrespective of the parameter setting.

The following commands provided by the C-867 can only be used in macros:
DEL (p. 166), JRC (p. 205), MEX (p. 212) and WAC (p. 247).

A macro can start another macro. The maximum number of nesting levels is 5. A macro can call itself to form an infinite loop.

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

Macro execution can be stopped with #24 (p. 154) and STP (p. 231).

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

A macro cannot be deleted while it is running.

You can query with #8 (p. 153) if a macro is currently running on the controller.

**Note: The number of write cycles in the nonvolatile memory is limited. Only record macros if this is necessary.**

### MAC? (List Macros)

Description: Lists macros or content of a specified macro.

Format: MAC? [<macroname>]

| | |
|---|---|
| Arguments | <macroname>: name of the macro where the content is to be listed; if not specified, the names of all stored macros are listed. |
| Response: | <string><br><br>If <macroname> was specified, <string> is the content of this macro;<br><br>If <macroname> was not specified, <string> is a list with the names of all stored macros |
| Troubleshooting: | Macro <macroname> not found |

### MAN? (Get Help String For Command)

Description: Shows a detailed help text for individual commands.

Format: MAN? <CMD>

Arguments: <CMD> is the command mnemonic of the command for which the help text is to be displayed (see below).

Response: A string that describes the command.

Notes: A detailed help text can be displayed for the following GCS commands:
CTO, CTO?, HIA, HIA?, HIS?, HIT, HIT?, WPA

Example: Send: `MAN? CTO`

Receive:

```
CTO {<TrigOutID> <CTOPam> <Value>} Set
Configuration Of Trigger Output
#AvailableCTOparameters
<CTOPam> <Description>
1 Trigger Step
2 Axis
3 Trigger Mode
7 Polarity
8 Start Threshold
9 Stop Threshold
10 Trigger Position
11 PulseWidth (only for HardwareTrigger)
#AvailableTriggerModes
<Value> <Description>
0 Position Distance
2 On Target
```

```
5 Motion Error
6 In Motion
7 Position+Offset
8 Single Position
9 HardwareTrigger (only for output line
1)
#AvailablePolarities
<Value> <Description>
0 Active Low
1 Active High
end of help
```

### MAT (Calculate And Save To Variable)

Description: Carries out a mathematical operation or bit operation and saves the result as a variable (p. 146).

The variable is in volatile memory (RAM) only.

Format: MAT <Variable> "=" <FLOAT1> <OP> <FLOAT2>

Arguments: <Variable> is the name of the variable where the result is to be saved.

<FLOAT1> and <FLOAT2> are the values for calculating the result. They can be specified directly or via the value of a variable.

<OP> is the operator to be used: The following operators are possible:

| <OP> | Operation | Type |
|---|---|---|
| + | Addition | Mathematical operation |
| - | Subtraction | Mathematical operation |
| * | Multiplication | Mathematical operation |
| AND | UND | Bit operation |
| OR | ODER | Bit operation |
| XOR | XOR | Bit operation |

Important: There must be a blank space before and after each "=" and the operator!

Response: None

Notes: Using MAT to set local variables is only possible in macros.

Example 1: Send: `MAT TARGET = ${POS} * 2.0`
The `TARGET` variable contains 2.0 times the value of the `POS` variable.

| | |
|---|---|
| Example 2: | Send: `MAT TARGET = 2 * 0x10`<br>Send: `VAR? TARGET`<br>Receive: `TARGET=32`<br>NOTICE: The values from which the result is to be calculated can be written in hexadecimal or decimal format. The result is always output in decimal format. |
| Example 3: | Send: `MAT INVERT = 0x45 XOR 0xFF`<br>Send: `VAR? INVERT`<br>Receive: `INVERT=186`<br>NOTICE: The bit operation XOR with the value 0xFF corresponds to an inversion of the value 0x45. The result is output in decimal format. |

**MEX (Stop Macro Execution Due To Condition)**

Description: Stops running the macro due to a specified condition of the following type: a specified value is compared with a queried value according to a specified rule.

Can only be used in macros.

When the macro interpreter accesses this command, the condition is checked. If it is true, the current macro is stopped; otherwise the macro continues to run with the next line. Should the condition be fulfilled later, the interpreter will ignore it.

See also the WAC command (p. 247).

Format: MEX <CMD?> <OP> <Value>

Arguments: <CMD?> is one query command in its usual syntax. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:
= <= < > >= !=
Important: There must be a blank space before and after the operator!

<Value> is the value that is compared with the response to <CMD?>.

Response: None

Example: Send: `MAC START LOOP`

Note:
LOOP macro contains the following:
```
MAC START KEY1
MAC START KEY2
MEX DIO? 4 = 1
MAC START LOOP
```

KEY1 macro contains the following:
```
MEX DIO? 4 = 1
 MEX DIO? 1 = 0
 MVR 1 1.0
DEL 100
```

KEY2 macro contains the following:
```
MEX DIO? 4 = 1
MEX DIO? 2 = 0
 MVR 1 -1.0
DEL 100
```

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

KEY1 checks the state of the digital input channel 1 (is on the I/O socket (p. 316)). If it is not set (0), the macro is aborted, otherwise the macro will move axis 1 by 1.0 in positive direction (relative move).

KEY2 checks the state of the digital input channel 2 and moves axis 1 in negative direction accordingly.

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

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

### MOV (Set Target Position)

Description: Sets an absolute target position for the specified axis.

Format: MOV {<AxisID> <Position>}

| | |
|---|---|
| Arguments: | <AxisID> is one axis of the controller.<br><br><Position> is the absolute target position in physical units. |
| Response: | none |
| Notes: | The servo mode must be switched on when this command is used (closed-loop operation).<br><br>The target position must be within the soft limits. Use TMN? (p. 241) and TMX? (p. 241) to query the current valid soft limits.<br><br>The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 27).<br><br>The motion can be stopped by #24 (p. 154), STP (p. 231), and HLT (p. 197).<br><br>During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: Motion commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other. |
| Example 1: | Send: `MOV 1 10`<br>Note: Axis 1 moves to 10 (target position in mm) |
| Example 2: | Send: `MOV 1 243`<br>Send: `ERR?`<br>Receive: `7`<br>Note: The axis does not move. The error code "7" in the reply to the ERR? command (p. 178) indicates that the target position specified in the motion command is out of limits. |

**MOV? (Get Target Position)**

| | |
|---|---|
| Description: | Returns last valid commanded target position. |
| Format: | MOV? [{<AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller |
| Response: | {<AxisID>"="<float> LF}<br><br>where<br><br><float> is the last commanded target position in physical |

| | |
|---|---|
| | units |
| Troubleshooting: | Illegal axis identifier |
| Notes: | The target position can be changed by commands that cause motion (e.g., MOV (p. 213), MVR (p. 215), MVE, GOH (p. 230), STE (p. 181)) or by HID control (when HID control is disabled, the target position is set to the current position for HID-controlled axes in closed-loop operation).<br><br>MOV? gets the commanded positions. Use POS? (p. 217) to get the current positions. |

**MVR (Set Target Relative To Current Position)**

| | |
|---|---|
| Description: | Moves the specified axis relative to the last commanded target position. |
| Format: | MVR {<AxisID> <Distance>} |
| Arguments: | <AxisID> is one axis of the controller.<br><br><Distance> specifies the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units). |
| Response: | none |
| Notes: | The servo mode must be switched on when this command is used (closed-loop operation).<br><br>The target position must be within the soft limits. Use TMN? (p. 241) and TMX? (p. 241) to get the currently valid soft limits, and MOV? (p. 214) to get the current target.<br><br>The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 27).<br><br>The motion can be stopped by #24 (p. 154), STP (p. 231), and HLT (p. 197).<br><br>During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: Motion commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other. |
| Example: | Send: `MOV 1 0.5`<br>Note: This is an absolute motion. |

Send: `POS? 1`
Receive: `1=0.500000`
Send: `MOV? 1`
Receive: `1=0.500000`
Send: `MVR 1 2`
Note: This is a relative motion.
Send: `POS? 1`
Receive: `1=2.500000`
Send: `MVR 1 2000`
Note: New target position of axis 1 would exceed motion range. Command is ignored, i.e., the target position remains unchanged, and the axis does not move.
Send: `MOV? 1`
Receive: `1=2.500000`
Send: `POS? 1`
Receive: `1=2.500000`

**ONT? (Get On-Target State)**

Description: Queries the on-target state of the specified axis.

If all arguments are left out, queries state of all axes.

Format: ONT? [{<AxisID>}]

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

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

where

<uint> = "1" when the specified axis has reached the target value, otherwise "0".

Troubleshooting: Illegal axis identifier

Notes: The detection of the on-target state is only possible in closed-loop operation (servo mode ON).

The on-target state is influenced by the settings for the settling window (parameter 0x406 and 0x407) and the delay time (parameter 0x3F). For details, see "On-Target State" (p. 44).

### POS (Set Real Position)

Description: Sets the current position of the axis (does not cause motion).

Format: POS {<AxisID> <Position>}

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

<Position> is the new current position in physical units.

Response: None

Troubleshooting: Illegal axis identifier

Notes: It is only possible to set the current position with POS when the referencing method "0" is selected, see RON (p. 218).

An axis is considered to be "referenced" when the position has been set with POS (for more information, see "Referencing" (p. 50)).

The minimum and maximum commandable positions (TMN? (p. 241), TMX? (p. 241)) are not adapted when a position is set with POS. This can result in target positions which are allowed by the C-867 but cannot be reached by the hardware. Target positions are also possible that can be reached by the hardware but are refused by the C-867. Furthermore, the zero position can be outside of the physical travel range after using POS.

### POS? (Get Real Position)

Description: Queries the current axis position.

If no arguments are specified, the current position of all axes is queried.

Format: POS? [{<AxisID>}]

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

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

where

<float> is the current axis position in physical units.

Troubleshooting: Illegal axis identifier

### RBT (Reboot System)

Description: Reboots system. The controller behaves the same as after switching on.

Format: RBT

Arguments: none

Response: none

Notes: RBT cannot be used in macros. This is to avoid problems with startup macro execution.

### RMC? (List Running Macros)

Description: Lists macros which are currently running.

Format: RMC?

Arguments: None

Response: {<macroname> LF}

where

<macroname> is the name of one macro which is saved on the controller and currently running. The response is an empty line when no macro is running.

### RON (Set Reference Mode)

Description: Selects the referencing method for the specified axes

Format: RON {<AxisID> <ReferenceOn>}

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

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

Response: None

Troubleshooting: Illegal axis identifier

Notes: <ReferenceOn> = 0: To reference the axis, an absolute position value can be assigned with POS (p. 217) or a reference move can be started with FRF (p. 180). Relative motion is possible with MVR (p. 215), even when the axis has not been referenced.

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

Further information, see "Referencing" (p. 50).

**RON? (Get Reference Mode)**

| | |
|---|---|
| Description: | Queries referencing method of specified axes. |
| Format: | RON? [{ <AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller. |
| Response: | {<AxisID>"="<ReferenceOn> LF}<br><br>where<br><br><ReferenceOn> is the currently selected referencing method for the axis |
| Troubleshooting: | Illegal axis identifier |
| Note: | Further information can be found in the description of the RON command (p. 218). |

**RPA (Reset Volatile Memory Parameters)**

| | |
|---|---|
| Description: | Resets the specified parameter of the specified element. The value from nonvolatile memory is written into volatile memory.<br><br>Related commands:<br><br>With HPA? (p. 198) you can obtain a list of the available parameters. SPA (p. 226) influences the parameter settings in volatile memory, WPA (p. 248) writes parameter settings from volatile to nonvolatile memory, and SEP (p. 222) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).<br><br>See SPA for an example. |
| Format: | RPA [{<ItemID> <PamID>}] |

| | |
|---|---|
| Arguments: | <ItemID> is the element for resetting a parameter. See below for details.<br><br><PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details. |
| Response: | none |
| Troubleshooting: | Illegal element identifier, wrong parameter ID |
| Notes: | The information from the positioner's ID chip and the positioner databases are loaded into the volatile memory of the C-867. The loaded data is overwritten by RPA. Only use RPA if you are sure that the C-867 functions correctly with the parameter values from the nonvolatile memory.<br><br>With RPA, you can reset either all parameters or specific individual parameters for the C-867.<br><br>DPA (p. 171) resets parameter values and parameter-independent settings to the default settings. |
| Available item IDs and parameter IDs: | An item is an axis (the identifier can be changed with SAI (p. 221)) or the entire system. For further information, see "Commandable Items" (p. 18).<br><br>Valid parameter IDs are given in "Parameter Overview" (p. 282). |

**RTR (Set Record Table Rate)**

| | |
|---|---|
| Description: | Sets the record table rate, i.e., the number of cycles to be used in data recording operations. Settings larger than 1 make it possible to cover longer time periods. |
| Format: | RTR <RecordTableRate> |
| Arguments: | <RecordTableRate> is the record table rate to be used for recording operations (unit: number of cycles), must be an integer value larger than zero. |
| Response: | None |
| Notes: | The duration of the recording can be calculated as follows:<br><br>Rec. duration = cycle time of the servo loop * RTR value * number of points<br><br>where<br><br>the cycle time of the servo loop for the C-867 is 50 µs |

the number of points for the C-867 is 8192 (length of data recorder table)

For further information, see "Data Recorder" (p. 98).

The record table rate set with RTR is saved in volatile memory (RAM) only.

**RTR? (Get Record Table Rate)**

Description: Queries the current record table rate, i.e., the number of cycles used in data recording operations.

Format: RTR?

Arguments: None

Response: <RecordTableRate> is the table rate used for recording operations (unit: number of cycles).

**SAI (Set Current Axis Identifiers)**

Description: Sets the axis identifiers for the specified axes.

After it was set with SAI, the new axis identifier must be used as <AxisID> in all axis-related commands.

Format: SAI {<AxisID> <NewIdentifier>}

Arguments: <AxisID> is one axis of the controller

<NewIdentifier> is the new identifier to use for the axis, see below for details

Response: none

Notes: An axis could be identified with up to 8 characters. Use TVI? (p. 244) to ask for valid characters.
The new axis identifier is saved automatically and is therefore still available after rebooting or the next power-on.
DPA (p. 171) resets the axis identifier as well as parameter values and further parameter-independent settings to the default settings.

### SAI? (Get List Of Current Axis Identifiers)

| | |
|---|---|
| Description: | Queries the axis identifiers.<br><br>Refer also to "Commandable Elements" (p. 18). |
| Format: | SAI? [ALL] |
| Arguments: | [ALL] is optional. For controllers that allow deactivating the axis, [ALL] ensures that the response also includes the axes that are "deactivated". |
| Response: | {<AxisID> LF}<br><br><AxisID> is one axis of the controller. |
| Notes: | If the ***Stage Name*** parameter (0x3C) has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g. motion commands or position queries) and is only included in the response to `SAI? ALL`. |

### SEP (Set Non-Volatile Memory Parameters)

| | |
|---|---|
| Description: | Sets a parameter of a specified element to a different value in nonvolatile memory, where it becomes the new default.<br><br>After parameters were set with SEP, you can use RPA (p. 219) to activate them (write them to volatile memory) without controller reboot.<br><br>**Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!**<br><br>Related commands:<br><br>HPA? (p. 198) returns a list of the available parameters.<br><br>SPA (p. 226) writes parameter settings into volatile memory (without changing the settings in nonvolatile memory).<br><br>WPA (p. 248) writes parameter settings from volatile to nonvolatile memory. |
| Format: | SEP <Pswd> {<ItemID> <PamID> <PamValue>} |

| | |
|---|---|
| Arguments | <Pswd> is the password for writing to the nonvolatile memory; the default value is "100".<br><br><ItemID> is the element for changing a parameter in the nonvolatile memory. See below for details.<br><br><PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details.<br><br><PamValue> is the value for setting the specified parameter of the specified element. |
| Response: | None |
| Troubleshooting: | Illegal item identifier, wrong parameter ID, invalid password |
| Notes: | **Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.**<br><br>With the C-867, you can write up to two parameters per SEP command.<br><br>DPA (p. 171) resets parameter values and parameter-independent settings to the default settings. |
| Available item IDs and parameter IDs: | An item is an axis (the identifier can be changed with SAI (p. 221)) or the entire system. For further information, see "Commandable Items" (p. 18).<br><br>Valid parameter IDs are given in "Parameter Overview" (p. 282). |

### SEP? (Get Nonvolatile Memory Parameters)

| | |
|---|---|
| Description: | Queries the value of a parameter of a specified element from nonvolatile memory.<br><br>With HPA? (p. 198) you can obtain a list of the available parameters and their IDs. |
| Format: | SEP? [{<ItemID> <PamID>}] |
| Arguments: | <ItemID> is the element for querying a parameter value from nonvolatile memory. See below for details.<br><br><PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details. |

Response: {<ItemID> <PamID>"="<PamValue> LF}

where

<PamValue> is the value of the specified parameter for the specified element

Troubleshooting: Illegal element identifier, wrong parameter ID

Notes: With the C-867, you can query either all parameters or specific individual parameters with each SEP? command.

Available item IDs and parameter IDs: An item is an axis (the identifier can be changed with SAI (p. 221)) or the entire system. For further information, see "Commandable Items" (p. 18).

Valid parameter IDs are given in "Parameter Overview" (p. 282).

### SMO (Set Open-Loop Control Value)

Description: Sets control value directly to move the axis. Profile generator (if present), sensor feedback and servo algorithm are not taken into account.

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

Format: SMO {<AxisID> <ControlValue>}

Arguments <AxisID> is one axis of the controller.

<ControlValue> is the new control value (dimensionless). See below for details.

Response: None

Troubleshooting: Illegal axis identifier

Servo mode is switched on for one of the specified axes.

Notes: **NOTICE: In the case of large control values, the positioner can strike the hard stop despite the limit switch function. This can cause damage to equipment.**

The unsigned control value may not be greater than the value of the ***Maximum Motor Output*** parameter (0x9).

When this parameter is set to its maximum (32767), <ControlValue> ranges from -32766 to 32766 (dimensionless). <ControlValue> controls the piezo voltage for the axis. The sign of the value determines the direction of motion: -32766 corresponds to the maximum amplitude of the piezo voltage in the negative direction of motion and 32766 corresponds to the maximum amplitude of the piezo voltage in the positive direction of motion. For further information, see "Supported Motor Types" (p. 23).

When a high control value remains set over a long period of time, the connected positioner can heat up. Overheating can result in damage to the positioner.

The ***PID Maximum Output Time (s)*** parameter (ID 0x7B) specifies the maximum time period for which a high control value may be set in closed-loop operation. A high control value is present when the following applies:

Current absolute measure of the control value ≥ 95 % of ***Maximum Motor Output*** (ID 0x9). For further information, see "Protection Against Overheating" (p. 93).

The ***Range Limit Min*** (0x07000000) and ***Range Limit Max*** (0x07000001) parameters can be used as soft limits for motions in open-loop operation with SMO: When the current position reaches these values, the control value is set to zero and the motion is stopped. The axis can be moved again as soon as the value for the soft limit has been increased or decreased.

Example: Send: `SMO 1 -16000`

Note: The control value is about half the maximum control value. The axis moves in negative direction.

**SMO? (Get Control Value)**

Description: Gets last valid control value of given axis.

Format: SMO? [{<AxisID>}]

Arguments <AxisID> is one axis of the controller

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

where

<float> is the last valid control value (dimensionless). For details see below.

Troubleshooting: Illegal axis identifier

Notes: The control value which is returned by SMO? can be the result of the servo algorithm and other corrections, or it can be the value set by an SMO command in open-loop operation.

For further information, see SMO (p. 224) and the block diagram (p. 18).

