

MS243E C-884 DC Motor Controller User Manual

Version: 1.4.0

Date: 26.10.2018



This document describes the following products:

- **C-884.4DC**
Controller for DC motors, 4 axes, USB, RS-232, Ethernet, SPI, I/O, joystick connector
- **C-884.6DC**
Controller for DC motors, 6 axes, USB, RS-232, Ethernet, SPI, I/O, joystick connector



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

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

Notes on brand names and third-party trademarks:

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

EtherCAT is a registered and licensed brand of Beckhoff Automation GmbH.

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

Python® is a registered trademark of Python Software Foundation.

BiSS is a registered trademark of iC-Haus GmbH.

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

Linux, MATLAB, MathWorks

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

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

http://www.physikinstrumente.com/download/EULA_PhysikInstrumenteGmbH_Co_KG.pdf and in the Third-Party Software Notes

http://www.physikinstrumente.com/download/TPSWNote_PhysikInstrumenteGmbH_Co_KG.pdf on our website.

© 2018 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: 26.10.2018

Document number: MS243E, ASt, Version 1.4.0

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 This manual is superseded by any new release.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	Figures	3
1.5	Other Applicable Documents	4
1.6	Downloading Manuals.....	4
2	Safety	7
2.1	Intended Use	7
2.2	General Safety Instructions	7
2.3	Organizational Measures.....	8
3	Product Description	9
3.1	Product View	9
3.1.1	Front Panel	9
3.1.2	Type Plate	12
3.2	Scope of Delivery.....	12
3.3	Accessories	13
3.4	Overview of PC Software.....	13
3.5	Positioner Databases.....	15
3.6	ID Chip Detection.....	15
3.7	Communication Interfaces	16
3.8	Functional Principles	18
3.8.1	Block Diagram.....	18
3.8.2	Commandable Items.....	18
3.8.3	Important Components of the Firmware	20
3.8.4	Operating Modes.....	21
3.8.5	Physical Units.....	22
3.8.6	Motion Triggering	23
3.8.7	Generation of Dynamics Profile.....	25
3.8.8	Servo Algorithm and Other Control Value Corrections	29
3.8.9	On-Target State	31
3.8.10	Reference Point Switch Detection.....	32
3.8.11	Limit Switch Detection.....	33
3.8.12	Travel Range and Soft Limits	34
3.8.13	Reference Point Definition	38
3.8.14	Deactivation of Axes	42

4	Unpacking	43
5	Installation	45
5.1	General Notes on Installation.....	45
5.2	Mounting the C-884	45
5.3	Grounding the C-884.....	46
5.4	Connecting the Power Supply to the C-884	47
5.5	Connecting the Positioner	48
5.6	Connecting a Human Interface Device	48
5.7	Connecting Digital Inputs and Outputs	49
5.7.1	Connecting the Digital Outputs	49
5.7.2	Connecting the Digital Inputs	50
5.8	Connecting Analog Signal Sources.....	50
5.9	Installing the PC Software	51
5.9.1	Performing Initial Installation	51
5.9.2	Installing Updates	52
5.9.3	Installing Custom Positioner Databases	54
5.10	Connecting the PC	54
5.10.1	Connecting the C-884 via the TCP/IP Interface	55
5.10.2	Connecting the C-884 via the RS-232 Interface.....	55
5.10.3	Connecting the C-884 via the USB interface	56
6	Startup	57
6.1	General Notes on Startup.....	57
6.2	Switching on the C-884.....	58
6.3	Establishing Communication	59
6.3.1	Establishing Communication via the RS-232 Interface.....	59
6.3.2	Establishing Communication via the USB Interface	60
6.3.3	Establishing Communication via the TCP/IP Interface	62
6.4	Starting Motion	68
6.5	Setting the Notch Filter	73
6.6	Optimizing the Servo Control Parameters	79
7	Operation	83
7.1	Protective Functions of the C-884.....	83
7.1.1	Behavior with Motion Errors	83
7.1.2	Re-establishing Readiness for Operation	84
7.2	Trajectories for Motion Paths	85
7.2.1	Operating Principle of the Trajectory Buffer	85
7.2.2	Commands and Parameters for Trajectories.....	85
7.2.3	Working with Trajectories	86
7.3	Data Recorder.....	89
7.3.1	Setting up the Data Recorder	89

7.3.2	Starting the Recording.....	91
7.3.3	Reading Recorded Data	91
7.4	Digital Output Signals	91
7.4.1	Commands for Digital Outputs	92
7.4.2	Setting Up "Position Distance" Trigger Mode	93
7.4.3	Setting Up "On Target" Trigger Mode	95
7.4.4	Setting Up "Motion Error" Trigger Mode	96
7.4.5	Setting Up "In Motion" Trigger Mode	96
7.4.6	Setting Up "Position + Offset" Trigger Mode	97
7.4.7	Setting Up "Single Position" Trigger Mode.....	98
7.4.8	Setting up the "HardwareTrigger" Trigger Mode	99
7.4.9	Setting Signal Polarity	100
7.5	Digital Input Signals	101
7.5.1	Commands and Parameters for Digital Inputs	101
7.5.2	Using Digital Input Signals in Macros	103
7.5.3	Using Digital Input Signals as Switch Signals	103
7.6	Analog Input Signals	105
7.6.1	Commands for Analog Inputs	105
7.6.2	Using Analog Input Signals in Macros.....	106
7.7	Control with a Human Interface Device	106
7.7.1	Functionality of HID Control	106
7.7.2	Commands and Parameters for Human Interface Devices	107
7.7.3	Testing a Human Interface Device.....	108
7.7.4	Calibrating the Axes of Human Interface Devices	110
7.7.5	Setting Up and Enabling HID Control.....	113
7.7.6	Saving the Configuration of HID Control Permanently.....	115
7.8	Controller Macros.....	117
7.8.1	Overview: Macro Functionality and Example Macros.....	117
7.8.2	Commands and Parameters for Macros.....	117
7.8.3	Working with Macros	119
7.8.4	Macro Example: Synchronization of Two Controllers	126
7.8.5	Macro Example: Stopping Motion by Pushbutton	127
7.8.6	Macro Example: HID Control with Storage of Positions	128
8	GCS Commands	133
8.1	Notation.....	133
8.2	GCS Syntax for Syntax Version 2.0	133
8.3	Variables	135
8.4	Command Overview	137
8.5	Command Descriptions for GCS 2.0	142
8.6	Error Codes.....	248
9	Adapting Settings	271
9.1	Settings of the C-884	271
9.2	Changing Parameter Values in the C-884.....	271

9.2.1	General Commands for Parameters	272
9.2.2	Commands for Fast Access to Individual Parameters	273
9.2.3	Saving Parameter Values in a Text File	273
9.2.4	Changing Parameter Values: General Procedure	274
9.3	Creating or Modifying a Positioner Type.....	277
9.4	Parameter Overview.....	280
10	Maintenance	291
10.1	Cleaning the C-884	291
10.2	Updating Firmware.....	291
11	Troubleshooting	295
12	Customer Service	301
13	Technical Data	303
13.1	Specifications.....	303
13.1.1	Data Table.....	303
13.1.2	Maximum Ratings	304
13.1.3	Ambient Conditions and Classifications	305
13.2	System Requirements.....	305
13.3	Dimensions	307
13.4	Pin Assignment	308
13.4.1	DC motor	308
13.4.2	I/O	309
13.4.3	RS-232.....	311
13.4.4	Power supply connection 24 V DC.....	311
14	Old Equipment Disposal	313
15	EU Declaration of Conformity	315

1 About this Document

In this Chapter

Objective and Target Audience of this User Manual 1

Symbols and Typographic Conventions 1

Definition of Terms 2

Figures..... 3

Other Applicable Documents..... 4

Downloading Manuals 4

1.1 Objective and Target Audience of this User Manual

This manual contains information necessary for the intended use of the C-884.

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

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

1.2 Symbols and Typographic Conventions

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

CAUTION



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

➤ Actions to take to avoid the situation.

NOTICE




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

➤ Actions to take to avoid the situation.

INFORMATION

Information for easier handling, tricks, tips, etc.

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

1.3 Definition of Terms

Term	Explanation
Axis	Also referred to as "logical axis". The logical axis represents the motion of the positioner in the firmware of the C-884. For positioners that allow motion in several directions (e.g., in X, Y, and Z), each direction of motion corresponds to a logical axis.
Positioner	Mechanical system connected to the C-884. For positioners with just one motion axis the designation "axis" is synonymous with "positioner". Positioners that allow motion in several axes are also designated as "multi-axis positioners". For these positioners, a difference must be made between the individual axes.
Control value	The control value is the input for the PWM converter of the C-884. The PWM converter converts the control value into the PWM signal for the axis of the positioner.
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,

Term	Explanation
	absolute target positions can be commanded and reached immediately. A reference point definition is not necessary.
Incremental position sensor	Sensor (encoder) for capturing changes of position or changes of angle. Signals from the incremental position sensor are used for axis position feedback. After the controller is switched on, a reference point definition must be performed before absolute target positions can be commanded and reached.
Dynamics profile	Comprises the target position, velocity, and acceleration of the axis calculated by the profile generator of the C-884 for any point in time of the motion. The calculated values are called "commanded values".
Trajectory	A trajectory is a motion along a path made up of points that were externally calculated and loaded to the C-884 (target positions) and that are travelled according to a specified chronological interval.
HID	Abbreviation for "Human Interface Device". "HID" refers to an input or output device that is connected to the controller and is intended for manual operation. Depending on the controller, the connection can be made via USB, analog or digital interfaces. Typical human interface devices are joysticks and gamepads.
HID control	Control of the motion parameters of the axes of the C-884 via the displacement of the axes of human interface devices.
Volatile memory	RAM module in which the parameters are saved when the controller is switched on (working memory). The parameter values in the volatile memory determine the current behavior of the system. In the PC software of PI, the parameter values in the volatile memory are also referred to as "Active Values".
Nonvolatile memory	Memory module (read-only memory, e. g., EEPROM or flash memory), from which the default values of the parameters are loaded to the volatile memory when the controller is started. In the PC software of PI, the parameter values in the nonvolatile memory are also referred to as "startup values".
Firmware	Software that is installed on the controller.
PC software	Software that is installed on the PC.
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.

1.4 Figures

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

1.5 Other Applicable Documents

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

Description	Document
Short instructions for the installation and startup of the C-884	MS242EK Short instructions for digital motor controllers
C-884 GCS driver library for use with NI LabVIEW software	MS215E Software Manual
PIPython	SM157E User Manual
PI MATLAB Driver GCS 2.0	SM155E Software Manual
PI GCS 2.0 DLL0	SM151E Software Manual
GCS array data format description	SM146E Software Manual
PIMikroMove	SM148E Software Manual
PIStages3Editor Software for the management of the positioner database	SM156E Software Manual
PIUpdateFinder: Search for and download updates	A000T0028 Technical Note
PIUpdateFinder: Update the PC without an Internet connection	A000T0032 Technical Note

1.6 Downloading Manuals

INFORMATION

If a manual is missing or problems occur with downloading:

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

INFORMATION

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

You need the product CD to get the access data.

For products with CD: Get access data

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. Get the access data for downloading protected content in the "User login for software download" section of the Release News. Possible methods for getting the access data:
 - Link to a page for registering and requesting the access data
 - User name and password is specified
5. If the access data needs to be requested via a registration page:
 - a) Follow the link in the Release News.
 - b) Enter the required information in the browser window.
 - c) Click **Show login data** in the browser window.
 - d) Note the user name and password shown in the browser window.

Downloading manuals

If you have requested access data for protected contents via a registration page (see above):

- Click the links in the browser window to change to the content for your product and log in using the access data that you received.

General procedure:

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 bottom of the list.
 - c) Click the corresponding product in the list.
7. Click the **Downloads** tab.

The manuals are shown 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.....	7
General Safety Instructions.....	7
Organizational Measures.....	8

2.1 Intended Use

The C-884 is a laboratory device as defined by DIN EN 61010. 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-884 is intended for operating PI positioners equipped with DC motors or voice coil drives.

The C-884 is intended for closed-loop operation with incremental or absolute-measuring position sensors. In addition, it can read and process the reference point and limit switch signals from the positioner connected.

The C-884 may only be used in compliance with the technical specifications and instructions in this user manual. The user is responsible for process validation.

The C-884 may not be used for purposes other than those stated in this user manual.

2.2 General Safety Instructions

The C-884 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-884.

- Only use the C-884 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-884.

- Install the C-884 near the power source so that the power plug can be quickly and easily disconnected from the mains.

- Use the supplied components (power supply, adapter and power cord (p. 13)) to connect the C-884 to the power source.
- If one of the supplied components for connecting to the power source has to be replaced, use a sufficiently dimensioned component.

2.3 Organizational Measures

User manual

- Always keep this user manual available with the C-884.
The latest versions of the user manuals are available for download (p. 4) on our website.
- Add all information from the manufacturer to the user manual, for example supplements or technical notes.
- If you give the C-884 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-884 after you have read and understood this user manual.

Personnel qualification

The C-884 may only be installed, started up, operated, maintained, and cleaned by authorized and appropriately qualified personnel.

3 Product Description

In this Chapter

Product View	9
Scope of Delivery	12
Accessories.....	13
Overview of PC Software	13
Positioner Databases	15
ID Chip Detection	15
Communication Interfaces.....	16
Functional Principles.....	18

3.1 Product View








3.1.1 Front Panel














Figure 1: Front panel of the C-884.4DC






Figure 2: Front panel of the C-884.6DC

Element	Labeling	Type	Function
	24 V 9.0 A 	Sub-D 2W2C (p. 311)	Connector for the supply voltage
		M4 threaded pin	Ground connection (p. 46) If potential equalization is required, the C-884 can be connected to the grounding system.
per axis respectively: 	DC motor #	Sub-D 15 (f)	Connector for the positioning axis. Only for DC motors and voice coil drives! <ul style="list-style-type: none"> Output of PWM signals for the positioner Input of the signals of the position sensor Input of the signals from the limit switches and reference point switch Input for signals of the ID chip
	MAC	LED green/red/off	Macro: <ul style="list-style-type: none"> Off: Macro is not running. Green: Macro is running Red: Macro error The error code can be queried with the <code>MAC ERR?</code> command. The query resets the error code to zero and the LED is switched off.
	PWR	LED green/off	Power: <ul style="list-style-type: none"> Flashing: Booting of the Linux firmware component Lights up continuously: Booting of the Linux firmware component is finished. Off: C-884 is not connected to the supply voltage.

Element	Labeling	Type	Function
	STA	LED green/off	<p>State:</p> <ul style="list-style-type: none"> Flashing: Booting of the DSP and FPGA firmware Lights up continuously: Booting the DSP and FPGA firmware is complete and the controller is ready for normal operation. Off: C-884 is not connected to the supply voltage.
	ERR	LED red/off	<p>Error indicator:</p> <ul style="list-style-type: none"> Lights up continuously: Error (error code $\neq 0$) Off: No error (error code = 0) <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>
	RS-232	Sub-D 9 (m) (p. 311)	Serial connection to PC
	SPI	Display port	<p>Serial connection to a serial peripheral interface (SPI) master unit</p> <p>➤ If you want to use the SPI interface, contact the customer service department (p. 301).</p>
		RJ45	Ethernet interface for communication via TCP/IP
		USB type A	USB connection for human interface device (e.g., joystick or gamepad)
		Mini-B USB	Universal serial bus for connection to the PC
	I/O	HD Sub-D 26 (f) (p. 309)	<p>Digital inputs/outputs:</p> <ul style="list-style-type: none"> Outputs: Trigger external devices Inputs: Use in macros, as switch signals or for HID control <p>Analog inputs:</p> <ul style="list-style-type: none"> Use in macros or for scanning processes

3.1.2 Type Plate

Labeling	Function
	Data matrix code (example; contains the serial number)
C-884.4DC	Product name (example), the characters following the period refer to the model
PI	Manufacturer's logo
116056789	Serial number (example), individual for each C-884 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. 313)
CE	CE conformity mark
WWW.PI.WS	Manufacturer's address (website)

3.2 Scope of Delivery

Item	Component
C-884.4DC or C-884.6DC	Motion controller for DC motors, 4 axes or Motion controller for DC motors, 6 axes
000023194	Separate 24 V wide input range power supply (120 W/5 A) for use with line voltages from 100 to 240 V AC and voltage frequencies of 50 or 60 Hz, with barrel connector socket
3763	Power cord
K050B0004	Adapter from barrel connector to Sub-D 2W2C for the power adapter connection
C-815.34	RS-232 null modem cable, 3 m, 9/9-pin
000036360	USB cable (type A to Mini-B) for connection to the PC
C-815.553	Straight-through network cable for connecting the PC via a TCP/IP network
C-815.563	Crossover network cable for direct connection to the PC via TCP/IP
C-884.CD	Product CD for the C-884 with software and user manuals
MS242EK	Short instructions for digital motor controllers

3.3 Accessories

Article	Component
C-815.38	motor cable, 3 m, Sub-D, 15-pin. (m/f)

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

3.4 Overview of PC Software

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

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

PC software	Operating system	Short description	Recommended use
Dynamic program library for GCS	Windows, Linux	Allows software programming for the C-884 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 NI LabVIEW drivers if communication is to be established via USB (with Linux only via virtual COM port) or a daisy chain network.
PIPython	Windows, Linux, OS X	Collection of Python modules for convenient use of PI devices and GCS data. The modules can be used with 2.7+ and 3.4+ in the specified operating systems as well as via sockets on any other operating system of a PC.	For users who want to use Python for operating the controller. The use of Python as programming language offers a variety of possibilities and extends the scope of functions of the GCS commands considerably. Furthermore, debugging of macros is possible.
NI LabVIEW drivers	Windows, Linux	NI LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments). The C-884 NI LabVIEW software of PI is a collection of virtual instrument drivers (VI drivers) for the C-884 controller. These drivers support the GCS.	For users who want to use NI LabVIEW to program their application.

PC software	Operating system	Short description	Recommended use
NI LabVIEW Merge Tool	Windows	The NI LabVIEW Merge Tool allows you to combine product-specific NI LabVIEW drivers from PI with each other.	For users who want to operate several products from PI at the same time while using NI LabVIEW.
PIMikroMove	Windows	<p>Graphical user interface for Windows, which can be used for controllers from PI:</p> <ul style="list-style-type: none"> Start the system without programming effort Graphic representation of the motion Macro functionality for storing command sequences on the PC (host macros) Support of human interface devices Complete environment for command entry <p>PIMikroMove uses the dynamic program library to supply commands to the controller.</p>	<p>For users who want to perform simple automation tasks or test their equipment before or instead of programming an application.</p> <p>No command knowledge is necessary to operate PIMikroMove.</p>
PITerminal	Windows, Linux	Simple user interface that can be used for nearly all PI controllers.	For users who want to send GCS commands directly to the controller.
PIStages3Editor	Windows	Program for opening and editing positioner databases in .db format.	For users who want to deal intensively with the contents of positioner databases.
PIUpdateFinder	Windows	Checks the PI software installed on the PC. If newer versions of the PC software are available on the PI server, they are offered for download.	For users who want to update the PC software.
C-887/C-884 controller update wizard	Windows	Program for user support when updating the firmware of the C-884.	For users who want to update the firmware.
USB driver	Windows	Driver for the USB interface	For users who want to connect the controller to the PC via the USB interface.

3.5 Positioner Databases

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

Database file name	Description
PISTAGES3.DB	Delivery includes parameter sets for all standard positioners from PI and PI miCos, and is saved to the PC automatically during installation of the PC software New parameter sets can be created, edited, and saved.
<Produkt>.db e.g.: M-xxxxxxx.db	Includes the parameter set for a custom positioner. In order for the parameter set to be selected in the PC software, it must be added to the PISTAGES3.DB first, see "Installing Custom Positioner Databases" (p. 54).

The positioner database only contains some of the information that is required to operate a positioner with the C-884. Further information is loaded as parameter values to the volatile memory of the C-884 from the ID chip (p. 15) of the positioner when the C-884 is switched on or rebooted.

Parameters that are loaded from the positioner database or the ID chip, are described in the parameter overview (p. 280).

For more information on the positioner database, see the manuals for the PISTages3Editor and the PI GCS program library.

INFORMATION

If the pistages2.dat and pimicosstages2.dat positioner databases are on your PC:
Positioner databases in .dat format are only installed for compatibility reasons and **not** used for the C-884 described in this manual.

3.6 ID Chip Detection

Positioners offered by PI with DC motor have an ID chip in the connector where the following data is stored as parameters:

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

The data of the connected positioner is loaded from the ID-Chip into the volatile memory of the C-884 when the C-884 is switched on (p. 58) or rebooted.

The parameter values in the volatile memory of the C-884 can be queried and written to the nonvolatile memory, see "Adapting Settings" (p. 271).

INFORMATION

The ID chip only contains some of the information that is required to operate the positioner with the C-884. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 15) into the volatile memory of the C-884.

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

3.7 Communication Interfaces

The C-884 can be controlled with ASCII commands from a PC via the following communication interfaces:

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

INFORMATION

The communication interfaces 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.

- Configure the triggering of axis motion via the communication interfaces with the **Inhibit Motion Commands** parameter (0x130), for further information, see "Motion Triggering" (p. 23).

Default communication settings

Interface parameters of the C-884 for RS-232 communication:

Interface parameter	Factory setting	Note
Baud rate (RSBAUD)	115200	Indicates the baud rate of the C-884 for RS-232 communication. Further possible values are 9600, 19200, 38400, 57600. To successfully establish communication, the baud rates of the C-884 and PC must match.

Interface parameters of the C-884 for TCP/IP communication:

Interface parameter	Factory setting	Note
Default IP address (IPADR)	192.168.0.75:50000	Allows the definition of a static (i. e. fixed) address. This static address is not used when the C-884 is configured for assignment of an IP address by a DHCP server (default setting of the startup behavior for configuration of the IP address).
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.
Max. connections (IPMAXCONN)	1	Number of permissible simultaneous TCP/IP connections to the C-884
Startup 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-884 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.

3.8 Functional Principles

3.8.1 Block Diagram

The C-884.4DC controls the motion up to four logical axes. The C-884.6DC controls the motion up to six logical axes. The following block diagram shows how the C-884 generates the output signal for a connected axis:

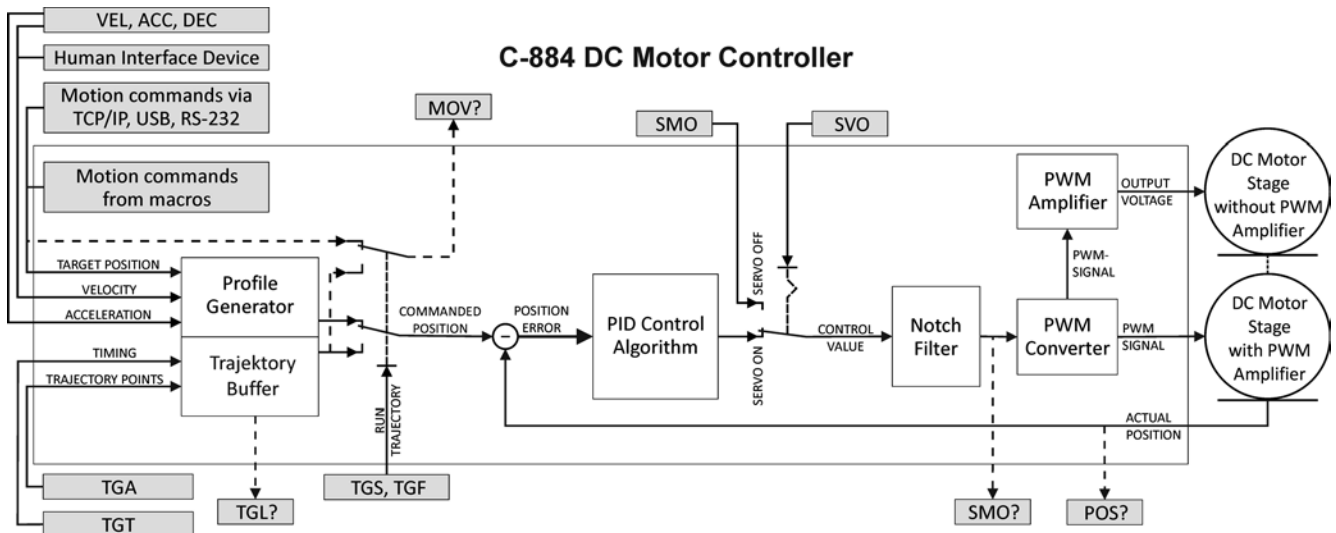


Figure 3: C-884: Generation of the output signal

The C-884 supports both positioners with PWM amplifier and positioners without PWM amplifier (p. 48).

