

MS245E C-886 Parallel-Kinematic Controller User Manual

Version: 1.0.0

Date: 24.11.2017



This document describes the following products:

- **C-886.1**
Controller for parallel-kinematic positioners, 6 axes, for DC motors, TCP/IP, USB, 2 optional single axes
- **C-886.2**
Controller for parallel-kinematic positioners, 6 axes, for 2-phase stepper motors, TCP/IP, USB, 2 optional single axes
- **C-886.31**
Controller for parallel-kinematic positioners, 6 axes, for Q-Motion® piezo inertia drives, TCP/IP, USB, 2 optional single axes



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

PI®, NanoCube®, PICMA®, PLine®, NEXLINE®, PiezoWalk®, NEXACT®, Picoactuator®, PInano®, PIMag®, Q-Motion®

Notes on Third-Party Brand Names and Trademarks:

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

The following designations are registered trademarks, brands or registered trademarks of their respective owners:

Linux, LabVIEW, MATLAB, National Instruments, and MathWorks, EtherCAT

The patents owned by PI can be found in our patent list (<http://www.physikinstrumente.com/en/about-pi/patents>).

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

© 2017 Physik Instrumente (PI) GmbH & Co. KG, Karlsruhe, Germany. The text, photographs, and drawings in this manual are protected by copyright. With regard thereto, Physik Instrumente (PI) GmbH & Co. KG retains all the rights. The use of any text, images and drawings is permitted only in part and only when indicating the source.

Original instructions

First printing: 24.11.2017

Document number: MS245E, BRo, Version 1.0.0

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 This manual is superseded by any new release.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

Contents

1	About this Document	1
1.1	Objective and Target Audience of this User Manual.....	1
1.2	Symbols and Typographic Conventions.....	1
1.3	Definition of Terms.....	2
1.4	Other Applicable Documents	6
1.5	Downloading Manuals.....	7
2	Safety	9
2.1	Intended Use	9
2.2	General Safety Instructions	9
2.2.1	Organizational Measures.....	10
2.2.2	Safety Measures during Installation.....	10
2.2.3	Measures during Startup and Operation.....	10
2.2.4	Safety Measures during Maintenance.....	12
3	Product Description	13
3.1	Features and Applications	13
3.1.1	Overview of Functions.....	13
3.1.2	Master and Slave Modules	14
3.2	Model Overview	15
3.3	Product View	16
3.3.1	Front Panel	16
3.3.2	Rear Panel.....	20
3.4	Type Plate	21
3.5	Scope of Delivery.....	21
3.6	Accessories	22
3.7	Commandable Items	23
3.7.1	Identifier for Normal Operation of the C-886	23
3.7.2	Identifier for Direct Access to the Slave Modules	24
3.8	Important Components of the Firmware	26
3.9	Motion of the Positioner	27
3.9.1	Supported Motion Types.....	27
3.9.2	Profile Generator for Point-to-Point Motion.....	28
3.9.3	Cyclic Transfer of Target Positions	30
3.9.4	Coordinate Systems.....	32
3.9.5	Rotations.....	33
3.10	Communication Interfaces	34
3.11	Overview of PC Software.....	34

4	Unpacking	37
5	Installation	39
5.1	Installing the PC Software	39
5.1.1	Performing Initial Installation	39
5.1.2	Installing Updates	40
5.2	Ensuring Ventilation	42
5.3	Connecting the C-886 to the Protective Earth Conductor	42
5.4	Installing Slave Modules in the C-886	43
5.5	Connecting the Power Supply to the C-886	44
5.6	Connecting Drives	45
5.7	Connecting the PC	46
5.7.1	Connecting the C-886 via the TCP/IP Interface	46
5.7.2	Connecting the C-886 via the USB interface	46
6	Startup	49
6.1	General Notes on Startup	49
6.2	Switching on the C-886	52
6.3	Establishing Communication via the TCP/IP Interface	53
6.3.1	Preparing the PC and C-886 for Using Static IP Addresses	54
6.3.2	Establishing Communication via TCP/IP in the PC Software	57
6.4	Establishing Communication via the USB Interface	60
6.5	Configuring Slave Modules for Single Axes	62
6.6	Starting Motion	66
7	Operation	71
7.1	General Notes on Operation	71
7.2	Data Recorder	72
7.2.1	Data Recorder Properties	72
7.2.2	Setting up the Data Recorder	73
7.2.3	Starting the Recording	74
7.2.4	Reading Out Recorded Data	74
7.3	Wave Generator	75
7.3.1	Functionality of the Wave Generator	75
7.3.2	Commands and Parameters for the Wave Generator	76
7.3.3	Defining the Waveform	78
7.3.4	Configuring a Wave Generator	86
7.3.5	Starting and Stopping Output	88
7.3.6	Application Tips: Loading Customer-Specific Data	91
7.3.7	Application Tips: Using Macros for the Wave Generator	95
7.4	Controller Macros	97
7.4.1	Overview: Macro Functionalities and Example Macros	97
7.4.2	Commands and Parameters for Macros	98
7.4.3	Working with Macros	99

7.4.4	Variables	106
8	GCS Commands	109
8.1	Notation.....	109
8.2	GCS Syntax for Syntax Version 2.0	109
8.3	Command Overview	111
8.4	Command Descriptions for GCS 2.0	117
8.5	Error Codes.....	218
9	Adapting Settings	233
9.1	Overview of the Settings of the C-886	233
9.2	Changing Parameter Values of the C-886 Master Module	234
9.3	Saving Parameter Values in a Text File.....	235
9.4	Parameter Overview.....	236
10	Maintenance	241
10.1	Cleaning the C-886	241
10.2	Updating Firmware.....	242
11	Troubleshooting	247
12	Customer Service	251
13	Technical Data	253
13.1	Specifications.....	253
13.1.1	Data Table.....	253
13.1.2	Maximum Ratings.....	256
13.1.3	Ambient Conditions and Classifications	257
13.2	System Requirements.....	257
13.3	Dimensions	258
13.4	Pin Assignment	259
13.4.1	Axis 1, Axis 2 (C-886.1 only).....	259
13.4.2	Motor (C-886.2 only)	260
13.4.3	Motor & Sensor (C-886.31 only).....	261
14	Old Equipment Disposal	263
15	EU Declaration of Conformity	265

1 About this Document

In this Chapter

Objective and Target Audience of this User Manual	1
Symbols and Typographic Conventions	1
Definition of Terms	2
Other Applicable Documents	6
Downloading Manuals	7

1.1 Objective and Target Audience of this User Manual

This manual contains information necessary for the intended use of the C-886.

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

The latest versions of the user manuals are available for download (p. 7) on our website.

1.2 Symbols and Typographic Conventions

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

CAUTION



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

- Actions to take to avoid the risk.



NOTICE




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

- Actions to take to avoid the situation.

INFORMATION

Information for easier handling, tricks, tips, etc.

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

1.3 Definition of Terms

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. A reference point definition is not necessary.
Axis	Also referred to as the "logical axis". Logical axes reproduce the translations and rotations of the motion platform of the positioner and the motion of the optionally usable stages in the firmware of the C-886. Each direction of motion corresponds to a logical axis. Further information on translations and rotations can be found in the manual of the positioner. All motion commands of the C-886 refer to logical axes.

Orientational coordinate system	The orientational coordinate system can be used to change the direction of the translation axes X and/or Y and/or Z permanently (e.g., when Z is to always point in the direction of the default X axis). In the default setting, the orientational coordinate system PI_Base is enabled.
Workspace	<p>The entirety of all combinations of translations and rotations that the positioner can approach from the current position is referred to as the workspace.</p> <p>The workspace can be limited by the following external factors:</p> <ul style="list-style-type: none"> ▪ Installation space ▪ Dimensions and position of the load
Operating coordinate system	With the operating coordinate system, the position display, direction of motion, and the center of rotation for the motion platform of the positioner is adapted to the application. It is also possible to use --> work-and-tool coordinate systems. In the default setting, the operating coordinate system ZERO is active.
Center of rotation	<p>The center of rotation describes the intersection of the rotational axes U, V, and W.</p> <p>The center of rotation always moves together with the platform.</p> <p>Depending on the active --> operating coordinate system, the center of rotation can be moved from the origin of the coordinate system in the X and/or Y and/or Z direction with the <code>SPI</code> command. The center of rotation that can be moved using the <code>SPI</code> command is also referred to as "pivot point".</p>
Dynamics profile	Comprises the target position, velocity, and acceleration of the axis calculated for each point in time of the motion. The calculated values are called "commanded values". The dynamics profile can be created by the profile generator of the C-886 or by the wave generators, or it can be externally created and transferred to the C-886 through the cyclic transfer of target positions.
Firmware	Software that is installed on the controller.
Volatile memory	<p>RAM module in which the parameters are saved when the controller is switched on (working memory). The parameter values in the volatile memory determine the current behavior of the system.</p> <p>In the PC software of PI, the parameter values in the volatile memory are also referred to as "Active Values".</p>
GCS	PI General Command Set; command set for PI controllers. Piezo drivers and servo controllers can be operated conjointly with minimal programming effort thanks to the GCS.
Positioner system	The combination of positioner, controller, cable set, and power supply is referred to as "positioner system" in this manual.

Incremental position sensor	<p>Sensor (encoder) for capturing 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, a reference point definition must be performed before absolute target positions can be commanded and reached.</p>
Coordinate system	<p>The position display, direction of motion, and center of rotation for the motion platform of the positioner are determined by coordinate systems that are linked in a chain. The chain is basically structured as follows (starting point > end point): --> HEXAPOD coordinate system , --> leveling coordinate system, --> orientational coordinate system, --> operating coordinate system.</p> <p>The coordinate systems are always right-handed systems.</p> <p>The HEXAPOD coordinate system determines the fundamental properties of all other coordinate systems. HEXAPOD is based on the configuration file with the geometrical data of the positioner. The dimensional drawing in the manual of the positioner shows the respective position of the HEXAPOD coordinate system.</p> <p>Using the controller, custom coordinate systems can be defined and used instead of the default coordinate systems.</p> <p>Based on HEXAPOD, the orientational and leveling coordinate systems adapt fundamental properties of the enabled operating coordinate system, and in most applications their custom definition and enabling is not necessary at all or only once.</p>
Leveling coordinate system	<p>The leveling coordinate system can be used to correct errors in the orientation of the positioner permanently (e.g., installation errors). In the default setting, the leveling coordinate system PI_Levelling is active.</p>
Master module	<p>The master module of the C-886 carries out central tasks:</p> <ul style="list-style-type: none"> ▪ Parallel-kinematic control: Conversion among the Cartesian coordinates of the motion platform and the positions of the individual drives ▪ Interface for communication with the PC ▪ Internal communication with the slave modules for controlling the drives of the parallel-kinematic positioner and the optional single axes ▪ Provision of a data recorder, wave generators, and macro functionality ▪ Provision of user-defined coordinate systems for the parallel-kinematic positioner

Parallel-kinematic positioner	<p>The C-886 is intended for operating a parallel-kinematic positioner, which is also referred to as a "positioner" for short in this manual.</p> <p>A parallel-kinematic positioner can be a hexapod or an assembly that is equipped with passive struts that are each moved by a combination of two linear drives. In order to move a logical axis of the platform of the positioner, several linear drives must be moved.</p> <p>The C-886 calculates the target positions for the individual linear drives from the target positions given for the translational and rotational axes of the positioner. The velocities and accelerations of the linear drives are calculated in such a way that all drives start and stop at the same time.</p>
PC software	Software that is installed on the PC.
Nonvolatile memory	<p>EEPROM memory chip (read-only memory), from which the default values of the parameters are loaded to the volatile memory when the controller is started.</p> <p>In the PC software of PI, the parameter values in the nonvolatile memory are also referred to as "Startup Values".</p>
Pivot point	Depending on the active --> operating coordinate system, the center of rotation can be moved from the origin of the coordinate system in the X and/or Y and/or Z direction with the <code>SPI</code> command. The center of rotation that can be moved using the <code>SPI</code> command is also referred to as "pivot point".
Slave module	<p>The slave modules of the C-886 (also "drive modules") are the interfaces to the drives of the parallel-kinematic positioner and the optional single axes. The slave modules receive the control commands for the individual drives from the master module.</p> <ul style="list-style-type: none"> ▪ Slave modules for the parallel-kinematic positioner: On delivery of the C-886, the modules are configured for the positioner used. The modules may be replaced and the settings of the modules directly accessed only after consultation with PI. ▪ Slave modules for the two optional single axes: The modules are available as optional accessories. The modules can be installed and configured by the user.

Work-and-tool coordinate systems

The work-and-tool concept can be used for work with user-defined coordinate systems.

The work-and-tool concept uses a combination of two enabled --> operating coordinate systems ("work coordinate system" and "tool coordinate system"). The X, Y, and Z axes of the tool coordinate system are always permanently connected to the motion platform of the positioner; i.e., the tool coordinate system moves together with the platform. The X, Y, and Z axes of the work coordinate system are always spatially fixed; i.e., the work coordinate system does **not** move when the platform of the positioner moves.

The current position of the motion platform of the positioner can be considered the position of the tool coordinate system in the work coordinate system.

The center of rotation always lies at the origin of the tool coordinate system and it therefore moves just as the tool coordinate system with the platform.

1.4 Other Applicable Documents

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

Documentation for the C-886 and the supplied PC software:

Description	Document
Coordinate Systems for Hexapod Microrobots	C887T0007 Technical Note
Motion of the Hexapod Position and Orientation in Space, Center of Rotation	C887T0021 Technical Note
GCS LabVIEW Driver Library for the C-886	MS246E Software Manual
LabVIEW Merge Tool	SM154E Software Manual
PI GCS 2.0 DLL	SM151E Software Manual
PI MATLAB Driver GCS 2.0	SM155E Software Manual
PIPython	SM157E User Manual
GCS array data format description	SM146E Software Manual
PIMikroMove	SM148E Software Manual
PI Frequency Generator Tool Available in PIMikroMove for controllers with wave generator	A000T0057 Technical Note

Description	Document
PIStages3Editor	SM156E User Manual
PI Update Finder: Search and download updates	A000T0028 Technical Note
PI Update Finder: Updating PC without Internet connection	A000T0032 Technical Note

User manual of the parallel-kinematic positioner

User manuals of the optional single axes

When slave modules are used for optional single axes:

Description	Document
C-663.12C885 Controller Module	C663T0004 User Manual, MS241E User Manual
C-863.20C885 Controller Module	C863T0005 User Manual, MS205E User Manual
E-873.10C885 Controller Module	E873T0002 User Manual, PZ273E User Manual

1.5 Downloading Manuals

INFORMATION

If a manual is missing or problems occur with downloading:

- Contact our customer service department (p. 251).

INFORMATION

For products that are supplied with software (CD in the scope of delivery), access to the manuals is protected by a password. Protected manuals are only displayed on the website after entering the password.

The password is included on the CD of the product.

For products with CD: Identify the password

1. Insert the product CD into the PC drive.
2. Switch to the Manuals directory on the CD.

3. In the Manuals directory, open the Release News (file including **releasenews** in the file name).
4. Find the user name and the password in the section "User login for software download" in the Release News.

Downloading manuals

1. Open the website **www.pi.ws**.
2. If access to the manuals is protected by a password:
 - a) Click **Login**.
 - b) Log in with the user name and password.
3. Click **Search**.
4. Enter the product number up to the period (e.g., P-882) or the product family (e.g., PICMA® Bender) into the search field.
5. Click **Start search** or press the **Enter** key.
6. Open the corresponding product detail page in the list of search results:
 - a) If necessary: Scroll down the list.
 - b) If necessary: Click **Load more results** at the end of the list.
 - c) Click the corresponding product in the list.
7. Scroll down to the **Downloads** section on the product detail page.

The manuals are displayed under **Documentation**.
8. Click the desired manual and save it to the hard disk of your PC or to a data storage medium.

2 Safety

In this Chapter

Intended Use.....	9
General Safety Instructions.....	9

2.1 Intended Use

The C-886 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-886 is intended for the closed-loop operation of a parallel-kinematic positioner from PI. Only positioners with a suitable drive type according to specifications (p. 253) may be connected to the C-886.

The C-886 must not be used for purposes other than those stated in this user manual.

The C-886 may only be used in compliance with the technical specifications and instructions in this user manual. The operator is responsible for process validation.

2.2 General Safety Instructions

The C-886 is built according to state-of-the-art technology and recognized safety standards. Improper use can result in personal injury and/or damage to the C-886.

- Only use the C-886 for its intended purpose, and only use it if it is in a good working order.
- Read the user manual.
- Immediately eliminate any faults and malfunctions that are likely to affect safety.

The operator is responsible for the correct installation and operation of the C-886.

- Install the C-886 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Use the supplied components (power supply, adapter and power cord (p. 22)) to connect the C-886 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.

2.2.1 Organizational Measures

User manual

- Always keep this user manual available with the C-886. The latest versions of the user manuals are available for download (p. 7) on our website.
- Add all information from the manufacturer to the user manual, for example supplements or technical notes.
- If you give the C-886 to other users, also include this user manual as well as other relevant information provided by the manufacturer.
- Only use the device on the basis of the complete user manual. Missing information due to an incomplete user manual can result in minor injury and damage to equipment.
- Only install and operate the C-886 after you have read and understood this user manual.

Personnel qualification

The C-886 may only be installed, started up, operated, maintained, and cleaned by authorized and appropriately qualified personnel.

2.2.2 Safety Measures during Installation

Impermissible mechanical load and collisions between the positioner, the load to be moved, and the environment can damage the positioner.

- Observe the safety information and instructions in the manual of the positioner.
- Do **not** exceed the forces and torques that are specified for the positioner; see the manual of the positioner.

Connecting an unsuitable drive type can cause damage to the drive or the C-886.

- Only connect a suitable drive to the slave modules of the C-886.

2.2.3 Measures during Startup and Operation

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the C-886 in the case of malfunction or failure of the system. If touch voltages exist, touching the C-886 can result in minor injuries from electric shock.

- Connect the C-886 to a protective earth conductor (p. 42) before start-up.
- 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-886 to the protective earth conductor before starting it up again.

There is a risk of minor injuries from crushing between the moving parts of the positioner and a stationary part or obstacle.

- Keep your fingers away from areas where they can get caught by moving parts.

When the communication between the C-886 and the PC is established via TCP/IP, the PC software offers all controllers present in the same network for selection. After a C-886 has been selected for the connection, all commands are sent to this controller. If the wrong controller is selected, risk of minor injury risk to the operating and maintenance staff from crushing due to unexpectedly commanded motion of the positioner.

- If several C-886 are displayed in the PC software, make sure that you select the right C-886.

The configuration of the C-886 must be adapted to the positioner. If an incorrect configuration is used, the positioner can be damaged by uncontrolled motion or collisions. The C-886 is configured for the used positioner on delivery.

- Check the configuration: Once you have established communication via TCP/IP (p. 53) or USB (p. 60), send the `CST?` command. The response shows the positioner to which the C-886 is adapted.
- Operate the positioner only with a C-886 whose configuration data is adapted to the positioner.
- Change the settings of the slave modules for the drives of the positioner only after consultation with PI.

Collisions can damage the positioner, the load to be moved, and the surroundings.

General measures for avoiding collisions:

- Make sure that no collisions are possible between the positioner, the load to be moved, and the surroundings in the workspace of the positioner.
- Do not place any objects in areas where they can be caught by moving parts.
- Stop the motion immediately if a controller malfunction occurs.
- Note that the positioner moves unpredictably during a reference move.

Depending on the source of the dynamics profile, the platform of the positioner can move along an undefined path under certain conditions. As a result, collisions are possible between the positioner, the load to be moved, and the surroundings.

When the dynamics profile is determined by the profile generator of the C-886 (default):

- Avoid sending new target positions while the positioner (axes X, Y, Z, U, V, W) is still moving.
- If new target positions have to be sent while the positioner is still moving (axes X, Y, Z, U, V, W): Only use motion commands to set target positions that maximally deviate from the current position by the value of the **Path Control Step Size** parameter (ID 0x19001504).

When the dynamics profile is determined by consecutive **MOV** commands:

- Only set target positions whose distance from each other is maximally as large as the value of the **Path Control Step Size** parameter (ID 0x19001504) with consecutive **MOV** commands.

When the actual load of the motion platform of the positioner exceeds the maximum holding force based on the self-locking of the linear drives, switching off the servo mode can cause unintentional position changes of the positioner. As a result, collisions are possible between the positioner, the load to be moved, and the surroundings.

- Make sure that the actual load of the motion platform of the positioner does not exceed the maximum holding force based on the self-locking of the linear drives before you switch off the servo mode, reboot or switch off the C-886.
- Ensure an uninterruptible power supply in order to prevent an unintentional deactivation of the positioner system and resulting unintentional position changes of the positioner.

Unsuitable parameter settings can lead to improper operation or damage to the connected mechanical system.

- Only change parameters after careful consideration.

2.2.4 Safety Measures during Maintenance

The C-886 contains electrostatic-sensitive devices (ESD) that can be damaged in the case of short circuits or flashovers.

- Before cleaning the housing, disconnect the C-886 from the power supply by pulling the power plug.

3 Product Description

In this Chapter

Features and Applications	13
Model Overview.....	15
Product View.....	16
Type Plate	21
Scope of Delivery	21
Accessories.....	22
Commandable Items.....	23
Important Components of the Firmware	26
Motion of the positioner	27
Communication Interfaces.....	34
Overview of PC Software	34

3.1 Features and Applications

3.1.1 Overview of Functions

The C-886 controller is intended for controlling a parallel-kinematic positioner in six degrees of freedom. The motion commands use Cartesian coordinates. The C-886 converts these into the respective positions and velocities of the individual drives before the platform moves to the desired position.

- TCP/IP and USB interface for communication
- When the corresponding slave modules are installed in the C-886: Connections for up to two optional single axes with freely selectable drive type
- User-defined coordinate systems possible, including work-and-tool coordinate systems
- Nonvolatile macro memory
- Data recorder
- Wave generator
- Extensive software package available

3.1.2 Master and Slave Modules

To support the wide range of drive concepts offered by PI, the C-886 has a modular structure. Each C-886 consists of a master module and several slave modules.

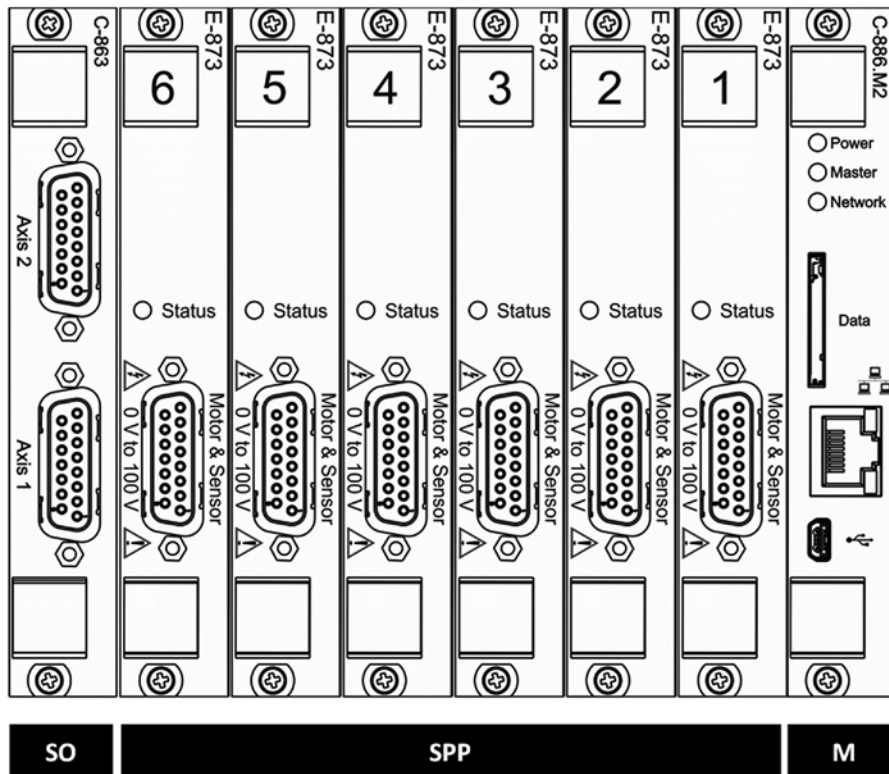


Figure 1: Example: C-886.31 with C-863.20C885 module for optional single axes

SO Slave module for optional single axes

SPP Slave modules for the parallel-kinematic positioner

M Master module

The master module of the C-886 carries out central tasks:

- Parallel-kinematic calculation: Conversion between the Cartesian coordinates of the motion platform and the positions of the individual drives
- Interface for communication with the PC
- Internal communication with the slave modules for controlling the drives of the parallel-kinematic positioner and the optional single axes
- Provision of a data recorder, wave generators, and macro functionality
- Provision of user-defined coordinate systems for the parallel-kinematic positioner

The slave modules of the C-886 (also "drive modules") are the interfaces to the drives of the parallel-kinematic positioner and the optional single axes. The slave modules receive the commands for controlling the individual drives from the master module.

- Slave modules for the parallel-kinematic positioner
The modules are configured for the positioner used before delivery of the C-886. The modules may be replaced and the settings of the modules accessed only after consultation with PI.
- Slave modules for the optional single axes:
The modules are available as optional accessories (p. 22). The modules can be installed and configured by the user. The entries of the PIStages3 stage database can be used to configure the modules.

3.2 Model Overview

The C-886 controller is available in the following versions:

Model	Designation
C-886.1	Controller for parallel-kinematic positioners, 6 axes, for DC motors, TCP/IP, USB, 2 optional single axes
C-886.2	Controller for parallel-kinematic positioners, 6 axes, for 2-phase stepper motors, TCP/IP, USB, 2 optional single axes
C-886.31	Controller for parallel-kinematic positioners, 6 axes, for Q-Motion® piezo inertia drives, TCP/IP, USB, 2 optional single axes

3.3 Product View

3.3.1 Front Panel

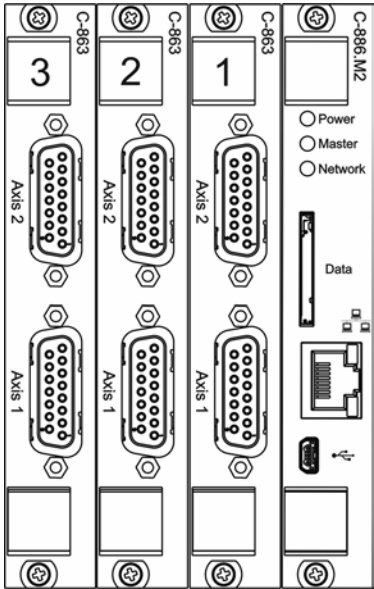


Figure 2: C-886.1 front panel, without slave modules for optional single axes

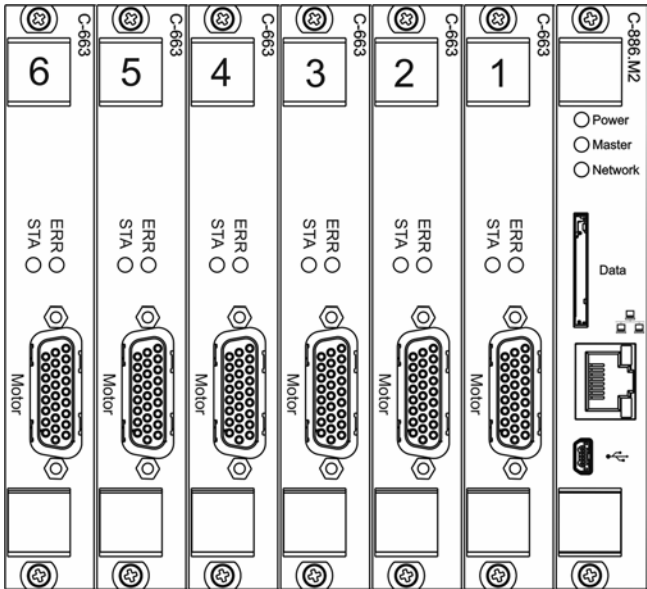


Figure 3: C-886.2 front panel, without slave modules for optional single axes

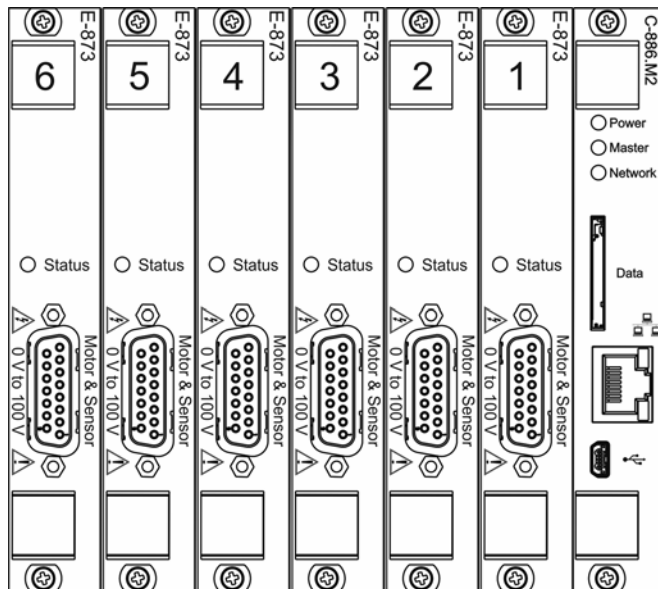





Figure 4: C-886.31 front panel, without slave modules for optional single axes

Master module:

Labeling	Type	Function
Power	LED green/off	Power: <ul style="list-style-type: none"> On: The C-886 is ready for normal operation. Off: The C-886 is booting. / The C-886 is not connected to the power supply.
Master	LED red/off	Error indicator for the master module: <ul style="list-style-type: none"> On: Error (error code $\neq 0$) Off: No error (error code = 0). / The C-886 is not connected to the power supply. <p>The error code can be queried with the <code>ERR?</code> command. The query resets the error code to zero and the LED is switched off.</p>


Labeling	Type	Function
Network	LED red/green/off	Status display for slave modules: <ul style="list-style-type: none"> Off: No error is present (error code = 0). / The C-886 is booting. / The C-886 is not connected to the power supply. Green: All slave modules are ready for normal operation. Red: Error in the internal communication or in at least one of the slave modules (error code \neq 0). If the affected slave module is equipped with an LED for error indication, this LED will also show the error.
Data	Slot for SD card	For future use; currently no function.
	RJ45 socket	Network connection via TCP/IP
	Mini-USB type B	Universal serial bus for connection to the PC

Only C-886.1: Slave modules for the parallel-kinematic positioner:




Labeling	Type	Function
Axis 1 Axis 2	Sub-D 15 (f) 	Connections for the drives of the parallel-kinematic positioner. Only for DC motors!

Only C-886.2: Slave modules for the parallel-kinematic positioner:

Labeling	Type	Function
STA	LED green/off	Status of the slave module: <ul style="list-style-type: none"> Continuous illumination: The module is ready for normal operation Flashing: The module is in firmware update mode Off: The module is not connected to the supply voltage

Labeling	Type	Function
ERR	LED red/off	Error indicator for slave module: <ul style="list-style-type: none"> ▪ On: Error. The error is also displayed by the Network LED of the master module. ▪ Off: No error
Motor	Sub-D 26 (f) 	Connection for a drive of the parallel-kinematic positioner. Only for 2-phase stepper motors!

Only C-886.31: Slave modules for the parallel-kinematic positioner:

Labeling	Type	Function
Status	LED red/green/off	Status of the slave module: <ul style="list-style-type: none"> ▪ Red: Error. The error is also displayed by the Network LED of the master module. ▪ Continuous green: Module is ready for normal operation ▪ Flashing green: The module is in firmware update mode ▪ Off: The module is not connected to the supply voltage
Motor & Sensor  0 V to 100 V 	Sub-D 15 (f) 	Connections for the drives of the parallel-kinematic positioner. Only for Q-Motion® piezo inertia drives!

3.3.2 Rear Panel

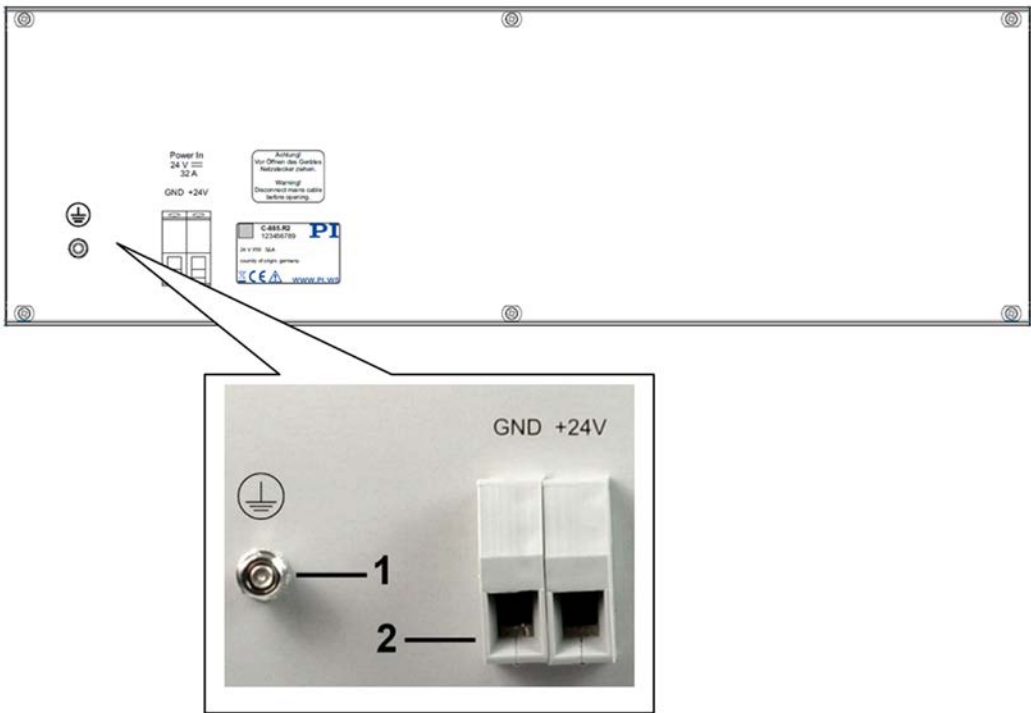






Figure 5: C-886 rear panel


- 1 Protective earth connection
- 2 Connection for the supply voltage

Labeling	Type	Function
	Threaded bolt with fastening material for protective earth conductor	Protective earth connection (p. 42) The threaded bolt must be connected to a protective earth conductor, because the C-886 is not grounded via the power supply connector.
GND +24V	Feedthrough terminals with screw connection, cross section 0.5 mm ² to 16 mm ²	Connection for the supply voltage Pin assignment according to labeling Maximum current consumption: 32 A

3.4 Type Plate

Labeling	Function
	Data matrix code (example; contains the serial number)
C-886.31	Product name (example), the characters following the period refer to the model
PI	Manufacturer's logo
117567891	Serial number (example), individual for each C-886 Meaning of the places (counting from left): 1 = internal information, 2 and 3 = year of manufacture, 4 to 9 = consecutive numbers
Country of origin: Germany	Country of origin
	Warning sign "Observe manual!"
	Old equipment disposal (p. 263)
CE	CE conformity mark
WWW.PI.WS	Manufacturer's address (website)

3.5 Scope of Delivery

Item ID	Components
C-886	Controller according to your order
000036360	USB cable (type A to Mini-B) for connection to the PC
C-885.PS	Separate 24 V wide-range-input power supply (85 W/3.54 A) for use with line voltages from 100 to 240 VAC and voltage frequencies of 50 or 60 Hz Connection to the C-886: Molex Mini Fit 6-pin (f)
3763	Power cord
000058055	Adapter for the power supply connection; Molex Mini Fit 6-pin (m) to ferrules 4 mm ² 
C-886.CD	CD with PC software and documentation
MS245E	User manual for the C-886

3.6 Accessories

The following modules can be used as slave modules for optional single axes in the C-886:

Order number	Description
C-863.20C885	Motion controller module for DC motors, 2 axes, for PIMotionMaster, PID controller
C-663.12C885	Mercury Step stepper motor controller module, 1 axis, HD Sub-D 26, for PIMotionMaster, 1 axis, closed-loop and open-loop operation, support of external sensors
E-873.10C885	Q-Motion® controller module for PIMotionMaster, 1 axis, for systems with piezoelectric inertia drive

After slave modules are installed for optional single axes, it can be necessary to cover free slots.
Available cover plates:

Order number	Description
C-885.AP1	Cover plate for PIMotionMaster, 3 RU, 4 HP
C-885.AP2	Cover plate for PIMotionMaster, 3 RU, 8 HP
C-885.AP4	Cover plate for PIMotionMaster, 3 RU, 16 HP
C-885.AP8	Cover plate for PIMotionMaster, 3 RU, 32 HP

- To order, contact our customer service department (p. 251).

3.7 Commandable Items

3.7.1 Identifier for Normal Operation of the C-886

The following table contains the items that can be commanded or read out with commands of the GCS (p. 109).

Item	Number	Identifier	Description																														
Logical axis	6	X, Y, Z, U, V, W	The logical axes X to W reproduce the translations and rotations of the motion platform of the parallel-kinematic positioner in the firmware of the C-886. Translational axes: X, Y, and Z Rotational axes: U, V, and W																														
Logical axis	2	S1, S2	The logical axes S1 and S2 reproduce the motion of the optional single axes in the firmware of the C-886.																														
Wave generator	8	1 to 8	Each wave generator (p. 75) is permanently allocated to a logical axis: <table><tr><th>Mechanics</th><th colspan="6">Parallel-kinematic positioner</th></tr><tr><th>Axis</th><td>X</td><td>Y</td><td>Z</td><td>U</td><td>V</td><td>W</td></tr><tr><th>Wave generator</th><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table> <table><tr><th>Mechanics</th><th colspan="2">Single axes</th></tr><tr><th>Axis</th><td>S1</td><td>S2</td></tr><tr><th>Wave generator</th><td>7</td><td>8</td></tr></table>	Mechanics	Parallel-kinematic positioner						Axis	X	Y	Z	U	V	W	Wave generator	1	2	3	4	5	6	Mechanics	Single axes		Axis	S1	S2	Wave generator	7	8
Mechanics	Parallel-kinematic positioner																																
Axis	X	Y	Z	U	V	W																											
Wave generator	1	2	3	4	5	6																											
Mechanics	Single axes																																
Axis	S1	S2																															
Wave generator	7	8																															
Wave table	100	1 to 100	The wave tables contain the saved data (a total of 10,000 points) for the waveforms that are output by the wave generators. The value of the Number Of Waves parameter (ID 0x1300010A) indicates the number of wave tables (p. 75).																														

Item	Number	Identifier	Description
Coordinate system	unlimited	Name is assigned when the coordinate system is defined	<p>Using the controller, custom coordinate systems can be defined and used instead of the default coordinate systems.</p> <p>For more information, see "Motion of the positioner" (p. 27).</p> <p>Naming conventions for coordinate systems:</p> <p>Permissible characters: 1234567890ABCDEFGHIJKLMNQRSTUWXYZ_ The number of characters is unlimited.</p> <p>The name must start with an alphabetic character.</p> <p>Reserved names which must not be used for defining, copying or deleting: HEXAPOD, PI_LEVelling, PI_BASE, ZERO, 0, NULL, XML, KLF, KLF(USER), KLF(PI), KLD, KLD(USER), KLD(PI), KSB, KSB(USER), KSB(PI), KSD, KSF, KST, KSW</p> <p>Each name can exist only once. Any existing coordinate system not in use will be overwritten when a coordinate system with the same name is created (defining, generating a copy).</p>
Data recorder table	36	1 to 36	<p>The data recorder tables contain the recorded data. The C-886 has 36 data recorder tables (query with <code>TNR?</code> (p. 195)) with max. 8192 data points per table.</p> <p>The number of points per data recorder table can be set with the Data Recorder Points Per Table parameter (ID 0x16000201).</p>
System	1	1	C-886 as an overall system.

3.7.2 Identifier for Direct Access to the Slave Modules

The identifiers in the following table are only relevant when direct access to slave modules or individual drives is necessary. Typical use case: The slave modules must be configured for the optional single axes, see "Configuring Slave Modules for Single Axes" (p. 62) for details.

The slave modules for the parallel-kinematic positioner are configured ready to use on delivery of the C-886. Changing settings or commanding motion by directly accessing the slave modules can lead to damage to the positioner.

- Do not directly access the slave modules for the drives of the parallel-kinematic positioner without prior consultation with PI.

Basically, the identifiers of the slave module to be addressed and the drive to be addressed are required in each command line for direct access to the slave modules. For further information, see documentation of the slave modules (p. 6), in particular "Target and Sender Address".

Note: The designations in this manual deviate from the designations in the documentation of the slave modules:

- Slave module = controller to be addressed (target)
- Drive supported = axis

INFORMATION

When the slave modules are accessed directly, commands are sent to the master module of the C-886 without module identifier or with module identifier 1.

If you access the slave modules directly with the PIMikroMove PC software: PIMikroMove takes care of addressing the slave modules and the module identifier in the **Command entry** window must be omitted.

Counting of the slave modules begins with the module that is in the housing of the C-886 immediately next to the master module.

Item	Number	Identifier	Description
Slave modules for the parallel-kinematic positioner	3 / 6	1 to 3 / 6 or 2 to 4 / 7	<p>The number of slave modules for the parallel-kinematic positioner depends on how many drives per module are supported:</p> <ul style="list-style-type: none"> ▪ Three slave modules when two drives per module are supported ▪ Six slave modules when one drive per module is supported <p>The counting method of the identifiers is different:</p> <ul style="list-style-type: none"> ▪ 1 to 3 / 6 in the response to the <code>SPA?</code> command ▪ 2 to 4 / 7 in the response to the <code>VER?</code> command
Slave modules for optional single axes	maximum of 2	5, 6 or 8, 9	<p>The identifiers of the slave modules for optional single axes depend on the number of slave modules for the parallel-kinematic positioner:</p> <ul style="list-style-type: none"> ▪ If there are three modules for the positioner: 5, 6 ▪ If there are six modules for the positioner: 8, 9 <p>The maximum number of slave modules for optional single axes depends on how many drives per module are supported. Options:</p> <ul style="list-style-type: none"> ▪ One slave module that supports two drives ▪ Two slave modules that support one drive respectively
Drives per slave module	maximum of 2	1, 2	<p>The number of drives per slave module supported depends on the module type: 1 or 2.</p> <p>The drives are referred to as "axes" in the documentation of the slave modules.</p>

3.8 Important Components of the Firmware

The firmware of the C-886 provides the following functional units:

Firmware component	Description
ASCII commands	<p>Communication with the C-886 can be managed using the commands of the PI General Command Set (GCS; version 2.0). The GCS is independent of the hardware (controller, stages connected).</p> <p>Examples of the use of GCS:</p> <ul style="list-style-type: none"> ▪ Starting motion of the positioner ▪ Getting system and motion values ▪ Defining a coordinate system <p>You can find a list of the available commands in the "Command Overview" section (p. 111).</p>
Configuration parameters	<ul style="list-style-type: none"> ▪ For interface parameters, see "Establishing Communication via the TCP/IP Interface" (p. 53) ▪ For further parameters, see "Adapting Settings" (p. 233)
Command levels	<p>The availability of commands and the write permission for the parameters are determined by command levels. The current command level can be changed with the CCL command. This may require entering a password.</p> <p>For further information, see "Adapting Settings" (p. 233).</p>
Data recorder	<p>The C-886 contains a real-time data recorder (p. 72) for position values and status register bits.</p>
Wave generator	<p>Each axis can be controlled by a wave generator that outputs waveforms. The wave generator is especially suited for dynamic applications in which periodic motion of the axis is executed (p. 75).</p>
Macros	<p>The C-886 can save macros. Command sequences can be defined and permanently stored via the macro function. A startup macro can be defined that runs each time the C-886 is switched on or rebooted. This simplifies operation without a connected PC. Further information can be found in the "Controller Macros" section (p. 97).</p>

The firmware can be updated with a tool (p. 242).

3.9 Motion of the Positioner

3.9.1 Supported Motion Types

The C-886 supports the following motion types:

Motion type	Commands for triggering motion
Reference move (p. 132)	FRF
Point-to-point motion; profile generator creates the dynamics profile (p. 28)	MOV: Motion to absolute target position The Trajectory Source parameter (ID 0x19001900) must have the value 0 (default).
	STE, IMP: Start step or impulse, with data recording MVR: Motion to relative target position MRT, MRW: Moves the given axis relatively in the tool or work coordinate system. See "Definition of Terms" (p. 2) for more details on coordinate systems
Cyclic transfer of target positions (p. 30)	Consecutive MOV commands The Trajectory Source parameter (ID 0x19001900) must have the value 1 = "Dynamics profile is determined by consecutive MOV commands".
Wave generator (p. 75)	WGO: Starts/stops the wave generator output

INFORMATION

The C-886 can save and process command sequences as controller macros (p. 97).

All commands can be sent via the communication interfaces of the C-886 when a macro is running on the C-886. The macro content and commands received via the communication interfaces can overwrite each other.

INFORMATION

For axes with incremental sensors, motion can only be commanded after a successful reference move (p. 132) (also referred to as "initialization").

The behavior of the positioner after the reference move is determined by the **Behaviour After Reference Move** parameter (ID 0x07030401) and **Target For Motion After Reference Move** parameter (ID 0x07030402). Depending on the parameter values, the platform can be moved automatically to a specified position, e.g., after the reference move.

- Value of the parameter 0x07030401 = 0: Motion platform remains in the reference position after the reference move.
- Value of the parameter 0x07030401 = 1: After the reference move, the motion platform moves to the absolute target position, which is given by parameter 0x07030402.

No reference move is required for axes with absolute-measuring sensors. The use of the FRF command is still recommended for these axes, however. FRF does **not** start a reference move for axes with absolute-measuring sensors but sets the target positions to the current position

values. The above-described parameter values also go into effect, so that the axes can be moved to a defined "initial position", for example.

The following settings apply to all motion types:

Parameter	Description and Possible Values
Maximum System Velocity (Phys. Unit/s) 0x19001500	Maximum system velocity This parameter is write-protected and is adapted to the positioner of the system before delivery. Upper limit of the velocity for the motion platform of the positioner that can be set with VLS (p. 200).
Minimum System Velocity (Phys. Unit/s) 0x19001501	Minimum system velocity This parameter is write-protected and is adapted to the positioner of the system before delivery. Depends on the smallest step size of the positioner. Lower limit of the velocity for the motion platform of the positioner that can be set with VLS.
Maximum System Acceleration (Phys. Unit/s²) 0x19001502	Maximum system acceleration This parameter is write-protected and is adapted to the positioner of the system before delivery.

3.9.2 Profile Generator for Point-to-Point Motion

For point-to-point motion, the profile generator of the C-886 determines the dynamics profile.

INFORMATION

During a point-to-point motion, any new motion command sets the target position to a new value and the motion platform immediately approaches the new target position on an undefined path.

Parameters for the profile generator

Parameter	Description and Possible Values
Path Control Step Size (mm) 0x19001504	Step size for calculating the dynamics profile of the platform This parameter is write-protected and is adapted to the positioner of the system before delivery.
Trajectory Velocity (Phys. Unit/s) 0x19001510	Velocity for the motion platform of the positioner Changing the velocity with the VLS command overwrites the value of the parameter in the volatile memory.

Parameter	Description and Possible Values
Trajectory Acceleration (Phys. Unit/s²) 0x19001511	Acceleration for the motion platform of the positioner
Trajectory Jerk (Phys. Unit/s³) 0x19001512	Jerk for the motion platform of the positioner
Trajectory Source 0x19001900	Source of the dynamics profile for MOV commands For point-to-point motion that are triggered with the MOV command, the parameter must have the value 0 (default).

Commands for the profile generator

Command	Syntax	Function
VLS	VLS <SystemVelocity>	Sets the velocity for the motion platform of the positioner. Limited by the Maximum System Velocity (Phys. Unit/s) parameter (ID 0x19001500) and Minimum System Velocity (Phys. Unit/s) parameter (ID 0x19001501).

Check of the dynamics profile created by the profile generator

When the dynamics profile for the positioner is created by the profile generator, it is checked before the start of each motion whether the motion platform can actually reach the nodes of the calculated profile and the commanded target position. If a node or the target position cannot be reached, the motion is not executed. The following is checked:

- Are the nodes and the target position outside of the travel range limits that can be queried by **TMN?** (p. 193) and **TMX?** (p. 194) or **TRA?** (p. 195)?
- Have the soft limits set with **NLM** (p. 178) and **PLM** (p. 180) been activated with **SSL** (p. 188), and if yes, are the nodes and the target position outside of these soft limits?
- Are the individual drives able to move the platform to the required nodes and the specified target position?

The **VMO?** command (p. 201) queries whether a specified target position can be reached.