**SPA (Set Volatile Memory Parameters)**

Description: Sets a parameter of the specified element in the volatile memory (RAM) to a specific value. Parameter changes are lost when the controller is switched off or rebooted.

Format: SPA {<ItemID> <PamID> <PamValue>}

Arguments: <ItemID> is the element for which a parameter is changed in volatile memory. See below for details.

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

<PamValue> is the value to which the specified parameter of the specified element is set.

Response: None

Parameter changes are also lost when the parameters are reset to their default values with RPA (p. 219).

**Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!**

Related commands:

HPA? (p. 198) returns a list of the available parameters.

SEP (p. 222) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).

WPA (p. 248) writes parameter settings from volatile to nonvolatile memory.

RPA resets volatile memory to the value in nonvolatile memory.

DPA (p. 171) resets parameter values and parameter-independent settings to the default settings.

| | |
|---|---|
| Troubleshooting: | Illegal item identifier, wrong parameter ID, value out of range |
| Notes: | With the C-867, you can write up to two parameters per SPA command. |
| Available item IDs and parameter IDs: | An item is an axis (the identifier can be changed with SAI (p. 221)) or the entire system. For further information, see "Commandable Items" (p. 18).<br><br>Valid parameter IDs can be found in the parameter overview (p. 282). |
| Example 1: | Send: `SPA 1 0x411 100`<br><br>Note: Sets the P term of the servo algorithm for axis 1 to 100 for parameter group 1; the parameter ID is written in hexadecimal format<br><br>Send: `SPA 1 1041 150`<br><br>Note: Sets the P term of the servo algorithm for axis 1 to 150 for parameter group 1; the parameter ID is written in decimal format |
| Example 2: | For parameter group 2, the P, I, and D parameters of the servo algorithm must be adapted to a new load that is applied to the connected mechanical system.<br><br>Send: `SPA 1 0x421 150`<br><br>Note: The P term is set to 150 for axis 1. The setting is made in volatile memory only.<br><br>Use SPA to set the I and D terms in volatile memory and then test the function of the system. If the closed-loop system performance proves satisfactory and you want to use this system configuration as default, save the parameter settings from volatile to nonvolatile memory.<br><br>Send: `WPA 100`<br><br>Note: See the command description for WPA (p. 248) for details on the extent of the saved settings. |

### SPA? (Get Volatile Memory Parameters)

| | |
|---|---|
| Description: | Queries the value of a parameter of a specified element from volatile memory (RAM).<br><br>You can obtain a list of the available parameters with HPA? (p. 198). |
| Format: | SPA? [{<ItemID> <PamID>}] |
| Arguments: | <ItemID> is the element for querying a parameter in volatile memory. See below for details.<br><br><PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details. |
| Response: | {<ItemID> <PamID>"="<PamValue> LF}<br><br>where<br><br><PamValue> is the value of the specified parameter for the specified element |
| Troubleshooting: | Illegal element identifier, wrong parameter ID |
| Notes: | With the C-867, you can query either all parameters or specific individual parameters for each SPA? command. |
| Available element IDs and parameter IDs: | An element is an axis (the identifier can be changed with SAI (p. 221)) or the entire system. For further information, see "Commandable Elements" (p. 18).<br><br>Valid parameter IDs can be found in the parameter overview (p. 282). |

### SRG? (Query Status Register Value)

| | |
|---|---|
| Description: | Returns register values for queried elements and registers. |
| Format: | SRG? [{<ItemID> <RegisterID>}] |
| Arguments: | <ItemID> is the element for querying a register. See below for details.<br><br><RegisterID> is the ID of the specified register; see below for available registers. |

Response: {<ItemID><RegisterID>"="<Value> LF}

where

<Value> is the value of the register; see below for more details.

Note: This command is identical in function to #4 (p. 151) which should be preferred when the controller is performing time-consuming tasks.

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

<RegisterID> can be 1.

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

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Descript-ion | On-target state | Is referencing | In motion | Servo mode on | - | - | - | Error flag |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Descript-ion | Digital in-put line 4 | Digital in-put line 3 | Digital in-put line 2 | Digital in-put line 1 | - | Positiv e limit switch | Ref-erence switch | Nega-tive limit switch |

Example: Send: `SRG? 1 1`

Receive: `1 1=0x9002`

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

### SST (Set Step Size)

Description: Sets the distance ("step size") for motions of the given axis that are triggered by a manual control unit.

Format: SST {<AxisID> <StepSize>}

Arguments: <AxisID> is one axis of the controller

<StepSize> is the distance, format: float

Response: None

Troubleshooting: Illegal value
Illegal axis identifier

Note: The distance set with SST is used when any relative motion of the axis of the C-867 is triggered by an axis of the HID. For details, see HIA (p. 185).

<StepSize> is specified in the physical unit of the axis position.

### SST? (Get Step Size)

Description: Gets the distance ("step size") for motions of the given axis that are triggered by a manual control unit.

Format: SST? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<StepSize> is the distance in physical units, see SST (p. 229).

### STE (Start Step And Response Measurement)

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

The data recorder configuration, i.e., the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 172).

The recorded data can be read with the DRR? command (p. 174).

Format: STE <AxisID> <Amplitude>

Arguments: <AxisID> is one axis of the controller

<Amplitude> is the size of the step. See below for details.

Response: None

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

The target position must be inside the soft limits. Use TMN? (p. 241) and TMX? (p. 241) to get the currently valid soft limits, and MOV? (p. 214) to get the current target.

Motion commands such as STE are not permitted when HID control is activated for the axis. Refer to "Controlling with an HID" (p. 114) for further information.

Notes: A "step" consists of motion with the specified amplitude which is performed relative to the current position.

**STP (Stop All Axes)**

Description: Stops all axes abruptly. See the notes below for further details.

Sets error code to 10.

This command is identical in function to #24 (p. 154).

Format: STP

Arguments: None

Response: None

Troubleshooting: Communication breakdown

Notes: SPA stops all motion caused by motion commands (e. g., MOV (p. 213), MVR (p. 215), GOH (p. 181), STE (p. 230), SMO (p. 224)), trajectory execution (TGS (p. 237)), the command for referencing (FRF (p. 180)), and macros (MAC (p. 207)). Also stops macro running.

After the axes are stopped, their target positions are set to their current positions.

HLT (p. 197) in contrast to STP stops motion with given system deceleration with regard to system inertia. Does not apply to trajectories.

Setting the ***Controller Disable Error 10*** parameter (0xE000301) prevents the error code 10 from being output when axis motion is stopped with STP.

### SVO (Set Servo Mode)

Description: Sets the servo mode for specified axes (open-loop or closed-loop operation).

Format: SVO {<AxisID> <ServoState>}

Arguments: <AxisID> is one axis of the controller

<ServoState> can have the following values:
0 = servo mode off (open-loop operation)
1 = servo mode on (closed-loop operation)

Response: None

Troubleshooting: Illegal axis identifier

Notes: When switching from open-loop to closed-loop operation, the target is set to the current position to avoid jumps of the mechanical system.

The current state of the servo mode determines the applicable motion commands:

- Servo mode ON: Use the MOV (p. 213), MVR (p. 215), and GOH (p. 181) commands for point-to-point motion, TGS (p. 237) for trajectory execution or use HID control (p. 114).
- Servo mode OFF: Use SMO (p. 224).

When the servo mode is switched off while the axis is moving, the axis stops.

If a motion error occurs or there is a continuously high control value, the servo mode is switched off. For further information, see "Protective Functions of the C-867" (p. 93).

### SVO? (Get Servo Mode)

Description: Queries the servo mode for the axes specified.

If arguments are not specified, queries the servo mode of all axes.

Format: SVO? [{<AxisID>}]

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

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

where

<ServoState> is the current servo mode for the axis:
0 = servo mode off (open-loop operation)
1 = servo mode on (closed-loop operation)

Troubleshooting: Illegal axis identifier

### TAC? (Tell Analog Channels)

Description: Gets the number of installed analog lines.

Format: TAC?

Arguments: None

Response: <uint> indicates the total number of analog lines (inputs and outputs).

Notes: Gets the number of analog input lines on the **I/O** socket (p. 316) of the C-867 (Input 1 to Input 4). Note that these lines can also be used for digital input. For further information, refer to "Commandable Elements" (p. 18).

### TAV? (Get Analog Input Voltage)

Description: Get voltage at analog input.

Format: TAV? [{<AnalogInputID>}]

Arguments: <AnalogInputID> is the identifier of the analog input channel; see below for details.

Response: {<AnalogInputID>"="<float> LF}

where

<float> is the current voltage at the analog input in volts

Notes: Using the TAV? command, you can directly read the Input 1 to Input 4 lines on the **I/O** socket (p. 316) of the C-867. The identifiers of the lines are 1 to 4. Refer to "Commandable Elements" (p. 18) for further information.

You can record the values of the analog input lines using the DRC record option 81 (p. 172).

### TCV? (Get Commanded Closed-Loop Velocity)

| | |
|---|---|
| Description: | Queries the current value of the velocity (value calculated by the profile generator). |
| Format: | TCV? [{<AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller. |
| Response: | {<AxisID>"="<float> LF}<br><br>where<br><br><float> is the velocity value in physical units per second. |

### TGA (Append Value To Trajectory)

| | |
|---|---|
| Description: | Command for motion paths: Loads trajectory points to the buffer of the specified trajectory.<br><br>Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer. The maximum number of points in the trajectory buffer is determined by the ***Maximum Buffer Size*** parameter (0x22000020). |
| Format: | TGA {<Trajectory> <Point>} |
| Arguments: | <Trajectory>: Identifier of the trajectory<br><Point>: Value in FLOAT format; indicates a trajectory point as the absolute position in physical units |
| Response: | none |
| Example: | Controller with 2 axes:<br>Trajectory 1 belongs to axis 1, trajectory 2 to axis 2<br>`TGA 2 3.4 1 5.6`<br>A trajectory point with the value 3.4 is added to the trajectory of axis 2; a trajectory point with the value 5.6 is added to the trajectory of axis 1 |
| Troubleshooting: | ▪ Invalid trajectory identifier<br>▪ Trajectory buffer full (> ***Maximum Buffer Size***) |
| Notes: | The timing for trajectories is set with the TGT (p. 238) command.<br><br>The TGS (p. 237) command starts the execution of a trajectory.<br><br>The TGF (p. 236) command properly completes the execution of a trajectory. |

If the execution of a trajectory is cancelled after an error or stopped with STP (p. 231), #24 (p. 154), or HLT (p. 197), the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 237), deletion with TGC (p. 235)).

The C-867 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

- Values of the trajectory points
- Timing for the trajectories
- Sufficiently fast refilling of the trajectory buffer

Following an unsuitable trajectory can cause the positioner to oscillate or stop motion abruptly. Oscillation or stopping abruptly can damage the positioner and/or the load fixed to it.

➢ Therefore, pay attention to the following when working with trajectories:
  - The path that is specified by the trajectory points must be continuously differentiable at least twice.
  - During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
  - When following the trajectory, an abrupt stop may not damage the load on the positioner.
  - To generate the trajectory points and continuously transfer them to the C-867 during the trajectory execution, it is recommended to use a suitable program.

For further information, see "Trajectories for Motion Paths" (p. 95).

### TGC (Clear All Values In Trajectory)

Description: Command for motion paths: Deletes the trajectory points in the buffer of the specified trajectory

If no argument is given, the points in the buffer of all trajectories are deleted.

| | |
|---|---|
| Format: | TGC [{<Trajectory>}] |
| Arguments: | <Trajectory>: Identifier of the trajectory |
| Response: | none |
| Example: | Controller with 2 axes:<br>Trajectory 1 belongs to axis 1, trajectory 2 to axis 2<br>`TGC 2 1`<br>Points of the trajectories of axis 2 and axis 1 are deleted |
| Troubleshooting: | Invalid trajectory identifier |
| Notes: | If the execution of a trajectory is cancelled after an error or stopped with STP (p. 231), #24 (p. 154), or HLT (p. 197), the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading a new trajectory, make sure that there are no invalid trajectory points in the buffer.<br><br>The number of points in the buffer can be queried with the TGL? (p. 237) command.<br><br>For further information, see "Trajectories for Motion Paths" (p. 95). |

**TGF (Finalize Trajectory)**

| | |
|---|---|
| Description: | Command for motion paths: Completes the execution of the specified trajectory<br><br>TGF must be sent after the last trajectory point has been loaded. If the trajectory execution is not properly completed with TGF, an error will occur when the number of points in the buffer falls below the required minimum (4).<br><br>For further information, see "Trajectories for Motion Paths" (p. 95).<br><br>A trajectory will only be executed as long as there are at least 4 points in the trajectory buffer. For trajectories to be executed to the end, this command must be sent after all trajectory points have been loaded. It signals to the firmware that no more points will be supplied for the specified trajectory. In this case, the remaining trajectory points will be processed without an error occurring when the minimum number of points is no longer present. |
| Format: | TGF [{<Trajectory>}] |

Arguments: <Trajectory>: Identifier of the trajectory

Response: none

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2

`TGF 2 1`

Trajectories of axis 2 and axis 1 are completed

Troubleshooting:
- Invalid trajectory identifier
- Trajectory is not currently being executed

### TGL? (Get Number Of Values In Trajectory)

Description: Command for motion paths: Queries the number of points in the buffer of the specified trajectory.
If no argument is given, the points for all trajectories are queried.

For further information, see "Trajectories for Motion Paths" (p. 95).

Format: TGL? [{<Trajectory>}]

Arguments: <Trajectory>: Identifier of the trajectory

Response: <Trajectory>=<int>LF
where
<int> is the current number of points in the buffer of the trajectory

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2

Send: `TGL? 2 1`

Receive:
```
2=12
1=18
```

Trajectory buffer of axis 2 contains 12 points,
trajectory buffer of axis 1 contains 18 points

### TGS (Start Trajectory)

Description: Command for motion paths: Starts the execution of the specified trajectory or trajectories

If no argument is given, the execution of all valid trajectories is started.

Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with TGA (p. 234). During the execution of a trajectory, the buffer must be refilled fast enough. The execution of a trajectory must be completed with TGF (p. 236).

Format: TGS [{<Trajectory>}]

Arguments: <Trajectory>: Identifier of the trajectory

Response: none

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2

```
TGS 2 1
```

Trajectories of axis 2 and axis 1 are started

Troubleshooting:

- Invalid trajectory identifier
- Trajectory buffer contains too few points (< 4)
- Servo mode switched off

Notes: The individual points of a trajectory are loaded to the buffer with the TGA (p. 234) command.

The timing for trajectories is set with the TGT (p. 238) command.

In addition to proper completion with TGF (p. 236), the execution of a trajectory can be stopped at any time with STP (p. 231), #24 (p. 154), or HLT (p. 197).

If an error occurs, the execution of all trajectories is cancelled.

After the execution has been stopped or cancelled, the unprocessed trajectory points remain in the buffer. Therefore, before executing a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 237), deletion with TGC (p. 235)).

For further information, see "Trajectories for Motion Paths" (p. 95).

### TGT (Set Trajectory Timing)

Description: Command for motion paths: Sets the timing for trajectories.

The timing specifies the time interval at which the individual points are output during the execution of the trajectories.

The specified value is valid for all trajectories.

Format: TGT <NoOfServoCycles>

Arguments: <NoOfServoCycles>: Time interval between the output of the individual points of a trajectory (unit: Number of servo cycles)

Response: None

Notes: The TGA (p. 234) command loads the points to the trajectory buffer.

The TGS (p. 237) command starts the execution of a trajectory.

The C-867 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

- Values of the trajectory points
- Timing for the trajectories
- Sufficiently fast refilling of the trajectory buffer

Following an unsuitable trajectory can cause the positioner to oscillate or stop motion abruptly. Oscillation or stopping abruptly can damage the positioner and/or the load fixed to it.

➢ Therefore, pay attention to the following when working with trajectories:
  - The path that is specified by the trajectory points must be continuously differentiable at least twice.
  - During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
  - When following the trajectory, an abrupt stop may not damage the load on the positioner.
  - To generate the trajectory points and continuously transfer them to the C-867 during the trajectory execution, it is recommended to use a suitable program.

For further information, see "Trajectories for Motion Paths" (p. 95).

### TGT? (Get Trajectory Timing)

Description: Command for motion paths: Queries the timing for trajectories.

The returned value is valid for all trajectories.

For further information, see "Trajectories for Motion Paths" (p. 95).

Format: TGT?

Arguments: None

Response: <NoOfServoCycles>: Time interval between the output of the individual points of a trajectory (unit: Number of servo cycles)

### TIM (Set Timer Value)

Description: Sets the timer to the given value. When the value is omitted, the timer is reset to zero.

The timer is incremented each servo cycle and can be used for measuring the time. Incrementing starts at zero each time the C-867 is switched on or rebooted.

Format: TIM [<Float>]

Arguments: <Float> is the value to which the timer is set, in milliseconds. The minimum possible value is zero.

Response: None

Notes: The servo cycle time of the C-867 is 50 µs (***Servo Update Time*** parameter, ID 0xE000200).

The current value of the timer can be recorded with the data recorder (see DRC (p. 172), record option 44).

### TIM? (Get Timer Value)

Description: Gets the current value of the timer.

Format: TIM?

Arguments: None

Response: <Float> is the current value of the timer, in milliseconds. For details, see TIM (p. 240).

### TIO? (Tell Digital I/O Lines)

Description: Tells number of installed digital I/O lines

Format: TIO?

Arguments: none

Response: I=<uint1>
O=<uint2>

where

<uint1> is the number of digital input lines.
<uint2> is the number of digital output lines.

Notes: The digital output lines reported by TIO? are Output 1 to Output 4. The states of the Output 1 to Output 4 lines can be set using the DIO command (p. 169). Furthermore, you can program the Output 1 to Output 4 lines using the CTO command (p. 161) (trigger configuration) and the TRO command (p. 242) (trigger enabling/disabling).

The digital input lines reported by TIO? are Input 1 to Input 4. They can be read with DIO? (p. 170), #4 (p. 151) and SRG? (p. 228). In addition, you can use the Input 1 and 2 or Input 3 and 4 lines for HID control of the C-867's axis. See HIA (p. 185) and "Connecting an HID" (p. 60) for details.

All line are on the C-867's **I/O** socket (p. 316).

### TMN? (Get Minimum Commandable Position)

Description: Get the minimum commandable position in physical units.

Format: TMN? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response {<AxisID>"="<float> LF}

where

<float> is the minimum commandable position in physical units

Note: The minimum commandable position is defined by the parameter 0x30. When redefining the zero position with the DFH (p. 166) command, the minimum commandable position is automatically adapted to the new zero position.