The output voltage level for positioners without PWM amplifier depends on the supply voltage that is provided by the power adapter connected to the C-884.

3.8.2 Commandable Items

The following table contains the elements that can be commanded with GCS commands (p. 142).

Element	Number	Identifier	Description
Logical axes	4 or 6	1 to 4 or 1 to 6 (modifiable)	<p>A logical axis represents the motion of the positioner in the firmware of the C-884. It corresponds to the axis of a linear coordinate system.</p> <p>Motion for logical axes is commanded in the firmware of the C-884 (i.e., for the directions of motion of a positioner). The motion commands MOV, MVE and MVR, for example, are available in closed-loop operation. The motion command for open-loop operation is SMO.</p>

Element	Number	Identifier	Description
			<p>The axis identifier can be queried with the <code>SAI?</code> command and modified with the <code>SAI</code> command. It can comprise up to 8 characters; valid characters are 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ_</p> <p>The new axis identifier is saved automatically in the nonvolatile memory and is therefore still available even after a reboot or after the next switch-on.</p> <p>If the Stage Name parameter (0x3C) has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries). The identifier of a deactivated axis can only be queried with <code>SAI? ALL</code>.</p>
Trajectories	4 or 6	1 to 4 or 1 to 6	<p>The number of trajectories corresponds to the number of logical axes. Each trajectory is permanently allocated to a logical axis.</p> <p>Trajectories are commanded with TG* commands.</p> <p>For further information, see "Trajectories for Motion Paths" (p. 85).</p>
Analog inputs	4	1 to 4	<p>1 to 4 identify analog input lines 1 to 4 of the I/O (p. 309) socket.</p> <p>Their number is displayed with the <code>TAC?</code> command, and their values can be queried with the <code>TAV?</code> command.</p> <p>The values of all analog inputs can be recorded via record option 81 of the <code>DRC</code> command.</p> <p>Further information see "Analog Input Signals" (p. 105).</p>
Digital outputs	4	1 to 4	<p>1 to 4 identify digital output lines 1 to 4 of the I/O socket (p. 309).</p> <p>For further information, see "Digital Output Signals" (p. 91).</p>
Digital inputs	4	1 to 4	<p>1 to 4 identify digital input lines 1 to 4 of the I/O socket (p. 309).</p> <p>For further information, see "Digital Input Signals" (p. 101).</p>
Human interface devices	5	1 to 5	Up to five human interface devices can be connected to the USB socket (type A) of the C-884 via a USB hub.
Axes, buttons and LEDs of human interface devices	x	1 to x	<p>The number of commandable axes, buttons, and LEDs per human interface device depends on the human interface device connected. After a human interface device has been connected, information on the commandable axes, buttons and LEDs can be queried with the <code>HIS?</code> command.</p> <p>For further information, see "Control with a Human Interface Device" (p. 106).</p>

Element	Number	Identifier	Description
Data recorder tables	8	1 to 8	The C-884 has 8 data recorder tables (query with <code>TNR?</code>) with 8192 data points per table. For further information, see "Data Recorder" (p. 89).
Overall system	1	1	C-884 as an overall system

3.8.3 Important Components of the Firmware

The firmware of the C-884 provides the following functional units:

Component	Description
Parameters	<p>Parameters reflect the properties of the positioner connected (e.g., travel range) and specify the behavior of the C-884 (e.g., settings for the servo algorithm).</p> <p>The parameters can be divided into the following categories:</p> <ul style="list-style-type: none"> Protected parameters whose default settings cannot be changed Parameters that must be set by the user to adapt to the application <p>For further information, see "Adapting Settings" (p. 271).</p> <p>In the case of positioners with ID chip, the values of some parameters are stored on the ID chip. They are loaded to the volatile memory when switching on or rebooting the C-884.</p>
Command levels	<p>The command levels determine the write permission for the parameters. The current command level can be changed with the <code>CCL</code> command. This may require entering a password.</p>
ASCII commands (GCS)	<p>Communication with the C-884 can be managed using the commands of the PI General Command Set (GCS; version 2.0). The GCS is independent of the hardware (controller, positioners connected).</p> <p>Examples of the use of GCS:</p> <ul style="list-style-type: none"> Configuring the C-884 Setting the operating mode Starting motion of the positioner Getting system and position values <p>You can find a list of the available commands in the "Command Overview" section (p. 137).</p>
Profile generator	<p>During point-to-point motion in closed-loop operation, the profile generator performs calculations to specify the target position, velocity, and acceleration of an axis for each point in time during a motion. The result is the dynamics profile.</p> <p>For further information, see "Generation of Dynamics Profile" (p. 25).</p>

Component	Description
Trajectory buffer	For motion along freely definable paths, externally calculated trajectory points (target points) are loaded to the trajectory buffer of the C-884. The trajectory points are travelled according to a specified chronological interval in closed-loop operation. For further information, see "Trajectories for Motion Paths" (p. 85).
Servo algorithm	Closed-loop operation: The position error that results from the difference between the commanded target position and the actual position (sensor feedback) runs through a PID servo algorithm. For further information, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
Data recorder	The C-884 contains a real-time data recorder (p. 89). The data recorder can record various signals (e. g., position, control value) from different data sources (e. g., logical axes).
Macros	The C-884 can save macros (p. 117). Command sequences can be defined and stored permanently in the nonvolatile memory of the device via the macro function. A startup macro can be defined that runs each time the C-884 is switched on or rebooted. The startup macro simplifies stand-alone operation (operation without a connection to the PC). Further information can be found in the "Controller Macros" section (p. 117).

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

3.8.4 Operating Modes

The C-884 supports the following operating modes:

Operating Mode	Description
Closed-Loop Operation Servo mode on	<p>The commanded position for the axes comes from one of the two following sources:</p> <ul style="list-style-type: none"> ▪ Dynamics profile (p. 25): A profile generator calculates the dynamics profile from the values specified for target position, velocity, acceleration, and deceleration. ▪ Trajectory buffer (p. 85): The motion follows a path made up of points that were externally calculated and loaded to the C-884 (target positions) and that are travelled according to a specified chronological interval. <p>The position error that results from the difference between the commanded target position and the actual position (sensor feedback) runs through a PID servo algorithm (proportional integral derivative). Additional corrections can be made as well.</p> <p>The result is the control value for the PWM converter integrated in the C-884.</p>

Operating Mode	Description
	For further information, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
Open-loop operation Servo mode off	In open-loop operation, the C-884 does not calculate a dynamics profile, and no trajectory can be executed. The C-884 does not evaluate the signals of the position sensor. As a result, the positioner can move unbraked to the end of the travel range and, despite the limit switch function, strike the hard stop.

INFORMATION

The C-884 is intended for closed-loop operation with position sensors (servo mode On). After switch-on, open-loop operation is active by default (servo mode Off).

- Get the current operating mode with the `SVO?`, `#4` or `SRG?` commands.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a startup macro that starts the C-884 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 125).
- Avoid motion in open-loop operation.

3.8.5 Physical Units

The C-884 supports various units of length for positions. The adaptation is made via a factor with which the counts of the incremental encoder are converted into the physical unit of length required. The conversion factor is set with the following parameters:

Parameters	Description and Possible Values
Numerator Of The Counts-Per-Physical-Unit Factor 0xE	Numerator and denominator of the factor for counts per physical length unit 1 to 1.000.000.000 for each parameter.
Denominator Of The Counts-Per-Physical-Unit Factor 0xF	The factor for the counts per physical unit of length specifies the unit of length for position queries and motion commands in closed-loop operation. The values of every parameter, whose unit is either the physical unit of length itself or a unit of measurement based on it, are automatically adapted to the set factor. The factor for the counts per physical unit of length has no impact on the stability of the servo loop but is used for the input and output scaling of position values.

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

Parameters	Description and Possible Values
Axis Unit 0x07000601	Unit symbol Maximum of 20 characters. For example, the unit symbol is "MM", if the factor for the counts per physical unit of length is set with the 0xE and 0xF parameters so that the encoder counts are converted into millimeters. The unit symbol for rotation stages is normally "deg". The value of the parameter 0x07000601 is not evaluated by the C-884 but is used by the PC software for display purposes. Examples: 1 encoder count = 100 nm Counts per physical length unit: 10000:1 → Unit symbol: mm 1 encoder count = 0.254 mm Counts per physical length unit: 100:1 → Unit symbol: inch

3.8.6 Motion Triggering

Motion in closed-loop operation

In closed-loop operation, motion is either triggered via commands (p. 137) or via a human interface device, e.g., a joystick.

Motion commands and the execution of trajectories are not allowed when HID control (p. 106) is enabled for the axis.

Trigger of the motion	Commands	Description
Commands for point-to-point motion sent from the command line or via the PC software	MOV, MVR, MVE	Motion to absolute or relative target position
	GOH	Motion to zero position
	STE	Initiates a step of a specified distance and records the step response
	FRF	Starts reference moves
	FED	Starts moves to signal edges
Controller macros with commands for point-to-point motion	MAC	Calls a macro function. Permits recording, deleting, and running macros on the controller. Any commands can be sent from the command line when a macro is running on the controller. The macro content and motion commands received from the command line can overwrite each other.
	Additional macro commands and information see "Controller Macros" (p. 117).	

Trigger of the motion	Commands	Description
Trajectory execution	TGS	Starts the execution of an externally calculated trajectory
	For further commands and information, see "Trajectories for Motion Paths" (p. 85).	
HID control	HIN	Enables or disables control of the axes of the C-884 by the axes of human interface devices.
	HIA	Configures HID control for the axes of the C-884. The following motion variables of the C-884 axis can be controlled via the axes of human interface devices: <ul style="list-style-type: none"> ▪ Velocity of the axis ▪ Maximum velocity of the axis
	For further commands, see "Commands and Parameters for human interface devices" (p. 107).	

INFORMATION

For positioners with incremental position sensor, absolute target positions can be commanded only if the reference point for the axis has been defined beforehand; see "Reference Point Definition" (p. 38).

Motion in open-loop operation

Motion is triggered with the **SMO** command which specifies the control value for the PWM converter directly in the C-884.

The execution of trajectories and HID control are not possible in open-loop operation.

Parameters

The triggering of axis motion can be configured with the following parameter:

Parameters	Description and Possible Values
Inhibit Motion Commands 0x130	Configures triggering of motion via motion commands: 0 = All motion commands are executed (default setting). 1 = Motion commands from controller macros are not permitted. 2 = Motion commands sent via RS-232 are not permitted. 4 = Motion commands sent via USB are not permitted. 8 = Motion commands sent via TCP/IP are not permitted. 16 = Motion commands sent via SPI are not permitted. Effective in closed-loop or open-loop operation. The value of the parameter has no influence on HID control of the axes of the C-884. The use of several sources of motion commands simultaneously can be

Parameters	Description and Possible Values
	<p>excluded by addition of the option numbers.</p> <p>Examples:</p> <p>Prevent execution of all motion commands Parameter 0x130 has the following value: 31 (= 1 + 2 + 4 + 8 + 16)</p> <p>Only permit motion commands from controller macros Parameter 0x130 has the following value: 30 (= 2 + 4 + 8 + 16)</p>

Stopping motion

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

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

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

During the execution of trajectories, HLT also triggers an abrupt stopping of the motion.

3.8.7 Generation of Dynamics Profile

The profile generator of the C-884 is also used for point-to-point motion in closed-loop operation. The profile generator performs calculations to specify the target position, velocity, and acceleration of the axis for any point in time during motion (dynamics profile). The values calculated are called commanded values.

INFORMATION

During the execution of trajectories (p. 85), the profile generator does **not** calculate a dynamics profile.

The dynamics profile generated by the profile generator of the C-884 depends on the motion parameters which are given by commands, parameters (p. 280), and/or by human interface device.

Motion parameter	Commands	Parameters	Remarks
Acceleration (A)	ACC (p. 146) ACC? (p. 146)	0xB Closed-Loop Acceleration (phys. unit/s ²) Change with the ACC command or with SPA (p. 223) / SEP (p. 219); query with ACC?	Is limited by parameter 0x4A (maximum acceleration in closed-loop operation). The maximum acceleration during HID control is given by parameter 0x75.
Deceleration (D)	DEC (p. 158) DEC? (p. 158)	0xC Closed-Loop Deceleration (phys. unit/s ²) Change with the DEC command or with SPA / SEP; query with DEC?	Is limited by parameter 0x4B (maximum deceleration in closed-loop operation). The maximum deceleration during HID control is given by parameter 0x76.
Velocity (V)	VEL (p. 244) VEL? (p. 245)	0x49 Closed-Loop Velocity (Phys. Unit/s) Change with the VEL command or with SPA / SEP; query with VEL?	Is limited by parameter 0xA (maximum velocity in closed-loop operation). In principle, the maximum velocity during HID control is given by parameter 0x74. For further information, see "Control with a human interface device" (p. 106) and the description of the HIA command (p. 176).
Target position at the end of the motion	MOV (p. 209) MVR (p. 212) MVE (p. 211) GOH (p. 172) STE (p. 227)	-	During HID control of the velocity, the soft limits are set as the respective target position. For further information, see "Control with a Human Interface Device" (p. 106). The C-884 sets the target position to the current position of the axis in the following cases: <ul style="list-style-type: none"> Disabling HID control for the axis Switching on the servo mode with the SVO (p. 228) command Stopping the motion with the #24 (p. 145), STP (p. 228), or HLT (p. 193) commands

The profile generator of the C-884 only supports trapezoidal velocity profiles: The axis accelerates linearly (based on the acceleration value specified) until it reaches the specified velocity. It continues to move with this velocity until it decelerates linearly (based on the deceleration value specified) and stops at the specified target position.

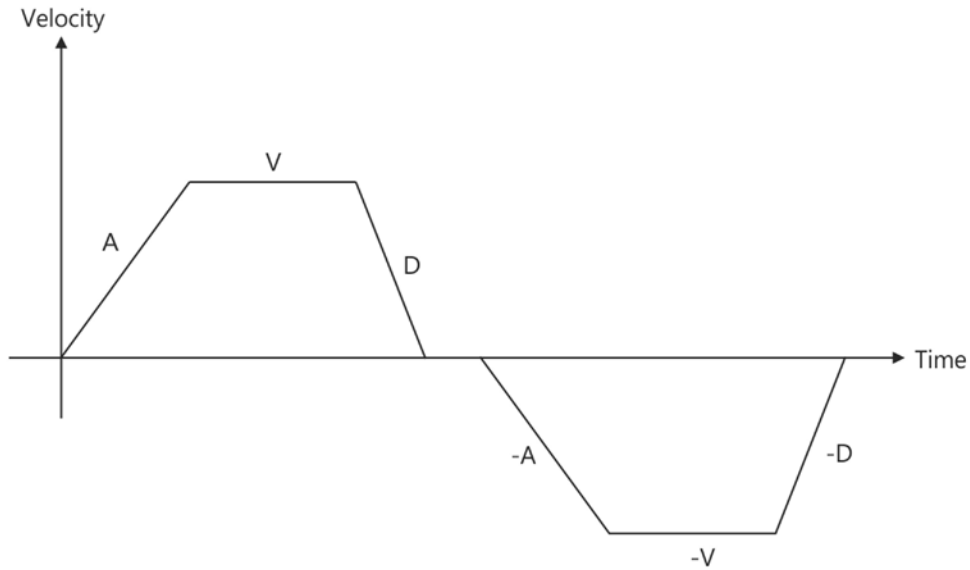


Figure 4: Basic trapezoidal velocity profile; A = acceleration, D = deceleration, V = velocity

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

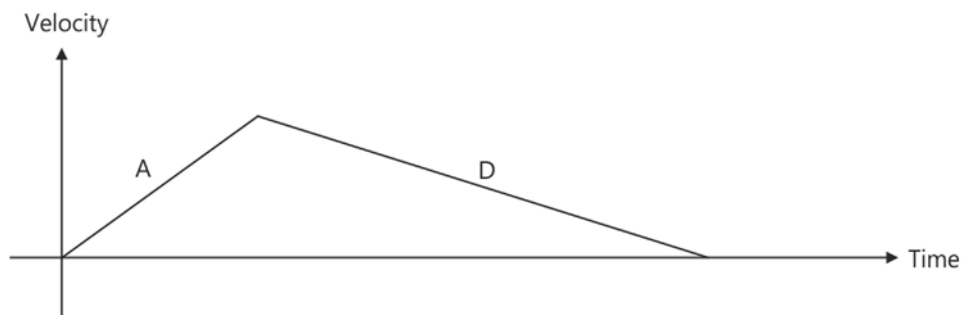


Figure 5: Basic trapezoidal velocity profile; A = acceleration, D = deceleration, no constant velocity

The edges for acceleration and deceleration can be symmetrical (acceleration = deceleration) or asymmetrical (acceleration \neq deceleration). The acceleration value is always used at the start of the motion. After that, the acceleration value is used during an increase in the absolute velocity and the deceleration value during a decrease in the absolute velocity. If no motion parameters are changed during the course of the motion, the acceleration value is used until the maximum velocity is reached and the deceleration value is used for the decrease in velocity down to zero.

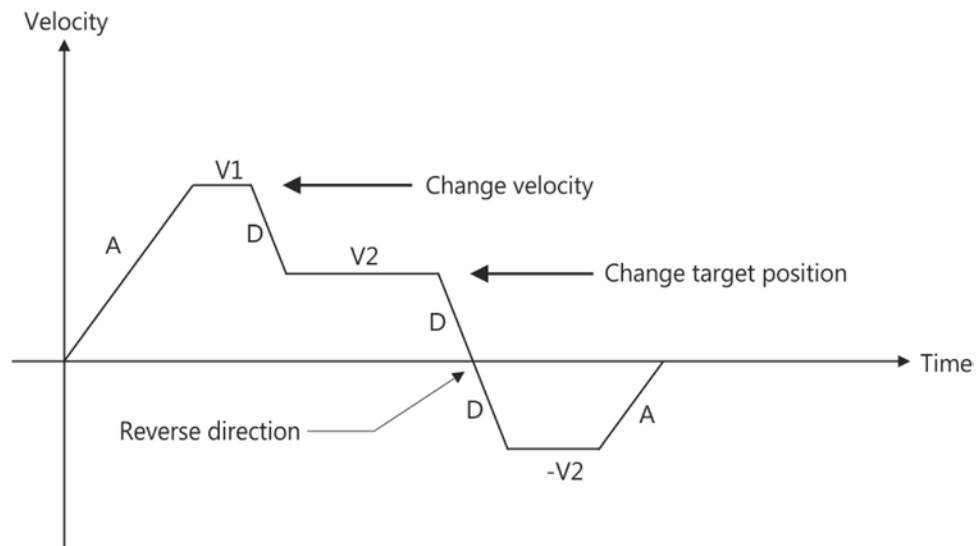


Figure 6: Complex trapezoidal profile with parameter changes; A = acceleration; D = deceleration; V1, V2, -V2 = velocities

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

3.8.8 Servo Algorithm and Other Control Value Corrections

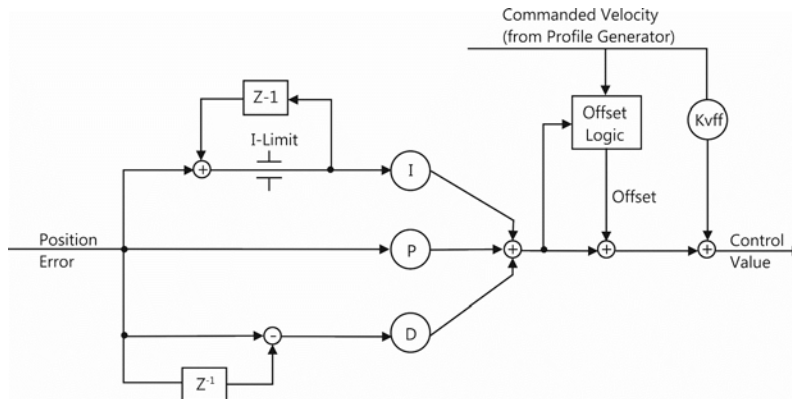


Figure 7: PID algorithm, offset correction and feed-forward control of the velocity (KVff); for one axis of the C-884; the notch filter is not shown here

In closed-loop operation, the control value per axis for the PWM converter integrated in the C-884 and therefore the settling behavior of the system is optimized by the following corrections:

- Servo algorithm: The position error that results from the difference between the commanded position (from the dynamics profile (p. 25) or trajectory (p. 85)) and the actual position (sensor feedback) runs through a PID servo algorithm (proportional integral derivative).
- Corrections of the control value: The dynamics profile or the trajectory can be subjected to an offset correction and a feed-forward control of the velocity.

Regardless of the operating mode, the control value can be subjected to an additional correction via the notch filter.

Servo algorithm

The servo algorithm uses the following servo control parameters. The optimum servo control parameter setting depends on your application and your requirements; see "Optimizing Servo Control Parameters" (p. 79).

Parameters	Description and Possible Values
P-Term 0x1	Proportional constant (dimensionless) 0 to 32767 Aim: Rapid correction of the position error
I-Term 0x2	Integration constant (dimensionless) 0 to 32767 Aim: Reduction of the static position error
D-Term 0x3	Differential constant (dimensionless) 0 to 32767 Aim: Damping of rapid control oscillation

Parameters	Description and Possible Values
I-Limit 0x4	Limit of the integration constant (dimensionless) 0 to 32767
D Term Delay (No. Of Servo Cycles) 0x71	D term delay The D term can be calculated as a floating average over several servo cycles. The parameter specifies how many values (i.e., servo cycles) are to be used for averaging.

The input of the servo algorithm can be configured for the C-884 with the following parameters:

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

Corrections of the control value

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

Parameters	Description and Possible Values
Motor Offset Positive 0x33	Offset for the positive direction of motion (dimensionless). 0 to 32766
Motor Offset Negative 0x34	Offset for the negative direction of motion (dimensionless). 0 to 32766
Motor Drive Offset 0x48	Velocity-dependent offset (dimensionless). Is used if the commanded velocity does not equal zero (i. e. if the end of the dynamics profile has not been reached yet). 0 to 32766
Kvff 0x5	Feed-forward control of the commanded velocity 0 to 32767 Aim: Minimization of the position error

Settings for the notch filter

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

Parameters	Description and Possible Values
Notch Filter Frequency 1 (Hz) 0x94	Frequency of the first notch filter 40 to 20000 Hz The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanical system. An adjustment can be particularly useful in the case of very high loads.
Notch Filter Edge 1 0x95	Rise of the edge of the first notch filter (dimensionless) 0.1 to 10 This parameter value should not be changed.

For setting the notch filter, see "Setting the Notch Filter" (p. 73).

3.8.9 On-Target State

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

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

The C-884 determines the on-target state on the basis of the following criteria:

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

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

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

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

In the *On Target* trigger mode (p. 95), the on-target state of the selected axis is output at the selected trigger output.

Parameter	Description and Possible Values
Settling Time (s) 0x3F	Delay time for setting the on-target state 0 to 1.000 s

Parameter	Description and Possible Values
Settling Window (encoder counts) 0x36	Settling window around the target position 0 to 2^{31} counts of the encoder Specifies the window limits. If the current position exits the settling window, the target position is no longer considered as reached. The parameter value corresponds to half the width of the window. It can be changed only if the servo mode is switched off.

3.8.10 Reference Point Switch Detection

The C-884 receives reference point switch signals of an axis respectively on pin 13 of the **DC Motor 1** to **DC Motor 4** (C-884.4DC) or **DC Motor 1** to **DC Motor 6** (C-884.6DC) sockets.