3.9.3 Cyclic Transfer of Target Positions

For the cyclic transfer of target positions, the dynamics profile is specified by consecutive MOV commands.

NOTICE



Cyclic transfer of target positions!

Acceleration / deceleration, velocity, and steadiness of the motion depend on the following factors during the cyclic transfer of target positions:

- Target position values
- Observance of the cycle time

The execution of an unsuitable dynamics profile can tilt the positioner. Tilting can damage the positioner and/or the load affixed to it.

- For this reason, observe the following during the cyclic transfer of target positions by consecutive MOV commands:
 - The path that is specified by the target positions must be continuously differentiable at least twice.
 - During the execution of the dynamics profile, the maximum permissible velocity and acceleration of the positioner must **not** be exceeded.
 - To generate the target positions and continuously transfer them to the C-886 during the motion, it is recommended to use a suitable program.
- Use the buffer of the C-886 to make sure that the cycle time is observed:
 - Store the dynamics profile in the buffer of the C-886 before execution. For this purpose, use `SPA` to set the **Trajectory Execution** parameter (ID 0x19001901) to the value 1.
 - For a meaningful use of the buffer, increase the value of the **Threshold for Trajectory Execution** parameter (0x19001903) with `SPA` (default = 1).

Parameters for the cyclic transfer of target positions

Parameter	Description and Possible Values
Path Control Step Size (mm) 0x19001504	Step size for calculating the dynamics profile of the platform This parameter is write-protected and is adapted to the positioner of the system before delivery. The distance between the target positions set with consecutive MOV commands may only be maximally as large as the value of the parameter 0x19001504, in order to avoid tilting the positioner.
Trajectory Source 0x19001900	Source of the dynamics profile for MOV commands For the cyclic transfer of target positions by consecutive MOV commands, the parameter must have the value 1.

Parameter	Description and Possible Values
Trajectory Execution 0x19001901	Execution of the dynamics profile Determines how the dynamics profile defined by consecutive MOV commands is executed: 0 = Dynamics profile is executed immediately (default) 1 = Dynamics profile is stored in a buffer before execution This parameter is only evaluated when the parameter 0x19001900 has the value 1.
Maximum Number of Trajectory Points 0x19001902	Maximum number of dynamics profile points Indicates the maximum size of the buffer. This parameter is write-protected and only evaluated when the parameters 0x19001900 and 0x19001901 both have the value 1.
Threshold for Trajectory Execution 0x19001903	Threshold value for executing the dynamics profile Determines how many dynamics profile points have to be stored in the buffer (by consecutive MOV commands) until the execution of the dynamics profile begins. This parameter is only evaluated when the parameters 0x19001900 and 0x19001901 both have the value 1.
Current Number Threshold for Trajectory Execution 0x19001904	Shows the current number of dynamics profile points in the buffer. The parameter value is always 0 when the dynamics profile is determined by the profile generator or the dynamics profile defined by consecutive MOV commands is executed immediately. This parameter is write-protected.

Commands for the cyclic transfer of target positions

Command	Syntax	Function
MOV	MOV {<AxisID> <Position>}	Consecutive MOV commands specify the single target positions. Other motion commands and starting the wave generator output are not allowed.
SCT	SCT "T" <CycleTime>	Determines the cycle time for performing a dynamics profile. The cycle time is used to calculate the velocity during the motion so that the specified points of the dynamics profile are each reached precisely at the end of the time interval (where possible with regard to the velocity and acceleration limits).
VLS	VLS <SystemVelocity>	Limits the velocities of the individual drives.

Command	Syntax	Function
#11	#11	Queries the free memory space of the buffer, the contents of which determine the dynamics profile of the positioner, when the parameters 0x19001900 and 0x19001901 both have the value 1.

3.9.4 Coordinate Systems

The position display, direction of motion, and center of rotation for the motion platform of the positioner are determined by coordinate systems that are linked in a chain. The chain is basically structured as follows (starting point > end point): --> HEXAPOD coordinate system, --> leveling coordinate system, --> orientational coordinate system, --> operating coordinate system.

The coordinate systems are always right-handed systems.

The HEXAPOD coordinate system determines the fundamental properties of all other coordinate systems. HEXAPOD is based on the configuration file with the geometrical data of the positioner. The dimensional drawing in the manual of the positioner shows the respective position of the HEXAPOD coordinate system.

Using the controller, custom coordinate systems can be defined and used instead of the default coordinate systems.

Based on HEXAPOD, the orientational and leveling coordinate systems adapt fundamental properties of the enabled operating coordinate system, and in most applications their custom definition and enabling is not necessary at all or only once.

With the operating coordinate system, the position display, direction of motion, and the center of rotation for the motion platform of the positioner is adapted to the application. It is also possible to use --> work-and-tool coordinate systems. In the default setting, the operating coordinate system ZERO is active.

The most important commands for working with user-defined coordinate systems:

- KSD (p. 160): Define operating coordinate system by entering the offset values for the axes X, Y, Z, U, V, and W
- KSF (p. 162): Define operating coordinate system at the current position of the motion platform of the positioner ("home coordinate system")
- KLN (p. 154): Link coordinate systems with each other
- KEN (p. 144): Activate coordinate systems
- WPA (p. 213): Save the currently valid settings for coordinate systems
- DPA (p. 126): Reset settings for parameters and coordinate systems to default settings

INFORMATION

Coordinate systems can be conveniently defined, linked, enabled, and saved in PIMikroMove.

The **Define Home Coordinate System (KSF)** and **Manage Coordinate Systems ...** buttons are available in the **Positioner Platform** window for this. The positioner and the active coordinate system are graphically displayed on the **Positioner 3D View** card.

If the **Positioner Platform** window (normally docked) and the **Positioner 3-D View** card are **not** displayed in the main window:

- Display the **Positioner Platform** window with the **C-886 > Show Positioner platform settings** menu item.
- Display the **Positioner 3-D View card** with the **C-886 > Positioner 3-D View > Show** menu item.

Further information on coordinate systems and the work-and-tool concept can be found in the technical note "Coordinate Systems for Hexapod Microrobots" (C887T0007) and in "Definition of Terms" (p. 2).

3.9.5 Rotations

Rotations take place around the center of rotation. A given rotation in space is calculated from the individual rotations in the order $U \rightarrow V \rightarrow W$. This takes place regardless of whether the values for U , V , and W were explicitly given with the current command or result from the previous commands.

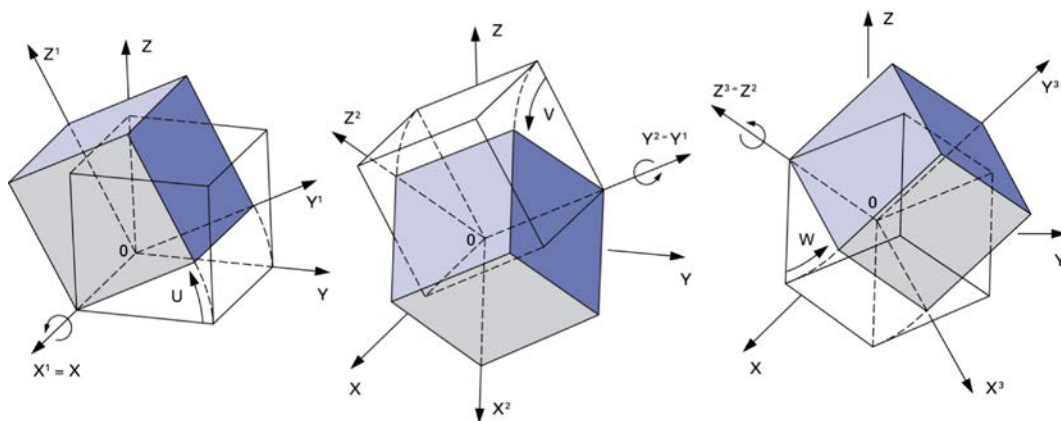


Figure 1: Order of elementary rotations of the hexapod platform when reaching a position (from left to right)

The center of rotation describes the intersection of the rotational axes U , V , and W .

The center of rotation always moves together with the platform.

Depending on the active \rightarrow operating coordinate system, the center of rotation can be moved from the origin of the coordinate system in the X and/or Y and/or Z direction with the **SPI** command. The center of rotation that can be moved using the **SPI** command is also referred to as "pivot point".

The pivot point defined using SPI is only used for rotations and can be changed only if the ZERO coordinate system or a coordinate system of the KSF type is enabled as the operating coordinate system.

For further information on calculating translations and rotations by the C-886, see the "Motion of the Hexapod - Position and Orientation in Space, Center of Rotation" technical note (C887T0021).

3.10 Communication Interfaces

Available communication interfaces

The C-886 can be controlled with ASCII commands via the following interfaces:

- TCP/IP
- USB

3.11 Overview of PC Software

The following table shows the PC software that is included in the C-886 CD. The given operating systems stand for the following versions:

- Windows: Vista Service Pack 1, Windows 7, 8, and 10 (32 bit, 64 bit)
- Linux: Kernel 2.6, GTK 2.0, glibc 2.4 (configuration used to develop the PC software)

PC software	Supporting operating system	Short description	Recommended use
PIMikroMove	Windows	Graphical user interface for Windows with which the C-886 and other controllers from PI can be used: <ul style="list-style-type: none"> ▪ The system can be started without programming effort ▪ Graphic representation of the motion ▪ Macro functionality for storing command sequences on the PC (host macros) ▪ Complete environment for command entry, for trying out different commands No command knowledge is necessary to operate PIMikroMove.	For users who want to perform simple automation tasks or test their equipment before or instead of programming an application. A log window showing the commands sent makes it possible to learn how to use the commands.
Dynamic program library for GCS	Windows, Linux	Allows software programming for the C-886 with programming languages such as C++. The functions in the dynamic program library are based on the PI General Command Set (GCS).	For users who would like to use a dynamic program library for their application. Is required for PIMikroMove. Is required for the LabVIEW drivers.

PC software	Supporting operating system	Short description	Recommended use
PI Hexapod 3D Library	Windows	<p>Program library that provides the following functions for PIMikroMove:</p> <ul style="list-style-type: none"> 3-D display of the motion of the parallel-kinematic positioner and the active coordinate system Configuration and administration of user-defined coordinate systems 	Is required for PIMikroMove.
LabVIEW drivers	Windows, Linux	LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments). The C-886 LabVIEW software is a collection of virtual instrument drivers (VI drivers) for the C-886 controller. The drivers support the PI General Command Set.	For users who want to use LabVIEW to program their application.
LabVIEW Merge Tool	Windows	The LabVIEW Merge Tool allows you to combine product-specific LabVIEW drivers from PI with each other.	For users who want to operate several products from PI at the same time while using LabVIEW.
MATLAB drivers	Windows	<p>MATLAB is a development environment and programming language for numerical calculations (must be ordered separately from MathWorks).</p> <p>The PI MATLAB driver consists of a MATLAB class that can be included in any MATLAB script. This class supports the PI General Command Set.</p> <p>The PI MATLAB driver does not require any additional MATLAB toolboxes.</p>	For users who want to use MATLAB to program their application.
PIPython	Windows, Linux, OS X	<p>Python is a programming language that is also used as a script language (obtainable separately as open-source software).</p> <p>PIPython is a collection of Python modules that support the GCS.</p> <p>The PIPython modules can be used with Python 2.7+ and 3.4+. Use on other operating systems is also possible via sockets.</p>	For users who want to use Python to program scripts for their application.
PI Terminal	Windows	Terminal program that can be used for nearly all PI controllers (see the description of the Command Entry window in the PIMikroMove user manual).	For users who want to send GCS commands directly to the controller.
PI Update Finder	Windows	Checks the PI software installed on the PC. If more current versions of the PC software are available on the PI server, downloading is offered.	For users who want to update the PC software.
PI Hexapod DataFiles	Windows	Configuration files for graphic display of the parallel-kinematic positioners in the PC software	Needed for the PI Hexapod 3-D library.

PC software	Supporting operating system	Short description	Recommended use
PIStages3Editor	Windows	Program for opening and editing stage databases in .db format.	For users who use optional single axes.
PIFirmware-Manager	Windows	Program for updating the firmware of the C-886.	For users who want to update the firmware.
USB driver	Windows	Driver for the USB interface	For users who want to connect the controller to the PC via the USB interface.

4 Unpacking

1. Unpack the C-886 with care.
2. Compare the contents with the items listed in the contract and the packing list.
3. Inspect the contents for signs of damage. If parts are missing or you notice signs of damage, contact PI immediately.
4. Keep all packaging materials in case the product needs to be returned.

5 Installation

In this Chapter

Installing the PC Software.....	39
Ensuring Ventilation.....	42
Connecting the C-886 to the Protective Earth Conductor.....	42
Installing Slave Modules in the C-886.....	43
Connecting the Power Supply to the C-886.....	44
Connecting Drives	45
Connecting the PC.....	46

5.1 Installing the PC Software

The communication between the C-886 and a PC is necessary to configure the C-886 and send motion commands using the commands of the GCS. Various PC software applications are available for this purpose.

5.1.1 Performing Initial Installation

Accessories

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

Installing the PC software in Windows

1. Start the installation wizard by double-clicking the **PI_C-886.CD_Setup.exe** file in the root directory (main directory of the CD).

The **InstallShield Wizard** window for the installation of programs and manuals for the C-886 opens.

2. Follow the instructions on the screen.

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

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

- LabVIEW driver
Exception: The *Analog LabVIEW drivers* component is provided for some PI controllers. This component is only available through user-defined installation.
- Dynamic program library for GCS
- PIMikroMove
- PC software for updating the firmware of the C-886
- PI Update Finder for updating the PC software
- For controllers that have a USB interface for communication with the PC: USB drivers

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

Installing the PC software in Linux

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

You can select individual components for installation.

5.1.2 Installing Updates

PI is constantly improving the PC software.

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

Requirements

- ✓ Active connection to the Internet.
- ✓ If your PC uses a Windows operating system:
 - You have installed (p. 39) the PI Update Finder from the product CD.
 - You have the A000T0028 Technical Note for the PI Update Finder ready. You will find the document on the product CD.
 - If the PC to be updated is **not** directly connected to the Internet:
You have the A000T0032 Technical Note for the PI Update Finder at hand. You will find the document on the product CD.

- ✓ If your PC uses a Linux operating system:
 - You have the user name and password for the C-886 ready. Both of these can be found in the file "xxx_Releasenews.pdf" (x_x_x: version number of the CD) in the \Manuals folder on the product CD.

Updating the PC software on Windows

- Use the PI Update Finder:
 - When the PC to be updated is directly connected to the Internet: Follow the instructions in the A000T0028 Technical Note (TECHNICAL_NOTE_PI_UPDATE_FINDER_xx.pdf).
 - When the PC to be updated is **not** directly connected to the Internet: Follow the instructions in the A000T0032 Technical Note .

Updating the PC software in Linux

1. Open the website www.pi.ws.
2. Click **Login**.
3. Log in with the user name and the password from the "xxx_Releasenews.pdf" file on the product CD.
4. Click **Search**.
5. Enter the product number up to the period (e. g., C-886) in the search field.
6. Click **Start search** or press the **Enter** key.
7. Open the corresponding product detail page in the list of search results:
 - a) If necessary: Scroll down the list.
 - b) If necessary: Click **Load more results** at the bottom of the list.
 - c) Click the corresponding product in the list.
8. Scroll down to the **Downloads** section on the product detail page.
The software files are shown under **Software Downloads**.
9. Click on the archive file "CD Mirror" or on the associated download link.
10. In the following request, select the option to save the file to your PC.
If you do not specify anything else, the "CD Mirror" archive file is stored in the default download directory of your PC.
11. Unpack the archive file to a separate installation directory.
12. In the directory with the unpacked files, go to the **Linux** subdirectory.
13. Unpack the archive file in the **Linux** directory by entering the command `tar -xvpf <name of the archive file>` on the console.

14. Read the accompanying information on the software update (readme file and/or "xxx_Releasenews.pdf" file) and decide whether the update makes sense for your application.
 - If no: Stop the update procedure.
 - If yes: Perform the following steps.
15. Log onto the PC as a superuser (root rights).
16. Install the update.

INFORMATION

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

- Contact our customer service department (p. 251).

5.2 Ensuring Ventilation

High temperatures can overheat the C-886.

- Set up the C-886 with a distance of at least 10 cm to the top and rear sides and at least 5 cm to the sides. If this is not possible, make sure that the area is cooled sufficiently.
- Ensure sufficient ventilation at the place of installation.
- Keep the ambient temperature to a non-critical level (<50° C).

5.3 Connecting the C-886 to the Protective Earth Conductor

INFORMATION

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

Requirements


- ✓ The C-886 is switched off, i. e., the power supply is **not** connected to the C-886.

Tools and accessories

- Suitable protective earth conductor:
 - Cable cross section $\geq 0.75 \text{ mm}^2$
 - Contact resistance < 0.1 ohm at 25 A at all connection points relevant for attaching the protective earth conductor

- Fastening material for the protective earth conductor, sits on the protective earth connection (threaded bolt) in the following order on delivery of the C-886, starting from the housing:
 - Toothed washer
 - Nut
 - Flat washer
 - Safety washer
 - Nut
- Suitable wrench

Connecting the C-886 to the protective earth conductor

1. If necessary, attach a suitable cable lug to the protective earth conductor.
2. Remove the outer nut from the protective earth connection on the rear panel of the C-886 (threaded bolt (p. 16) marked with ).
3. Connect the protective earth conductor:
 - a) Push the cable lug of the protective earth conductor onto the threaded bolt.
 - b) Screw the nut onto the threaded bolt. In this way, the cable lug of the protective earth conductor is wedged between the toothed washer and the nut.
 - c) Tighten the nut with at least three turns and a torque of 1.2 Nm to 1.5 Nm.

5.4 Installing Slave Modules in the C-886

INFORMATION

Slave modules can be installed in the C-886 for up to two optional single axes.

If you have ordered the optional single axes with the positioner system, PI will configure the C-886 according to your order so that the corresponding slave modules are already installed.

Requirements

- ✓ The C-886 is **not** connected to the power socket via the power cord.

Tools and Accessories

- Slave module that has been ordered as an optional accessory (p. 22)
- If necessary: Cover plates for free slots, available as optional accessories (p. 22)
- Phillips-head screwdriver

Installing slave modules in the C-886

1. Remove the cover of the slot, which is located in the housing of the C-886 directly next to the last occupied slot.

2. Insert the slave module into the C-886:
 - a) Push the slave module carefully into the slot, **in the correct orientation**.
 - b) If you feel resistance, continue to push the slave module until it engages in the terminal strip of the housing.
 - c) Check that the slave module fits correctly.
 - d) Affix the slave module with screws on the housing of the C-886.
3. If there are free slots in the C-886 without a cover, close these slots with a suitable cover plate.

5.5 Connecting the Power Supply to the C-886

Requirements

- ✓ The power cord is **not** connected to the power socket.
- ✓ The C-886 is installed near the power supply so that the power plug can be quickly and easily disconnected from the mains.

Tools and accessories

- Included 24 V wide-range-input power supply (for line voltages between 100 and 240 VAC at 50 or 60 Hz)
- Alternative: Sufficiently dimensioned power supply
- Adapter provided for the power supply connection (p. 21); Molex Mini Fit 6-pin (m) to 4 mm² ferrules
- Alternative: Sufficiently dimensioned adapter
- Included power cord
- Alternative: Sufficiently dimensioned power cord

Connecting the supplied power supply to the C-886

- Connect the adapter with the 24 V connection of the C-886:
 - a) Insert the ferrule of the black cable into the **GND** feed-through terminal.
 - b) Insert the ferrule of the red cable into the **+24V** feed-through terminal.
 - c) Use the integrated screws to secure the connections against accidental disconnection.
- Connect the Molex connector of the adapter to the Molex socket of the power supply.
 - Make sure that the lock of the connection is engaged in the socket.
- Connect the power cord to the power supply.

5.6 Connecting Drives

NOTICE

**Damage if an unsuitable drive type is connected!**

Connecting an unsuitable drive type can cause damage to the drive or the C-886.

- Only connect a suitable drive to the slave modules of the C-886.

INFORMATION

C-886 and positioner are delivered as a preconfigured system.

- If a connection assignment is specified on the labels of the C-886 and/or positioner, observe this assignment when connecting the positioner.

Requirements

- ✓ The C-886 is switched off, i.e., the power supply is **not** connected to the power socket with the power cord.
- ✓ If you want to connect optional single axes: The corresponding slave modules are installed in the C-886 (p. 43).
- ✓ You have read and understood the user manuals of all drives to be connected (parallel-kinematic positioner, optional single axes).
- ✓ You have installed the drives to be connected according to the instructions in the corresponding user manuals.

Tools and Accessories

- Parallel-kinematic positioner with suitable drive type
- If you want to use optional single axes: Stage with suitable drive type

Connecting drives

1. Connect the drives to the sockets of the slave modules.
2. Use the integrated screws to secure the connections against accidental disconnection.

5.7 Connecting the PC

Communication between the C-886 and a PC is required to configure the C-886 and to command motion using the GCS commands. The C-886 has the following interfaces for this purpose:

- TCP/IP interface
- USB interface

In this section, you learn how to establish proper cable connections between the C-886 and a PC as well as in a TCP/IP network. All other steps required for establishing communication between C-886 and PC are described in the following sections:

- "Establishing Communication via the TCP/IP Interface" (p. 53)
- "Establishing Communication via USB" (p. 60)

5.7.1 Connecting the C-886 via the TCP/IP Interface

Requirements

- ✓ If the C-886 is to be directly connected to the PC:
The PC has a free RJ45 Ethernet connection socket.
- ✓ If the C-886 and a PC are to be operated together in a network:
A free access point to the network is available for the C-886; a suitable hub or switch is connected to the network for this purpose if necessary.

Tools and Accessories

- If the C-886 is to be directly connected to the PC: Cross-over network cable
- If the C-886 is to be connected to a network access point: Straight-through network cable

Connecting the C-886 directly to the PC

- Connect the RJ45 socket of the C-886 to the RJ45 Ethernet connection socket of the PC via the cross-over network cable.

Connecting the C-886 to the network in which the PC is also located

- Connect the RJ45 socket of the C-886 with the network access point via the straight-through network cable.

5.7.2 Connecting the C-886 via the USB interface

Requirements

- ✓ The PC has a free USB interface.

Tools and accessories

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

Connecting the C-886 to the PC

- Connect the USB socket of the C-886 and the USB interface of the PC with the USB cable.

6 Startup

In this Chapter

General Notes on Startup	49
Switching on the C-886	52
Establishing Communication via the TCP/IP Interface.....	53
Establishing Communication via the USB Interface.....	60
Configuring Slave Modules for Single Axes.....	62
Starting Motion.....	66

6.1 General Notes on Startup

CAUTION



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

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the C-886 in the case of malfunction or failure of the system. If touch voltages exist, touching the C-886 can result in minor injuries from electric shock.

- Connect the C-886 to a protective earth conductor (p. 42) before start-up.
- 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-886 to the protective earth conductor before starting it up again.

CAUTION



Risk of crushing by moving parts!

There is a risk of minor injuries from crushing between the moving parts of the positioner and a stationary part or obstacle.

- Keep your fingers away from areas where they can get caught by moving parts.

NOTICE**Incorrect configuration of the C-886!**

The configuration of the C-886 must be adapted to the positioner. If an incorrect configuration is used, the positioner can be damaged by uncontrolled motion or collisions. The C-886 is configured for the used positioner on delivery.

- Check the configuration: Once you have established communication via TCP/IP (p. 53) or USB (p. 60), send the `CST?` command. The response shows the positioner to which the C-886 is adapted.
- Operate the positioner only with a C-886 whose configuration data is adapted to the positioner.
- Change the settings of the slave modules for the drives of the positioner only after consultation with PI.

NOTICE**Damage from collisions!**

Collisions can damage the positioner, the load to be moved, and the surroundings.

- Make sure that no collisions are possible between the positioner, the load to be moved, and the surroundings in the workspace of the positioner.
- Do not place any objects in areas where they can be caught by moving parts.
- Stop the motion immediately if a controller malfunction occurs.

NOTICE**Damage from unintentional position changes!**

The maximum holding force when the servo mode is switched off is based on the self-locking of the drives and can be lower than the maximum load capacity when the servo mode is switched on (see manual of the positioner).

When the actual load of the positioner exceeds the maximum holding force based on the self-locking of the drives, unintentional position changes of the positioner can occur in the following cases:

- Switching off the C-886
- Rebooting the C-886
- Switching off the servo mode for the axes of the motion platform of the positioner

As a result, collisions are possible between the positioner, the load to be moved, and the surroundings. Collisions can damage the positioner, the load to be moved, or the surroundings.

- Make sure that the actual load of the motion platform of the positioner does not exceed the maximum holding force based on the self-locking of the drives before you switch off the servo mode, reboot the C-886, or switch it off.

NOTICE**Damage from collisions during the reference move!**

During a reference move, the positioner moves unpredictably.

As a result, collisions are possible between the positioner, the load to be moved, and the surroundings. Collisions can damage the positioner, the load to be moved, and the surroundings.

- Make sure that no collisions between the positioner, the load to be moved, and the surroundings are possible during the reference move of the positioner.
- Do not place any objects in areas where they can be caught by moving parts during the reference move.
- After a successful reference move, supply a command for the corresponding target position in order to return to the reference position (default: zero position) from any given position. Do **not** start a new reference move.

NOTICE**Damage from uncontrolled motion of the positioner!**

The velocity and acceleration of the motion platform of the positioner are **not** specified by the C-886 in the following cases:

- Cyclic transfer of target positions (der **Trajectory Source** parameter (ID 0x19001900) has the value 1)
- The positioner (axes X, Y, Z, U, V, W) is still moving while a new motion command is sent. The previous target position is thereby overwritten without the velocity and acceleration of the motion platform of the positioner being recalculated.

The platform of the positioner then moves on an undefined path. On this undefined path, collisions with the environment of the positioner are possible. Collisions can damage the positioner, the load to be moved, and the surroundings.

If you trigger motion by the cyclic transfer of target positions (p. 30):

- Only set target positions whose distance from each other is maximally as large as the value of the **Path Control Step Size** parameter (ID 0x19001504) with consecutive **MOV** commands.

If you command point-to-point motion with motion commands (p. 27):

- Avoid sending new target positions while the positioner (axes X, Y, Z, U, V, W) is still moving.
- If new target positions have to be sent while the positioner is still moving (axes X, Y, Z, U, V, W): Only use motion commands to set target positions that maximally deviate from the current position by the value of the **Path Control Step Size** parameter (ID 0x19001504).

INFORMATION

The communication between the C-886 and a PC can be used to configure the C-886 and send motion commands with the commands of the GCS.

- Communication is possible via the USB interface without further settings.
- For communication via TCP/IP, it may be necessary to adjust the interface parameters accordingly once.

INFORMATION

The communication interfaces of the C-886 (TCP/IP, USB) are active at the same time. Commands are executed in the order in which the complete command lines arrive. The simultaneous use of several communication interfaces can cause problems with the PC software, however.

- Always only use one interface of the C-886.

INFORMATION

For axes with incremental sensors, motion can only be commanded after a successful reference move (p. 132) (also referred to as "initialization").

The behavior of the positioner after the reference move is determined by the **Behaviour After Reference Move** parameter (ID 0x07030401) and **Target For Motion After Reference Move** parameter (ID 0x07030402). Depending on the parameter values, the platform can be moved automatically to a specified position, e.g., after the reference move.

- Value of the parameter 0x07030401 = 0: Motion platform remains in the reference position after the reference move.
- Value of the parameter 0x07030401 = 1: After the reference move, the motion platform moves to the absolute target position, which is given by parameter 0x07030402.

No reference move is required for axes with absolute-measuring sensors. The use of the FRF command is still recommended for these axes, however. FRF does **not** start a reference move for axes with absolute-measuring sensors but sets the target positions to the current position values. The above-described parameter values also go into effect, so that the axes can be moved to a defined "initial position", for example.

6.2 Switching on the C-886

Requirements

- ✓ You have read and understood the general notes on startup (p. 49).
- ✓ The C-886 has been installed properly (p. 39).

Switching on the C-886

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

The C-886 starts the firmware. The starting procedure takes around 10 seconds. After the end of the starting procedure, the **Power** LED lights up green.

6.3 Establishing Communication via the TCP/IP Interface

Adaptation of the interface parameters

Before communication is established, it can be necessary to adapt the factory settings of the interface parameters once. The following interface parameters for the TCP/IP communication can be adapted in the nonvolatile memory of the C-886 with the `IFS` command (p. 140):

Interface parameters	Factory setting	Note
Default IP address (IPADR)	192.168.1.28:50000	Allows the definition of a static (i.e., fixed) address. This static address is not used when the C-886 is configured for assignment of an IP address by a DHCP server (default setting of the startup behavior for configuration of the IP address). If the static address is to be used: <ul style="list-style-type: none"> ▪ The startup behavior must be changed so that the C-886 uses the IP address defined with "IPADR". ▪ The IP addresses and subnet masks of the C-886 and PC as well as of all other network devices must be compatible with each other. For details, see "Preparing the PC and C-886 for Using Static IP Addresses".
Start-up behavior for configuring the IP address for TCP/IP communication (IPSTART)	DHCP is used to obtain the IP address	The IP address of the C-886 is assigned via DHCP by the default setting of the startup behavior. The default setting of the startup behavior only has to be changed if the network devices are to use static addresses instead.
Subnet mask (IPMASK)	255.255.255.0	The default setting of the subnet mask may have to be changed if the network devices are to use static addresses. For details, see "Preparing the PC and C-886 for Using Static IP Addresses".

After switching on or rebooting the C-886

The starting procedure of the C-886 must be finished before the communication between the C-886 and PC can be established. The starting procedure takes around 10 seconds.

When the IP address of the C-886 is assigned via DHCP, an additional waiting time is required at the end of the starting procedure of the C-886 before communication is possible via TCP/IP.

Connection of the network cable when the controller is switched on

The establishment of communication via TCP/IP can fail if the network cable was connected to the RJ45 socket of the C-886 while the C-886 was switched on.

- If the establishment of communication fails, switch the C-886 off and back on again while the network cable is plugged in.

Port setting

For communication via TCP/IP, the C-886 has one unchangeable port (50000) available.

6.3.1 Preparing the PC and C-886 for Using Static IP Addresses

If a network does not have a DHCP server or the C-886 is directly connected to the Ethernet connection socket of the PC, the following adaptations of the interface parameters are necessary:

- Set the startup behavior for configuring the IP address of the C-886 so that a static address is used
- The IP addresses and subnet masks of the C-886 and PC as well as all other network devices must be compatible with each other

To adapt IP addresses and subnet masks, you can choose one of the two following options:

- Adapt the PC settings and if necessary the settings of other network devices. The settings of the C-886 remain unchanged.
- Adapt the settings of the C-886. The PC settings and if necessary the settings of other network devices remain unchanged.

Requirements

- ✓ You have established communication between the C-886 and the PC via USB in to determine the settings of the C-886 and to change them if necessary.

Determining the IP address and subnet mask of the PC

1. Open the window in which the properties of the Internet protocol TCP/IP are displayed and set, in a suitable way on your PC. The necessary steps depend on the operating system used.

If your operating system distinguishes between Internet protocol version 4 (TCP/IPv4) and version 6 (TCP/IPv6) (e.g. Windows 7), open the window for version 4.

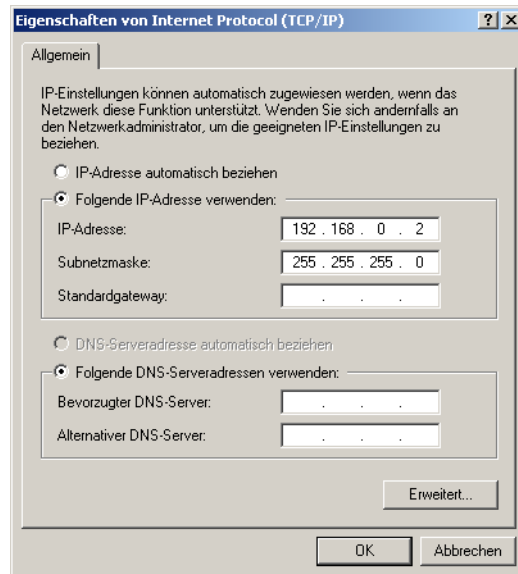


Figure 6: "Internet Protocol (TCP/IP) Properties" window with example settings (not necessarily suitable for your system)

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

2. Write down the settings.

Determining the IP address and subnet mask of the C-886, adapting the startup behavior of the C-886

1. If you have established communication between the C-886 and PC via the USB interface, open the window for entering commands in the used program.
2. Enter the `IFS?` command.
This command gets the values of the interface parameters in the nonvolatile memory
3. Write down the settings **IPMASK** and **IPADR**.
4. Make sure that the **IPSTART** parameter is correctly set:
 - If **IPSTART** = 0 (the static IP address defined with **IPADR** is used), the setting is correct.
 - If **IPSTART** ≠ 0: Send the command `IFS 100 IPSTART 0`.

INFORMATION

In PIMikroMove, you can determine the IP address and subnet mask of the C-886 and adapt the startup behaviour of the C-886 in the **Configure Interface** window without sending commands.

Adapting IP settings of the PC

- If you want to leave the PC settings unchanged, continue with the section "Adapting C-886 settings".
- 1. Activate **Use following IP address** in the window in which the properties of the Internet protocol TCP/IP (TCP/IPv4) are displayed and set.
- 2. Adapt the IP address and the subnet mask to the settings of the C-886:
 - a) Copy the first three sections of the IP address of the C-886 for the IP address on the PC.
 - b) Make sure that the last section of the IP address on the PC differs from the last section of the IP address of the C-886 and is not "255" or "0".
 - c) Copy the subnet mask of the C-886 for the subnet mask on the PC.
- Example:
 New IP address of the PC: 192.168.1.29 (if the C-886 has the IP address 192.168.1.28)
 New subnet mask of the PC: 255.255.255.0 (if the C-886 has the subnet mask 255.255.255.0)
- 3. Confirm the settings with the **OK** button.
- 4. If further network devices have to be adapted:
 Adapt the IP addresses and subnet masks as in the previous steps.
 Assign a separate, unique IP address to each network device.
 IP addresses must not occur twice in the same network.
- 5. Close the connection via the USB interface, e. g., in PIMikroMove via the **Connections > Close > C-886** menu item in the main window.
- 6. Switch off the C-886.
- 7. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 57).

Adapting C-886 settings

- 1. Adapt the settings of the C-886 to those of the PC with the **IFS** command:
 - a) Change the subnet mask with the **IFS 100 IPMASK xxx.xxx.xxx.xxx** command, whereby xxx.xxx.xxx.xxx is the subnet mask of the PC.
 - b) Change the IP address with the **IFS 100 IPADR xxx.xxx.xxx.yyy:50000** command, whereby the following applies:
 - xxx.xxx.xxx. matches the first three sections of the IP address of the PC
 - yyy differs from the last section of the IP address of the PC and every other device in the same network

- `yyy` is not "255" and not "0" and is in the address range that is given by the last section of the subnet mask
- The port address "50000" must not be changed

Example:

If the IP address of the PC is 192.168.0.1 and no other device has the IP address 192.168.0.2, send the `IFS 100 IPADR 192.168.0.2:50000` command.

2. Close the connection via the USB interface, e. g., in PIMikroMove via the **Connections > Close > C-886** menu item in the main window.
3. Switch off the C-886.
4. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 57).

6.3.2 Establishing Communication via TCP/IP in the PC Software

CAUTION



Risk of crushing from unexpected motion

When the communication between the C-886 and the PC is established via TCP/IP, the PC software offers all controllers present in the same network for selection. After a C-886 has been selected for the connection, all commands are sent to this controller. If the wrong controller is selected, the operating and maintenance staff of the connected positioner are at risk of slight injury from crushing due to unexpectedly supplied motion commands.

- If several C-886 are displayed in the PC software, make sure that you select the right C-886.
- Assign a unique name affix for each C-886 in the same network: Enter the name affix as a value of the **Customer Device Name** parameter (ID 0x0D001000).

INFORMATION

A freely selectable name affix can be assigned for the C-886, in order to distinguish among several C-886 in the same network or on the same PC. Proceed as follows for assigning the name affix:

1. Establish the communication via one of the available interfaces.
2. Change to command level 1 by sending:
`CCL 1 advanced`
3. Enter a unique name affix as value of the **Customer Device Name** parameter (ID 0x0D001000) by sending:
`SPA 1 0x0D001000 name affix`
4. Save the new parameter values to the nonvolatile memory by sending:
`WPA 101 1 0x0D001000.`
5. Close the connection.

Alternative:

- Identify the C-886 to be connected in the list of found controllers on the basis of its serial number (SN). The serial number of the controller is on the type plate on the rear panel of the C-886.

Requirements

- ✓ You have read and understood the general notes on startup (p. 49).
- ✓ The C-886 is connected to the network via the RJ45 socket or directly to the PC.
- ✓ If the C-886 is connected to a network:
The PC to be used for communication with the C-886 is suitably connected to the same network as the C-886.
- ✓ If the used network does not have a DHCP server or if the C-886 is directly connected to the Ethernet connection socket of the PC, so that static IP addresses have to be used:
By adapting the interface parameters, you have set the correct startup behavior for configuring the IP address of the C-886 and adapted the IP addresses and subnet masks of the C-886 and PC as well as all other network devices to each other.
- ✓ If several C-886 are connected with the same network via their TCP/IP interfaces: You have assigned a unique name affix for the C-886 with which the communication is to be established, via the **Customer Device Name** parameter (ID 0x0D001000). Alternatively, you have the serial number of this C-886 ready. The serial number is on the type plate on the rear panel of the C-886.
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 39).
- ✓ You have read and understood the manual of the PC software used. The software manuals are on the product CD.
- ✓ The C-886 is switched off.

Establishing communication via TCP/IP

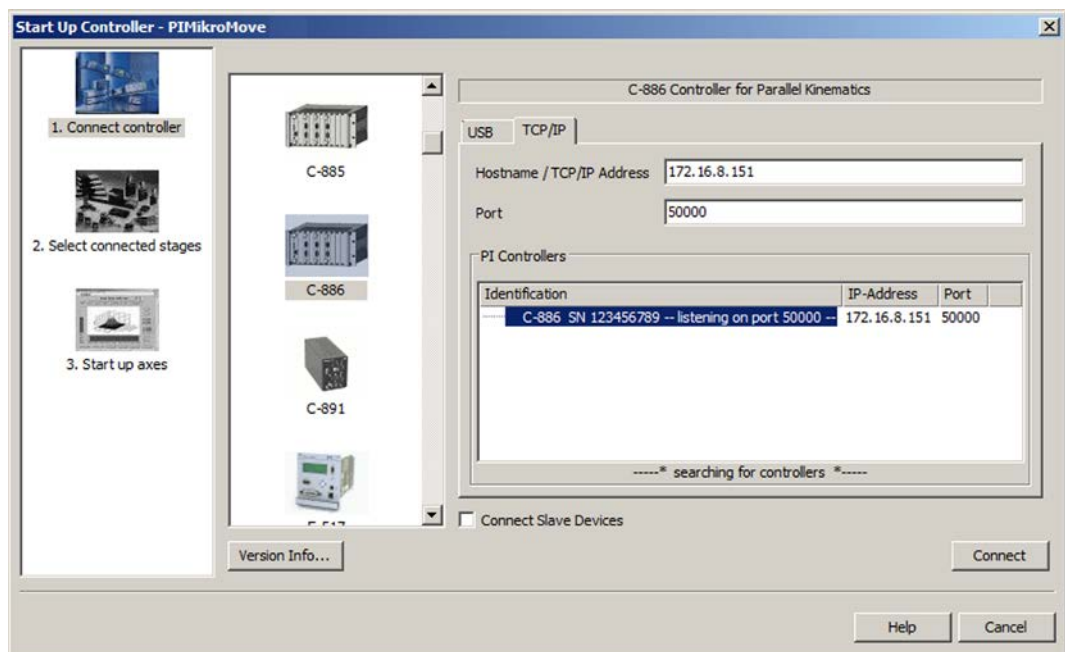
The procedure for PIMikroMove is described in the following. The procedure for other PC software programs (e.g., PITerminal, LabVIEW driver) is similar.

1. Switch on the C-886.
2. 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.
3. Select **C-886** in the field for controller selection.
 4. Select the **TCP/IP** tab on the right side of the window.

All controllers in the same network are shown.



5. Click the **C-886 ...** entry in the list of controllers found .
 - If several **C-886 ...** entries are displayed, identify your C-886 by the name affix that you have previously assigned or by its nine-digit serial number (**SN ...**).
 - If the C-886 is not displayed in the list of the controllers found, check the network settings (p. 247). Consult your network administrator if necessary.
6. Check the IP address in the **Hostname / TCP/IP Address** field and the port number in the **Port** field.
7. Make sure that the **Connect Slave Devices** check box is deactivated.

The **Connect Slave Devices** check box may only be activated when direct communication with the slave modules for the optional single axes is necessary; for further information, see "Configuring Slave Modules for Single Axes" (p. 62) and "Identifier for Direct Access to the Slave Modules" (p. 24).

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

When communication has been successfully established, the **Start up controller** window goes to the **Start up axes** step.

6.4 Establishing Communication via the USB Interface

INFORMATION

A freely selectable name affix can be assigned for the C-886, in order to distinguish among several C-886 in the same network or on the same PC. Proceed as follows for assigning the name affix:

1. Establish the communication via one of the available interfaces.
2. Change to command level 1 by sending:
`CCL 1 advanced`
3. Enter a unique name affix as value of the **Customer Device Name** parameter (ID 0x0D001000) by sending:
`SPA 1 0x0D001000 name affix`
4. Save the new parameter values to the nonvolatile memory by sending:
`WPA 101 1 0x0D001000.`
5. Close the connection.

Alternative:

- Identify the C-886 to be connected in the list of found controllers on the basis of its serial number (*SN*). The serial number of the controller is on the type plate on the rear panel of the C-886.

Requirements

- ✓ You have read and understood the general notes on startup (p. 49).
- ✓ The C-886 is connected to the USB interface of the PC
- ✓ The C-886 is switched on and the starting procedure of the C-886 has finished (p. 52).
- ✓ The PC is switched on.
- ✓ The required software and USB drivers are installed on the PC.
- ✓ You have read and understood the manual of the PC software used. The software manuals are on the product CD.

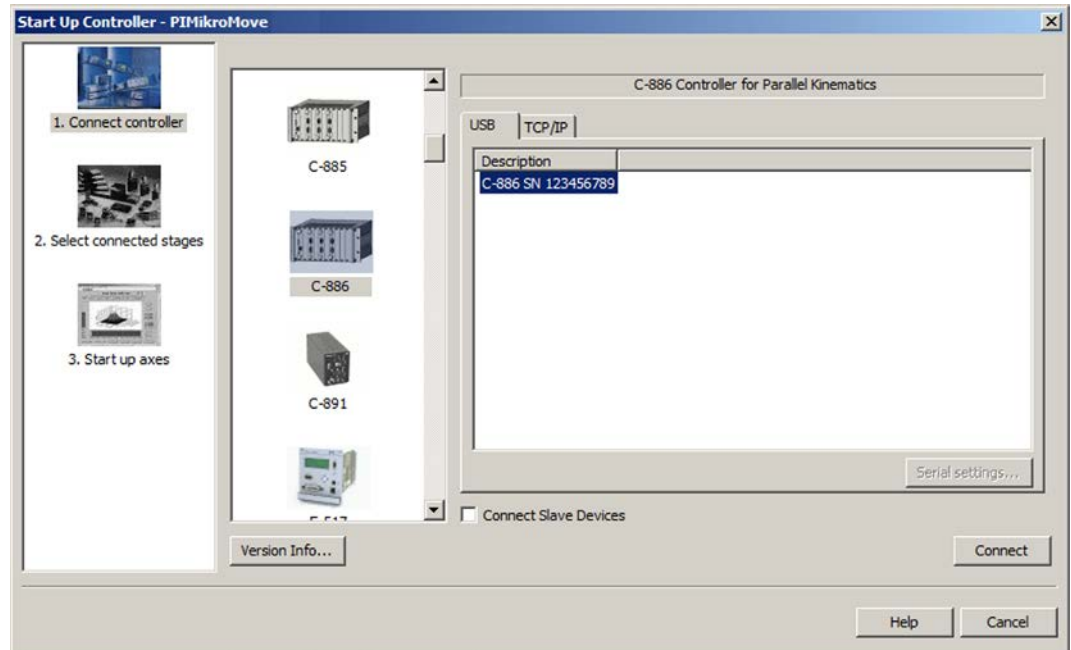
Establishing Communication via USB

The procedure for PIMikroMove is described in the following. The procedure for other PC software programs (e.g., PITerminal, LabVIEW driver) is similar.

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-886** 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-886** connected.
5. Make sure that the **Connect Slave Devices** check box is deactivated.

The **Connect Slave Devices** check box may only be activated when direct communication with the slave modules for the optional single axes is necessary; for further information, see "Configuring Slave Modules for Single Axes" (p. 62) and "Identifier for Direct Access to the Slave Modules" (p. 24).

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

When communication has been successfully established, the **Start up controller** window goes to the **Start up axes** step.

6.5 Configuring Slave Modules for Single Axes

NOTICE



Damage to the positioner due to accessing the slave modules directly!

The slave modules are configured for the positioner and are ready for use on delivery of the C-886. Changing settings or commanding motion by directly accessing these slave modules can lead to damage to the positioner.

- Directly access the slave modules for the drives of the positioner only after consultation with PI.

NOTICE



Damage from directly accessing slave modules for single axes!

The slave modules for the optional single axes can be installed (p. 43) and configured in the C-886 by the user. Unsuitable parameter settings of the slave modules can lead to improper operation or damage to the connected mechanics.

- Load the parameter settings for the optional single axes from a stage database from PI; see "Configuring slave modules for single axes in PIMikroMove."
- Only change parameter values for the optional single axes after careful consideration.
- Find out about adjusting settings in the manual of the corresponding module (p. 6).

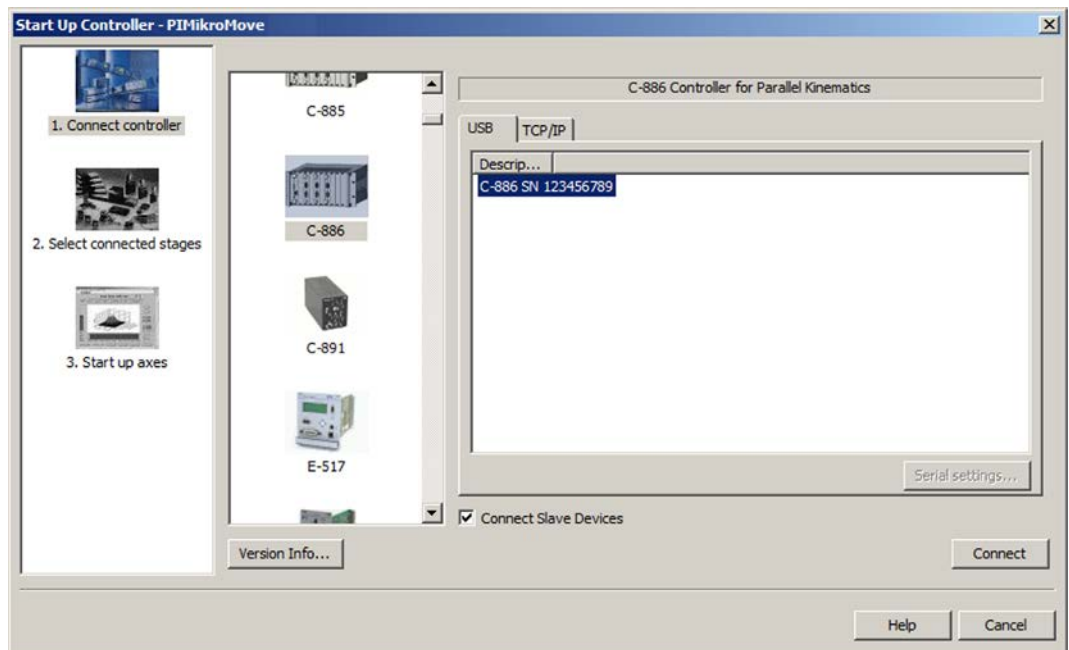
Requirements

- ✓ The C-886 has been installed properly (p. 39).
- ✓ If the C-886 is connected to the USB interface of the PC: You have carried out steps 1 to 4 from "Establishing Communication via the USB Interface" (p. 60) in PIMikroMove.
- ✓ If the C-886 is connected to the network via the RJ45 socket or directly to the PC: You have carried out step 1 to 6 from "Establishing Communication via TCP/IP in the PC Software" (p. 57) in PIMikroMove.