### TMX? (Get Maximum Commandable Position)

| | |
|---|---|
| Description: | Get the maximum commandable position in physical units. |
| Format: | TMX? [{ <AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller |
| Response | {<AxisID>"="<float> LF}<br><br>where<br><br><float> is the maximum commandable position in physical units |
| Note: | The maximum commandable position is defined by the parameter 0x15. When redefining the zero position with the DFH (p. 166) command, the maximum commandable position is automatically adapted to the new zero position. |

### TNR? (Get Number of Record Tables)

| | |
|---|---|
| Description: | Queries the number of data recorder tables currently available on the controller. |
| Format: | TNR? |
| Arguments: | none |
| Response | <uint> is the number of data recorder tables which are currently available |
| Notes: | The C-867 has four data recorder tables with 8192 data points per table.<br><br>For further information, see "Data Recorder" (p. 98). |

### TRO (Set Trigger Output State)

| | |
|---|---|
| Description: | Activates or deactivates the trigger output conditions set with CTO (p. 161) for the specified digital output line. |
| Format: | TRO {<TrigOutID> <TrigMode>} |
| Arguments: | <TrigOutID> is a digital output line of the controller; see below for further details.<br><br><TrigMode> can have the following values:<br>0 = Trigger output deactivated |

| | |
|---|---|
| | 1 = Trigger output activated |
| Response: | None |
| Troubleshooting: | Illegal identifier of the digital output line |
| Notes: | <TrigOutID> corresponds to the digital output lines Output 1 to Output 4, IDs = 1 to 4; for further information, see "I/O" (p. 316).<br><br>Do not use DIO (p. 169) on digital output lines where the trigger output is activated by TRO. |

**TRO? (Get Trigger Output State)**

| | |
|---|---|
| Description: | Queries the activation status of the trigger output configuration made with CTO (p. 161) for the specified digital output line.<br><br>If no arguments are specified, queries state of all digital output lines. |
| Format: | TRO? [{<TrigOutID>}] |
| Arguments: | <TrigOutID> is one digital output line of the controller, see TRO (p. 242) for more details. |
| Response: | {<TrigOutID>"="<TrigMode> LF}<br><br>where<br><br><TrigMode> is the current state of the digital output line:<br>0 = Trigger output deactivated<br>1 = Trigger output activated |
| Troubleshooting: | Illegal identifier of the digital output line |

**TRS? (Indicate Reference Switch)**

| | |
|---|---|
| Description: | Indicates whether axes have a reference switch with direction sensing. |
| Format: | TRS? [{<AxisID>}] |
| Arguments: | <AxisID> is one axis of the controller |

| | |
|---|---|
| Response: | {<AxisID>"="<uint> LF}<br><br>where<br><br><uint> indicates whether the axis has a direction-sensing reference switch (=1) or not (=0). |
| Troubleshooting: | Illegal axis identifier |
| Notes: | The C-867 firmware detects the presence or absence of a reference switch via a parameter (ID 0x14). The C-867 activates or deactivates reference moves to the reference switch (FRF command (p. 180)) according to the value of this parameter. Adapt the parameter value to your hardware using SPA (p. 226) or SEP (p. 222). For further information, see "Reference Switch Detection" (p. 45).<br><br>You can use a digital input line instead of the reference switch as source of the reference point signal for the FRF command. For further information, see "Digital Input Signals" (p. 109). |

**TVI? (Tell Valid Character Set For Axis Identifiers)**

| | |
|---|---|
| Description: | Returns a string with characters which can be used for axis identifiers.<br><br>Use SAI (p. 221) to change the axis identifiers and SAI? (p. 222) to ask for the current valid axis identifiers. |
| Format: | TVI? |
| Arguments: | None |
| Response: | <string> is a list of characters |
| Notes: | With the C-867, the string consists of<br>1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ-_ |

**VAR (Set Variable Value)**

| | |
|---|---|
| Description: | Sets a variable to a certain value.<br><br>Local variables can be set using VAR in macros only. See “Variables” (p. 146) for more details on local and global variables. |

| | |
|---|---|
| | The variable is present in RAM only. |
| Format: | VAR <Variable> <String> |
| Arguments: | <Variable> is the name of the variable whose value is to be set.<br><br><String> is the value to which the variable is to be set. If not specified, the variable is deleted.<br><br>The value can be specified directly or via the value of a variable.<br><br>Refer to "Variables" (p. 146) for more details on conventions regarding variable names and values. |
| Response: | None |
| Example: | It is possible to set the value of one variable (e.g., TARGET) to that of another variable (e.g., SOURCE):<br><br>`VAR TARGET ${SOURCE}`<br><br>Use braces if the name of the variable is longer than one character:<br><br>`VAR A ONE`<br>`VAR VARB TWO`<br>`VAR $A 1`<br>`VAR ${VARB} 2`<br>`VAR $VARB 2` // this will result in an unwanted behavior<br>`VAR?`<br>`A=ONE`<br>`VARB=TWO`<br>`ONE=1`<br>`TWO=2 // ${VARB}`: is replaced by its value "TWO".<br>`ARB=2 // $VARB`: $V is replaced by its (empty) value.<br><br>See ADD (p. 156) for another example. |

### VAR? (Get Variable Values)

| | |
|---|---|
| Description: | Gets values of variables.<br><br>If VAR? is combined with CPY (p. 159), JRC (p. 205), MEX (p. 212) or WAC (p. 247), the response to VAR? has to be a single value and not more.<br><br>Refer to "Variables" (p. 146) for more details on local and |

| | |
|---|---|
| | global variables. |
| Format: | VAR? [{<Variable>}] |
| Arguments: | <Variable> is the name of the variable to be queried. Refer to "Variables" (p. 146) for more details on name conventions.<br><br>All global variables present in RAM are listed if <Variable> is not specified. |
| Response: | {<Variable>"="<String>LF}<br><br>where<br><br><String> gives the value to which the variable is set. |
| Notes: | Local variables can be queried using VAR? only when a macro with local variables is running. See "Variables" (p. 146) for details regarding local and global variables. |
| Example: | See ADD (p. 156) for an example. |

**VEL (Set Closed-Loop Velocity)**

| | |
|---|---|
| Description: | Set velocity of specified axes. |
| Format: | VEL {<AxisID> <Velocity>} |
| Arguments: | <AxisID> is one axis of the controller.<br><br><Velocity> is the velocity value in physical units/s. |
| Response: | None |
| Troubleshooting: | Illegal axis identifiers |
| Notes: | The VEL setting only takes effect when the specified axis is in closed-loop operation (servo mode ON).<br><br>The lowest possible value for <Velocity> is 0.<br><br>The velocity can be changed with VEL while the axis is moving.<br><br>VEL changes the value of the ***Closed-Loop Velocity (Phys. Unit/s)*** parameter (ID 0x49) in the volatile memory of C-867. The parameter value can be stored as default with WPA (p. 248), for details see "Adapting Settings" (p. 273).<br><br>The maximum value that can be set with the VEL command is specified by the ***Maximum Closed-Loop Velocity (Phys. Unit/s)*** parameter, ID 0xA. |

### VEL? (Get Closed-Loop Velocity)

Description: Queries the commanded velocity.

If no arguments are specified, queries the value of all axes.

Format: VEL? [{<AxisID>}]

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

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

where

<float> is the currently valid velocity value commanded in physical units per second.

Notes: VEL? gets the value of the velocity for closed-loop operation commanded with VEL (value of the ***Closed Loop Velocity (Phys. Unit/s)*** parameter (ID 0x49) in the volatile memory).

### VER? (Get Versions Of Firmware And Drivers)

Description: Gets the versions of the firmware of the C-867 as well as of further components like, for example, drivers and libraries.

Format: VER?

Arguments: None

Response {<string1>":" <string2> [<string3>]LF}

where

<string1> is the name of the component;
<string2> is the version information of the component <string1>;
<string3> is an optional note.

### WAC (Wait For Condition)

Description: Waits until a specified condition of the following type occurs: a specified value is compared with a queried value according a specified rule.

Can only be used in macros.

See also the MEX command (p. 212).

Format: WAC <CMD?> <OP> <value>

Arguments <CMD?> is one query command in its usual notation. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:
= <= < > >= !=
Important: There must be a blank space before and after the operator!

<value> is the value to be compared with the response to <CMD?>.

Response: None

Example: Send:

```
MAC BEG LPMOTION
MVR 1 1
 WAC ONT? 1 = 1
MVR 1 -1
 WAC ONT? 1 = 1
MAC START LPMOTION
MAC END
MAC START LPMOTION
```

Note: Macro LPMOTION is first recorded and then started. WAC ONT? 1 = 1 waits until the response to ONT? 1 is 1=1. To form an infinite loop, the macro calls itself.

### WPA (Save Parameters To Non-Volatile Memory)

Description: Writes the currently valid value of a parameter of a specified element from volatile memory (RAM) to nonvolatile memory. The values saved this way become the default values.

**Note: If the current parameter values are incorrect, this can cause a fault in the system. Make sure that the parameter settings are correct before you execute the WPA command.**

RAM settings not saved with WPA will be lost when the controller is switched off or rebooted or when RPA (p. 219) is used to restore the parameters.

You can obtain a list of all available parameters with HPA?

| | |
|---|---|
| | (p. 198).<br><br>Use SPA? (p. 226) to check the current parameter settings in volatile memory.<br><br>See SPA (p. 226) for an example. |
| Format: | WPA <Pswd> [{<ItemID> <PamID>}] |
| Arguments: | <Pswd> is the password for writing to the nonvolatile memory. See below for details.<br><br><ItemID> is the element for which a parameter is to be saved from the volatile to the nonvolatile memory. See below for details.<br><br><PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details. |
| Response: | None |
| Troubleshooting: | Illegal element identifier, wrong parameter ID, invalid password<br>**Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.** |
| Notes: | Parameters can be changed in the volatile memory with SPA (p. 226), ACC (p. 155), DEC (p. 165), and VEL (p. 246). Some of the parameters are loaded into the volatile memory of the C-867 from the connected positioner's ID chip (p. 15) when the C-867 is switched on or rebooted. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 14) into the volatile memory of the C-867.<br><br>WPA can also save parameter-independent settings set with the following commands:<br>HDT (p. 183) assigns an HID axis to a lookup table<br>HIA (p. 185), configures HID control<br>HIT (p. 193) fills lookup tables with values<br><br>Saving settings with WPA has no effect on the default settings that are loaded with DPA (p. 171).<br><br>The used password determines what is saved with WPA: |

| | | |
|---|---|---|
| Valid passwords for writing to the nonvolatile memory: | 100 | Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT |
| | 101 | Saves the currently valid values of all parameters |
| | HID | Saves the currently valid settings for HDT, HIA and HIT |

| | |
|---|---|
| Available element IDs and parameter IDs: | It is not possible to specifically select individual items and parameters for saving with the C-867; i. e., <ItemID> and <PamID> are ignored. |

## 8.7 Error Codes

The error codes listed here are those of the PI General Command Set. As such, some may be not relevant to your controller and will simply never occur.

### Controller Errors

| | | |
|---|---|---|
| 0 | PI_CNTR_NO_ERROR | No error |
| 1 | PI_CNTR_PARAM_SYNTAX | Parameter syntax error |
| 2 | PI_CNTR_UNKNOWN_COMMAND | Unknown command |
| 3 | PI_CNTR_COMMAND_TOO_LONG | Command length out of limits or command buffer overrun |
| 4 | PI_CNTR_SCAN_ERROR | Error while scanning |
| 5 | PI_CNTR_MOVE_WITHOUT_REF_OR_NO_SERVO | Unallowable move attempted on unreferenced axis, or move attempted with servo off |
| 6 | PI_CNTR_INVALID_SGA_PARAM | Parameter for SGA not valid |
| 7 | PI_CNTR_POS_OUT_OF_LIMITS | Position out of limits |
| 8 | PI_CNTR_VEL_OUT_OF_LIMITS | Velocity out of limits |
| 9 | PI_CNTR_SET_PIVOT_NOT_POSSIBLE | Attempt to set pivot point while U,V and W not all 0 |
| 10 | PI_CNTR_STOP | Controller was stopped by command |
| 11 | PI_CNTR_SST_OR_SCAN_RANGE | Parameter for SST or for one of the embedded scan algorithms out of range |
| 12 | PI_CNTR_INVALID_SCAN_AXES | Invalid axis combination for fast scan |
| 13 | PI_CNTR_INVALID_NAV_PARAM | Parameter for NAV out of range |
| 14 | PI_CNTR_INVALID_ANALOG_INPUT | Invalid analog channel |
| 15 | PI_CNTR_INVALID_AXIS_IDENTIFIER | Invalid axis identifier |
| 16 | PI_CNTR_INVALID_STAGE_NAME | Unknown stage name |
| 17 | PI_CNTR_PARAM_OUT_OF_RANGE | Parameter out of range |
| 18 | PI_CNTR_INVALID_MACRO_NAME | Invalid macro name |
| 19 | PI_CNTR_MACRO_RECORD | Error while recording macro |

| | | |
|---|---|---|
| 20 | PI_CNTR_MACRO_NOT_FOUND | Macro not found |
| 21 | PI_CNTR_AXIS_HAS_NO_BRAKE | Axis has no brake |
| 22 | PI_CNTR_DOUBLE_AXIS | Axis identifier specified more than once |
| 23 | PI_CNTR_ILLEGAL_AXIS | Illegal axis |
| 24 | PI_CNTR_PARAM_NR | Incorrect number of parameters |
| 25 | PI_CNTR_INVALID_REAL_NR | Invalid floating point number |
| 26 | PI_CNTR_MISSING_PARAM | Parameter missing |
| 27 | PI_CNTR_SOFT_LIMIT_OUT_OF_RANGE | Soft limit out of range |
| 28 | PI_CNTR_NO_MANUAL_PAD | No manual pad found |
| 29 | PI_CNTR_NO_JUMP | No more step-response values |
| 30 | PI_CNTR_INVALID_JUMP | No step-response values recorded |
| 31 | PI_CNTR_AXIS_HAS_NO_REFERENCE | Axis has no reference sensor |
| 32 | PI_CNTR_STAGE_HAS_NO_LIM_SWITCH | Axis has no limit switch |
| 33 | PI_CNTR_NO_RELAY_CARD | No relay card installed |
| 34 | PI_CNTR_CMD_NOT_ALLOWED_FOR_STAGE | Command not allowed for selected stage(s) |
| 35 | PI_CNTR_NO_DIGITAL_INPUT | No digital input installed |
| 36 | PI_CNTR_NO_DIGITAL_OUTPUT | No digital output configured |
| 37 | PI_CNTR_NO_MCM | No more MCM responses |
| 38 | PI_CNTR_INVALID_MCM | No MCM values recorded |
| 39 | PI_CNTR_INVALID_CNTR_NUMBER | Controller number invalid |
| 40 | PI_CNTR_NO_JOYSTICK_CONNECTED | No joystick configured |
| 41 | PI_CNTR_INVALID_EGE_AXIS | Invalid axis for electronic gearing, axis 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_FAILE | Move to limit switch failed |

| | | |
|---|---|---|
| | D | |
| 50 | PI_CNTR_REF_WITH_REF_DISABLED | Attempt to reference axis with referencing disabled |
| 51 | PI_CNTR_AXIS_UNDER_JOYSTICK_CONTRO L | Selected axis is controlled by joystick |
| 52 | PI_CNTR_COMMUNICATION_ERROR | Controller detected communication error |
| 53 | PI_CNTR_DYNAMIC_MOVE_IN_PROCESS | MOV! motion still in progress |
| 54 | PI_CNTR_UNKNOWN_PARAMETER | Unknown parameter |
| 55 | PI_CNTR_NO_REP_RECORDED | No commands were recorded with REP |
| 56 | PI_CNTR_INVALID_PASSWORD | Password invalid |
| 57 | PI_CNTR_INVALID_RECORDER_CHAN | Data Record Table does not exist |
| 58 | PI_CNTR_INVALID_RECORDER_SRC_OPT | Source does not exist; number too low or too high |
| 59 | PI_CNTR_INVALID_RECORDER_SRC_CHAN | Source Record Table number too low or too high |
| 60 | PI_CNTR_PARAM_PROTECTION | Protected Param: current Command Level (CCL) too low |
| 61 | PI_CNTR_AUTOZERO_RUNNING | Command execution not possible while Autozero is running |
| 62 | PI_CNTR_NO_LINEAR_AXIS | Autozero requires at least one linear axis |
| 63 | PI_CNTR_INIT_RUNNING | Initialization still in progress |
| 64 | PI_CNTR_READ_ONLY_PARAMETER | Parameter is read-only |
| 65 | PI_CNTR_PAM_NOT_FOUND | Parameter not found in 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). |
| 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. |
| 94 | PI_CNTR_OPEN_LOOP_JOYSTICK_ENABLED | Servo cannot be switched on when open-loop joystick control is activated. |
| 95 | PI_CNTR_INVALID_SERVO_STATE_FOR_PARAMETER | This parameter cannot be changed in current servo mode. |
| 96 | PI_CNTR_UNKNOWN_STAGE_NAME | Unknown stage name |
| 97 | PI_CNTR_INVALID_VALUE_LENGTH | Invalid length of value (too much characters) |
| 98 | PI_CNTR_AUTOZERO_FAILED | AutoZero procedure was not successful |
| 99 | PI_CNTR_SENSOR_VOLTAGE_OFF | Sensor voltage is off |
| 100 | PI_LABVIEW_ERROR | PI driver for use with NI LabVIEW reports error. See source control for details. |
| 200 | PI_CNTR_NO_AXIS | No stage connected to axis |
| 201 | PI_CNTR_NO_AXIS_PARAM_FILE | File with axis parameters not found |
| 202 | PI_CNTR_INVALID_AXIS_PARAM_FILE | Invalid axis parameter file |
| 203 | PI_CNTR_NO_AXIS_PARAM_BACKUP | Backup file with axis parameters not found |
| 204 | PI_CNTR_RESERVED_204 | PI internal error code 204 |
| 205 | PI_CNTR_SMO_WITH_SERVO_ON | SMO with servo on |
| 206 | PI_CNTR_UUDECODE_INCOMPLETE_HEADER | uudecode: incomplete header |
| 207 | PI_CNTR_UUDECODE_NOTHING_TO_DECODE | uudecode: nothing to decode |
| 208 | PI_CNTR_UUDECODE_ILLEGAL_FORMAT | uudecode: illegal UUE format |
| 209 | PI_CNTR_CRC32_ERROR | CRC32 error |
| 210 | PI_CNTR_ILLEGAL_FILENAME | Illegal file name (must be 8-0 format) |
| 211 | PI_CNTR_FILE_NOT_FOUND | File not found on controller |
| 212 | PI_CNTR_FILE_WRITE_ERROR | Error writing file on controller |
| 213 | PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE | VEL command not 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 |
| 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_LI | Wave parameter out of range |

| | | |
|---|---|---|
| | MIT | |
| 406 | PI_CNTR_WGO_BIT_NOT_SUPPORTED | WGO command bit not supported |
| 500 | PI_CNTR_EMERGENCY_STOP_BUTTON_ACTIVATED | The \"red knob\" is still set and disables system |
| 501 | PI_CNTR_EMERGENCY_STOP_BUTTON_WAS_ACTIVATED | The \"red knob\" was activated and still disables system - reanimation required |
| 502 | PI_CNTR_REDUNDANCY_LIMIT_EXCEEDED | Position consistency check failed |
| 503 | PI_CNTR_COLLISION_SWITCH_ACTIVATED | Hardware collision sensor(s) are activated |
| 504 | PI_CNTR_FOLLOWING_ERROR | Strut following error occurred, e.g. caused by overload or encoder failure |
| 505 | PI_CNTR_SENSOR_SIGNAL_INVALID | One sensor signal is not valid |
| 506 | PI_CNTR_SERVO_LOOP_UNSTABLE | Servo loop was unstable due to wrong parameter setting and switched off to avoid damage. |
| 507 | PI_CNTR_LOST_SPI_SLAVE_CONNECTION | Digital connection to external SPI slave device is lost |
| 508 | PI_CNTR_MOVE_ATTEMPT_NOT_PERMITTED | Move attempt not permitted due to customer or limit settings |
| 509 | PI_CNTR_TRIGGER_EMERGENCY_STOP | Emergency stop caused by trigger input |
| 530 | PI_CNTR_NODE_DOES_NOT_EXIST | A command refers to a node that does not exist |
| 531 | PI_CNTR_PARENT_NODE_DOES_NOT_EXIST | A command refers to a node that has no parent node |
| 532 | PI_CNTR_NODE_IN_USE | Attempt to delete a node that is in use |
| 533 | PI_CNTR_NODE_DEFINITION_IS_CYCLIC | Definition of a node is cyclic |
| 536 | PI_CNTR_HEXAPOD_IN_MOTION | Transformation cannot be defined as long as Hexapod is in motion |
| 537 | PI_CNTR_TRANSFORMATION_TYPE_NOT_SUPPORTED | Transformation node cannot be activated |
| 539 | PI_CNTR_NODE_PARENT_IDENTICAL_TO_CHILD | A node cannot be linked to itself |
| 540 | PI_CNTR_NODE_DEFINITION_INCONSISTENT | Node definition is erroneous or not complete (replace or delete it) |