The following parameters can be used to configure how the C-884 detects the reference point switch:

Parameters	Description and Possible Values
Invert Reference? 0x31	Should the reference signal be inverted? 0 = Reference signal not inverted 1 = Reference signal inverted This parameter is used for inverting the reference signal whose source can be either the reference point switch or a digital input which is used instead of the reference switch .
Has Reference? 0x14	Does the positioner have a reference point switch? 0 = Reference point switch not installed 1 = Reference point switch (signal input at the axis connections) This parameter enables or disables reference moves to the reference point switch installed.
Reference Signal Type 0x70	Reference signal type 0 = Direction-sensing reference point switch. The signal level changes when passing the reference point switch. 1 = Pulse signal with a pulse width of several nanoseconds (parameter 0x47 must be set correctly). 2 = Index pulse. The reference point switch is approached via the negative limit of the travel range. 3 = Index pulse. Approaching the reference point switch is done via the positive limit of the travel range. 4 = No reference signal 5 = The reference signal is output at the negative limit switch. 6 = The reference signal is output at the positive limit switch.

The signal from the reference point switch of the positioner can be used for reference moves. In the case of a positioner with incremental position sensor, the controller knows the absolute axis position after a reference move to the reference point switch; see "Reference Point Definition" (p. 38).

3.8.11 Limit Switch Detection

The C-884 receives limit switch signals on the **DC Motor 1** to **DC Motor 4** (C-884.4DC) or **DC Motor 1** to **DC Motor 6** (C-884.6DC) sockets:

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

The following parameters can be used to configure how the C-884 detects the limit switches:

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

The signals from the limit switches (also end-of-travel sensors) of a linear stage are used to stop the motion prior to the hard stop at both ends of the travel range. Because the set deceleration is not taken into account here, there is a risk at high velocities that the positioner will hit the hard stop anyway. To prevent this, soft limits (p. 34) can be set via parameters of the C-884.

The limit switch signals can also be used for reference moves. In the case of a positioner with incremental position sensor, the controller knows the absolute axis position after a reference move to a limit switch; see "Reference Point Definition" (p. 38).

3.8.12 Travel Range and Soft Limits

The following parameters of the C-884 reflect the physical travel range of the axes and define soft limits:

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

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

INFORMATION

The C-884 supports two parameter pairs for establishing soft limits. They are intended for different applications:

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

Examples

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

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

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

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

INFORMATION

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

Example 1: Maximum travel range available

After reference moves (p. 38), the current position is to have the following values:

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

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

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

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

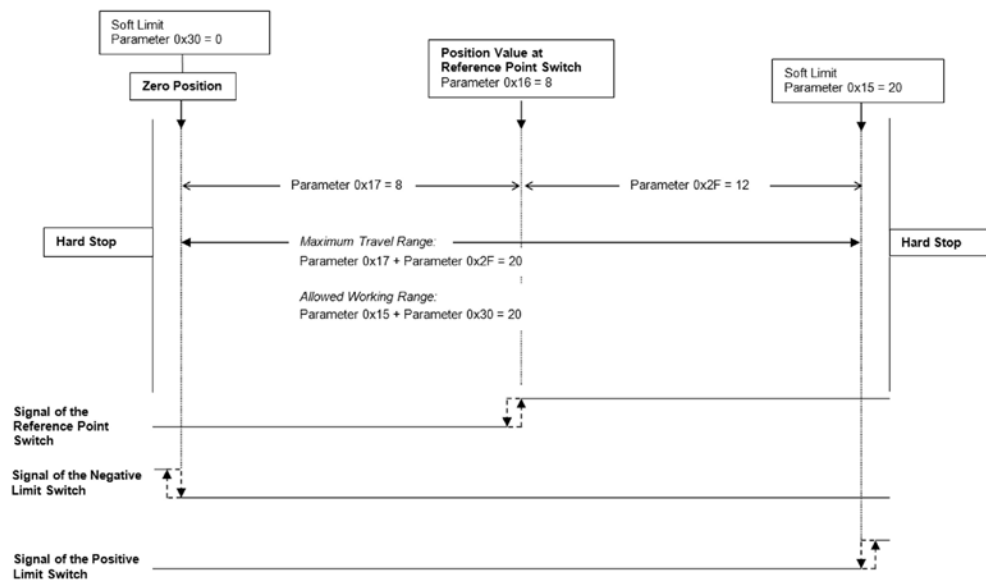


Figure 8: The travel range of the stage is not limited by soft limits.

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

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

Example 2: Travel range limited by soft limits

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

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

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

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

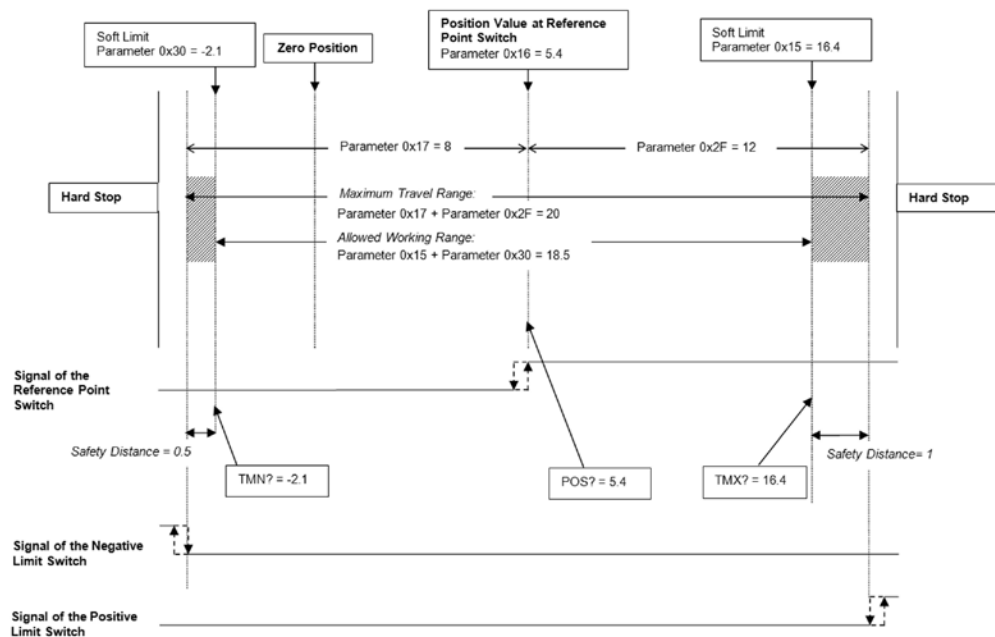


Figure 9: The travel range of the stage is limited by soft limits.

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

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

3.8.13 Reference Point Definition

INFORMATION

Whether a reference point definition is necessary for the axis depends on the signal type of the position sensor:

- Absolute-measuring position sensor: Reference point definition is **not** necessary.
- Incremental position sensor: Reference point definition is necessary.

The information on the signal type of the position sensor is loaded from the ID chip (p. 15) and given by the value of the **Sensor Signal Type** parameter (ID 0x3003320) (see "Parameter Overview" (p. 280)).

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

The reference point definition can be performed in different ways:

- **Reference move** (default): A reference move moves the axis to a defined point, e.g., to the reference point switch or to a limit switch. At this point, the current position is set to a defined value. The controller now knows the absolute axis position.
- **Setting the absolute position manually:** If this type of reference point definition was enabled with the **RON** command (p. 215), you can set the current position of the axis to an arbitrary value at an arbitrary point using the **POS** command (p. 213). Here the axis is not moved. The controller knows the absolute axis position afterwards.

INFORMATION

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

INFORMATION

To achieve maximum repeatability of the reference point definition, every reference move comprises the following steps:

1. First move to the switch selected. The maximum velocity is specified via parameter 0x49 (**Closed-Loop Velocity (Phys. Unit/s)**, equivalent to setting with the VEL command).
2. Stop upon reaching the switch edge. The higher the velocity on approach, the farther the axis overruns the edge of the switch (overshooting).
3. Move in the opposite direction to compensate for the overshoot.
4. Second move to the switch selected. The maximum velocity is specified via parameter 0x50 (**Velocity For Reference Moves (Phys. Unit/s)**, specific given velocity for reference moves).

only).

5. Stop upon reaching the switch edge.
6. Move in the opposite direction to compensate for the overshoot.
7. Setting the current position to a defined value, the reference point definition is finished.

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

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

- Parameter 0x49 or 0x50
- Parameter 0x63 (***Distance Between Limit And Hard Stop (Phys. Unit)***)
- Parameter 0xC (***Closed-Loop Deceleration (Phys. Unit/s²)***)

Commands

The following commands are available for reference point definition:

Com-mand	Syntax	Function
RON	RON {<AxisID> <ReferenceOn>}	Sets mode of reference point definition: <ul style="list-style-type: none"> ▪ <ReferenceOn> = 0: To define the reference point of the axis, an absolute position value can be assigned with POS or a reference move can be started with FRF. ▪ <ReferenceOn> = 1 (default): For reference point definition of the axis, a reference move must be started with FRF. Using POS is not allowed.
RON?	RON? [{<AxisID>}]	Gets mode of reference point definition.
FRF	FRF [{<AxisID>}]	Starts a reference move to the reference point switch. The approach depends on the value of the <i>Reference Signal Type</i> parameter (0x70): <ul style="list-style-type: none"> ▪ 0 or 1: The approach always takes place from the same side irrespective of the axis position when the command is sent. ▪ 2: The approach takes place via the negative limit switch. ▪ 3: The approach takes place via the positive limit switch. ▪ 4: No reference signal ▪ 5: The reference move is made to the negative limit switch; this is set as reference position.

		<ul style="list-style-type: none"> 6: The reference move is made to the positive limit switch; this is set as reference position.
FRF?	FRF? [{<AxisID>}]	<p>Queries whether the reference point for an axis has already been defined.</p> <p>1 = Reference point has been defined</p> <p>0 = Reference point has not been defined</p>
POS	POS {<AxisID> <Position>}	Sets the current position (does not trigger a motion) and therefore defines the reference point.

Parameters

Reference moves can be configured with the following parameters:

Parameters	Description and Possible Values
Closed-Loop Deceleration (Phys. Unit/s²) 0xC	Deceleration in closed-loop operation For details, see "Generation of the Dynamics Profile (p. 25)".
Reference Travel Direction 0x47	Default direction for the reference move 0 = automatic detection 1 = negative direction 2 = positive direction
Closed-Loop Velocity (Phys. Unit/s) 0x49	Velocity in closed-loop operation For details, see "Generation of the Dynamics Profile (p. 25)".
Velocity For Reference Moves (Phys. Unit/s) 0x50	Velocity for reference move Specifies the maximum velocity during a reference move for the second approach of the switch selected. For high repeatability during a reference point definition, this value should at a maximum be as great as the value of parameter 0x49. If the value of parameter 0x50 is set to 0, reference moves are not possible.
Distance Between Limit And Hard Stop (Phys. Unit) 0x63	Distance between the built-in limit switch and the hard stop Determines the maximum stopping distance during reference moves. The actual velocities during a reference move are calculated on the basis of this value, the deceleration set (0xC) and the velocities set (0x49 and 0x50).
Distance From Limit To Start Of Ref. Search (Phys. Unit) 0x78	Distance between limit switch and the starting position for the motion to the index pulse For details, see explanation below the table.
Distance For Reference Search (Phys. Unit) 0x79	Maximum distance for the motion to the index pulse For details, see explanation below the table.

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

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

Sequence of the reference move:

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

INFORMATION

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

INFORMATION

The limit switches can be used for reference moves only if the travel range is not limited by soft limits (p. 34).

INFORMATION

For reference moves, you can also use the digital inputs of the C-884 as the source of the reference signal, the negative limit switch signal or the positive limit switch signal. See "Using Digital Input Signals as Switch Signals" (p. 103) for more information.

INFORMATION

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

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

INFORMATION

If the current parameter settings of the C-884 are written to the nonvolatile memory in PIMikroMove or by entering the WPA command using the password 100 or 101, the axis will no longer be considered "referenced" (the response to `FRF?` is 0).

3.8.14 Deactivation of Axes

If an axis may not be moved under any circumstances, it can be deactivated. A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries). The identifier of a deactivated axis can only be queried with `SAI? ALL`.

The criterion for deactivating an axis of the C-884 is the value of the **Stage Name** parameter (ID 0x3C): The axis is deactivated when the **Stage Name** parameter has the NOSTAGE value (default setting).

INFORMATION

In PIMikroMove, axes can be deactivated in the **Select connected stages** step with the **No Stage ->** button. For details, see the PIMikroMove manual.

4 Unpacking

1. Unpack the C-884 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

General Notes on Installation	45
Mounting the C-884	45
Grounding the C-884	46
Connecting the Power Supply to the C-884	47
Connecting the Positioner	48
Connecting a Human Interface Device	48
Connecting Digital Inputs and Outputs	49
Connecting Analog Signal Sources	50
Installing the PC Software	51
Connecting the PC	54

5.1 General Notes on Installation

- Install the C-884 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Only use cables and connections that meet local safety regulations.

5.2 Mounting the C-884

The C-884 can be used as a benchtop device or mounted on a surface in any alignment at the base plate.

The C-884 can be installed into a control cabinet.

The housings of the C-884.4DC and .6DC models are identical.

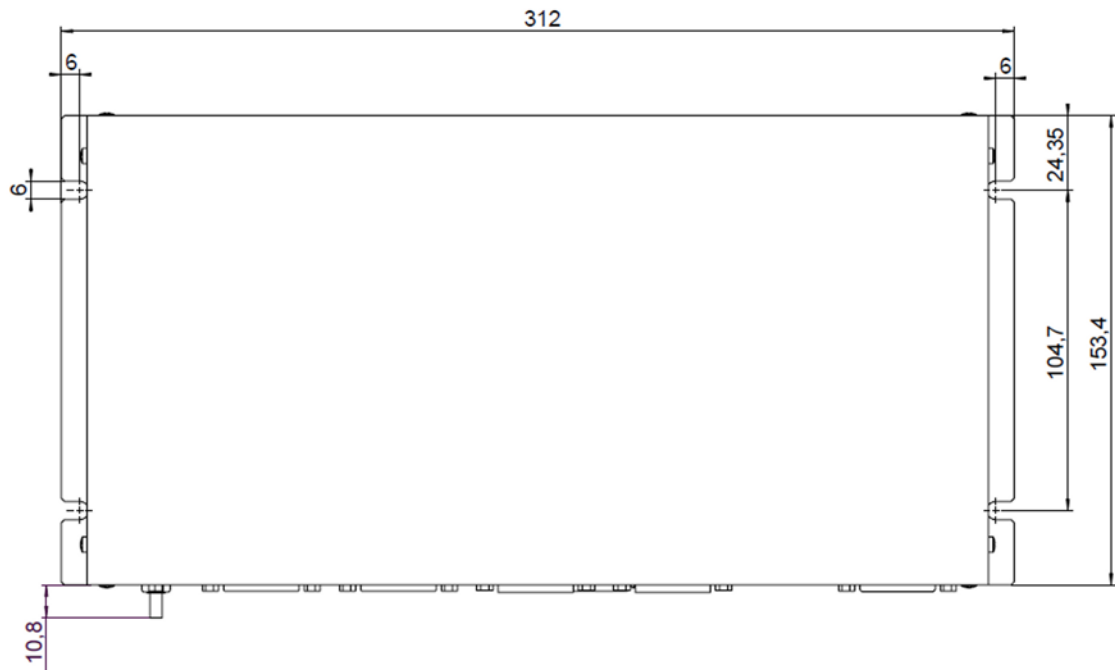


Figure 10: C-884.4DC and .6DC: Arrangement of the mounting holes

Tools and accessories

- Suitable screws
- Suitable screwdriver

Mounting the C-884

1. Make the necessary holes in the surface.

The arrangement of the recesses in the mounting rails of the C-884.4DC and .6DC models can be viewed in the figure.

2. Use two suitable screws respectively for each side to affix the C-884 to the mounting holes provided.

5.3 Grounding the C-884



The C-884 is not grounded via the power supply connection.

- Connect the threaded pin with the protective earth conductor symbol (see figure) on the housing of the C-884 to the protective earth conductor.

5.4 Connecting the Power Supply to the C-884

NOTICE



Motor damage due to excessively high operating voltage!

The output voltage at the **Motor +** and **Motor -** pins of **DC Motor 1** to **DC Motor 4** sockets in the case of the C-884.4DC or the **DC Motor 1** to **DC Motor 6** sockets in the case of the C-884.6DC can be as high as the voltage provided by the supply voltage. Positioners without PWM amplifier are connected to these pins and can be damaged by excessive output voltage.

- Connect a power adapter whose output voltage does not exceed the permissible operating voltage of the positioner.
- Adjust the maximum output voltage of the C-884 to the permissible operating voltage of the positioner with the **Maximum Motor Output** (0x9) parameter.

Requirements

- ✓ The power adapter is **not** connected to the power socket via the power cord.
- ✓ The C-884 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 input range power supply (for line voltages between 100 and 240 VAC at 50 or 60 Hz)
Alternative: Sufficiently dimensioned power adapter that provides 24 V direct current
- Included adapter for the power adapter connection; barrel connector to Sub-D 2W2C
Alternative: Sufficiently dimensioned adapter
- Included power cord
Alternative: Sufficiently dimensioned power cord

Connecting the C-884 to the power supply

1. Connect the Sub-D 2W2C of the adapter to the **24 V DC** connection of the C-884.
2. Connect the barrel connector of the adapter to the barrel connector socket of the power adapter.
3. Connect the power cord to the power adapter.

5.5 Connecting the Positioner

NOTICE

**Damage if a wrong motor is connected!**

Connecting a Positioner with stepper motor to a DC motor controller can cause irreparable damage.

- Only connect one Positioner with DC motor or PWM motor driver to the C-884.

INFORMATION

The C-884 supports both positioners with PWM amplifier and positioners without PWM amplifier. Separate lines for both positioner variants are available at **DC Motor 1 to DC Motor 4** of the C-884.4DC or **DC Motor 1 to DC Motor 6** of C-884.6DC sockets; see the pin assignment of the sockets. Makes sure that suitable lines are selected via the connector of the positioner. Proper connection is ensured for positioners from PI.

Requirements

- ✓ The C-884 is switched off, i.e., the power adapter is **not** connected to the power socket with the power cord.
- ✓ You have read and understood the user manual for the positioner.

Tools and accessories

- Positioner with DC motor or voice coil drive
- If the distance between C-884 and positioner is too long:
C-815.38 motor cable, 3 m, available as optional accessory (p. 13)

Connecting the positioner

1. Connect the positioner to the following sockets:
 - **DC Motor 1 to DC Motor 4** of the C-884.4DC
 - **DC Motor 1 to DC Motor 6** of the C-884.6DC
2. Use the integrated screws to secure the connections against accidental disconnection.

5.6 Connecting a Human Interface Device

INFORMATION

The C-884 assigns identifiers to connected human interface devices as follows:

- When a human interface device is directly connected to the C-884, it always receives the identifier 1.

- When human interface devices are connected to a USB hub that is connected to the C-884 when the C-884 is switched off, the C-884 assigns the identifiers of the human interface devices according to the numbering of the physical slots when it is switched on.
 - When human interface devices are connected to a USB hub connected to the C-884 when the C-884 is switched on, the C-884 assigns the identifiers in the order of connection. The identifiers of the human interface devices can therefore change after the C-884 is rebooted.
 - For a consistent assignment of the identifiers, always connect the human interface devices in the same chronological order and in the same slots on the hub when the C-884 is switched on.
-

Tools and accessories

- Suitable human interface device with USB connection (type A), e.g., a joystick or gamepad
- If several human interface devices are to be connected: suitable USB hub

Connecting a human interface device

- Connect a single human interface device directly to the USB connection type A (p. 9) of the C-884.

If you are using a USB hub to connect several human interface devices:

- Connect the human interface devices to the USB hub.
- Connect the USB hub to the USB connection type A (p. 9) of the C-884.

5.7 Connecting Digital Inputs and Outputs

The digital inputs and outputs on the **I/O** socket of the C-884 can be used as follows:

- Outputs: Triggering of external devices; see "Digital Output Signals" (p. 91).
- Inputs: Use in macros (p. 103) and/or as source for the reference and limit switch signals of the axis (p. 103)

5.7.1 Connecting the Digital Outputs

INFORMATION

Digital output signals are available on pins 8, 9, 18 and 26 of the **I/O** socket.

Tools and accessories

- Suitable Sub-D 26-pin (m) connector, with cable
- Device to be triggered having digital input for TTL signals

Connecting a device to be triggered

- Connect a suitable device to one of pins 8, 9, 18 or 26 of the **I/O** socket of the C-884.

5.7.2 Connecting the Digital Inputs**INFORMATION**

Digital TTL input signals can be fed via pins 7, 16, 17, and 25 of the **I/O** socket into the C-884.

Tools and accessories

- Suitable signal source: If the digital inputs are to be used as the source for the reference point and limit switch signals of an axis, the signal level may only change once across the entire travel range.
- Suitable Sub-D 26-pin (m) connector, with cable

Connecting a digital signal source

- Connect a suitable signal source to one of pins 7, 16, 17, und, 25 to the **I/O** socket of the C-884.

5.8 Connecting Analog Signal Sources

The analog inputs on the **I/O** socket of the C-884 can be used as follows:

- Use in macros (p. 106): Details and examples of macros are found in "Controller Macros" (p. 117).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

INFORMATION

Analog input signals (-10 V to 10 V) can be fed via pins 1, 10, 11, and 19 of the **I/O** in socket into the C-884.

Tools and accessories

- Suitable signal source
- Suitable Sub-D 26-pin (m) connector, with cable

Connecting an analog signal source

- Connect a suitable signal source to one of pins 1, 10, 11, and 19 of the **I/O** socket of the C-884.

5.9 Installing the PC Software

The communication between the C-884 and a PC is necessary to configure the C-884 and send motion commands using the commands of the GCS. Various PC software applications are available for this purpose.

5.9.1 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-884.CD_Setup.exe** file in the installation directory (root directory of the CD).

The **InstallShield Wizard** window for the installation of programs and manuals for the C-884 opens.

2. Follow the instructions on the screen.

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

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

- Driver for use with NI LabVIEW software
Exception: The *Analog LabVIEW drivers* component is provided for some PI controllers. This component is only available through user-defined installation.
- Dynamic program library for GCS
- PIMikroMove
- PC software for updating the firmware of the C-884
- PI Update Finder for updating the PC software
- For controllers that have a USB interface for communication with the PC: USB drivers

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

Installing the PC software in Linux

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

You can select individual components for installation.

5.9.2 Installing Updates

PI is constantly improving the PC software.

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

Requirements

- ✓ Active connection to the Internet.
- ✓ If your PC uses a Windows operating system:
 - You have installed (p. 51) the PI Update Finder from the product CD.
 - You have the A000T0028 Technical Note for the PI Update Finder ready. You will find the document on the product CD.
 - If the PC to be updated is **not** directly connected to the Internet:
You have the A000T0032 Technical Note for the PI Update Finder ready. You will find the document on the product CD.
- ✓ If your PC uses a Linux operating system:
 - You have the access data (user name and password) for the C-884. Information regarding the access data can be found in the file "C-884_Releasenews_V_x_x_x.pdf" (x_x_x: Version number of the CD) in the \Manuals folder on the product CD.

Updating the PC software and PISTAGES3.DB in Windows

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

Updating the PC software in Linux

1. Open the website www.pi.ws.
2. Click **Login**.
3. Log in with the user name and password for the C-884.
4. Click **Search**.
5. Enter the product number up to the period (e. g., C-884) into the search field.
6. Click **Start search** or press the **Enter** key.
7. Open the corresponding product detail page in the list of search results:
 - a) If necessary: Scroll down the list.
 - b) If necessary: Click **Load more results** at the bottom of the list.
 - c) Click the corresponding product in the list.
8. 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 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 into a separate installation directory.
12. Go to the **linux** subdirectory in the directory with the unpacked files.
13. Unpack the archive file in the **linux** directory by entering the command `tar -xvpf <name of the archive file>` on the console.
14. Read the accompanying information on the software update (readme file and/or "C-884_Releasenews_V_x_x_x.pdf" file) and decide whether the update makes sense for your application.
 - If no: Stop the update procedure.
 - If yes: Go through the following steps.
15. Log into the PC as superuser (root privileges).
16. Install the update.