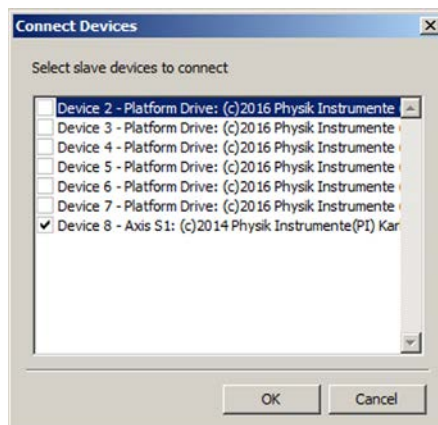
Configuring slave modules for single axes in PIMikroMove

The example in following instructions uses a C-886 with six slave modules for the parallel-kinematic positioner that is expanded with a slave module for an optional single axis.

1. In the **Start up controller** window, activate the **Connect Slave Devices** check box.



2. Click **Connect**.
The **Connect Devices** window opens.
3. In the **Connect Devices** window, select the slave modules for the optional single axes by activating the corresponding check boxes (**Device 8** and/or **Device 9**).
 - Do **not** enable the check boxes of the slave modules for the drives of the positioner (**Device 2** to **Device 7**). Before delivery of the C-886, the slave modules for the drives of the positioner were configured by PI for immediate use.

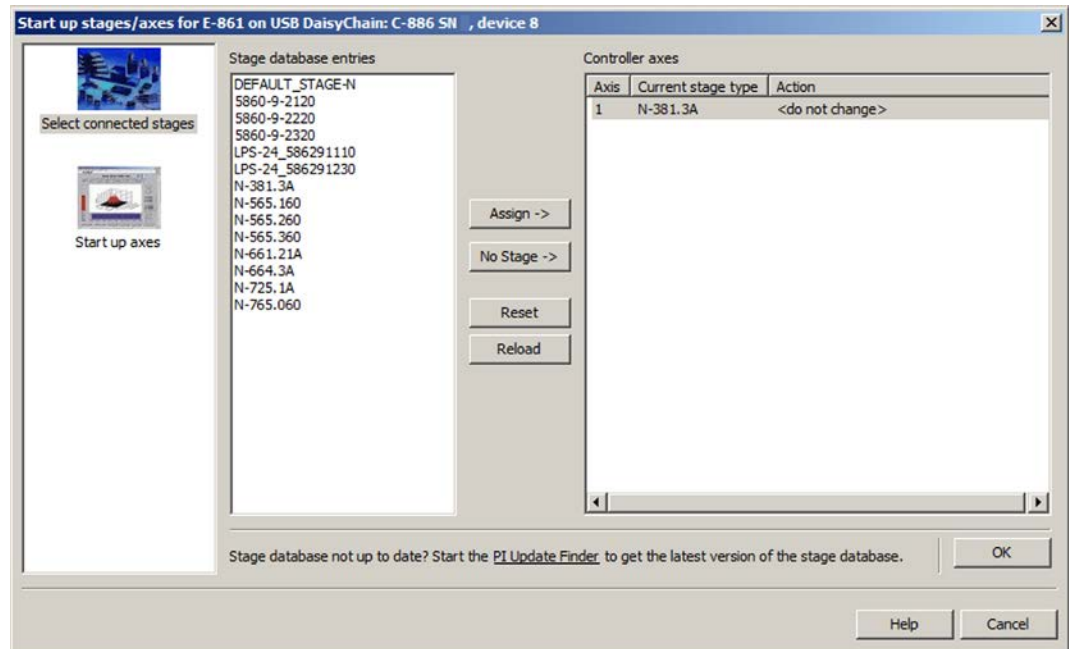
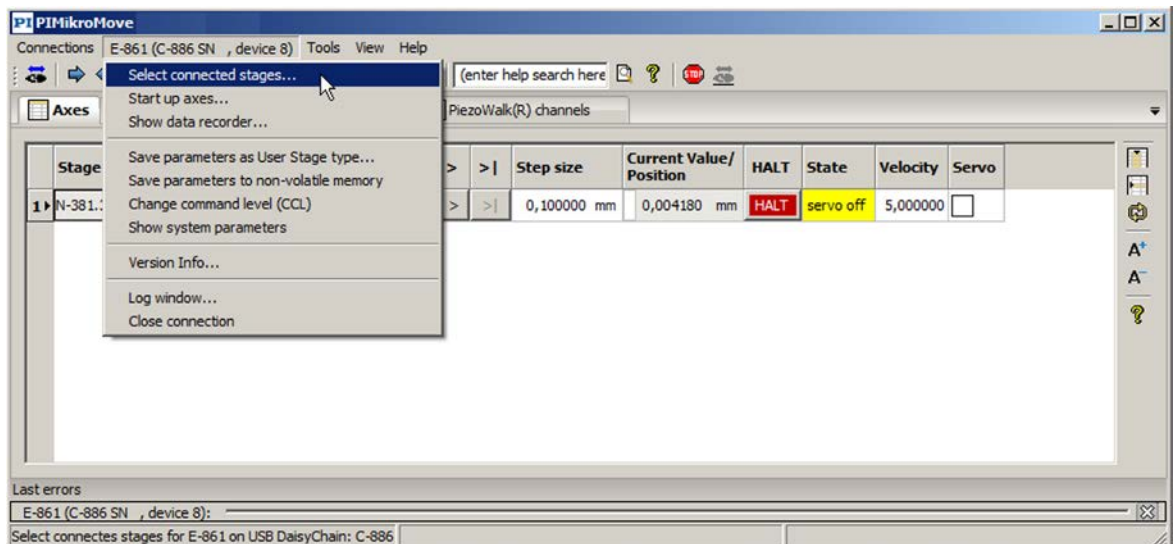


4. In the **Connect Devices** window, click **OK** to establish direct communication with the slave modules for the optional single axes.

It can take a few seconds to establish the communication. When communication has been successfully established, the **Start up controller** window closes and the main window of PIMikroMove opens.

5. Configure the parameter settings of the slave modules for the optional single axes:

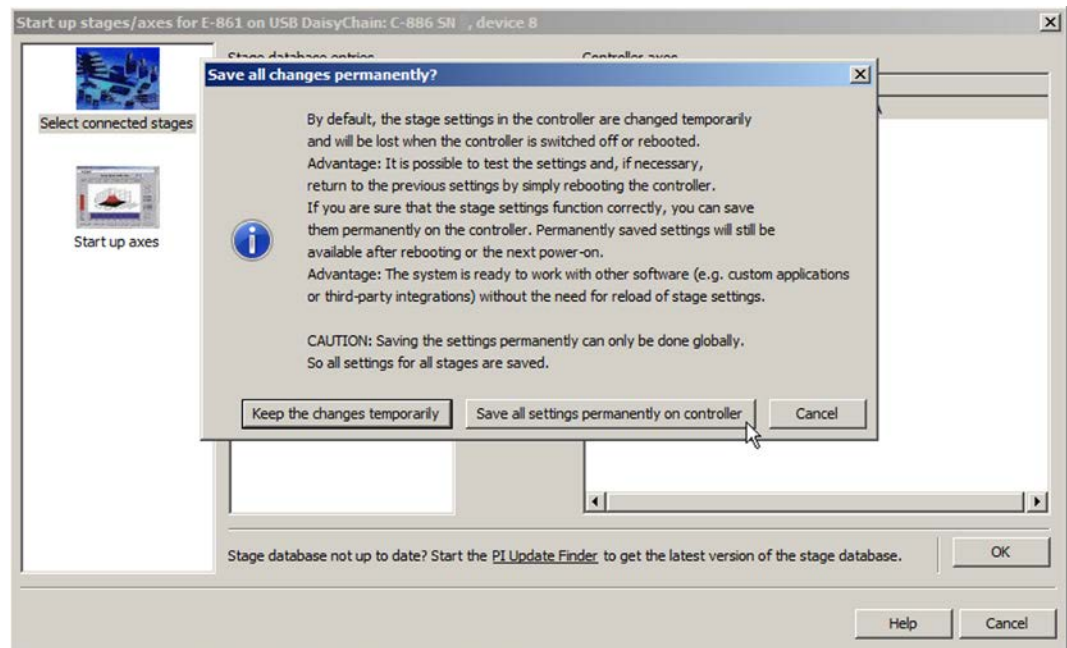
- a) Open the **Start up controller** window with the **Select connected stages** step by selecting the **Select connected stages ...** entry in the menu of the slave module in the main window.



- b) Select the matching stage type. You have two options:

- Click **Assign from ID Chip** (only present when the slave module can read ID chips).
- Mark the appropriate stage type in the **Stage database entries** list and click on **Assign**.

- c) Confirm selection with **OK** to load the parameter settings for the selected stage type from the stage database to the slave module. The **Save all changes permanently?** dialog opens.



6. In the **Save all changes permanently?** dialog box, enter how you want to load the parameter settings to the slave module:
 - Loading as default values (recommended): Click **Save all settings permanently on controller** to load the parameter settings into the nonvolatile memory of the slave module. The settings are available immediately after switch-on or reboot of the C-886 and do not need to be reloaded.
 - Loading temporarily: Click **Keep the changes temporarily** to load the parameter settings into the volatile memory of the slave module. The settings are lost when the C-886 is switched off or rebooted.

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

7. Close the **Start up controller** window by clicking **Close**.
8. Repeat steps 5 to 7 if you want to configure a further slave module for optional single axes.
9. End the direct communication with the slave modules by selecting the **Connections > Close all** menu entry in the main menu.

6.6 Starting Motion

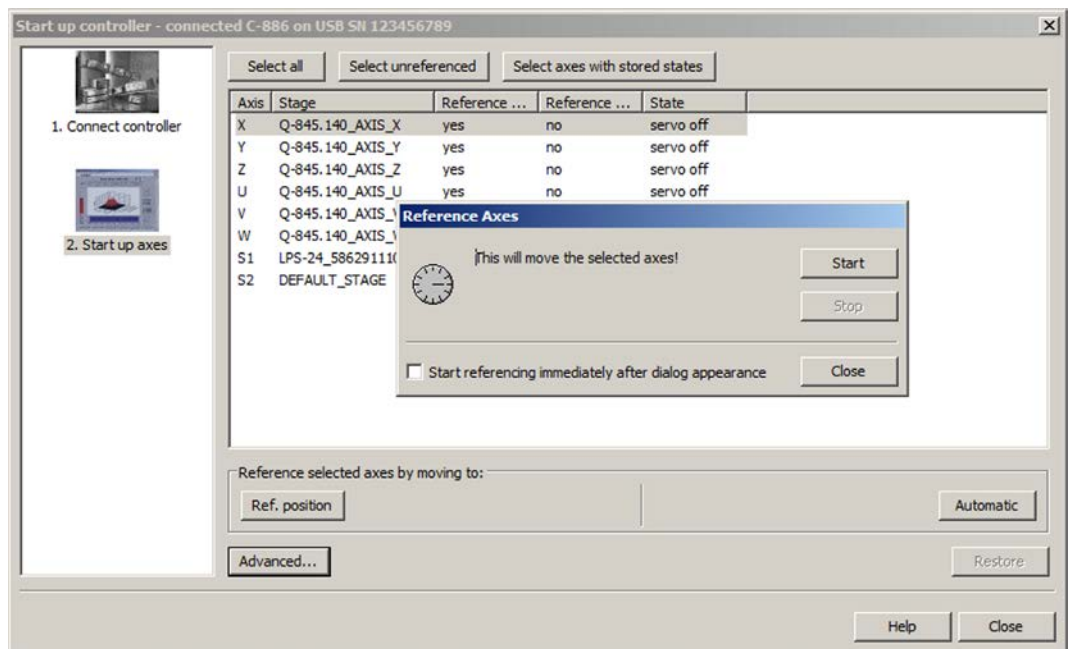
In the following, PIMikroMove is used to move the axes.

Requirements

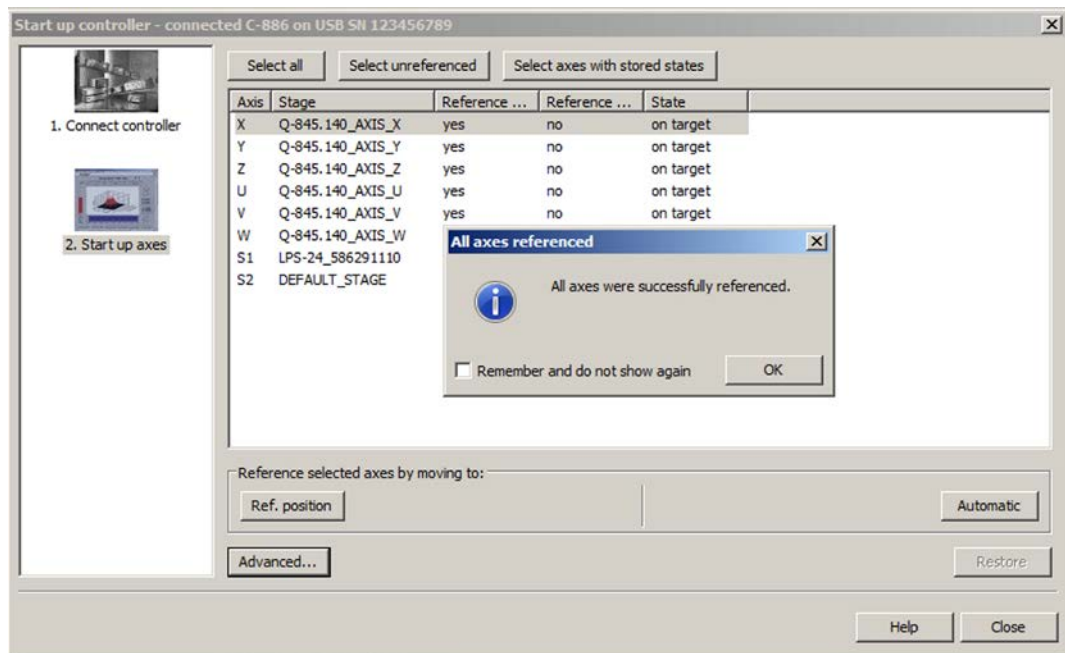
- ✓ You have read and understood the general notes on startup (p. 49).
- ✓ PIMikroMove is installed on the PC (p. 39).
- ✓ You have read and understood the PIMikroMove manual. The manual is found on the product CD.
- ✓ If necessary: You have installed (p. 43) and configured (p. 62) the slave modules for the optional single axes in the C-886.
- ✓ You have read and understood the user manuals of all connected drives (parallel-kinematic positioner, optional single axes).
- ✓ You have installed the connected drives according to the instructions in the corresponding user manuals and connected them to the C-886 (p. 45).
- ✓ You have established communication between the C-886 and the PC with PIMikroMove via the TCP/IP interface (p. 57) or the USB interface (p. 60).

Starting motion with PIMikroMove

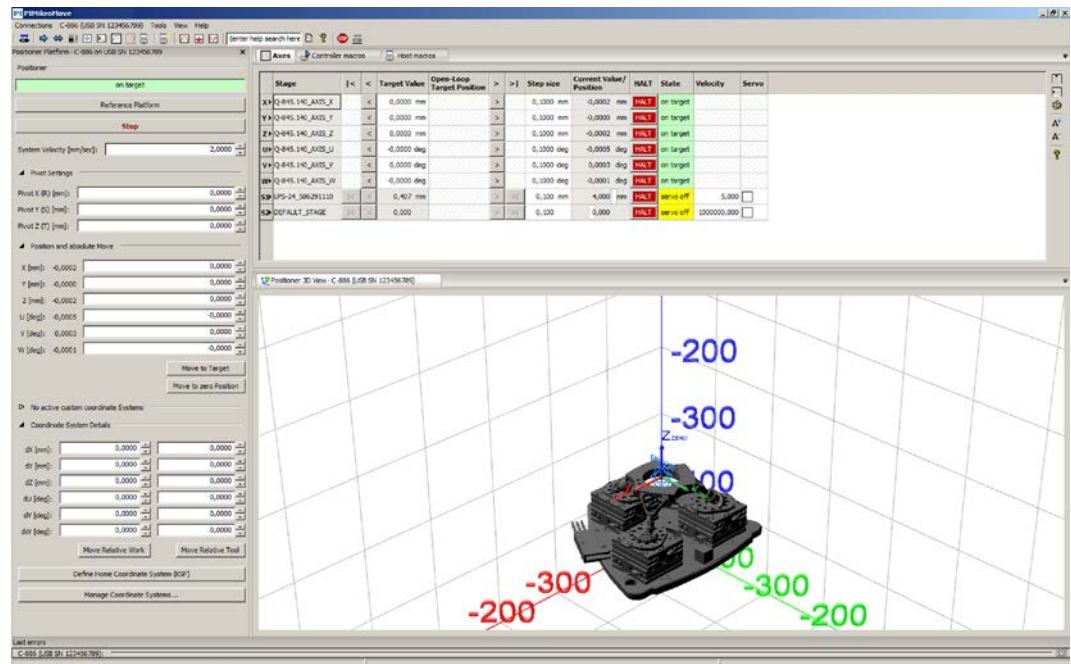
1. Perform the reference move for the axes in the **Start up axes** step (required for axes with incremental sensors). To do this, click **Ref.position** or **Automatic**.



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



3. If the **Positioner Platform** window (left, docked) and the **Positioner 3D View** card are **not** displayed in the main window of PIMikroMove:
- Display the **Positioner Platform** window with the **C-886 > Show Positioner platform settings** menu item.
 - Display the **Positioner 3-D View card** with the **C-886 > Positioner 3-D View > Show** menu item.
 - If the **Positioner 3-D View** card is not displayed in the foreground, click on the corresponding tab.



4. Test the motion of the axes of the motion platform of the positioner (X to W):
 - a) In the **Positioner Platform** window, enter a target position for at least one axis of the motion platform of the positioner in the **Position and absolute Move** area.
 - b) Click **Move to Target** to start the motion towards the given target position.
 - c) When the motion is finished, repeat steps a and b for a new target position.

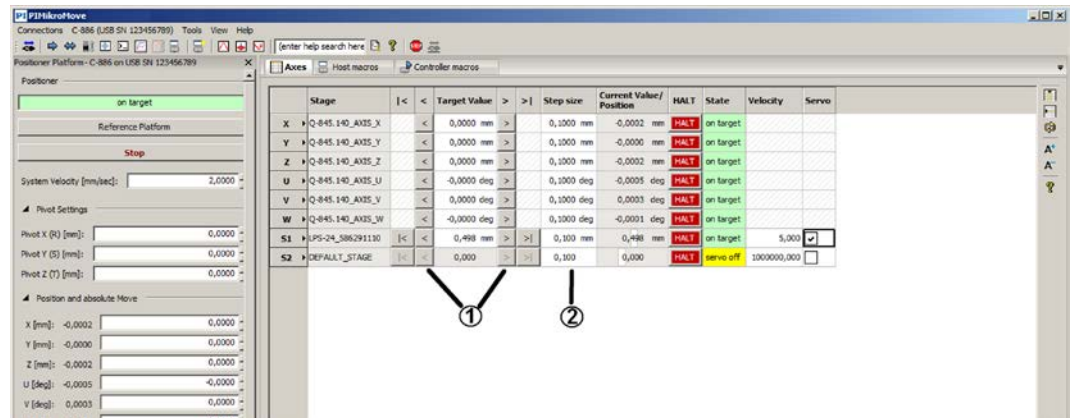
If a node of the calculated dynamics profile or the target position cannot be reached, the motion will not be executed and a window will open with an error message.

A new target position cannot be entered during a motion.

The motion is graphically displayed on the **Positioner 3-D View** card.

5. When optional single axes are present: In the main window of PIMikroMove, start motion tests of the optional single axes (S1 and S2) on the **Axes** card.
 - a) If necessary, switch on the servo mode for the optional single axes by activating the corresponding check boxes in the **Servo** column.

- b) Start test motion: You can, for example, execute steps with a particular step size (2) by clicking the corresponding arrow keys (1) for an axis.



7 Operation

In this Chapter

General Notes on Operation.....	71
Data Recorder	72
Wave Generator.....	75
Controller Macros	97

7.1 General Notes on Operation

CAUTION



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

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the C-886 in the case of malfunction or failure of the system. If touch voltages exist, touching the C-886 can result in minor injuries from electric shock.

- Connect the C-886 to a protective earth conductor (p. 42) before start-up.
- 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-886 to the protective earth conductor before starting it up again.

CAUTION



Risk of crushing by moving parts!

There is a risk of minor injuries from crushing between the moving parts of the positioner and a stationary part or obstacle.

- Keep your fingers away from areas where they can get caught by moving parts.

NOTICE**Damage from collisions!**

Collisions can damage the positioner, the load to be moved, and the surroundings.

- Make sure that no collisions are possible between the positioner, the load to be moved, and the surroundings in the workspace of the positioner.
- Do not place any objects in areas where they can be caught by moving parts.
- Stop the motion immediately if a controller malfunction occurs.

NOTICE**Damage from unintentional position changes!**

The maximum holding force when the servo mode is switched off is based on the self-locking of the drives and can be lower than the maximum load capacity when the servo mode is switched on (see manual of the positioner).

When the actual load of the positioner exceeds the maximum holding force based on the self-locking of the drives, unintentional position changes of the positioner can occur in the following cases:

- Switching off the C-886
- Rebooting the C-886
- Switching off the servo mode for the axes of the motion platform of the positioner

As a result, collisions are possible between the positioner, the load to be moved, and the surroundings. Collisions can damage the positioner, the load to be moved, or the surroundings.

- Make sure that the actual load of the motion platform of the positioner does not exceed the maximum holding force based on the self-locking of the drives before you switch off the servo mode, reboot the C-886, or switch it off.

7.2 Data Recorder

7.2.1 Data Recorder Properties

The C-886 contains a real-time data recorder for position values and status register bits.

The recorded data is temporarily stored in 36 data recorder tables with up to 8192 points each. Each data recorder table contains the data of one data source.

You can specify how the recording is to be started.

7.2.2 Setting up the Data Recorder

Reading information from the data recorder

- Send the `HDR?` command (p. 136).
The options available for recording and triggering are displayed together with the information on additional parameters and commands for data recording.
- Send the `DRC?` (p. 127) command to read the configuration of the data recorder. The data recorder tables of the C-886 record the following:
 - Data recorder table 1, 3, 5, 7, 9, 11: Commanded position of the axes of the parallel-kinematic positioner (X, Y, Z, U, V, and W)
 - Data recorder table 2, 4, 6, 8, 10, 12: Current position of the axes of the parallel-kinematic positioner (X, Y, Z, U, V, and W)
 - Data recorder table 13, 16, 19, 22, 25, 28: Commanded position of the drives of the parallel-kinematic positioner (1, 2, 3, 4, 5, and 6)
 - Data recorder table 14, 17, 20, 23, 26, 29: Current position of the drives of the parallel-kinematic positioner (1, 2, 3, 4, 5, and 6)
 - Data recorder table 15, 18, 21, 24, 27, 30: Status register for the drives of the parallel-kinematic positioner (1, 2, 3, 4, 5, and 6)
 - Data recorder table 31, 34: Commanded position of the optional single axes (S1, S2)
 - Data recorder table 32, 35: Current position of the optional single axes (S1, S2)
 - Data recorder table 33, 36: Status register for the optional single axes (S1, S2)

INFORMATION

The bits of the status register can be recorded with record option 80.

- To record the status register, use the **Data Recorder** window in PIMikroMove, which allows the targeted selection of single bits for the graphic display of the recorded data. For details, see the PIMikroMove manual

Configuring the data recorder

You can specify how the recording is to be triggered.

- Get the current trigger option with `DRT?` (p. 131).
- Change the trigger option with the `DRT` command (p. 130). The trigger option applies to all data recorder tables.

You can specify the maximum number of points that are to be recorded for each data recorder table.

- Use the `SPA` command (p. 185) to change the **Data Recorder Points Per Table** parameter, ID 0x16000201. Maximum value: 8192 points.

Querying the recording rate

- Send the `RTR?` command (p. 183) to read out the record table rate.
The record table rate indicates via a factor the frequency with which data points are recorded. The factor 1 corresponds to a frequency of 100 Hz.

7.2.3 Starting the Recording

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

Regardless of the trigger option set, the data recording is always triggered in the following cases:

- Start of a step response measurement with `STE` (p. 191)
- Start of an impulse response measurement with `IMP` (p. 142)
- Start of the wave generator output with `WGO` (p. 209)
- During wave generator output: `WGR` (p. 211) starts recording on the next output cycle

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

7.2.4 Reading Out Recorded Data

INFORMATION

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

- Read out the last recorded data with the `DRR?` command (p. 128).
The data is output in the GCS array format (see the SM146E user manual on the product CD).
- Get the number of points contained in the last recording with the `DRL?` command (p. 128).

7.3 Wave Generator

7.3.1 Functionality of the Wave Generator

The wave generators are permanently assigned to the axes of the C-886; see "Commandable Items" (p. 23).

A wave generator outputs absolute target positions for the axis motion on the basis of defined waveforms and thereby determines the dynamics profile. The wave generator output is especially suited to dynamic applications with periodic axis motion.

The following block diagram shows the integration of a wave generator in the C-886.

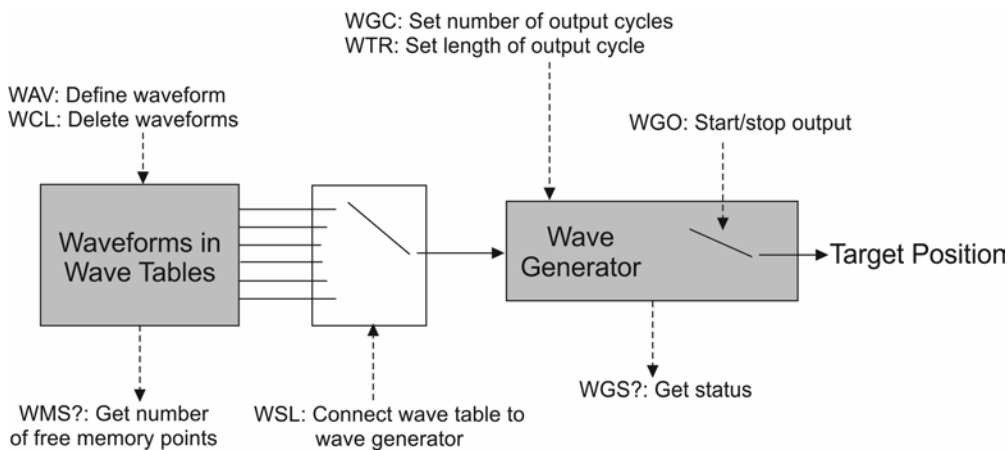


Figure 7: Block diagram of a wave generator

Waveforms can be defined and temporarily stored in up to 100 wave tables in the volatile memory of the C-886 (p. 78). Each wave table contains the data of one waveform. A total of 10,000 memory points are available for wave tables. The memory points are distributed to the individual wave tables when the waveforms are defined.

The wave tables can be assigned to the wave generators and thus the axes as desired (p. 86). A wave table can be used by several wave generators at the same time.

The number of output cycles (p. 86) and the output rate (p. 86) of the wave generator can be set.

INFORMATION

The settings for using the wave generator can only be changed in the volatile memory of the C-886 and are lost when the C-886 is switched off or rebooted.

INFORMATION

It is recommended to use the PI Frequency Generator Tool or the PI Wave Generator Tool for work with the wave generator (both available in PIMikroMove).

No command knowledge is necessary to work with PIMikroMove. Nevertheless, it is recommended to familiarize yourself with the wave generator in this chapter.

All examples in this chapter can be entered in PIMikroMove or PITerminal as command sequences.

The use of the PI Frequency Generator Tool is described in the A000T0057 technical note. The use of the PI Wave Generator Tool is described in the PIMikroMove manual (SM148E).

You can use macros to define waveforms and configure wave generators. For work with macros, see "Controller Macros" (p. 97).

7.3.2 Commands and Parameters for the Wave Generator

Commands

The following commands are available for using the wave generator:

Command	Syntax	Function
GWD?	GWD? [<StartPoint> <NumberOfPoints> [<WaveTableID>]]	Gets the content of the wave tables (i.e., the waveforms).
TWG?	TWG?	Gets the number of wave generators (= number of axes).
WAV	WAV <WaveTableID> <AppendWave> <WaveType> <WaveTypeParameters>	Defines the waveform.
WAV?	WAV? [{<WaveTableID> <WaveParameterID>}]	Gets the current length of the wave tables (number of points).
WCL	WCL {<WaveTableID>}	Deletes the contents of the wave tables.
WGC	WGC {<WaveGenID> <Cycles>}	Sets the number of output cycles.
WGC?	WGC? [{<WaveGenID>}]	Gets the number of output cycles.
WGO	WGO {<WaveGenID> <StartMode>}	Starts and stops the output of the wave generator.
WGO?	WGO? [{<WaveGenID>}]	Gets the activation state that was last commanded for the wave generator.
WGR	WGR	Starts the data recording again while the wave generator is running.

Command	Syntax	Function
WGS?	[<WaveGenID> [<InfoType>]]	Gets the status of the wave generator.
WMS?	[[<WaveTableID>]]	Gets the number of free memory points for the wave table.
WSL	WSL {<WaveGenID> <WaveTableID>}	Establishes the connection between the wave table and the wave generator.
WSL?	WSL? [{<WaveGenID>}]	Gets the connection between the wave table and the wave generator.
WTR	WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}	Sets the table rate of the wave generator (thus influencing the duration of an output cycle).
WTR?	WTR? [{<WaveGenID>}]	Gets the table rate of the wave generator.
#9	#9	Gets the current activation state of the wave generator.

Parameter

The following parameters define the characteristics of the wave generator:

Parameter	Description and Possible Values
Maximum Number Of Wave Points (ID 0x13000004)	Total number of available points for waveforms The wave tables of the C-886 have a total of 10,000 points. The available points are distributed among the wave tables when waveforms are defined with the WAV command (p. 203). This parameter is write-protected.
Number of Wave Tables (ID 0x1300010A)	Number of wave tables for saving waveforms The C-886 has 100 wave tables. This parameter is write-protected.
Start All Hexapod Wave Generators (ID 0x19002001)	Start behavior of the wave generators for the axes of the motion platform of the positioner (X, Y, Z, U, V, W): 0 = Every wave generator that is to be started must be addressed in the WGO command (default setting). 1 = Starting a wave generator starts all wave generators that are connected with a wave table. This option exist only for compatibility reasons. For the axes of the motion platform of the positioner whose wave generator has not been started, the last valid target position is always commanded.

7.3.3 Defining the Waveform

Waveforms are defined with the following steps:

- Optional: Getting information on wave tables (p. 78)
- Creating a waveform in a wave table (p. 78)
- Optional: Deleting the wave table content (p. 79)

This manual contains examples for creating waveforms (p. 79).

INFORMATION

The wave table content (= defined waveforms) is only present in the volatile memory of the C-886 and is lost when the C-886 is switched off or rebooted.

Optional: Getting information on wave tables

- Send the `SPA? 1 0x13000004` command to get the total number of points that the C-886 provides for defining waveforms in wave tables.
- Send the `SPA? 1 0x1300010A` command to get the number of wave tables available in the C-886.
- Get the current number of already defined waveform points for the wave tables with the `WAV?` command (p. 207).
- Get the current contents of the wave tables (= already defined waveforms) with the `GWD?` command (p. 134).
The response contains the contents of the wave table in the GCS array format (see separate manual for GCS array, SM 146E).
- Get the number of available memory points for the wave tables with the `WMS?` command (p. 212).

Creating a waveform in a wave table

1. Make sure that the selected wave table is **not** connected to a wave generator for which the output has been started. For details see "Configuring a Wave Generator" (p. 86) and "Stopping the Wave Generator Output" (p. 89).
2. Create the waveform in the selected wave table from single segments with the `WAV` command (p. 203) (`WAV` + max. 12 arguments). Supported wave types:
 - "PNT" (user-defined wave)
 - "SIN_P" (inverted cosine wave)
 - "RAMP" (ramp wave)
 - "LIN" (wave in the form of a single scan line)

The waveform is written to the selected wave table in the volatile memory. For details, see "Examples for creating waveforms" (p. 79).

INFORMATION

When a waveform is defined with `WAV` (p. 203), the target positions and the resulting velocities are **not** checked. The check is not performed until during the wave generator output (p. 88).

INFORMATION

The waveform influences the velocity during motion.

The velocity is limited by the following factors, among others:

- Mechanics type
 - Combination of the axes to be moved
 - Current settings for coordinate system and center of rotation
 - Amplitude of the motion
- Define the waveform so that the specifications of the connected mechanics are observed during the wave generator output.

Optional: Deleting the wave table content

1. Make sure that the selected wave table is **not** connected to a wave generator for which the output has been started. For details see "Configuring a Wave Generator" (p. 86) and "Stopping the Wave Generator Output" (p. 89).
2. Delete the content of the wave tables with the `WCL` command (p. 208).

The complete content of the selected wave table is deleted. It is **not** possible to delete the wave table content one segment at a time.

INFORMATION

When the C-886 is switched off or rebooted, the wave table content is automatically deleted.

Examples for creating waveforms

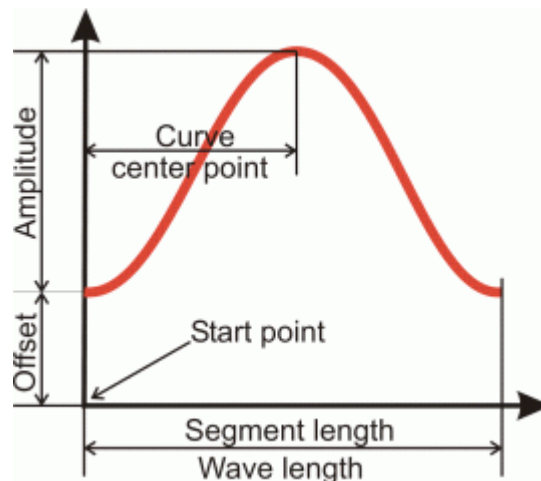
The following examples will help you to create the waveform.

Sine wave 1

- Symmetrical sine wave with offset
- Segment overwrites the contents of the wave table

Command: `WAV 2 X SIN_P 2000 0.2 0.1 2000 0 1000`

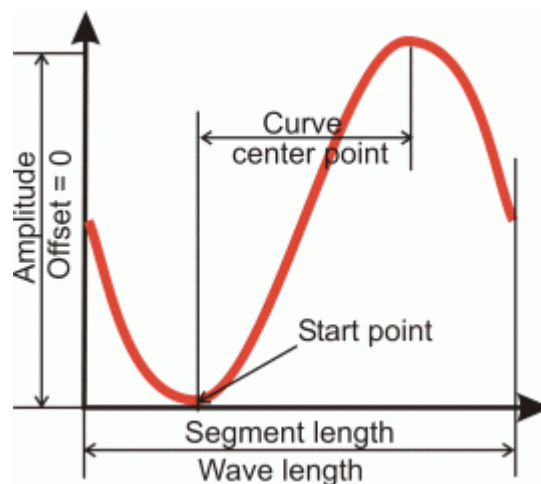
<WaveTableID> = 2
 <AppendWave> = X
 <WaveType> = SIN_P
 <SegLength> = 2000
 <Amp> = 0.2
 <Offset> = 0.1
 <WaveLength> = 2000
 <StartPoint> = 0
 <CurveCenterPoint> = 1000

**Sine wave 2**

- Symmetrical sine wave without offset
- Segment overwrites the contents of the wave table

Command: `WAV 2 X SIN_P 2000 0.3 0 2000 499 1000`

<WaveTableID> = 2
 <AppendWave> = X
 <WaveType> = SIN_P
 <SegLength> = 2000
 <Amp> = 0.3
 <Offset> = 0
 <WaveLength> = 2000
 <StartPoint> = 499
 <CurveCenterPoint> = 1000

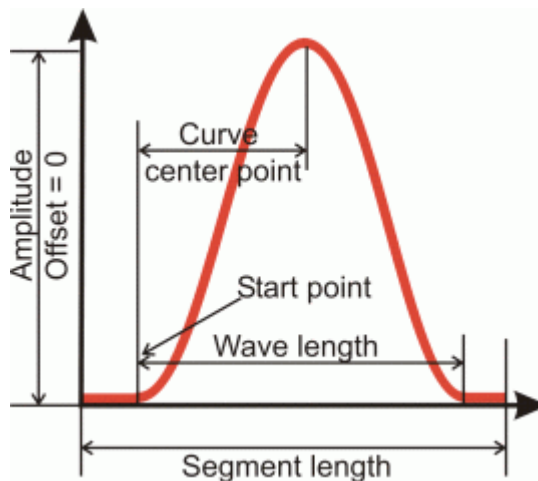


Sine wave 3

- Symmetrical sine wave without offset
- Segment is attached to the content of the wave table

Command: `WAV 2 & SIN_P 2000 0.25 0 1800 100 900`

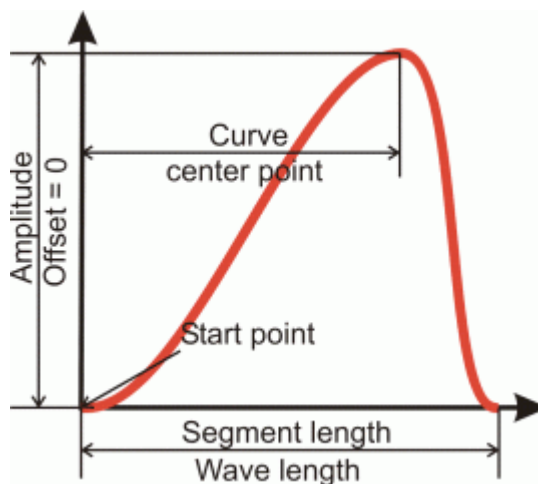
<WaveTableID> = 2
 <AppendWave> = &
 <WaveType> = SIN_P
 <SegLength> = 2000
 <Amp> = 0.25
 <Offset> = 0
 <WaveLength> = 1800
 <StartPoint> = 100
 <CurveCenterPoint> = 900

**Sine wave 4**

- Asymmetrical wave without offset
- Segment overwrites the contents of the wave table

Command: `WAV 3 X SIN_P 4000 0.2 0 4000 0 3100`

<WaveTableID> = 3
 <AppendWave> = X
 <WaveType> = SIN_P
 <SegLength> = 4000
 <Amp> = 0.2
 <Offset> = 0
 <WaveLength> = 4000
 <StartPoint> = 0
 <CurveCenterPoint> = 3100

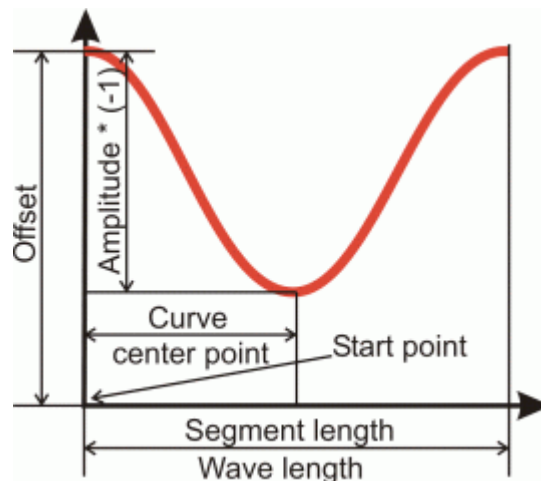


Sine wave 5

- Symmetrical wave with negative amplitude
- Segment overwrites the contents of the wave table

Command: `WAV 1 X SIN_P 1000 -0.3 0.45 1000 0 500`

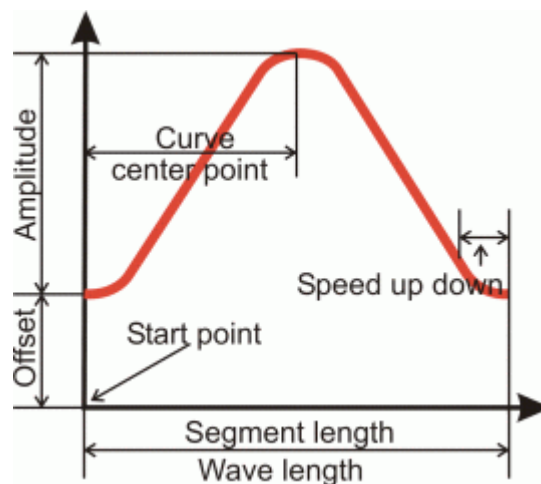
<WaveTableID> = 1
 <AppendWave> = X
 <WaveType> = SIN_P
 <SegLength> = 1000
 <Amp> = -0.3
 <Offset> = 0.45
 <WaveLength> = 1000
 <StartPoint> = 0
 <CurveCenterPoint> = 500

**Ramp wave 1**

- Symmetrical ramp wave with offset
- Segment overwrites the contents of the wave table

Command: `WAV 4 X RAMP 2000 0.2 0.1 2000 0 300 1000`

<WaveTableID> = 4
 <AppendWave> = X
 <WaveType> = RAMP
 <SegLength> = 2000
 <Amp> = 0.2
 <Offset> = 0.1
 <WaveLength> = 2000
 <StartPoint> = 0
 <SpeedUpDown> = 300
 <CurveCenterPoint> = 1000

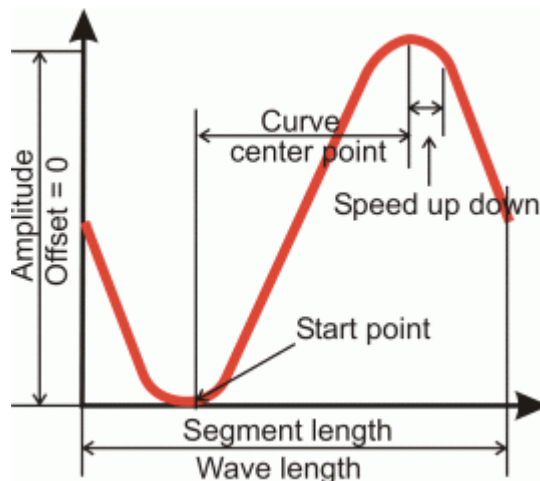


Ramp wave 2

- Symmetrical ramp wave without offset
- Segment overwrites the contents of the wave table

Command: `WAV 4 X RAMP 2000 0.35 0 2000 499 300 1000`

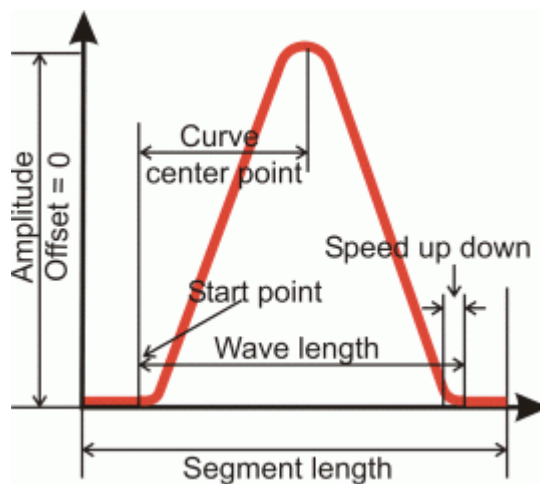
<WaveTableID> = 4
 <AppendWave> = X
 <WaveType> = RAMP
 <SegLength> = 2000
 <Amp> = 0.35
 <Offset> = 0
 <WaveLength> = 2000
 <StartPoint> = 499
 <SpeedUpDown> = 300
 <CurveCenterPoint> = 1000

**Ramp wave 3**

- Symmetrical ramp wave without offset
- Segment overwrites the contents of the wave table

Command: `WAV 5 X RAMP 2000 0.15 0 1800 120 150 900`

<WaveTableID> = 5
 <AppendWave> = X
 <WaveType> = RAMP
 <SegLength> = 2000
 <Amp> = 0.15
 <Offset> = 0
 <WaveLength> = 1800
 <StartPoint> = 120
 <SpeedUpDown> = 150
 <CurveCenterPoint> = 900

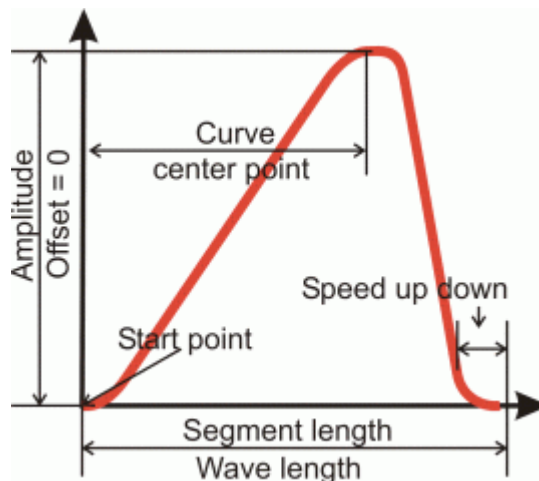


Ramp wave 4

- Asymmetrical ramp wave without offset
- Segment is attached to the content of the wave table

Command: `WAV 5 & RAMP 3000 0.35 0 3000 0 200 2250`

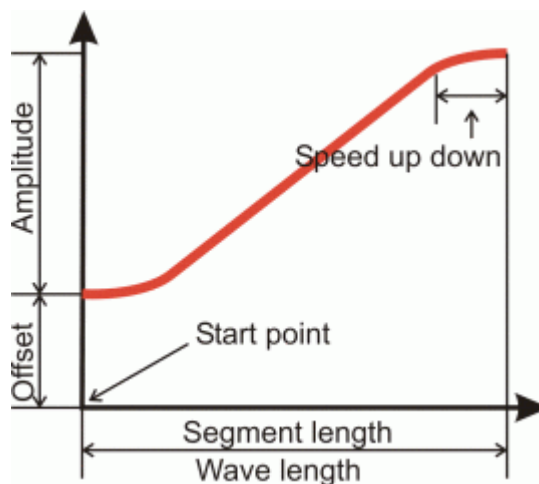
<WaveTableID> = 5
 <AppendWave> = &
 <WaveType> = RAMP
 <SegLength> = 3000
 <Amp> = 0.35
 <Offset> = 0
 <WaveLength> = 3000
 <StartPoint> = 0
 <SpeedUpDown> = 200
 <CurveCenterPoint> = 2250

**Single scan line 1**

- Scan line with offset
- Segment overwrites the contents of the wave table

Command: `WAV 1 X LIN 1500 0.3 0.15 1500 0 370`

<WaveTableID> = 1
 <AppendWave> = X
 <WaveType> = LIN
 <SegLength> = 1500
 <Amp> = 0.3
 <Offset> = 0.15
 <WaveLength> = 1500
 <StartPoint> = 0
 <SpeedUpDown> = 370

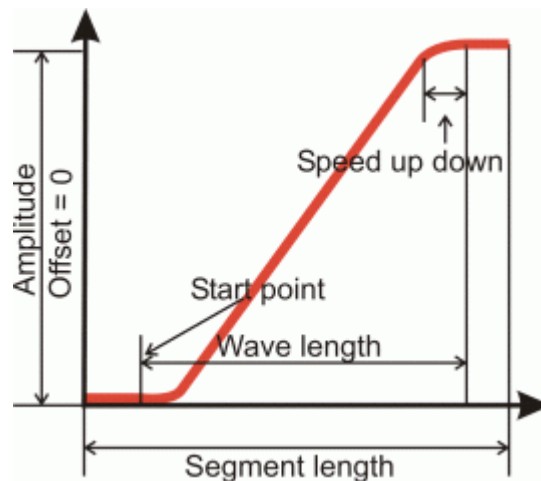


Single scan line 2

- Scan line without offset
- Segment overwrites the contents of the wave table

Command: `WAV 2 X LIN 1500 0.4 0 1100 210 180`

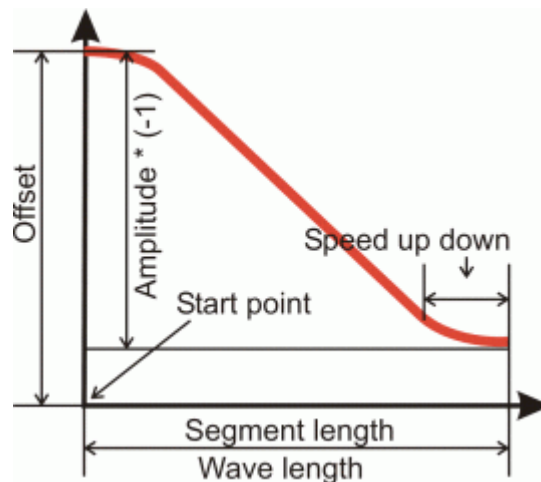
<WaveTableID> = 2
 <AppendWave> = X
 <WaveType> = LIN
 <SegLength> = 1500
 <Amp> = 0.4
 <Offset> = 0
 <WaveLength> = 1100
 <StartPoint> = 210
 <SpeedUpDown> = 180

**Single scan line 3**

- Scan line with negative amplitude
- Segment is attached to the content of the wave table

Command: `WAV 2 & LIN 3000 -0.4 0.5 3000 0 650`

<WaveTableID> = 2
 <AppendWave> = &
 <WaveType> = LIN
 <SegLength> = 3000
 <Amp> = -0.4
 <Offset> = 0.5
 <WaveLength> = 3000
 <StartPoint> = 0
 <SpeedUpDown> = 650



7.3.4 Configuring a Wave Generator

The wave generator is configured with the following steps:

- Connecting or disconnecting a wave generator and a wave table (p. 86)
- Optional: Setting the number of output cycles (p. 86)
- Optional: Setting the output rate (p. 86)

This manual contains an example for setting the output rate (p. 87).