| | | |
|---|---|---|
| 542 | PI_CNTR_NODES_NOT_IN_SAME_CHAIN | The nodes are not part of the same chain |
| 543 | PI_CNTR_NODE_MEMORY_FULL | Unused nodes must be deleted before new nodes can be stored |
| 544 | PI_CNTR_PIVOT_POINT_FEATURE_NOT_SUPPORTED | With some transformations pivot point usage is not supported |
| 545 | PI_CNTR_SOFTLIMITS_INVALID | Soft limits invalid due to changes in coordinate system |
| 546 | PI_CNTR_CS_WRITE_PROTECTED | Coordinate system is write protected |
| 547 | PI_CNTR_CS_CONTENT_FROM_CONFIG_FILE | Coordinate system cannot be changed because its content is loaded from a configuration file |
| 548 | PI_CNTR_CS_CANNOT_BE_LINKED | Coordinate system may not be linked |
| 549 | PI_CNTR_KSB_CS_ROTATION_ONLY | A KSB-type coordinate system can only be rotated by multiples of 90 degrees |
| 551 | PI_CNTR_CS_DATA_CANNOT_BE_QUERIED | This query is not supported for this coordinate system type |
| 552 | PI_CNTR_CS_COMBINATION_DOES_NOT_EXIST | This combination of work-and-tool coordinate systems does not exist |
| 553 | PI_CNTR_CS_COMBINATION_INVALID | The combination must consist of one work and one tool coordinate system |
| 554 | PI_CNTR_CS_TYPE_DOES_NOT_EXIST | This coordinate system type does not exist |
| 555 | PI_CNTR_UNKNOWN_ERROR | BasMac: unknown controller error |
| 556 | PI_CNTR_CS_TYPE_NOT_ACTIVATED | No coordinate system of this type is activated |
| 557 | PI_CNTR_CS_NAME_INVALID | Name of coordinate system is invalid |
| 558 | PI_CNTR_CS_GENERAL_FILE_MISSING | File with stored CS systems is missing or erroneous |
| 559 | PI_CNTR_CS_LEVELING_FILE_MISSING | File with leveling CS is missing or erroneous |
| 601 | PI_CNTR_NOT_ENOUGH_MEMORY | not enough memory |
| 602 | PI_CNTR_HW_VOLTAGE_ERROR | hardware voltage error |
| 603 | PI_CNTR_HW_TEMPERATURE_ERROR | hardware temperature out of range |

| | | |
|---|---|---|
| 604 | PI_CNTR_POSITION_ERROR_TOO_HIGH | Position error of any axis in the system is too high |
| 606 | PI_CNTR_INPUT_OUT_OF_RANGE | Maximum value of input signal has been exceeded |
| 607 | PI_CNTR_NO_INTEGER | Value is not integer |
| 608 | PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_RUNNING | Fast alignment process cannot be paused because it is not running |
| 609 | PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_PAUSED | Fast alignment process cannot be restarted/resumed because it is not paused |
| 650 | PI_CNTR_UNABLE_TO_SET_PARAM_WITH_SPA | Parameter could not be set with SPA - SEP needed? |
| 651 | PI_CNTR_PHASE_FINDING_ERROR | Phase finding error |
| 652 | PI_CNTR_SENSOR_SETUP_ERROR | Sensor setup error |
| 653 | PI_CNTR_SENSOR_COMM_ERROR | Sensor communication error |
| 654 | PI_CNTR_MOTOR_AMPLIFIER_ERROR | Motor amplifier error |
| 655 | PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_I2T | Overcurrent protection triggered by I2T-module |
| 656 | PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_AMP_MODULE | Overcurrent protection triggered by amplifier module |
| 657 | PI_CNTR_SAFETY_STOP_TRIGGERED | Safety stop triggered |
| 658 | PI_SENSOR_OFF | Sensor off? |
| 659 | PI_CNTR_PARAM_CONFLICT | Parameter could not be set. Conflict with another parameter. |
| 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 |
| 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 |
| 1013 | PI_CNTR_NO_MACRO_OR_EMPTY | No macro with given name on controller, or macro is empty |
| 1015 | PI_CNTR_INVALID_ARGUMENT | One or more arguments given to function is invalid (empty string, index out of range, ...) |
| 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 |
| 1028 | PI_CNTR_UNKNOWN_CHANNEL_IDENTIFIER | Unknown channel identifier |
| 1063 | PI_CNTR_EXT_PROFILE_UNALLOWED_CMD | User Profile Mode: Command is not allowed, check for required preparatory commands |
| 1064 | PI_CNTR_EXT_PROFILE_EXPECTING_MOTION_ERROR | User Profile Mode: First target position in User Profile is too far from current position |
| 1065 | PI_CNTR_PROFILE_ACTIVE | Controller is (already) in User Profile Mode |
| 1066 | PI_CNTR_PROFILE_INDEX_OUT_OF_RANGE | User Profile Mode: Block or Data Set index out of allowed range |
| 1071 | PI_CNTR_PROFILE_OUT_OF_MEMORY | User Profile Mode: Out of memory |
| 1072 | PI_CNTR_PROFILE_WRONG_CLUSTER | User Profile Mode: Cluster is not assigned to this axis |
| 1073 | PI_CNTR_PROFILE_UNKNOWN_CLUSTER_IDENTIFIER | Unknown cluster identifier |
| 1090 | PI_CNTR_TOO_MANY_TCP_CONNECTIONS_OPEN | There are too many open tcpip connections |
| 2000 | PI_CNTR_ALREADY_HAS_SERIAL_NUMBER | Controller already has a serial number |
| 2100 | PI_CNTR_FEATURE_LICENSE_INVALID | Entered license is invalid |
| 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 |
| 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 |
| -22 | COM_GPIB_EOIP | IEEE488: I/O operation started before previous operation completed |
| -23 | COM_GPIB_ECAP | IEEE488: No capability for intended operation |
| -24 | COM_GPIB_EFSO | IEEE488: File system operation error |
| -25 | COM_GPIB_EBUS | IEEE488: Command error during device call |
| -26 | COM_GPIB_ESTB | IEEE488: Serial poll-status byte lost |
| -27 | COM_GPIB_ESRQ | IEEE488: SRQ remains asserted |
| -28 | COM_GPIB_ETAB | IEEE488: Return buffer full |
| -29 | COM_GPIB_ELCK | IEEE488: Address or board locked |
| -30 | COM_RS_INVALID_DATA_BITS | RS-232: 5 data bits with 2 stop bits is an invalid combination, as is 6, 7, or 8 data bits with 1.5 stop bits |
| -31 | COM_ERROR_RS_SETTINGS | RS-232: Error configuring the COM port |
| -32 | COM_INTERNAL_RESOURCES_ERROR | Error dealing with internal system resources (events, threads, ...) |
| -33 | COM_DLL_FUNC_ERROR | A DLL or one of the required functions could not be loaded |
| -34 | COM_FTDIUSB_INVALID_HANDLE | FTDIUSB: invalid handle |
| -35 | COM_FTDIUSB_DEVICE_NOT_FOUND | FTDIUSB: device not found |
| -36 | COM_FTDIUSB_DEVICE_NOT_OPENED | FTDIUSB: device not opened |
| -37 | COM_FTDIUSB_IO_ERROR | FTDIUSB: IO error |
| -38 | COM_FTDIUSB_INSUFFICIENT_RESOURCES | FTDIUSB: insufficient resources |
| -39 | COM_FTDIUSB_INVALID_PARAMETER | FTDIUSB: invalid parameter |

| | | |
|---|---|---|
| -40 | COM_FTDIUSB_INVALID_BAUD_RATE | FTDIUSB: invalid baud rate |
| -41 | COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_ERASE | FTDIUSB: device not opened for erase |
| -42 | COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_WRITE | FTDIUSB: device not opened for write |
| -43 | COM_FTDIUSB_FAILED_TO_WRITE_DEVICE | FTDIUSB: failed to write device |
| -44 | COM_FTDIUSB_EEPROM_READ_FAILED | FTDIUSB: EEPROM read failed |
| -45 | COM_FTDIUSB_EEPROM_WRITE_FAILED | FTDIUSB: EEPROM write failed |
| -46 | COM_FTDIUSB_EEPROM_ERASE_FAILED | FTDIUSB: EEPROM erase failed |
| -47 | COM_FTDIUSB_EEPROM_NOT_PRESENT | FTDIUSB: EEPROM not present |
| -48 | COM_FTDIUSB_EEPROM_NOT_PROGRAMMED | FTDIUSB: EEPROM not programmed |
| -49 | COM_FTDIUSB_INVALID_ARGS | FTDIUSB: invalid arguments |
| -50 | COM_FTDIUSB_NOT_SUPPORTED | FTDIUSB: not supported |
| -51 | COM_FTDIUSB_OTHER_ERROR | FTDIUSB: other error |
| -52 | COM_PORT_ALREADY_OPEN | Error while opening the COM port: was already open |
| -53 | COM_PORT_CHECKSUM_ERROR | Checksum error in received data from COM port |
| -54 | COM_SOCKET_NOT_READY | Socket not ready, you should call the function again |
| -55 | COM_SOCKET_PORT_IN_USE | Port is used by another socket |
| -56 | COM_SOCKET_NOT_CONNECTED | Socket not connected (or not valid) |
| -57 | COM_SOCKET_TERMINATED | Connection terminated (by peer) |
| -58 | COM_SOCKET_NO_RESPONSE | Can't connect to peer |
| -59 | COM_SOCKET_INTERRUPTED | Operation was interrupted by a nonblocked signal |
| -60 | COM_PCI_INVALID_ID | No device with this ID is present |
| -61 | COM_PCI_ACCESS_DENIED | Driver could not be opened (on Vista: run as administrator!) |
| -62 | COM_SOCKET_HOST_NOT_FOUND | Host not found |
| -63 | COM_DEVICE_CONNECTED | Device already connected |
| -64 | COM_INVALID_COM_PORT | Invalid COM port |
| -65 | COM_USB_DEVICE_NOT_FOUND | USB device not found |
| -66 | COM_NO_USB_DRIVER | No USB driver installed |

| | | |
|---|---|---|
| -67 | COM_USB_NOT_SUPPORTED | USB is not supported |

**DLL Errors**

| | | |
|---|---|---|
| -1001 | PI_UNKNOWN_AXIS_IDENTIFIER | Unknown axis identifier |
| -1002 | PI_NR_NAV_OUT_OF_RANGE | Number for NAV out of range--must be in [1,10000] |
| -1003 | PI_INVALID_SGA | Invalid value for SGA--must be one of 1, 10, 100, 1000 |
| -1004 | PI_UNEXPECTED_RESPONSE | Controller sent unexpected response |
| -1005 | PI_NO_MANUAL_PAD | No manual control pad installed, calls to SMA and related commands are not allowed |
| -1006 | PI_INVALID_MANUAL_PAD_KNOB | Invalid number for manual control pad knob |
| -1007 | PI_INVALID_MANUAL_PAD_AXIS | Axis not currently controlled by a manual control pad |
| -1008 | PI_CONTROLLER_BUSY | Controller is busy with some lengthy operation (e.g., reference move, fast scan algorithm) |
| -1009 | PI_THREAD_ERROR | Internal error--could not start thread |
| -1010 | PI_IN_MACRO_MODE | Controller is (already) in macro mode--command not valid in macro mode |
| -1011 | PI_NOT_IN_MACRO_MODE | Controller not in macro mode--command not valid unless macro mode active |
| -1012 | PI_MACRO_FILE_ERROR | Could not open file to write or read macro |
| -1013 | PI_NO_MACRO_OR_EMPTY | No macro with given name on controller, or macro is empty |
| -1014 | PI_MACRO_EDITOR_ERROR | Internal error in macro editor |
| -1015 | PI_INVALID_ARGUMENT | One or more arguments given to function is invalid (empty string, index out of range, ...) |
| -1016 | PI_AXIS_ALREADY_EXISTS | Axis identifier is already in use by a connected stage |
| -1017 | PI_INVALID_AXIS_IDENTIFIER | Invalid axis identifier |
| -1018 | PI_COM_ARRAY_ERROR | Could not access array data in COM server |
| -1019 | PI_COM_ARRAY_RANGE_ERROR | Range of array does not fit |

| | | |
|---|---|---|
| | | the number of parameters |
| -1020 | PI_INVALID_SPA_CMD_ID | Invalid parameter ID given to SPA or SPA? |
| -1021 | PI_NR_AVG_OUT_OF_RANGE | Number for AVG out of range--must be >0 |
| -1022 | PI_WAV_SAMPLES_OUT_OF_RANGE | Incorrect number of samples given to WAV |
| -1023 | PI_WAV_FAILED | Generation of wave failed |
| -1024 | PI_MOTION_ERROR | Motion error: position error too large, servo is switched off automatically |
| -1025 | PI_RUNNING_MACRO | Controller is (already) running a macro |
| -1026 | PI_PZT_CONFIG_FAILED | Configuration of PZT stage or amplifier failed |
| -1027 | PI_PZT_CONFIG_INVALID_PARAMS | Current settings are not valid for desired configuration |
| -1028 | PI_UNKNOWN_CHANNEL_IDENTIFIER | Unknown channel identifier |
| -1029 | PI_WAVE_PARAM_FILE_ERROR | Error while reading/writing wave generator parameter file |
| -1030 | PI_UNKNOWN_WAVE_SET | Could not find description of wave form. Maybe WG.INI is missing? |
| -1031 | PI_WAVE_EDITOR_FUNC_NOT_LOADED | The WGWaveEditor DLL function was not found at startup |
| -1032 | PI_USER_CANCELLED | The user cancelled a dialog |
| -1033 | PI_C844_ERROR | Error from C-844 Controller |
| -1034 | PI_DLL_NOT_LOADED | DLL necessary to call function not loaded, or function not found in DLL |
| -1035 | PI_PARAMETER_FILE_PROTECTED | The open parameter file is protected and cannot be edited |
| -1036 | PI_NO_PARAMETER_FILE_OPENED | There is no parameter file open |
| -1037 | PI_STAGE_DOES_NOT_EXIST | Selected stage does not exist |
| -1038 | PI_PARAMETER_FILE_ALREADY_OPENED | There is already a parameter file open. Close it before opening a new file |
| -1039 | PI_PARAMETER_FILE_OPEN_ERROR | Could not open parameter file |
| -1040 | PI_INVALID_CONTROLLER_VERSION | The version of the connected controller is invalid |
| -1041 | PI_PARAM_SET_ERROR | Parameter could not be set |

| | | |
|---|---|---|
| | | with SPA--parameter not defined for this controller! |
| -1042 | PI_NUMBER_OF_POSSIBLE_WAVES_EXCEEDED | The maximum number of wave definitions has been exceeded |
| -1043 | PI_NUMBER_OF_POSSIBLE_GENERATORS_EXCEEDED | The maximum number of wave generators has been exceeded |
| -1044 | PI_NO_WAVE_FOR_AXIS_DEFINED | No wave defined for specified axis |
| -1045 | PI_CANT_STOP_OR_START_WAV | Wave output to axis already stopped/started |
| -1046 | PI_REFERENCE_ERROR | Not all axes could be referenced |
| -1047 | PI_REQUIRED_WAVE_NOT_FOUND | Could not find parameter set required by frequency relation |
| -1048 | PI_INVALID_SPP_CMD_ID | Command ID given to SPP or SPP? is not valid |
| -1049 | PI_STAGE_NAME_ISNT_UNIQUE | A stage name given to CST is not unique |
| -1050 | PI_FILE_TRANSFER_BEGIN_MISSING | A uuencoded file transferred did not start with "begin" followed by the proper filename |
| -1051 | PI_FILE_TRANSFER_ERROR_TEMP_FILE | Could not create/read file on host PC |
| -1052 | PI_FILE_TRANSFER_CRC_ERROR | Checksum error when transferring a file to/from the controller |
| -1053 | PI_COULDNT_FIND_PISTAGES_DAT | The PiStages.dat database could not be found. This file is required to connect a stage with the CST command |
| -1054 | PI_NO_WAVE_RUNNING | No wave being output to specified axis |
| -1055 | PI_INVALID_PASSWORD | Invalid password |
| -1056 | PI_OPM_COM_ERROR | Error during communication with OPM (Optical Power Meter), maybe no OPM connected |
| -1057 | PI_WAVE_EDITOR_WRONG_PARAMNUM | WaveEditor: Error during wave creation, incorrect number of parameters |
| -1058 | PI_WAVE_EDITOR_FREQUENCY_OUT_OF_RANGE | WaveEditor: Frequency out of range |
| -1059 | PI_WAVE_EDITOR_WRONG_IP_VALUE | WaveEditor: Error during |

| | | |
|---|---|---|
| | | wave creation, incorrect index for integer parameter |
| -1060 | PI_WAVE_EDITOR_WRONG_DP_VALUE | WaveEditor: Error during wave creation, incorrect index for floating point parameter |
| -1061 | PI_WAVE_EDITOR_WRONG_ITEM_VALUE | WaveEditor: Error during wave creation, could not calculate value |
| -1062 | PI_WAVE_EDITOR_MISSING_GRAPH_COMPONENT | WaveEditor: Graph display component not installed |
| -1063 | PI_EXT_PROFILE_UNALLOWED_CMD | User Profile Mode: Command is not allowed, check for required preparatory commands |
| -1064 | PI_EXT_PROFILE_EXPECTING_MOTION_ERROR | User Profile Mode: First target position in User Profile is too far from current position |
| -1065 | PI_EXT_PROFILE_ACTIVE | Controller is (already) in User Profile Mode |
| -1066 | PI_EXT_PROFILE_INDEX_OUT_OF_RANGE | User Profile Mode: Block or Data Set index out of allowed range |
| -1067 | PI_PROFILE_GENERATOR_NO_PROFILE | ProfileGenerator: No profile has been created yet |
| -1068 | PI_PROFILE_GENERATOR_OUT_OF_LIMITS | ProfileGenerator: Generated profile exceeds limits of one or both axes |
| -1069 | PI_PROFILE_GENERATOR_UNKNOWN_PARAMETER | ProfileGenerator: Unknown parameter ID in Set/Get Parameter command |
| -1070 | PI_PROFILE_GENERATOR_PAR_OUT_OF_RANGE | ProfileGenerator: Parameter out of allowed range |
| -1071 | PI_EXT_PROFILE_OUT_OF_MEMORY | User Profile Mode: Out of memory |
| -1072 | PI_EXT_PROFILE_WRONG_CLUSTER | User Profile Mode: Cluster is not assigned to this axis |
| -1073 | PI_UNKNOWN_CLUSTER_IDENTIFIER | Unknown cluster identifier |
| -1074 | PI_INVALID_DEVICE_DRIVER_VERSION | The installed device driver doesn't match the required version. Please see the documentation to determine the required device driver version. |
| -1075 | PI_INVALID_LIBRARY_VERSION | The library used doesn't match the required version. Please see the documentation |