INFORMATION

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

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

Updating PISTAGES3.DB on Linux

1. Contact the customer service department (p. 301) to get the latest version of the positioner database PISTAGES3.DB.
2. Log into the PC as superuser (root privileges).
3. Install the update that you received from our customer service department onto your PC.

5.9.3 Installing Custom Positioner Databases

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

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

In order for the parameter set to be selected in the PC software, it must first be inserted into the positioner database PISTages3 with the Import PI Custom Stage program.

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

The parameter set from the custom positioner database is inserted into PISTages3.

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

- a) Update the PISTages3 database on your PC, see "Installing updates" (p. 52).
- b) Repeat the installation of the custom positioner database.

5.10 Connecting the PC

The communication between the C-884 and a PC is necessary to configure the C-884 and to command motion with the GCS commands. The C-884 has the following interfaces for this purpose:

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

In this section, you learn how to make the proper cable connections between the C-884 and a PC and in a TCP/IP network. All other steps required for establishing communication between the C-884 and PC are described in the following sections:

- "Establishing Communication via the TCP/IP Interface" (p. 62)
- "Establishing Communication via RS-232" (p. 59)
- "Establishing Communication via USB" (p. 60)

5.10.1 Connecting the C-884 via the TCP/IP Interface

Prerequisites

- ✓ If the C-884 is to be directly connected to the PC:
The PC has a free RJ45 Ethernet connection socket.
- ✓ If the C-884 and a PC are to be operated together in a network:
A free access point to the network is available for the C-884; a suitable hub or switch is connected to the network for this purpose if necessary.

Tools and accessories

- If the C-884 is to be directly connected to the PC:
crossover network cable (C-815.563 in the scope of delivery)
- If the C-884 is to be connected to a network access point:
straight-through network cable (C-815.553 in the scope of delivery)

Connecting the C-884 directly to the PC

- Connect the RJ45 socket on the front panel of the C-884 to the RJ45 Ethernet connection socket of the PC via the included crossover network cable.

Connecting the C-884 to the network in which the PC is also located

- Connect the RJ45 socket on the front panel of the C-884 with the network access point via the included straight-through network cable.

5.10.2 Connecting the C-884 via the RS-232 Interface

Prerequisites

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

Tools and accessories

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

Connecting the C-884 to the PC

- Connect the **RS-232** socket on the front panel of the C-884 and the RS-232 interface of the PC (a Sub-D 9(m) panel plug) to the null-modem cable.

5.10.3 Connecting the C-884 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-884 to the PC

- Connect the USB socket of the C-884 and the USB interface of the PC with the USB cable.

6 Startup

In this Chapter

General Notes on Startup	57
Switching on the C-884	58
Establishing Communication.....	59
Starting Motion.....	68
Setting the Notch Filter.....	73
Optimizing the Servo Control Parameters.....	79

6.1 General Notes on Startup

NOTICE



Damage due to disabled limit switch evaluation!

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

- Avoid motion in open-loop operation.
- If motion in open-loop operation is necessary:
 - Set the control value with the SMO command so that the axis moves with low velocity.
 - Stop the axis in time. For this purpose, use the #24, STP or HLT command, or set the control value to zero with the SMO command.
- Do not disable the evaluation of the limit switches by the C-884 via parameter setting.
- Check the function of the limit switches at about 10 % to 20 % of the maximum velocity.
- In the event of a malfunction of the limit switches, stop the motion immediately.

INFORMATION

The communication between the C-884 and a PC can be used to configure the C-884 and send motion commands with the commands of the GCS.

- Communication is possible via the RS-232 interface and 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-884 (TCP/IP, RS-232, and 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-884.
- Configure the triggering of axis motion via the interfaces with the **Inhibit Motion Commands** parameter (ID 0x130); for further information, see "Motion Triggering" (p. 23).

6.2 Switching on the C-884

INFORMATION

The C-884 is intended for closed-loop operation with position sensors (servo mode On). After switch-on, open-loop operation is active by default (servo mode Off).

- Get the current operating mode with the `SVO?`, `#4` or `SRG?` commands.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a startup macro that starts the C-884 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 125).
- Avoid motion in open-loop operation.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ The C-884 has been installed properly (p. 45).

Switching on the C-884

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

The C-884 performs the following steps:

- The C-884 boots the firmware components. During the booting process, the **PWR** and **STA** LEDs on the front panel of the C-884 flash one after the other.
- The C-884 copies information from the nonvolatile memory to the volatile memory.

After booting, the LEDs show the status of the C-884:

- The **PWR** LED lights up permanently green: Booting of the Linux firmware component is finished.
 - The **STA** LED lights up continuously green: Booting the DSP and FPGA firmware is finished and the C-884 is ready for normal operation.
- If the **PWR** and **STA** LEDs do not light up after switching on, contact our customer service department (p. 301).

6.3 Establishing Communication

The procedure for PIMikroMove is described in the following.

6.3.1 Establishing Communication via the RS-232 Interface

INFORMATION

The interface parameters of the C-884 for RS-232 communication have the following default settings:

- Baud rate (RSBAUD): 115200

To establish communication successfully, the baud rates of the C-884 and the PC must match, for further information, see "Communication Interfaces" (p. 16).

INFORMATION

The following commands are available for the interface parameters of the C-884:

- Values in the nonvolatile memory:
 - Get with IFS? (p. 200)
 - Set with IFS (p. 198)
- Values in the volatile memory:
 - Get with IFC? (p. 197)
 - Set with IFC (p. 196)
 - Get possible settings with `MAN? IFC`

For querying and setting the interface parameters, it is recommended to use the **Configure Interface** window in PIMikroMove. There you can comfortably read, modify and save the values of the interface parameters. For details, see the PIMikroMove manual.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ The C-884 is connected to the RS-232 interface of the PC (p. 55).
- ✓ The C-884 is switched on (p. 58).
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 51).
- ✓ You have read and understood the manual of the PC software used. The software manuals are on the product CD.

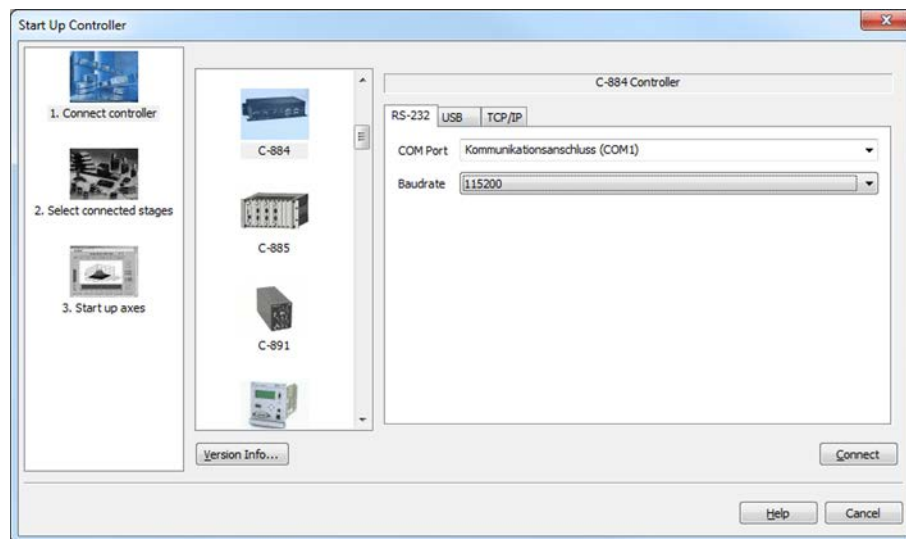
Establishing communication via RS-232

1. Start PIMikroMove.

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

- If the **Start up controller** window does not automatically open, select the **Connections > New...** menu item in the main window.
- 2. Select **C-884** in the field for controller selection.
- 3. Select the **RS-232** tab on the right-hand side of the window.
- 4. In the **COM Port** field, select the COM port of the PC to which you have connected the C-884.
- 5. In the **Baudrate** field, set the value that is set for the C-884.

This adapts the baud rate of the PC to the baud rate of the C-884.



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

If communication was established successfully, PIMikroMove guides you through the configuration of the C-884 for the connected positioner; see "Starting Motion" (p. 68).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 295).

6.3.2 Establishing Communication via the USB Interface

INFORMATION

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

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ The C-884 is connected to the USB interface of the PC (p. 56).

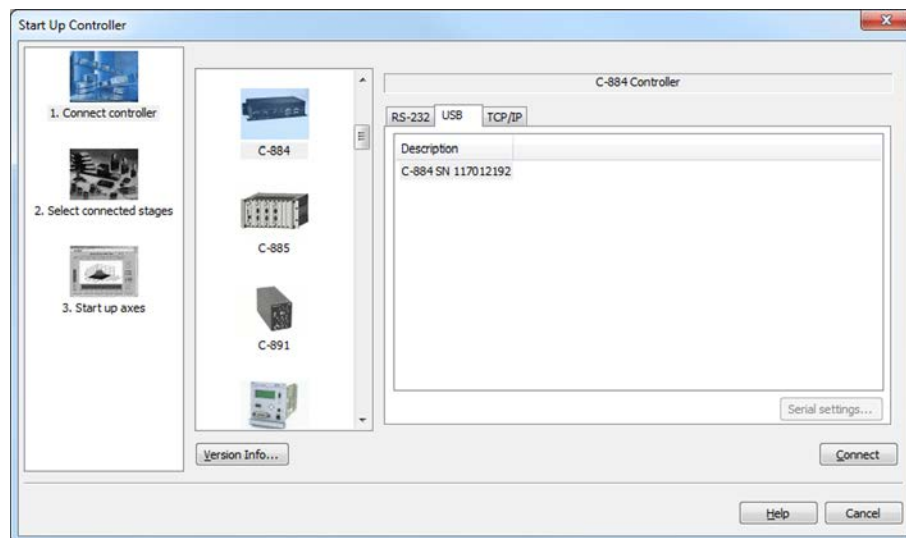
- ✓ The C-884 is switched on (p. 58).
- ✓ The PC is switched on.
- ✓ The required software and USB drivers are installed on the PC (p. 51).
- ✓ You have read and understood the manual of the PC software used. The software manuals are on the product CD.

Establishing communication via USB

1. Start PIMikroMove.

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

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



2. Select **C-884** 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-884** connected.
5. Click **Connect** to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the C-884 for the connected positioner; see "Starting Motion" (p. 68).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 295).

6.3.3 Establishing Communication via the TCP/IP Interface

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

- **Network with DHCP server:** No adjustment of the factory settings of the C-884 interface parameters is necessary
- **Network without DHCP server or direct connection** (C-884 directly connected to the Ethernet connection socket of the PC):
 - The startup behavior of the C-884 for configuring the IP address must be changed so that the C-884 uses a static IP address.
 - The IP addresses and subnet masks of the C-884 and PC as well as all other network devices must be compatible with each other.

INFORMATION

The following commands are available for the interface parameters of the C-884:

- Values in the nonvolatile memory:
 - Get with IFS? (p. 200)
 - Set with IFS (p. 198)
- Values in the volatile memory:
 - Get with IFC? (p. 197)
 - Set with IFC (p. 196)
 - Get possible settings with `MAN? IFC`

For querying and setting the interface parameters, it is recommended to use the **Configure Interface** window in PIMikroMove. There you can comfortably read, modify and save the values of the interface parameters. For details, see the PIMikroMove manual.

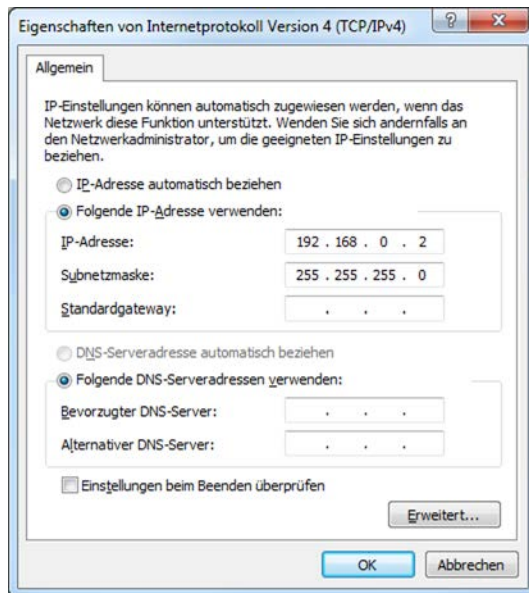
Requirements

- ✓ You have established communication between the C-884 and the PC via RS-232 (p. 59) or USB (p. 60) in PIMikroMove in order to determine the settings of the C-884 and to be able to change them if necessary.

Determining the IP address and subnet mask of the PC

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

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

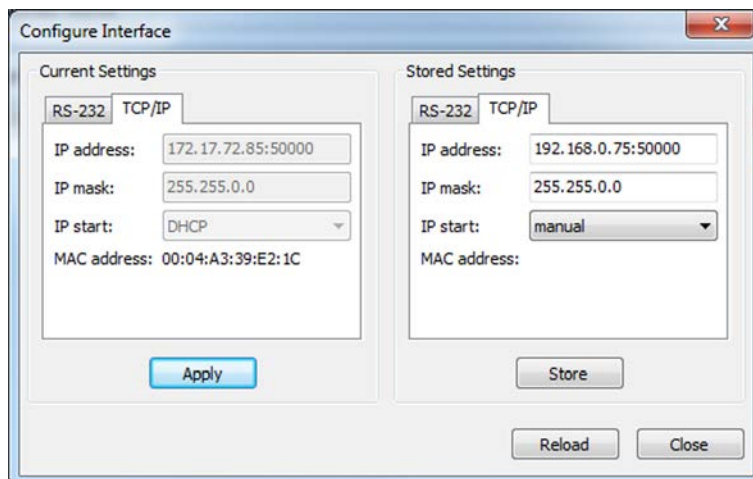


The figure shows example settings that may not apply to your system.

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

Determine the IP address and subnet mask of the C-884 and adapt the starting behavior

1. In the main window, open the **Configure Interface** window in PIMikroMove by selecting the **C-884 > Configure interface** menu item.
2. In the **Configure Interface** window, select the **TCP/IP** tab in the **Stored Settings** area.



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

3. Write down the values from the following fields of the **TCP/IP** tab in the **Stored Settings** area:
 - **IP address**

– **IP mask**

4. On the **TCP/IP** tab, select the *manual* value in the **IP start** field in the **Stored Settings** area.

This changes the startup behavior of the C-884 so that it uses a static IP address.

5. Save the changed setting of the startup behavior in the nonvolatile memory of the C-884:
 - a) Click the **Store** button in the **Stored Settings** area. The **Store interface settings** dialog opens.
 - b) In the **Store interface settings** dialog, click **Store settings**. The dialog closes.

Adapting IP settings of the PC

- If the PC is already configured for using a static IP address and you want to leave the PC settings unchanged, continue with the section "Adapting the IP Settings of the C-884" (p. 65).
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 (see "Determining the IP Address and Subnet Mask of the PC" (p. 62)).
 2. Adapt the IP address and the subnet mask to the settings of the C-884:
 - a) Copy the subnet mask of the C-884 for the subnet mask on the PC.
 - b) Copy the first three sections of the IP address of the C-884 for the IP address on the PC (see "Determining the IP Address and Subnet Mask of the C-884, Adapting the Starting Behavior").
 - c) Make sure that the last section of the IP address on the PC differs from the last section of the IP address of the C-884 and is not "255" or "0".

Example:

If the C-884 has the subnet mask 255.255.255.0, set the subnet mask 255.255.255.0 on the PC.

If the C-884 has the IP address 192.168.0.75, set the IP address 192.168.0.76 on the PC.

3. Confirm the settings with the **OK** button.
4. If further network devices have to be adapted:

Adapt the IP addresses and subnet masks as in the previous steps.
Assign a separate, clear IP address to each network device.
No IP address may occur twice in the same network.
5. Close the connection of the C-884 via the RS-232 or USB interface:
 - a) Close the **Configure Interface** window by clicking **Close**.
 - b) In the main window of PIMikroMove, select the **Connections > Close > C-884** menu item.
6. Switch off the C-884.
7. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 65).

Adapting IP settings of the C-884

1. Adapt the settings of the C-884 to those of the PC (see "Determining the IP address and subnet mask of the PC" (p. 62)). To do this make the following settings on the **TCP/IP** tab in the **Stored Settings** area in the **Configure Interface** window of PIMikroMove.
 - a) Change the subnet mask in the **IP mask** field to the subnet mask of the PC.
 - b) Change the IP address (format: xxx.xxx.xxx.yyy) in the **IP address** field, whereby the following applies:
 - xxx.xxx.xxx. matches the first three sections of the IP address of the PC.
 - yyy differs from the last section of the IP address of the PC and every other device in the same network.
 - yyy is not "255" and not "0" and is in the address range which is given by the last section of the subnet mask.
 - The port address "50000" must not be changed.

Example:
If the IP address of the PC is 192.168.0.2 and no other device has the IP address 192.168.0.3, set the IP address 192.168.0.3:50000.
2. Save the changed settings in the nonvolatile memory of the C-884:
 - a) Click the **Store** button in the **Stored Settings** area. The **Store interface settings** dialog opens.
 - b) In the **Store interface settings** dialog, click **Store settings**. The dialog closes.
3. Close the **Configure Interface** window by clicking **Close**.
4. Close the connection of the C-884 via the RS-232 or USB interface by selecting the **Connections > Close > C-884** menu item in the main window of PIMikroMove.
5. Switch off the C-884.
6. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 65).

Establishing Communication via TCP/IP in the PC Software

CAUTION



Risk of crushing from unexpected motion

When communication between the C-884 and the PC has been established via TCP/IP, the PC software offers all controllers present in the same network for selection. After a C-884 has been selected for the connection, all commands are sent to this controller. If the wrong controller is selected, unexpected motion of the mechanical system could be commanded and cause minor crush injuries to the operating and maintenance staff.

- If several C-884 are displayed in the PC software, make sure that you select the right C-884.

INFORMATION

Communication via TCP/IP can fail if the network cable was connected to the Ethernet socket of the C-884 while the C-884 was switched on.

- If communication cannot be established, switch the C-884 off. Now reconnect the network cable and switch the C-884 on again.

INFORMATION

- For communication via TCP/IP, the C-884 only has one unchangeable port (50000) available that cannot be used for more than one connection at a time.

INFORMATION

In the list of controllers found in the same network, the C-884 is shown with its model name, followed by the 9-digit serial number.

Example of the display of a C-884 to which no connection has been established via TCP/IP yet:

C-884.4DC SN 123456789 -- listening on port 50000 -- 172.17.72.55 50000

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

- If several C-884 are connected with the same network via TCP/IP, identify the C-884 to be connected in the list of found controllers on the basis of its serial number. The serial number of the controller can be found on the type plate of the C-884.
- If you have established communication in advance via USB (p. 60) or RS-232 (p. 59) (e. g., with PIMikroMove or PITerminal), you can also find the serial number of the C-884 in the response to the `*IDN?` (p. 145) command.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ The C-884 is connected to the network or directly to the PC via its RJ45 Ethernet socket (p. 55).
- ✓ If the C-884 is connected to a network:
The PC to be used for communication with the C-884 is appropriately connected to the same network as the C-884.
- ✓ If the used network does not have a DHCP server or if the C-884 is directly connected to the Ethernet connection socket of the PC:
By adapting the interface parameters, you have set the correct startup behavior for configuring the IP address of the C-884 and adapted the IP addresses and subnet masks of the C-884 and PC to each other (p. 59).
- ✓ If several C-884s are connected to the same network via their TCP/IP interfaces: You have the serial number of the C-884 ready with which the communication is to be established. The serial number can be found on the type plate (p. 12) of the C-884.
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 51).

- ✓ You have read and understood the manual for the PC software used. The software manuals are on the product CD.

Establishing communication via TCP/IP

1. Start PIMikroMove.

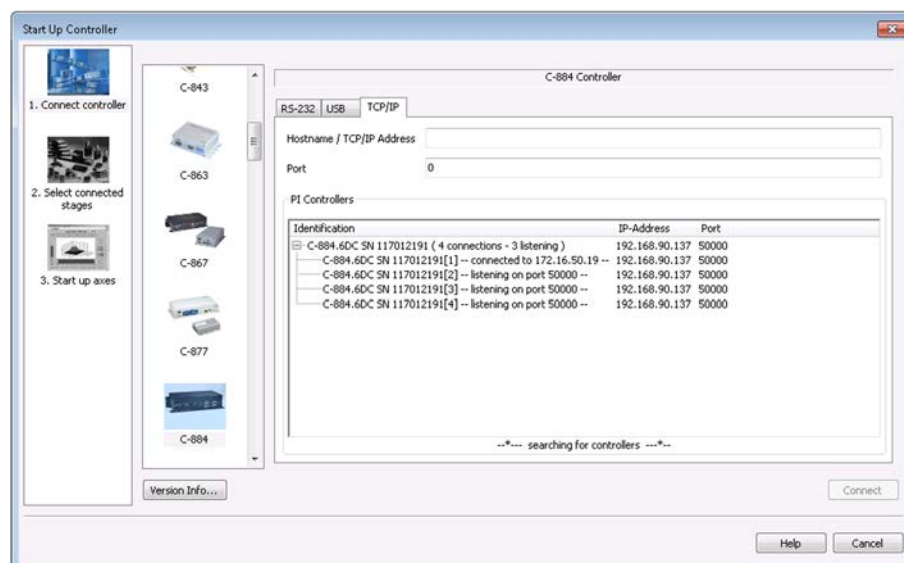
The **Start up controller** window opens with the **Connect controller** step.

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

2. Select **C-884** in the field for controller selection.

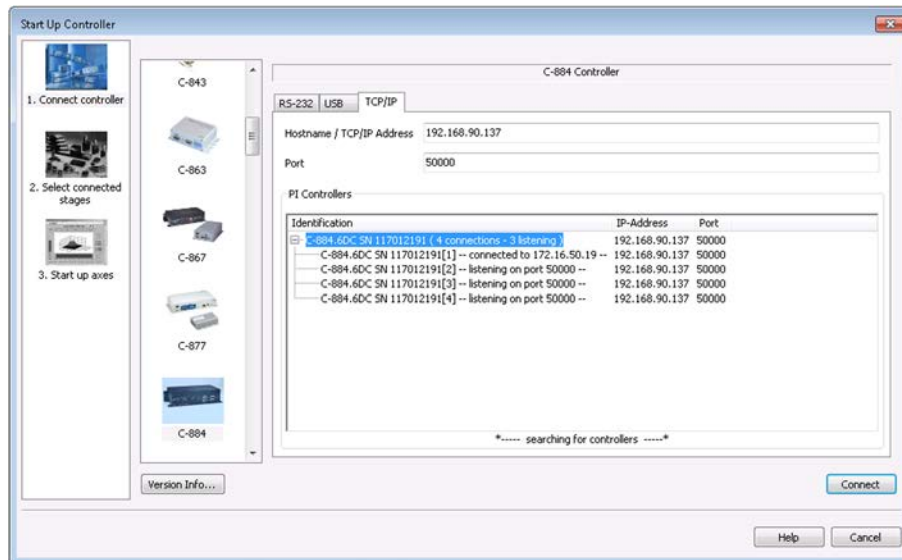
3. Select the **TCP/IP** tab on the right-hand side of the window.

The software now searches in the network for all controllers of the type C-884. After a successful search, all controllers that are in the same network are shown in the **PI Controllers** field.



4. Click the entry of your C-884 model in the list of found controllers. This must show the status "listening on port 50000".
 - If several entries with the same name are shown, identify your C-884 on the basis of its nine-digit serial number.
 - If the C-884 is not displayed in the list of the controllers found, check the network settings (p. 295). Consult your network administrator if necessary.
 - Do **not** select a controller that is already connected via TCP/IP (status "connected to ..."). Otherwise, an error message will be displayed as soon as you try to establish communication with this controller.