Connecting or disconnecting a wave generator and a wave table

- Get the current connection of the wave generator and wave table with the `WSL?` command (p. 215).
- Connect or disconnect the wave generator and the wave table:
 - a) Make sure that the output has **not** been started for the selected wave generator. For details see "Stopping the wave generator output" (p. 89).
 - b) Use the `WSL` command (p. 214) to connect the selected wave table with the selected wave generator or terminate the connection of the selected generator to a wave table.

Two or more generators can be connected to the same wave table, but a generator cannot be connected to more than one wave table.

Optional: Setting the number of output cycles

The factory default setting for the number of output cycles is 0. With the default setting, the waveform is output without a time limitation until it is stopped with the `WGO` (p. 209) or `HLT` (p. 137) or `#24` (p. 121) or `STP` (p. 191) commands.

- Send the `WGC?` command (p. 209) to get the current setting for the number of output cycles of the wave generator.
- Set the number of output cycles of the wave generator with the `WGC` command (p. 208).

Optional: Setting the output rate

The individual output cycles of the waveform can be extended by adjusting the output rate. The duration of an output cycle for the waveform can be calculated as follows:

Output duration = cycle time of the C-886 * output rate * number of points

where

- the cycle time of the C-886 is specified by the 0x0E000200 parameter (in seconds)
- the output rate is the number of the C-886 cycles, for which the output of a waypoint lasts
- The number of points is the length of the waveform (i.e., the length of the wave table)

The wave points are output with interpolation as standard, in order to avoid position jumps of the axes.

- Send the `WTR?` (p. 217) command to get the current settings for output rate and interpolation.
- Set the output rate and the interpolation with the `WTR` (p. 215) command.

Different output rates can be set for the individual wave generators of the C-886.

INFORMATION

If the wave generator identifier 0 is used when setting the output rate with the `WTR` command, the output rate for all wave generators is set to the same value.

Example for setting the output rate

Action	Command	Result
Define a sine curve for wave table 2.	<code>WAV 2 X SIN_P 2000 0.2 0.1 2000 0 1000</code>	The length of the waveform and thus the number of points in the wave table is 2000.
Read cycle time of the C-886.	<code>SPA? 1 0x0E000200</code>	The cycle time of the C-886 is 10 ms
Read the current output rate.	<code>WTR?</code>	Default values for the output rate = 1 each point in the wave table is output during one C-886 cycle) Duration of an output cycle (see calculation formula above): $0.01\text{ s} \cdot 1 \cdot 2000 = 20\text{ s}$
Increase the number of C-886 cycles per point of the wave table for all wave generators to thirty times the default value.	<code>WTR 0 30 1</code>	Duration of an output cycle (see calculation formula above): $0.01\text{ s} \cdot 30 \cdot 2000 = 10\text{ min}$ Between the output of the points, the C-886 interpolates linearly to avoid position jumps.

7.3.5 Starting and Stopping Output

The wave generator outputs absolute target positions.

- Starting the wave generator output (p. 88)
- Stopping the wave generator output (p. 89)
- Optional: Getting the activation state and status of the wave generator (p. 89)
- Optional: Starting data recording during the wave generator output (p. 89)

This manual contains examples for starting/stopping the wave generator output (p. 90).

INFORMATION

When the wave generator output is active, commands for starting or configuring motion and executing corresponding macros are **not** permitted.

INFORMATION

During the wave generator output, the C-886 continuously checks whether the motion is still possible. In the following cases, the C-886 abruptly stops the motion and sets an error code:

- The target positions to be output cannot be achieved.
- The necessary velocity cannot be achieved.
- The motion would cause a collision.
- Use the `WGS?` command to get the current status of the wave generators, especially the index of the waveform points at which an error has occurred.
- Get the error code of the last error that occurred with the `ERR?` command (p. 132).

Requirements

- ✓ You have created the desired waveform (p. 78).
- ✓ You have connected the wave generator with the corresponding wave table (p. 86).

Starting the wave generator

- Start the wave generator output with the `WGO` command (p. 209).

All wave generators whose output has to be active at the same time must be started in the same command. For the axes of the motion platform of the positioner whose wave generator is not started, the last valid target position is always commanded.

The output takes place synchronously with the servo cycles of the C-886.

When the wave generator output is started, a data recording cycle automatically starts.

Stopping the wave generator output

- Stop the wave generator output by sending one of the following commands:

- **WGO** (p. 209) stops the specified wave generators
- **STP** (p. 191) stops all wave generators
- **#24** (p. 121) stops all wave generators
- **HLT** (p. 137) stops the wave generators of the specified axes

When the wave generator output is stopped by sending **STP**, **#24**, or **HLT**, the C-886 sets the error code 10 (get with the **ERR?** command (p. 132)).

When the number of output cycles has been limited (p. 86), the wave generator output is automatically stopped when the specified number of cycles is reached.

INFORMATION

Exiting the PC software does **not** stop the wave generator output.

Optional: Getting the activation state of the wave generator

- Get whether the wave generator output is running with the **#9** command (p. 120).
- Get the last-commanded start/stop settings of the wave generator with the **WGO?** command (p. 210).

The response to **WGO?** is also affected by stopping the wave generator output with **#24** (p. 121), **STP** (p. 191), or **HLT** (p. 137) and the end of a specified number of output cycles.

- Get the status of the wave generator with the **WGS?** command:
 - Is the wave generator running?
 - How many cycles has the wave generator output since the last start?
 - Error code of the last error that occurred during the output
 - Index of the waveform point at which the error occurred

Optional: Starting data recording during the wave generator output

- Start the data recording during the wave generator output by sending the command **WGR** (p. 211).

When the wave generator output is started (p. 88), an initial data recording cycle automatically starts.

The recorded data can be read out with the **DRR?** command (p. 128). For further information, see "Data Recorder" (p. 72).

Examples for starting/stopping the wave generator output

Action	Command	Result
Define a sine curve for wave table 4.	WAV 4 X SIN_P 2000 1 0 2000 0 1000	The length of the waveform and thus the number of points in the wave table is 2000.
Define a sine curve for wave table 5.	WAV 5 X SIN_P 2000 0.2 0 2000 0 1000	The length of the waveform and thus the number of points in the wave table is 2000.
Connect wave generator 1 (axis X) with wave table 4. Connect wave generator 3 (axis Z) with wave table 5.	WSL 1 4 3 5	Prerequisite for wave generator output fulfilled: No wave generator output is possible without allocation of a wave table.
Start wave generators 1 and 3.	WGO 1 1 3 1	The waveforms defined in wave tables 4 and 5 are output. The last valid target positions are commanded for the Y, U, V, and W axes of the motion platform of the positioner, whose wave generators have not been started.
Stop wave generators 1 and 3.	WGO 1 0 3 0	The output of the wave points (and therefore motion of the positioner) is stopped.

All commands except for the last two lines as in the example above; the last two lines are replaced by the following commands:

Action	Command	Result
Limit the number of output cycles for wave generators 1 and 3 to 100.	WGC 1 100 3 100	A command to stop the wave generators is not necessary.
Start wave generators 1 and 3.	WGO 1 1 3 1	The output of the wave generators ends automatically after 100 output cycles.

7.3.6 Application Tips: Loading Customer-Specific Data

You can load customer-specific data (time series) from files from the PC into the C-886 and use it as user-defined waves. Before loading, you have to convert the data into GCS array format. To load the converted data to the C-886, use the PI Wave Generator Tool or the PI Frequency Generator Tool (both available in PIMikroMove). The use of the PI Wave Generator Tool is described in the following; for the use of the PI Frequency Generator Tool, see A000T0057 technical note.

Converting data to the GCS array format

The GCS array format is based on ASCII characters. A GCS array file consists of a header section and a data table section. A comma or a period can be used as a decimal sign. For a detailed description of the GCS array format, see the separate manual (SM146E).

Convert the data of your time series to GCS array format as follows:

1. Create a text file with the file extension ".dat" on the PC, e.g., "New_GCS_Array.dat".
2. Add the header to the file. The structure and contents of the header must be as follows:

Template for the GCS array header:

```
[ GCS_ARRAY Target Positions XY ]
# VERSION = 1
# TYPE = 1
# SEPARATOR = 32
# DIM = 2
# SAMPLE_TIME = 0.001
# NDATA = 256
#
# NAME0 = TARGET POSITION X
# NAME1 = TARGET POSITION Y
#
# DISP_UNIT0 = mm
# DISP_UNIT1 = mm
# END_HEADER
```

Adapt the following information in the header to your data (**marked** in the list above):

DIM

Number of columns in the data table (i.e., number of time series), the minimum is 1

NDATA

Number of lines in the data table (i.e., number of data points per time series)

`NAME%`

Optional: Description of the column % (whereby % is a positive integer value < DIM; the counting begins at 0)

`DISP_UNIT%`

Optional: Description of the physical unit for column % (whereby % is a positive integer value < DIM; the counting begins at 0)

`SAMPLE_TIME`

Sample time of the data (in s), i.e., the time interval between two data points in the time series

`SEPARATOR`

Separator for the columns in the data table. Must be given as a decimal ASCII character (e.g., 9 for TAB or 32 for SPACE)

`[GCS_ARRAY %]`

Optional: % is the name of the data table

3. Add your data to the file according to the header information for `DIM`, `NDATA`, and `SEPARATOR`:

Data table section of the GCS array file, spaces are shown here as `SP`:

```
Value1_TimeSeries0SPValue1_TimeSeries1SP...
Value2_TimeSeries0SPValue2_TimeSeries1SP...
Value3_TimeSeries0SPValue3_TimeSeries1SP...
...
```

The following examples shows a complete GCS array with the data for two time series. The first time series, "TARGET POSITION X", contains the values 1/2/3, and the second time series, "TARGET POSITION Y", contains the values 0.1/0.2/0.3.

```
# REM data recorded with C-886 controller
#
# VERSION = 1
# TYPE = 1
# SEPARATOR = 32
# DIM = 2
# SAMPLE_TIME = 0.009009
# NDATA = 3
#
# NAME0 = TARGET POSITION X
# NAME1 = TARGET POSITION Y
#
# DISP_UNIT0 = mm
```

```
# DISP_UNIT1 = mm
# END_HEADER
1 0.1
2 0.2
3 0.3
```

The result of the conversion is a GCS array file with the file extension ".dat" that contains the time series data. The file can be imported to the C-886 with the PI Wave Generator Tool (see below) or the PI Frequency Generator Tool.

Loading GSC array data

The PI Wave Generator Tool is available in PIMikroMove. For further information on the PI Wave Generator Tool, see the PIMikroMove manual (SM148E).

1. Open the PI Wave Generator Tool from the main window of PIMikroMove via the **C-886 > Show wave generator...** menu item.
2. In the main window of the PI Wave Generator Tool, click **Load Data Set** to open the **Load Data Set for Wave Tables** window.

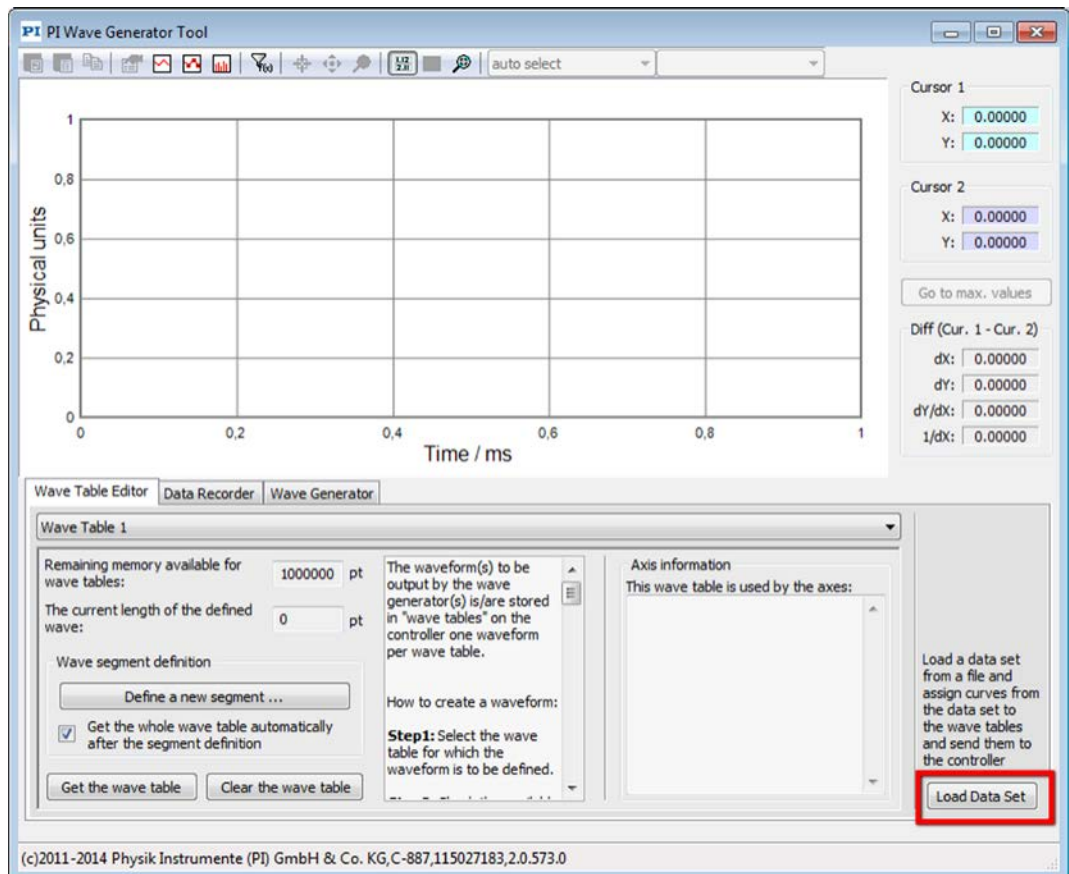


Figure 8: Main window of the PI Wave Generator Tool

3. Import the GCS array file in the **Load Data Set for Wave Tables** window.

The time series data that has been loaded from the file is shown in the graphic field. The figure below shows the data from the example in "Converting data to the GCS array format".

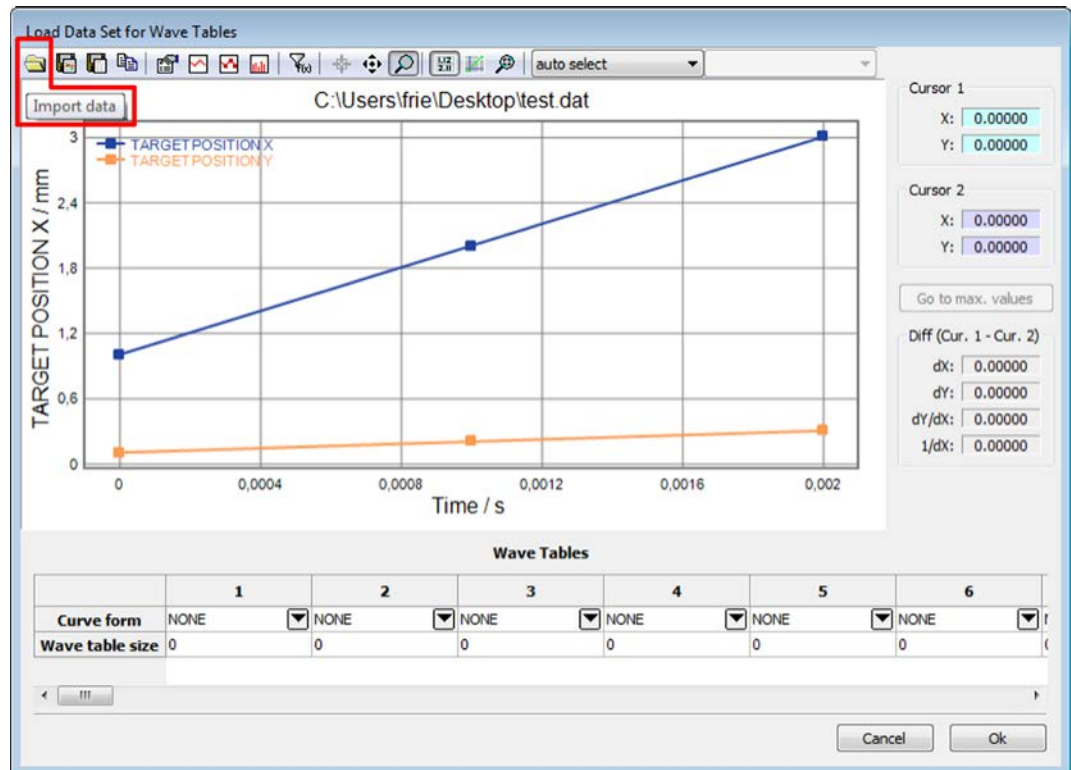


Figure 9: "Load Data Set for Wave Tables" window with graphic display of the loaded data

4. Assign the individual time series to the wave tables of the C-886. For the assignment, open the selection menu of a wave table and select the desired data (see figure below).

Note that the data is not sent to the C-886 until you click **Ok** in the lower right corner of the window.

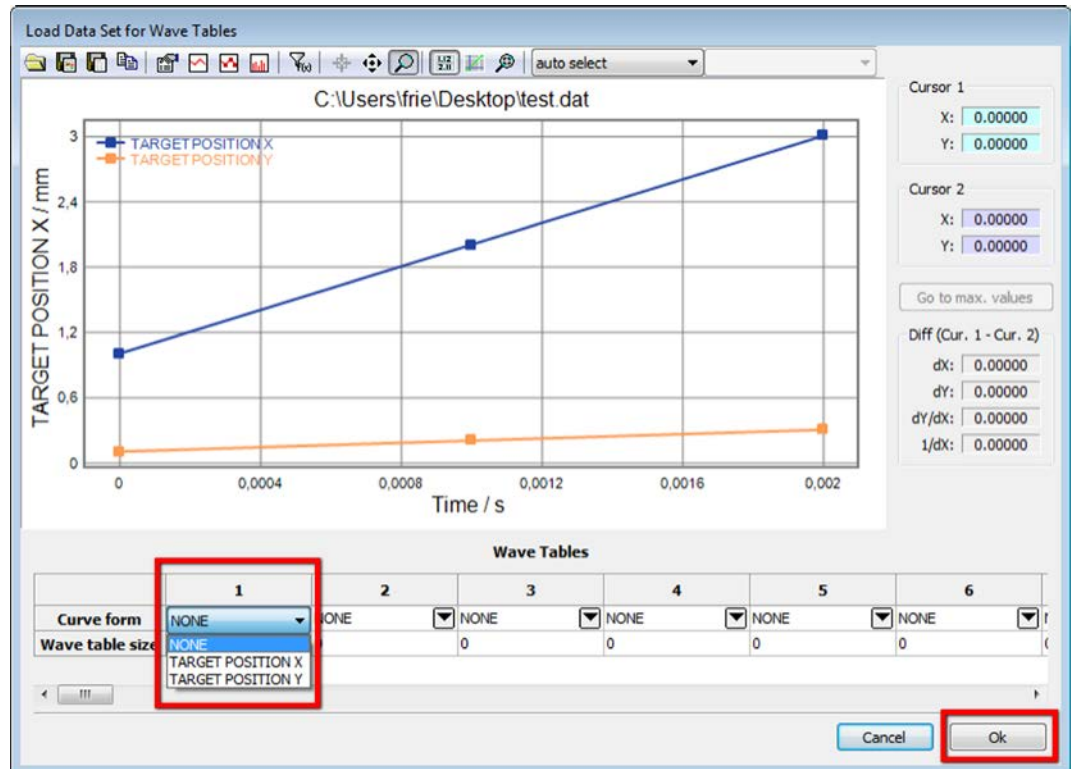


Figure 10: Assignment of the loaded data to wave tables, here for wave table 1

7.3.7 Application Tips: Using Macros for the Wave Generator

The following settings for the wave generator can only be changed in the volatile memory of the C-886 and are lost when the C-886 is switched off or rebooted:

- Contents of the wave tables (defined with WAV)
- Connection of wave tables with wave generators (set with WSL)
- Output rate (set with WTR)
- Number of output cycles (set with WGC)

You can permanently save the settings of the wave generator in the C-886 with the macro functionality of the C-886. You can also use a startup macro to configure the wave generator and start the output each time that the C-886 is switched on or rebooted.

For more information on macro functionality, see "Controller Macros" (p. 97).

In the following example, two macros are used to configure and start the wave generator output for the axes U and V (wave generators 4 and 5) of the positioner:

- The "UVdata" macro contains the definition of the waveforms for wave tables 1 and 2 (WAV commands).
- The "Start" macro executes the following actions:
 - a) Write the waveforms in wave tables 1 and 2 by calling up the "UVdata" macro.
 - b) Connect wave table 1 with wave generator 4 and wave table 2 with wave generator 5 (WSL command).
 - c) Set the output rate for wave generators 4 and 5 to 10, with linear interpolation (WTR command).
 - d) Set the number of output cycles for wave generators 4 and 5 to 100 (WGC command).
 - e) Optional but recommended for preventing axes from jumping: Command axes U and V to the starting position of the wave generator output.
 - f) Start the output for wave generators 4 and 5 - and thus the motion of axes U and V (WGO command).

Contents of the "UVdata" macro:

```

WAV 1 X PNT 1 1 0
WAV 1 & PNT 1 1 5.654625319357E-06
WAV 1 & PNT 1 1 3.091004951757E-05
[...]
WAV 1 & PNT 1 1 0.000148233661

WAV 2 X PNT 1 1 0
WAV 2 & PNT 1 1 -4.742444189387E-06
WAV 2 & PNT 1 1 -3.027352414704E-05
[...]
WAV 2 & PNT 1 1 -0.000257643502

```

Content of the "Start" macro, whereby `startposU` and `startposV` designate the starting position of the wave generator output:

```

MAC START UVdata
WSL 4 1 5 2
WTR 4 10 1 5 10 1
WGC 4 100 5 100
MOV U startposU V startposV
WGO 4 1 5 1

```

Remarks:

In the example macro "UVdata", only one wave point (= target position) is given per WAV command for a better overview. For faster processing, it is recommended to define more than one point per WAV command (max. 256 characters are permitted per command line).

Waveform definitions can be easily created by copying the contents of the log window of PIMikroMove to the macro:

1. Open the log window via the **C-886 > Log window...** menu item in the main window of PIMikroMove.
2. Open the PI Frequency Generator Tool or the PI Wave Generator Tool from the **C-886** menu in the main window of PIMikroMove.
3. Define a waveform with the tool and send the waveform to the C-886.

The sent commands are listed in the log window.

For example, you can define a sine wave with a particular frequency using the PI Frequency Generator Tool. You can then copy the corresponding WAV ... PNT commands from the log window to a macro. The WAV ... PNT commands in the log window are created by the PI_WAV_PNT() function of the PI GCS2 DLL used by PIMikroMove. The PI_WAV_PNT() function automatically distributes the wave points to the single WAV ... PNT commands.

When the example macro "Start" is used as a startup macro, a reference move can be necessary before the wave generator output is started. In this case, the following lines must be added to the macro before the `MOV U startposU V startposV` line:

```
FRF X
```

```
WAC FRF? X = 1
```

The lines start a reference move and wait until the reference move has been successfully completed.

7.4 Controller Macros

7.4.1 Overview: Macro Functionalities and Example Macros

The C-886 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-886 is switched on or rebooted.
- Processing a macro and stopping macro execution can be linked to conditions. In this way, loops can be realized as well.
- Macros can call up themselves or other macros in several nesting levels.
- Variables (p. 106) can be set for the macro and in the macro itself and used in different operations.

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

- Moving an axis back and forth (p. 101)
- Moving an axis with a variable travel range back and forth (p. 103)
- Triggering a reference move for the positioner with a startup macro (p. 105)

7.4.2 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. 122)	ADD <Variable> <FLOAT1> <FLOAT2>	Can only be used in macros. Adds two values and saves the result to a variable (p. 106).
CPY (p. 124)	CPY <Variable> <CMD?>	Can only be used in macros. Copies a command response to a variable (p. 106).
DEL (p. 125)	DEL <uint>	Can only be used in macros. Causes a delay of <uint> milliseconds.
JRC (p. 143)	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. 170)	MAC BEG <macroname>	Starts the recording of a macro with the name <i>macroname</i> on the controller. <i>macroname</i> can consist of up to 8 characters.
	MAC DEF <macroname>	Defines the given macro as the start-up macro.
	MAC DEF?	Gets the start-up macro.
	MAC DEL <macroname>	Deletes the given macro.
	MAC END	Ends the macro recording.
	MAC ERR?	Reports the last error which occurred during macro execution.
	MAC FREE?	Gets the free memory space for the macro recording
	MAC NSTART <macroname> <uint> [<String>]	Starts the given macro n times in succession (n = number of executions). The values of local variables can be set for the macro with <String>.
	MAC START <macroname> [<String>]	Starts one execution of specified macro. The values of local variables can be set for the macro with <String>.
MAC? (p. 173)	MAC? [<macroname>]	Lists all macros or the content of a given macro.

Command	Syntax	Function
MEX (p. 174)	MEX <CMD?> <OP> <value>	Can only be used in macros. Stops the macro execution depending on a condition.
RMC? (p. 182)	RMC?	Lists macros which are currently running.
VAR (p. 197)	VAR <Variable> <String>	Sets a variable (p. 106) to a certain value or deletes it.
VAR? (p. 198)	VAR? [{<Variable>}]	Gets variable values.
WAC (p. 202)	WAC <CMD?> <OP> <value>	Can only be used in macros. Waits until a condition is met.
#8 (p. 119)	-	Tests if a macro is running on the controller.

INFORMATION

A maximum of 256 characters are permitted per command line.

Parameters

The following parameter is available for working with macros:

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

7.4.3 Working with Macros

Work with macros comprises the following:

- Recording of macros (p. 100)
- Starting macro execution (p. 102)
- Stopping macro execution (p. 104)
- Setting up a start-up macro (p. 105)
- Deleting of macros (p. 106)

INFORMATION

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

Recording a macro**INFORMATION**

The C-886 can save an unlimited number of macros at the same time. A maximum of 10 nesting levels are possible in macros.

INFORMATION

Basically all GCS commands (p. 109) can be included in a macro. Exceptions:

- `RBT` for rebooting the C-886
- `MAC BEG` and `MAC END` for macro recording
- `MAC DEL` for deleting a macro
- `#27` for stopping the C-886

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 transmit any responses to interfaces.

INFORMATION

To make the use of macros more flexible, you can use local and global variables in macros. For further information, see "Variables" (p. 106).

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. 106) in macros to make macros more flexible, and give the corresponding variable values when starting macro execution.
- Contact our customer service department (p. 251) if the C-886 shows unexpected behavior.

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

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

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

3. End the macro recording.
 - If you are working with PITerminal or in the **Command entry** window of PIMikroMove: Send the `MAC END` command.
 - If you are working in PIMikroMove in 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 non-volatile memory of the C-886.

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-886 by sending the `MAC?` command.
- Get the contents of the *macroname* macro by sending the `MAC? macroname` command.

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

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

Example: Moving an axis back and forth

INFORMATION

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

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

- Record the macros by sending:

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

Starting macro execution

INFORMATION

When a macro is running on the C-886, commands can be sent to the C-886 via all communication interfaces. Commands are processed in the order that they arrive. The processing time for the individual commands depends on the command. The execution of a time-consuming command that was sent via a communication interface can therefore pause macro execution. Macros are therefore not suitable for time-critical running of defined dynamics profiles, for example.

- Do not use a macro for running a fixed, time-critical dynamics profile but consecutive MOV commands that are stored in a buffer; see "Cyclic Transfer of Target Positions" (p. 30).
- If possible, do not start or use PC software like PIMikroMove or LabVIEW driver when a macro is running.
- If possible, avoid sending commands when a macro is running.

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

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

Further information on changing parameters can be found in "Adapting Settings" (p. 233).

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

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

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

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

INFORMATION

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

The axis X 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-886 is switched on or rebooted, since they are only written to the volatile memory of the C-886.
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 X $1
WAC ONT? X = 1
MAC END
```

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

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

```
MAC NSTART movlr 5
```

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

Stopping macro execution

INFORMATION

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

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

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

Setting up a start-up macro

Any macro can be defined as the start-up macro. The start-up macro is executed each time the C-886 is switched on or rebooted.

INFORMATION

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

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

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

Example: Triggering a reference move for the positioner with a startup macro

INFORMATION

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

The STARTCL macro starts a reference move. STARTCL is defined as a startup macro so that the positioner executes the reference move immediately after switch-on.

- Send:
`MAC BEG startcl`
`DEL 1000`
`FRF X`
`MAC END`
`MAC DEF startcl`

Deleting a macro

INFORMATION

A macro cannot be deleted while it is running.

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

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

7.4.4 Variables

For more flexibility in programming, the C-886 supports the use of variables in macros. While global variables can be used for all macros, local variables are only valid for a particular macro. The number of global or local variables is not restricted.

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

Working with variables comprises the following:

- Generating variables:
 - Local variables: see "Specifics of Local Variables"
 - Global variables: within a macro with the `ADD` (p. 122), `CPY` (p. 124) and `VAR` (p. 197) commands, outside a macro with `VAR`
- Changing variable values: within a macro with the `ADD`, `CPY` and `VAR` commands
- Getting variable values: with `VAR?` within or outside a macro
- Deleting variables: with `VAR` within a macro; global variables even outside a macro

The following conventions apply to variable names:

- Variable names may not contain special characters (especially not "\$").
- The maximum number of characters is 8.
- Names of global variables can consist of characters A to Z and 0 to 9. They must start with an alphabetic character.
- Names of local variables may not contain alphabetic characters. Possible characters are 0 to 9.
- The variable name can also be given via the value of another variable.

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

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

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

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

Specifics of Local Variables

- The names of local variables used in a macro must form a continuous series. Example of permissible designation: 1, 2, 3, 4. Designation with, e.g., 1, 2, 5, 6 is not allowed.
- The values of local variables are given as arguments <String> of the `MAC START` or `MAC NSTART` command when starting the macro.

The command formats are:

```
MAC START <macroname> [{<String>}]
```

```
MAC NSTART <macroname> <uint> [{<String>}]
```

<String> stands for the value of a local variable contained in the macro. The sequence of the values in the input must correspond to the numbering of the appropriate local variables, starting with the value of the local variable 1. The individual values must be separated from each other by spaces. A maximum of 256 characters are permitted per command line. <String> can be given directly or via the value of another variable.

<uint> gives the number of times the macro is to be run. See the `MAC` command (p. 170) description for more information.

- The local variable 0 is read-only. Its value indicates the number of arguments (i.e., values of local variables) set when starting the macro.
- As long as the macro is running, the values of local variables can be queried with `VAR?`.



8 GCS Commands

In this Chapter

Notation	109
GCS Syntax for Syntax Version 2.0	109
Command Overview	111
Command Descriptions for GCS 2.0	117
Error Codes	218

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

8.2 GCS Syntax for Syntax Version 2.0

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

Command mnemonic:

CMD ::= character1 character2 character3 [?]

Exceptions:

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

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

CMD[{SP}<Argument>]LF

CMD?[{SP}<Argument>]LF

Exception:

- Single-character commands are not followed by a termination character. However, the response to a single-character command is followed by a termination character.

The argument <AxisID> is used for the logical axes of the controller. Depending on the controller, an axis identifier can consist of up to 16 characters. All alphanumeric characters and the underscore are allowed. See "Commandable Items" (p. 23) for the identifiers supported by the C-886.

Example 1:

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

Send: MOV SP1 SP10.0 LF

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

Example 2:

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

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

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

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

Example 3:

All parameters in the volatile memory are to be reset.

Send: RPA LF

Example 4:

The position of all axes is to be queried.

Send: POS? LF

The response syntax is as follows:

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

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

{["<Argument>[{"SP"<Argument>"}=""]<Value>SPLF}

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

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

Query command:

CMD?{"SP"<Arg3>{"SP"<Arg1>{"SP"<Arg2>LF

Response to this command:

<Arg3>=" "<Val3>SPLF

<Arg1>=" "<Val1>SPLF

<Arg2>=" "<Val2>LF

Example 5:

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

Receive: 2=-1158.4405SPLF

1=+0000.0000LF

INFORMATION

A maximum of 256 characters are permitted per command line.

8.3 Command Overview

Command	Format	Description
#3 (p. 117)	#3	Get Real Position
#4 (p. 117)	#4	Request Status Register
#5 (p. 118)	#5	Request Motion Status
#6 (p. 118)	#6	Query for Position Change
#7 (p. 119)	#7	Request Controller Ready Status
#8 (p. 119)	#8	Query if Macro is Running
#9 (p. 120)	#9	Get Wave Generator Status

Command	Format	Description
#11 (p. 120)	#11	Get Memory Space for Trajectory Points
#24 (p. 121)	#24	Stop All Axes
#27 (p. 121)	#27	System Abort
*IDN? (p. 121)	*IDN?	Get Device Identification
ADD (p. 122)	ADD <Variable> <FLOAT1> <FLOAT2>	Add and Save to Variable
CCL (p. 123)	CCL <Level> [<PSWD>]	Set Command Level
CCL? (p. 124)	CCL?	Get Command Level
CPY (p. 124)	CPY <Variable> <CMD?>	Copy into Variable
CST? (p. 124)	CST? [{<AxisID>}]	Get Assignment Of Stages To Axes
CSV? (p. 125)	CSV?	Get Current Syntax Version
DEL (p. 125)	DEL <uint>	Delay the Command Interpreter
DIA? (p. 125)	DIA? [{<MeasureID>}]	Get Diagnosis Information
DPA (p. 126)	DPA <Pswd> [{<ItemID> <PamID>}]	Reset Volatile Memory Settings To Default
DRC? (p. 127)	DRC? [{<RecTableID>}]	Get Data Recorder Configuration
DRL? (p. 128)	DRL? [{<RecTableID>}]	Get Number Of Recorded Points
DRR? (p. 128)	DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]	Get Recorded Data Values
DRT (p. 130)	DRT [<RecTableID> <TriggerSource> <Value>]	Set Data Recorder Trigger Source
DRT? (p. 131)	DRT? [{<RecTableID>}]	Get Data Recorder Trigger Source
ECO? (p. 131)	ECO? <String>	Echo a String
ERR? (p. 132)	ERR?	Get Error Number
FRF (p. 132)	FRF [{<AxisID>}]	Fast Reference Move To Reference Switch
FRF? (p. 134)	FRF? [{<AxisID>}]	Get Referencing Result
GWD? (p. 134)	GWD? [<StartPoint> <NumberOfPoints> [{<WaveTableID>}]]	Get Wave Table Data
HDI? (p. 135)	HDI?	Get Help For Interpretation Of DIA? Response
HDR? (p. 136)	HDR?	Get All Data Recorder Options
HLP? (p. 137)	HLP?	Get List of Available Commands

Command	Format	Description
HLT (p. 137)	HLT [{<AxisID>}]	Halt Motion Smoothly
HPA? (p. 138)	HPA?	Get List Of Available Parameters
IFC? (p. 139)	IFC? [{<InterfacePam>}]	Get Current Interface Parameters
IFS (p. 140)	IFS <Pswd> {<InterfacePam> <PamValue>}	Set Interface Parameters as Default Values
IFS? (p. 141)	IFS? [{<InterfacePam>}]	Get Interface Parameters as Default Values
IMP (p. 142)	IMP <AxisID> <Amplitude>	Start Impulse and Response Measurement
JRC (p. 143)	JRC <Jump> <CMD?> <OP> <Value>	Jump Relatively Depending on Condition
KCP (p. 144)	KCP <CSNameSource> <CSNameCopy>	Copy Coordinate System
KEN (p. 144)	KEN <CSName>	Enable Coordinate System
KEN? (p. 146)	KEN? [{<CSName>}]	Get Enabled Coordinate Systems
KET? (p. 147)	KET? [{<CSType>}]	Get Enabled Coordinate System Types
KLC? (p. 147)	KLC? [<CSName1>[<CSName2>[<Item1>[<Item2>]]]]	Get Properties Of Work-And-Tool Combinations
KLD (p. 150)	KLD <CSName> [{<AxisID> <Offset>}]	Define Leveling Coordinate System By Specifying Values Before defining a leveling coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).
KLF (p. 152)	KLF <CSName>	Define Leveling Coordinate System At Current Position Before defining a leveling coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).
KLN (p. 154)	KLN <CSName1> <CSName2>	Link Coordinate Systems
KLN? (p. 155)	KLN? [{<CSName>}]	Get Coordinate System Chains
KLS? (p. 156)	KLS? [<CSName>[<Item1>[<Item2>]]]	Get Coordinate System Properties

Command	Format	Description
KLT? (p. 158)	KLT? [<StartCS> [<EndCS>]]	Get Offsets Resulting From A Chain
KRM (p. 159)	KRM <CSName>	Remove Coordinate System
KSB (p. 159)	KSB <CSName> [{<AxisID> <Angle>}]	Define Orientational Coordinate System Before defining an orientational coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).
KSD (p. 160)	KSD <CSName> [{<AxisID> <Offset>}]	Define Operating Coordinate System By Specifying Values
KSF (p. 162)	KSF <CSName>	Define Operating Coordinate System At Current Position
KST (p. 164)	KST <CSName> [{<AxisID> <Offset>}]	Define “Tool” Operating Coordinate System
KSW (p. 166)	KSW <CSName> [{<AxisID> <Offset>}]	Define “Work” Operating Coordinate System
LIM? (p. 170)	LIM? [{<AxisID>}]	Indicate Limit Switches
MAC (p. 170)	MAC <keyword> {<parameter>}	Call Macro Function
MAC? (p. 173)	MAC? [<macroname>]	List Macros
MAN? (p. 173)	MAN? [<CMD>]	Get Help String For Command
MEX (p. 174)	MEX <CMD?> <OP> <value>	Stop Macro Execution due to Condition
MOV (p. 174)	MOV {<AxisID> <Position>}	Set Target Position
MOV? (p. 176)	MOV? [{<AxisID>}]	Get Target Position
MRT (p. 168)	MRT {<AxisID> <Distance>}	Set Target Relative In Tool Coordinate System
MRW (p. 169)	MRW {<AxisID> <Distance>}	Set Target Relative In Work Coordinate System
MVR (p. 176)	MVR {<AxisID> <Distance>}	Set Target Relative To Current Position
NLM (p. 178)	NLM {<AxisID> <LowLimit>}	Set Low Position Soft Limit
NLM? (p. 179)	NLM? [{<AxisID>}]	Get Low Position Soft Limit
ONT? (p. 179)	ONT? [{<AxisID>}]	Get On-Target State
PLM (p. 180)	PLM {<AxisID> <HighLimit>}	Set High Position Soft Limit
PLM? (p. 181)	PLM? [{<AxisID>}]	Get High Position Soft Limit
POS? (p. 181)	POS? [{<AxisID>}]	Get Real Position
PUN? (p. 182)	PUN? [{<AxisID>}]	Get Axis Unit

Command	Format	Description
RBT (p. 182)	RBT	Reboot System
RMC? (p. 182)	RMC?	List Running Macros
RON? (p. 183)	RON? [{<AxisID>}]	Get Reference Mode
RTR? (p. 183)	RTR?	Get Record Table Rate
SAI? (p. 183)	SAI? [ALL]	Get List Of Current Axis Identifiers
SCT (p. 184)	SCT "T" <CycleTime>	Set Cycle Time
SCT? (p. 185)	SCT? [<T>]	Get Cycle Time
SPA (p. 185)	SPA {<ItemID> <PamID> <PamValue>}	Set Volatile Memory Parameters
SPA? (p. 186)	SPA? [{<ItemID> <PamID>}]	Get Volatile Memory Parameters
SPI (p. 187)	SPI {<PPCoordinate> <Position>}	Set Pivot Point
SPI? (p. 188)	SPI? [{<PPCoordinate>}]	Get Pivot Point
SSL (p. 188)	SSL {<AxisID> <SoftLimitsOn>}	Set Soft Limit
SSL? (p. 189)	SSL? [{<AxisID>}]	Get Soft Limit Status
SSN? (p. 189)	SSN?	Get Device Serial Number
STA? (p. 190)	STA?	Query Status Register Value
STE (p. 191)	STE <AxisID> <Amplitude>	Start Step And Response Measurement
STP (p. 191)	STP	Stop All Axes
SVO (p. 192)	SVO {<AxisID> <ServoState>}	Set Servo Mode
SVO? (p. 192)	SVO? [{<AxisID>}]	Get Servo Mode
TMN? (p. 193)	TMN? [{<AxisID>}]	Get Minimum Commandable Position
TMX? (p. 194)	TMX? [{<AxisID>}]	Get Maximum Commandable Position
TNR? (p. 195)	TNR?	Get Number Of Record Tables
TRA? (p. 195)	TRA? {<AxisID> <Component>}	Get Maximum Commandable Position For Direction Vector
TRS? (p. 196)	TRS? [{<AxisID>}]	Indicate Reference Switch
TWG? (p. 197)	TWG?	Get Number of Wave Generators
VAR (p. 197)	VAR <Variable> <String>	Set Variable Value
VAR? (p. 198)	VAR? [{<Variable>}]	Get Variable Value
VEL (p. 198)	VEL {<AxisID> <Velocity>}	Set Closed-Loop Velocity
VEL? (p. 199)	VEL? [{<AxisID>}]	Get Closed-Loop Velocity

Command	Format	Description
VER? (p. 199)	VER?	Get Version
VLS (p. 200)	VLS <SystemVelocity>	Set System Velocity
VLS? (p. 201)	VLS?	Get System Velocity
VMO? (p. 201)	VMO? {<AxisID> <Position>}	Virtual Move
WAC (p. 202)	WAC <CMD?> <OP> <value>	Wait for Condition
WAV (p. 203)	WAV <WaveTableID> <AppendWave> <WaveType> <WaveTypeParameters>	Set Waveform Definition
WAV? (p. 207)	WAV? [{<WaveTableID> <WaveParameterID>}]	Get Waveform Definition
WCL (p. 208)	WCL {<WaveTableID>}	Clear Wave Table Data
WGC (p. 208)	WGC {<WaveGenID> <Cycles>}	Set Number Of Wave Generator Cycles
WGC? (p. 209)	WGC? [{<WaveGenID>}]	Get Number Of Wave Generator Cycles
WGO (p. 209)	WGO {<WaveGenID> <StartMode>}	Set Wave Generator Start/Stop Mode
WGO? (p. 210)	WGO? [{<WaveGenID>}]	Get Wave Generator Start/Stop Mode
WGR (p. 211)	WGR	Starts Recording In Sync With Wave Generator
WGS? (p. 211)	WGS? [<WaveGenID> [<ItemID>]]	Get Status Information Of Wave Generator
WMS? (p. 212)	WMS? [{<WaveTableID>}]	Get Maximum Number of Values for the Waveform
WPA (p. 213)	WPA <Pswd> [{<ItemID> <PamID>}]	Save Parameters To Non-Volatile Memory
WSL (p. 214)	WSL {<WaveGenID> <WaveTableID>}	Set Connection Of Wave Table To Wave Generator
WSL? (p. 215)	WSL? [{<WaveGenID>}]	Get Connection Of Wave Table To Wave Generator
WTR (p. 215)	WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}	Set Wave Generator Table Rate
WTR? (p. 217)	WTR? [{<WaveGenID>}]	Get Wave Generator Table Rate

8.4 Command Descriptions for GCS 2.0

#3 (Get Real Position)

Description: Gets the current axis position.

Format: #3 (single ASCII character number 3)

Arguments: None

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

where

<float> is the current axis position in physical units.

Notes: This command is identical in function to POS? (p. 181), but only one character has to be sent via the interface. Therefore #3 can also be used while the controller is performing time-consuming tasks.

The current position of axes X, Y, Z, U, V, and W is calculated from the measured positions of the individual drives.

The physical unit in which the axis position is given can be queried with PUN? (p. 182).

#4 (Request Status Register)

Description: Requests system status information.

Format: #4

Arguments: None

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

Notes: This command is identical in function to STA? (p. 190), but only one character has to be sent via the interface.

The response is the sum of the following codes in hexadecimal format. When analyzing the response, the following must be taken into account:

- Bits 14 and 15 for the motion status of the optional single axes S1 and S2 are only set when the motion has been triggered by a command.
- Unassigned bits have the value 0.

Bit:	23	22	21	20	19	18	17	16
	-	-	-	-	Reference move is executed	Reference move for axis S2 successful	Reference move for axis S1 successful	Reference move for positioner successful
Bit:	15	14	13	12	11	10	9	8
	Axis S2 in motion	Axis S1 in motion	Drive 6 in motion	Drive 5 in motion	Drive 4 in motion	Drive 3 in motion	Drive 2 in motion	Drive 1 in motion
Bit:	7	6	5	4	3	2	1	0
	Motion error axis S2	Motion error axis S1	Motion error drive 6	Motion error drive 5	Motion error drive 4	Motion error drive 3	Motion error drive 2	Motion error drive 1

Example:

Send: #4

Receive: 0x71804

Note: The response is given in hexadecimal format. This means: A motion error was reported for drive 3; drives 4 and 5 are in motion. The reference move of the positioner and the optional single axes S1 and S2 was completed successfully.

#5 (Request Motion Status)

Description: Requests motion state of the axes.

Format: #5

Arguments: None

Response: The response <uint> is bit-encoded and returned as the hexadecimal sum of the following codes:

1=First axis is in motion
 2=Second axis is in motion
 4=Third axis is in motion
 ...

Examples: 0 indicates motion of all axes complete
 3 indicates that the first and the second axis are in motion
 49 indicates that axes X, U, and S1 are moving.

Note: Axes 1 to 8 correspond to axes X, Y, Z, U, V, W, S1, and S2 in this order.

#6 (Query for Position Change)

Description: Queries whether the axis positions have changed since the last position query was sent.

Format:	#6 (single ASCII character number 6)
Arguments:	None
Response:	The response <uint> is bit-mapped and returned as the hexadecimal sum of the following codes: 1 = Position of the first axis has changed 2 = Position of the second axis has changed 4 = Position of the third axis has changed ...
Examples:	0 indicates that no axis position has changed. 3 indicates that the positions of the first and second axis have changed. 49 indicates that the positions of axes X, U, and S1 have changed.
Notes:	It is considered a position change when a new target position is specified with a command - even within a macro - since the last POS? (p. 181) or #3 (p. 117) was sent.

#7 (Request Controller Ready Status)

Description:	Asks controller for ready status (tests if controller is ready to perform a new command). Note: Use #5 (p. 118) instead of #7 to verify if motion has ended.
Format:	#7
Arguments:	None
Response:	B1h (ASCII character 177 = "±" in Windows) if controller is ready B0h (ASCII character 176 = "°" in Windows) if controller is not ready (e.g., performing a reference move)
Troubleshooting:	The response characters may appear differently in non-Western character sets or other operating systems.

#8 (Query if Macro Is Running)

Description:	Tests if a macro is running on the controller.
Format:	#8
Arguments:	None
Response:	<uint>=0 no macro is running <uint>=1 a macro is currently running

#9 (Get Wave Generator Status)

Description:	Requests the status of the wave generator(s).
Format:	#9
Arguments:	None
Response:	The <uint> response is bit-mapped and output as the hexadecimal sum of the following codes: 1 = Wave generator 1 is active, 2 = Wave generator 2 is active, 4 = Wave generator 3 is active, etc. "Active" = Wave generator output is running
Examples:	0 indicates that no wave generator is running 5 indicates that wave generators 1 and 3 are running

#11 (Get Memory Space for Trajectory Points)

Description:	Gets the free memory space for the points of the dynamics profile.
Format:	#11 (single ASCII character number 11)
Arguments:	None
Response:	<uint> is the free memory space, given as the number of the dynamics profile points.
Notes:	#11 gets the free memory space of a buffer that contains the dynamics profile points for the positioner. One dynamics profile point corresponds to a set of target positions for the axes of the positioner (X, Y, Z, U, V, W). The content of the buffer is only used to define the dynamics profile when the parameters 0x19001900 and 0x19001901 both have the value 1. For further information, see "Cyclic Transfer of Target Positions" (p. 30).

#24 (Stop All Axes)

Description:	Stops all axes abruptly. For further details, see the notes below.
	Sets error code to 10.
	This command is identical in function to STP (p. 191), but only one character is sent via the interface.
Format:	#24
Arguments:	None
Response:	None
Notes:	#24 stops all axes motion that is caused by motion commands or wave generator output and stops the reference move.
	#24 stops macros.
	After the axes are stopped, their target positions are set to their current positions.

#27 (System Abort)

Description:	Stops the controller.
Format:	#27 (single ASCII character number 27)
Arguments:	None
Response:	None
Notes:	#27 triggers the following: <ul style="list-style-type: none"> ▪ Drives are switched off ▪ Servo mode is switched off ▪ Commands are no longer processed ▪ Settings in the volatile memory are set to default values For a reboot, the C-886 must be switched off and back on again.

***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

Note: For C-886, *IDN? responds something like:

```
©2011-2017 Physik Instrumente (PI) GmbH & Co.
KG,C-886,117567891,2.3.3.30
```

ADD (Add and Save to Variable)

Description: Adds two values and saves the result to a variable (p. 106).

The variable is present in volatile memory (RAM) only.

Format: ADD <Variable> <FLOAT1> <FLOAT2>

Arguments: <Variable> is the name of the variable to which the result is to be saved.

<FLOAT1> is the first summand.

<FLOAT2> is the second summand.

For the summands, floating point numbers are expected. They can be given directly or via the value of a variable.

Response: None

Note: ADD can only be used in macros.

Example 1: Value \$B is added to value \$A, and the result is saved to variable C:

Send: ADD C \$A \$B

Example 2: The name of the variable to which the result is to be copied is given via the value of another variable:

Send: VAR?

Receive: A=468

B=123

3Z=WORKS

Send: ADD A\${3Z} \$A \$B

Send: VAR?

Receive: A=468

B=123

AWORKS=591

3Z=WORKS


```

Send:  ADD ${3Z} $A $B
Send:  VAR?
Receive:  A=468
         B=123
         AWORKS=591
         WORKS=591
         3Z=WORKS

```

CCL (Set Command Level)

Description: Changes the active "command level" and therefore determines the availability of commands and of write access to system parameters.

Format: CCL <Level> [<PSWD>]

Arguments: <Level> is a command level of the controller

<PSWD> is the password required for changing to the appropriate command level

The following command levels and passwords are valid:

Level = 0 is the default setting, all commands provided for "normal" users are available, read access to all parameters, no password required.

Level = 1 adds additional commands and write access to level-1 parameters (commands and parameters from level 0 are included). The required password is "advanced".

Level > 1 is provided for PI service personnel only. Users cannot change to a level > 1. Contact the customer service department if there seem to be problems with level 2 or higher parameters.

Response: None

Troubleshooting: Invalid password

Notes: HLP? (p. 137) lists all commands available in the current command level.

HPA? (p. 138) lists the parameters including the information about which command level allows write access to them. For more information about using parameters, see "Adjusting Settings" (p. 233).

After controller power-on or reboot, the active command level is always level 0.

CCL? (Get Command Level)

Description:	Get the active "command level".
Format:	CCL?
Arguments:	none
Response:	<Level> is the currently active command level; uint.
Notes:	<p><Level> should be 0 or 1.</p> <p><Level> = 0 is the default setting, all commands provided for "normal" users are available, as is read access to all parameters</p> <p><Level> = 1 adds additional commands and write access to level 1 parameters (commands and parameters from level 0 are included).</p>

CPY (Copy Into Variable)

Description:	<p>Copies a command response to a variable (p. 106).</p> <p>The variable is present in volatile memory (RAM) only.</p>
Format:	CPY <Variable> <CMD?>
Arguments:	<p><Variable> is the name of the variable to which the command response is to be copied.</p> <p><CMD?> is one query command in its usual notation. The response has to be a single value and not more.</p>
Response:	None
Note:	CPY can only be used in macros.
Example:	<p>It is possible to copy the value of one variable (e.g. SOURCE) to another variable (e.g. TARGET):</p> <p>Send: <code>CPY TARGET VAR? SOURCE</code></p>

CST? (Get Assignment Of Stages To Axes)

Description:	Gets the name of the stage type that is connected to the given axis.
Format:	CST? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller

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

where

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

Notes: The corresponding stage type is automatically assigned to the axes of the positioner when the controller is switched on or rebooted.

The assignment can be changed for the optional single axes S1 and S2; see "Configuring Slave Modules for Single Axes" (p. 62).

CSV? (Get Current Syntax Version)

Description: Gets the GCS syntax version used in the firmware.

Format: CSV?

Arguments: None

Response: The current GCS syntax version

Note: The only possible response is 2.0 (for GCS 2.0).

DEL (Delay the Command Interpreter)

Description: Delays <uint> milliseconds.

Format: DEL <uint>

Arguments: <uint> is the delay value in milliseconds.

Response: None

Note: DEL should only be used in macros.

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.

Notes: Use the response to the HDI? command (p. 135) to obtain descriptions and physical units of the supported measurands.

C-886 supports the following measurands:

<MeasureID>	<Description> (get with HDI?)
1	Hexapod Powered: Without meaning for the C-886
2	Controller E-Stop Activated: Without meaning for the C-886
3	Temperature of Controller: Without meaning for the C-886
4	Index of Faulty Point in Waveform: Index of the waveform point at which the last error occurred during the wave generator output

DPA (Reset Settings to Default)

Description: Resets parameter values and parameter-independent settings to the default settings.

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

Arguments: <Pswd> is the password for resetting the memory. See below for details.

<ItemID> is the item for which a parameter is to be reset. See below for details.

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

Response: None

Troubleshooting: Illegal item identifier, wrong parameter ID, invalid password

Notes:	<p>DPA resets the parameter values and the settings for the coordinate systems stored in the volatile memory of the C-886 to the default settings.</p> <p>For the C-886, DPA is not necessary for the specification of <ItemID> and <PamID>.</p> <p>The default settings loaded using DPA are independent of the settings in the nonvolatile memory, which can be overwritten using WPA (p. 213). The settings in the nonvolatile memory (saved using WPA) are automatically loaded to the volatile memory when the C-886 is switched on or rebooted.</p>	
Valid passwords:	100	Resets the values of all parameters and the settings for coordinate systems to the default settings (for details, see password SKS)
	SKS	<p>Default settings which are restored using DPA SKS:</p> <ul style="list-style-type: none"> ▪ Orientational coordinate system (KSB() type): PI_BASE is enabled ▪ Leveling coordinate system (KLD() or KLF() type): PI_Levelling is enabled ▪ ZERO operating coordinate system is enabled and based on PI_BASE and PI_Levelling ▪ Pivot point (see SPI (p. 187)), soft limits of the axes (see NLM (p. 178), PLM (p. 180), and SSL (p. 188))

DRC? (Get Data Recorder Configuration)

Description:	Gets the settings for the data to be recorded.
Format:	DRC? [{<RecTableID>}]
Arguments:	<RecTableID>: is a data recorder table of the controller; if this entry is omitted, the response will contain the settings for all tables.

Response: The current DRC settings:

```
{<RecTableID>="<Source> <RecOption> LF}
```

where

<Source>: is the data source, for example, an axis or a channel of the controller. The source type depends on the record option.

<RecOption>: is the type of data to be recorded (record option).

The available record options can be queried with HDR? (p. 136).

DRL? (Get Number of Recorded Points)

Description: Reads the number of points comprised by the last recording.

Format: DRL? [{<RecTableID>}]

Arguments: <RecTableID> is one data recorder table of the controller

Response: {<RecTableID>="<uint> LF}

where

<uint> gives the number of points recorded with the last recording

DRR? (Get Recorded Data Values)

Description: Gets the last recorded data.

Reading can take some time depending on the number of points to be read!

It is possible to read the data while recording is still in progress.

Format: DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]

Arguments: <StartPoint> is the first point to be read from the data recorder table; it starts with index 1.

<NumberOfPoints> is the number of points to be read per table.

<RecTableID> is one data recorder table of the controller.

Response: For the recorded data in GCS array format, see the separate manual for the GCS array, SM146E, and the example below.

Notes:

If the value -1 is set for <NumberOfPoints>, all valid points of the selected tables will be read out

If <RecTableID> is omitted, all tables are read out.

With HDR? (p. 136), you will obtain a list of all available record and trigger options as well as information on additional parameters and commands for data recording.

For further information, see "Data Recorder" (p. 72).

Example:

```
drt 1 1 0
mov x 2
drr? 1 5 1 2
# REM data recorded with C-886 controller
#
# TYPE = 1
# SEPARATOR = 32
# DIM = 2
# NDATA = 5
# SAMPLE_TIME = 0.009009
#
# NAME0 = TARGET POSITION X
# NAME1 = REAL POSITION X
# DISP_UNIT0 = mm
# DISP_UNIT1 = mm
# RATIO_NOM0 =1
# RATIO_DENOM0 = 1
# RATIO_NOM1 =1
# RATIO_DENOM1 = 1
# END_HEADER
0.004220971838 -0.000364157866
0.064530499279 0.036015000194
0.145174950361 0.140960231423
0.232607111335 0.216114446521
0.309781700373 0.30354321003
```

DRT (p. 130) is used to determine that a recording is to be triggered by the next motion command, e.g. by MOV (p. 174).

MOV triggers the motion of axis X to position 2.

DRR? is used to get the first five points of data recorder tables 1 and 2.

DRT (Set Data Recorder Trigger Source)

Description:	Defines a trigger source for the given data recorder table.
Format:	DRT <RecTableID> <TriggerSource> <Value>
Arguments:	<p><RecTableID> is one data recorder table of the controller. See below for details.</p> <p><TriggerSource> ID of the trigger source, see below for a list of available options.</p> <p><Value> depends on the trigger source, can be a dummy, see below.</p>
Response:	none
Notes:	<p>Regardless of the data recorder table selected with <RecTableID>, the trigger option selected with <Triggersource> is always set for all data recorder tables.</p> <p>With HDR? (p. 136), you will obtain a list of all available recording and triggering options as well as additional information on data recording.</p> <p>IMP (p. 142), STE (p. 191), WGO (p. 209), and WGR (p. 211) each trigger a recording by the data recorder regardless of the DRT settings (WGR only when the wave generator is active).</p> <p>The recording ends when the maximum number of points is reached (specified by the Data Recorder Points Per Table parameter, ID 0x16000201).</p> <p>For more information see "Data Recording" (p. 72).</p>
Available trigger options:	<p>0 = No trigger. Exception: STE, IMP, WGO, and WGR always trigger data recording; Data recording is not triggered, and data recording in progress is continued until the maximum number of points is reached. Exception: STE, IMP, WGO, and WGR always trigger a recording. <Value> must be a dummy.</p> <p>1 = Trigger with next command that changes the position; e.g. MOV (p. 174), MVR (p. 176); <Value> must be a dummy. Default setting</p> <p>2 = Trigger with next command; sets trigger to the trigger option 0 after execution; <Value> must be a dummy.</p>

4 = Trigger immediately;
sets trigger to the trigger option 0 after execution; <Value> must be a dummy.

6 = Trigger with next command that changes the position, e.g., MOV, MVR;
sets trigger to the trigger option 0 after execution; <Value> must be a dummy.

If trigger option 2, 4, or 6 is set, then trigger option 0 is automatically activated after the recording is triggered. As a result, the recording continues until the maximum number of points of the data recorder table(s) is reached. In this way, several motion can be recorded in succession. Trigger options 2, 4, or 6 are a good idea, e.g., when a dynamics profile is to be defined by consecutive MOV commands (see MOV).

DRT? (Get Data Recorder Trigger Source)

Description: Gets the trigger source for the data recorder tables.

Format: DRT? [{<RecTableID>}]

Arguments: <RecTableID> is one data recorder table of the controller.

Response: {<RecTableID>="<TriggerSource> <Value> LF}

where

<TriggerSource> is the identifier of the trigger source.

<Value> depends on the trigger source.

Further information can be found in the description of the DRT command (p. 130).

ECO? (Echo a String)

Description: Returns a string.

ECO? can be used to test the communication.

Format: ECO? <String>

Arguments:	<String> is any given combination of characters consisting of letters and numbers
Response:	<String> LF
Note:	<String> can consist of up to 100 characters.

ERR? (Get Error Number)

Description:	<p>Get error code <int> of the last occurred error and reset the error to 0.</p> <p>Only the last error is buffered. You should therefore call ERR? after each command.</p> <p>The error codes and their descriptions are listed in "Error Codes" (p. 218).</p>
Format:	ERR?
Arguments:	None
Response:	The error code of the last error that occurred (integer).
Troubleshooting:	Communication breakdown
Notes:	<p>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.</p> <ul style="list-style-type: none"> ➤ If possible, access the controller with one instance only. ➤ If incorrect system behavior does not cause the controller to send an error code, check whether the error code is queried regularly in the background by a macro, script or the PC software (e.g., PIMikroMove).

FRF (Fast Reference Move To Reference Switch)

Description:	<p>Starts a reference move.</p> <p>Moves the given axis to the reference point switch and sets the current position to a defined value. See below for details.</p> <p>If multiple axes are given in the command, they are moved synchronously.</p>
--------------	--

Format:	FRF [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller, if omitted, all axes are involved.
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	For axes with incremental sensors, motion can only be commanded after a successful reference move (also referred to as "initialization").

The behavior of the positioner after the reference move is determined by the **Behaviour After Reference Move** (ID 0x07030401) and **Target For Motion After Reference Move** parameters (ID 0x07030402). Depending on the parameter values, the platform can be moved automatically to a specified position, e.g., after the reference move.

- Value of the parameter 0x07030401 = 0: Motion platform remains in the reference position after the reference move.
- Value of parameter 0x07030401 = 1: After the reference move, the motion platform moves to the target position which is given by parameter 0x07030402.

No reference move is required for axes with absolute-measuring sensors. The use of the FRF command is still recommended for these axes, however. FRF does not start a reference move for axes with absolute-measuring sensors but sets the target positions to the current position values. The above-described parameter values also go into effect, so that the axes can be moved to a defined "initial position", for example.

During a reference move, the positioner moves unpredictably.

A common reference move is always performed for the axes of the motion platform of the positioner (X, Y, Z, U, V, W). It is therefore always sufficient to enter a single axis to start the reference move of the motion platform, e.g.:

FRF X

The servo mode is switched on for the axes of the motion platform of the positioner (X, Y, Z, U, V, W) after FRF has been successfully used.

FRF can be aborted by #24 (p. 121), STP (p. 191) and HLT (p. 137).

Use FRF? (p. 134) to check whether the reference move was successful.

FRF? (Get Referencing Result)

Description:	Gets whether the given axis is referenced or not.
Format:	FRF? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.
Response:	{<AxisID>="<uint> LF}
	where
	<uint> indicates whether the axis has been successfully referenced (=1) or not (=0).
Troubleshooting:	Illegal axis identifier
Note:	Axes are considered to be "referenced" when a reference move has been successfully completed with FRF (p. 132) or when the axes are equipped with absolute-measuring sensors.

GWD? (Get Wave Table Data)

Description:	Queries waveform for given wave table.
Format:	GWD? [<StartPoint> <NumberOfPoints> [{<WaveTableID>}]]
Arguments:	<StartPoint> is the start point in the wave table, begins with index 1
	<NumberOfPoints> is the number of points to be read per table
	<WaveTableID> is one wave table of the controller
Response:	The wave table contents (waveform) in GCS array format (see the separate manual for the GCS array, SM 146E, and the example below)
Notes:	<p>Depending on the waveform definition with WAV (p. 203), the wave tables can have different lengths.</p> <ul style="list-style-type: none"> ➤ Only get wave tables of the same length in the same command. ➤ Get the contents of wave tables of different lengths with separate commands. <p>If wave tables of different lengths are queried together, the response will contain a maximum of as many points as the shortest wave table or the query will produce an error.</p>

Example:

```
gwd? 1 10 1 2
# TYPE = 1
#
# SEPARATOR = 32
# DIM = 2
# SAMPLE_TIME = 0.009000
# NDATA = 10
# NAME0 = Wave table 1
# NAME1 = Wave table 2
# END_HEADER
-9.7773e-08 -9.7773e-08
9.9005486e-05 9.9005486e-05
0.000396286628 0.000396286628
0.000891734005 0.000891734005
0.001585328104 0.001585328104
0.002477041554 0.002477041554
0.00356683912 0.00356683912
0.004854677712 0.004854677712
0.006340506381 0.006340506381
0.008024266324 0.008024266324
```

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. 125).

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 concerned with data recording).

Format: HDR?

Arguments: None

Response #RecordOptions
 {<RecOption>=" "<DescriptionString>[of <Channel>]}

#TriggerOptions
 [{<TriggerOption>=" "<DescriptionString>}]

#Parameters to be set with SPA
 [{<ParameterID>=" "<DescriptionString>}]

#Additional information
 [{<Command description> "("<Command>")"}]

#Sources for Record Options
 [{<RecOption>=" "<Source>}]

end of help

Example:

```
hdr?
#RecordOptions
0=Nothing is recorded
1=Commanded position of axis
2=Real position of axis
80=Status register of axis
#TriggerOptions
0=No trigger. Exception: STE and IMP always
trigger data recorder
1=Trigger with next command that changes the
position, default setting
2=Trigger with next command, resets trigger
settings to 0
4=Trigger immediately, resets trigger settings
to 0
6=Trigger with next command that changes the
position, resets trigger settings to 0
#Additional information
Get data recorder configuration with DRC?
Get number of recorded points with DRL?
Get recorded data values with DRR?
Set data recorder trigger source with DRT
```

```

Get data recorder trigger source with DRT?
Tell number of data recorder tables with TNR?
#Sources for Record Options
0 = X Y Z U V W 1 2 3 4 5 6 S1 S2
1 = X Y Z U V W 1 2 3 4 5 6 S1 S2
2 = X Y Z U V W 1 2 3 4 5 6 S1 S2
80 = 1 2 3 4 5 6 S1 S2
end of help

```

Data sources 1 to 6 are the drives of the positioner.

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
Note:	The HLP? response contains the commands provided by the current command level. See CCL (p. 123) for more information.

HLT (Halt Motion Smoothly)

Description:	Halts the motion of given axes smoothly. For further details, see the notes below.
	Error code 10 is set.
	#24 (p. 121) and STP (p. 191) in contrast abort current motion as fast as possible for the controller without taking care of maximum velocity and acceleration.
Format:	HLT [{<AxisID>}]
Arguments:	<AxisID>: is one axis of the controller, if omitted all axes are halted
Response:	none
Troubleshooting:	Illegal axis identifier

Notes: HLT stops all axes motion that is caused by motion commands or wave generator output and stops the reference move.

After the axes are stopped, their target positions are set to their current positions.

HLT does not stop macros.

HPA? (Get List Of Available Parameters)

Description: Responds with a help string which contains all available parameters with short descriptions. For further information, see "Parameter Overview" (p. 236).

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.

Notes: The string has the 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 that the parameter affects. (The meaning of "item" depends on the parameter; this can refer to an axis, a drive, or the entire system.)

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

<FunctionGroupDescription> is the name of the function group to which the parameter belongs. (Parameters are grouped according to their purpose in order to clarify their interrelation.)

<ParameterDescription> is the parameter name

<PossibleValue> is one value from the allowed data range

<ValueDescription> is the meaning of the corresponding value

The parameter listing varies depending on the optional accessories that are installed.

The values of the listed parameters can be changed with SPA (p. 185). SPA affects the parameter settings in volatile memory (RAM).

IFC? (Get Current Interface Parameters)

Description: Gets 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> indicates the value of the interface parameter from volatile memory.

<InterfacePam> can be IPADR, IPSTART, IPMASK, IPMAXCONN, MACADR, or TERMSTR.

For <InterfacePam> = IPSTART, <PamValue> indicates the current setting of the startup behavior for the configuration of the IP address for TCP/IP communication:

0 = The IP address defined with IPADR is used.

1 = DHCP is used to obtain the IP address

For <InterfacePam> = IPADR, the first four parts of <PamValue> indicate the IP address which is currently used for TCP/IP communication; the last part indicates the port.

For <InterfacePam> = IPMASK, <PamValue> indicates the IP mask setting that is currently used for TCP/IP communication, in the format uint.uint.uint.uint.

For <InterfacePam> = IPMAXCONN, <PamValue> indicates the maximum permissible number of connections for TCP/IP communication.

For <InterfacePam> = MACADR, the <PamValue> indicates the unchangeable, unique address of the network hardware in the C-886.

For <InterfacePam> = TERMSTR, <PamValue> returns the termination character for the commands of the GCS:

0 = LineFeed (ASCII character 10)

IFS (Set Interface Parameters as Default Values)

Description: Stores interface parameters.

The default parameters for the interface are changed in the nonvolatile memory, but the currently active parameters are not. Settings made with IFS become active with the next switching on or reboot.

Format: IFS <Pswd> {<InterfacePam> <PamValue>}

Arguments: <Pswd> is the password for writing to the nonvolatile memory, default is "100"

<InterfacePam> is the interface parameter to be changed, see below

<PamValue> gives the value of the interface parameter, see below

Response: None

The following interface parameters can be set:

IPADR

The first four parts of <PamValue> specify the default IP address for TCP/IP communication, the last part specifies the default port to be used, default is 192.168.1.28:50000;

Note: While the IP address can be changed, the port must always be 50000!

IPSTART

<PamValue> defines the startup behavior for configuration of the IP address for TCP/IP communication,

0 = The IP address defined with IPADR is used

1 = DHCP is used to obtain the IP address. (default);

IPMASK

<PamValue> indicates the subnet mask to be used for TCP/IP communication, in the format uint.uint.uint.uint, default is 255.255.255.0

Response: None

Notes: **Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.**

Further interface parameters of the C-886 are write-protected: The default settings of these parameters can be queried with IFS? (p. 141).

For more information, see "Establishing Communication via the TCP/IP Interface" (p. 53).

IFS? (Get Interface Parameters as Default Values)

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

Format: IFS? [{<InterfacePam>}]

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

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

where

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

<InterfacePam> can be IPADR, IPSTART, IPMASK, IPMAXCONN, MACADR, and TERMSTR

The following interface parameters are write-protected:

For <InterfacePam> = IPMAXCONN, <PamValue> indicates the maximum permissible number of connections for TCP/IP communication.

For <InterfacePam> = MACADR, <PamValue> returns the unique address of the network hardware in the C-886.

For <InterfacePam> = TERMSTR, <PamValue> returns the termination character for the commands of the GCS:
0 = LineFeed (ASCII character 10)

For all other forms of <InterfacePam>, see IFS (p. 140).

IMP (Start Impulse and Response Measurement)

Description:	Starts performing an impulse and recording the impulse response for the given axis.
	The data recorder configuration, i.e., the assignment of data sources and record options to the recorder tables, can be set with DRC.
	The recorded data can be read with the DRR? command (p. 128).
Format:	IMP <AxisID> <Amplitude>
Arguments:	<AxisID> is one axis of the controller
	<Amplitude> is the height of the impulse. See below for details.
Response:	None
Troubleshooting:	The target position resulting from the specified impulse height is out of limits.

- Notes:
- An "impulse" consists of a relative motion with the given amplitude, followed by an equally large motion in the opposite direction. The impulse is executed in relation to the current position.
 - The pulse width of the impulse results from the value of the **Pulse Length Factor** parameter (ID 0x0E000900) multiplied by the C-886 cycle time.
 - The physical unit in which <Amplitude> is to be given can be queried with PUN? (p. 182).
 - For the axes of the positioner (X, Y, Z, U, V, W), the following is valid:
 - Before the start of every motion, it is checked whether the motion platform can actually reach the commanded target position. With VMO? (p. 201) you can query whether the target position can be reached.
 - Rotations take place around the center of rotation.
 - When the **Trajectory Source** parameter (ID 0x19001900) is set to 1, the dynamics profile must be defined by consecutive MOV commands. IMP is not allowed.
 - For more information, see "Motion of the positioner" (p. 27).

JRC (Jump Relatively Depending On Condition)

- Description: Jumps relatively depending on a given condition of the following type: one given value is compared with a queried value according to a given rule.
- Can only be used in macros.
- Format: JRC <Jump> <CMD?> <OP> <Value>
- Arguments: <Jump> is the size of the relative jump. -1 means the macro execution pointer jumps back to the previous line, 0 means the command is executed again, which is the same behavior as with WAC (p. 202). 1 jumps to the next line, making the command unnecessary, and 2 jumps over the next command. Only jumps within the current macro are allowed.
- <CMD?> is one query command in its usual notation. The response has to be a single value and not more. For an example see below.
- <OP> is the operator to be used. The following operators are possible:
 = <= < > >= !=
 Important: There must be a blank space before and after the operator!
- <Value> is the value to be compared with the response to <CMD?>.

Response: None

Troubleshooting: Check proper jump target

KCP (Copy Coordinate System)

Description: Generates a copy of a coordinate system.

Format: KCP <CSNameSource> <CSNameCopy>

Arguments: <CSNameSource> is the name of the coordinate system of which a copy is to be generated. PI_Base, PI_Levelling, ZERO and HEXAPOD cannot be copied.

<CSNameCopy> is the name of the copy of the coordinate system.

Response: None

Notes: PI_Base, PI_Levelling, ZERO and HEXAPOD cannot be copied.

Options for creating the copy:

- <CSNameCopy> is a new name. The copy is created as a new coordinate system with this name.
- <CSNameCopy> is the name of an existing coordinate system that is not in use. This overwrites the coordinate system.

The linking to the predecessor in a chain of coordinate systems is copied. The linking to successors is not copied.

The copy is generated in the volatile memory. WPA (p. 213) can be used to write the copy to the nonvolatile memory.

KEN (Enable Coordinate System)

Description: Enables the given coordinate system. The scope of the settings that are influenced by enabling depends on the coordinate system type, see below.

Position values for the motion platform of the positioner (query using POS? (p. 181)) refer to the enabled operating coordinate system.

When applying the work-and-tool concept:

- The work-and-tool concept uses a combination of two enabled operating coordinate systems. Generally, the combination consists of one active coordinate system of the KST type and one of the

KSW type. If a coordinate system is enabled for only one of the two types, a substitute is automatically used for the other type.

- The current position of the motion platform of the positioner queried with POS? can be considered the position of the tool coordinate system in the work coordinate system.

Enabling coordinate systems using KEN does not initiate a motion.

Format: KEN <CSName>

Arguments: <CSName> is the name of the coordinate system to be enabled.

Response: None

Notes: Before enabling a coordinate system, KEN checks that the definition of this coordinate system and its linking are correct. If the coordinate system is not correctly defined, it is not enabled.

KEN can be used to enable coordinate systems of the following types: KSD, KSF, KSW, KST, KSB(USER), KSB(PI), KLF(USER), KLF(PI), KLD(USER), KLD(PI), ZERO

Precisely one coordinate system or precisely one coordinate system combination is enabled from the following groups of coordinate systems:

- Operating coordinate system: A coordinate system of ZERO, or KSF, or KSD type or - for the work-and-tool concept - a combination of coordinate systems of the KSW/KST, or ZERO/KST, or KSW/ZERO type
- Orientational coordinate system (KSB (PI) or KSB(USER) type)
- Leveling coordinate system (of the KLD(USER), or KLD(PI), or KLF(USER), or KLF(PI) type)

By enabling a coordinate system for one of the groups, the coordinate system or the coordinate system combination that was previously enabled for this group is simultaneously disabled.

Sending KEN ZERO re-enables the ZERO operating coordinate system that is enabled by default. When the command level 1 is active (see CCL (p. 123)), KEN ZERO also re-enables the PI_Levelling leveling coordinate system, which was enabled by default, but not the PI_Base orientational coordinate system, which was enabled by default (this can be re-enabled by sending KEN PI_Base). When **DPA SKS** (p. 126) is sent, all coordinate systems enabled by default can be re-enabled independently of the currently active command level.

The enabled operating coordinate system specifies values for the following settings (for the work-and-tool concept, the values are

specified using the combination of two operating coordinate systems):

- NLM (p. 178): Limit for the low end of the axis travel range
- PLM (p. 180): Limit for the high end of the axis travel range
- SSL (p. 188): Activation state of the soft limits of the axis
- SPI (p. 187): Coordinates of the pivot point (only for coordinate systems of the KSF and ZERO types)
- If supported by the controller:
SST: Step size for motion initiated by a manual control unit

Before enabling leveling and orientational coordinate systems (of the KLD(), KLF() and KSB() types), it is necessary to switch to command level 1 (see CCL).

The coordinate systems are enabled and disabled in the volatile memory. WPA (p. 213) can be used to write the activation state to the nonvolatile memory.

KEN? (Get Enabled Coordinate System)

Description:	Lists the names of the enabled coordinate systems and displays their type.
Format:	KEN? [{<CSName>}]
Arguments:	<CSName> is the name of an enabled coordinate system. Illegal: ZERO.
Response:	When <CSName> is omitted, all enabled coordinate systems are listed. {<CSName>="<CSType>}
	where
	<CSType> indicates the coordinate system type.
Notes:	KEN? queries the volatile memory.

When the ZERO operating coordinate system is enabled, it is **not** displayed in the response to KEN? and the response contains only the following:

- The enabled leveling coordinate system, i.e., a coordinate system of the KLD(PI), or KLD(USER), or KLF(PI), or KLF(USER) type
- The enabled orientational coordinate system, i.e., a coordinate system of the KSB(PI) or KSB(USER) type

If <CSName> is given in the query and the corresponding coordinate system is not enabled, the C-886 sends an empty response and sets an error (get error code using ERR? (p. 132)).

KET? (Get Enabled Coordinate System Types)

Description:	Lists the enabled coordinate system types and displays the names of the corresponding coordinate systems.
Format:	KET? [{<CSType>}]
Arguments:	<p><CSType> is an enabled coordinate system type. Possible values: KSW, KST, KSF, KSD, KLD(PI), KLD(USER), KLF(PI), KLF(USER), KSB(PI), KSB(USER)</p> <p>When <CSType> is omitted, all enabled coordinate system types are listed.</p>
Response:	<p>{<CSType>="<CSName>}</p> <p>where</p> <p><CSName> indicates the name of the coordinate system.</p>
Notes:	<p>KET? queries the volatile memory.</p> <p>When the operating coordinate system of the ZERO type is enabled, this type is not displayed in the response to KET?, and the response contains only the following:</p> <ul style="list-style-type: none"> ▪ The type of the enabled leveling coordinate system, i.e., KLD(PI) or KLD(USER) or KLF(PI) or KLF(USER) ▪ The type of the enabled orientational coordinate system, i.e., KSB(PI) or KSB(USER) <p>If <CSType> is given in the query and no coordinate system of the corresponding type is enabled, the C-886 sends an empty response and sets an error (get the error code using ERR? (p. 132)).</p>

KLC? (Get Properties Of Work-And-Tool Combinations)

Description:	<p>Lists the properties of the coordinate system combinations for the work-and-tool concept that are present in the volatile memory.</p> <p>The work-and-tool concept uses a combination of two enabled operating coordinate systems. Generally, the combination consists of one active coordinate system of the KST type and one of the KSW type. If a coordinate system is enabled for only one of the two types, a substitute is automatically used for the other type. Coordinate systems used as a substitute are listed under the name "Zero" in the response to KLC?.</p> <p>A combination is created in the volatile memory when a coordinate system of the KST or KSW type is enabled. The combinations remain in</p>
--------------	--

the volatile memory even if the coordinate systems of the KST or KSW type contained therein are no longer enabled.

When a coordinate system of the KST or KSW type is deleted using KRM (p. 159), the response to KLC? no longer lists the combinations in which this coordinate system was involved.

WPA (p. 213) can be used to write the combinations in the volatile memory to the nonvolatile memory.

KLS? (p. 156) can be used to get the properties of the coordinate systems in the volatile memory.

Format: KLC? [<CSName1>[<CSName2>[<Item1>[<Item2>]]]]

Arguments: <CSName1> is the name of a coordinate system of the KST or KSW type that is part of a combination in the volatile memory.

<CSName2> is the name of a coordinate system of the KST or KSW type that is part of a combination in the volatile memory.

<Item1> is a property of the axes of the controller for the queried combination of coordinate systems. Possible values:

- NLM: Limit for the low end of the axis travel range ("soft limit")
- PLM: Limit for the high end of the axis travel range ("soft limit")
- SSL: Activation state of the soft limits of the axis
- If supported by the controller:
 - SST: Step size for motion initiated by a manual control unit

The settings for the properties of the currently enabled combination can be changed using the corresponding commands and saved using WPA.

<Item2> is an axis of the controller, possible values: X, Y, Z, U, V, W

If the properties of all coordinate system combinations are to be listed, all arguments are omitted.

Response: <String>

<String> contains information in XML format on the combinations of coordinate systems in the volatile memory.

The response structure depends on the number of arguments in the command sent. Possible responses, depending on the number of arguments in the example of coordinate systems Node1, Node2, Node3 and Node4:

Sent

KLC?

Response:

```
<CombinedCoordinateSystem>[SP][LF]
[TAB] <NODE1.NODE2 Name="NODE1.NODE2"
Work="NODE1" Tool="NODE2">[SP][LF]
[TAB] [TAB] <NLM X="-3.0" ... W="-5.0"/>[SP][LF]
[TAB] [TAB] <PLM X="3.0" ... W="5.0"/>[SP][LF]
[TAB] [TAB] <SSL X="1" Y="0" ... W="1"/> [SP][LF]
[TAB] [TAB] <SST X="0.1" ... W="0.2"/>[SP][LF]
[TAB] </NODE1.NODE2>[SP][LF]
[TAB] [TAB] <NODE1.NODE4 Name="NODE1.NODE4"
Work="NODE1" Tool="NODE4"> [SP][LF]
[TAB] [TAB] <NLM X="-4.0" ... W="-3.4"/>[SP][LF]
[TAB] [TAB] <PLM X="2.0" ... W="2.0"/>[SP][LF]
[TAB] [TAB] <SSL X="1" Y="0" ... W="1"/> [SP][LF]
[TAB] [TAB] <SST X="0.2" ... W="0.15"/>[SP][LF]
[TAB] </NODE1.NODE4>[SP][LF]
...
</CombinedCoordinateSystem>[LF]
```

Sent

KLC? Node1

Response:

```
<CombinedCoordinateSystem>[SP][LF]
[TAB] <NODE1.NODE2 Name="NODE1.NODE2"
Work="NODE1" Tool="NODE2"> [SP][LF]
[TAB] [TAB] <NLM X="-3.0" ... W="-5.0"/>[SP][LF]
[TAB] [TAB] <PLM X="3.0" ... W="5.0"/>[SP][LF]
[TAB] [TAB] <SSL X="1" Y="0" ... W="1"/> [SP][LF]
[TAB] [TAB] <SST X="0.1" ... W="0.2"/>[SP][LF]
[TAB] </NODE1.NODE2 >[SP][LF]
[TAB] <NODE1.NODE4 Name="NODE1.NODE4"
Work="NODE1" Tool="NODE4"> [SP][LF]
[TAB] [TAB] <NLM X="-1.0" ... W="-7.0"/>[SP][LF]
[TAB] [TAB] <PLM X="1.1" ... W="10.0"/>[SP][LF]
[TAB] [TAB] <SSL X="1" Y="0" ... W="1"/> [SP][LF]
[TAB] [TAB] <SST X="0.2" ... W="0.21"/>[SP][LF]
[TAB] </NODE1.NODE4 >[SP][LF]
...
</CombinedCoordinateSystem>[LF]
```

Sent

KLC? Node1 Node2

Response:

```
<CombinedCoordinateSystem>[SP][LF]
```

```
[TAB] <NODE1.NODE2 Name="NODE1.NODE2 "
Work="NODE1" Tool="NODE2"> [SP][LF]
[TAB] [TAB] <NLM X="-3.0" ... W="-5.0"/>[SP][LF]
[TAB] [TAB] <PLM X="3.0" ... W="5.0"/>[SP][LF]
[TAB] [TAB] <<SSL X="1" Y="0" ... W="1"/> [SP][LF]
[TAB] [TAB] <SST X="0.1" ... W="0.2"/>[SP][LF]
[TAB] </NODE1.NODE2 >[SP][LF]
</CombinedCoordinateSystem>[LF]
```

Sent

```
KLC? Node1 Node2 PLM
```

Response:

```
<PLM X="3.0" ... W="5.0"/>[LF]
```

Sent

```
KLC? Node1 Node2 PLM X
```

Response:

```
X = 3.0 [LF]
```

KLD (Define Leveling Coordinate System By Specifying Values)

Description: Defines a leveling coordinate system of the KLD(USER) type for permanently correcting errors in the positioner alignment (e.g., faulty mounting).

The leveling coordinate system is defined on the basis of measurements (e.g., using an interferometer) and corrects the linear displacement (X, Y, Z axes) and axis tilt (U, V, W axes) of the motion platform of the positioner.

If the linear displacement and axis tilt cannot be measured: Use KLF (p. 152) to define a leveling coordinate system.

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format: KLD <CSName> [{<AxisID> <Offset>}]

Arguments: <CSName> is the name of the coordinate system to be defined.

<AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.

<Offset> is an offset which is added to the current position value of the axis after the reference move; in physical units.

For axes not given in the KLD command, the offset is set to zero.

Response: None

Notes: Before defining a leveling coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).

Options for defining a coordinate system using KLD:

- <CSName> is a new name. The leveling coordinate system is created under this new name.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition created with KLD.

Recommended procedure for defining and enabling a leveling coordinate system of the KLD(USER) type:

1. Perform a reference move (see FRF (p. 132))
2. Measure the discrepancy between the position and orientation of the motion platform of the positioner and the position and orientation for which $X = 0$, $Y = 0$, $Z = 0$, $U = 0$, $V = 0$, $W = 0$ is to apply in the future (measurement using an external measuring instrument)
3. Switch to command level 1 by sending `CCL 1 advanced`
4. Define the leveling coordinate system using KLD by giving the measured discrepancies for the axes of the motion platform (offset values)
5. Enable the leveling coordinate system (see KEN (p. 144))
6. Optional: Define the behavior after the reference move by setting the **Behavior After Reference Move** (ID 0x07030401) and **Target For Motion After Reference Move** (ID 0x07030402) parameters. In this way, for example, the platform can automatically be moved to the zero position after the reference move.
 - Value of the parameter 0x07030401 = 0: Motion platform remains in the reference position after the reference move
 - Value of parameter 0x07030401 = 1: After the reference move, the motion platform moves to the target position which is given by parameter 0x07030402
7. Save the settings by sending `WPA SKS`

The offset values which are displayed in the response to KLS? (p. 156) for the coordinate systems of the KLD(USER) type result via recalculation from all currently enabled coordinate systems. Thus, they can change when changing the enabled coordinate systems. The offset

values listed in the response to KLT? (p. 158), however, refer in each case to the given predecessor in the chain and are thus independent of the currently enabled coordinate systems.

The following applies for leveling coordinate systems defined using KLD:

- The leveling coordinate system is always the direct successor to the default PI_Levelling leveling coordinate system (automatic linking).
- The leveling coordinate system **cannot** be linked to other coordinate systems using KLN (p. 154)

DPA SKS (p. 126) re-enables the default PI_Levelling leveling coordinate system independently of the currently enabled command level; for details see KEN (p. 144).

KLF (Define Leveling Coordinate System At Current Position)

Description:	<p>Defines a leveling coordinate system of the KLF(USER) type for permanently correcting errors in the positioner alignment (e.g., faulty mounting).</p> <p>To define the leveling coordinate system, the motion platform after the reference move is commanded into the position and orientation for which $X = 0$, $Y = 0$, $Z = 0$, $U = 0$, $V = 0$, $W = 0$ is applicable in future. Sending KLF defines a coordinate system with offset values which are added, after the reference move, to the current position values for the axes; in physical units.</p> <p>If the linear displacement (X, Y, Z axes) and axis tilt (U, V, W axes) are to be measured: Use KLD (p. 150) to define a leveling coordinate system.</p> <p>The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.</p>
Format:	KLF <CSName>
Arguments:	<CSName> is the name of the coordinate system to be defined.
Response:	None
Notes:	Before defining a leveling coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).

Options for defining a coordinate system using KLF:

- <CSName> is a new name. The leveling coordinate system is created under this new name.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition created with KLF.

The definition using KLF is possible only when the positioner is not in motion.

Recommended procedure for defining and enabling a leveling coordinate system of the KLF(USER) type:

1. Perform a reference move (see FRF (p. 132))
2. Approach the position and orientation of the motion platform of the positioner for which $X = 0$, $Y = 0$, $Z = 0$, $U = 0$, $V = 0$, $W = 0$ is to be applicable in the future
3. Switch to command level 1 by sending `CCL 1 advanced`
4. Define the leveling coordinate system using KLF
5. Enable the leveling coordinate system (see KEN (p. 144))
6. Optional: Define the behavior after the reference move by setting the **Behavior After Reference Move** (ID 0x07030401) and **Target For Motion After Reference Move** (ID 0x07030402) parameters. In this way, for example, the platform can automatically be moved to the zero position after the reference move.
 - Value of the parameter 0x07030401 = 0: Motion platform remains in the reference position after the reference move
 - Value of parameter 0x07030401 = 1: After the reference move, the motion platform moves to the target position which is given by parameter 0x07030402
7. Save the settings by sending `WPA SKS`

The offset values that are displayed in the response to KLS? (p. 156) for the coordinate systems of the KLF(USER) type result by means of recalculation from all currently enabled coordinate systems. Thus, they can change when changing the enabled coordinate systems. The offset values listed in the response to KLT? (p. 158), however, refer in each case to the given predecessor in the chain and are thus independent of the currently enabled coordinate systems.

The following applies for leveling coordinate systems defined using KLF:

- The leveling coordinate system is always the direct successor to the default PI_Levelling leveling coordinate system (automatic linking).
- The leveling coordinate system **cannot** be linked to other coordinate systems using KLN (p. 154)

DPA SKS (p. 126) re-enables the default PI_Levelling leveling coordinate system independently of the currently enabled command level; for details see KEN (p. 144).

KLN (Link Coordinate Systems)

Description: Links two coordinate systems to create a chain of predecessor and successor.

Format: KLN <CSName1> <CSName2>

Arguments: <CSName1> is the name of the coordinate system that is to be attached in the chain as the successor to <CSName2>.

<CSName2> is the name of the coordinate system that is to be the predecessor of <CSName1> in the chain.

Response: None

Notes: Each coordinate system is part of at least one chain. For the basic structure of coordinate system chains, see "Coordinate Systems" (p. 32).

By default, the following coordinate systems are linked to form a chain:

- The HEXAPOD coordinate system, which is based on the configuration file with the geometric data of the positioner, is the "origin" of all chains and the predecessor of the PI_Levelling leveling coordinate system (fixed linking)
- PI_Levelling is the predecessor of the PI_Base orientational coordinate system
- PI_Base is the predecessor of the ZERO operating coordinate system

The following applies for linked coordinate systems:

- The actual offset values for the position of the X, Y, Z, U, V, W axes result from the offset values for the predecessors linked to a coordinate system (for details, see KLT? (p. 158)).
- Each coordinate system has precisely one predecessor and can have at least one successor.
- When a coordinate system is enabled, all predecessors in the chain are likewise in use and cannot be deleted or overwritten.
- When a coordinate system (unused) is deleted, its predecessor and successor are linked to one another in the chain.

Limitations for creating chains using KLN:

- A coordinate system cannot be linked to itself.
- Although ring connections from at least two coordinate systems can be formed, they cannot be enabled using KEN (p. 144).
- Using KLN, a coordinate system in use cannot be attached as successor to another coordinate system.
- Using KLN, a coordinate system (not in use) can be attached as

successor to a coordinate system in use.

- Before linking orientational coordinate systems of the KSB(USER) type as successors, it is necessary to switch to command level 1 (see CCL (p. 123)).
- Using KLN, coordinate systems of the KLD(PI), KLF(PI), KLD(USER), KLF(USER) types and the HEXAPOD coordinate system cannot be linked.
- Using KLN, the PI_Base coordinate system cannot be attached as successor to another coordinate system.
- Using KLN, coordinate systems of the KSB(USER) type can only be linked to other coordinate systems of the KSB(USER) type or as successor to the PI_Base coordinate system.
- Using KLN, the ZERO coordinate system cannot be attached as successor to another coordinate system.

Using KLN, coordinate systems are linked in the volatile memory. WPA (p. 213) can be used to write the linking to the nonvolatile memory.

KLN? (Get Coordinate System Chains)

Description:	Lists the components of the existing coordinate system chains.
	Each coordinate system is part of at least one chain. For the basic structure of coordinate system chains, see "Coordinate Systems" (p. 32).
Format:	KLN? [{<CSName>}]
Arguments:	<CSName> is the name of a coordinate system, the predecessors of which are to be listed in the chain.
	If the predecessors of all coordinate systems are to be listed, <CSName> is omitted.
Response:	{<CSName>="<String>}
	where
	<String> contains the names of the coordinate system predecessors in the chain. The "origin" of the chain is always given at the end of the line.
Notes:	To ensure a clear overview, only the predecessors up to the ZERO coordinate system are listed for operating coordinate systems of the KSD, KSF, KSW and KST types. The predecessors of ZERO, however, can be specifically queried and are also contained as a separate line in the response if <CSName> is omitted from the query.

KLS? (Get Coordinate System Properties)

Description:	Lists the properties of the coordinate systems that are present in the volatile memory.
	WPA (p. 213) can be used to write the coordinate systems in the volatile memory to the nonvolatile memory.
	The properties of the coordinate system combinations for the work-and-tool concept that are present in the volatile memory can be queried using KLC? (p. 147).
Format:	KLS? [<CSName>[<Item1>[<Item2>]]]
Arguments:	<p><CSName> is the name of a coordinate system. Illegal: HEXAPOD.</p> <p><Item1> is a property of the axes of the controller. Possible values: For all types of coordinate systems:</p> <ul style="list-style-type: none"> ▪ POS: Offset for positions of the axes <p>Only for coordinate systems of the KSD, KSF and ZERO types:</p> <ul style="list-style-type: none"> ▪ NLM: Limit for the low end of the axis travel range ("soft limit") ▪ PLM: Limit for the high end of the axis travel range ("soft limit") ▪ SSL: Activation state of the soft limits of the axis ▪ If supported by the controller: <ul style="list-style-type: none"> SST: Step size for motion initiated by a manual control unit <p>Only for coordinate systems of the KSF and ZERO types:</p> <ul style="list-style-type: none"> ▪ SPI: Coordinates of the pivot point <p><Item2> is an axis of the controller, possible values: X, Y, Z, U, V, W</p> <p>If the properties of all coordinate systems are to be listed, all arguments are omitted.</p>
Response:	<p><String></p> <p><String> contains information in XML format on the coordinate systems in the volatile memory.</p> <p>The <String> structure depends on the number of arguments in the command sent.</p> <p>If no argument or only the name of a coordinate system is given, <String> lists, for each coordinate system contained in the response, the data applicable for <Item1> and <Item2> as well as the information below:</p> <ul style="list-style-type: none"> ▪ Name of the coordinate system (Name="") ▪ Name of the direct predecessor of the coordinate system in the chain (Parent="")

- Current use of the coordinate system;
 - Used="True": The coordinate system is in use, i.e., it is either enabled itself, or it is a predecessor of the enabled coordinate system in its chain.
 - Used="False": The coordinate system is not used.
- Coordinate system type (Type="")

Note:

The offset values which are displayed in the response to KLS? for the leveling coordinate systems of the KLD(USER) and KLF(USER) types result by means of recalculation from all currently enabled coordinate systems. Thus, they can change when changing the enabled coordinate systems.

Example:

The example below shows the response to KLS? for the coordinate systems existing by default.

KLS?