| | | |
|---|---|---|
| | | to determine the required library version. |
| -1076 | PI_INTERFACE_LOCKED | The interface is currently locked by another function. Please try again later. |
| -1077 | PI_PARAM_DAT_FILE_INVALID_VERSION | Version of parameter DAT file does not match the required version. Current files are available at www.pi.ws. |
| -1078 | PI_CANNOT_WRITE_TO_PARAM_DAT_FILE | Cannot write to parameter DAT file to store user defined stage type. |
| -1079 | PI_CANNOT_CREATE_PARAM_DAT_FILE | Cannot create parameter DAT file to store user defined stage type. |
| -1080 | PI_PARAM_DAT_FILE_INVALID_REVISION | Parameter DAT file does not have correct revision. |
| -1081 | PI_USERSTAGES_DAT_FILE_INVALID_REVISION | User stages DAT file does not have correct revision. |
| -1082 | PI_SOFTWARE_TIMEOUT | Timeout Error. Some lengthy operation did not finish within expected time. |
| -1083 | PI_WRONG_DATA_TYPE | A function argument has an unexpected data type. |
| -1084 | PI_DIFFERENT_ARRAY_SIZES | Length of data arrays is different. |
| -1085 | PI_PARAM_NOT_FOUND_IN_PARAM_DAT_FILE | Parameter value not found in parameter DAT file. |
| -1086 | PI_MACRO_RECORDING_NOT_ALLOWED_IN_THIS_MODE | Macro recording is not allowed in this mode of operation. |
| -1087 | PI_USER_CANCELLED_COMMAND | Command cancelled by user input. |
| -1088 | PI_TOO_FEW_GCS_DATA | Controller sent too few GCS data sets |
| -1089 | PI_TOO_MANY_GCS_DATA | Controller sent too many GCS data sets |
| -1090 | PI_GCS_DATA_READ_ERROR | Communication error while reading GCS data |
| -1091 | PI_WRONG_NUMBER_OF_INPUT_ARGUMENTS | Wrong number of input arguments. |
| -1092 | PI_FAILED_TO_CHANGE_CCL_LEVEL | Change of command level has failed. |
| -1093 | PI_FAILED_TO_SWITCH_OFF_SERVO | Switching off the servo mode has failed. |

| | | |
|---|---|---|
| -1094 | PI_FAILED_TO_SET_SINGLE_PARAMETER_WHILE_PERFORMING_CST | A parameter could not be set while performing CST: CST was not performed (parameters remain unchanged). |
| -1095 | PI_ERROR_CONTROLLER_REBOOT | Connection could not be reestablished after reboot. |
| -1096 | PI_ERROR_AT_QHPA | Sending HPA? or receiving the response has failed. |
| -1097 | PI_QHPA_NONCOMPLIANT_WITH_GCS | HPA? response does not comply with GCS2 syntax. |
| -1098 | PI_FAILED_TO_READ_QSPA | Response to SPA? could not be received. |
| -1099 | PI_PAM_FILE_WRONG_VERSION | Version of PAM file cannot be handled (too old or too new) |
| -1100 | PI_PAM_FILE_INVALID_FORMAT | PAM file does not contain required data in PAM-file format |
| -1101 | PI_INCOMPLETE_INFORMATION | Information does not contain all required data |
| -1102 | PI_NO_VALUE_AVAILABLE | No value for parameter available |
| -1103 | PI_NO_PAM_FILE_OPEN | No PAM file is open |
| -1104 | PI_INVALID_VALUE | Invalid value |
| -1105 | PI_UNKNOWN_PARAMETER | Unknown parameter |
| -1106 | PI_RESPONSE_TO_QSEP_FAILED | Response to SEP? could not be received. |
| -1107 | PI_RESPONSE_TO_QSPA_FAILED | Response to SPA? could not be received. |
| -1108 | PI_ERROR_IN_CST_VALIDATION | Error while performing CST: One or more parameters were not set correctly. |
| -1109 | PI_ERROR_PAM_FILE_HAS_DUPLICATE_ENTRY_WITH_DIFFERENT_VALUES | PAM file has duplicate entry with different values. |
| -1110 | PI_ERROR_FILE_NO_SIGNATURE | File has no signature |
| -1111 | PI_ERROR_FILE_INVALID_SIGNATURE | File has invalid signature |
| -1112 | PI_ERROR_CANNOT_DETERMINE_ACTUAL_END_OF_TRAVEL_WHILE_PLATFORM_IS_MOVING | Cannot determine actual end of travel range while platform is moving. |
| -1113 | PI_ERROR_AT_QIDN | Sending IDN? or receiving the response has failed. |
| -1114 | PI_ERROR_AT_MAC_DEF | Sending MAC_DEF or receiving the response has failed. |

| | | |
|---|---|---|
| -1115 | PI_CONTROLLER_OR_CONTROLLER_VERSION_DOES_NOT_EXIST_IN_PISTAGES_DATABASE | Sending Controller or controller version does not exist in PIStages database. |
| -1116 | PI_NOT_ENOUGH_MEMORY | Not enough memory |
| -1117 | PI_ERROR_AXIS_RUNTIME_ERROR | Runtime error indicated for axis, check error log with \"LOG?\" to find more details. |
| -1118 | PI_ERROR_SYSTEM_RUNTIME_CRITICAL_ERROR | Critical error indicated for system, check error log with \"LOG?\" to find more details. |
| -1119 | PI_ERROR_CANNOT_START_EMULATOR | Cannot start emulation software. |
| -1120 | COM_DEVICE_NOT_SUPPORTED | Device is not supported |
| -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. |
| -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. |
| -10016 | PI_PARAMETER_DB_MISSING_FIRMWARE_FEATURE_ON_CONTROLLER | Parameters could not be set on controller because the corresponding firmware feature is missing |

# 9 Adapting Settings

## 9.1 Settings of the C-867

The properties of the C-867 and the connected positioner are stored in the C-867 as parameter values (e.g., settings for the servo algorithm (p. 30)).

The parameters can be divided into the following categories:

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

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

Every parameter is in the volatile as well as in the nonvolatile memory of the C-867. The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-867. The values in the volatile memory determine the current behavior of the system.

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

## 9.2 Changing Parameter Values in the C-867

***NOTICE***

**Unsuitable parameter settings!**

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-867 and take effect immediately. Unsuitable parameter settings can cause damage to the connected mechanics.

➢ Change parameter values only after careful consideration.

➢ Save the current parameter values to the PC (p. 275) before you make changes in the nonvolatile memory.

***INFORMATION***

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

➢ Overwrite the default values only when it is necessary.

➢ Save the current parameter values to the PC (p. 275) before you make changes in the nonvolatile memory.

➢ Contact our customer service department (p. 309), if the C-867 exhibits unexpected behavior.

***INFORMATION***

If the connected positioner has an ID chip (p. 15), the data is loaded from the ID chip into the volatile memory of the C-867 after switching on or rebooting the C-867.

The ID chip only contains some of the information that is required to operate the positioner with the C-867. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 14) into the volatile memory of the C-867.

Parameters that are loaded from the ID chip or from a positioner database are marked in color in the parameter overview (p. 282).

### 9.2.1 General Commands for Parameters

The following commands are available for changing parameters:

| Command | Function |
|---|---|
| `CCL` | Change to a higher command level, e.g., to obtain write permission for particular parameters. |
| `CCL?` | Get active command level. |
| `DPA` | Reset parameter values and parameter-independent settings to default settings. |
| `HPA?` | Responds with a help string that contains all available parameters with short descriptions. |
| `RPA` | Copy a parameter value from the nonvolatile to the volatile memory. |
| `SEP` | Change parameters in the nonvolatile memory. |
| `SEP?` | Get parameter values from the nonvolatile memory. |
| `SPA` | Change parameters in the volatile memory. |
| `SPA?` | Get parameter values from the volatile memory. |
| `WPA` | Copy a current parameter value from the volatile to the nonvolatile memory. |

You can find details in the command descriptions (p. 151).

### 9.2.2 Commands for Fast Access to Individual Parameters

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 (p. 248).

***INFORMATION***

The parameters listed below can also be changed with the general commands.

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

You can find details in the command descriptions (p. 151).

### 9.2.3 Saving Parameter Values in a Text File

***INFORMATION***

The C-867 is configured via parameters, e.g., to adapt the mechanics connected. Changing parameter values can cause undesirable results.

- Create a backup copy on the PC before changing the parameter settings of the C-867. 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 C-867 to specific mechanics.

***INFORMATION***

Parameter values saved in a text file on the PC can be loaded back to the C-867 in PIMikroMove or PITerminal. The ***Send file...*** button is available for this purpose in the send command window. Before loading into the C-867, the individual lines of the text files must be converted into command lines that contain the corresponding SPA or SEP commands.

**Requirements**

- ✓ You have established communication with PIMikroMove or PITerminal between the C-867 and the PC (p. 70).

**Saving parameter values in a text file**

1. If you use PIMikroMove, open the window for sending commands:
   - Select the ***Tools > Command entry*** menu item in the main window or press the F4 key on the keyboard.

   In PITerminal the main window from which commands can be sent is opened automatically after establishing communication.
2. Get the parameter values from which you want to create a backup copy.
   - If you want to save the parameter values from the volatile memory of the C-867: Send the `SPA?` command.
   - If you want to save the parameter values from the nonvolatile memory of the C-867: Send the `SEP?` command.
3. Click on the ***Save...*** button.

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

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

### 9.2.4 Changing Parameter Values: General Procedure

For working with parameters, you can use the general commands (p. 274) and the commands for quick access (p. 274).

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

***NOTICE***

**Unsuitable parameter settings!**

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-867 and take effect immediately. Unsuitable parameter settings can cause damage to the connected mechanics.

- Change parameter values only after careful consideration.
- Save the current parameter values to the PC (p. 275) before you make changes in the nonvolatile memory.

***INFORMATION***

The following procedure is generally recommended for changing parameter values:

1. Change the parameter values in the volatile memory.
2. Check whether the C-867 works correctly with the changed parameter values.

If so:

- Write the changed parameter values into the nonvolatile memory.

If not:

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

***INFORMATION***

The write access for the parameters of the C-867 is defined by command levels. After the controller is switched on or rebooted, the active command level is always 0. On command levels > 1, write access is only available to PI service personnel.

- Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 309).

#### Requirements

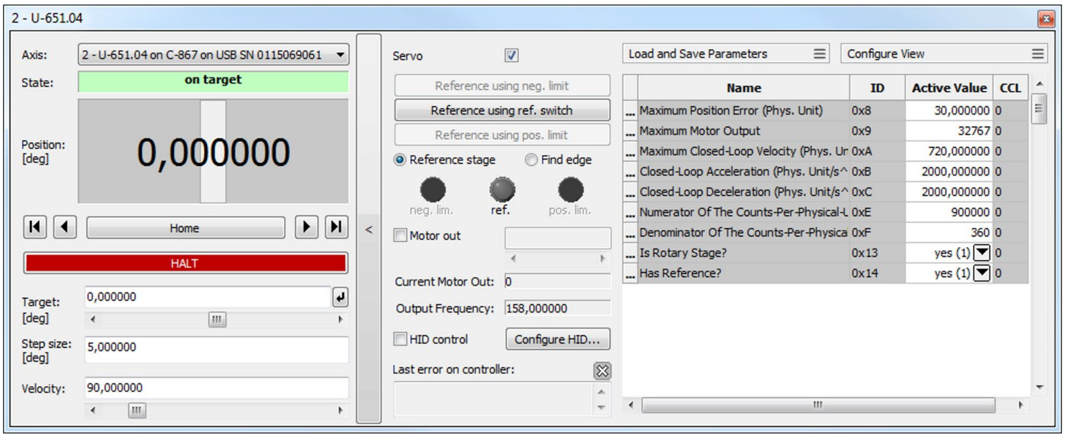
- ✓ If you want to change parameter values in the nonvolatile memory of the C-867: You have saved the parameter values of the C-867 in a text file on the PC (p. 275).
- ✓ You have established communication between the C-867 and the PC with PIMikroMove (p. 70).

### Changing parameter values: General procedure

1. Display the parameter list in PIMikroMove.

   If you want to change the axis-related parameters of the C-867:

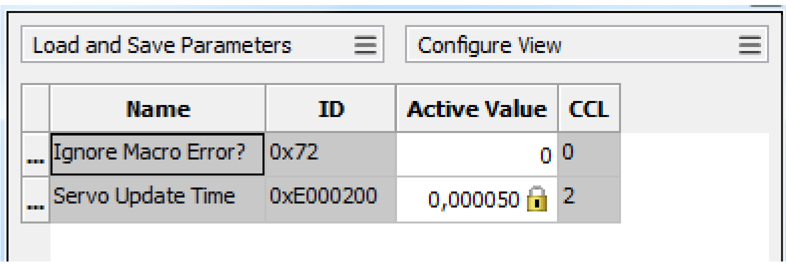
   a) Open the expanded single axis window for the connected positioner in the main window of PIMikroMove by clicking the right mouse button on the corresponding line of the ***Axes*** tab and selecting ***Show Expanded Single Axis Window*** in the context menu.




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

   If you want to change the system-related parameters of the C-867:

   – Open the window for the system-related parameters of the C-867 in the main window of PIMikroMove by selecting ***C-867 > Show system parameters*** in the menu.




2. Change the desired parameter values in the volatile or nonvolatile memory of the C-867 in the corresponding parameter list.

   If you want to change parameter values in the volatile memory, you have the following options:

   – Type the new parameter value into the corresponding input field in the ***Active Value*** column of the list. Press the Enter key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the volatile memory of the C-867.

   – Click ***Load and Save Parameters -> Load all startup parameters of the axis / system from controller*** in order to load the values of all axis-related / system-related parameters from the nonvolatile memory of the C-867.

- Click ***Load and Save Parameters > Load parameters from stage database…*** in the extended single-axis window to load a selected parameter set for the axis from the positioner database. You can use ***Load and Save Parameters > Reload parameters from stage database…*** to reload the currently loaded parameter set.

If you want to change parameter values in the nonvolatile memory, you have the following options:

- Type the new parameter value into the appropriate input field in the ***Startup Value*** column in the list. Press the Enter key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the nonvolatile memory of the C-867.
- Click ***Load and Save Parameters -> Save all currently active axis / system parameters as startup parameters to controller*** to write the values of all axis-related / system-related parameters from the volatile to the nonvolatile memory of the C-867. You can skip parameters that do not have write access on the current command level.

If a parameter value in the volatile memory (***Active Value*** column) is different from the parameter value in the nonvolatile memory (***Startup Value*** column), the line in the list is highlighted in color.

## 9.3 Creating or Changing a Positioner Type

You can select a 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 volatile or nonvolatile memory of the controller. For further information, see "Positioner Databases" (p. 14).

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.

***INFORMATION***

Possibilities for creating and editing parameter sets in the PISTAGES3.DB database:

- You can create a new positioner type easily by modifying an existing positioner type in PIMikroMove and saving it under a new name.
- You can open and edit the positioner database directly with the PIStages3Editor, which is included in the PI Software Suite.

PIMikroMove is used in the following for creating a new positioner type and for changing an existing positioner type.

## Requirements

- ✓ You have installed the latest version of the PISTAGES3.DB database onto your PC (p. 64).
- ✓ If PI provided a custom positioner database for your positioner, the dataset was imported into PIStages3 (p. 66).
- ✓ You have established communication with PIMikroMove between the C-867 and the PC (p. 70).

## Creating a positioner type in the positioner database

1. In the main window of PIMikroMove, select the ***C-867 > Select connected stages...*** menu item.

   The ***Start up stages/axes for C-867*** window opens, the step ***Select connected stages*** is active.

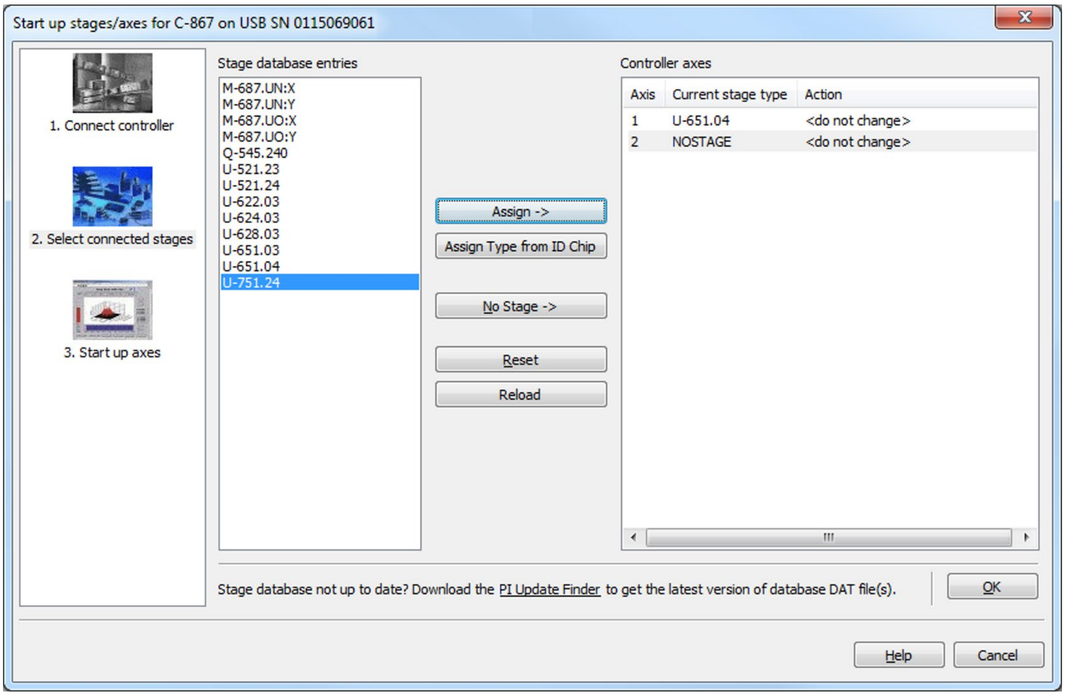
2. Select an appropriate type of positioner during the ***Select connected stages*** step:

   – Click on ***Assign Type from ID Chip***.

   or

   a) Mark the positioner type in the ***Stage database entries*** list.

   b) Click ***Assign***.



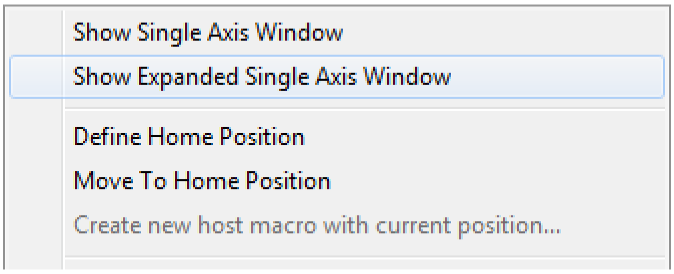

   c) Confirm the selection with ***OK***.

3. In the ***Save all changes permanently*** dialog, click ***Keep the changes temporarily*** to load the parameter settings into the volatile memory of the C-867.

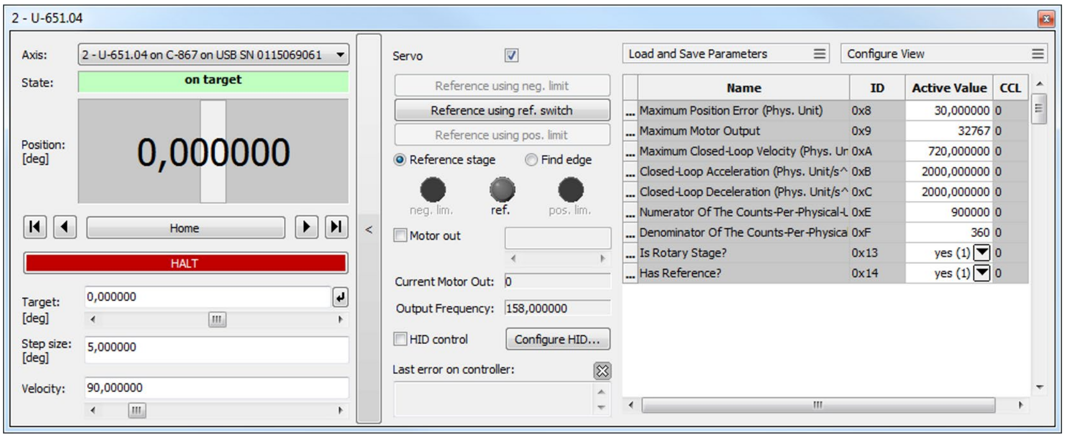
   The ***Start up stages/axes*** window changes to the step ***Start up axes***.