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



5. Check the IP address in the **Hostname / TCP/IP Address** field and the port number in the **Port** field.
6. Click the **Connect** button to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the C-884 for the connected positioner; see "Starting Motion" (p. 68).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 295).

6.4 Starting Motion

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

- Configuration of the C-884 for the connected positioner
- Switching on the servo mode (closed-loop operation)
- Performing a reference move; details see "Reference Point Definition" (p. 38).

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

NOTICE**Selecting an incorrect positioner type**

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

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

NOTICE**Oscillation!**

Unsuitable settings of the notch filter and the C-884's servo control parameters can cause the positioner to oscillate. Oscillation can damage the positioner and/or the load affixed to it.

- Secure the positioner and all loads adequately.
- If the Positioner is oscillating (unusual operating noise), immediately switch off the servo mode or the C-884.
- Only switch on the servo mode after you have modified the settings of the notch filter and the servo control parameters of the C-884; see "Setting the Notch Filter" (p. 73) and "Optimizing Servo Control Parameters" (p. 79).
- If, due to a very high load, oscillation occurs during the reference move, follow the instructions for the reference move in "Troubleshooting" (p. 295).

INFORMATION

After communication has been established between the C-884 and the PC, PIMikroMove guides you through the configuration of the C-884 for the connected positioner. Selection of the configuration steps offered by PIMikroMove is based on evaluation of the following parameter values in the volatile memory of the C-884:

- **Stage Name** (ID 0x3C): The value is used by PIMikroMove as a criterion for finding a suitable parameter set in the positioner databases.
- **Stage Type** (ID 0x0F000100): The value was loaded from the ID-Chip (p. 15) of the connected positioner when the C-884 is switched on.

Possible configuration steps:

- When the values of the parameters 0x3C and 0x0F000100 are identical, PIMikroMove assumes that all parameters of the C-884 have already been adapted to the connected positioner. The **Start up controller** window goes directly to the **Start up axes** step, where the reference move can be started.
- If the values of the parameters 0x3C and 0x0F000100 are not identical, the **Stage Type Configuration** window opens. The **Yes, configure for ...** button can be used to load a suitable parameter set from a positioner database to the C-884. After the parameter set has been loaded, the **Start up controller** window goes to the **Start up axes step**. If a matching parameter set is not in the positioner databases, a corresponding notice will appear in the **Stage Type Configuration** window.
- If the value of the parameter 0x0F000100 is empty because the positioner does not have an ID chip, for example, the **Start up controller** window will go to the **Select connected stages** step.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ PIMikroMove is installed on the PC (p. 51).
- ✓ You have read and understood the PIMikroMove manual. The manual is on the product CD.
- ✓ You have installed the latest version of the PISTAGES3.DB database onto your PC (p. 51).
- ✓ If PI provided a custom positioner database for your positioner, the dataset was imported (p. 54) into PISTages3.
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
- ✓ You have connected the C-884 to the positioner (p. 54).
- ✓ You have established communication with PIMikroMove between the C-884 and the PC (p. 59).

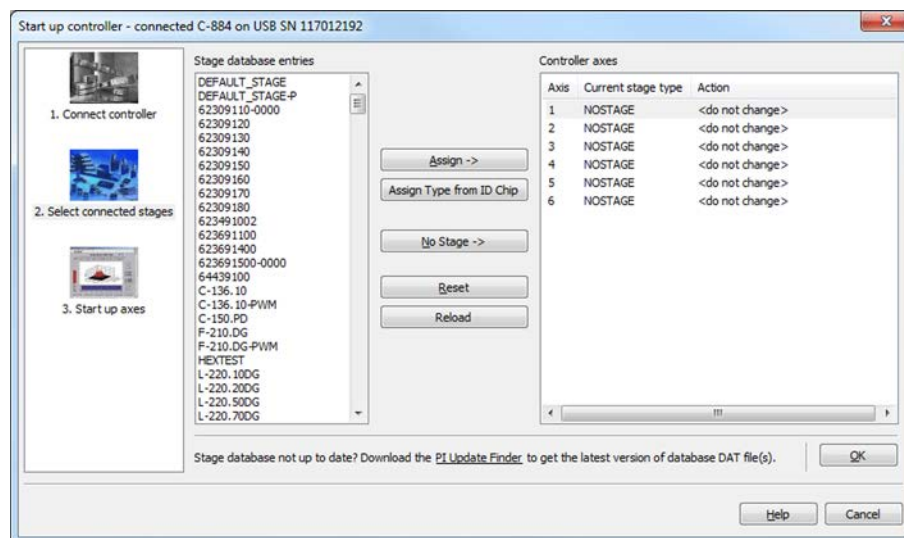
Starting motion with PIMikroMove

1. If one of the two following points applies, configure the C-884 for the connected positioner:
 - The **Stage Type Configuration** dialog has opened.
 - The **Select connected stages** step is displayed in the **Start up controller** window.

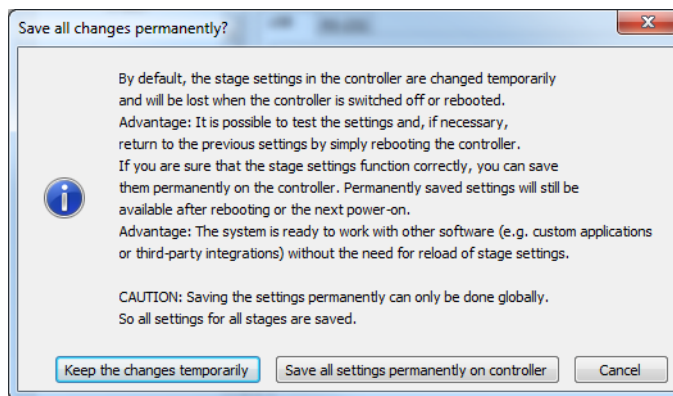
If the **Stage Type Configuration** dialog has opened:

- Click the **Yes, configure for ...** button to load the appropriate parameter set from the positioner database into the C-884. This opens the **Save all changes permanently?** dialog.

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



- a) Select the matching positioner type. You have two options:
 - Click on **Assign Type from ID Chip**.
 - Mark the appropriate positioner type in the **Stage database entries** list and click on **Assign**.
 - b) Confirm selection with **OK** to load the parameter settings for the selected positioner type from the positioner database into the C-884. This opens the **Save all changes permanently?** dialog.
2. Specify how you want to load the parameter settings into the C-884 in the **Save all changes permanently?** dialog box:



- Temporary load: Click **Keep the changes temporarily** to load the parameter settings into the volatile memory of the C-884. The settings are lost when the C-884 is switched off or rebooted.
- Load as default values: Click **Save all settings permanently on controller** to load the parameter settings into the nonvolatile memory of the C-884. The settings are available immediately after switching on or rebooting the C-884 and do not need to be reloaded.

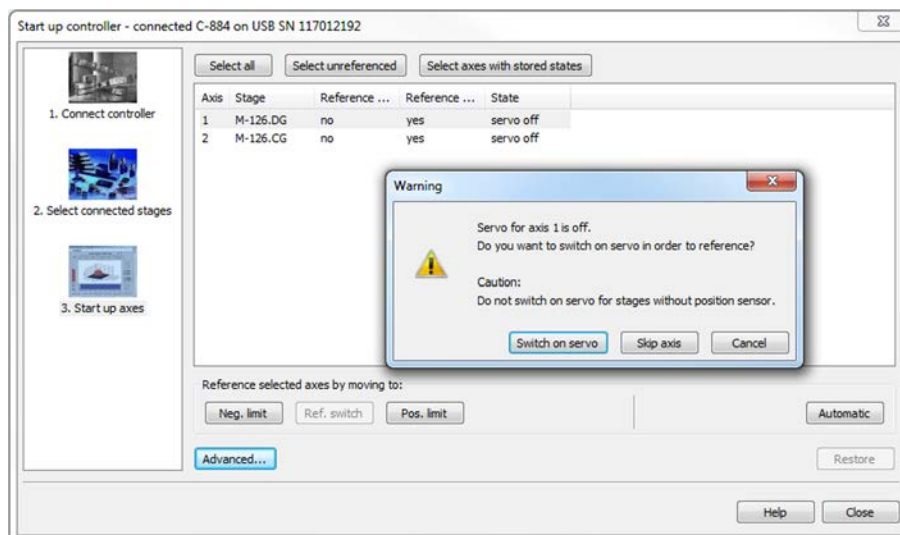
The **Start up controller** window changes to the **Start up axes** step.

3. In the **Start up axes** step, perform the reference move for the axis so that the controller knows the absolute axis position:

Options for starting the reference move:

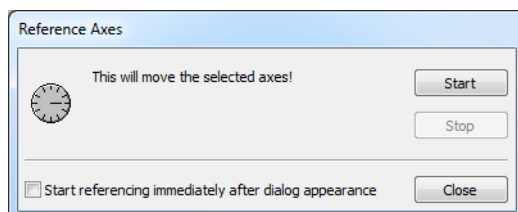
- If you want to start the reference move to the reference point switch, click on **Ref. switch**.
- If you want to start the reference move to the negative limit switch, click on **Neg. limit**.
- If you want to start the reference move to the positive limit switch, click on **Pos. limit**.

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



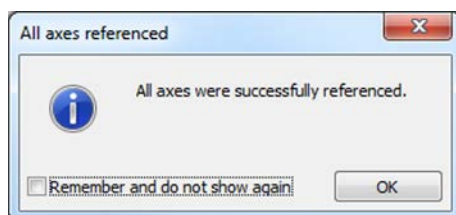
a) Switch on the servo mode by clicking on the **Switch on servo** button.

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



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

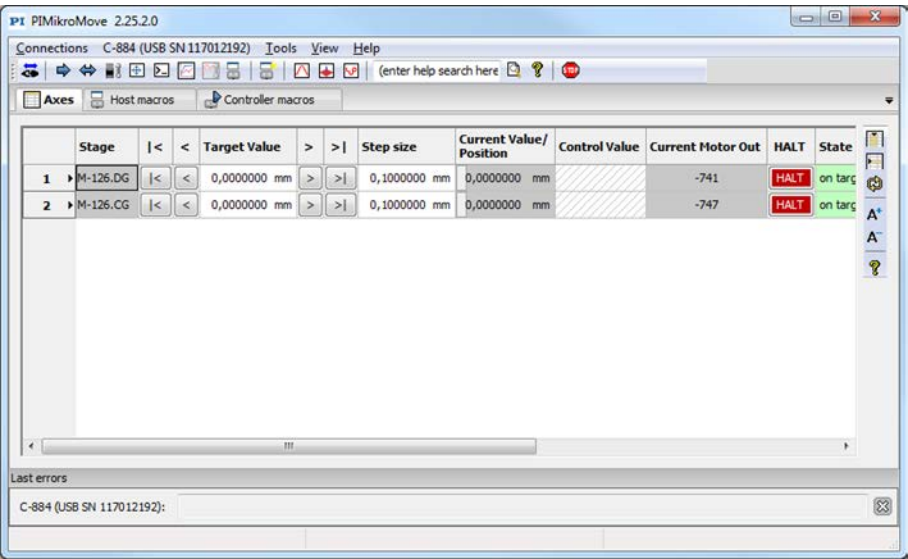
If the corresponding message is displayed after a successful reference move:



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

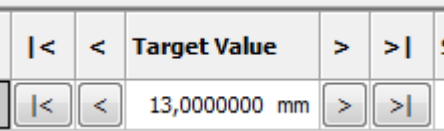
4. After a successful reference move, close the **Start up controller** window by clicking **Close**.

The main window of PIMikroMove opens.



5. Test the motion of the axis several times.

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



6.5 Setting the Notch Filter

The notch filter corrects the control value for the drive of the positioner connected to the C-884. The corrections by the notch filter take place in closed-loop and open-loop operation.

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

INFORMATION

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

- Set the notch filter before you optimize the servo control parameters (p. 79).

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

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

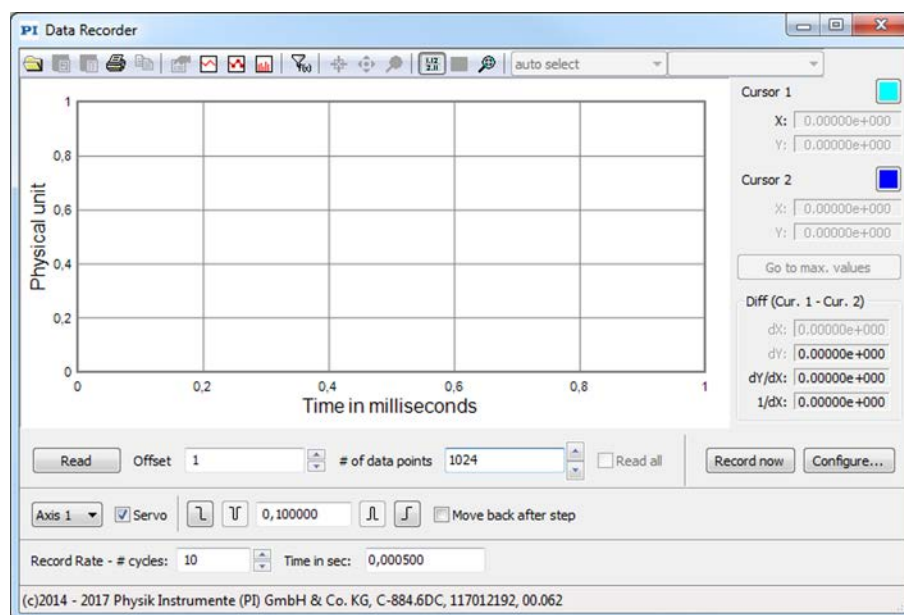
The procedure for PIMikroMove is described in the following.

Requirements

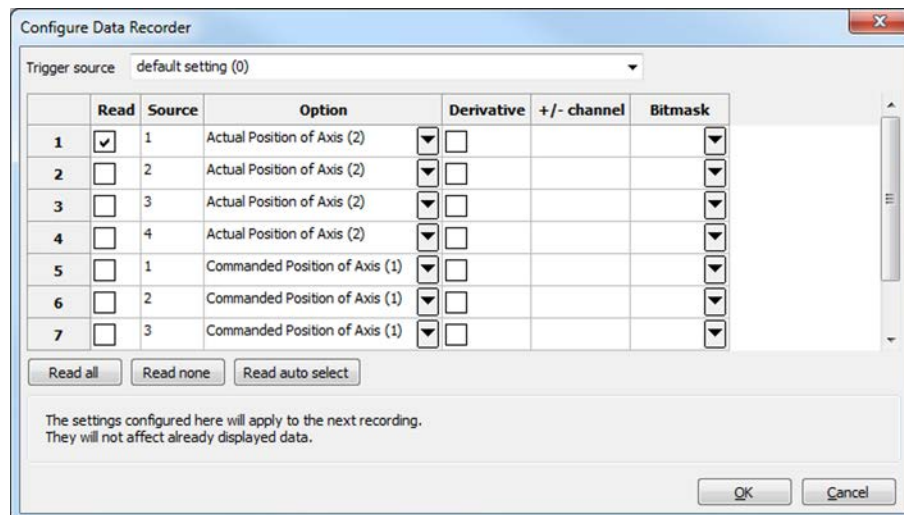
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
- ✓ Initial motion in the PIMikroMove was started (p. 68).
- ✓ All devices are still ready for operation.

Setting the notch filter


1. In the main window of PIMikroMove, open the **Data Recorder** window via the **C-884 > Show data recorder** menu item.
2. Switch off the servo mode with the **Servo** check box (uncheck).
3. Configure the data recorder.
 - a) Set the value 1 for the amplitude of the step to be performed (= 1 step in step mode).
 - b) Set the value 10 for the record table rate in the **Record Rate** field.
 - c) Set 8192 (or less) as the value for the number of data points to be read for the graphic display.



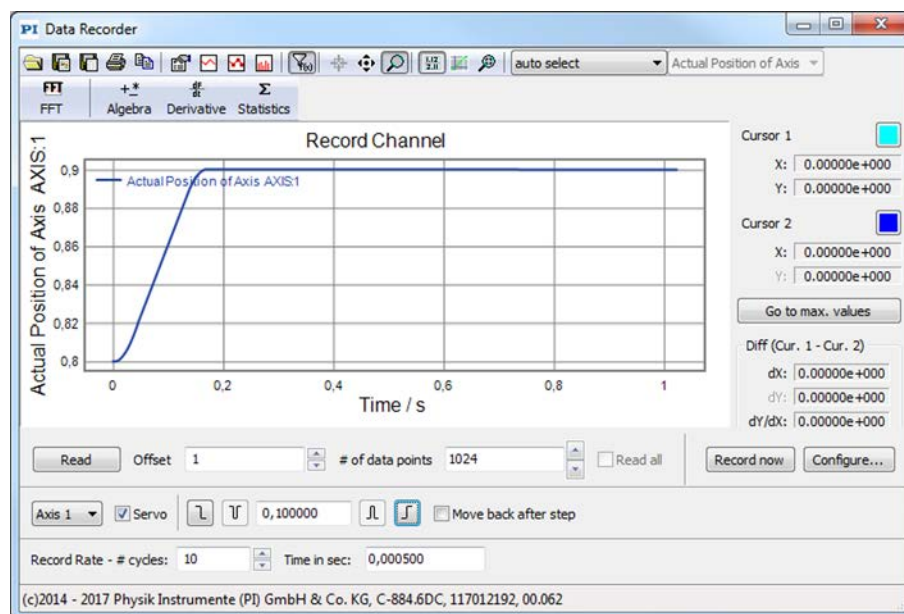
- d) Click the **Configure...** button and make sure that "Actual Position of Axis" is selected in the **Configure Data Recorder** window as the variable to be recorded.



Close the window with **OK**.


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

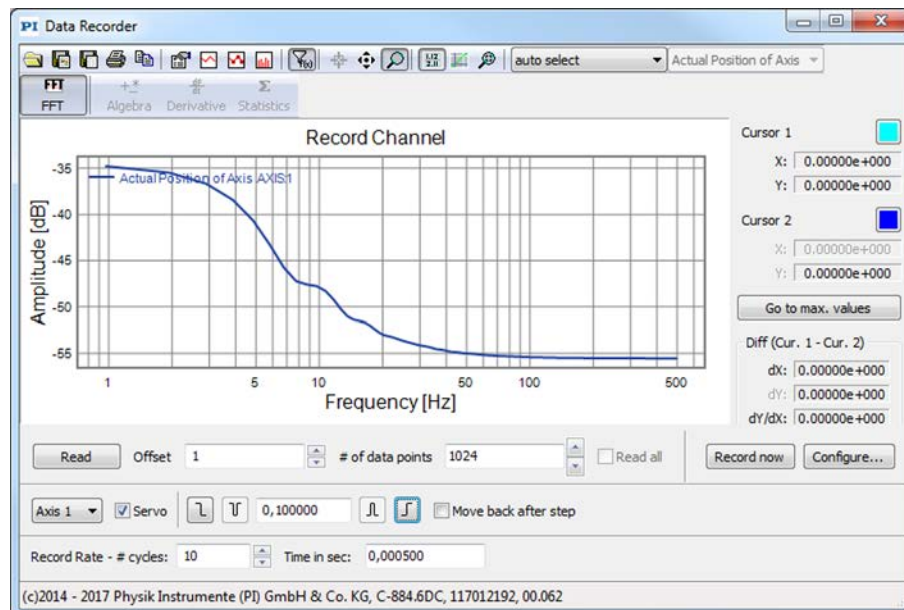
The axis performs the step and the step response is recorded and displayed graphically.






5. Determine the resonant frequency of the axis from the graphic display of the step response:

- a) Display the Data Toolbar via the  button.

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



- c) If necessary, enlarge the view by clicking the  button and, while pressing the left mouse button, dragging the mouse pointer, which has turned into a magnifying glass, over a section of the graphic display (clicking the right mouse button in the graphics field reduces the view back to the original size).
- d) Display the cursors in the graphic display by clicking the  button.
- e) Activate the cursor movement with the mouse by clicking the  button.

- f) Position cursor 1 and 2 over the resonant frequencies by clicking the **Go to max. values** button.
 The resonant frequency can be identified by the distinct maximum in the FFT graph. In the example shown, only one resonant frequency is visible; it is at approx. 1 Hz. This maximum value can be read in the **X:** field, in the **Cursor 1** area on the right next to the graphic display.

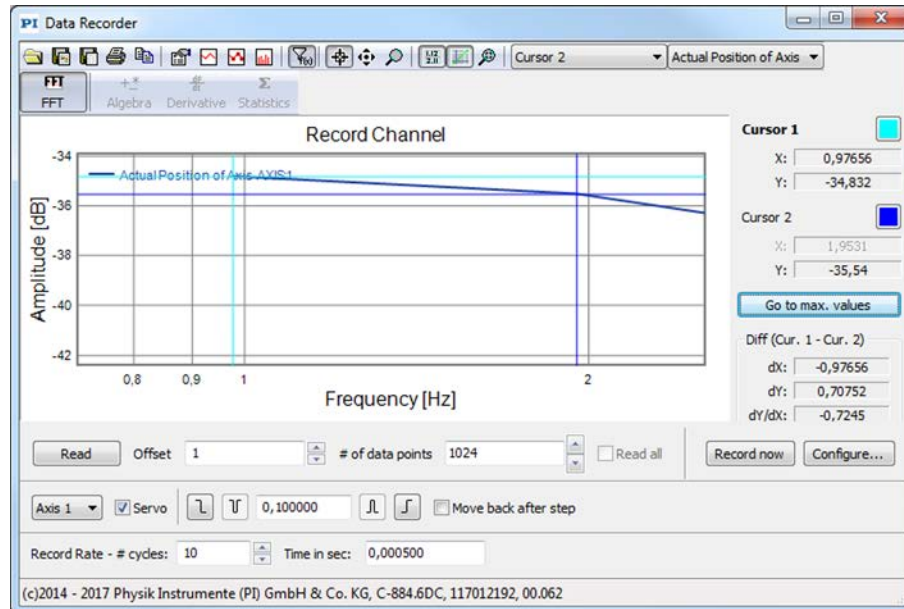
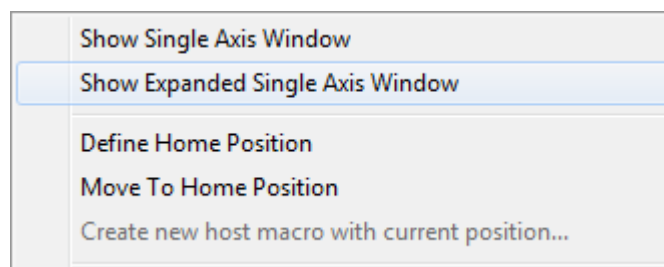


Figure 11: Step response as FFT graph: Resonant frequencies marked by the cursor

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



7. Set the notch filter:

The value of the **Notch Filter Frequency 1** parameter must be set to the resonant frequency determined in step 5. In the example shown, it is not necessary to adapt the parameter value for **Notch Filter Edge 1**.

Proceed as follows to enter the values:

- a) If the **Notch Filter Frequency 1** and **Notch Filter Edge 1** parameters are not in the list on the right-hand side of the window, click on **Configure View -> Select parameters...** and add them to the list.