```
<SingleCoordinateSystem>
  <ZERO Name="ZERO" Parent="PI_BASE" Used="True" Type="ZERO">
    <POS X="0.000000" Y="0.000000" Z="0.000000"
U="0.000000" V="0.000000" W="0.000000"/>
    <NLM X="-10.000100" Y="-10.000100" Z="-10.000100"
U="-1.000100" V="-1.000100" W="-1.000100"/>
    <PLM X="10.000100" Y="10.000100" Z="10.000100"
U="1.000100" V="1.000100" W="1.000100"/>
    <SSL X="1" Y="1" Z="1" U="1" V="1" W="1"/>
    <SPI R="0.000000" S="0.000000" T="481.000000"/>
    <SST X="0.010000" Y="0.010000" Z="0.010000"
U="0.010000" V="0.010000" W="0.010000"/>
  </ZERO>
  <PI_BASE Name="PI_BASE" Parent="PI_LEVELLING" Used="True"
Type="KSB(PI)">
    <POS X="0.000000" Y="0.000000" Z="0.000000"
U="0.000000" V="0.000000" W="0.000000"/>
  </PI_BASE>
  <PI_LEVELLING Name="PI_LEVELLING" Parent="HEXAPOD"
Used="True" Type="KLD(PI)">
    <POS X="0.000000" Y="0.000000" Z="0.000000"
U="0.000000" V="0.000000" W="0.000000"/>
  </PI_LEVELLING></SingleCoordinateSystem>
```

KLT? (Get Offset Resulting From A Chain)

Description:	<p>Lists, for a coordinate system, the actual offset values for the position of the X, Y, Z, U, V, W axes which result from the offset values of the predecessors linked to this coordinate system.</p> <p>For operating coordinate systems, only the operating coordinate systems linked as predecessors up to the ZERO coordinate system are included in the calculation.</p> <p>For orientational coordinate systems, only the orientational coordinate systems linked as predecessors up to the HEXAPOD coordinate system are included in the calculation.</p> <p>For leveling coordinate systems, only the leveling coordinate systems linked as predecessors up to the HEXAPOD coordinate system are included in the calculation.</p>
Format:	KLT? [<StartCS> [<EndCS>]]
Arguments:	<p><StartCS> is the name of the coordinate system for which the offset values resulting from its predecessors are to be queried.</p> <p><EndCS> is the name of a coordinate system linked as a predecessor of <StartCS>, which is to be used as the starting point for the offset calculation. If <EndCS> is omitted, the starting point for the calculation depends on the value for <StartCS>:</p> <ul style="list-style-type: none"> ▪ <StartCS> is an operating coordinate system (KSD, KSF, KST, KSW, or ZERO type): Starting point is ZERO ▪ <StartCS> is an orientational or leveling coordinate system (KSB(PI), KSB(USER), KLD(PI), KLD(USER), KLF(PI), KLF(USER) types): Starting point is HEXAPOD <p>If the resultant offset values for all coordinate systems are to be listed, all arguments are omitted.</p>
Response:	<p><String></p> <p><String> contains, for each queried coordinate system, a line with the following data:</p> <ul style="list-style-type: none"> ▪ Name: The name of the coordinate system for which the resultant offset values are listed. ▪ EndCoordinateSystem: The name of the coordinate system which was used as the starting point for calculating the offset values. ▪ X, Y, Z, U, V, W: Resultant offset value for the corresponding axis.
Note:	<p>KLT? queries the volatile memory.</p> <p>KLS? (p. 156) can be used to get the properties of the coordinate systems in the volatile memory.</p>

Example: The example below shows the response to KLT? for the coordinate systems existing by default.

KLT?

```
Name=ZERO EndCoordinateSystem=ZERO X=0.000000 Y=0.000000
Z=0.000000 U=0.000000 V=0.000000 W=0.000000
Name=PI_BASE EndCoordinateSystem=HEXAPOD X=0.000000
Y=0.000000 Z=0.000000 U=0.000000 V=0.000000 W=0.000000
Name=PI_LEVELLING EndCoordinateSystem=HEXAPOD X=0.000000
Y=0.000000 Z=0.000000 U=0.000000 V=0.000000 W=0.000000
```

KRM (Remove Coordinate System)

Description: Deletes a coordinate system.

Format: KRM <CSName>

Arguments: <CSName> is the name of the coordinate system to be deleted.

Response: None

Notes: A coordinate system in use (i.e., it is enabled itself or linked to the enabled coordinate system as a predecessor) cannot be deleted.

When a coordinate system (unused) is deleted, its predecessor and its successor are linked to one another in the coordinate system chain.

Before deleting a leveling coordinate system of the KLD(USER), KLF(USER), and KSB(USER) types, it is necessary to switch to command level 1 (see CCL (p. 123)).

The coordinate system is deleted from the volatile memory. WPA (p. 213) can be used to transfer the deletion to the nonvolatile memory.

KSB (Define Orientational Coordinate System)

Description: Defines an orientational coordinate system of the KSB(USER) type for permanently changing the direction of the X, and/or Y, and/or Z axes.

The direction of the axes is changed by rotating the coordinate system in 90° increments as follows:

- Rotating around X, i.e., angular value given for U, changes the direction of the Y and Z axes
- Rotating around Y, i.e., angular value given for V, changes the

direction of the X and Z axes

- Rotating around Z, i.e., angular value given for W, changes the direction of the X and Y axes

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format: KSB <CSName> [{<AxisID> <Angle>}]

Arguments: <CSName> is the name of the coordinate system to be defined.

<AxisID> is one axis of the controller. Possible values: U, V, W.

<Angle> is the angle around which the axis is to be rotated. Possible values: 0, 90, 180, 270, -90, -180, -270 (unit: Degree).

For axes not given in the KSB command, the angle is set to zero.

Response: None

Notes: Before defining an orientational coordinate system, it is necessary to switch to command level 1 (see CCL (p. 123)).

Options for defining a coordinate system using KSB:

- <CSName> is a new name. The orientational coordinate system is newly created under this name and automatically has the enabled orientational coordinate system as its predecessor
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition using KSB and automatically attached as successor to the enabled orientational coordinate system

KLN (p. 154) can be used to link orientational coordinate systems of the KSB(USER) type to other coordinate systems of the KSB(USER) type or attach them as successors to the PI_Base coordinate system.

Oriental coordinate systems of the KSB(USER) type can be enabled using KEN (p. 144). DPA SKS (p. 126) re-enables the PI_Base orientational coordinate system that is enabled by default; for details see KEN.

KSD (Define Operating Coordinate System By Specifying Values)

Description: Defines an operating coordinate system of the KSD type.

A coordinate system of the KSD type can be placed and aligned in space arbitrarily. For more details, see "Work-and-Tool Concept" and in particular "Consideration of Coordinate Systems of the KSD and KSF

Type" from the "Work" and "Tool" Perspective" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2).

The placement of the coordinate system in space is defined by giving offset values for the X, Y, Z axes. The alignment of the coordinate system is defined by giving offset values for the U, V, W axes:

- Rotating around X, i.e., offset given for U, changes the direction of the Y and Z axes
- Rotating around Y, i.e., offset given for V, changes the direction of the X and Z axes
- Rotating around Z, i.e., offset given for W, changes the direction of the X and Y axes

If an operating coordinate system of the KSD type has been enabled using KEN (p. 144):

- The coordinate system determines the position values for the motion platform of the positioner (get the current position using POS? (p. 181)).
- The coordinates of the center of rotation **cannot** be changed with SPI (p. 187), and the pivot point defined with SPI is **not** used.

If a coordinate system of the KSD type is not enabled itself, but it nonetheless is linked to the enabled operating coordinate system as a predecessor, its offset values are included in the calculation of the offset values for the enabled operating coordinate system (see KLT? (p. 158)).

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format: KSD <CSName> [{<AxisID> <Offset>}]

Arguments: <CSName> is the name of the coordinate system to be defined.

<AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.

<Offset> is an offset value for placing or aligning the axis; in physical units.

For axes not given in the KSD command, the offset is set to zero.

Response: None

Notes: When an operating coordinate system of the KSD type is enabled:

- In addition to the offset values for the X, Y, Z, U, V, W axes, the coordinate system specifies values for the following settings:
 - NLM (p. 178): Limit for the low end of the axis travel range

- PLM (p. 180): Limit for the high end of the axis travel range
- SSL (p. 188): Activation state of the soft limits of the axis
- If supported by the controller:
SST: Step size for motion initiated by a manual control unit
- The current settings can be changed using the corresponding commands.
- The current settings can be queried with KLS? (p. 156).
- The current settings are saved for the coordinate system with WPA (p. 213).
- For details on triggering motion, see "Commanding Motion" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007).

Options for defining a coordinate system using KSD:

- <CSName> is a new name. The operating coordinate system is newly created under this name and automatically has the ZERO operating coordinate system as its predecessor.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition created with KSD.
 - If the overwritten coordinate system was also of the KSD type, the chain with its predecessor and successor is retained.
 - If the overwritten coordinate system was not of the KSD type, its type is changed to KSD and it is attached to ZERO as the new predecessor.

KLN (p. 154) can be used to link operating coordinate systems of the KSD, KSF, KST, KSW types to other operating coordinate systems of these types or reattach them as successors to the ZERO coordinate system.

`KEN ZERO` and `DPA SKS` re-enable the ZERO operating coordinate system that is enabled by default; for details see KEN (p. 144).

KSF (Define Operating Coordinate System At Current Position)

Description: Defines an operating coordinate system of the KSF type at the current position of the motion platform of the positioner.

For more details, see "Work-and-Tool Concept" and in particular "Consideration of Coordinate Systems of the KSD and KSF Type" from the "Work" and "Tool" Perspective" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2).

If an operating coordinate system of the KSF type was enabled using KEN (p. 144), it determines the position values for the motion platform of the positioner (get the current position using POS? (p. 181)), and the pivot point defined using SPI (p. 187) is used as center of rotations.

If a coordinate system of the KSF type is not enabled itself, but it nonetheless is linked to the enabled operating coordinate system as a predecessor, its offset values are included in the calculation of the offset values for the enabled operating coordinate system (see KLT? (p. 158)).

The definition using KSF is possible only when the positioner is not in motion.

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format:	KSF <CSName>
Arguments:	<CSName> is the name of the coordinate system to be defined.
Response:	None
Notes:	If a coordinate system of the KSF type is defined, its pivot point coordinates (R, S, T, see SPI (p. 187)) are set to the values valid for the currently enabled operating coordinate system. If the enabled operating coordinate system does not support the pivot point set using SPI (KSD, KST/KSW types), the pivot point coordinates of the KSF coordinate system are set to $R = S = T = 0$.

When an operating coordinate system of the KSF type is enabled:

- In addition to the offset values for the X, Y, Z, U, V, W axes, the coordinate system specifies values for the following settings:
 - NLM (p. 178): Limit for the low end of the axis travel range
 - PLM (p. 180): Limit for the high end of the axis travel range
 - SSL (p. 188): Activation state of the soft limits of the axis
 - SPI (p. 187): Pivot point coordinates R, S, T
 - If supported by the controller:
 - SST: Step size for motion initiated by a manual control unit
- The current settings can be changed using the corresponding commands. The pivot point coordinates can only be changed using SPI if $U = V = W = 0$ applies for the current position of the platform.
- The current settings can be queried with KLS? (p. 156).
- The current settings are saved for the coordinate system with WPA (p. 213).
- For details on triggering motion, see "Commanding Motion" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007).

Options for defining a coordinate system using KSF:

- <CSName> is a new name. The operating coordinate system is newly created under this name and automatically has the ZERO operating coordinate system as its predecessor.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition using KSF and automatically attached as successor to the ZERO operating coordinate system.

KLN (p. 154) can be used to link operating coordinate systems of the KSD, KSF, KST, KSW types to other operating coordinate systems of these types or reattach them as successors to the ZERO coordinate system.

`KEN ZERO` and `DPA SKS` re-enable the ZERO operating coordinate system that is enabled by default; for details see KEN (p. 144).

KST (Define "Tool" Operating Coordinate System)

Description: Defines an operating coordinate system of the KST type ("tool coordinate system") for the work-and-tool concept.

See "The Work-and-Tool Concept" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2) for further explanations.

The placement of the tool coordinate system is defined by giving offset values for the X, Y, Z axes. The alignment of the tool coordinate system is defined by giving offset values for the U, V, W axes:

- Rotating around X, i.e., offset given for U, changes the direction of the Y and Z axes
- Rotating around Y, i.e., offset given for V, changes the direction of the X and Z axes
- Rotating around Z, i.e., offset given for W, changes the direction of the X and Y axes

When an operating coordinate system of the KST type is enabled, the work-and-tool concept is used and the following applies:

- In addition to the tool coordinate system, a work coordinate system must be enabled. When no work coordinate system of the KSW type is enabled, an automatically generated work coordinate system is used as a substitute.
- The coordinates of the center of rotation **cannot** be changed with SPI (p. 187), and the pivot point defined with SPI is **not** used.

If a coordinate system of the KST type is not enabled itself, but it nonetheless is linked to the enabled operating coordinate system as a predecessor, its offset values are included in the calculation of the offset values for the enabled operating coordinate system (see KLT? (p. 158)).

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format:

KST <CSName> [{<AxisID> <Offset>}]

Arguments:

<CSName> is the name of the coordinate system to be defined.

<AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.

<Offset> is an offset value for placing or aligning the axis; in physical units.

For axes not given in the KST command, the offset is set to zero.

Response:

None

Notes:

When a combination of coordinate systems of the KSW/KST, or ZERO/KST, or KSW/ZERO types is enabled:

- In addition to the offset values for the X, Y, Z, U, V, W axes, the combination specifies values for the following settings:
 - NLM (p. 178): Limit for the low end of the axis travel range
 - PLM (p. 180): Limit for the high end of the axis travel range
 - SSL (p. 188): Activation state of the soft limits of the axis
 - If supported by the controller:
 - SST: Step size for motion initiated by a manual control unit
- The current settings can be changed using the corresponding commands.
- The current settings can be queried with KLC? (p. 147).
- The current settings are saved with WPA (p. 213) for the coordinate system combination.
- For details on triggering motion, see "Commanding Motion" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007).

Options for defining a coordinate system using KST:

- <CSName> is a new name. The operating coordinate system is newly created under this name and automatically has the ZERO operating coordinate system as its predecessor.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition created with KST.

- If the overwritten coordinate system was also of the KST type, the chain with its predecessor and successor is retained.
- If the overwritten coordinate system was not of the KST type, its type is changed to KST and it is attached to ZERO as the new predecessor.

KLN (p. 154) can be used to link operating coordinate systems of the KSD, KSF, KST, KSW types to other operating coordinate systems of these types or reattach them as successors to the ZERO coordinate system.

`KEN ZERO` and `DPA SKS` re-enable the ZERO operating coordinate system that is enabled by default; for details see KEN (p. 144).

KSW (Define "Work" Operating Coordinate System)

Description: Defines an operating coordinate system of the KSW type ("work coordinate system") for the work-and-tool concept.

See "The Work-and-Tool Concept" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2) for further explanations.

The placement of the work coordinate system is defined by giving offset values for the X, Y, Z axes. The alignment of the work coordinate system is defined by giving offset values for the U, V, W axes:

- Rotating around X, i.e., offset given for U, changes the direction of the Y and Z axes
- Rotating around Y, i.e., offset given for V, changes the direction of the X and Z axes
- Rotating around Z, i.e., offset given for W, changes the direction of the X and Y axes

When an operating coordinate system of the KSW type is enabled, the work-and-tool concept is used and the following applies:

- In addition to the work coordinate system, a tool coordinate system must be enabled. When no tool coordinate system of the KST type is enabled, an automatically generated tool coordinate system is used as a substitute.
- The coordinates of the center of rotation **cannot** be changed with SPI (p. 187), and the pivot point defined with SPI is **not** used.

If a coordinate system of the KSW type is not enabled itself, but it nonetheless is linked to the enabled operating coordinate system as a predecessor, its offset values are included in the calculation of the offset values for the enabled operating coordinate system (see KLT?

(p. 158)).

The coordinate system is defined in the volatile memory. WPA (p. 213) can be used to write the definition to the nonvolatile memory.

Format: KSW <CSName> [{<AxisID> <Offset>}]

Arguments: <CSName> is the name of the coordinate system to be defined.

<AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.

<Offset> is an offset value for placing or aligning the axis; in physical units.

For axes not given in the KSW command, the offset is set to zero.

Response: None

Notes: When a combination of coordinate systems of the KSW/KST, or ZERO/KST, or KSW/ZERO types is enabled:

- In addition to the offset values for the X, Y, Z, U, V, W axes, the combination specifies values for the following settings:
 - NLM (p. 178): Limit for the low end of the axis travel range
 - PLM (p. 180): Limit for the high end of the axis travel range
 - SSL (p. 188): Activation state of the soft limits of the axis
 - If supported by the controller:
 - SST: Step size for motion initiated by a manual control unit
- The current settings can be changed using the corresponding commands.
- The current settings can be queried with KLC? (p. 147).
- The current settings are saved with WPA (p. 213) for the coordinate system combination.
- For details on triggering motion, see "Commanding Motion" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007).

Options for defining a coordinate system using KSW:

- <CSName> is a new name. The operating coordinate system is newly created under this name and automatically has the ZERO operating coordinate system as its predecessor.
- <CSName> is the name of an existing coordinate system that is not in use. The coordinate system is overwritten by the definition created with KSW.
 - If the overwritten coordinate system was also of the KSW type, the chain with its predecessor and successor is retained.
 - If the overwritten coordinate system was not of the KSW type, its type is changed to KSW and it is attached to ZERO as the new predecessor.

KLN (p. 154) can be used to link operating coordinate systems of the KSD, KSF, KST, KSW types to other operating coordinate systems of these types or reattach them as successors to the ZERO coordinate system.

`KEN ZERO` and `DPA SKS` re-enable the ZERO operating coordinate system that is enabled by default; for details see KEN (p. 144).

MRT (Set Target Relative In Tool Coordinate System)

Description: Moves the given axis relative in the tool coordinate system.

See "The Work-and-Tool Concept" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2) for further explanations.

While the target position to be approached is being determined from the values for <Distance>, the translation is calculated first and then the rotation.

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

Format: MRT {<AxisID> <Distance>}

Arguments: <AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.

<Distance> gives the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units).

Response: None

Troubleshooting:

- Target position outside of the current workspace.
- The **Trajectory Source** parameter (ID 0x19001900) is set to 1 (but must be set to 0 when MRT is used).
- Servo mode is OFF for one of the axes specified.
- The reference move has not been successfully completed for at least one axis.

Notes: When the work-and-tool concept is not applied, motion is performed using MRT in the tool coordinate system, which exists according to the enabled operating coordinate system (see "Consideration of Coordinate Systems of the KSD and KSF Type" from the "Work" and "Tool" Perspective" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007)).

The physical unit in which <Distance> is to be given can be queried with PUN? (p. 182).

In order to determine whether a motion has been completed, it is recommended to send #5 (p. 118).

The motion can be aborted by #24 (p. 121), STP (p. 191) and HLT (p. 137).

MRW (Set Target Relative In Work Coordinate System)

Description:	<p>Moves the given axis relative in the work coordinate system.</p> <p>See "The Work-and-Tool Concept" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007) as well as "Definition of Terms" (p. 2) for further explanations.</p> <p>While the target position to be approached is being determined from the values for <Distance>, the translation is calculated first and then the rotation.</p> <p>Servo mode must be switched on for the commanded axis prior to using this command (closed-loop operation).</p>
Format:	MRW {<AxisID> <Distance>}
Arguments:	<p><AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.</p> <p><Distance> gives the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units).</p>
Response:	None
Troubleshooting:	<ul style="list-style-type: none"> ▪ Target position outside of the current workspace. ▪ The Trajectory Source parameter (ID 0x19001900) is set to 1 (but must be set to 0 when MRW is used). ▪ Servo mode is OFF for one of the axes specified. ▪ The reference move has not been successfully completed for at least one axis.
Notes:	<p>When the work-and-tool concept is not applied, motion is performed using MRW in the work coordinate system, which exists according to the enabled operating coordinate system (see "Consideration of Coordinate Systems of the KSD and KSF Type" from the "Work" and "Tool" Perspective" in the Technical Note "Coordinate Systems for Hexapod Microrobots" (C887T0007)).</p> <p>The physical unit in which <Distance> is to be given can be queried with PUN? (p. 182).</p>

In order to determine whether a motion has been completed, it is recommended to send #5 (p. 118).

The motion can be aborted by #24 (p. 121), STP (p. 191) and HLT (p. 137).

LIM? (Indicate Limit Switches)

Description: Gets whether axes have limit switches.

Format: LIM? [{<AxisID>}]

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

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

where

<uint> indicates whether the axis has limit switches (=1) or not (=0).

Troubleshooting: Illegal axis identifier

MAC (Call Macro Function)

Description: Calls a macro function. Permits recording, deleting and running macros on the controller.

Format: MAC <keyword> {<parameter>}

in particular:

MAC BEG <macroname>

MAC DEF <macroname>

MAC DEF?

MAC DEL <macroname>

MAC END

MAC ERR?

MAC FREE?

MAC NSTART <macroname> <uint> [{<String>}]

MAC START <macroname> [{<String>}]

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 DEF <macroname>

Sets specified macro as start-up macro. This macro will be automatically executed after the next switch-on or reboot of the controller. If <macroname> is omitted, the current start-up macro selection is canceled.

MAC DEF?

Asks for the start-up macro

Response: <macroname>

If no start-up macro is defined, the response is an empty string with the terminating character.

MAC DEL <macroname>

Deletes specified macro.

MAC END

Stops macro recording (cannot become part of a macro)

MAC ERR?

Reports the last error which occurred during macro execution.

Response: <macroname> <uint1> "=" <uint2> <"<"CMD">">

where <macroname> is the name of the macro, <uint1> is the line in the macro, <uint2> is the error code and <"<"CMD">"> is the erroneous command which was sent to the parser.

MAC 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 <macroname> <uint> [{<String>}]

Repeats the specified macro <uint> times. Another execution is started when the last one is finished.

<String> stands for the value of a local variable contained in the macro. The sequence of the values in the input must correspond to the numbering of the appropriate local variables, starting with the value of the local variable 1. The individual values must be separated from each other by spaces. A maximum of 256 characters are permitted per

command line. <String> can be given directly or via the value of another variable. See “Variables” (p. 106) for further details.

MAC START <macroname> [{<String>}]

Starts one execution of specified macro. <String> has the same function as with MAC NSTART.

Response: None

Troubleshooting: Macro recording is active (keywords BEG, DEL) or inactive (END)
Macro contains a disallowed MAC command

Notes: During macro recording no macro execution is allowed.

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

Macros can contain local and global variables. The names of local variables used in a macro must form a continuous series. Example of permissible designation: 1, 2, 3, 4. Designation with e.g. 1, 2, 5, 6 is not allowed. Further information can be found in the section “Variables” (p. 106).

A running macro sends no responses to any interface.

Depending on the value of parameter 0x72 (Ignore Macro Error), there are the following possibilities when an error is caused by the running macro:

0 = macro execution is aborted

1 = the error is ignored and macro execution will be continued

Irrespective of the parameter setting, MAC ERR? always reports the last error that occurred during a macro execution.

The following commands provided by the C-886 can only be used in macros:

ADD (p. 122), CPY (p. 124), DEL (p. 125), JRC (p. 143), MEX (p. 174) and WAC (p. 202).

A macro can start another macro. The maximum number of nesting levels is 10. A macro can call itself to form an infinite loop.

Any commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other, and only the last motion command will be executed, irrespective of its source.

Macro execution can be stopped with #24 (p. 121) and STP (p. 191).

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

A macro cannot be deleted while it is running.

You can query with #8 (p. 119) if a macro is currently running on the controller.

Warning: The number of write cycles of nonvolatile memory is limited.

MAC? (List Macros)

Description:	Lists macros or content of a given macro.
Format:	MAC? [<macroname>]
Arguments	<macroname>: name of the macro whose content shall be listed; if omitted, the names of all stored macros are listed.
Response:	<string>
	If <macroname> was given, <string> is the content of this macro;
	If <macroname> was omitted, <string> is a list with the names of all stored macros
Troubleshooting:	Macro <macroname> not found

MAN? (Get Help String For Command)

Description:	Shows a detailed help text for individual commands.
Format:	MAN? <CMD>
Arguments:	<CMD> is the command mnemonic of the command for which the help text is to be displayed (see below).
Response:	A string that describes the command.
Notes:	A detailed help text can be displayed for the following GCS commands: WPA, WAV, WTR, WGO, WAV?

Example:

Send: MAN? WPA

Receive:

```
WPA <Password> [{<ItemID> <PamID>}] Save
Parameters To Non-Volatile Memory
#AvailablePasswords
<Pswd> <Param_Setting>
100 All Parameters, Settings of Coordinate
System Configuration
101 All Parameters
SKS Settings of Coordinate System Configuration
end of help
```

MEX (Stop Macro Execution Due To Condition)

Description: Stops macro execution due to a given condition of the following type: a given value is compared with a queried value according to a given rule.

Can only be used in macros.

When the macro interpreter accesses this command, the condition is checked. If it is true, the current macro is stopped; otherwise macro execution is continued with the next line. Should the condition be fulfilled later, the interpreter will ignore it.

See also the WAC command (p. 202).

Format: MEX <CMD?> <OP> <Value>

Arguments <CMD?> is one query command in its usual syntax. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:
= <= < > >= !=

Important: There must be a blank space before and after the operator!

<Value> is the value that is compared with the response to <CMD?>.

Response: None

MOV (Set Target Position)

Description: Sets an absolute target position for the given axis.

Format: MOV {<AxisID> <Position>}

Arguments:	<p><AxisID> is one axis of the controller.</p> <p><Position> is the absolute target position in physical units.</p>
Response:	none
Troubleshooting:	<ul style="list-style-type: none"> ▪ Target position outside of the current workspace. ▪ Illegal axis identifier See section "Commandable Items" (p. 23) for more information. ▪ Servo mode is OFF for one of the axes specified. ▪ The reference move has not been successfully completed for at least one axis.
Notes:	<p>The servo mode must be switched on when this command is used (closed-loop operation).</p> <p>The physical unit in which <Position> is to be given can be queried with PUN? (p. 182).</p> <p>In order to determine whether a motion has been completed, it is recommended to send #5 (p. 118).</p> <p>The motion can be aborted by #24 (p. 121), STP (p. 191) and HLT (p. 137).</p> <p>For the axes of the positioner (X, Y, Z, U, V, W), the following is valid:</p> <ul style="list-style-type: none"> ▪ Before the start of every motion, it is checked whether the motion platform can actually reach the commanded target position. With VMO? (p. 201) you can query whether the target position can be reached. ▪ Rotations take place around the center of rotation. ▪ Depending on the setting of the Trajectory Source parameter (ID 0x19001900), the dynamics profile for the axes of the positioner (X, Y, Z, U, V, W) is defined by one of the following two sources: <ul style="list-style-type: none"> – Profile generator of the C-886 – Cyclic transfer of target positions by consecutive MOV commands <p>For more information, see "Motion of the positioner" (p. 27).</p>
Example 1:	<p>Send: MOV X 10 U 5</p> <p>Note: Axis X moves to 10 (target position in mm), axis U moves to 5 (target position in °)</p>
Example 2:	<p>Send: MOV X 4 Y 2.3 Z -3 U -5.3 V 3 W 1</p> <p>Note: Target positions can be given for all six axes of the positioner with a single motion command.</p>

Example 3: Send: MOV Z 100
 Send: ERR?
 Receive: 7
 Note: The axis does not move. The error code "7" in the response to the ERR? command (p. 132) indicates that the target position given in the motion command is out of limits

MOV? (Get Target Position)

Description: Returns last valid commanded target position.
 Format: MOV? [{<AxisID>}]
 Arguments: <AxisID> is one axis of the controller
 Response: {<AxisID>="<float> LF}

 where

 <float> is the last commanded target position in physical units

 Troubleshooting: Illegal axis identifier

 Notes: The target position can be changed by different sources; see "Motion of the positioner" (p. 27).
 MOV? gets the commanded positions. Use POS? (p. 181) to get the current positions.

MVR (Set Target Relative To Current Position)

Description: Moves the given axis relative to the last commanded target position.
 Format: MVR {<AxisID> <Distance>}
 Arguments: <AxisID> is one axis of the controller.

 <Distance> gives the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units).

 Response: none

Troubleshooting:

- Target position outside of the current workspace.
- Illegal axis identifier
See section "Commandable Items" (p. 23) for more information.
- Servo mode is OFF for one of the axes specified.
- The reference move has not been successfully completed for at least one axis.

Notes:

The servo mode must be switched on when this command is used (closed-loop operation).

The physical unit in which <Distance> is to be given can be queried with PUN? (p. 182).

In order to determine whether a motion has been completed, it is recommended to send #5 (p. 118).

The motion can be aborted by #24 (p. 121), STP (p. 191) and HLT (p. 137).

For the axes of the positioner (X, Y, Z, U, V, W), the following is valid:

- Before the start of every motion, it is checked whether the motion platform can actually reach the commanded target position. With VMO? (p. 201) you can query whether the target position can be reached.
- Rotations take place around the center of rotation.
- When the **Trajectory Source** parameter (ID 0x19001900) is set to 1, the dynamics profile must be defined by consecutive MOV commands. MVR is not allowed.

For more information, see "Motion of the positioner" (p. 27).

Example:

Send: MOV X 0.5

Note: This is an absolute motion.

Send: POS? X

Receive: X=0.500000

Send: MOV? X

Receive: X=0.500000

Send: MVR X 2

Note: This is a relative motion.

Send: POS? X

Receive: X=2.500000

Send: MVR X 2000

Note: New target position of axis X would exceed the motion range. Command is ignored, i.e. the target position remains unchanged, and the axis does not move.

Send: MOV? X

Receive: X=2.500000

Send: POS? X

Receive: X=2.500000

NLM (Set Low Position Soft Limit)

Description: Limits the low end of the axis travel range in closed-loop operation ("soft limit").

Format: NLM {<AxisID> <LowLimit>}

Arguments: <AxisID> is one axis of the controller

<LowLimit> is the limit position for the low end of the travel range, in physical units

Response: None

Notes: For the axes of the motion platform of the positioner, the default values for NLM are specified by the enabled operating coordinate system (with the work-and-tool concept, from the combination of two operating coordinate systems).

The value set with NLM must be smaller than the current position value. Only negative values are allowed for axes X, Y, Z, U, V and W.

The soft limits are activated and deactivated with SSL (p. 188).

Soft limits can only be set when the axis is not moving (query with #5 (p. 118)).

The physical unit in which <LowLimit> is to be given can be queried with PUN? (p. 182).

When the pivot point is changed with SPI (p. 188), the soft limits for the rotational axes U, V and W are not adjusted.

Example: Send: NLM? X
 Receive: X = -22.5
 Send: POS? X
 Receive: X = -10
 Send: NLM X -5
 Send: ERR?
 Receive: 27 - (error 27 - "Soft limit out of range")
 Send: NLM? X
 Receive: X = -22.5

NLM? (Get Low Position Soft Limit)

Description: Get the position "soft limit" which determines the low end of the axis travel range in closed-loop operation.

Format: NLM? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<LowLimit> is the limit position for the low end of the travel range, in physical units.

ONT? (Get On-Target State)

Description: Gets the on-target state of the given axis.

If all arguments are omitted, gets state of all axes.

Format: ONT? [{<AxisID>}]

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

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

where

<uint> = "1" when the given axis has reached the target value, otherwise "0".

Troubleshooting: Illegal axis identifier

PLM (Set High Position Soft Limit)

Description:	Limits the high end of the axis travel range in closed-loop operation ("soft limit").
Format:	PLM {<AxisID> <HighLimit>}
Arguments:	<p><AxisID> is one axis of the controller</p> <p><HighLimit> is the limit position for the high end of the travel range, in physical units.</p>
Response:	None
Notes:	<p>For the axes of the motion platform of the positioner, the default values for PLM are specified by the enabled operating coordinate system (with the work-and-tool concept, from the combination of two operating coordinate systems).</p> <p>The value set with PLM must be greater than the current position value. Only positive values are allowed for axes X, Y, Z, U, V and W.</p> <p>The soft limits are activated and deactivated with SSL (p. 188).</p> <p>Soft limits can only be set when the axis is not moving (query with #5 (p. 118)).</p> <p>The physical unit in which <HighLimit> is to be given can be queried with PUN? (p. 182).</p> <p>When the pivot point is changed with SPI (p. 188), the soft limits for the rotational axes U, V, and W are not adjusted.</p>
Example:	<p>Send: PLM? X</p> <p>Receive: X = 22.5</p> <p>Send: POS? X</p> <p>Receive: X = 10</p> <p>Send: PLM X 5</p> <p>Send: ERR?</p> <p>Receive: 27 - (error 27 - "Soft limit out of range")</p> <p>Send: PLM? X</p> <p>Receive: X = 22.5</p>

PLM? (Get High Position Soft Limit)

Description: Gets the position "soft limit" which determines the high end of the axis travel range in closed-loop operation.

Format: PLM? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<HighLimit> is the limit position for the high end of the travel range, in physical units.

POS? (Get Real Position)

Description: Gets the current axis position.

If all arguments are omitted, the current position of all axes is queried.

Format: POS? [{<AxisID>}]

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

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

where

<float> is the current axis position in physical units.

Troubleshooting: Illegal axis identifier

Notes: This command is identical in function to #3 (p. 117) which should be preferred when the controller is performing time-consuming tasks.

The current position of axes X, Y, Z, U, V, and W is calculated from the measured positions of the individual drives.

The physical unit in which the axis position is given can be queried with PUN? (p. 182).

PUN? (Get Axis Unit)

Description:	Gets the current unit of the axis. If all arguments are omitted, the current unit for all axes is queried.
Format:	PUN? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.
Response:	{<AxisID>="<string> LF} where <string> is the current unit of the axis.
Troubleshooting:	Illegal axis identifier
Note:	The following units are used for the positions of the axes: X, Y, Z, S1, S2: Millimeters U, V, W: degrees

RBT (Reboot System)

Description:	Reboots system. The controller behaves the same as after switch-on.
Format:	RBT
Arguments:	none
Response:	none

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? (Get Reference Mode)

Description:	Gets mode of reference point definition of given axes.
Format:	RON? [{ <AxisID>}]
Arguments:	<AxisID> is one axis of the controller.
Response:	{<AxisID>="<ReferenceOn> LF}
	where
	<ReferenceOn> is the currently set mode of reference point definition for the axis
Troubleshooting:	Illegal axis identifier
Notes:	<p>RON? always returns 1. For axes, whose position is measured by incremental sensors, this means:</p> <ul style="list-style-type: none"> ▪ Motion commands are only executed after a successful reference point definition. ▪ The reference point definition must take place with a reference move. <p>With the C-886, the reference move takes place to the reference point switch (start with FRF (p. 132)).</p>

RTR? (Get Record Table Rate)

Description:	Gets the current record table rate, i.e., the number of cycles used in data recording operations.
Format:	RTR?
Arguments:	None
Response:	<RecordTableRate> is the table rate used for recording operations (unit: number of cycles).
Note:	<p>Gets the Data Recorder Table Rate parameter value (ID 0x16000000).</p> <p>For further information, see "Data Recording" (p. 72).</p>

SAI? (Get List Of Current Axis Identifiers)

Description:	Gets the axis identifiers.
	See also "Commandable Items" (p. 23).
Format:	SAI? [ALL]

Arguments: [ALL] is optional. For controllers which allow for axis deactivation, [ALL] ensures that the answer also includes the axes which are "deactivated".

Response: {<AxisID> LF}

<AxisID> is one axis of the controller.

SCT (Set Cycle Time)

Description: Determines the cycle time for executing a dynamics profile.

Used in the cyclic transfer of target positions, see also "Motion of the positioner" (p. 27).

Format: SCT "T" <CycleTime>

Arguments: "T" is the required keyword for the argument <CycleTime>

<CycleTime> is the cycle time in ms, format: float.

Troubleshooting: Permissible value range exceeded

Notes: The permissible range of values for <CycleTime> is 1 to 100000 ms; the standard value is 100 ms.

The cycle time set with SCT is only effective when the **Trajectory Source** parameter (ID 0x19001900) has the value 1 (= the dynamics profile is defined by consecutive MOV commands).

The cycle time is used to calculate the velocity during the motion so that the given points of the dynamics profile are reached precisely at the end of the time interval.

When the **Trajectory Execution** parameter (ID 0x19001901) has the value 1, the intermediate storage of the dynamics profile points guarantees that the dynamics profile is executed at the specified time intervals.

When the **Trajectory Execution** parameter (ID 0x19001901) has the value 0, the MOV commands are executed immediately after being sent.

- Make sure that the MOV commands are sent in the time intervals specified in the cycle time so that the dynamics profile can be maintained.

Example: Send: SCT T 200
Sets the cycle time for executing a dynamics profile that is defined by consecutive MOV commands to 200 ms.

SCT? (Get Cycle Time)

Description: Gets the current cycle time for running a defined dynamics profile.

Format: SCT? [<T>]

Arguments: "T" serves as a keyword and can be omitted for the query.

Response: T=<float> LF

<float> is the cycle time in ms.

SPA (Set Volatile Memory Parameters)

Description: Sets a parameter of the given item in the volatile memory (RAM) to a specific value. Parameter changes are lost when the controller is switched off or rebooted.

Format: SPA {<ItemID> <PamID> <PamValue>}

Arguments: <ItemID> is the item 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 given parameter of the given item is set.

Response: None

Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Troubleshooting: With HPA? (p. 138) you can obtain a list of the available parameters.
Illegal item identifier, wrong parameter ID,value out of range,
command level too low for write access

Parameter values can only be set when the axis is not moving (query with #5 (p. 118)).

Available item IDs and parameter IDs: The item can be an axis, a drive, or the whole system; the item type depends on the parameter. See "Parameter Overview" (p. 236) for the item type concerned; for item identifiers, see "Commandable Items" (p. 23).

Valid parameter IDs are given in "Parameter Overview" (p. 236).

Example: With the following commands, you go to command level 1 and set the name affix for the C-886 to the value *testfarm*, parameter ID written in hexadecimal format:

```
CCL 1 advanced
SPA 1 0x0D001000 testfarm
```

SPA? (Get Volatile Memory Parameters)

Description: Gets the value of a parameter of a given item from volatile memory (RAM).

With HPA? (p. 138) you can obtain a list of the available parameters.

Format: SPA? [{<ItemID> <PamID>}]

Arguments: <ItemID> is the item for which a parameter is to be queried in volatile memory. See below for details.

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

Response: {<ItemID> <PamID>="<PamValue> LF}

where

<PamValue> is the value of the given parameter for the given item

Troubleshooting: Illegal item identifier, wrong parameter ID

Available item IDs and parameter IDs: The item can be an axis, a drive, or the whole system; the item type depends on the parameter. See "Parameter Overview" (p. 236) for the item type concerned; for item identifiers, see "Commandable Items" (p. 23).

Valid parameter IDs are given in "Parameter Overview" (p. 236).

SPI (Set Pivot Point)

Description:	<p>Sets the pivot point coordinates in the volatile memory.</p> <p>Can only be set when the following holds true for the rotation coordinates of the moving platform: $U = V = W = 0$</p>
Format:	SPI {<PPCoordinate> <Position>}
Arguments:	<p><PPCoordinate> is a pivot point coordinate, see below.</p> <p><Position> is the value of the pivot point coordinate, see below.</p>
Response:	None
Troubleshooting:	At least one of the rotation coordinates U, V and W is not equal to 0
Notes:	<p><PPCoordinate> can be R, S and T. X, Y and Z can also be used as alias identifiers for R, S and T.</p> <p><Position> is given in mm. The default values are specified by the enabled operating coordinate system (only for coordinate systems of the KSF and ZERO types).</p> <p>For further information on the pivot point, see "Definition of Terms" (p. 2).</p> <p>If the pivot point is moved with SPI, the possible travel ranges for the rotational axes U, V and W change and thus the workspace as well. The following values are not adapted to the changed travel ranges, however:</p> <ul style="list-style-type: none"> ▪ Responses to TMN? (p. 193) and TMX? (p. 194) ▪ Soft limits that are set with NLM (p. 178) and PLM (p. 180) <p>Use VMO? (p. 201) to query whether a target position can be reached.</p>
Example:	<p>Send: SPI?</p> <p>Receive: R=0</p> <p>S=0</p> <p>T=0</p> <p>Send: SPI S 2</p> <p>Send: SPI?</p>

```

Receive: R=0
S=2
T=0
Send: SPI Z 2
Send: SPI?
Receive: R=0
S=2
T=2

```

SPI? (Get Pivot Point)

Description: Gets the pivot point coordinates.

Format: SPI? [{<PPCoordinate>}]

Arguments: <PPCoordinate> is a pivot point coordinate, see below

Response: {<PPCoordinate>}"="<Position> LF}

where

<Position> is the value of the pivot point coordinate in physical units.

Note: <PPCoordinate> can be R, S and T. X, Y and Z can also be used as alias identifiers for R, S and T.

SSL (Set Soft Limit)

Description: Activates or deactivates the soft limits that are set with NLM (p. 178) and PLM (p. 180).

Format: SSL {<AxisID> <SoftLimitsOn>}

Arguments: <AxisID> is one axis of the controller

<SoftLimitsOn> is the status of the soft limits:

0 = soft limits deactivated
1 = soft limits activated

Response: None

Note: For the axes of the motion platform of the positioner, the default values for SSL are specified by the enabled operating coordinate concept (with the work-and-tool system, from the combination of two operating coordinate systems).

Example: Soft limits can only be activated/deactivated when the axis is not moving (query with #5 (p. 118)).

Send: SSL X 1
The soft limits for axis X are activated.

Send: SSL Y 0 Z 1 W 1
The soft limits are deactivated for the axis Y and activated for axes Z and W.

SSL? (Get Soft Limit Status)

Description: Gets the status of the soft limits that are set with NLM (p. 178) and PLM (p. 180).

If all arguments are omitted, the status is queried for all axes.

Format: SSL? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<SoftLimitsOn> is the status of the soft limits:
0 = soft limits deactivated
1 = soft limits activated

Troubleshooting: Illegal axis identifier

SSN? (Get Device Serial Number)

Description: Gets the serial number of the C-886.

Format: SSN?

Arguments: None

Response: <SerialNumber> is the serial number of the device.

STA? (Query Status Register Value)

Description: Requests system status information.

Format: STA?

Arguments: None

Response: The answer is bit-mapped. See below for the individual codes.

Notes: This command is identical in function to #4 (p. 117).

The response is the sum of the following codes in hexadecimal format. When analyzing the response, the following must be taken into account:

- Bits 14 and 15 for the motion status of the optional single axes S1 and S2 are only set when the motion has been triggered by a command.
- Unassigned bits have the value 0.

Bit:	23	22	21	20	19	18	17	16
	-	-	-	-	Reference move is executed	Reference move for axis S2 successful	Reference move for axis S1 successful	Reference move for positioner successful
Bit:	15	14	13	12	11	10	9	8
	Axis S2 in motion	Axis S1 in motion	Drive 6 in motion	Drive 5 in motion	Drive 4 in motion	Drive 3 in motion	Drive 2 in motion	Drive 1 in motion
Bit:	7	6	5	4	3	2	1	0
	Motion error axis S2	Motion error axis S1	Motion error drive 6	Motion error drive 5	Motion error drive 4	Motion error drive 3	Motion error drive 2	Motion error drive 1

Example:

Send: STA?

Receive: 0x71804

Note: The response is given in hexadecimal format. This means: A motion error was reported for drive 3; drives 4 and 5 are in motion. The reference move of the positioner and the optional single axes S1 and S2 was completed successfully.

STE (Start Step And Response Measurement)

Description:	<p>Starts performing a step and recording the step response for the given axis.</p> <p>The data recorder configuration, i.e., the assignment of data sources and record options to the recorder tables, can be set with DRC.</p> <p>The recorded data can be read with the DRR? command (p. 128).</p>
Format:	STE <AxisID> <Amplitude>
Arguments:	<p><AxisID> is one axis of the controller</p> <p><Amplitude> is the size of the step. See below for details.</p>
Response:	None
Troubleshooting:	The target position resulting from the specified step height is out of limits.
Notes:	<p>A "step" consists of a relative move of the specified amplitude. The step is executed in relation to the current position.</p> <p>The physical unit in which <Amplitude> is to be given can be queried with PUN? (p. 182).</p> <p>For the axes of the positioner (X, Y, Z, U, V, W), the following is valid:</p> <ul style="list-style-type: none"> ▪ Before the start of every motion, it is checked whether the motion platform can actually reach the commanded target position. With VMO? (p. 201) you can query whether the target position can be reached. ▪ Rotations take place around the center of rotation. ▪ When the Trajectory Source parameter (ID 0x19001900) is set to 1, the dynamics profile must be defined by consecutive MOV commands. STE is not allowed. <p>For more information, see "Motion of the positioner" (p. 27).</p>

STP (Stop All Axes)

Description:	<p>Stops all axes abruptly. For further details, see the notes below.</p> <p>Sets error code to 10.</p> <p>This command is identical in function to #24 (p. 121).</p>
Format:	STP

Arguments:	None
Response:	None
Troubleshooting:	Communication breakdown
Notes:	<p>STP stops all axis motion that is caused by motion commands or wave generator output and stops the reference move.</p> <p>STP stops macros.</p> <p>After the axes are stopped, their target positions are set to their current positions.</p>

SVO (Set Servo Mode)

Description:	Sets the servo mode for given axes (open-loop or closed-loop operation).
Format:	SVO {<AxisID> <ServoState>}
Arguments:	<p><AxisID> is one axis of the controller</p> <p><ServoState> can have the following values: 0 = servo mode off (open-loop operation) 1 = servo mode on (closed-loop operation)</p>
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>The servo mode is always switched on or off for all axes of the motion platform of the positioner (X, Y, Z, U, V, W) together. For this reason, it is sufficient to enter a single axis, e.g.: SVO X 1, to set the servo mode for the axes of the motion platform</p> <p>Motion of the axes can only be triggered when servo mode is switched on.</p> <p>Servo mode can only be switched off when the axis is not moving (query with #5 (p. 118)).</p>

SVO? (Get Servo Mode)

Description:	Gets the servo mode for the axes specified.
	If all arguments are omitted, gets the servo mode of all axes.

Format:	SVO? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.
Response:	{<AxisID>="<ServoState> LF}
	where
	<ServoState> is the current servo mode for the axis: 0 = servo mode off (open-loop operation) 1 = servo mode on (closed-loop operation)
Troubleshooting:	Illegal axis identifier

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
Notes:	<p>The travel ranges in X, Y, Z, U, V, W are dependent on each other. Depending on the current position of the motion platform of the positioner, the actually available travel range for axes X, Y, Z, U, V, and W can be smaller than the travel range given in the response to TMN?. The response to TMN? only corresponds to the actually available travel range of an axis when the following two conditions are met:</p> <ul style="list-style-type: none"> ▪ All other axes are at zero position. ▪ The defaults for the pivot point coordinates apply.

If the pivot point is moved with SPI (p. 187), the available travel ranges for the rotational axes U, V and W change. The values for the minimum commandable positions are **not** adapted to the changed travel ranges, however.

Use VMO? (p. 201) to query whether a target position can be reached.

Use TRA? (p. 195) to get the maximum absolute position that can be commanded when the platform of the positioner moves along a specified direction vector.

The physical unit in which the minimum commandable position is given can be queried with PUN? (p. 182).

TMX? (Get Maximum Commandable Position)

Description: Get the maximum commandable position in physical units.

Format: TMX? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response {<AxisID>="<float> LF}

where

Notes: <float> is the maximum commandable position in physical units
The travel ranges in X, Y, Z, U, V, W are dependent on each other. Depending on the current position of the motion platform of the positioner, the actually available travel range for axes X, Y, Z, U, V, and W can be smaller than the travel range given in the response to TMX?. The response to TMX? only corresponds to the actually available travel range of an axis when the following two conditions are met:

- All other axes are at zero position.
- The defaults for the pivot point coordinates apply.

If the pivot point is moved with SPI (p. 187), the available travel ranges for the rotational axes U, V and W change. The values for the maximum commandable positions are **not** adapted to the changed travel ranges, however.

Use VMO? (p. 201) to query whether a target position can be reached.

Use TRA? (p. 195) to get the maximum absolute position that can be commanded when the platform of the positioner moves along a specified direction vector.

The physical unit in which the maximum commandable position is given can be queried with PUN? (p. 182).

TNR? (Get Number of Record Tables)

Description:	Gets the number of data recorder tables currently available on the controller.
Format:	TNR?
Arguments:	none
Response	<uint> is the number of data recorder tables which are currently available
Notes:	The response indicates the value of the Data Recorder Channel Number parameter (ID 0x16000300).

For more information see "Data Recording" (p. 72).

TRA? (Get Maximum Commandable Position For Direction Vector)

Description:	<p>Queries the maximum absolute position that can be commanded when the platform of the positioner moves along the direction vector that is defined by the given axis components.</p> <p>The maximum commandable position is calculated from the current position and it can be queried only if the platform of the positioner is not in motion.</p> <p>Note: "Maximum" refers to the amount of the position value. Therefore, in this description the largest possible deflection in negative direction is referred to as the "maximum" position as well (and not as the "minimum" position).</p>
Format:	TRA? {<AxisID> <Component>}
Arguments:	<p><AxisID> is one axis of the controller. Possible values: X, Y, Z, U, V, W.</p> <p><Component> is the component of the axis on the direction vector. Must be different from zero for at least one queried axis. Can have a negative sign.</p> <p>Axes not given in the query have no component on the direction vector and are not contained in the response.</p>
Response:	<p>{<AxisID>="<float> LF}</p> <p>where</p>

Notes:	<p><float> is the maximum commandable absolute position for the axis when the platform of the positioner moves along the given direction vector; in physical units.</p> <p>Included in the calculation are the current settings for the soft limits (see NLM (p. 178), PLM (p. 180), SSL (p. 188)) and for the pivot point defined with SPI (see SPI (p. 187)) if it is used by the enabled operating coordinate system.</p>
Example:	<p>The physical unit in which the maximum commandable position is given can be queried using PUN? (p. 182).</p> <p>The motion platform is to move in the direction of the vector (X, Z) = (2, 4). The PLM command sets the upper soft limit for the X axis to the value 1.</p> <p>Send:</p> <p>TRA? X 2 Z 4</p> <p>The response indicates the maximum absolute position that can be approached in the direction of the vector (X, Z) = (2, 4) from the current position:</p> <p>X=1.00000</p> <p>Z=1.99869</p>

TRS? (Indicate Reference Switch)

Description:	Indicates whether axes have a reference point switch with direction sensing.
Format:	TRS? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	<p>{<AxisID>="<uint> LF}</p> <p>where</p> <p><uint> indicates whether the axis has a direction-sensing reference point switch (=1) or not (=0).</p>
Troubleshooting:	Illegal axis identifier

TWG? (Get Number of Wave Generators)

Description:	Gets the number of wave generators available in the controller.
Format:	TWG?
Arguments:	None
Response	<uint> is the number of wave generators which are available

VAR (Set Variable Value)

Description:	<p>Sets a variable to a certain value.</p> <p>Local variables can be set using VAR in macros only. See “Variables” (p. 106) for more details on local and global variables.</p> <p>The variable is present in RAM only.</p>
Format:	VAR <Variable> <String>
Arguments:	<p><Variable> is the name of the variable whose value is to be set.</p> <p><String> is the value to which the variable is to be set. If omitted, the variable is deleted.</p> <p>The value can be given directly or via the value of a variable.</p> <p>See “Variables” (p. 106) for more details on conventions regarding variable names and values.</p>
Response:	None
Example:	It is possible to set the value of one variable (e.g. TARGET) to that of another variable (e.g. SOURCE):