4. In the step ***Start up axes*** click on ***Close*** to close the ***Start up stages/axes*** window.

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




6. Enter new values for the parameters to be changed:

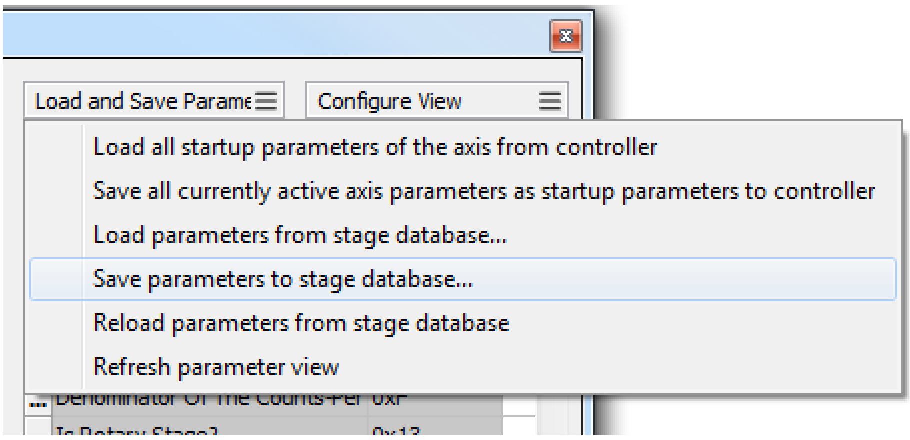



a) If the parameter to be modified is not included in the list on the right-hand side of the window, click on ***Configure view > Select parameters...*** and add it to the list. You can also display certain groups of parameters or all axes-related parameters.

b) Type the new parameter value into the corresponding input field in the ***Active Value*** column of the list.

c) Press the Enter key on the PC keyboard or click outside the input field with the mouse to transfer the parameter value to the volatile memory of the controller. Note: If a parameter value in the volatile memory (***Active Value*** column) is different to the parameter value in the nonvolatile memory (***Startup Value*** column), the line in the list is highlighted in color.

7. Click on ***Load and Save Parameters -> Save parameters to stage database...***.




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

8. 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 positioner database. The display of the connected positioner type was updated in the single axis window and in the main window of PIMikroMove. The new positioner type is also available immediately for selection in the ***Select connected stages*** step.

### Changing a positioner type in the positioner database

1. Select the ***C-867 > Select connected stages...*** menu item in the main window of PIMikroMove.

   The ***Start up stages/axes for C-867*** window opens, the ***Select connected stages*** step is active.

2. Select one of the positioners you created as described above (p. 279): Proceed with the selection as described in step 2 of the **Creating a positioner type in the positioner database** instruction.

3. Proceed with steps 3 to 7 in **Creating a positioner type in the positioner database**.

4. Save the modified parameter values of the positioner type in the ***Save Parameters as User Stage Type*** dialog:

   a) Leave the entry in the ***Parameters of axis*** field unchanged.

   b) Leave the entry in the ***Save as*** field unchanged.

   c) Click ***OK***.

   d) Click ***Change settings i***n the ***Stage type already defined*** dialog. The ***Save Parameters as User Stage Type*** dialog closes automatically after a short time.

   The parameter values of the positioner type have been updated in the PISTAGES3.DB positioner database and in the main window of PIMikroMove.

## 9.4 Parameter Overview

***INFORMATION***

The write access for the parameters of the C-867 is defined by command levels. After the controller is switched on or rebooted, the active command level is always level 0. For particular parameters, write access is only allowed on command level 1. On command levels > 1, write access is only available to PI service personnel.

The C-867 ignores the active command level in the following cases:

- The C-867 reads parameter values from the ID chip of the positioner.
- The positioner type is selected in the PC software.
- The current parameter values are written from the volatile to the nonvolatile memory (directly with WPA or in the PC software).

➢ If necessary, send the `CCL 1 advanced` command or enter the password `advanced` to change to command level 1.

➢ Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 309).

***INFORMATION***

To save parameter values in the nonvolatile memory, it is necessary to enter a password.
Usable passwords:

| | |
|---|---|
| 100 | Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT<br>Use with the WPA and SEP commands |
| 101 | Saves the currently valid values of all parameters<br>Use with the WPA command |

Designations in the header of the following table:

- ID = Parameter ID, hexadecimal format
- Type = Data type:
  - INT = integer value, including Boolean values
  - FLOAT = floating point number
  - CHAR = String format
- CL = Command Level for write access
- Element = Element type that the parameter refers to, refer to "Commandable Items" (p. 18) for further information
- Parameter name = Name of the parameter
- Description = Explanation of the parameter

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x8 | FLOAT | 0 | Axis | Maximum Position Error (Phys. Unit) | Maximum position error<br>Is used for detecting motion errors.<br>Refer to Behavior with Motion Error (p. 93) for details |
| 0x9 | INT | 0 | Axis | Maximum Motor Output | Maximum permissible absolute measure of the control value (dimensionless)<br>For details see Supported Motor Types (p. 23) |
| 0xA | FLOAT | 0 | Axis | Maximum Closed-Loop Velocity (Phys. Unit/s) | Maximum velocity in closed-loop operation with dynamics profile<br>Specifies the maximum value for parameter 0x49.<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0xB | FLOAT | 0 | Axis | Closed-Loop Acceleration (Phys. Unit/$s^2$) | Acceleration in closed-loop operation with dynamics profile<br>Limited by parameter 0x4A.<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0xC | FLOAT | 0 | Axis | Closed-Loop Deceleration (Phys. Unit/$s^2$) | Deceleration in closed-loop operation with dynamics profile<br>Limited by parameter 0x4B.<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0xE | INT | 0 | Axis | Numerator Of The Counts-Per-Physical-Unit Factor | Numerator and denominator of the factor for counts per physical length unit<br>For details, refer to Physical Units (p. 22) |
| 0xF | INT | 0 | Axis | Denominator Of The Counts-Per-Physical-Unit Factor | |
| 0x13 | INT | 0 | Axis | Is Rotation Stage? | Is this a rotation stage?<br>0 = Not a rotation stage<br>1 = Rotation stage<br>No evaluation by the C-867, but only by the PC software: PIMikroMove determines which motion is permissible on the basis of this value. |
| 0x14 | INT | 0 | Axis | Has Reference? | Do the mechanics have a reference switch?<br>For details, refer to Reference Switch Detection (p. 45) |
| 0x15 | FLOAT | 0 | Axis | Maximum Travel In Positive Direction (Phys. Unit) | Soft limit in positive direction<br>See examples in Travel Range and Soft Limits (p. 48) |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x16 | FLOAT | 0 | Axis | Value At Reference Position (Phys. Unit) | Position value at the reference switch<br>See examples in Travel Range and Soft Limits (p. 48) |
| 0x17 | FLOAT | 0 | Axis | Distance From Negative Limit To Reference Position (Phys. Unit) | Distance between the reference switch and the negative limit switch<br>See examples in Travel Range and Soft Limits (p. 48) |
| 0x18 | INT | 0 | Axis | Limit Mode | Signal logic of the limit switches<br>Refer to Detecting Limit Switches (p. 46) for details |
| 0x1B | INT | 0 | Axis | Profile Mode | Type of dynamics profile<br>0 = Trapezoidal, point-to-point<br>Refer to Generation of the Dynamics Profile (p. 27) |
| 0x2F | FLOAT | 0 | Axis | Distance From Reference Position To Positive Limit (Phys. Unit) | Distance between reference switch and positive limit switch<br>See examples in Travel Range and Soft Limits (p. 48) |
| 0x30 | FLOAT | 0 | Axis | Maximum Travel In Negative Direction (Phys. Unit) | Soft limit in a negative direction<br>See examples in Travel Range and Soft Limits (p. 48) |
| 0x31 | INT | 0 | Axis | Invert Reference? | Should the reference signal be inverted?<br>For details, refer to Reference Switch Detection (p. 45) |
| 0x32 | INT | 0 | Axis | Has No Limit Switches? | Does the positioner have limit switches?<br>Refer to Detecting Limit Switches (p. 46) for details |
| 0x33 | INT | 0 | Axis | Motor Offset Positive | Offset for the positive direction of motion<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x34 | INT | 0 | Axis | Motor Offset Negative | Offset for the negative direction of motion<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x35 | INT | 1 | Axis | Motor Output - Phase's Magnitude Dependencies | Behavior of both motor phases:<br>0 = constant<br>1 = correlation<br>2 = adaptive |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| | | | | | Refer to Optional Two-Phase Control (p. 43) for details |
| 0x36 | INT | 0 | Axis | Settling Window (encoder counts) | Available for compatibility reasons only.<br>Value identical to the value of parameter 0x406. |
| 0x3C | CHAR | 0 | Axis | Stage Name | Positioner name<br>Maximum of 20 characters; default value: NOSTAGE<br>The value NOSTAGE "deactivates" the axis. A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries). |
| 0x3F | FLOAT | 0 | Axis | Settling Time (s) | Delay time for setting the on-target state.<br>Refer to On-Target State (p. 44) for details |
| 0x47 | INT | 0 | Axis | Reference Travel Direction | Default direction for the reference move<br>Refer to Referencing (p. 50) for details |
| 0x48 | INT | 0 | Axis | Motor Drive Offset | Velocity-dependent offset<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x49 | FLOAT | 0 | Axis | Closed-Loop Velocity (Phys. Unit/s) | Velocity in closed-loop operation with dynamics profile<br>Limited by parameter 0xA<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0x4A | FLOAT | 0 | Axis | Maximum Closed-Loop Acceleration (Phys. Unit/$s^2$) | Maximum acceleration in closed-loop operation with dynamics profile<br>Specifies the maximum value for parameter 0xB.<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0x4B | FLOAT | 0 | Axis | Maximum Closed-Loop Deceleration (Phys. Unit/$s^2$) | Maximum deceleration in closed-loop operation with dynamics profile<br>Specifies the maximum value for parameter 0xC.<br>Refer to Generation of the Dynamics Profile (p. 27) for details |
| 0x4D | INT | 0 | Axis | Servo Window Mode | Reference variable for position windows for switching between |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| | | | | | parameter groups<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| For information on parameter 0x4F, see below: "Parameters for setting the electronic camming" | | | | | |
| 0x50 | FLOAT | 0 | Axis | Velocity For Reference Moves (Phys. Unit/s) | Maximum velocity for reference move<br>Refer to Referencing (p. 50) for details |
| 0x51 | FLOAT | 0 | Axis | Motor Output Frequency (kHz) | Piezo voltage frequency output to the motor (1st motor phase)<br>For details see Supported Motor Types (p. 23) |
| 0x52 | INT | 0 | Axis | Dynamic Frequency Control | State of the frequency control<br>Refer to Automatic Frequency Control (p. 25) for details |
| 0x53 | FLOAT | 0 | Axis | Minimum Motor Output Frequency (kHz) | Minimum piezo voltage frequency (1st motor phase)<br>Refer to Automatic Frequency Control (p. 25) for details |
| 0x54 | FLOAT | 0 | Axis | Maximum Motor Output Frequency (kHz) | Maximum piezo voltage frequency (1st motor phase)<br>Refer to Automatic Frequency Control (p. 25) for details |
| 0x55 | INT | 0 | Axis | Minimum Motor Output For Dynamic Frequency Control | Minimum control value for activating the frequency control<br>Refer to Automatic Frequency Control (p. 25) for details |
| 0x56 | INT | 0 | Axis | Sensor Power Supply | Supply voltage for sensor activated?<br>0 = Supply voltage disabled<br>1 = Supply voltage enabled |
| 0x5A | INT | 0 | Axis | Numerator Of The Servo-Loop Input Factor | Numerator and denominator of the servo-loop input factor<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x5B | INT | 0 | Axis | Denominator Of The Servo-Loop Input Factor | |
| 0x5C | INT | 0 | Axis | Source Of Reference Signal | Reference signal source for axis motion to the reference switch<br>Refer to Commands and Parameters for Digital Inputs (p. 109) and Using Digital Input Signals as Switch Signals (p. 111) for details |
| 0x5D | INT | 0 | Axis | Source Of Negative Limit Signal | Reference signal source for axis motion to the negative travel range |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| | | | | | limit<br>Refer to Commands and Parameters for Digital Inputs (p. 109) and Using Digital Input Signals as Switch Signals (p. 111) for details |
| 0x5E | INT | 0 | Axis | Source Of Positive Limit Signal | Reference signal source for axis motion to the positive travel range limit<br>Refer to Commands and Parameters for Digital Inputs (p. 109) and Using Digital Input Signals as Switch Signals (p. 111) for details |
| 0x5F | INT | 0 | Axis | Invert Digital Input Used For Negative Limit | Inverts the polarity of the digital inputs that are used as the source of the negative limit switch signal<br>Refer to Commands and Parameters for Digital Inputs (p. 109) and Using Digital Input Signals as Switch Signals (p. 111) for details |
| 0x60 | INT | 0 | Axis | Invert Digital Input Used For Positive Limit | Inverts the polarity of the digital inputs that are used as the source of the positive limit switch signal<br>Refer to Commands and Parameters for Digital Inputs (p. 109) and Using Digital Input Signals as Switch Signals (p. 111) for details |
| 0x61 | INT | 0 | Axis | Invert Direction Of Motion For Joystick-Controlled Axis? | Should the direction of motion for HID-controlled axes be inverted?<br>Refer to Commands and Parameters for HID Control (p. 115) |
| 0x62 | FLOAT | 0 | Axis | Window 0 Delay (s) | Delay time for activating parameter group 0<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x63 | FLOAT | 0 | Axis | Distance Between Limit And Hard Stop (Phys. Unit) | Distance between the built-in limit switch and the hard stop<br>Refer to Referencing (p. 50) for details |
| 0x64 | INT | 0 | Axis | Motor Output Frequency Shift | Phase shift between current and voltage on the drive |
| 0x69 | FLOAT | 1 | Axis | Motor Output Phase 2 - Frequency (kHz) | Parameter for configuring the "Slow Motion" feature (C-867.L01).<br>For details, refer to the C-867.L01 user manual |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x6A | INT | 0 | Axis | Motor Output Two-Phase Increment | Increase in the amplitude of the 2nd phase of the motor per servo cycle<br>Refer to Optional Two-Phase Control (p. 43) for details |
| 0x6B | INT | 0 | Axis | Motor Output Always On | Maintain output piezo voltage after the target position has been reached?<br>0 = No, set power to 0<br>1 = Yes, maintain power |
| 0x6C | INT | 0 | Axis | Motor Output Two-Phase Decrement | Decrease in the amplitude of the 2nd phase of the motor per servo cycle<br>Refer to Optional Two-Phase Control (p. 43) for details |
| 0x6D | INT | 1 | Axis | Motor Output Phase 2 - Inverted | Parameter for configuring the "Slow Motion" feature (C-867.L01).<br>For details, refer to the C-867.L01 user manual |
| 0x6E | INT | 1 | Axis | Motor Output Phase 2 - Frequency Dependency | Frequency behavior of the second motor phase<br>0 = synchronous<br>1 = free run (for "Slow Motion" only)<br>2 = follow (for "Slow Motion" only)<br>Refer to Optional Two-Phase Control (p. 43) for details |
| 0x6F | INT | 0 | Axis | Motor Output Two-Phase Magnitude | Maximum amplitude of the 2nd phase of the motor<br>Refer to Optional Two-Phase Control (p. 43) for details |
| 0x70 | INT | 0 | Axis | Reference Signal Type | Reference signal type<br>For details, refer to Reference Switch Detection (p. 45) |
| 0x71 | INT | 0 | Axis | D-Term Delay (No. Of Servo Cycles) | D term delay<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x72 | INT | 0 | System | Ignore Macro Error? | Ignore macro error?<br>Refer to Commands and Parameters for Macros (p. 127) for details |
| 0x74 | FLOAT | 0 | Axis | Closed-Loop Velocity For HI Control (Phys. Unit/s) | Maximum velocity during HID control<br>Refer to Commands and Parameters for HIDs (p. 115) for details |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x75 | FLOAT | 0 | Axis | Closed-Loop Acceleration For HI Control (Phys. Unit/s²) | Maximum acceleration during HID control<br>Refer to Commands and Parameters for HIDs (p. 115) for details |
| 0x76 | FLOAT | 0 | Axis | Closed-Loop Deceleration For HI Control (Phys. Unit/s²) | Maximum deceleration during HID control<br>Refer to Commands and Parameters for HIDs (p. 115) for details |
| 0x77 | INT | 0 | Axis | Use Limit Switches Only For Reference Moves? | Should the limit switches only be used for reference moves?<br>Refer to Detecting Limit Switches (p. 46) for details |
| 0x78 | FLOAT | 0 | Axis | Distance From Limit To Start Of Ref. Search (Phys. Unit) | Distance between the limit switch or hard stop and the starting position for the reference move to the index pulse<br>Refer to Referencing (p. 50) for details |
| 0x79 | FLOAT | 0 | Axis | Distance For Reference Search (Phys. Unit) | Maximum distance for the reference move to the index pulse<br>Refer to Referencing (p. 50) for details |
| 0x7B | FLOAT | 0 | Axis | Maximum Motor Output Time (s) | Maximum time period for which a high control value can be set in closed-loop operation.<br>Refer to Protecting Against Overheating (p. 93) for details |
| 0x7C | FLOAT | 0 | Axis | Maximum Motor Output (V) | Maximum permissible piezo voltage.<br>For details see Supported Motor Types (p. 23) |
| 0x400 | INT | 0 | Axis | Number Of Control Parameter Groups | Number of parameter groups used<br>Refer to Control Algorithm and Other Control Value Corrections (p. 30) for details |
| 0x401 | INT | 0 | Axis | P Term 0 | Proportional constant of parameter group 0 |
| 0x402 | INT | 0 | Axis | I Term 0 | Integral constant of parameter group 0 |
| 0x403 | INT | 0 | Axis | D Term 0 | Differential constant of parameter group 0 |
| 0x404 | INT | 0 | Axis | I Limit 0 | Limitation of the integral constant of parameter group 0 |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x405 | INT | 0 | Axis | Kvff 0 | Feed-forward control of the commanded velocity for parameter group 0 |
| 0x406 | INT | 0 | Axis | Window Enter 0 (Encoder Counts) | Beginning of the position window of parameter group 0 (activation of the parameters) |
| 0x407 | INT | 0 | Axis | Window Exit 0 (Encoder Counts) | End of the position window of parameter group 0 (deactivation of the parameters) |
| 0x411 | INT | 0 | Axis | P Term 1 | Proportional constant of parameter group 1 |
| 0x412 | INT | 0 | Axis | I Term 1 | Integral constant of parameter group 1 |
| 0x413 | INT | 0 | Axis | D Term 1 | Differential constant of parameter group 1 |
| 0x414 | INT | 0 | Axis | I Limit 1 | Limitation of the integral constant of parameter group 1 |
| 0x415 | INT | 0 | Axis | Kvff 1 | Feed-forward control of the commanded velocity for parameter group 1 |
| 0x416 | INT | 0 | Axis | Window Enter 1 (Encoder Counts) | Beginning of the position window of parameter group 1 (activation of the parameters) |
| 0x417 | INT | 0 | Axis | Window Exit 1 (Encoder Counts) | End of the position window of parameter group 1 (deactivation of the parameters) |
| 0x421 | INT | 0 | Axis | P Term 2 | Proportional constant of parameter group 2 |
| 0x422 | INT | 0 | Axis | I Term 2 | Integral constant of parameter group 2 |
| 0x423 | INT | 0 | Axis | D Term 2 | Differential constant of parameter group 2 |
| 0x424 | INT | 0 | Axis | I Limit 2 | Limitation of the integral constant of parameter group 2 |
| 0x425 | INT | 0 | Axis | Kvff 2 | Feed-forward control of the commanded velocity for parameter group 2 |
| 0x426 | INT | 0 | Axis | Window Enter 2 (Encoder Counts) | Beginning of the position window of parameter group 2 (activation of the parameters) |
| 0x427 | INT | 0 | Axis | Window Exit 2 (Encoder Counts) | End of the position window of parameter group 2 (deactivation of the parameters) |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x431 | INT | 0 | Axis | P Term 3 | Proportional constant of parameter group 3 |
| 0x432 | INT | 0 | Axis | I Term 3 | Integral constant of parameter group 3 |
| 0x433 | INT | 0 | Axis | D Term 3 | Differential constant of parameter group 3 |
| 0x434 | INT | 0 | Axis | I Limit 3 | Limitation of the integral constant of parameter group 3 |
| 0x435 | INT | 0 | Axis | Kvff 3 | Feed-forward control of the commanded velocity for parameter group 3 |
| 0x436 | INT | 0 | Axis | Window Enter 3 (Encoder Counts) | Beginning of the position window of parameter group 3 (activation of the parameters) |
| 0x437 | INT | 0 | Axis | Window Exit 3 (Encoder Counts) | End of the position window of parameter group 3 (deactivation of the parameters) |
| 0x441 | INT | 0 | Axis | P Term 4 | Proportional constant of parameter group 4 |
| 0x442 | INT | 0 | Axis | I Term 4 | Integral constant of parameter group 4 |
| 0x443 | INT | 0 | Axis | D Term 4 | Differential constant of parameter group 4 |
| 0x444 | INT | 0 | Axis | I Limit 4 | Limitation of the integral constant of parameter group 4 |
| 0x445 | INT | 0 | Axis | Kvff 4 | Feed-forward control of the commanded velocity for parameter group 4 |
| 0x446 | INT | 0 | Axis | Window Enter 4 (Encoder Counts) | Beginning of the position window of parameter group 4 (activation of the parameters) |
| 0x447 | INT | 0 | Axis | Window Exit 4 (Encoder Counts) | End of the position window of parameter group 4 (deactivation of the parameters) |
| For information on parameters 0x451 to 0x485, see below: "Parameters for setting the adapted PID servo control" | | | | | |
| 0x1000 | INT | 1 | Axis | Control Algorithm Mode | Type of servo algorithm<br>0 = Position-dependent PID control<br>1 = Adaptive PID control (evaluated only when the "Adaptive Control" feature is activated, refer to Activating Additional Features (p. 85)) |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x3003300 | FLOAT | 2 | Axis | Sensor Interpolation | Interpolation rate for the signals of the incremental sensor |
| 0x3003301 | FLOAT | 2 | Axis | Sensor Hysteresis (Deg) | Correction of the incremental sensor hysteresis |
| 0x3003302 | FLOAT | 2 | Axis | Sensor Board Gain | Gain value for correcting the digitized signals of the incremental sensor |
| 0x3003303 | FLOAT | 2 | Axis | Sensor Digital Offset 0 (V) | Offset 0 for correcting the digitized signals of the incremental sensor |
| 0x3003304 | FLOAT | 2 | Axis | Sensor Digital Offset 1 (V) | Offset 1 for correcting the digitized signals of the incremental sensor |
| 0x3003305 | FLOAT | 2 | Axis | Sensor Digital Phase (Deg) | Phase correction for the signals of the incremental sensor |
| 0x3003306 | FLOAT | 2 | Axis | Sensor Analog Gain (dB) | Gain value for correcting the analog signals of the incremental sensor |
| 0x3003307 | FLOAT | 2 | Axis | Sensor Analog Offset 0 (V) | Offset 0 for correcting the analog signals of the incremental sensor |
| 0x3003308 | FLOAT | 2 | Axis | Sensor Analog Offset 1 (V) | Offset 1 for correcting the analog signals of the incremental sensor |
| 0x3003309 | FLOAT | 1 | Axis | Sensor Autoadaption | Activate function for auto-adaptation of sensor interpolation?<br>0 = Auto-adaptation deactivated<br>1 = Auto-adaptation activated (default setting, recommended for most applications)<br>The parameter must be set to 0 for the ***Sensor Analog Gain (dB)***, ***Sensor Analog Offset 0 (V)***, and ***Sensor Analog Offset 1 (V)*** settings to be evaluated. |
| 0x3003320 | INT | 2 | Axis | Sensor Signal Type | Signal type output by the position sensor:<br>0=No sensor<br>1=A/B<br>2=SIN/COS<br>3=BISS relative (32-bit)<br>4=BISS absolute (32-bit) |
| 0x3003330 | FLOAT | 1 | Axis | Sensor Position Offset (Phys. Unit/s) | Fixed offset to the position for absolute encoders |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x3003340 | INT | 3 | Axis | Sensor Error Counter | Error counter for absolute encoder:<br>The value of this parameter is incremented by 1 each time the position value of the encoder cannot be read out, e.g., due to communication problems.<br>This value therefore provides information on whether continuous communication with the encoder has taken place.<br>Restarting the controller, the parameter is reset. |
| For information on parameters 0x6010B60 to 0x6010B66, see below: "Parameters for setting the electronic camming" | | | | | |
| 0x7000000 | FLOAT | 0 | Axis | Range Limit Min | Additional soft limit for the negative direction of motion (physical unit)<br>Refer to Travel Range and Soft Limits (p. 47) for details |
| 0x7000001 | FLOAT | 0 | Axis | Range Limit Max | Additional soft limit for the positive direction of motion (physical unit)<br>Refer to Travel Range and Soft Limits (p. 47) for details |
| 0x07000601 | CHAR | 0 | Axis | Axis Unit | Unit symbol of the axis<br>For details, refer to Physical Units (p. 22) |
| 0xD000000 | CHAR | 2 | System | Device S/N | Serial number of the C-867<br>9-digit number |
| 0xE000102 | INT | 0 | Axis | Number Of Decimal Places | Number of decimal points for floating point numbers |
| 0xE000200 | FLOAT | 2 | System | Servo Update Time | Servo cycle time in seconds |
| 0xE000301 | FLOAT | 3 | System | Controller Disable Error 10 | Suppress error 10 when an axis motion is stopped?<br>0 = No (default setting)<br>1 = Yes<br>Setting this parameter to 1 prevents an error (error number: 10) from being output when axis motion is stopped with STP, #24, or HLT. |
| 0x0F000100 | CHAR | 2 | Axis | Stage Type | Positioner type<br>Format for standard positioners: x-xxx<br>Format for customized positioners: x-xxxKxxx |
| 0x0F000200 | CHAR | 2 | Axis | Stage Serial Number | Serial number of the positioner |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x0F000300 | CHAR | 2 | Axis | Stage Assembly Date | Date of manufacture of the positioner<br>Date format: DDMMYY |
| 0x0F000400 | INT | 2 | Axis | Stage HW Version | Version number of the positioner hardware |
| 0x16000001 | INT | 0 | System | Recorded Points Per Trigger | Number of data points to be recorded per trigger<br>Refer to Setting up the Data Recorder (p. 98) for details |
| 0x16000002 | INT | 0 | System | Clearing Of Record Table On Trigger | Write mode during the recording<br>Refer to Setting up the Data Recorder (p. 98) for details |
| 0x16000003 | INT | 0 | System | Data Recorder Buffer Mode | Behavior with full data recorder tables<br>Refer to Setting up the Data Recorder (p. 98) for details |
| 0x16000004 | INT | 3 | System | Data Recorder Buffer Overflow | Buffer overflow counter of the data recorder<br>Refer to Setting up the Data Recorder (p. 98) for details |
| 0x22000020 | INT | 2 | System | Maximum FIFO Buffer Size | Maximum number of trajectory points in the trajectory buffer<br>Refer to Trajectories for Motion Paths (p. 95) for details |
| 0xE002700 | CHAR | 1 | System | Slow Motion Feature License Number | License number for the C-867.L01 firmware feature: "Slow Motion"<br>Refer to Activating Additional Features (p. 85) |
| 0xE002701 | INT | 3 | System | Slow Motion Feature License Valid? | License number for the C-867.L01 firmware feature valid?<br>After rebooting, the controller checks whether the license number entered is valid and sets this parameter accordingly:<br>0 = License number invalid<br>1 = License number valid |
| 0xE002702 | INT | 1 | System | Enable Slow Motion Feature | Activate "Slow Motion" feature (C-867.L01)?<br>0 = not activated<br>1 = activated<br>Activating is only possible when the license number for the feature has been entered and is valid, refer to Activating Additional Features (p. |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| | | | | | 85) |
| 0xE002712 | INT | 1 | System | Enable Adaptive Control Feature | Activate "Adaptive Control"?<br>0 = not activated<br>1 = activated<br>Refer to Activating Additional Features (p. 85) |
| 0xE002722 | INT | 1 | System | Enable Camming Feature | Activate "Camming" feature?<br>0 = not activated<br>1 = activated<br>Refer to Activating Additional Features (p. 85) |