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

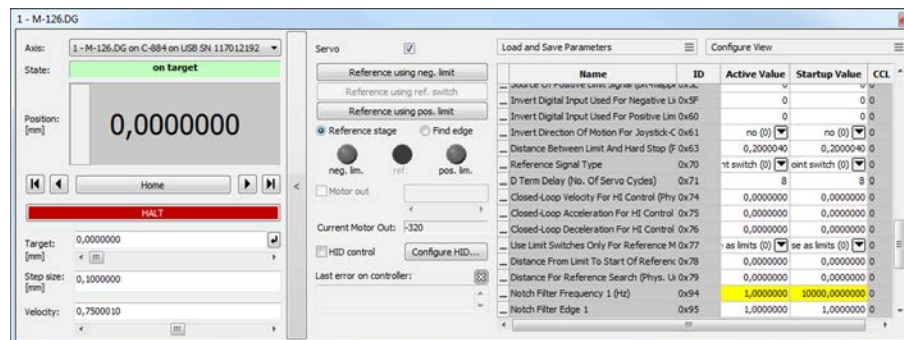
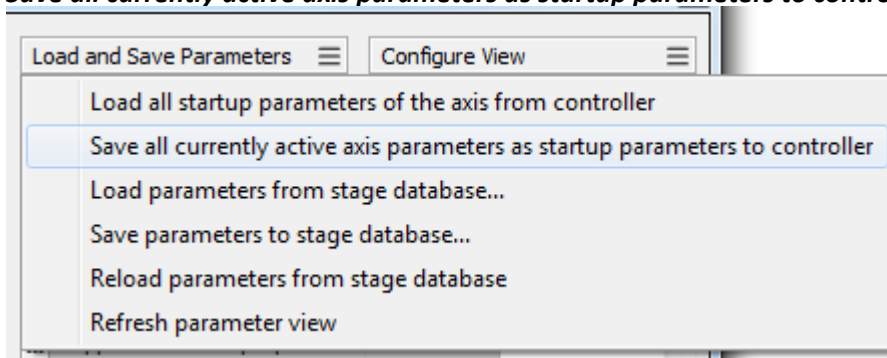


Figure 12: Step response as FFT graph: Resonant frequencies marked by the cursor

8. Save the new settings. You have the following options:
 - Save a parameter set in the positioner database on the PC by clicking on **Load and Save Parameters -> Save parameters to stage database...**, see "Creating or Modifying a Positioner Type" (p. 277).
 - Transfer the current values of the listed parameters from the volatile memory to the non-volatile memory of the C-884 by clicking **Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller**.



6.6 Optimizing the Servo Control Parameters

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

Typically, optimization is determined empirically, i.e, the behavior of the positioner is monitored with different values in closed-loop operation. Optimization is done via the following parameters, for details, see "Servo Algorithm and Other Control Value Corrections" (p. 29):

- **P-Term** (0x1)
- **I-Term** (0x2)
- **D-Term** (0x3)
- **I-Limit** (0x4)

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

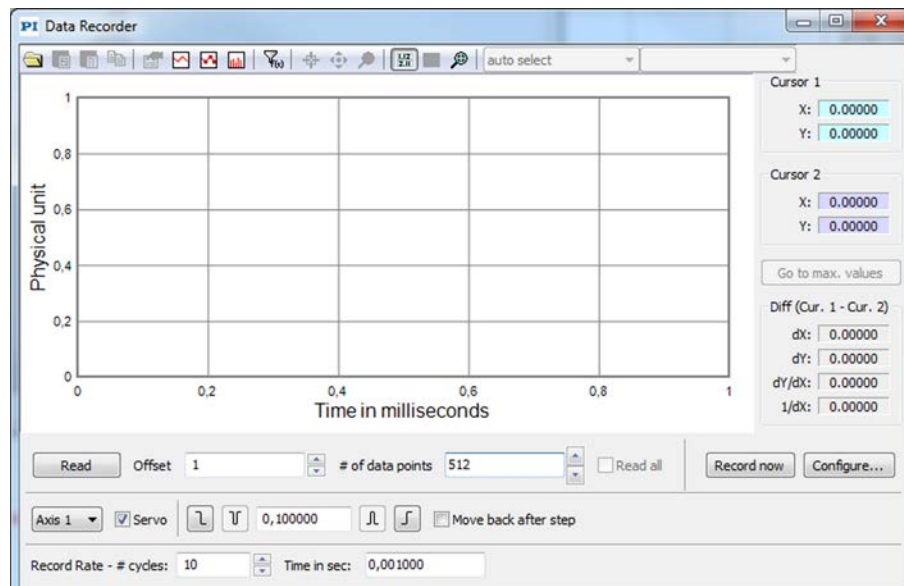
Requirements

- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
- ✓ With PIMikroMove, you have started initial motion (p. 68).
- ✓ If necessary, you have set the notch filter (p. 73).
- ✓ All devices are still ready for operation.

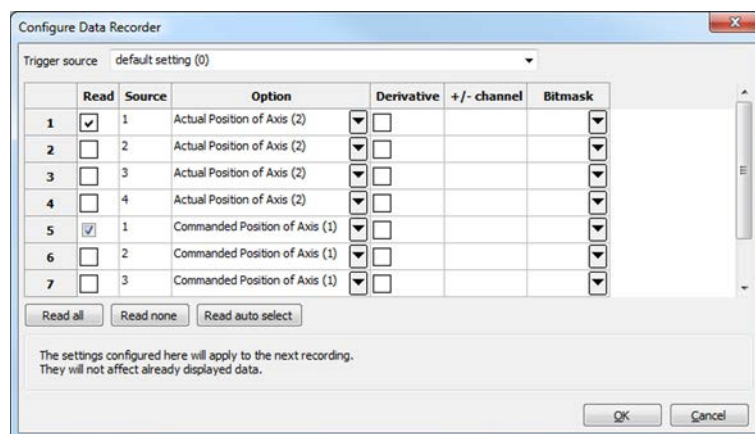
Checking the servo control parameters: Measuring the step response

1. In the main window of PIMikroMove, open the **Data Recorder** window via the **C-884 > Show data recorder** menu item.
2. Switch on the servo mode for the axis with the **Servo** check box (set check).
3. Configure the data recorder.
 - a) Set the size of the step to be made to a value that is typical for your application, e.g., 0.100000 (specified as physical units).
 - b) Set the value 10 for the record table rate in the **Record Rate - # cycles** field.

- c) Set the value 8192 (or less) for the number of data points to be read for the graphic display in the field **# of data points**.





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

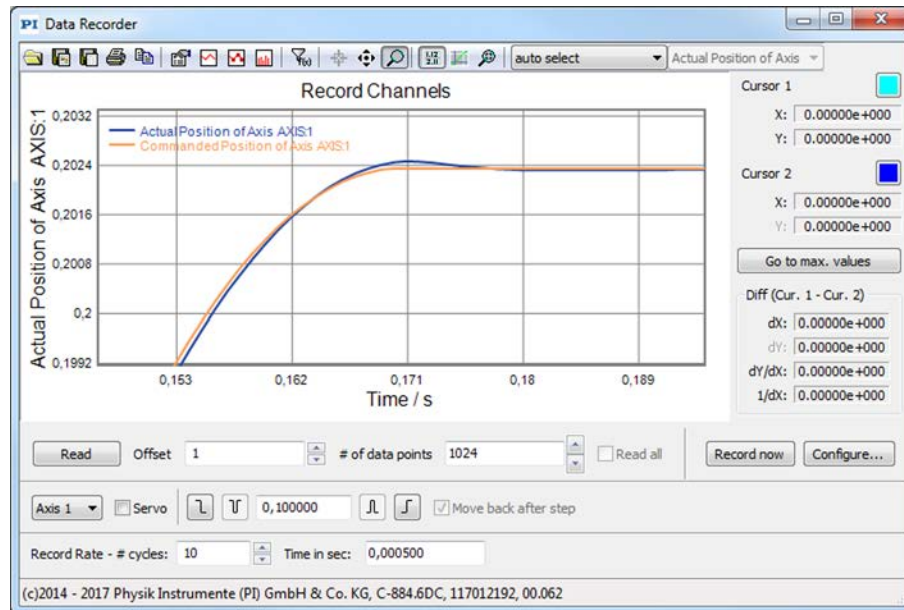


	Read	Source	Option	Derivative	+/- channel	Bitmask
1	<input checked="" type="checkbox"/>	1	Actual Position of Axis (2)	▼	<input type="checkbox"/>	▼
2	<input type="checkbox"/>	2	Actual Position of Axis (2)	▼	<input type="checkbox"/>	▼
3	<input type="checkbox"/>	3	Actual Position of Axis (2)	▼	<input type="checkbox"/>	▼
4	<input type="checkbox"/>	4	Actual Position of Axis (2)	▼	<input type="checkbox"/>	▼
5	<input checked="" type="checkbox"/>	1	Commanded Position of Axis (1)	▼	<input type="checkbox"/>	▼
6	<input type="checkbox"/>	2	Commanded Position of Axis (1)	▼	<input type="checkbox"/>	▼
7	<input type="checkbox"/>	3	Commanded Position of Axis (1)	▼	<input type="checkbox"/>	▼

Close the window with **OK**.

4. In the **Data Recorder** window, start the step in the positive direction as well as the recording by clicking the  button.
- The axis performs the step and the step response is recorded and displayed graphically.
5. Check the displayed step response.

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



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

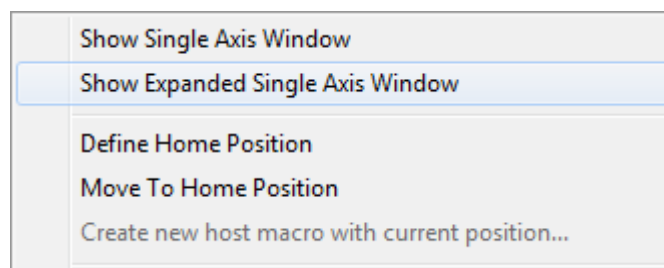
- You already have optimum parameter settings and do not have to do anything further.

If the result is not satisfactory:

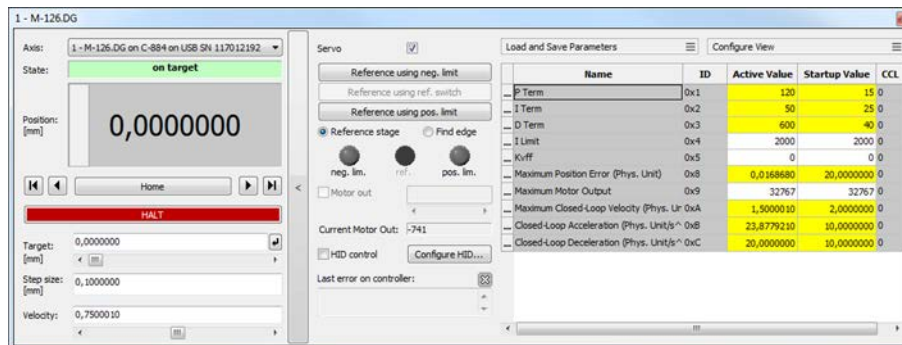
- Optimize the servo control parameters, see below.

Optimizing the servo control parameters

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



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



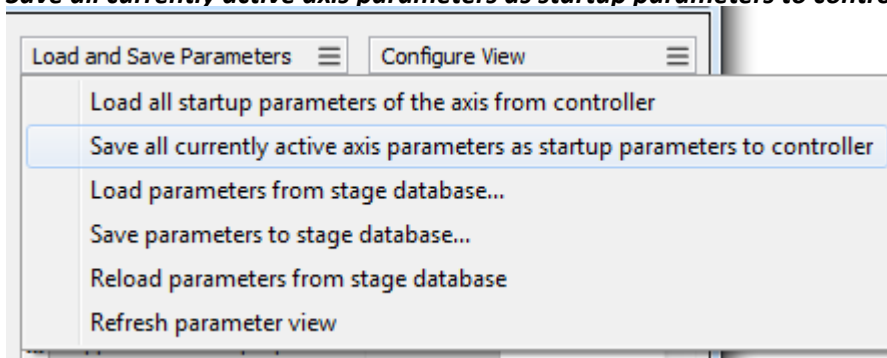
- a) If the parameter to be modified is not included in the list on the right-hand side of the window, click on **Configure view > Select parameters...** and add it to the list. You can also display certain groups of parameters or all axes-related parameters.
 - b) Type the new parameter value into the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

If the result is not satisfactory:

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

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

- Save a parameter set in the positioner database on the PC by clicking on **Load and Save Parameters -> Save parameters to stage database...**, see "Creating or Modifying a Positioner Type" (p. 277).
- Transfer the current values of the listed parameters from the volatile memory to the non-volatile memory of the C-884 by clicking **Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller**.



7 Operation

In this Chapter

Protective Functions of the C-884	83
Trajectories for Motion Paths	85
Data Recorder	89
Digital Output Signals.....	91
Digital Input Signals.....	101
Analog Input Signals.....	105
Control with a Human Interface Device	106
Controller Macros	117

7.1 Protective Functions of the C-884

7.1.1 Behavior with Motion Errors

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

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

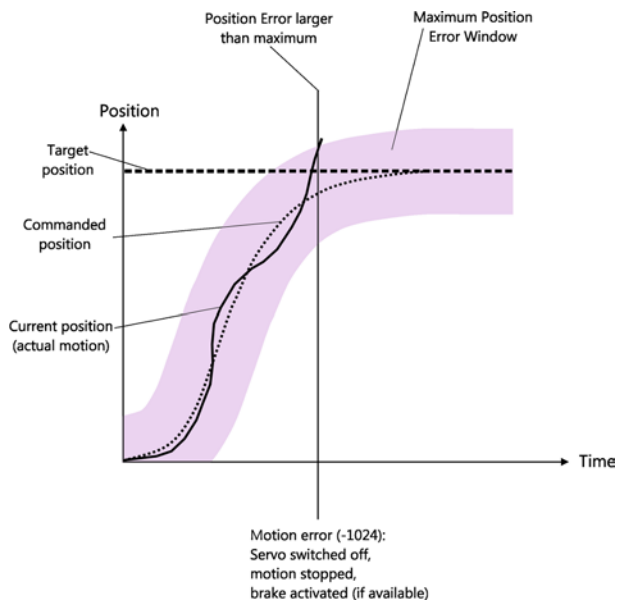
Motion errors can have the following causes, for example:

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

If a motion error occurs, the C-884 reacts as follows to protect the system against damage:

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

Then restore the operational readiness (p. 84) for the C-884.



INFORMATION

With the **CTO** (p. 153) and **TRO** (p. 240) commands, you can program the digital output lines of the C-884 so that they are activated in the case of motion errors. The programmed output lines remain activated until the error code is reset to 0. For details, see "Setting Up "Motion Error" Trigger Mode" (p. 96).

7.1.2 Re-establishing Readiness for Operation

NOTICE



Unintentional motion after brake deactivated!

If servo mode is switched off, e.g., after a motion error occurs, the brake of the positioner can be deactivated by command. Deactivating the brake can cause the positioner to move unintentionally.

- Secure the positioner against unintentional motion before you deactivate the brake by command!

Re-establishing readiness for operation

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

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

2. Check your system and make sure that all axes can be moved safely.
3. Switch on the servo mode for the axis in question with the **SVO** command (p. 228).

When switching on the servo mode, the target position is set to the current axis position and the brake is deactivated, if necessary. Now the axis can move again and you can command a new target position.

INFORMATION

With the **CTO** (p. 153) and **TRO** (p. 240) commands, you can program the digital output lines of the C-884 so that they are activated in the case of motion errors. The programmed output lines remain activated until the error code is reset to 0. For details, see "Setting Up "Motion Error" Trigger Mode" (p. 96).

7.2 Trajectories for Motion Paths

7.2.1 Operating Principle of the Trajectory Buffer

In closed-loop operation, the C-884 can process externally calculated one- or two-dimensional motion paths (e.g., circles, sine curves) as trajectories (p. 2).

The individual target positions of the motion path must be loaded into the trajectory buffer of the C-884 as trajectory points. During the execution of the trajectory, the buffer outputs the points with a fixed chronological interval. The points are output in the order they were loaded to the trajectory buffer (FIFO principle: **F**irst **I**n **F**irst **O**ut).

The content of a trajectory buffer is only present in the volatile memory of the C-884 and cannot be saved to the permanent memory.

The trajectories are permanently assigned to the axes of the C-884: Trajectory 1 to axis 1, trajectory 2 to axis 2 etc.

7.2.2 Commands and Parameters for Trajectories

The trajectory buffer of the C-884 can be configured via the following parameters:

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

The following commands are available for trajectories:

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

7.2.3 Working with Trajectories

NOTICE



Execution of trajectories!

The C-884 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

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

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

- Therefore, pay attention to the following when working with trajectories:
 - The path that is specified by the trajectory points must be continuously differentiable at least twice.
 - During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
 - When following the trajectory, an abrupt stop may **not** damage the load on the positioner.
 - To generate the trajectory points and continuously transfer them to the C-884 during the trajectory execution, it is recommended to use a suitable program.

INFORMATION

For working with trajectories, it is recommended to use the **Trajectory Assistant** in PIMikroMove (call via the menu of the C-884). This allows you to define and execute trajectories conveniently.

The timing for trajectories is set with the TGT (p. 236) command.

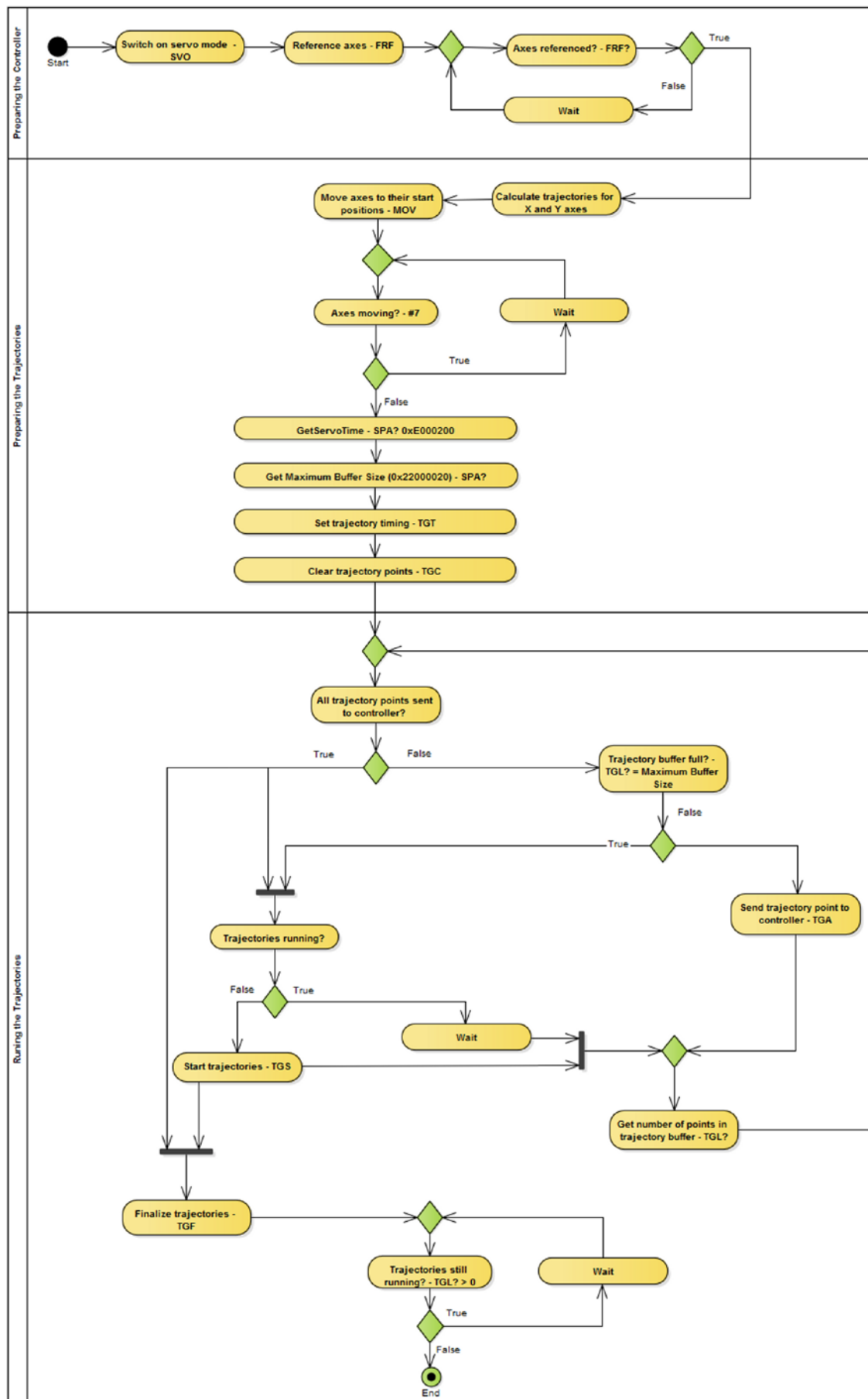
Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with the TGA (p. 231) command. The maximum number of points in the trajectory buffer is determined by the **Maximum Buffer Size** parameter (0x22000020).

The TGS (p. 235) command starts the execution of a trajectory. During the execution of a trajectory, the buffer must be refilled fast enough.

The TGF (p. 234) command properly completes the execution of a trajectory.

If the execution of a trajectory is cancelled after an error or stopped with **STP**, **#24**, or **HLT**, the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading or executing a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 234), deletion with TGC (p. 233)).

The following diagram shows an example of the sequence of a trajectory execution. Corresponding example programs are found on the PC after the PC software for the C-884 has been installed.



7.3 Data Recorder

7.3.1 Setting up the Data Recorder

The C-884 contains a real-time data recorder. The data recorder can record different variables for the axes (e.g., current position).

The recorded data is temporarily stored in 8 data recorder tables with 8192 points each. Each data recorder table contains the data of one data source.

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

INFORMATION

The following settings of the data recorder can only be changed in the volatile memory of the C-884:

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

After the C-884 has been switched on or rebooted, factory settings will be active unless a configuration takes place with a startup macro.

Reading general information from the data recorder

- Send the `HDR?` command (p. 173).

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

Configuring data to be recorded

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

- Send the `DRC?` command (p. 164) to read out the current configuration. Data recorder tables with the record option 0 are deactivated, i.e., nothing is recorded. By default, the data recorder tables of the C-884 record the following:
 - Data recorder table 1: current position of axis 1
 - Data recorder table 2: current position of axis 2
 - Data recorder table 3: current position of axis 3
 - Data recorder table 4: current position of axis 4
 - Data recorder table 5: commanded position of axis 1
 - Data recorder table 6: commanded position of axis 2
 - Data recorder table 7: commanded position of axis 3
 - Data recorder table 8: commanded position of axis 4

- Configure the data recorder with the **DRC** command (p. 163).

INFORMATION

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

Configuring the recording trigger

You can specify how the recording is to be triggered.

- Get the current trigger option with **DRT?** (p. 168).
- Change the trigger option with the **DRT** command (p. 166). The trigger option applies to all data recorder tables whose record option is not set to 0.

Setting the record table rate

- Send the **RTR?** command (p. 218) to read out the record table rate of the data recorder.

The parameter indicates the number of servo cycles required for recording each data point. The default value is 10 servo cycles. The servo cycle time of the C-884 is 100 µs.

- Change the record table rate with the **RTR** command. (p. 217)

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

Configuring data processing

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

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

Parameters	Description and Possible Values
Recorded Points Per Trigger 0x16000001	Number of data points to be recorded per trigger. 0 = Unlimited number of points (default setting) $n = n$ points; n is a whole-number value, lowest possible value is 1 The behavior of the C-884 when the data recorder tables are full depends on the value of the Data Recorder Buffer Mode parameter.
Clearing Of RecTable On Trigger 0x16000002	Write mode during the recording Determines how the data points are written to the data recorder tables when the recording is started by a trigger. 0 = Recorded points are added to the already present contents of the data recorder tables (default setting) 1 = The trigger deletes the data recorder tables, i. e., the recording always starts with the first point of the data recorder tables.
Data Recorder Buffer	Behavior with full data recorder tables

Parameters	Description and Possible Values
Mode 0x16000003	0 = Recording ends (default setting) 1 = Recording is continued with the first point of the data recorder table and overwrites the existing contents. The value of the Data Recorder Buffer Overflow parameter is increased by 1.
Data Recorder Buffer Overflow 0x16000004	Buffer overflow counter of the data recorder Counts how often the recording starts again with the first point of the data tables when the Data Recorder Buffer Mode parameter has the value 1. Reading out the recorded data with DRR? resets the value of the buffer overflow counter to zero. The parameter is write-protected.

7.3.2 Starting the Recording

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

Regardless of the trigger option set, the data recording is always triggered when a step response measurement is started with **STE** (p. 227).

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.

7.3.3 Reading Recorded Data

INFORMATION

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

- Read the last recorded data with the **DRR?** command (p. 165).
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. 164).