```
VAR TARGET ${SOURCE}
```

Use curly brackets if the name of the variable is longer than one character:

```
VAR A ONE
VAR VARB TWO
VAR $A 1
VAR ${VARB} 2
VAR $VARB 2 // this will result in an unwanted behavior
VAR?
A=ONE
```

```

VARB=TWO
ONE=1
TWO=2 // ${VARB}: is replaced by its value "TWO"
ARB=2 // $VARB: $V is replaced by its (empty) value

```

See ADD (p. 122) for another example.

VAR? (Get Variable Values)

Description:	Gets values of variables. If VAR? is combined with CPY (p. 124), JRC (p. 143), MEX (p. 174) or WAC (p. 202), the response to VAR? has to be a single value and not more. See "Variables" (p. 106) for more details on local and global variables.
Format:	VAR? [{<Variable>}]
Arguments:	<Variable> is the name of the variable to be queried. See "Variables" (p. 106) for more details on name conventions. If <Variable> is omitted, all global variables present in the RAM are listed.
Response:	{<Variable>=" "<String>LF} where <String> gives the value to which the variable is set.
Note:	Within a macro, VAR? can only be meaningfully used in combination with CPY (p. 124), JRC (p. 143), MEX (p. 174) or WAC (p. 202).

VEL (Set Closed-Loop Velocity)

Description:	Set velocity of given axes.
Format:	VEL {<AxisID> <Velocity>}
Arguments:	<AxisID> is one axis of the controller. <Velocity> is the velocity value in physical units/s.
Response:	None
Troubleshooting:	Illegal axis identifiers

Notes: The command is only permissible for the optional single axes S1 and S2.

<Velocity> must be ≥ 0 .

The velocity set with VEL is saved in volatile memory (RAM) only.

VEL? (Get Closed-Loop Velocity)

Description: Gets the commanded velocity.

If all arguments are omitted, gets 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 in physical units per second that is commanded.

VER? (Get Versions Of Firmware And Drivers)

Description: Gets the versions of the firmware of the C-886 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.

Example: For C-886, VER? responds something like:

```
ver?
2: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117004372, 02.012
3: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117007103, 02.012
4: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117004369, 02.012
5: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117007100, 02.012
6: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117007101, 02.012
7: (c)2016 Physik Instrumente (PI) GmbH & Co.
KG, E-873.10C885, 117007102, 02.012
8: (c)2014 Physik Instrumente(PI) Karlsruhe,E-
861.11C885, 000000000, 00.012
FW: V 2.3.3.186
Macro: V 0.15.0.0
OS: V #13 SMP PREEMPT Fri Sep 29 09:53:55 CEST
2017
Hexdata: V 1.0.0.0
C886: (c)2017 Physik Instrumente(PI) Karlsruhe,
C-886.M2 Master
```

VLS (Set System Velocity)

Description: Sets the velocity for the motion platform of the positioner.

Format: VLS <SystemVelocity>

Arguments: <SystemVelocity> is the velocity value in physical units.

Response: None

Notes: The unit for <SystemVelocity> is mm/s.

The lower limit for <SystemVelocity> is a result of the smallest step size of the positioner (model-dependent) and is specified by the value of the **Minimum System Velocity** parameter (ID 0x19001501). The parameter can be queried with SPA? (p. 186).

The velocity can only be set with VLS when the positioner is not moving (axes X, Y, Z, U, V, W; get with #5 (p. 118)).

For the optional single axes S1 and S2, the velocity can be set with VEL (p. 198).

VLS? (Get System Velocity)

Description:	Gets the velocity of the motion platform of the positioner that is set with VLS (p. 200).
Format:	VLS?
Arguments:	None
Response:	<SystemVelocity> is the velocity value in physical units, see VLS.

VMO? (Virtual Move)

Description:	Checks whether the motion platform of the positioner can approach a specified position from the current position. VMO? does not trigger any motion.
Format:	VMO? {<AxisID> <Position>}
Arguments:	<AxisID> is an axis of the controller, see below <Position> is a target position value to be checked
Response:	<uint> indicates whether the motion platform can approach the position resulting from the given target position values: 0 = specified position cannot be approached 1 = specified position can be approached
Notes:	Axes X, Y, Z, U, V, W are permissible. VMO? checks the following: <ul style="list-style-type: none"> Are the nodes of the calculated profile and the target position outside of the travel range limits that can be queried with TMN? (p. 193) and TMX? (p. 194) or TRA? (p. 195)? Have the soft limits set with NLM (p. 178) and PLM (p. 180) been activated with SSL (p. 188), and if yes, are the nodes and the target position outside of these soft limits? Are the individual drives able to move the platform to the required nodes and the specified target position? <p>In order to receive a reliable response, send VMO? only after a successful reference move (start with FRF (p. 132)) and only when the positioner is not moving (get with #5 (p. 118)).</p>

WAC (Wait For Condition)

Description:	<p>Waits until a given condition of the following type occurs: a specified value is compared with a queried value according a specified rule.</p> <p>Can only be used in macros.</p> <p>See also the MEX command (p. 174).</p>
Format:	WAC <CMD?> <OP> <value>
Arguments	<p><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.</p> <p><OP> is the operator to be used. The following operators are possible: = < > >= !=</p> <p>Important: There must be a blank space before and after the operator!</p> <p><value> is the value to be compared with the response to <CMD?>.</p>
Response:	None
Example:	<p>Send:</p> <pre>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</pre> <p>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.</p>

WAV (Set Waveform Definition)

Description: Defines a waveform of given type for given wave table.

To allow a flexible definition, a waveform (wave table contents) can be built up by stringing together "segments". Each segment is defined with a separate WAV command. A segment can be added to the existing wave table contents with the <AppendWave> argument (see below). To change individual segments or to modify their order, the complete waveform must be recreated segment-by-segment.

A segment can be based on predefined "curve" shapes (see the <WaveType> argument below).

Waveforms cannot be changed while they are being output by a wave generator. Before a waveform is modified with WAV, the wave generator output from the associated wave table must be stopped first.

The waveform values are absolute values.

The duration of one output cycle for the waveform can be calculated as follows:

Output duration = cycle time of the C-886 * WTR value * number of points

where

the cycle time of the C-886 is specified by the 0x0E000200 parameter (in seconds)

the WTR (wave table rate of the wave generator) value specifies the number of C-886 cycles the output of a waveform point lasts; default is 1

Number of Points corresponds to the wave table length (sum of the lengths of all segments in this table)

For more information, see "Wave Generator" (p. 75).

Format: WAV <WaveTableID> <AppendWave> <WaveType>
<WaveTypeParameters>

Arguments:

<WaveTableID> is the wave table identifier.

<AppendWave> can be "X" or "&":

"X" clears the wave table and starts writing at the first point in the table.

"&" attaches the defined segment to the existing wave table contents in order to extend the waveform.

<WaveType> The type of curve used to define the segment. This can be one of

"PNT" (user-defined curve)

"SIN_P" (inverted cosine curve)

"RAMP" (ramp curve)

"LIN" (single scan line curve)

<WaveTypeParameters> stands for the parameters of the curve:

For "PNT":

<WaveStartPoint> <WaveLength> {<WavePoint>}

<WaveStartPoint>: The index of the starting point. Must be 1.

<WaveLength>: The number of points to be written in the wave table (= segment length).

<WavePoint>: The value of one single point.

For "SIN_P":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint>
<CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the sine curve.

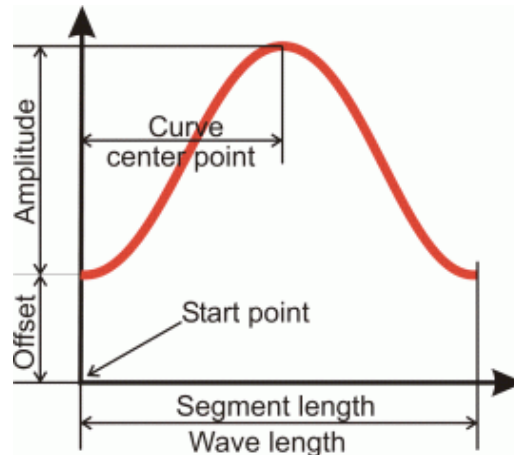
<Offset>: The offset of the sine curve.

<WaveLength>: The length of the sine curve in points.

<StartPoint>: The index of the starting point of the sine curve in the segment. Gives the phase shift. Lowest possible value is 0.

<CurveCenterPoint>: The index of the center point of the sine curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for further examples, see "Defining the Waveform" (p. 78)):



For "RAMP":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint>
<SpeedUpDown> <CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the ramp curve.

<Offset>: The offset of the ramp curve.

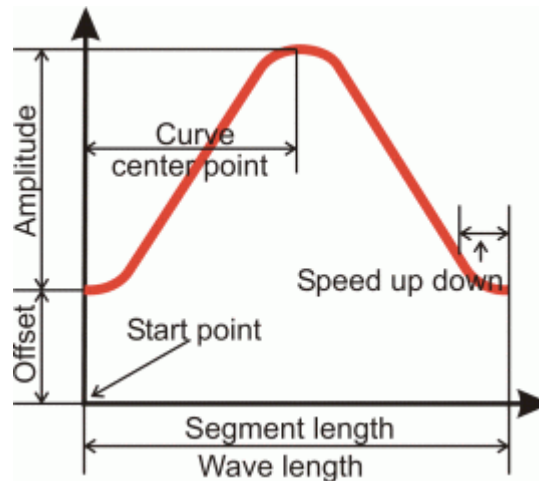
<WaveLength>: The length of the ramp curve in points.

<StartPoint>: The index of the starting point of the ramp curve in the segment. Gives the phase shift. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

<CurveCenterPoint>: The index of the center point of the ramp curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for further examples, see "Defining the Waveform" (p. 78)):



For "LIN":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint>
<SpeedUpDown>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the scan line.

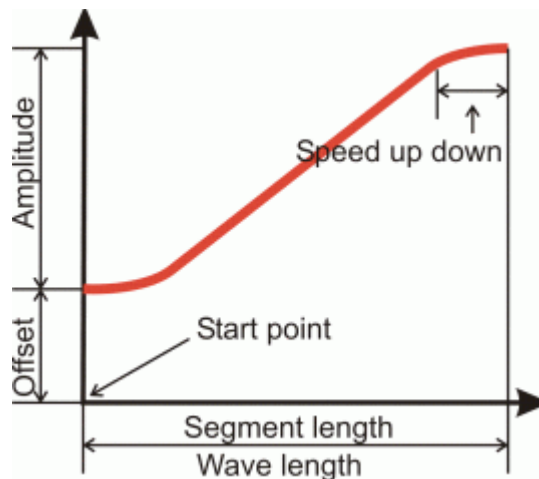
<Offset>: The offset of the scan line.

<WaveLength>: The length of the single scan line curve in points.

<StartPoint>: The index of the starting point of the scan line in the segment. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

Example (for further examples, see "Defining the Waveform" (p. 78)):



Response: None

Troubleshooting: Invalid wave table identifier

Notes: The total number of points for the waveform (which may consist of several segments) exceeds the available number of memory points.

When a waveform is defined with WAV, the target positions and the resulting velocities are not checked. The check is not performed until during the wave generator output.

The waveform influences the velocity during the motion.

The velocity is limited by the following factors, among others:

- Mechanics type
- Combination of the axes to be moved
- Current settings for coordinate system and center of rotation
- Amplitude of the motion
- Define the waveform so that the specifications of the connected mechanics are observed during the wave generator output.

WAV? (Get Waveform Definition)

Description: Gets the value of a wave parameter for a given wave table.

For more information, see "Wave Generator" (p. 75).

Format: WAV? [{<WaveTableID> <WaveParameterID>}]

Arguments: <WaveTableID> is the wave table identifier.

<WaveParameterID> is the wave parameter ID:

1 = Current wave table length as a number of points

Response: {<WaveTableID> <WaveParameterID>="<float> LF}

where

<float> depends on the <WaveParameterID>; gives the current number of waveform points in the wave table for <WaveParameterID> = 1

Troubleshooting: Invalid wave table identifier

WCL (Clear Wave Table Data)

Description: Clears the content of the given wave table.

As long as a wave generator is running, it is not possible to clear the connected wave table.

For more information, see "Wave Generator" (p. 75).

Format: WCL {<WaveTableID>}

Arguments: <WaveTableID> is the wave table identifier.

Response: None

WGC (Set Number Of Wave Generator Cycles)

Description: Sets the number of output cycles for the given wave generator (the output itself is started with WGO (p. 209)).

For more information, see "Wave Generator" (p. 75).

Format: WGC {<WaveGenID> <Cycles>}

Arguments: <WaveGenID> is the wave generator identifier

<Cycles> is the number of wave generator output cycles.

Response: None

Notes: With <WaveGenID> = 0, all wave generators are addressed.

If cycles = 0, then the waveform is output without limitation until it is stopped by WGO or #24 (p. 121) or STP (p. 191) or HLT (p. 137).

WGC? (Get Number Of Wave Generator Cycles)

Description: Gets the number of output cycles set for the given wave generator.

For more information, see "Wave Generator" (p. 75).

Format: WGC? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>="<Cycles> LF}

where

<Cycles> is the number of wave generator output cycles set with WGC (p. 208).

WGO (Set Wave Generator Start/Stop Mode)

Description: Starts and stops the specified wave generator. When the wave generator output is started, a data recording cycle automatically starts.

All wave generators whose output has to be active at the same time must be started in the same command.

For the axes of the motion platform of the positioner whose wave generator is **not** started, the last valid target position is always commanded.

The number of output cycles can be limited by WGC (p. 208).

With the WTR command (p. 215), you can lengthen the individual output cycles of the waveform.

The wave generator output is also continued when the PC software with which it was started is exited.

The #9 command can be used to query the current activation state of the wave generators. WGO? gets the last start options commanded for the wave generator. Further status information on the wave generator can be queried with WGS? (p. 211).

For more information, see "Wave Generator" (p. 75).

Format: WGO {<WaveGenID> <StartMode>}

Arguments:	<p><WaveGenID> is the wave generator identifier</p> <p><StartMode> is the start mode for the specified wave generator and can be given in hexadecimal or decimal format. Possible values:</p> <p>0: wave generator output is stopped. You can also stop the wave generator output with #24 (p. 121) or STP (p. 191) or HLT (p. 137).</p> <p>bit 0 = 0x1 (hex format) or 1 (decimal format): start wave generator output immediately, synchronized by servo cycle</p>
Response:	None
Troubleshooting:	<p>Invalid wave generator identifier</p> <p>There is no wave table connected to the wave generator. Connect a wave table with WSL (p. 214).</p> <p>When the wave generator output is active, commands for starting or configuring motion and executing corresponding macros are not permitted.</p> <p>During the wave generator output, the C-886 continuously checks whether the motion is still possible. In the following cases, the C-886 abruptly stops the motion and sets an error code:</p> <ul style="list-style-type: none"> ▪ The target positions to be output cannot be achieved. ▪ The necessary velocity cannot be achieved. ▪ The motion would cause a collision. ➤ Use the <code>WGS?</code> command to get the current status of the wave generators, especially the index of the waveform points at which an error has occurred. ➤ Get the error code of the last error that occurred with the <code>ERR?</code> command (p. 132).

WGO? (Get Wave Generator Start/Stop Mode)

Description:	<p>Gets the start/stop mode of the given wave generator.</p> <p>For more information, see "Wave Generator" (p. 75).</p>
Format:	WGO? [{<WaveGenID>}]
Arguments:	<WaveGenID> is the wave generator identifier

Response: {<WaveGenID>="<StartMode> LF}

where

<StartMode> is the last commanded start mode of the wave generator, in decimal format. For further information, see WGO (p. 209).

Notes: <StartMode> is not only 0 when WGO <WaveGenID> 0 is sent but also when the wave generator output has ended or #24 (p. 121) or STP or HLT (p. 137) has been used for stopping.

WGR (Starts Recording In Sync With Wave Generator)

Description: Starts the data recording when the wave generator is active.

For more information, see "Wave Generator" (p. 75) and "Data Recorder" (p. 72).

Format: WGR

Arguments: None

Response: None

Notes: After WGR has been sent, the data recording starts with the next output cycle of the wave generator.

The recorded data can be read with DRR? (p. 128).

Starting the wave generator output with WGO (p. 209) starts an initial data recording cycle at the same time.

For further trigger options for starting the data recording, see DRT (p. 131).

WGS? (Get Status Information On Wave Generator)

Description: Gets status information on the given wave generator.

For more information, see "Wave Generator" (p. 75).

Format: WGS? [<WaveGenID> [<ItemID>]]

Arguments: <WaveGenID> is the wave generator identifier

<ItemID> is the identifier of a variable to be queried. Possible identifiers:

STATUS

Gets the status of the wave generator.

Possible response values, in hexadecimal format:

0x0 = wave generator not active (output not running)

0x1 = Wave generator active (output running)

ITERATIONS

Gets the number of output cycles since the last start of the wave generator.

Stopping the wave generator stops the counter. The counter is reset when the wave generator is started again with the WGO command.

ERRORTYPE

Gets the error code of the last error that occurred during the output.

Possible response values:

0 = No error occurred

7 = Position out of limits

8 = Velocity out of limits

91 = Move not possible, would cause collision

ERRORINDEX

Gets the index of the waveform point at which the error occurred.

Response: {<WaveGenID> <ItemID>="<Value> LF}

where

<Value> is the current value of the queried variable.

WMS? (Get Maximum Number of Wave Table Points)

Description: Gets the number of available memory points for the given wave table.

Format: WMS? [{<WaveTableID>}]

Arguments: <WaveTableID> is the wave table identifier.

Response {<WaveTableID>="<NumberOfPoints> LF}

where

<NumberOfPoints> is the number of memory points that are available for the wave table (sum of the points already defined with WAV (p. 203) and the still unused points).

WPA (Save Parameters To Non-Volatile Memory)

Description: Writes the current settings from the volatile to the nonvolatile memory.

The settings saved with WPA are automatically loaded to the volatile memory when the C-886 is switched on or rebooted.

Note: Incorrect settings can cause the system to malfunction. Make sure that the current settings are correct before you execute the WPA command.

Settings in the volatile memory not saved using WPA will be lost when the C-886 is switched off or rebooted, or when settings are restored.

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

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

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

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

Response: None

Troubleshooting: Illegal item identifier, wrong parameter ID, invalid password

Notes: When parameter settings are to be saved:

- Parameter values can be changed in the volatile memory with the SPA command (p. 185).
- An item can be an axis, a drive, or the entire system. The item type depends on the parameter. For further information, see "Adapting Settings" (p. 233).
- With HPA? (p. 138) you can obtain a list of all available parameters. Valid parameter IDs are also given in "Parameter Overview" (p. 236).

In addition to setting the parameters, WPA can also be used to write the settings for coordinate systems to the nonvolatile memory (for details, see the table below).

Saving using WPA does **not** overwrite the default settings that can be restored using DPA (p. 126).

Note: Avoid switching off the C-886 during the WPA procedure.

Valid passwords for writing in the nonvolatile memory:

- | | |
|-----|---|
| 100 | Saves the currently valid values of all parameters and the currently valid settings for coordinate systems (for details, see the password SKS) |
| 101 | Saves the currently valid parameters. The specification of <ItemID> and <PamID> is optional. |
| SKS | <p>When the SKS password is used, <ItemID> and <PamID> are not needed.</p> <p>Saves the currently valid settings for coordinate systems:</p> <ul style="list-style-type: none"> ▪ Properties of the coordinate systems and combinations of coordinate systems which are present in the volatile memory, see KLS? and KLC? ▪ Activation state of coordinate systems, see KEN ▪ Linking of coordinate systems, see KLN <p>When ZERO is enabled: The current values for NLM, PLM, SSL, and SPI are not saved. This ensures that KEN ZERO fully re-enables the default settings for the operating coordinate system.</p> |

WSL (Set Connection Of Wave Table To Wave Generator)

- | | |
|--------------|---|
| Description: | <p>Wave table selection: connects a wave table to a wave generator or disconnects the selected generator from any wave table.</p> <p>Two or more generators can be connected to the same wave table, but a generator cannot be connected to more than one wave table.</p> <p>Deleting wave table content with WCL (p. 208) has no effect on the WSL settings.</p> <p>As long as a wave generator is running, it is not possible to change its wave table connection.</p> <p>For more information, see "Wave Generator" (p. 75).</p> |
| Format: | WSL {<WaveGenID> <WaveTableID>} |
| Arguments: | <p><WaveGenID> is the wave generator identifier</p> <p><WaveTableID> is the wave table identifier. If <WaveTableID> = 0, the selected generator is disconnected from any wave table.</p> |
| Response: | None |

WSL? (Get Connection Of Wave Table To Wave Generator)

Description: Gets current wave table connection settings for the specified wave generator.

For more information, see "Wave Generator" (p. 75).

Format: WSL? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>="<WaveTableID> LF}

where

<WaveTableID> is the wave table identifier. If <WaveTableID> = 0, no wave table is connected to the wave generator.

WTR (Set Wave Generator Table Rate)

Description: Sets wave generator table rate and interpolation type.

Format: WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}

Arguments: <WaveGenID> is the wave generator identifier. See below for details.

<WaveTableRate> is the wave generator table rate (unit: number of servo cycles); must be an integer value that is greater than zero

<InterpolationType> Available interpolation types: See below.

Response: None

Notes:

Different output rates can be set for the individual wave generators of the C-886. The output rate is set to the same value for all wave generators when <WaveGenID> has the value zero.

With the WTR command, the individual output cycles of the waveform can be lengthened. The duration of an output cycle for the waveform can be calculated as follows:

Output duration = cycle time of the C-886 * WTR value * number of points

where

the cycle time of the C-886 is specified by the 0x0E000200 parameter (in seconds)

the WTR value specifies the number of C-886 cycles the output of a waveform point lasts; default is 1

Number of Points is the length of the waveform (i.e. the length of the wave table)

WTR also sets the type of interpolation to use for the wave generator output. If the output rate of the wave table is greater than 1, interpolation helps to avoid sudden position jumps of an axis controlled by the wave generator.

For more information, see "Wave Generator" (p. 75). An application example can be found under "Configuring the Wave Generator" (p. 86).

Interpolation types available:

The following interpolation types are available:

0 = No interpolation

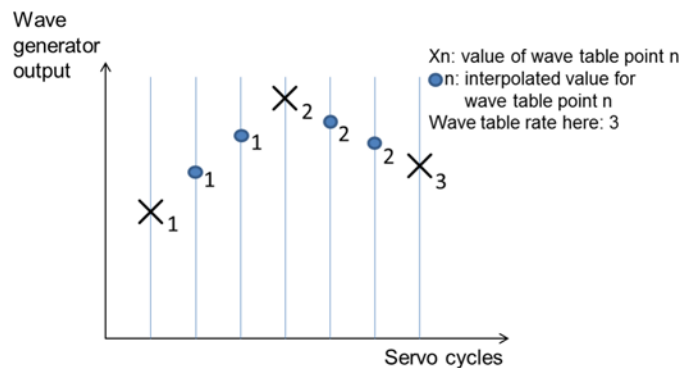
1 = Straight line (default)

Examples:

Interpolation type: Straight line (1; default)

The output rate for wave generator 5 is set to 3, with linear interpolation:

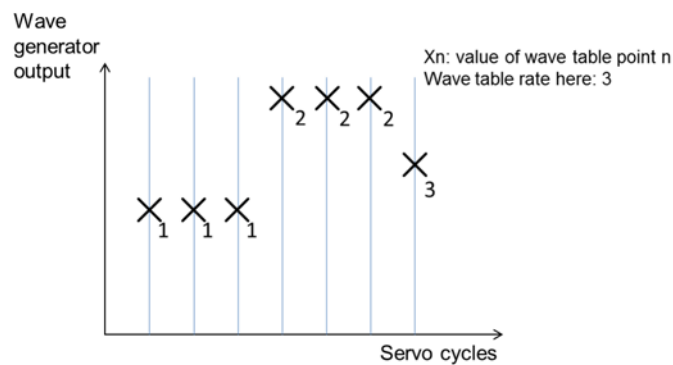
WTR 5 3 1



Interpolation type: No interpolation (0)

The output rate for wave generator 5 is set to 3, without interpolation:

WTR 5 3 0



WTR? (Get Wave Generator Table Rate)

Description: Gets the current wave generator table rate and the used interpolation type.

For more information, see "Wave Generator" (p. 75). An application example can be found under "Configuring the Wave Generator" (p. 86).

Format: WTR? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>="<WaveTableRate> <InterpolationType> LF}

where

<WaveTableRate> is the wave generator table rate (unit: number of servo cycles)

<InterpolationType> is the interpolation type applied to outputs between wave table points when the output rate is higher than 1. For available interpolation types, see WTR (p. 215).

8.5 Error Codes

The error codes listed here are those of the PI General Command Set. As such, some may be not relevant to your controller and will simply never occur.

Controller errors

0	PI_CNTR_NO_ERROR	No error
1	PI_CNTR_PARAM_SYNTAX	Parameter syntax error
2	PI_CNTR_UNKNOWN_COMMAND	Unknown command
3	PI_CNTR_COMMAND_TOO_LONG	Command length out of limits or command buffer overrun
4	PI_CNTR_SCAN_ERROR	Error while scanning
5	PI_CNTR_MOVE_WITHOUT_REF_OR_NO_SERVO	Unallowable move attempted on unreferenced axis, or move attempted with servo off
6	PI_CNTR_INVALID_SGA_PARAM	Parameter for SGA not valid
7	PI_CNTR_POS_OUT_OF_LIMITS	Position out of limits
8	PI_CNTR_VEL_OUT_OF_LIMITS	Velocity out of limits
9	PI_CNTR_SET_PIVOT_NOT_POSSIBLE	Attempt to set pivot point while U, V, and W not all 0
10	PI_CNTR_STOP	Controller was stopped by command
11	PI_CNTR_SST_OR_SCAN_RANGE	Parameter for SST or for one of the embedded scan algorithms out of range
12	PI_CNTR_INVALID_SCAN_AXES	Invalid axis combination for fast scan
13	PI_CNTR_INVALID_NAV_PARAM	Parameter for NAV out of range
14	PI_CNTR_INVALID_ANALOG_INPUT	Invalid analog channel
15	PI_CNTR_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
16	PI_CNTR_INVALID_STAGE_NAME	Unknown stage name
17	PI_CNTR_PARAM_OUT_OF_RANGE	Parameter out of range
18	PI_CNTR_INVALID_MACRO_NAME	Invalid macro name
19	PI_CNTR_MACRO_RECORD	Error while recording macro

20	PI_CNTR_MACRO_NOT_FOUND	Macro not found
21	PI_CNTR_AXIS_HAS_NO_BRAKE	Axis has no brake
22	PI_CNTR_DOUBLE_AXIS	Axis identifier specified more than once
23	PI_CNTR_ILLEGAL_AXIS	Illegal axis
24	PI_CNTR_PARAM_NR	Incorrect number of parameters
25	PI_CNTR_INVALID_REAL_NR	Invalid floating point number
26	PI_CNTR_MISSING_PARAM	Parameter missing
27	PI_CNTR_SOFT_LIMIT_OUT_OF_RANGE	Soft limit out of range
28	PI_CNTR_NO_MANUAL_PAD	No manual pad found
29	PI_CNTR_NO_JUMP	No more step-response values
30	PI_CNTR_INVALID_JUMP	No step-response values recorded
31	PI_CNTR_AXIS_HAS_NO_REFERENCE	Axis has no reference sensor
32	PI_CNTR_STAGE_HAS_NO_LIM_SWITCH	Axis has no limit switch
33	PI_CNTR_NO_RELAY_CARD	No relay card installed
34	PI_CNTR_CMD_NOT_ALLOWED_FOR_STAGE	Command not allowed for selected stage(s)
35	PI_CNTR_NO_DIGITAL_INPUT	No digital input installed
36	PI_CNTR_NO_DIGITAL_OUTPUT	No digital output configured
37	PI_CNTR_NO_MCM	No more MCM responses
38	PI_CNTR_INVALID_MCM	No MCM values recorded
39	PI_CNTR_INVALID_CNTR_NUMBER	Controller number invalid
40	PI_CNTR_NO_JOYSTICK_CONNECTED	No joystick configured
41	PI_CNTR_INVALID_EGE_AXIS	Invalid axis for electronic gearing, axis cannot be slave
42	PI_CNTR_SLAVE_POSITION_OUT_OF_RANGE	Position of slave axis is out of range
43	PI_CNTR_COMMAND_EGE_SLAVE	Slave axis cannot be commanded directly when electronic gearing is enabled
44	PI_CNTR_JOYSTICK_CALIBRATION_FAILED	Calibration of joystick failed
45	PI_CNTR_REFERENCING_FAILED	Referencing failed
46	PI_CNTR_OPM_MISSING	OPM (Optical Power Meter) missing
47	PI_CNTR_OPM_NOT_INITIALIZED	OPM (Optical Power Meter) not initialized or cannot be initialized
48	PI_CNTR_OPM_COM_ERROR	OPM (Optical Power Meter) communication error
49	PI_CNTR_MOVE_TO_LIMIT_SWITCH_FAILED	Move to limit switch failed
50	PI_CNTR_REF_WITH_REF_DISABLED	Attempt to reference axis with referencing disabled
51	PI_CNTR_AXIS_UNDER_JOYSTICK_CONTROL	Selected axis is controlled by joystick
52	PI_CNTR_COMMUNICATION_ERROR	Controller detected communication error
53	PI_CNTR_DYNAMIC_MOVE_IN_PROCESS	MOV! motion still in progress
54	PI_CNTR_UNKNOWN_PARAMETER	Unknown parameter
55	PI_CNTR_NO_REP_RECORDED	No commands were recorded with REP
56	PI_CNTR_INVALID_PASSWORD	Password invalid
57	PI_CNTR_INVALID_RECORDER_CHAN	Data record table does not exist
58	PI_CNTR_INVALID_RECORDER_SRC_OPT	Source does not exist; number too low or too high

59	PI_CNTR_INVALID_RECORDER_SRC_CHAN	Source record table number too low or too high
60	PI_CNTR_PARAM_PROTECTION	Protected Param: Current Command Level (CCL) too low
61	PI_CNTR_AUTOZERO_RUNNING	Command execution not possible while autozero is running
62	PI_CNTR_NO_LINEAR_AXIS	Autozero requires at least one linear axis
63	PI_CNTR_INIT_RUNNING	Initialization still in progress
64	PI_CNTR_READ_ONLY_PARAMETER	Parameter is read-only
65	PI_CNTR_PAM_NOT_FOUND	Parameter not found in nonvolatile memory
66	PI_CNTR_VOL_OUT_OF_LIMITS	Voltage out of limits
67	PI_CNTR_WAVE_TOO_LARGE	Not enough memory available for requested wave curve
68	PI_CNTR_NOT_ENOUGH_DDL_MEMORY	Not enough memory available for DDL table; DDL cannot be started
69	PI_CNTR_DDL_TIME_DELAY_TOO_LARGE	Time delay larger than DDL table; DDL cannot be started
70	PI_CNTR_DIFFERENT_ARRAY_LENGTH	The requested arrays have different lengths; query them separately
71	PI_CNTR_GEN_SINGLE_MODE_RESTART	Attempt to restart the generator while it is running in single step mode
72	PI_CNTR_ANALOG_TARGET_ACTIVE	Motion commands and wave generator activation are not allowed when analog target is active
73	PI_CNTR_WAVE_GENERATOR_ACTIVE	Motion commands are not allowed when wave generator is active
74	PI_CNTR_AUTOZERO_DISABLED	No sensor channel or no piezo channel connected to selected axis (sensor and piezo matrix)
75	PI_CNTR_NO_WAVE_SELECTED	Generator started (WGO) without having selected a wave table (WSL).
76	PI_CNTR_IF_BUFFER_OVERRUN	Interface buffer 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 enabled.
95	PI_CNTR_INVALID_SERVO_STATE_FOR_PARAMETER	This parameter cannot be changed in current servo mode.
96	PI_CNTR_UNKNOWN_STAGE_NAME	Unknown stage name
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 LabVIEW driver reports error. See source control for details.
200	PI_CNTR_NO_AXIS	No stage connected to axis
201	PI_CNTR_NO_AXIS_PARAM_FILE	File with axis parameters not found
202	PI_CNTR_INVALID_AXIS_PARAM_FILE	Invalid axis parameter file
203	PI_CNTR_NO_AXIS_PARAM_BACKUP	Backup file with axis parameters not found
204	PI_CNTR_RESERVED_204	PI internal error code 204
205	PI_CNTR_SMO_WITH_SERVO_ON	SMO with servo on
206	PI_CNTR_UUDECODE_INCOMPLETE_HEADER	uudecode: incomplete header
207	PI_CNTR_UUDECODE_NOTHING_TO_DECODE	uudecode: nothing to decode
208	PI_CNTR_UUDECODE_ILLEGAL_FORMAT	uudecode: illegal UUE format
209	PI_CNTR_CRC32_ERROR	CRC32 error
210	PI_CNTR_ILLEGAL_FILENAME	Illegal file name (must be 8-0 format)
211	PI_CNTR_FILE_NOT_FOUND	File not found on controller
212	PI_CNTR_FILE_WRITE_ERROR	Error writing file on controller
213	PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE	VEL command not allowed in DTR command mode
214	PI_CNTR_POSITION_UNKNOWN	Position calculations failed
215	PI_CNTR_CONN_POSSIBLY_BROKEN	The connection between controller and stage may be broken
216	PI_CNTR_ON_LIMIT_SWITCH	The connected stage has driven into a limit switch, some controllers need CLR to resume operation
217	PI_CNTR_UNEXPECTED_STRUT_STOP	Strut test command failed because of an unexpected strut stop
218	PI_CNTR_POSITION_BASED_ON_ESTIMATION	While MOV! is running position can only be estimated!
219	PI_CNTR_POSITION_BASED_ON_INTERPOLATION	Position was calculated during MOV motion

220	PI_CNTR_INTERPOLATION_FIFO_UNDERRUN	FIFO buffer underrun during interpolation
221	PI_CNTR_INTERPOLATION_FIFO_OVERFLOW	FIFO buffer underrun during interpolation
230	PI_CNTR_INVALID_HANDLE	Invalid handle
231	PI_CNTR_NO_BIOS_FOUND	No bios found
232	PI_CNTR_SAVE_SYS_CFG_FAILED	Save system configuration failed
233	PI_CNTR_LOAD_SYS_CFG_FAILED	Load system configuration failed
301	PI_CNTR_SEND_BUFFER_OVERFLOW	Send buffer overflow
302	PI_CNTR_VOLTAGE_OUT_OF_LIMITS	Voltage out of limits
303	PI_CNTR_OPEN_LOOP_MOTION_SET_WHEN_SERVO_ON	Open-loop motion attempted when servo ON
304	PI_CNTR_RECEIVING_BUFFER_OVERFLOW	Received command is too long
305	PI_CNTR_EEPROM_ERROR	Error while reading/writing EEPROM
306	PI_CNTR_I2C_ERROR	Error on I2C bus
307	PI_CNTR_RECEIVING_TIMEOUT	Timeout while receiving command
308	PI_CNTR_TIMEOUT	A lengthy operation has not finished in the expected time
309	PI_CNTR_MACRO_OUT_OF_SPACE	Insufficient space to store macro
310	PI_CNTR_EUI_OLDVERSION_CFGDATA	Configuration data has old version number
311	PI_CNTR_EUI_INVALID_CFGDATA	Invalid configuration data
333	PI_CNTR_HARDWARE_ERROR	Internal hardware error
400	PI_CNTR_WAV_INDEX_ERROR	Wave generator index error
401	PI_CNTR_WAV_NOT_DEFINED	Wave table not defined
402	PI_CNTR_WAV_TYPE_NOT_SUPPORTED	Wave type not supported
403	PI_CNTR_WAV_LENGTH_EXCEEDS_LIMIT	Wave length exceeds limit
404	PI_CNTR_WAV_PARAMETER_NR	Wave parameter number error
405	PI_CNTR_WAV_PARAMETER_OUT_OF_LIMIT	Wave parameter out of range
406	PI_CNTR_WGO_BIT_NOT_SUPPORTED	WGO command bit not supported
500	PI_CNTR_EMERGENCY_STOP_BUTTON_ACTIVATED	The \"red knob\" is still set and disables system
501	PI_CNTR_EMERGENCY_STOP_BUTTON_WAS_ACTIVATED	The \"red knob\" was activated and still disables system - reanimation required
502	PI_CNTR_REDUNDANCY_LIMIT_EXCEEDED	Position consistency check failed
503	PI_CNTR_COLLISION_SWITCH_ACTIVATED	Hardware collision sensor(s) are activated
504	PI_CNTR_FOLLOWING_ERROR	Strut following error occurred, e.g., caused by overload or encoder failure
505	PI_CNTR_SENSOR_SIGNAL_INVALID	One sensor signal is not valid
506	PI_CNTR_SERVO_LOOP_UNSTABLE	Servo loop was unstable due to wrong parameter setting and switched off to avoid damage.
507	PI_CNTR_LOST_SPI_SLAVE_CONNECTION	Digital connection to external SPI slave device is lost
508	PI_CNTR_MOVE_ATTEMPT_NOT_PERMITTED	Move attempt not permitted due to customer or limit settings
509	PI_CNTR_TRIGGER_EMERGENCY_STOP	Emergency stop caused by trigger input
530	PI_CNTR_NODE_DOES_NOT_EXIST	A command refers to a node that does not exist

531	PI_CNTR_PARENT_NODE_DOES_NOT_EXIST	A command refers to a node that has no parent node
532	PI_CNTR_NODE_IN_USE	Attempt to delete a node that is in use
533	PI_CNTR_NODE_DEFINITION_IS_CYCLIC	Definition of a node is cyclic
536	PI_CNTR_HEXAPOD_IN_MOTION	Transformation cannot be defined as long as Hexapod is in motion
537	PI_CNTR_TRANSFORMATION_TYPE_NOT_SUPPORTED	Transformation node cannot be activated
539	PI_CNTR_NODE_PARENT_IDENTICAL_TO_CHILD	A node cannot be linked to itself
540	PI_CNTR_NODE_DEFINITION_INCONSISTENT	Node definition is erroneous or not complete (replace or delete it)
542	PI_CNTR_NODES_NOT_IN_SAME_CHAIN	The nodes are not part of the same chain
543	PI_CNTR_NODE_MEMORY_FULL	Unused nodes must be deleted before new nodes can be stored
544	PI_CNTR_PIVOT_POINT_FEATURE_NOT_SUPPORTED	With some transformations pivot point usage is not supported
545	PI_CNTR_SOFTLIMITS_INVALID	Soft limits invalid due to changes in coordinate system
546	PI_CNTR_CS_WRITE_PROTECTED	Coordinate system is write protected
547	PI_CNTR_CS_CONTENT_FROM_CONFIG_FILE	Coordinate system cannot be changed because its content is loaded from a configuration file
548	PI_CNTR_CS_CANNOT_BE_LINKED	Coordinate system may not be linked
549	PI_CNTR_KSB_CS_ROTATION_ONLY	A KSB-type coordinate system can only be rotated by multiples of 90 degrees
551	PI_CNTR_CS_DATA_CANNOT_BE_QUERIED	This query is not supported for this coordinate system type
552	PI_CNTR_CS_COMBINATION_DOES_NOT_EXIST	This combination of work-and-tool coordinate systems does not exist
553	PI_CNTR_CS_COMBINATION_INVALID	The combination must consist of one work and one tool coordinate system
554	PI_CNTR_CS_TYPE_DOES_NOT_EXIST	This coordinate system type does not exist
555	PI_CNTR_UNKNOWN_ERROR	BasMac: unknown controller error
556	PI_CNTR_CS_TYPE_NOT_ACTIVATED	No coordinate system of this type is activated
557	PI_CNTR_CS_NAME_INVALID	Name of coordinate system is invalid
558	PI_CNTR_CS_GENERAL_FILE_MISSING	File with stored CS systems is missing or erroneous
559	PI_CNTR_CS_LEVELING_FILE_MISSING	File with leveling CS is missing or erroneous
601	PI_CNTR_NOT_ENOUGH_MEMORY	Not enough memory
602	PI_CNTR_HW_VOLTAGE_ERROR	Hardware voltage error
603	PI_CNTR_HW_TEMPERATURE_ERROR	Hardware temperature out of range
604	PI_CNTR_POSITION_ERROR_TOO_HIGH	Position error of any axis in the system is too high
606	PI_CNTR_INPUT_OUT_OF_RANGE	Maximum value of input signal has been exceeded
607	PI_CNTR_NO_INTEGER	Value is not integer

608	PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_RUNNING	Fast alignment process cannot be paused because it is not running
609	PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_PAUSED	Fast alignment process cannot be restarted/resumed because it is not paused
650	PI_CNTR_UNABLE_TO_SET_PARAM_WITH_SPA	Parameter could not be set with SPA - SEP needed?
651	PI_CNTR_PHASE_FINDING_ERROR	Phase finding error
652	PI_CNTR_SENSOR_SETUP_ERROR	Sensor setup error
653	PI_CNTR_SENSOR_COMM_ERROR	Sensor communication error
654	PI_CNTR_MOTOR_AMPLIFIER_ERROR	Motor amplifier error
655	PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_I2T	Overcurrent protection triggered by I2T-module
656	PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_AMP_MODULE	Overcurrent protection triggered by amplifier module
657	PI_CNTR_SAFETY_STOP_TRIGGERED	Safety stop triggered
658	PI_SENSOR_OFF	Sensor off?
700	PI_CNTR_COMMAND_NOT_ALLOWED_IN_EXTERNAL_MODE	Command not 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 cycle time invalid
720	PI_CNTR_BRAKE_ACTIVATED	Brake is activated
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 answer was received while executing WAC/MEX/JRC/...

1011	PI_CMD_NOT_VALID_IN_MACRO_MODE	Command not valid during macro execution
1024	PI_CNTR_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
1025	PI_CNTR_MAX_MOTOR_OUTPUT_REACHED	Maximum motor output reached
1063	PI_CNTR_EXT_PROFILE_UNALLOWED_CMD	User profile mode: command is not allowed, check for required preparatory commands
1064	PI_CNTR_EXT_PROFILE_EXPECTING_MOTION_ERROR	User profile mode: first target position in user profile is too far from current position
1065	PI_CNTR_PROFILE_ACTIVE	Controller is (already) in user profile mode
1066	PI_CNTR_PROFILE_INDEX_OUT_OF_RANGE	User profile mode: block or data set index out of allowed range
1071	PI_CNTR_PROFILE_OUT_OF_MEMORY	User profile mode: out of memory
1072	PI_CNTR_PROFILE_WRONG_CLUSTER	User profile mode: cluster is not assigned to this axis
1073	PI_CNTR_PROFILE_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
1090	PI_CNTR_TOO_MANY_TCP_CONNECTIONS_OPEN	There are too many open tcpip connections
2000	PI_CNTR_ALREADY_HAS_SERIAL_NUMBER	Controller already has a serial number
4000	PI_CNTR_SECTOR_ERASE_FAILED	Sector erase failed
4001	PI_CNTR_FLASH_PROGRAM_FAILED	Flash program failed
4002	PI_CNTR_FLASH_READ_FAILED	Flash read failed
4003	PI_CNTR_HW_MATCHCODE_ERROR	HW match code missing/invalid
4004	PI_CNTR_FW_MATCHCODE_ERROR	FW match code missing/invalid
4005	PI_CNTR_HW_VERSION_ERROR	HW version missing/invalid
4006	PI_CNTR_FW_VERSION_ERROR	FW version missing/invalid
4007	PI_CNTR_FW_UPDATE_ERROR	FW update failed
4008	PI_CNTR_FW_CRC_PAR_ERROR	FW Parameter CRC wrong
4009	PI_CNTR_FW_CRC_FW_ERROR	FW CRC wrong
5000	PI_CNTR_INVALID_PCC_SCAN_DATA	PicoCompensation scan data is not valid
5001	PI_CNTR_PCC_SCAN_RUNNING	PicoCompensation is running, some actions cannot be performed during scanning/recording
5002	PI_CNTR_INVALID_PCC_AXIS	Given axis can not be defined as PPC axis
5003	PI_CNTR_PCC_SCAN_OUT_OF_RANGE	Defined scan area is larger than the travel range
5004	PI_CNTR_PCC_TYPE_NOT_EXISTING	Given PicoCompensation type is not defined
5005	PI_CNTR_PCC_PAM_ERROR	PicoCompensation parameter error
5006	PI_CNTR_PCC_TABLE_ARRAY_TOO_LARGE	PicoCompensation table is larger than maximum table length
5100	PI_CNTR_NEXLINE_ERROR	Common error in NEXLINE® firmware module
5101	PI_CNTR_CHANNEL_ALREADY_USED	Output channel for NEXLINE® cannot be redefined for other usage
5102	PI_CNTR_NEXLINE_TABLE_TOO_SMALL	Memory for NEXLINE® signals is too small

5103	PI_CNTR_RNP_WITH_SERVO_ON	RNP cannot be executed if axis is in closed loop
5104	PI_CNTR_RNP_NEEDED	Relax procedure (RNP) needed
5200	PI_CNTR_AXIS_NOT_CONFIGURED	Axis must be configured for this action
5300	PI_CNTR_FREQU_ANALYSIS_FAILED	Frequency analysis failed
5301	PI_CNTR_FREQU_ANALYSIS_RUNNING	Another frequency analysis is running
6000	PI_CNTR_SENSOR_ABS_INVALID_VALUE	Invalid preset value of absolute sensor
6001	PI_CNTR_SENSOR_ABS_WRITE_ERROR	Error while writing to sensor
6002	PI_CNTR_SENSOR_ABS_READ_ERROR	Error while reading from sensor
6003	PI_CNTR_SENSOR_ABS_CRC_ERROR	Checksum error of absolute sensor
6004	PI_CNTR_SENSOR_ABS_ERROR	General error of absolute sensor
6005	PI_CNTR_SENSOR_ABS_OVERFLOW	Overflow of absolute sensor position

Interface errors

0	COM_NO_ERROR	No error occurred during function call
-1	COM_ERROR	Error during com operation (could not be specified)
-2	SEND_ERROR	Error while sending data
-3	REC_ERROR	Error while receiving data
-4	NOT_CONNECTED_ERROR	Not connected (no port with given ID open)
-5	COM_BUFFER_OVERFLOW	Buffer overflow
-6	CONNECTION_FAILED	Error while opening port
-7	COM_TIMEOUT	Timeout error
-8	COM_MULTILINE_RESPONSE	There are more lines waiting in buffer
-9	COM_INVALID_ID	There is no interface or DLL handle with the given ID
-10	COM_NOTIFY_EVENT_ERROR	Event/message for notification could not be opened
-11	COM_NOT_IMPLEMENTED	Function not supported by this interface type
-12	COM_ECHO_ERROR	Error while sending "echoed" data
-13	COM_GPIB_EDVR	IEEE488: System error
-14	COM_GPIB_ECIC	IEEE488: Function requires GPIB board to be CIC
-15	COM_GPIB_ENOL	IEEE488: Write function detected no listeners
-16	COM_GPIB_EADR	IEEE488: Interface board not addressed correctly
-17	COM_GPIB_EARG	IEEE488: Invalid argument to function call
-18	COM_GPIB_ESAC	IEEE488: Function requires GPIB board to be SAC
-19	COM_GPIB_EABO	IEEE488: I/O operation aborted
-20	COM_GPIB_ENEB	IEEE488: Interface board not found
-21	COM_GPIB_EDMA	IEEE488: Error performing DMA
-22	COM_GPIB_EOIP	IEEE488: I/O operation started before previous operation completed

-23	COM_GPIB_ECAP	IEEE488: No capability for intended operation
-24	COM_GPIB_EFSO	IEEE488: File system operation error
-25	COM_GPIB_EBUS	IEEE488: Command error during device call
-26	COM_GPIB_ESTB	IEEE488: Serial poll-status byte lost
-27	COM_GPIB_ESRQ	IEEE488: SRQ remains asserted
-28	COM_GPIB_ETAB	IEEE488: Return buffer full
-29	COM_GPIB_ELCK	IEEE488: Address or board locked
-30	COM_RS_INVALID_DATA_BITS	RS-232: 5 data bits with 2 stop bits is an invalid combination, as is 6, 7, or 8 data bits with 1.5 stop bits
-31	COM_ERROR_RS_SETTINGS	RS-232: Error configuring the COM port
-32	COM_INTERNAL_RESOURCES_ERROR	Error dealing with internal system resources (events, threads, ...)
-33	COM_DLL_FUNC_ERROR	A DLL or one of the required functions could not be loaded
-34	COM_FTDIUSB_INVALID_HANDLE	FTDIUSB: invalid handle
-35	COM_FTDIUSB_DEVICE_NOT_FOUND	FTDIUSB: device not found
-36	COM_FTDIUSB_DEVICE_NOT_OPENED	FTDIUSB: device not opened
-37	COM_FTDIUSB_IO_ERROR	FTDIUSB: IO error
-38	COM_FTDIUSB_INSUFFICIENT_RESOURCES	FTDIUSB: insufficient resources
-39	COM_FTDIUSB_INVALID_PARAMETER	FTDIUSB: invalid parameter
-40	COM_FTDIUSB_INVALID_BAUD_RATE	FTDIUSB: invalid baud rate
-41	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_ERASE	FTDIUSB: device not opened for erase
-42	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_WRITE	FTDIUSB: device not opened for write
-43	COM_FTDIUSB_FAILED_TO_WRITE_DEVICE	FTDIUSB: failed to write device
-44	COM_FTDIUSB_EEPROM_READ_FAILED	FTDIUSB: EEPROM read failed
-45	COM_FTDIUSB_EEPROM_WRITE_FAILED	FTDIUSB: EEPROM write failed
-46	COM_FTDIUSB_EEPROM_ERASE_FAILED	FTDIUSB: EEPROM erase failed
-47	COM_FTDIUSB_EEPROM_NOT_PRESENT	FTDIUSB: EEPROM not present
-48	COM_FTDIUSB_EEPROM_NOT_PROGRAMMED	FTDIUSB: EEPROM not programmed
-49	COM_FTDIUSB_INVALID_ARGS	FTDIUSB: invalid arguments
-50	COM_FTDIUSB_NOT_SUPPORTED	FTDIUSB: not supported
-51	COM_FTDIUSB_OTHER_ERROR	FTDIUSB: other error
-52	COM_PORT_ALREADY_OPEN	Error while opening the COM port: was already open
-53	COM_PORT_CHECKSUM_ERROR	Checksum error in received data from COM port
-54	COM_SOCKET_NOT_READY	Socket not ready, you should call the function again
-55	COM_SOCKET_PORT_IN_USE	Port is used by another socket
-56	COM_SOCKET_NOT_CONNECTED	Socket not connected (or not valid)
-57	COM_SOCKET_TERMINATED	Connection terminated (by peer)
-58	COM_SOCKET_NO_RESPONSE	Can't connect to peer
-59	COM_SOCKET_INTERRUPTED	Operation was interrupted by a nonblocked signal

-60	COM_PCI_INVALID_ID	No device with this ID is present
-61	COM_PCI_ACCESS_DENIED	Driver could not be opened (on Vista: run as administrator!)
-62	COM_SOCKET_HOST_NOT_FOUND	Host not found
-63	COM_DEVICE_CONNECTED	Device already connected

DLL errors

-1001	PI_UNKNOWN_AXIS_IDENTIFIER	Unknown axis identifier
-1002	PI_NR_NAV_OUT_OF_RANGE	Number for NAV out of range--must be in [1.10000]
-1003	PI_INVALID_SGA	Invalid value for SGA--must be one of 1, 10, 100, 1000
-1004	PI_UNEXPECTED_RESPONSE	Controller sent unexpected response
-1005	PI_NO_MANUAL_PAD	No manual control pad installed, calls to SMA and related commands are not allowed
-1006	PI_INVALID_MANUAL_PAD_KNOB	Invalid number for manual control pad knob
-1007	PI_INVALID_MANUAL_PAD_AXIS	Axis not currently controlled by a manual control pad
-1008	PI_CONTROLLER_BUSY	Controller is busy with some lengthy operation (e.g., reference move, fast scan algorithm)
-1009	PI_THREAD_ERROR	Internal error--could not start thread
-1010	PI_IN_MACRO_MODE	Controller is (already) in macro mode--command not valid in macro mode
-1011	PI_NOT_IN_MACRO_MODE	Controller not in macro mode--command not valid unless macro mode active
-1012	PI_MACRO_FILE_ERROR	Could not open file to write or read macro
-1013	PI_NO_MACRO_OR_EMPTY	No macro with given name on controller, or macro is empty
-1014	PI_MACRO_EDITOR_ERROR	Internal error in macro editor
-1015	PI_INVALID_ARGUMENT	One or more arguments given to function is invalid (empty string, index out of range, ...)
-1016	PI_AXIS_ALREADY_EXISTS	Axis identifier is already in use by a connected stage
-1017	PI_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
-1018	PI_COM_ARRAY_ERROR	Could not access array data in COM server
-1019	PI_COM_ARRAY_RANGE_ERROR	Range of array does not fit the number of parameters
-1020	PI_INVALID_SPA_CMD_ID	Invalid parameter ID given to SPA or SPA?
-1021	PI_NR_AVG_OUT_OF_RANGE	Number for AVG out of range--must be >0
-1022	PI_WAV_SAMPLES_OUT_OF_RANGE	Incorrect number of samples given to WAV
-1023	PI_WAV_FAILED	Generation of wave failed

-1024	PI_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
-1025	PI_RUNNING_MACRO	Controller is (already) running a macro
-1026	PI_PZT_CONFIG_FAILED	Configuration of PZT stage or amplifier failed
-1027	PI_PZT_CONFIG_INVALID_PARAMS	Current settings are not valid for desired configuration
-1028	PI_UNKNOWN_CHANNEL_IDENTIFIER	Unknown channel identifier
-1029	PI_WAVE_PARAM_FILE_ERROR	Error while reading/writing wave generator parameter file
-1030	PI_UNKNOWN_WAVE_SET	Could not find description of wave form. Maybe WG.INI is missing?
-1031	PI_WAVE_EDITOR_FUNC_NOT_LOADED	The WGWaveEditor DLL function was not found at startup
-1032	PI_USER_CANCELLED	The user cancelled a dialog
-1033	PI_C844_ERROR	Error from C-844 Controller
-1034	PI_DLL_NOT_LOADED	DLL necessary to call function not loaded, or function not found in DLL
-1035	PI_PARAMETER_FILE_PROTECTED	The open parameter file is protected and cannot be edited
-1036	PI_NO_PARAMETER_FILE_OPENED	There is no parameter file open
-1037	PI_STAGE_DOES_NOT_EXIST	Selected stage does not exist
-1038	PI_PARAMETER_FILE_ALREADY_OPENED	There is already a parameter file open. Close it before opening a new file
-1039	PI_PARAMETER_FILE_OPEN_ERROR	Could not open parameter file
-1040	PI_INVALID_CONTROLLER_VERSION	The version of the connected controller is invalid
-1041	PI_PARAM_SET_ERROR	Parameter could not be set with SPA-- parameter not defined for this controller!
-1042	PI_NUMBER_OF_POSSIBLE_WAVES_EXCEEDED	The maximum number of wave definitions has been exceeded
-1043	PI_NUMBER_OF_POSSIBLE_GENERATORS_EXCEEDED	The maximum number of wave generators has been exceeded
-1044	PI_NO_WAVE_FOR_AXIS_DEFINED	No wave defined for specified axis
-1045	PI_CANT_STOP_OR_START_WAV	Wave output to axis already stopped/started
-1046	PI_REFERENCE_ERROR	Not all axes could be referenced
-1047	PI_REQUIRED_WAVE_NOT_FOUND	Could not find parameter set required by frequency relation
-1048	PI_INVALID_SPP_CMD_ID	Command ID given to SPP or SPP? is not valid
-1049	PI_STAGE_NAME_ISNT_UNIQUE	A stage name given to CST is not unique
-1050	PI_FILE_TRANSFER_BEGIN_MISSING	A uuencoded file transferred did not start with "begin" followed by the proper filename
-1051	PI_FILE_TRANSFER_ERROR_TEMP_FILE	Could not create/read file on host PC
-1052	PI_FILE_TRANSFER_CRC_ERROR	Checksum error when transferring a file to/from the controller
-1053	PI_COULDNT_FIND_PISTAGES_DAT	The PiStages.dat database could not be found. This file is required to connect a

-1054	PI_NO_WAVE_RUNNING	stage with the CST command 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

-1078	PI_CANNOT_WRITE_TO_PARAM_DAT_FILE	are available at www.pi.ws . Cannot write to parameter DAT file to store user defined stage type.
-1079	PI_CANNOT_CREATE_PARAM_DAT_FILE	Cannot create parameter DAT file to store user defined stage type.
-1080	PI_PARAM_DAT_FILE_INVALID_REVISION	Parameter DAT file does not have correct revision.
-1081	PI_USERSTAGES_DAT_FILE_INVALID_REVISION	User stages DAT file does not have correct revision.
-1082	PI_SOFTWARE_TIMEOUT	Timeout Error. Some lengthy operation did not finish within expected time.
-1083	PI_WRONG_DATA_TYPE	A function argument has an unexpected data type.
-1084	PI_DIFFERENT_ARRAY_SIZES	Length of data arrays is different.
-1085	PI_PARAM_NOT_FOUND_IN_PARAM_DAT_FILE	Parameter value not found in parameter DAT file.
-1086	PI_MACRO_RECORDING_NOT_ALLOWED_IN_THIS_MODE	Macro recording is not allowed in this mode of operation.
-1087	PI_USER_CANCELLED_COMMAND	Command cancelled by user input.
-1088	PI_TOO_FEW_GCS_DATA	Controller sent too few GCS data sets
-1089	PI_TOO_MANY_GCS_DATA	Controller sent too many GCS data sets
-1090	PI_GCS_DATA_READ_ERROR	Communication error while reading GCS data
-1091	PI_WRONG_NUMBER_OF_INPUT_ARGUMENTS	Wrong number of input arguments.
-1092	PI_FAILED_TO_CHANGE_CCL_LEVEL	Change of command level has failed.
-1093	PI_FAILED_TO_SWITCH_OFF_SERVO	Switching off the servo mode has failed.
-1094	PI_FAILED_TO_SET_SINGLE_PARAMETER_WHILE_PERFORMING_CST	A parameter could not be set while performing CST: CST was not performed (parameters remain unchanged).
-1095	PI_ERROR_CONTROLLER_REBOOT	Connection could not be reestablished after reboot.
-1096	PI_ERROR_AT_QHPA	Sending HPA? or receiving the response has failed.
-1097	PI_QHPA_NONCOMPLIANT_WITH_GCS	HPA? response does not comply with GCS2 syntax.
-1098	PI_FAILED_TO_READ_QSPA	Response to SPA? could not be received. Response to SPA? could not be received.
-1099	PI_PAM_FILE_WRONG_VERSION	Version of PAM file cannot be handled (too old or too new)
-1100	PI_PAM_FILE_INVALID_FORMAT	PAM file does not contain required data in PAM-file format
-1101	PI_INCOMPLETE_INFORMATION	Information does not contain all required data
-1102	PI_NO_VALUE_AVAILABLE	No value for parameter available
-1103	PI_NO_PAM_FILE_OPEN	No PAM file is open
-1104	PI_INVALID_VALUE	Invalid value
-1105	PI_UNKNOWN_PARAMETER	Unknown parameter
-1106	PI_RESPONSE_TO_QSEP_FAILED	Response to SEP? could not be received.

-1107	PI_RESPONSE_TO_QSPA_FAILED	Response to SPA? could not be received. Response to SPA? could not be received.
-1108	PI_ERROR_IN_CST_VALIDATION	Error while performing CST: One or more parameters were not set correctly.
-1109	PI_ERROR_PAM_FILE_HAS_DUPLICATE_ENTRY_WITH_DIFFERENT_VALUES	PAM file has duplicate entry with different values.
-1110	PI_ERROR_FILE_NO_SIGNATURE	File has no signature
-1111	PI_ERROR_FILE_INVALID_SIGNATURE	File has invalid signature
-10000	PI_PARAMETER_DB_INVALID_STAGE_TYPE_FORMAT	PI stage database: String containing stage type and description has invalid format.
-10001	PI_PARAMETER_DB_SYSTEM_NOT_AVAILABLE	PI stage database: Database does not contain the selected stage type for the connected controller.
-10002	PI_PARAMETER_DB_FAILED_TO_ESTABLISH_CONNECTION	PI stage database: Establishing the connection has failed.
-10003	PI_PARAMETER_DB_COMMUNICATION_ERROR	PI stage database: Communication was interrupted (e.g. because database was deleted).
-10004	PI_PARAMETER_DB_ERROR_WHILE_QUERYING_PARAMETERS	PI stage database: Querying data failed.
-10005	PI_PARAMETER_DB_SYSTEM_ALREADY_EXISTS	PI stage database: System already exists. Rename stage and try again.
-10006	PI_PARAMETER_DB_QHPA_CONTAINS_UNKNOWN_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.

9 Adapting Settings

In this Chapter

Overview of the Settings of the C-886.....	233
Changing Parameter Values of the C-886 Master Module.....	234
Saving Parameter Values in a Text File	235
Parameter Overview	236

9.1 Overview of the Settings of the C-886

Various settings can be stored as default values for the C-886 in the nonvolatile memory so that they are preserved when the C-886 is switched off or rebooted.

Settings of the master module

The settings of the master module determine central characteristics and functions of the system. The default values for the settings of the master module come from different sources:

- Configuration files: Only accessible for the customer service department (p. 251)
- Interface parameters: Adaptation in the nonvolatile memory possible with the **IFS** command (p. 140), further information in "Establishing Communication via the TCP/IP Interface" (p. 53) and "Establishing Communication via the USB Interface" (p. 60)
- Commands for work with user-defined coordinate systems: Further information in "Coordinate Systems" (p. 32)
- Parameters that are changed with the **SPA** (p. 185) command and saved with the **WPA** (p. 213) command. These 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 (p. 123).

Further information in "Changing Parameter Values of the C-886 Master Module" (p. 234).

Settings of the slave modules for the parallel-kinematic positioner

The settings of the slave modules for the parallel-kinematic positioner are configured for the positioner used before delivery of the C-886. The settings of the modules may be accessed only after consultation with PI.

Settings of the slave modules for the optional single axes

The settings of the slave modules for the optional single axes can be configured by the user; for further information, see "Configuring Slave Modules for Single Axes" (p. 62) and the manuals of the corresponding modules (p. 6).

9.2 Changing Parameter Values of the C-886 Master Module

Every parameter is present in the volatile as well as in the nonvolatile memory of the C-886 master module. The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-886. The values in the volatile memory determine the current behavior of the system.

Determining available parameters

- Send the `HPA?` command (p. 138) to obtain a list of all available parameters of the master module with a short description.

Changing parameter values in the volatile memory

INFORMATION

Changing parameter values can cause undesirable results.

- Create a backup copy on the PC before changing the parameter settings of the C-886; see "Saving Parameter Values in a Text File" (p. 235). You can then restore the original settings at any time.
-
- Send the `SPA?` command (p. 186) to obtain a list of parameter values in the volatile memory of the master module.
 - Change the parameter values in the volatile memory:
 - a) If necessary for write access to the parameter values, send the `CCL 1 advanced` command to go to command level 1.
 - b) Change the parameter values in the volatile memory of the master module with the `SPA` command (p. 185).

Writing parameter values from the volatile memory to the nonvolatile memory

INFORMATION

To save parameter values in the nonvolatile memory with the `WPA` command, it is necessary to enter a password. Usable passwords:

- | | |
|-----|--|
| 100 | Saves the currently valid values of all parameters and the currently valid settings for coordinate systems |
| 101 | Saves the currently valid values of all parameters. Parameters can be selected individually. |
| SKS | Saves the currently valid settings for coordinate systems |

- Write the current parameter values to the nonvolatile memory of the master module with the `WPA` command (p. 213).

Alternative procedure if you are working with PIMikroMove:

- a) In the main window of PIMikroMove, select the **C-886 > Save parameters to nonvolatile memory** menu item. The **Save Parameters to Non-Volatile Memory** dialog opens.
- b) In the selection field of the **Save Parameters to Non-Volatile Memory** dialog, either enter the corresponding password or select the corresponding entry.
- c) Click **OK** to save and close the dialog.

9.3 Saving Parameter Values in a Text File

INFORMATION

To save the settings of the slave modules for the optional single axes, direct communication with the slave modules is necessary; see "Configuring Slave Modules for Single Axes" (p. 62) and "Identifiers for Direct Access to the Slave Modules" (p. 24). Further information on the settings of the slave modules can be found in the manuals of the corresponding modules (p. 6).

Requirements

- ✓ You have read and understood the general notes on startup (p. 49).
- ✓ You have established communication between the C-886 and the PC with PIMikroMove or PITerminal via TCP/IP (p. 57) or USB (p. 60).

Saving parameter values in a text file

1. If you use PIMikroMove, open the window for sending commands:
 - In the main window select the **Tools > Command entry** menu item or press the **F4** key on the keyboard.

In PITerminal the main window from which commands can be sent is opened automatically after establishing communication.

2. Get the current parameter values with the `SPA?` command.

The response contains the parameter values of the C-886 master module as standard.

If you communicate (p. 62) directly with slave modules in PIMikroMove, the response will contain the parameter values for the slave module that is selected in the window for sending commands.

3. Click on the **Save...** button.

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

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

9.4 Parameter Overview

INFORMATION

The write access for the parameters of the C-886 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.

- 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. 251).

Parameter ID (hexa\decimal)	Command level for write access	Affected item type	Data type	Parameter name (Unit)	Description
0x3C	0	Positioner drive	CHAR	Stage Name	Name of the parallel-kinematic positioner for which the slave module of the drive is configured
0x72	0	System	INT	Ignore Macro Error?	Ignore macro error? Determines whether the controller macro is stopped if an error occurs when it is running. 0 = Stop macro when error occurs (default) 1 = Ignore macro error

Parameter ID (hexa\decimal)	Command level for write access	Affected item type	Data type	Parameter name (Unit)	Description
0x07030401	0	Positioner axis	INT	Behaviour After Reference Move	Behavior of the motion platform of the positioner after the reference move For details, see "Motion of the positioner" (p. 27).
0x07030402	0	Positioner axis	FLOAT	Target For Motion After Reference Move	
0x0D000700	2	System	CHAR	Device Name	Product name of the C-886
0x0D001000	1	System	CHAR	Customer Device Name	Freely selectable name affix for the C-886 Can be adapted by the user to distinguish among several C-886 in the same network or on the same PC. Is shown in the list of controllers found, e.g., when communication is established via TCP/IP.
0x0E000200	3	System	FLOAT	Servo Update Time	Cycle time of the C-886 in seconds
0x0E000900	0	System	INT	Pulse Length Factor	Multiplied by the cycle time Product is the pulse width of the impulse Default: 5
0x13000004	3	System	INT	Max Wave Points	Total number of available points for waveforms For details, see "Wave Generator" (p. 75).
0x1300010A	3	System	INT	Number Of Wave Tables	Number of wave tables for saving waveforms For details, see "Wave Generator" (p. 75).
0x16000000	3	System	INT	Data Recorder Table Rate	Record table rate (= 1)
0x16000100	3	System	INT	Max. Number of Data Recorder Channels	Maximum number of data recorder tables
0x16000200	3	System	INT	Data Recorder Max. Points	Maximum number of all points of the data recorder tables
0x16000201	0	System	INT	Data Recorder Points Per Table	Number of points per data recorder table 1 to 8192 Default: 8192

Parameter ID (hexa\decimal)	Command level for write access	Affected item type	Data type	Parameter name (Unit)	Description
0x16000300	3	System	INT	Channel Number	Number of data recorder tables
0x19001500	3	System	FLOAT	Maximum System Velocity (mm/s)	Maximum system velocity See specifications in the user manual of the positioner and "Motion of the positioner" (p. 27).
0x19001501	3	System	FLOAT	Minimum System Velocity (mm/s)	Minimum system velocity For details, see "Motion of the positioner" (p. 27).
0x19001502	3	System	FLOAT	Maximum System Acceleration (mm/s ²)	Maximum system acceleration For details, see "Motion of the positioner" (p. 27).
0x19001504	3	System	FLOAT	Path Control Step Size (mm)	Step size for calculating the dynamics profile of the platform motion For details, see "Motion of the positioner" (p. 27).
0x19001510	0	System	FLOAT	Trajectory Velocity (Phys. Unit/s)	Velocity, acceleration, and jerk for the motion platform of the positioner For details, see Profile Generator for Point-to-Point Motion" (p. 28).
0x19001511	0	System	FLOAT	Trajectory Acceleration (Phys. Unit/s)	
0x19001512		System	FLOAT	Trajectory Jerk (Phys. Unit/s)	
0x19001800	0	System	INT	Coordination Mode	Coordination mode 0 = multi-axis mode 1 = Hexapod mode (= Parallel-kinematics mode; default) The parameter value must not be changed.
0x19001900	0	System	INT	Trajectory Source	Source of the dynamics profile for MOV commands 0 = Dynamics profile is determined by profile generator (default) 1 = Dynamics profile is determined by consecutive MOV commands For details, see "Motion of the positioner" (p. 27).

Parameter ID (hexa\decimal)	Command level for write access	Affected item type	Data type	Parameter name (Unit)	Description
0x19001901	0	System	INT	Trajectory Execution	Execution of the dynamics profile 0 = Dynamics profile is executed immediately (default) 1 = Dynamics profile is stored in a buffer before execution For details, see "Cyclic Transfer of Target Positions" (p. 30).
0x19001902	3	System	INT	Maximum Number of Trajectory Points	Maximum number of dynamics profile points For details, see "Cyclic Transfer of Target Positions" (p. 30).
0x19001903	0	System	INT	Threshold for Trajectory Execution	Threshold value for executing the dynamics profile For details, see "Cyclic Transfer of Target Positions" (p. 30).
0x19001904	3	System	INT	Current Number of Trajectory Points	Shows the current number of dynamics profile points in the buffer. For details, see "Cyclic Transfer of Target Positions" (p. 30).
0x19002001	0	System	INT	Start All Hexapod Wave Generators	Start behavior of the wave generators for the axes of the motion platform of the positioner (X, Y, Z, U, V, W). For details, see "Wave Generator" (p. 75).
0x19003000	2	Positioner drive	FLOAT	Hexapod Position Vector A0_X	Position and direction vectors for the drives and joints of the positioner
0x19003001	2	Positioner drive	FLOAT	Hexapod Position Vector A0_Y	
0x19003002	2	Positioner drive	FLOAT	Hexapod Position Vector A0_Z	
0x19003010	2	Positioner drive	FLOAT	Hexapod Joint Vector A0_X	
0x19003011	2	Positioner drive	FLOAT	Hexapod Joint Vector A0_Y	
0x19003012	2	Positioner drive	FLOAT	Hexapod Joint Vector A0_Z	
0x19003100	2	Positioner drive	FLOAT	Hexapod Position Vector B0_X	

Parameter ID (hexa\decimal)	Command level for write access	Affected item type	Data type	Parameter name (Unit)	Description
0x19003101	2	Positioner drive	FLOAT	Hexapod Position Vector B0_Y	
0x19003102	2	Positioner drive	FLOAT	Hexapod Position Vector B0_Z	
0x19003110	2	Positioner drive	FLOAT	Hexapod Joint Vector B0_X	
0x19003111	2	Positioner drive	FLOAT	Hexapod Joint Vector B0_Y	
0x19003112	2	Positioner drive	FLOAT	Hexapod Joint Vector B0_Z	

10 Maintenance

In this Chapter

Cleaning the C-886 241

Updating Firmware 242

10.1 Cleaning the C-886

NOTICE



Short circuits or flashovers!

The C-886 contains electrostatically sensitive devices that can be damaged by short circuits or flashovers when cleaning fluids penetrate the case.

- Before cleaning, disconnect the C-886 from the power source by pulling the power plug.
- Prevent cleaning fluid from penetrating the case.

- When necessary, clean the C-886 case surface with a cloth lightly dampened with a mild cleanser or disinfectant.

10.2 Updating Firmware

NOTICE



Damage from unintentional position changes!

The maximum holding force when the servo mode is switched off is based on the self-locking of the drives and can be lower than the maximum load capacity when the servo mode is switched on (see manual of the positioner).

When the actual load of the positioner exceeds the maximum holding force based on the self-locking of the drives, unintentional position changes of the positioner can occur in the following cases:

- Switching off the C-886
- Rebooting the C-886
- Switching off the servo mode for the axes of the motion platform of the positioner

As a result, collisions are possible between the positioner, the load to be moved, and the surroundings. Collisions can damage the positioner, the load to be moved, or the surroundings.

- Make sure that the actual load of the motion platform of the positioner does not exceed the maximum holding force based on the self-locking of the drives before you switch off the servo mode, reboot the C-886, or switch it off.

INFORMATION

The firmware of the C-886 consists of several components that can be updated separately.

INFORMATION

The firmware of the slave modules of the C-886 **cannot** be updated with the PIFirmwareManager.

- If an update of the firmware of the slave modules is required contact our customer service department (p. 251).

INFORMATION

To update the firmware, the C-886 and PC must communicate via the TCP/IP interface.

Device identification string of the C-886

The `*IDN?` command reads out the device identification string of the C-886, which also contains the version of the firmware of the master module.

Example: `(c)2011-2017 Physik Instrumente (PI) GmbH & Co. KG,C-886,117567891,2.3.3.30`

- `C-886` device name
- `117567891` serial number of the C-886, contains the encrypted date of manufacture
- `2.3.3.30` Version of the firmware of the master module (firmware component FW)

Firmware components

The versions of the firmware components can be read out with the `VER?` command.

Example:

- Device identification strings of all slave modules of the C-886, also contain the version of the firmware of the slave module:
 - `2: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117004372, 02.012`
 - `3: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117007103, 02.012`
 - `4: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117004369, 02.012`
 - `5: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117007100, 02.012`
 - `6: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117007101, 02.012`
 - `7: (c)2016 Physik Instrumente (PI) GmbH & Co. KG, E-873.10C885, 117007102, 02.012`
 - `8: (c)2014 Physik Instrumente(PI) Karlsruhe,E-861.11C885, 000000000, 00.012`
- `FW: V 2.3.3.30` Version of the firmware of the master module
- `Macro: V 0.15.0.0` Version of the macro functionality
- `OS: V #13 SMP PREEMPT Fri Sep 29 09:53:55 CEST 2017` Version of the operating system of the C-886
- `Hexdata: V 1.0.0.0` Version of the configuration file with the geometrical data for the parallel-kinematic positioner
- `C886: (c)2017 Physik Instrumente(PI) Karlsruhe, C-885.M2 Master` Hardware of the master module

Ordering current firmware of the C-886

- Contact our customer service department (p. 251) to obtain current versions of the firmware components.

Tools and accessories for updating the firmware

- PC with Windows operating system that has been prepared as follows:
 - The PIFirmwareManager program is installed. For further information, see "Installing the PC Software" (p. 39).
 - The current files of the firmware components that you have received from our customer service department (file type "Update Package", file extension "IPK") are located in a directory on the PC.

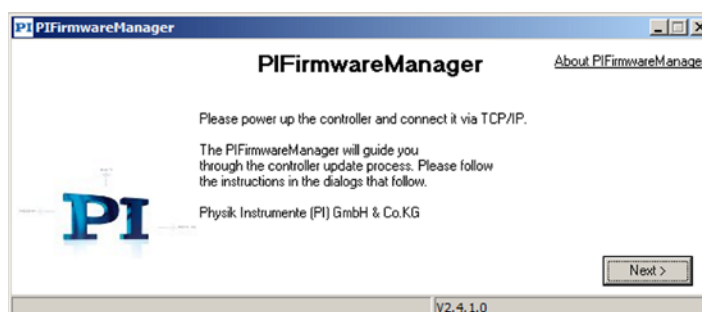
Prerequisites for updating the firmware

- ✓ You have made all necessary preparations for communication via the TCP/IP interface, see "Establishing Communication via TCP/IP in the PC-Software" (p. 57).
- ✓ The C-886 is switched on and the starting procedure of the C-886 has finished (p. 52).
- ✓ The PC is switched on.

Updating firmware components of the C-886

1. Start the PIFirmwareManager program on the PC via the **All Programs > PI > PIFirmwareManager** start menu item.

The **PIFirmwareManager** window opens.



- Click the **Next >** button.

The **Search for controllers** button is displayed.

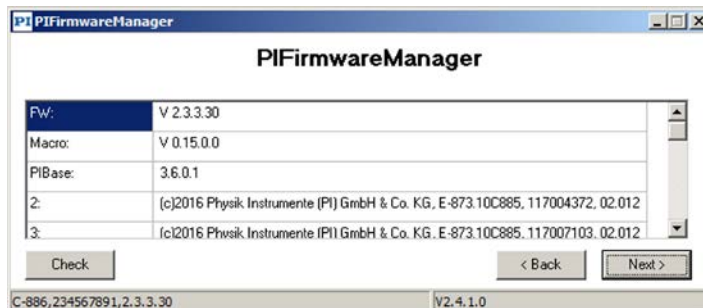


- Click the **Search for controllers** button.

The found C-886 is displayed in the **Identification:** field (serial number, port, IP address).

- Mark the line of the C-886 in the **Identification:** field.
- Click the **Next >** button.

The currently active firmware components of the C-886 as well as some of the software components installed on the PC are listed with their version numbers.



- Click the **Next >** button.

The **Add new package** button for selecting the firmware components to be updated is displayed.



- Click the **Add new package** button.

A file selection window opens.

8. In the file selection window, go to the directory that has the files that you have received from the customer service department.
9. In the file selection window, select the files for the firmware components to be updated, e.g.:
 - FW component: hexapod-firmware_c886_x.x.x.x.ipk
 - Macro component: hexapod-macro-manager_c886_x.x.x.x.ipk
10. Click the **Open** button in the file selection window to confirm the selection.
11. Click the **Update** button to start updating the selected firmware components on the C-886.

The **Information** window is displayed.



12. Click the **OK** button.

The C-886 reboots.

The updating of the firmware components is finished.

11 Troubleshooting

Problem	Possible causes	Solution
Positioner does not move or moves in an uncontrolled way	The cable is defective or is not connected correctly	➤ Check the cable connections.
	Incorrect command or incorrect syntax	➤ Send the <code>ERR?</code> command (p. 132) and check the error code that is returned.
	Incorrect configuration	<p>The configuration data used by the C-886 must be adapted to the positioner.</p> <ul style="list-style-type: none"> ➤ Use the <code>CST?</code> command (p. 124) to check to which positioner the C-886 is adapted. ➤ If the stage type in the response to <code>CST?</code> does not match the connected positioner, contact our customer service department (p. 251). Change the settings of the slave modules for the drives of the positioner only after consultation with PI.
	Error in a slave module, e.g.: <ul style="list-style-type: none"> ▪ Module overheated ▪ Drive fault ▪ Malfunction of the position sensor 	<p>In the case of faults, a slave module switches the drive off.</p> <ol style="list-style-type: none"> 1. Check the LEDs of the master module and the slave modules. For details, see "Front Panel" (p. 16). 2. Send <code>ERR?</code> command (p. 132) and check the error code that is returned. 3. Switch off the C-886 and let it cool down. 4. Check the system and make sure that all axes can be moved safely. 5. Start up the C-886 again; see "Startup" (p. 49). 6. If the fault occurs again: Contact our customer service department (p. 251).
Reduced accuracy of the positioner	Warped base plate	➤ Mount the positioner on an even surface (see user manual of the positioner).
	Wear resulting from small motion over a long period of time	➤ If only small motion is performed over a long period of time: Perform a maintenance run over the entire travel range at regular intervals.

Problem	Possible causes	Solution
Communication with the C-886 failed	Communication cable is incorrect or defective	<ul style="list-style-type: none"> ➤ Check the cable. <ul style="list-style-type: none"> – Use a straight-through network cable for the TCP/IP connection via a hub or a router (with DHCP server). – Use a cross-over network cable for direct connection with the Ethernet connection socket of the PC. ➤ If necessary, check whether the cable works on a fault-free system.
	Communication interface is not correctly configured	<p>When using the TCP/IP connection:</p> <ul style="list-style-type: none"> ➤ Connect the controller to the network before you switch it on. Otherwise, you will have to switch the controller off and on again. ➤ Check the network settings (p. 53). ➤ Make sure that the network is not blocked for unknown devices. ➤ Make sure that several PC software applications cannot access the C-886 at the same time. ➤ Make sure that you have selected the correct C-886 when establishing communication. ➤ If you cannot solve the problems, consult your network administrator.
	The starting procedure of the C-886 has not finished yet	<ol style="list-style-type: none"> 1. Wait a few seconds after switching on or rebooting the C-886. 2. Try again to establish communication or send commands.
	Another program is accessing the interface	<ul style="list-style-type: none"> ➤ Close the other program.
	Problems with special PC software	<ul style="list-style-type: none"> ➤ Check whether the system works with different PC software, such as e. g. a terminal program or a development environment. <p>You can test the communication by starting a terminal program (such as for example, PI Terminal) and entering <code>*IDN?</code> or <code>HLP?</code>.</p> <ul style="list-style-type: none"> ➤ Make sure that you end the commands with an LF (line feed). <p>A command is only executed when the LF has been received.</p>

Problem	Possible Causes	Solution
The customer software does not run with the PI drivers	Incorrect combination of driver routines/ Vis	<ul style="list-style-type: none">➤ Check whether the system works with a terminal program. If so: <ul style="list-style-type: none">➤ Read the information in the corresponding software manual and compare the sample code on the C-886.CD with your program code.

If the problem that occurred with your system is not listed in the table above or it cannot be solved as described, contact our customer service department (p. 251).

12 Customer Service

For inquiries and orders, contact your PI sales engineer or send us an email (<mailto:service@pi.de>).

- If you have questions concerning your system, have the following information ready:
 - Product and serial numbers of all products in the system
 - Firmware version of the controller (if present)
 - Version of the driver or the software (if present)
 - Operating system on the PC (if present)
- If possible: Take photographs or make videos of your system that can be sent to our customer service department if requested.

The latest versions of the user manuals are available for download (p. 7) on our website.

13 Technical Data

In this Chapter

Specifications	253
System Requirements	257
Dimensions	258
Pin Assignment	259

13.1 Specifications

13.1.1 Data Table

	C-886.1
Function	Controller for parallel-kinematic positioners
Axes	6 Optional: 2 additional single axes
Motion and control	
Drive type	DC motor Optional single-axis drive type
Motor connector	Sub-D 15 (f)
Controller type	PID controller
Servo cycle time of slave modules	50 µs
Cycle time of the C-886	10 ms
Encoder input	A/B quadrature single-ended or differential TTL signal acc. to RS-422; 60 MHz
Stall detection	Servo off, triggered by position error
Limit switches	2 × TTL per drive (polarity programmable)
Reference Point Switch	1 × TTL per drive
Characteristics of single axes	Depending on the drive type
Electrical properties	
Output voltage	0 to 24 VDC
Current limitation	3 A per drive
Characteristics of single axes	Depending on the drive type




C-886.1	
Interfaces and operation	
Communication interfaces	TCP/IP: RJ45/Ethernet; USB: Mini-USB type B
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	API for C / C++ / C# / VB.NET / MATLAB / python, drivers for LabVIEW
Supported functions	User-defined coordinate system. Startup macro. Data recorder for recording operating data. Wave generator for periodic motion.
Miscellaneous	
Operating voltage	External power supply 24 V / 10 A included in the scope of delivery
Max. current consumption	32 A
Operating temperature range	10 to 40 °C
Mass	4.4 kg without drive modules for single axes
Dimensions	482.6 mm × 132.55 mm × 278.55 mm
C-886.2	
Function	Controller for parallel-kinematic positioners
Axes	6 Optional: 2 additional single axes
Motion and control	
Drive type	2-phase stepper motor Optional single-axis drive type
Motor connector	HD Sub-D 26 (f)
Controller type	PID controller
Microstep resolution	1/2048 full step
Servo cycle time of slave modules	50 µs
Cycle time of the C-886	10 ms
Encoder input	A/B quadrature, TTL, RS-422; 60 MHz
Stall detection	Servo off, triggered by position error
Limit switches	2 × TTL per drive (polarity programmable)
Reference Point Switch	1 × TTL per drive
Index switch	1 × RS-422 per drive, for index pulse
Characteristics of single axes	Depending on the drive type

	C-886.2
Electrical properties	
Max. output voltage	48 V DC
Max. power consumption, full load	48 W per drive
Power consumption, no load	3 W per drive
Max. output power (< 2 ms)	100 W per drive
Average output power	< 48 W per drive
Current limitation per motor phase	2.5 A
Characteristics of single axes	Depending on the drive type
Interfaces and operation	
Communication interfaces	TCP/IP: RJ45/Ethernet; USB: Mini-USB type B
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	API for C / C++ / C# / VB.NET / MATLAB / python, drivers for LabVIEW
Supported functions	User-defined coordinate system. Startup macro. Data recorder for recording operating data. Wave generator for periodic motion.
Miscellaneous	
Operating voltage	External power supply 24 V / 10 A included in the scope of delivery
Max. current consumption	32 A
Operating temperature range	10 to 40 °C
Mass	4.4 kg without drive modules for single axes
Dimensions	482.6 mm × 132.55 mm × 278.55 mm
	C-886.31
Function	Controller for parallel-kinematic positioners
Axes	6 Optional: 2 additional single axes
Motion and control	
Drive type	Q-Motion® piezo inertia drive Optional single-axis drive type
Motor connector	Sub-D 15 (f)
Controller type	PID controller
Servo cycle time of slave modules	50 µs

	C-886.31
Cycle time of the C-886	10 ms
Encoder input	Sin/cos (differential), BiSS interface
Stall detection	Servo off, triggered by position error
Limit switches	2 × TTL per drive (polarity programmable)
Reference Point Switch	1 × TTL per drive
Characteristics of single axes	Depending on the drive type
Electrical properties	
Output voltage	0 to 48 V
Max. output power	30 W per drive
Characteristics of single axes	Depending on the drive type
Interfaces and operation	
Communication interfaces	TCP/IP: RJ45/Ethernet; USB: Mini-USB type B
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	API for C / C++ / C# / VB.NET / MATLAB / python, drivers for LabVIEW
Supported functions	User-defined coordinate system. Startup macro. Data recorder for recording operating data. Wave generator for periodic motion.
Miscellaneous	
Operating voltage	External power supply 24 V / 10 A included in the scope of delivery
Max. current consumption	32 A
Operating temperature range	10 to 40 °C
Mass	4.4 kg without drive modules for single axes
Dimensions	482.6 mm × 132.55 mm × 278.55 mm

13.1.2 Maximum Ratings

The C-886 is designed for the following operating data:

Input on:	Maximum operating voltage		Operating frequency		Maximum current consumption	
Feedthrough terminals GND and +24V	24 V		---		32 A	

13.1.3 Ambient Conditions and Classifications

The following ambient conditions and classifications for the C-886 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 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

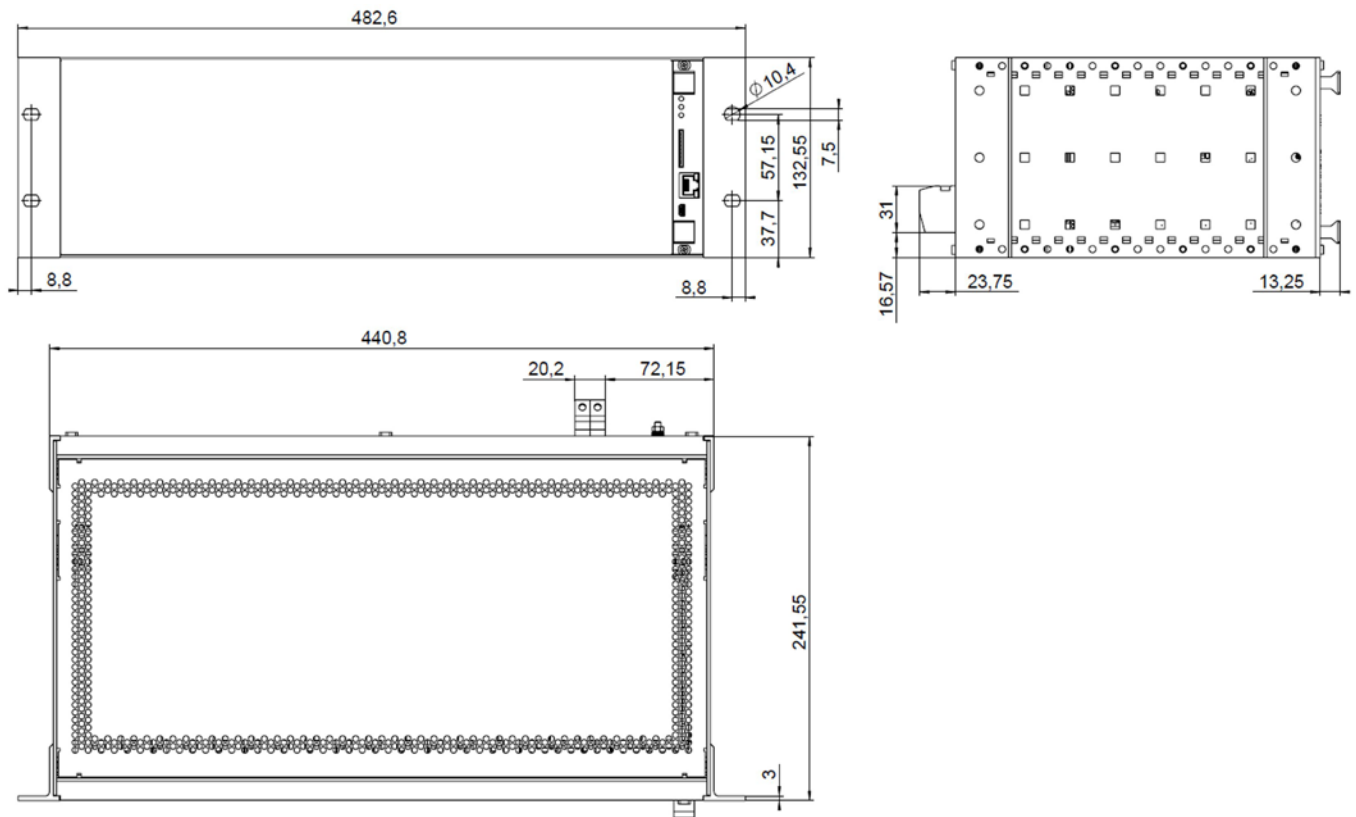
13.2 System Requirements

The following system requirements must be met to operate the system:

- Suitable parallel-kinematic positioner from PI
- C-886 with power supply
- PC with at least 30 MB of free memory and one of the following operating systems:
 - Windows: Vista Service Pack 1, Windows 7, 8, and 10 (32 bit, 64 bit)
 - Linux
- Communication interface to the PC:
 - USB interface on the PC
 - or
 - Ethernet connection in the PC or a free access point in the network to which the PC is connected via TCP/IP
- USB or network cable to connect the C-886 with the PC or with the network
- Product CD with PC software

13.3 Dimensions

C-886, here without drive modules. Dimensions in mm. Note that the decimal places are separated by a comma in the drawings.



13.4 Pin Assignment

13.4.1 Axis 1, Axis 2 (C-886.1 only)

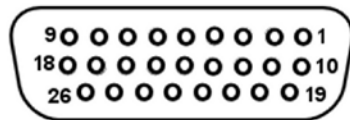
Sub-D socket, 15-pin, female



Pin	Function
1	Output: Programmable motor brake (0 or + 5 V)
2	Output: Motor + (differential; power PWM); for stages without PWM amplifier
3	Output: PWM magnitude (TTL); for stages with PWM amplifier
4	Output: +5 V
5	Input: Positive limit switch
6	GND limit switch
7	Input: Encoder: A (-)
8	Input: Encoder: B (-)
9	Output: Motor – (differential; power PWM); for stages without PWM amplifier
10	Power GND
11	Output: PWM sign (TTL); for stages with PWM amplifier
12	Input: Negative limit switch
13	Input: Reference point switch
14	Input: encoder: A (+) / ENCA
15	Input: encoder: B (+) / ENCB

13.4.2 Motor (C-886.2 only)

HD Sub-D 26 (f)

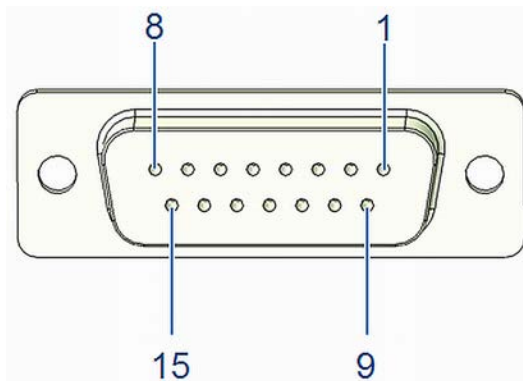


Pin	Signal	Direction	Function
1	OUT0a	Output	Phase I, pos. (48 V, max. 2 A, 20 kHz PWM)
2	OUT0b	Output	Phase I, pos. (48 V, max. 2 A, 20 kHz PWM)
3	OUT1a	Output	Phase I, neg. (48 V, max. 2 A, 20 kHz PWM)
4	OUT1b	Output	Phase I, neg. (48 V, max. 2 A, 20 kHz PWM)
5	OUT2a	Output	Phase II, pos. (48 V, max. 2 A, 20 kHz PWM)
6	OUT2b	Output	Phase II, pos. (48 V, max. 2 A, 20 kHz PWM)
7	OUT3a	Output	Phase II, neg. (48 V, max. 2 A, 20 kHz PWM)
8	OUT3b	Output	Phase II, neg. (48 V, max. 2 A, 20 kHz PWM)
9	-	-	Reserved
10	REF	Input	Reference point switch (5 V TTL input, single-ended)
11	NLIM	Input	Negative limit switch (5 V TTL input)
12	PLIM	Input	Positive limit switch (5 V TTL input)
13	-	-	Reserved
14	-	-	Reserved
15	-	-	Reserved
16	-	-	Reserved
17	ID chip	Bidirectional	ID chip (intended for future use)
18	VCC_ENC	Output	Position sensor power supply (5 V, 200 mA)
19	ENCA+	Input	Encoder input A+ (RS-422)
20	ENCA-	Input	Encoder input A- (RS-422)
21	ENCB+	Input	Encoder input B+ (RS-422)
22	ENCB-	Input	Encoder input B- (RS-422)
23	INDEX+	Input	Reference point switch, differential
24	INDEX-	Input	Reference point switch, differential
25	GND		GND
26	VCC_ENC	Output	Position sensor power supply (5 V, 200 mA)

Do not connect anything to reserved pins.

13.4.3 Motor & Sensor (C-886.31 only)

Sub-D 15 socket



Pin	Signal	Function
1	REF-	Reference point switch, differential (-)
2	PIEZO-	Motor signal (-)
3	PIEZO+	Motor signal (+)
4	5 V	Supply voltage +5 V
5	PLIM	Positive limit switch
6	ID_CHIP	ID chip data
7	ENCA-	Encoder channel A, differential (-)
8	ENCB-	Encoder channel B, differential (-)
9	PIEZO-	Motor signal (-)
10	GND	GND
11	PIEZO+	Motor signal (+)
12	NLIM	Negative limit switch
13	REF+	Reference point switch, differential (+)
14	ENCA+	Encoder channel A, differential (+)
15	ENCB+	Encoder channel B, differential (+)

14 Old Equipment Disposal

In accordance with EU law, electrical and electronic equipment may not be disposed of in EU member states via the municipal residual waste.

Dispose of your old equipment according to international, national, and local rules and regulations.

In order to fulfil its responsibility as the product manufacturer, Physik Instrumente (PI) GmbH & Co. KG undertakes environmentally correct disposal of all old PI equipment made available on the market after 13 August 2005 without charge.

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

Physik Instrumente (PI) GmbH & Co. KG
Auf der Roemerstr. 1
D-76228 Karlsruhe, Germany



15 EU Declaration of Conformity

For the C-886, an EU Declaration of Conformity has been issued in accordance with the following European directives:

EMC Directive

RoHS Directive

The applied standards certifying the conformity are listed below.

EMC: EN 61326-1

Safety: EN 61010-1

RoHS: EN 50581