### Parameters for setting the adapted PID control

The parameters for setting the adaptive PID control (p. 36) are only visible when the "Adaptive Control" firmware feature is activated (p. 85).

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x451 | INT | 0 | Axis | P-Term 0 (Target) | Proportional constant for *Target* state of axis motion |
| 0x452 | INT | 0 | Axis | I-Term 0 (Target) | Integral constant for *Target* state of axis motion |
| 0x453 | INT | 0 | Axis | D-Term 0 (Target) | Differential constant for axis *Target* state |
| 0x454 | INT | 0 | Axis | I-Limit 0 (Target) | Limitation of integral constant for *Target* state of axis motion |
| 0x455 | INT | 0 | Axis | Window enter 0 (Target) | Position window for activating the parameter group 0 (*Target*) |
| 0x456 | INT | 0 | Axis | Window exit 0 (Target) | Position window for deactivating parameter group 0 (*Target*) |
| 0x461 | INT | 0 | Axis | P-Term 1 (Global Stable) | Proportional constant for *Global Stable* state of axis motion |
| 0x462 | INT | 0 | Axis | I-Term 1 (Global Stable) | Integral constant for *Global Stable* state of axis motion |
| 0x463 | INT | 0 | Axis | D-Term  (Global Stable) | Differential constant for axis *Global Stable* state |
| 0x464 | INT | 0 | Axis | I-Limit 1 (Global Stable) | Limitation of the integral constant for axis *Global Stable* state |
| 0x471 | INT | 0 | Axis | P-Term 2 (Motion) | Proportional constant for *Motion* state of axis motion |
| 0x472 | INT | 0 | Axis | I Term 2 (Motion) | Integral constant for *Motion* state of axis motion |

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x473 | INT | 0 | Axis | D Term 2 (Motion) | Differential constant for *Motion* state of axis motion |
| 0x474 | INT | 0 | Axis | I-Limit 2 (Motion) | Limitation of the integral constant for *Motion* state of axis motion |
| 0x475 | INT | 0 | Axis | KVFF 2 (Motion) | Feed-forward control of the commanded velocity in the *Motion* status of axis motion |
| 0x476 | INT | 0 | Axis | Velocity adaptive PID-Terms? | Adapt the PID values for the *Motion* state depending on velocity? |
| 0x477 | INT | 0 | Axis | P-Term Max at min. Velocity (Motion) | P term for the *Motion* state at minimum velocity |
| 0x478 | FLOAT | 0 | Axis | Max. Velocity for adaptive PID-Term (Motion) | Maximum and minimum value of the velocity for velocity-adaptive PID control<br>The amplitude of the second motor phase is adjusted only if the current velocity is between the defined values. |
| 0x479 | FLOAT | 0 | Axis | Min. Velocity for adaptive PID-Term (Motion) | |
| 0x47A | FLOAT | 0 | Axis | Velocity to detect end of Motion (Phys. Unit/s) (Motion) | Threshold value of the velocity for switching to the *End Position* state |
| 0x47B | INT | 0 | Axis | Max I-Term at min. Velocity (Motion) | I term for the *Motion* state at minimum velocity |
| 0x47C | INT | 0 | Axis | Max D-Term at min. Velocity (Motion) | D term for the *Motion* state at minimum velocity |
| 0x47D | INT | 0 | Axis | Phase Two Motor Output - Max. Magnitude at min. Velocity | Maximum amplitude of the second motor phase at minimum velocity |
| 0x481 | INT | 0 | Axis | P-Term 3 (End Position) | Proportional constant for *End Position* state of axis motion |
| 0x482 | INT | 0 | Axis | I Term 3 (End Position) | Integral constant for *End Position* state of axis motion |
| 0x483 | INT | 0 | Axis | D Term 3 (End Position) | Differential constant for *>End Position* state of axis motion |
| 0x484 | INT | 0 | Axis | I-Limit 3 (End Position) | Limitation of the integral constant for *End Position* state of axis motion |
| 0x485 | FLOAT | 0 | Axis | Velocity to detect vibration (Phys. Unit/s) (End Position) | Threshold value of the velocity for switching to the *Global Stable* state |

### Parameters for setting the electronic camming

The parameters for setting the electronic camming (p. 41) are only visible when the "Camming" firmware feature is activated (p. 85).

| ID | Type | CL | Element | Parameter name | Description |
|---|---|---|---|---|---|
| 0x4F | INT | 0 | System | Electronic Camming Mode | Activate camming?<br>0 = Camming not activated<br>1 = Camming activated<br>If the camming is activated (= 1), the 2nd axis moves automatically when the 1st axis is moved. |
| 0x6010B60 | FLOAT | 1 | System | Camming Polynomial - Coefficient 0 | Value for coefficient 0 for adapting the camming |
| 0x6010B61 | FLOAT | 1 | System | Camming Polynomial - Coefficient 1 | Value for coefficient 1 for adapting the camming |
| 0x6010B62 | FLOAT | 1 | System | Camming Polynomial - Coefficient 2 | Value for coefficient 2 for adapting the camming |
| 0x6010B63 | FLOAT | 1 | System | Camming Polynomial - Coefficient 3 | Value for coefficient 3 for adapting the camming |
| 0x6010B64 | FLOAT | 1 | System | Camming Polynomial - Coefficient 4 | Value for coefficient 4 for adapting the camming |
| 0x6010B65 | FLOAT | 1 | System | Camming Polynomial - Coefficient 5 | Value for coefficient 5 for adapting the camming |
| 0x6010B66 | FLOAT | 1 | System | Camming Polynomial - Coefficient 6 | Value for coefficient 6 for adapting the camming |

# 10 Maintenance

## 10.1 Cleaning the C-867

*NOTICE*

**Short circuits or flashovers!**

The C-867 contains electrostatic-sensitive devices that can be damaged by short-circuiting or flashovers when cleaning fluids penetrate the housing.

- Before cleaning, disconnect the C-867 from the power source by removing the mains plug.
- Prevent cleaning fluid from penetrating the housing.

- When necessary, clean the surfaces of the C-867's housing using a cloth dampened with a mild cleanser or disinfectant.

## 10.2 Updating Firmware

*INFORMATION*

The `*IDN?` command reads the version number of the firmware among other things.

Example of a C-867 response:

```
(c)2024 Physik Instrumente (PI) GmbH & Co. KG, C-867.2U2,
117048994, 01.400
```

- C-867.2U2: Device name
- 117048994: Serial number of the device.
- 01.400: Firmware version

*INFORMATION*

The **STA** LED flashes when the C-867 is in firmware update mode. The C-867 does not leave the firmware update mode until it is **restarted** after a **successful** firmware update. If the firmware update was unsuccessful or aborted, the C-867 remains in the firmware update mode after a reboot.

If the **STA** LED still flashes, even though the C-867 has been restarted after the firmware update:

- Repeat the firmware update.
- If the update of the firmware fails, contact our customer service department (p. 309).

### *INFORMATION*

If new parameters are introduced with the firmware update or the C-867 memory management is changed, an initialization of the C-867 is required after updating the firmware.

### Requirements

- ✓ You have connected the C-867 to the PC via the USB or RS-232 interface (p. 66).
- ✓ You have made sure that the C-867 is **not** a part of a daisy chain network.
- ✓ You have made sure that **no** cable is connected to the **RS-232 Out** socket.
- ✓ The ***PIFirmwareManager*** program is installed on the PC (p. 64).
- ✓ You have copied the new firmware file(s) that you have received from our customer service department to a directory on the PC. If you have received two firmware files (.hex and .jed), the update must be carried out for each of the files. This is done directly in succession in the update program.
- ✓ You have read and understood the documentation which you received from our customer service department together with the new firmware. You have learned from the documentation whether new parameters are introduced with the firmware update or the memory management of the C-867 changes.
- ✓ You have saved (p. 275) the parameter values of the C-867 to a text file on the PC.
- ✓ You have saved (p. 135) the C-867 controller macros to files on the PC.
- ✓ You have established (p. 70) communication between the C-867 and the PC with PIMikroMove or PITerminal.

### Updating the firmware of the C-867

1. Start the ***PIFirmwareManager*** program on the PC and update the controller firmware.

   Proceed as described in the user manual SM164E (p. 4).

### Restarting the C-867

1. Switch off the C-867.
2. Switch the C-867 on again.

   If the firmware update was successful, the C-867 exits the firmware update mode and the **STA** LED lights up continuously.

Have new parameters been added by the firmware update, or has the memory management of the C-867 been changed?

- If no: Firmware update is finished.
- If yes: An initialization of the C-867 is required, see below.

### Initializing the C-867 after a firmware update

The initialization of the C-867 resets **all** parameters to their factory settings and deletes all controller macros. Consequently, parameter values and controller macros that are not saved are lost during the initialization process.

1. Make sure that the current parameter values and controller macros of the C-867 have been saved on the PC.
2. On the PC, start PITerminal or PIMikroMove, connect to the C-867, and, if necessary, open the window to send commands.
3. Initialize theC-867, by sending the following commands one by one:

   `ZZZ 100 parameter`

   `ZZZ 100 macros`

   After successful initialization, the controller issues a corresponding message.
4. Adapt the parameter values of the C-867.

   For instructions on the general procedure, see "Changing Parameter Values: General Procedure" (p. 276).

   – Reset the parameters that were already present prior to the firmware update to the saved values from the text file.

   – Set the parameters that were introduced with the firmware update to the appropriate values.
5. If you have saved controller macros on the PC: Load the controller macros back to the C-867, see "Making Backups and Loading Controller Macros" (p. 135).