7.4 Digital Output Signals

The digital outputs of the C-884 are available at the **I/O** socket (p. 309).

- Get the number of the output lines available on the C-884 with the **TIO?** command (p. 238).

External devices can be triggered via the digital outputs of the C-884. Potential applications:

- Linking the trigger output to the motion of the axis. Details and examples can be found in this section.
- Direct switching of output lines, e. g., in macros. Details and examples of macros can be found in "Controller Macros" (p. 117).

7.4.1 Commands for Digital Outputs

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

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

Up to four settings for triggering a device via the digital outputs can be transferred to the C-884 in one command line (CTO + max. 32 characters) with the following command:

CTO {<TrigOutID> <CTOPam> <Value>}

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

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

<Value>	Trigger mode	Short description
0 (default)	Position Distance	Once the axis has moved a specified distance, a trigger pulse is output (p. 93). Optionally, start and stop values can be defined to limit triggering to one position range and one particular direction of motion (negative or positive).
2	On Target	The on-target state of the axis selected is output at the selected trigger output (p. 95).
5	Motion Error	The selected digital output line becomes active when a motion error occurs (p. 96). The line stays active until the error code is reset to 0 (by a query with ERR?).
6	In Motion	The selected digital output line is active as long as the selected axis is in motion (p. 96).

<Value>	Trigger mode	Short description
7	Position+Offset	The first trigger pulse is output when the axis has reached a specified position. The next trigger pulses are output respectively when the axis position equals the sum of the last valid trigger position and a specified distance. The trigger output is stopped when a stop value is reached. The polarity sign of the distance value determines the direction of motion in which trigger pulses are to be output. For details, see "Setting Up "Position + Offset" Trigger Mode" (p. 97).
8	Single Position	The selected digital output line is active when the axis position has reached or exceeded a given position (p. 98).
9	Hardware Trigger	Corresponds to the <i>Position+Offset</i> trigger mode, but is executed by the FPGA of the C-884 (shorter processing time). For details, see "Setting Up "Position + Offset" Trigger Mode" (p. 97).

In addition, the polarity (active high / active low) of the signal, that is output at the digital output, can be set (p. 100).

INFORMATION

The settings for the configuration of the digital output lines can only be modified in the volatile memory of the C-884. After the C-884 has been switched on or rebooted, factory settings are active, provided a configuration has not already been carried out with a startup macro.

7.4.2 Setting Up "Position Distance" Trigger Mode

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

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

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

Example:

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

➤ Send:

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

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

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

INFORMATION

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

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

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

Example

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

➤ Send:

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

```
CTO 1 8 0.0002
CTO 1 9 0.00055
TRO 1 1
```

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

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

Example:

➤ Send:

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

7.4.3 Setting Up "On Target" Trigger Mode

In the *On Target* trigger mode, the on-target state of the axis selected (p. 31) is output at the selected trigger output.

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

Example:

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

➤ Send:

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

7.4.4 Setting Up "Motion Error" Trigger Mode

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

INFORMATION

A motion error occurs when the current position differs too much from the commanded position during the motion.

For further information, see "Motion Error".

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

7.4.5 Setting Up "In Motion" Trigger Mode

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

The motion state can also be read with the `#5` (p. 144), `#4` (p. 142), and `SRG?` (p. 226) commands.

INFORMATION

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

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

Example:

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

➤ Send:

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

7.4.6 Setting Up "Position + Offset" Trigger Mode

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

The pulse width is one servo cycle.

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

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

Example 1:

The first trigger pulse is to be output on digital output line 1 if the absolute position of axis 1 is 1.5 mm. A pulse should then be output on this line every time axis 1 has covered a distance of 0.1 µm in the positive direction. The last trigger pulse is to be output if the absolute axis position is 2.5 mm.

➤ Send:

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

Example 2:

The first trigger pulse is to be output on digital output line 2 if the absolute position of axis B is 0.4 mm. A pulse should then be output on this line every time axis B has covered a distance of

1 μm in the negative direction. The last trigger pulse is to be output if the absolute axis position is 0.1 mm.

➤ Send:

```
CTO 2 2 B
CTO 2 3 7
CTO 2 1 -0.001
CTO 2 10 0.4
CTO 2 9 0.1
```

INFORMATION

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

Maximum velocity = distance * 20 kHz / 2

where 20 kHz is the servo cycle frequency of the C-884.

7.4.7 Setting Up "Single Position" Trigger Mode

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

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

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

Example:

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

➤ Send:

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


7.4.8 Setting up the "HardwareTrigger" Trigger Mode

The *HardwareTrigger* mode basically corresponds to the *Position+Offset* trigger mode (p. 97) but is executed by the FPGA circuit of the C-884 (shorter processing time).

INFORMATION

The *HardwareTrigger* mode can only be used for the C-884 when the encoder of the connected mechanics supplies A/B signals.

The *HardwareTrigger* mode does not function with the C-884 in conjunction with other signal types.

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

$$\text{Pulse width} = n * 33.3 \text{ ns}$$

For the *HardwareTrigger* trigger mode, there is a fixed assignment of the axes to the digital output lines: Axis 1 to line 1, axis 2 to line 2, axis 3 to line 3, axis 4 to line 4.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
 - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
 - Send CTO <TrigOutID> 3 9, where 9 indicates the HardwareTrigger trigger mode.
 - Send CTO <TrigOutID> 1 S, where S indicates the distance.
 - Send CTO <TrigOutID> 10 TriPos, where TriPos indicates the position for the output of the first trigger pulse.
 - Send CTO <TrigOutID> 9 Stop, where Stop indicates the stop value.
 - Send CTO <TrigOutID> 11 n, where n indicates the factor for calculating the pulse width.
2. If you want to enable the conditions for trigger output, send TRO <TrigOutID> 1.

Example 1:

The first trigger pulse is to be output on digital output line 1 if the absolute position of axis 1 is 1.5 mm. A pulse should then be output on this line every time axis 1 has covered a distance of 0.5 µm in the positive direction. The last trigger pulse is to be output if the absolute axis position is 2.5 mm. The pulse width should be approximately 0.8 µs.

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 9
```

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

Example 2:

The first trigger pulse is to be output on digital output line 1 if the absolute position of axis B is 0.4 mm. A pulse should then be output on this line every time axis B has covered a distance of 1 µm in the negative direction. The last trigger pulse is to be output if the absolute axis position is 0.1 mm. The pulse width should be approximately 0.166 µs.

➤ Send:

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

INFORMATION

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

Maximum velocity = distance * 20 kHz / 2

where 20 kHz is the servo cycle frequency of the C-884.

7.4.9 Setting Signal Polarity

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

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

Example:

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

➤ Send:

```
CTO 1 7 0
```

7.5 Digital Input Signals

The digital inputs of the C-884 are available on the **I/O** (p. 309) socket.

- Get the number of the input lines available on the C-884 with the **TIO?** command (p. 238).
- Get the state of the input lines with the **DIO?** command (p. 162).

Potential applications:

- Use in macros (p. 103). Details and examples of macros can be found in "Controller Macros" (p. 117).
- Use as switch signals (p. 103)

7.5.1 Commands and Parameters for Digital Inputs

Commands

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

Command	Syntax	Function
CPY	CPY <Variable> <CMD?>	Copies the state of a digital input line to a variable when used in conjunction with the DIO? query command. Use in macros to set local variables (p. 135).
DIO?	DIO? [{<DIOID>}]	Gets the state of the digital input lines.
FED	FED {<AxisID> <EdgeID> <Param>}	Starts a move to a signal edge. The signal source can be a digital input line.
FRF	FRF [{<AxisID>}]	Starts a reference move to the reference point switch. A digital input line can be used as the source of the reference switch signal instead of the reference point switch.
JRC	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the macro run pointer depending on the state of a digital input line when used in conjunction with the DIO? query command.
MEX	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops running of the macro depending on the state of a digital input line when used in conjunction with the DIO? query command.

Command	Syntax	Function
WAC	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until a digital input line reaches a certain state when used in conjunction with the DIO? query command.

Parameters

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

Parameters	Description and Possible Values
Source Of Reference Signal 0x5C	Specifies the source of the reference signal for the FRF and FED commands: 0 = reference point switch 1 = Digital input 1 2 = Digital input 2 3 = Digital input 3 4 = Digital input 4
Source Of Negative Limit Signal 0x5D	Specifies the source(s) of the negative limit switch signal for the FRF (with parameter 0x70 = 5) and FED commands via a bitmask: 0 = Negative limit switch (default setting) 1 = Digital input 1 (bit 0) 2 = Digital input 2 (bit 1) 4 = Digital input 3 (bit 2) 8 = Digital input 4 (bit 3)
Source Of Positive Limit Signal 0x5E	Specifies the source(s) of the positive limit switch signal for the FRF (with parameter 0x70 = 6) and FED commands via a bitmask: 0 = Positive limit switch (default setting) 1 = Digital input 1 (bit 0) 2 = Digital input 2 (bit 1) 4 = Digital input 3 (bit 2) 8 = Digital input 4 (bit 3)
Invert Digital Input Used For Negative Limit 0x5F	Inverts the polarity of the digital inputs, which are used for the source of the negative limit switch signal, via a bitmask: 0 = No digital input inverted (default setting). 1 = Digital input 1 inverted (bit 0) 2 = Digital input 2 inverted (bit 1) 4 = Digital input 3 inverted (bit 2) 8 = Digital input 4 inverted (bit 3)

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

7.5.2 Using Digital Input Signals in Macros

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

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

Further information and examples can be found in "Controller Macros" (p. 117).

7.5.3 Using Digital Input Signals as Switch Signals

The digital inputs on the **I/O** socket can be used as the source of reference point and limit switch signals (e.g., for reference moves (p. 38)) for an axis.

Using digital input as reference signal

INFORMATION

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

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

INFORMATION

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

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

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

Using digital inputs as source of the limit switch signals

INFORMATION

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

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

INFORMATION

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

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

INFORMATION

The **Has No Limit Switches?** parameter (ID 0x32) determines whether the C-884 evaluates the signals from the internal limit switches of the positioner. This parameter has no bearing on the use of digital input lines as the source of the limit switch signal.

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

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

Example:

Digital input lines 1, 3, and 4 are to be used for axis 1 as the sources of the positive limit switch signal. In addition, the signal polarity of lines 1 and 3 is to be inverted for axis 1. All adaptations are made in the volatile memory of the C-884 only.

- Send:
 SPA 1 0x5E 13, to select lines 1, 3, and 4.
 SPA 1 0x60 5, to invert the signal polarity of lines 1 and 3.

7.6 Analog Input Signals

The analog inputs of the C-884 are available on the **I/O** socket (p. 309).

- Get the number of the analog input lines available on the C-884 with the **TAC?** command (p. 230).
- Query the voltage on the analog inputs with the **TAV?** command (p. 230).
- Use the data recorder (p. 89) to record the analog input signals.

Potential applications:

- Use in macros (p. 106): Details and examples of macros are found in "Controller Macros" (p. 117).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

7.6.1 Commands for Analog Inputs

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

Command	Syntax	Function
CPY	CPY <Variable> <CMD?>	Copies the voltage value of an analog input line to a variable when used in combination with the TAV? query command. Use in macros to set local variables (p. 135).
DRC	DRC {<RecTableID> <Source> <RecOption>}	Configures the data recorder. Analog input values can be recorded using record option 81.
JRC	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the pointer when running the macro depending on the voltage at an analog input line when used in conjunction with the TAV? query command.
MEX	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops running of the macro depending on the voltage at an analog input line when used in conjunction with the TAV? query command.
TAC?	TAC?	Get the number of installed analog lines.
TAV?	TAV? [<AnalogInputID>]	Get voltage at analog input.
WAC	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until an analog input line reaches a certain voltage when used in conjunction with the TAV? query command.

7.6.2 Using Analog Input Signals in Macros

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

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

Further information and examples can be found in "Controller Macros" (p. 117).

7.7 Control with a Human Interface Device

7.7.1 Functionality of HID Control

HID Control of Motion Parameters

Axes of human interface devices can control the following motion variables of the positioner axes connected to the C-884:

- Velocity
- Maximum velocity

The relationship between the displacement of the axis of the human interface device and the motion variable of the positioner axis is created by the C-884 using a lookup table. The 256 values in the lookup table are factors that are applied to the motion parameter to be controlled during HID control. For details, see the description of the HIA command (p. 176). The values range from -1.0000 to 1.0000.

The firmware of the controller provides two predefined lookup table types to choose from (linear and parabolic) and allows four user-specific lookup tables to be filled with custom values. For details, see the descriptions of the HDT (p. 174) and HIT (p. 188) commands.

During HID control, the target position of the controlled axis of the C-884 is set to the soft limit that is given by the parameter 0x15 or 0x30. Details on the parameters can be found in "travel range and Soft Limits" (p. 34). When HID control is disabled, the target position is set to the current position of the controlled axis.

INFORMATION

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

Programming HID control

Buttons and LEDs (i.e., output units) of human interface devices can be used to program HID control, for example, in controller macros (p. 117).

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

7.7.2 Commands and Parameters for Human Interface Devices

Commands

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

Command	Syntax	Function
HDT	HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}	Assigns a lookup table to the axis of an HID device. The assignment can be saved in the nonvolatile memory with WPA.
HDT?	HDT? [{<HIDDeviceID> <HIDDeviceAxis>}]	Gets the current assignment of lookup tables to the axes of HID devices.
HIA	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}	Configures the control of axes of the C-884 by axes of HID devices ("HID control"). The configuration can be saved in the nonvolatile memory with WPA.
HIA?	HIA? [{<AxisID> <MotionParam>}]	Gets the current configuration of the HID control.
HIB?	HIB? [{<HIDDeviceID> <HIDDeviceButton>}]	Gets the current state of the buttons of HID devices.
HIE?	HIE? [{<HIDDeviceID> <HIDDeviceAxis>}]	Gets the current displacement of the axes of HID devices.
HIL	HIL {<HIDDeviceID> <HIDDeviceLED> <HIDLEDState>}	Sets the current state of the output units or characteristics ("LEDs") of HID devices.
HIL?	HIL? [{<HIDDeviceID> <HIDDeviceLED>}]	Gets the current state of the output units or characteristics ("LEDs") of HID devices.
HIN	HIN {<AxisID> <HIDControlState>}	Enables or disables the HID control for the axes of the C-884.
HIN?	HIN? [{<AxisID>}]	Gets the activation state of the HID control.
HIS	HIS {<HIDDeviceID> <HIDItemID> <HIDPropID> <HIDPropValue>}	Configures the given HID device. For the C-884, the functionality of HIS corresponds to that of the HIL command.
HIS?	HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]	Gets the characteristics of the operating elements of HID devices.

Command	Syntax	Function
HIT	HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fills lookup tables with values. The table contents can be saved in the nonvolatile memory with WPA.
HIT?	HIT? [<StartPoint> [<NumberOfPoints> [<HIDTableID>]]]	Gets the values of the points in the lookup tables.

Parameter

The following parameters are available for the HID control:

Parameter	Description and possible values
<i>Invert Direction Of Motion For Joystick-Controlled Axis?</i> 0x61	Specifies the direction of motion for HID-controlled axes of the C-884. 0 = direction of motion not inverted (default setting) 1 = direction of motion inverted
<i>Closed-Loop Velocity For HI Control (Phys. Unit/s)</i> 0x74	Maximum velocity during the HID control Is limited by parameter 0xA. If parameter 0x74 has the value zero, the value of the parameter 0x49 is used during the HID control.
<i>Closed-Loop Acceleration For HI Control (Phys. Unit/s²)</i> 0x75	Maximum acceleration during the HID control Is limited by parameter 0x4A.
<i>Closed-Loop Deceleration For HI Control (Phys. Unit/s²)</i> 0x76	Maximum deceleration during the HID control Is limited by parameter 0x4B.

7.7.3 Testing a Human Interface Device

After a human interface device has been connected to the C-884, it is recommended to test the operating elements of the human interface device in PIMikroMove.

INFORMATION

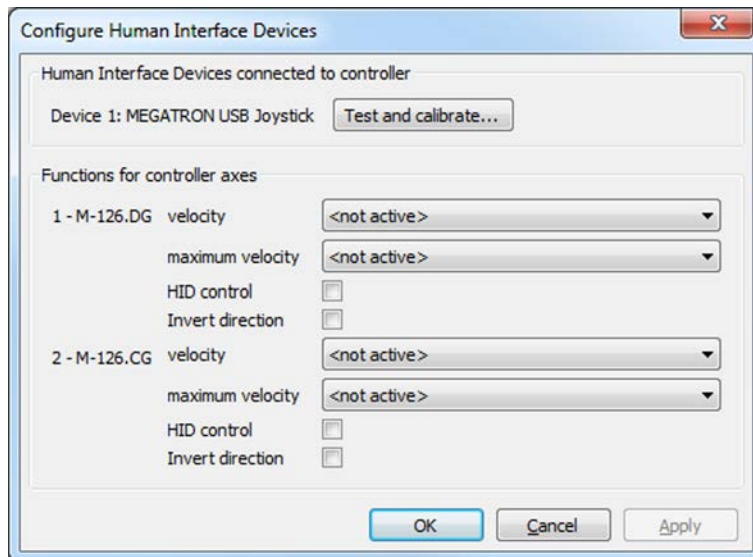
A positioner does not have to be connected to the C-884 to test operating elements of human interface devices in PIMikroMove.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ PIMikroMove is installed on the PC (p. 51).
- ✓ You have established communication with PIMikroMove between the C-884 and the PC (p. 59).
- ✓ You have connected the C-884 to the human interface device (p. 48).

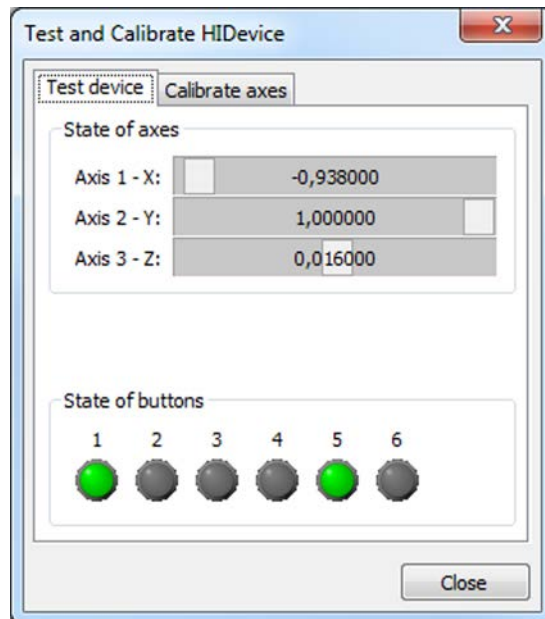
Testing the human interface device in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **C-884 > Configure controller HIDevice(s)...** menu item.



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

- Enter various values in the fields in the **State of LEDs** area on the **Test device** tab and observe the behavior of the corresponding operating elements on the human interface device.



In this example, a joystick with 3 axes and 6 buttons is connected to a C-884. Current status in the figure: The X axis of the joystick is displaced in the negative direction, the Y axis is displaced in the positive direction, the Z axis is displaced in the positive direction, and buttons 1 and 5 are pressed.

7.7.4 Calibrating the Axes of Human Interface Devices

If the response behavior of the axes of the human interface device does not meet your requirements, the axes can be calibrated in PIMikroMove.

INFORMATION

When the axes of human interface devices are calibrated in PIMikroMove, the lookup table to be used is selected and is filled with custom values if necessary. To do this, a positioner does not need to be connected to the C-884.

INFORMATION

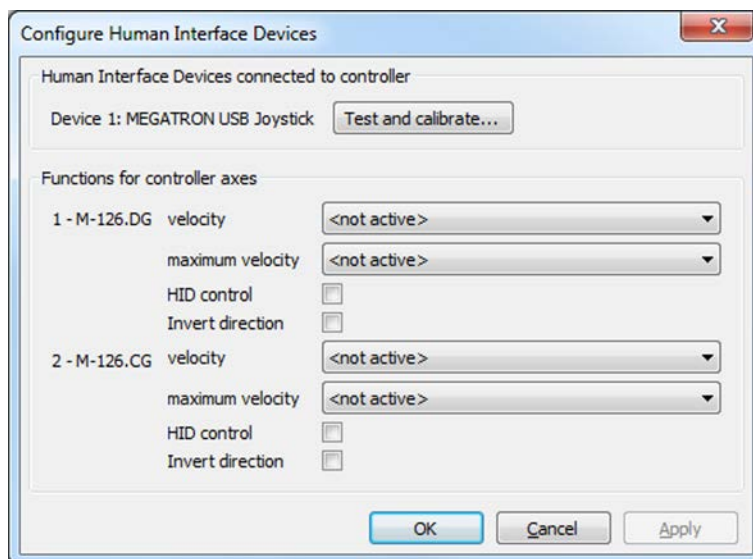
The parabolic lookup table allows for greater sensitivity when moving slowly.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ PIMikroMove is installed on the PC (p. 51).
- ✓ You have established communication with PIMikroMove between the C-884 and the PC (p. 59).
- ✓ You have connected the C-884 to the human interface device (p. 48).

Calibrating the axis of a human interface device in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **C-884 > Configure controller HIDevice(s)...** menu item.



2. Open the **Test and Calibrate HIDevice** window for the human interface device whose axes are to be calibrated by clicking the **Test and calibrate...** button.
3. In the **Test and Calibrate HIDevice** window, select the **Calibrate axes** tab.

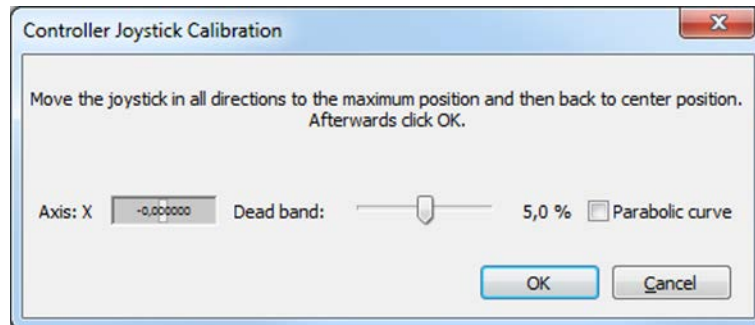
4. Select the lookup table to be used for the axes of the human interface device in the selection fields on the **Calibrate axes** tab. Lookup tables with the designation **User Table** are intended to be filled with custom values.



In this example, a joystick with 3 axes is connected to the C-884 as a human interface device. A user-defined lookup table was selected for axis 1. For axes 2 and 3, the respective predefined parabolic lookup table is selected.

5. If you have selected a user-defined lookup table and want to fill the table with values:
 - a) Click the corresponding **Calibrate...** button to open the **Controller Joystick Calibration** window. Here, "Joystick" refers to the axis of the human interface device.
 - b) Move the axis of the human interface device to all extreme positions. The custom lookup table values are determined in this way.
 - c) Let go of the axis.
 - d) If you want to change the neutral area of the axis (i.e., the area around the center position of the axis where no change in the controlled motion variable is triggered), set the **Dead band** slider in the **Controller Joystick Calibration** window correspondingly.
 - e) If the values in the user-defined lookup table are to describe a parabolic waveform, click the **Parabolic curve** checkbox in the **Controller Joystick Calibration** window.

- f) Click **OK** in the **Controller Joystick Calibration** window to write the lookup table values to the volatile memory of the C-884. You can watch the writing process in a separate window.



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

6. If you want to save the assignment of the lookup tables to the axes of the human interface device and the contents of user-defined lookup tables in the nonvolatile memory of the C-884:
 - a) Close the **Test and Calibrate HIDevice** window.
 - b) If necessary, adapt the settings in the **Configure Human Interface Devices** window to your application; see "Setting Up and Enabling HID control" (p. 113).
 - c) If necessary, click the **Apply** button to enable the settings in the **Configure Human Interface Devices** window.
 - d) Close the **Configure Human Interface Devices** window.
 - e) Follow the instructions in "Saving the Configuration of HID Control Permanently" (p. 115).

7.7.5 Setting Up and Enabling HID Control

Before you enable HID control, the following steps are recommended:

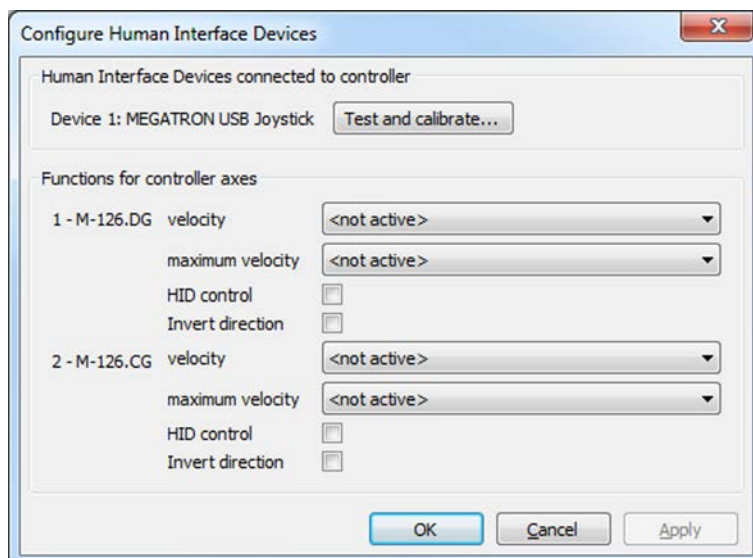
- Testing the human interface device (p. 108)
- Calibrating the axes of the human interface device (p. 110)

Requirements

- ✓ You have carried out a successful reference move for each axis of the C-884 with PIMikroMove; see "Starting Motion" (p. 68).
- ✓ You have connected the C-884 to the human interface device (p. 48).
- ✓ All devices are still ready for operation.

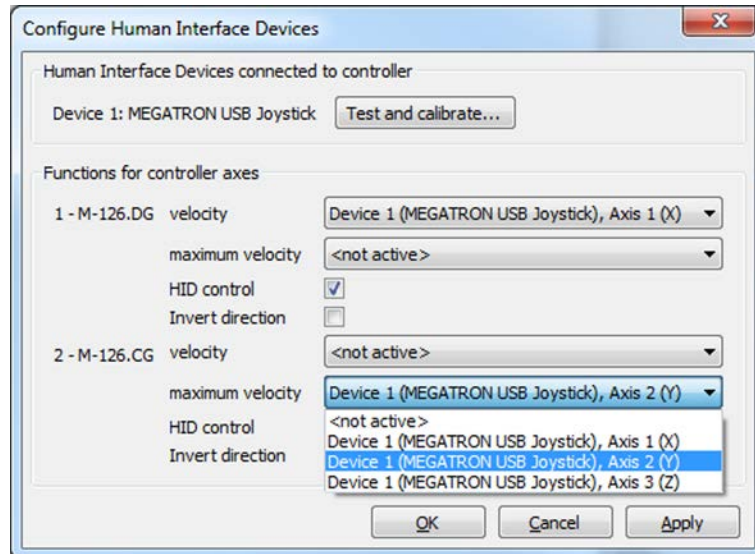
Setting up and enabling HID control in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **C-884 > Configure controller HIDevice(s)...** menu item.



2. Set up HID control in the **Configure Human Interface Devices** window. Proceed as follows for each of the axes of the C-884 that is listed under **Functions for controller axes**:
 - a) If you want to control the velocity of the axis of the C-884 by an axis of a human interface device, select the corresponding axis of the human interface device in the **Velocity** field.
 - b) If you want to control the maximum velocity of the axis of the C-884 by an axis of a human interface device, select the corresponding axis of the human interface device in the **Max. Velocity** field.
 - c) Enable HID control for the axis of the C-884 by clicking in the **HID Control** check box.

- d) If the direction of motion of the controlled axis of the C-884 is to be inverted during HID control, click in the **Invert direction** check box.



3. Send the settings for setting up HID control to the C-884 by clicking the **OK** button. The **Configure Human Interface Devices** window closes.
4. In PIMikroMove, make sure that the servo mode is switched on for the controlled axes of the C-884 (e.g., by clicking in the **Servo** check box on the **Axes** tab in the main window of PIMikroMove).
The axes of the C-884 can now be controlled by the axes of human interface devices according to the settings made in step 2.
5. If you want to save the assignment of the axes of human interface devices to the controlled motion variables in the nonvolatile memory of the C-884:
 - Follow the instructions in "Saving the Configuration of HID Control Permanently" (p. 115).

7.7.6 Saving the Configuration of HID Control Permanently

The following settings for the configuration of HID control can be saved in the nonvolatile memory of the C-884:

- Assignment of lookup tables to the axes of the human interface device; see "Calibrating the Axes of Human Interface Devices" (p. 110)
- For contents of user-defined lookup tables, see "Calibrating the Axes of Human Interface Devices" (p. 110)
- Assigning the axes of human interface devices to the motion variables to be controlled for the axis of the C-884, see "Setting up and Enabling HID Control" (p. 113)

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

INFORMATION

The values in the nonvolatile memory are loaded to the volatile memory when switching on or rebooting the C-884 and take effect immediately.

Requirements

- ✓ You have read and understood the general notes on startup (p. 57).
- ✓ PIMikroMove is installed on the PC (p. 51).
- ✓ You have read and understood the PIMikroMove manual. The manual is on the product CD.
- ✓ You have established communication between the C-884 and the PC with PIMikroMove (p. 59).

Saving the configuration of HID control permanently in PIMikroMove

If you want to write the current settings for the configuration of HID control to the nonvolatile memory of the C-884:

1. In the main window of PIMikroMove, select the **C-884 > Save parameters to nonvolatile memory** menu item. The **Save Parameters to Non-Volatile Memory** dialog opens.
2. In the selection field of the **Save Parameters to Non-Volatile Memory** dialog, enter either the password *HID* or select the entry *Settings of HDT, HIA, HIT (HID)*.
3. Click **OK** to save and close the dialog.

INFORMATION

The settings for the configuration of HID control are also written to the nonvolatile memory of the C-884 if you select the *All Parameters, Settings of HDT, HIA, HIT (100)* or enter the password *100*. However, the entry or the password *100* also saves the current values of all parameters of the C-884, see the description for the WPA (p. 247) command and "Adapting Settings" (p. 271).

7.8 Controller Macros

7.8.1 Overview: Macro Functionality and Example Macros

The C-884 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-884 is switched on or rebooted.
- Processing a macro and stopping macro execution can be linked to conditions. In this way, loops can be realized as well.
- Macros can call up themselves or other macros.
- Variables (p. 135) can be set for the macro and in the macro itself and used in different operations.
- Input signals can be evaluated for conditions and variables.

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

- Moving an axis back and forth (p. 121)
- Moving an axis with a variable travel back and forth (p. 123)
- Implementing multiple calls of a macro via a loop (p. 124)
- Preparing an axis via startup macro for closed-loop operation (p. 125)
- Synchronization of two controllers (p. 126)
- Stopping motion by pushbutton (p. 127)
- HID control with storage of positions (p. 128)

7.8.2 Commands and Parameters for Macros

Commands

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

Command	Arguments	Function
ADD (p. 147)	<Variable> <FLOAT1> <FLOAT2>	Adds two values and saves the result to a variable (p. 135). Can only be used for local variables in macros.
CPY (p. 151)	<Variable> <CMD?>	Copies a command response to a variable (p. 135). Can only be used for local variables in macros.
DEL (p. 159)	<uint>	Can only be used in macros. Delays <uint> milliseconds.

Command	Arguments	Function
JRC (p. 201)	<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. 203)	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.
	DEF <macroname>	Defines the given macro as the startup macro.
	DEF?	Gets the startup macro.
	DEL <macroname>	Deletes the given macro.
	END	Ends the macro recording.
	ERR?	Reports the last error that occurred while the macro was running.
	NSTART <macroname> <uint> [<String1> [<String2> [<String3> [<String4>]]]]	Starts the given macro n times in succession (n = number of executions). The values of local variables can be set for the macro with <String1>, <String2>, <String3> and <String4>.
	START <macroname> [<String1> [<String2> [<String3> [<String4>]]]]	Executes the specified macro. The values of local variables can be set for the macro with <String1>, <String2>, <String3> and <String4>.
MAC? (p. 205)	[<macroname>]	Lists all macros or the content of a given macro.
MEX (p. 207)	<CMD?> <OP> <Value>	Can only be used in macros. Stops the macro execution depending on a condition.
RMC? (p. 215)		Lists macros which are currently running.
VAR (p. 242)	<Variable> <String>	Sets a variable (p. 135) to a certain value or deletes it. Can only be used for local variables in macros.
VAR? (p. 243)	[<Variable>]	Gets variable values.
WAC (p. 245)	<CMD?> <OP> <Value>	Can only be used in macros. Waits until a condition is met.
#8 (p. 144)		Tests if a macro is running on the controller.

Parameters

The following parameter is available for working with macros:

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

7.8.3 Working with Macros

Work with macros comprises the following:

- Recording of macros (p. 119)
- Starting macro execution (p. 122)
- Stopping macro execution (p. 124)
- Setting up a startup macro (p. 125)
- Deleting of macros (p. 126)

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-884 can save up to 32 macros simultaneously. A maximum of 5 nesting levels are possible in macros.

INFORMATION

Basically all GCS commands (p. 133) can be included in a macro. Exceptions:

- **RBT** for rebooting the C-884
- **MAC BEG** and **MAC END** for macro recording
- **MAC DEL** for deleting a macro

Query commands can be used in macros in conjunction with the **CPY**, **JRC**, **MEX**, and **WAC**

commands. Otherwise they have no effect, since macros do not send any responses to interfaces.

INFORMATION

To make the use of macros more flexible, you can use local and global variables in macros. For further information, see "Variables" (p. 135).

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. 135) in macros to make macros more flexible, and give the corresponding variable values when starting macro execution.
- Contact our customer service department (p. 301) if the C-884 shows unexpected behavior.

INFORMATION

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

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

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

The macro has been stored in the nonvolatile memory of the C-884.

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-884 by sending the `MAC?` command.
- Get the contents of the *macroname* macro by sending the `MAC? macroname` command.

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

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

Example: Moving an axis back and forth

INFORMATION

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

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

- Record the macros by sending:

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

Starting macro execution

INFORMATION

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

INFORMATION

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

INFORMATION

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

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

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

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

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

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

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

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

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

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

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