# 11 Troubleshooting

| Fault: Positioner does not move | |
|---|---|
| **Possible causes** | **Remedial measures** |
| Cable not connected correctly | ➢ Check the cable connections. |
| Positioner or connecting cable is defective | ➢ If available, replace the defective positioner with another positioner and test the new combination. |
| Positioner not connected to power adapter | ➢ Connect the positioner to a suitable power adapter and make sure that the power adapter is functioning properly. |
| The positioner has been connected to the switched-on C-867 | The sensor electronics in the positioner has not been initialized, and the ID chip of the positioner (p. 15) has not been read out.<br>➢ Switch the C-867 off and on again, or reboot the C-867 with the `RBT` command or with the corresponding functions of the PC software. |
| Unsuitable connecting cable used | If unsuitable cables are used, interference can occur in the signal transmission between the positioner and the C-867.<br>➢ If the positioner, cable, and C-867 are marked as a related system, replace the system components with other components only after consulting PI.<br>➢ If you need extension cables, contact our customer service department (p. 309). |
| Incorrect configuration | ➢ Check the parameter settings of the C-867 with the `SPA?` (volatile memory) and `SEP?` (nonvolatile memory) commands; see "Adapting Settings" (p. 273). |
| Incorrect command or incorrect syntax | ➢ Send the `ERR?` command and check the error code that is returned. |
| Wrong axis commanded | ➢ Make sure that the correct axis identifier is used and that the commanded axis belongs to the correct positioner. |
| HID control is active | Motion commands and following trajectories are not permitted when the HID control is activated for the axis.<br>➢ Deactivate HID control with the `HIN` command. |
| When executing a trajectory:<br>Trajectory buffer empty | Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with TGA (p. 234). During the execution of a trajectory, the buffer must be refilled fast enough. The execution of a trajectory must be completed with TGF (p. 236).<br>➢ Make sure that a sufficient number of trajectory points is always in the buffer. |

| Fault: Positioner performs unintentional motion | |
|---|---|
| **Possible causes** | **Remedial measures** |
| HID is not connected but HID control is activated in the C-867 | ➢ Activate HID control only when there an HID is actually connected to the C-867. |
| HID axis/axes not calibrated | ➢ Calibrate the HID (p. 120) axis/axes. |
| Startup macro is run | ➢ Check whether a macro is specified as the startup macro and cancel the selection of the startup macro if necessary (p. 128). |
| Wrong trajectory executed | ➢ Start the desired trajectory for the desired axis. |

| Fault: Positioner is oscillating or positions inaccurately | |
|---|---|
| **Possible causes** | **Remedial measures** |
| The load was changed. | ➢ Readjust the system according to the changed load (p. 88). |
| When executing a trajectory:<br>Unsuitable trajectory design | Following an unsuitable trajectory can cause the positioner to oscillate or stop motion abruptly. Oscillation or stopping abruptly can damage the positioner and/or the load fixed to it.<br>➢ Make sure that the path that is specified by the trajectory points is continuously differentiable at least twice.<br>➢ Design the trajectory (trajectory points and timing) so that the maximum permissible velocity and acceleration of the axis is **not** exceeded. |

| Fault: Positioner is already oscillating during the reference move | |
|---|---|
| **Possible causes** | **Remedial measures** |
| Very high load on the positioner | In case of a very high load, proceed with PIMikroMove during the reference move as follows:<br>1. Do **not** start the reference move in the ***Start up axes*** step, but click on ***Close*** to close the ***Start up controller*** window instead.<br>2. In the main window, open the single axis window for the positioner connected by selecting the positioner in the ***View > Single Axis Window*** menu.<br>3. Expand the view of the single axis window by clicking on the **>** button at the right edge of the window.<br>4. With the ***Servo*** check box, make sure that the servo mode is switched on. |

| Fault: Positioner is already oscillating during the reference move | |
|---|---|
| **Possible causes** | **Remedial measures** |
| | 5. Start the reference move by clicking on one of the ***Reference…*** buttons.<br>6. If the positioner is oscillating: Stop the reference move immediately in the ***Reference Axes*** dialog, close the dialog and switch off the servo mode by removing the tick from the respective check box in the single axis window.<br>7. Enter new values for the servo control parameters, see "Optimizing the Servo Control Parameters" (p. 88).<br>8. Restart the reference move.<br>9. If the positioner is still oscillating, repeat steps 6 to 8 until the reference move has completed successfully without oscillation. |

| Fault: There is no communication between the controller and the PC | |
|---|---|
| **Possible causes** | **Remedial measures** |
| The wrong communication cable is used or it is defective | ➢ Use the following cables for TCP/IP connections:<br>– TCP/IP direct connection to the PC: Crossover network cable<br>– TCP/IP network: Straight-through network cable<br>➢ Use a null modem cable for the RS-232 connection.<br>➢ If necessary, check whether the cable works on a fault-free system. |
| Communication interface is not correctly configured | **When using the RS-232 interface:**<br>➢ Check the port settings, the baud rate and the handshake setting of the PC.<br>**When using the TCP/IP connection:**<br>➢ Connect the controller to the network **before** you switch it on. Otherwise, you will have to switch the controller off and on again.<br>➢ Check the network settings (p. 73).<br>➢ 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.<br>➢ Make sure that the network does not block unknown devices.<br>➢ Make sure that several PC software applications cannot access the C-867 at the same time. |

| Fault: There is no communication between the controller and the PC | |
|---|---|
| **Possible causes** | **Remedial measures** |
| | ➢ Make sure that you have selected the correct C-867 when establishing communication.<br>➢ If you cannot solve the problems, consult your network administrator if necessary. |
| Baud rate not configured correctly | ➢ In a daisy chain network make sure that the same baud rate is set for every controller. |
| Another program is accessing the interface. | ➢ Close the other program. |
| Problems with special software | ➢ Check whether the system works with other software, such as a terminal program or a development environment.<br>You can test communication by starting a terminal program (e.g., PITerminal for example) and entering `*IDN?` or `HLP?`.<br>➢ Make sure that you end the commands with an LF (line feed).<br>A command is only executed when LF has been received. |

| Fault: The customer software does not function with the PI drivers | |
|---|---|
| **Possible causes** | **Remedial measures** |
| Incorrect combination of driver routines/VIs | ➢ Check whether the system functions with a terminal program (e.g., PITerminal).<br>If so:<br>➢ Read the information in the corresponding software manual and compare your program code with the sample code on the data storage device with the PI Software Suite. |

| Fault: Controller does not send an error code in the case of incorrect system behavior | |
|---|---|
| **Possible causes** | **Remedial measures** |
| Error code was already queried by another instance | In the case of simultaneous access to the controller by several instances, the error code is only returned to the first instance that sent the `ERR?` command. The error code is reset to 0 during the query.<br>➢ If possible, access the controller with one instance only.<br>➢ Check whether the error code is regularly queried in the background by a macro, a script or PC software (e.g., PIMikroMove). |

If the problem that occurred with your system is not in the list above or cannot be solved as described, contact our customer service department (p. 309).

# 12 Customer Service Department

For inquiries and orders, contact your PI representative or send us an email (mailto:service@pi.de).

- If you have 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)
  - PC operating system (if applicable)

If possible: Take photographs or make videos of your system that can be sent to our customer service department if requested.

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

# 13 Technical Data

Subject to change. You can find the latest product specifications on the product web page at www.pi.ws (https://www.physikinstrumente.com/en/).

## 13.1 Specifications

### 13.1.1 Data Table

| | C-867.2U2 |
|---|---|
| Function | Controller for XY positioning, scanning or microscope stages |
| Interfaces for communication | USB, RS-232, Ethernet, SPI |
| Axes | 2 |
| **Motion and control** | |
| Controller type | PID controller, parameter changing during operation |
| Motion profile | Point-to-point (trapezoidal velocity profile) or motion path (freely definable trajectory) |
| Encoder input | Sin/cos (differential), A/B (TTL, differential), BiSS interface for absolute encoders |
| Stall detection | Servo off, triggered by programmable position error or power level |
| Input limit switch | 2 programmable TTL lines per axis |
| Input reference switch | 1 programmable TTL line per axis |
| **Electrical properties** | |
| Max. output power | 15 W per axis |
| Max. output voltage | 200 $V_{pp}$ per axis |
| **Interfaces and operation** | |
| Communication interfaces | USB, TCP/IP, RS-232, SPI |
| Motor / sensor connection | 2 x Sub-D 15 (f) |
| Controller network | Daisy chain with up to 16 units on a single interface** |
| I/O lines | 4 analog inputs (0 to +5 V)<br>4 digital inputs (5 V TTL)<br>4 digital outputs (5 V TTL) |
| Command set | PI General Command Set (GCS) |
| User software | PIMikroMove, PITerminal |
| Application programming interfaces | C, C++, C#, MATLAB, NI LabVIEW, Python |

| | |
|---|---|
| Supported functions | Startup macro; data recorder for recording operating data such as motor voltage, velocity, position or position error; ID chip detection |
| Manual control | Pushbuttons<br>Analog or digital joystick |
| **Miscellaneous** | |
| Operating voltage | 24 V DC (external power supply in the scope of delivery) |
| Max. current consumption | 600 mA plus motor current (max. 4 A) |
| Operating temperature range | 5 °C to 40 °C |
| Mass | 1.62 kg |
| Dimensions | 312 mm × 153.4 mm × 59.3 mm (incl. mounting rails) |

### 13.1.2 Maximum Ratings

The C-867 is designed for the following operating data:

| **Input on:** | **Maximum operating voltage** ⚠ | **Operating frequency** ⚠ | **Maximum current consumption** ⚠ |
|---|---|---|---|
| M8 panel plug, 4-pin (m) | 30 V | ⎓ | 4 A |

| **Output on:**<br><br>**each (for axis 1 and axis 2)** | **Maximum piezo voltage** ⚠ | **Maximum frequency of the piezo voltage** ⚠ | **Maximum output current** ⚠ |
|---|---|---|---|
| Sub-D 15 (f) | 200 $V_{pp}$ (71 $V_{rms}$) | 500 kHz | 600 $mA_{pp}$ |

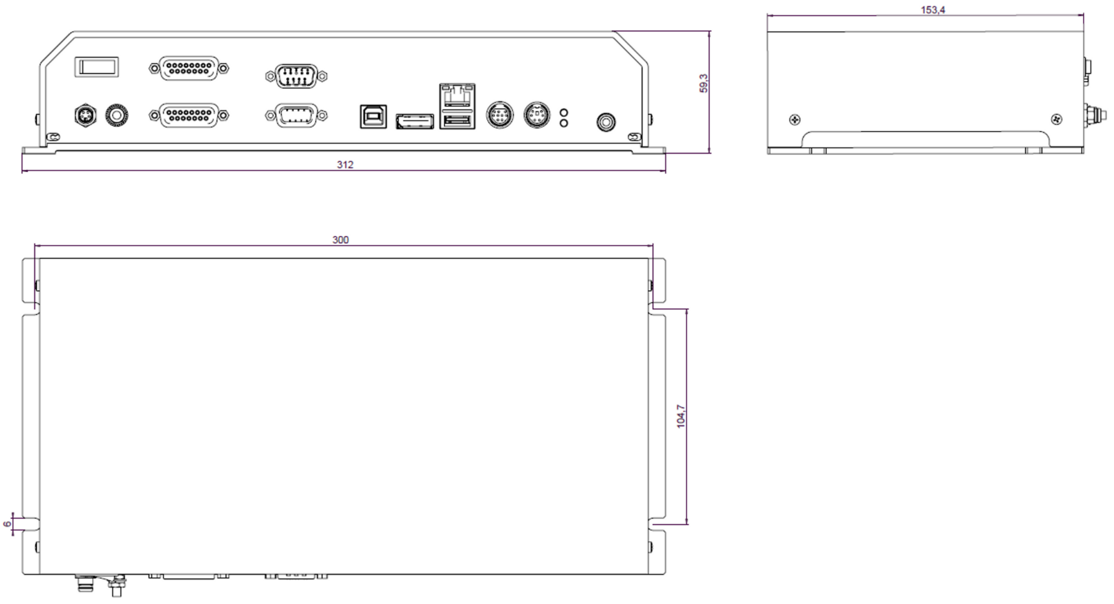
### 13.1.3 Ambient Conditions and Classifications

The following ambient conditions and classifications for the C-867 must be observed:

| | |
|---|---|
| Area of application | For indoor use only |
| Maximum altitude | 2000 m |
| Air pressure | 1100 hPa to 0.1 hPa |
| Relative humidity | Highest relative humidity 80 % for temperatures up to 31 °C<br>Decreasing linearly to 50 % relative air humidity at 40 °C |
| Storage temperature | 0 °C to 70 °C |
| Transport temperature | –25 °C to +85 °C |
| Overvoltage category | II |
| Protection class | I |
| Degree of pollution | 2 |
| Degree of protection according to IEC 60529 | IP20 |

## 13.2 Dimensions

Dimensions in mm. Note that the decimal points are separated by a comma in the drawings.



# 13.3 Pin Assignment

## 13.3.1 Sub-D 15 (f) motor connection

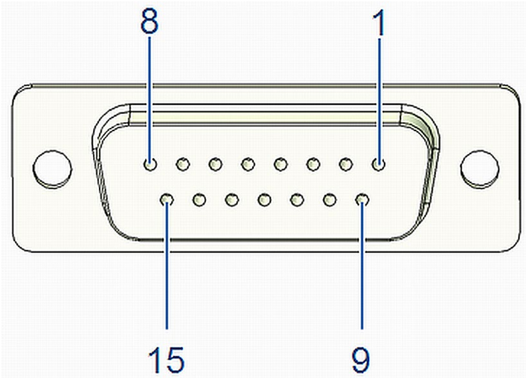
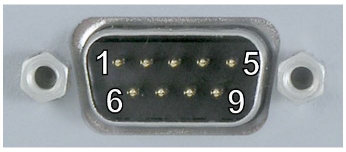



Figure 18: D-sub 15 socket

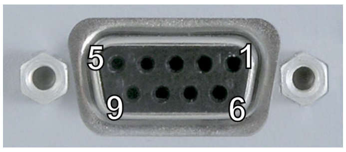
| Pin | Signal | Direction | Function | | | |
|---|---|---|---|---|---|---|
| 1 | NC | - | Not connected | | | |
| 2 | MOTOR GND | Output | Piezo | | | |
| 3 | MOTOR OUT 1 | Output | Piezo | | | |
| 4 | VDD | Output | +5 V | | | |
| 5 | PLIMIT | Input | Positive limit switch, TTL | | | |
| 6 | ID CHIP | Bidirectional | ID chip | | | |
| 7 | ENCA- | Input | Encoder: | A/B: A inverted, RS-422 | Sin/cos: SIN- | BiSS: MA- |
| 8 | ENCB- | Input | Encoder: | A/B: B inverted, RS-422 | Sin/cos: COS- | BiSS: SL- |
| 9 | MOTOR GND | Output | Piezo | | | |
| 10 | GND | - | 0 V | | | |
| 11 | MOTOR OUT 2 | Output | Piezo | | | |
| 12 | NLIMIT | Input | Negative limit switch, TTL | | | |
| 13 | REFSWITCH | Input | Reference switch, TTL | | | |
| 14 | ENCA+ | Input | Encoder: | A/B: A, RS-422 | Sin/cos: SIN+ | BiSS: MA+ |
| 15 | ENCB+ | Input | Encoder: | A/B: B, RS-422 | Sin/cos: COS+ | BiSS: SL+ |

### 13.3.2 RS-232 In and RS-232 Out

**RS-232 In: Sub-d 9 (m)**




**RS-232 Out: Sub-d 9 (w)**




| Pin | Function |
|---|---|
| 1 | Not connected |
| 2 | RxD (PC to controller) |
| 3 | TxD (controller to PC) |
| 4 | Not connected |
| 5 | GND |
| 6 | Not connected |
| 7 | Not connected |
| 8 | Not connected |
| 9 | Not connected |

***INFORMATION***

The pins of the **RS-232 In** and **RS-232 Out** sockets are connected to each other in the C-867 1:1.

***INFORMATION***

In a daisy chain network connected to the PC via the RS-232 interface of the first controller, only the PC feeds the RxD line. Depending on how performant the RS-232 driver of the PC is, the range of the network may be limited to 6 devices.

***INFORMATION***

The C-867 copies all signals that it receives from the PC via another communication interface (e.g., USB) to the RxD line of the **RS-232 In** and **RS-232 Out** sockets. The C-867 copies the signal of the TxD line via the corresponding interface to the PC.

## 13.3.3 I/O

### Mini-DIN socket, 9-pin, female

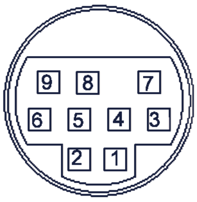



Figure 19: Front view of the mini-DIN socket

| Pin | Function |
|---|---|
| 1 | Input 1 (analog: 0 to +5V / digital: TTL) |
| 2 | Input 2 (analog: 0 to +5V/ digital: TTL) |
| 3 | Input 3 (analog: 0 to +5V/ digital: TTL) |
| 4 | Input 4 (analog: 0 to +5V/ digital: TTL) |
| 5 | Output 1 (digital: TTL) |
| 6 | Output 2 (digital: TTL) |
| 7 | Output 3 (digital: TTL) |
| 8 | Output 4 (digital: TTL) |
| 9 | Vcc (+5 V) |
| Shield | GND |

## 13.3.4 C-170.IO Cable for Connecting to the I/O Socket

### Mini-DIN connector, 9-pin, male, open end

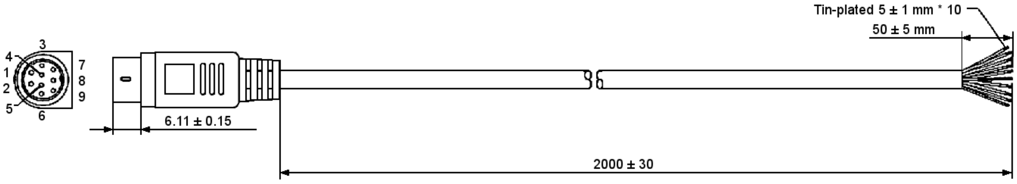



Figure 20: C-170.IO cable

| Pin | Wire Color | Function on the I/O socket of the C-867 |
|---|---|---|
| 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) |

| Pin | Wire Color | Function on the I/O socket of the C-867 |
|---|---|---|
| 5 | Purple | Output 1 (digital, TTL) |
| 6 | Blue | Output 2 (digital, TTL) |
| 7 | Green | Output 3 (digital, TTL) |
| 8 | Brown | Output 4 (digital, TTL) |
| 9 | Gray | Vcc (+5V) |
| Sheath | Shield, coated black (thicker than the wire connected to pin 1) | GND |

## 13.3.5 Analog Joystick

**Mini-DIN socket, 6-pole, female (PS/2)**

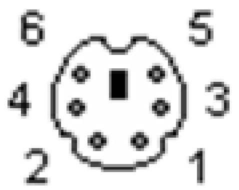



Figure 21: Front view of Mini-DIN socket

| Pin | Function |
|---|---|
| 1 | GND |
| 2 | Input: HID 1 axis 2 (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) |

## 13.3.6 Analog In

Analog input socket for TRS jack plugs

A digital HID (joystick) can be connected to this socket. Use a connector with the signal assignment shown here:

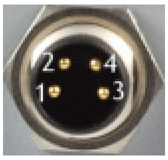

Signal specification:
±10 V
Max. 200 Hz
8 Bit

The axes of an HID connected to the **Analog In** socket are identified by the C-867 as follows:

- HID: **1**
  - X axis: **3**
  - Y axis: **4**

## 13.3.7 Power Supply Connector 24 V DC

### Phoenix M8 panel plug, 4-pole, male




| Pin | Function |
|---|---|
| 1 | GND (power) |
| 2 | GND (power) |
| 3 | Input: 24 V DC |
| 4 | Input: 24 V DC |

# 14 Old Equipment Disposal

In accordance with EU law, electrical and electronic equipment may not be disposed of in EU member states via the municipal residual waste.

Dispose of your old device according to international, national, and local rules and regulations.

To fulfill the responsibility as the product manufacturer, Physik Instrumente (PI) SE & Co. KG undertakes environmentally correct disposal of all old PI equipment made available on the market after 13 August 2005 without charge.

If you have an old device from PI, you can send it to the following address free of charge:

Physik Instrumente (PI) SE & Co. KG

Auf der Roemerstrasse 1

76228 Karlsruhe, Germany