```
VAR LEFT 5
```

```
VAR RIGHT 15
```

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

- Create the global variables again each time the C-884 is switched on or rebooted, since they are only written to the volatile memory of the C-884.

2. Record the MOVLR macro by sending:

```
MAC BEG movlr
```

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

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

```
MAC END
```

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

3. Record the MOVWAI macro by sending:

```
MAC BEG movwai
```

```
MOV 1 $1
```

```
WAC ONT? 1 = 1
```

```
MAC END
```

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

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

```
MAC NSTART movlr 5
```

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

Example: Implementing multiple calls of a macro via a loop**INFORMATION**

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

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

- Record the LOOPDION macro by sending:

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

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

Stopping macro execution**INFORMATION**

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

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

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

Setting up a startup macro

Any macro can be defined as the startup macro. The startup macro is executed each time the C-884 is switched on or rebooted.

INFORMATION

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

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

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

Example: Preparing an axis via a startup macro for closed-loop operation

INFORMATION

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

The STARTCL macro switches the HID control off and the servo mode on for axis 1 and starts a reference move. As STARTCL is defined as the startup macro, axis 1 is ready for closed-loop operation immediately after switch-on.

- Send:


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

INFORMATION

When using this macro, the parameter settings of the C-884 should be adapted in the nonvolatile memory to the connected positioner. Alternatively, the parameter settings can also be configured in the volatile memory via the startup macro. For further information, see "Adapting Settings" (p. 271).

Deleting a macro**INFORMATION**

A macro cannot be deleted while it is running.

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

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

7.8.4 Macro Example: Synchronization of Two Controllers**INFORMATION**

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

Action	Command	Result
Connect the digital output line 1 on the I/O socket of the master controller to digital input line 1 on the I/O socket of the slave controller.	- Use a suitable cable. Pin assignment see "I/O" (p. 309).	The digital output signal of the master controller can be used as the trigger for the motion of the axis connected to the slave controller.
Set up the motion on the master controller and on the slave controller.	SVO 1 1 FRF 1 1 VEL 1 0 MOV 1 5.5	For both controllers: The servo mode is switched on and the axis has executed a reference move – here to the reference point switch. The velocity is set to zero. The axis does not move for now as a result, even though the motion command for the move to absolute position 5.5 has already been sent.

Action	Command	Result
Record the MASTER macro on the master controller.	MAC BEG master DIO 1 1 VEL 1 100 MAC END	The macro has the following tasks: <ul style="list-style-type: none"> Switch the digital output line 1 of the master controller to high state to trigger the slave controller Set velocity to 100 to start the motion
Record the SLAVE macro on the slave controller.	MAC BEG slave WAC DIO? 1 = 1 VEL 1 100 MAC END	The macro has the following tasks: <ul style="list-style-type: none"> Set condition: The macro continues only if digital input line 1 has the high state (i.e., if the master controller outputs the trigger signal). Set velocity to 100 to start the motion
Start the SLAVE macro on the slave controller.	MAC START slave	The axis on the slave controller is still not moving because the condition for further macro execution has not yet been met.
Start the MASTER macro on the master controller.	MAC START master	Both axes are moving because their velocity is now each different from zero. The motion occurs synchronously.

7.8.5 Macro Example: Stopping Motion by Pushbutton

INFORMATION

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

Action	Command	Description
Connect digital input line 1 on the I/O socket to an appropriate signal source.	- Pin assignment see "I/O" (p. 309).	The digital input signal can be used e. g. for a conditional jump of the macro execution pointer.
Record the HALT macro on the controller.	MAC BEG halt MVR 1 5 JRC 2 DIO? 1 = 1 JRC -1 ONT? 1 = 0 HLT 1 MAC END	The macro has the following tasks: <ul style="list-style-type: none"> Start a relative motion of axis 1 Set a condition: If digital input line 1 has the "high" state (e. g. by pushbutton), the macro execution pointer jumps two lines forward. The axis is stopped as a result. Otherwise macro execution is continued with the next line. Set a condition: As long as axis 1 has not yet reached the target position, the macro execution pointer jumps back one line. A loop is established as a result.

Action	Command	Description
Start the HALT macro on the controller.	MAC START halt	Axis 1 starts to move. It is stopped by switching digital input line 1 to the high state (e. g. by pushbutton). Regardless of whether the axis has reached the target position or was halted previously, the error code is set to 10 via the HLT command.
If error code 10 interferes: Record alternative HALTVAR macro which uses a variable. Details see "Variables" (p. 135).	MAC BEG haltvar MVR 1 5 JRC 2 DIO? 1 = 1 JRC -1 ONT? 1 = 0 CPY TARGET POS? 1 MOV 1 \${TARGET} VAR TARGET MAC END	The macro has the same tasks as the HALT macro. However, axis 1 is not stopped by pushbutton via the HLT command; instead the result of the POS? 1 query is copied to the TARGET variable. Then this variable is used as the target position for the MOV command. As a result, the axis stays right where it was. To clean up, TARGET is defined as empty with the VAR command which deletes the variable.
Start the HALTVAR macro on the controller.	MAC START haltvar	Axis 1 starts to move. It is stopped by switching digital input line 1 to the high state (e. g. by pushbutton). Error code 10 is not set because no halt or stop command is used.

7.8.6 Macro Example: HID Control with Storage of Positions

INFORMATION

In the following example, a gamepad is connected as a human interface device. This gamepad has a motor for the vibration function. The vibration frequency of the motor can be adjusted (available as "LED" in the corresponding commands). Furthermore, the gamepad has at least one axis and at least 5 buttons. The values can be completely different for other human interface devices.

Task

Axis 1 of the C-884 was renamed "X" with SAI. The velocity of axis X is to be controlled with axis 1 of human interface device 1 (e.g., gamepad). HID control is only to be enabled when button 5 of the human interface device is pressed at the same time. This state is to be signaled by the slow running of the motor (LED 3) in the human interface device. Buttons 1 to 4 of the human interface device also allow up to four positions to be saved in the controller (long press of the button) or approached by the axis (short press of the button). Fast running (vibration) of the human interface device motor is intended to signal that saving has taken place.

Approach

- Initialization

- Loop for querying the state of the buttons of the human interface device (counter variable required)
- Querying of the state of button 5 of the human interface device (enabling / disabling of HID control)
- Querying buttons 1 to 4 of the human interface device; Depending on the button state, the current position is transmitted and saved in the STORE<n> variable or the saved position is approached.

INFORMATION

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

Action	Command	Description
Connect a human interface device to the USB (type A) connection.	-	-
Record the STARTUP macro on the controller.	<code>MAC BEG startup</code>	Start macro recording
	<code>HIN X 0</code>	▪ Disable control by human interface devices for axis X
	<code>SVO X 1</code>	▪ Switch on servo mode for axis X: The servo mode must be switched on for axis X to be controlled by the axis of a human interface device.
	<code>FRF X</code>	▪ Start a reference move for axis X: The axis starts a reference move. After this, absolute axis positions can be commanded.
	<code>HIA X 3 1 1</code>	▪ Configure HID control: The velocity of axis X of the controller is to be controlled by the displacement of axis 1 of human interface device 1.
	<code>CPY STORE1 POS? X</code> <code>CPY STORE2 POS? X</code> <code>CPY STORE3 POS? X</code> <code>CPY STORE4 POS? X</code>	▪ Initialize global variables for storing the position of axis X
	<code>MAC START MAINLOOP</code>	▪ Start MAINLOOP macro for the main loop
	<code>MAC END</code>	End macro recording
Record the MAINLOOP macro on the controller.	<code>MAC BEG mainloop</code>	Start macro recording

Action	Command	Description
	MAC START TESTJOYB	▪ Start TESTJOYB macro for HID control
	VAR COUNTER 1	▪ Assign the value 1 to the COUNTER variable
	MAC START TESTDION \${COUNTER}	▪ Start the TESTDION macro for all buttons of the human interface device in succession (loop)
	ADD COUNTER \${COUNTER} 1	
	JRC -2 VAR? COUNTER < 5	
	MAC START MAINLOOP	▪ Call itself to set up the main loop
	MAC END	End macro recording
Record the TESTJOYB macro on the controller.	MAC BEG testjoyb	Start macro recording
	MEX HIB? 1 5 = 0	▪ Stop macro execution if button 5 of human interface device 1 is no longer pressed
	HIN X 1	▪ Enable control by human interface devices for axis X
	HIL 1 3 100	▪ Switch motor on (slow running): Set vibration frequency of LED 3 of human interface device 1 to 100
	JRC 6 HIB? 1 5 = 0	▪ Jump forward 6 lines (to HIN X 0) if button 5 of human interface device 1 is no longer pressed
	DEL 50	▪ Wait 50 ms
	HIL 1 3 0	▪ Switch motor off: Set vibration frequency of LED 3 of human interface device 1 to 0
	JRC 3 HIB? 1 5 = 0	▪ Jump forward 3 lines (to HIN X 0) if button 5 of human interface device 1 is no longer pressed
	DEL 200	▪ Wait 200 ms
	JRC -6 HIB? 1 5 = 1	▪ Jump back 6 lines (to HIL 1 3 100), if button 5 of human interface device 1 is still pressed

Action	Command	Description
	HIN X 0	▪ Disable control by human interface devices for axis X
	HIL 1 3 0	▪ Switch motor off: Set vibration frequency of LED 3 of human interface device 1 to 0
	MAC END	End macro recording
Record the TESTDION macro on the controller.	MAC BEG testdion	Start macro recording
	MEX VAR? 0 != 1	▪ Stop macro execution if the number of local variables given when starting TESTDION is not 1
	MEX HIB? 1 \$1 = 0	▪ Stop macro execution if the button of human interface device 1 given by the local variable 1 is no longer pressed
	DEL 300	▪ Wait 300 ms
	JRC 3 HIB? 1 \$1 = 1	▪ If the button is still pressed, jump 3 lines forward (to DEL 400)
	MAC START MVAX2ST \$1	▪ Start the MVAX2ST macro because the button was only briefly pressed. The value of local variable 1 is also used for local variable 1 in MVAX2ST. MVAX2ST moves axis X to the position assigned for the button.
	MEX HIB? 1 \$1 = 0	▪ Stop macro execution if button is no longer pressed
	DEL 400	▪ Wait 400 ms
	MEX HIB? 1 \$1 = 0	▪ Stop macro execution if button is no longer pressed
	HIL 1 3 255	▪ Switch motor on (fast running): Set vibration frequency of LED 3 of human interface device 1 to 255
	WAC HIB? 1 \$1 = 0	▪ The macro is executed further only if the button is no longer pressed
	HIL 1 3 0	▪ Switch motor off: Set vibration frequency of LED 3 of human interface device 1 to 0.
	CPY STORE\$1 POS? X	▪ Save the current position of axis X in the global variable designated via local variable 1

Action	Command	Description
	MAC END	End macro recording
Record the MVAX2ST macro on the controller.	MAC BEG mvax2st	Start macro recording
	CPY 2 VAR? STORE\$1	<ul style="list-style-type: none"> Get the global variable designated via local variable 1 and copies its value to local variable 2
	MOV X \$2	<ul style="list-style-type: none"> Start motion of axis X to the target position specified via local variable 2
	MAC END	End macro recording
Start the STARTUP macro on the controller.	MAC START startup	<p>HID control is enabled by pressing button 5. When HID control is enabled, the motor of human interface device 1 runs (slow running) and indicates that the buttons of the human interface device are not to be actuated. After button 5 has been released, HID control is disabled and the motor switches off. Buttons 1 to 4 can now be used to approach saved positions or to save the current position.</p> <p>The respective button of the human interface device is pressed briefly to move the axis of the positioner to a saved position.</p> <p>To save the current position of the axis of the positioner, one button of the human interface device is pressed until the motor starts to vibrate (fast running).</p>



8 GCS Commands

In this Chapter

Notation	133
GCS Syntax for Syntax Version 2.0	133
Variables	135
Command Overview	137
Command Descriptions for GCS 2.0	142
Error Codes	248

8.1 Notation

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

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

8.2 GCS Syntax for Syntax Version 2.0

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

Command mnemonic:

CMD ::= character1 character2 character3 [?]

Exceptions:

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

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

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

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

Exception:

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

The argument <AxisID> is used for the logical axes of the controller. Depending on the controller, an axis identifier can consist of up to 16 characters. All alphanumeric characters and the underscore are allowed. See "Commandable Elements (p. 18) for the identifiers supported by the C-884.

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?SP2SP1LF`

Receive: `2=-1158.4405SPLF`

`1=+0000.0000LF`

INFORMATION

The following restrictions apply to the C-884:

- Up to four (C-884.4DC) or six (C-884.6DC) elements can be addressed per command line (e.g., axis, channel or parameter) if the command supports this.
- Maximum length of a command line: 512 bytes
- Maximum length of an argument: 31 characters

8.3 Variables

For more flexible programming, the C-884 supports variables. While global variables are always available, local variables are only valid for a given macro. Typically, variables are used when working with macros.

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

The following conventions apply to variable names:

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

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

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

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

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

Local variables:

- Local variables can only be used in macros.
- At present, the controller firmware supports five local variables: 0, 1, 2, 3, and 4.
- The values of the local variables 1 to 4 are given as arguments of the `MAC START` or `MAC NSTART` command when the macro is started.

The command formats are:

```
MAC START <macroname> [<String1> [<String2> [String3>
[<String4>]]]]
```

```
MAC NSTART <macroname> <uint> [<String1> [<String2>
[String3> [<String4>]]]]
```

<STRING1> to <STRING4> specify the values for the local variables 1 to 4 used in the macro. <STRING1> to <STRING4> can be given directly or via the values of variables. <uint> gives the number of times the macro is to be run. See the `MAC` command (p. 203) 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.
- Inside a macro, the values of local variables can be modified using `ADD` (p. 147), `CPY` (p. 151) or `VAR` (p. 242), and can be deleted with `VAR` (except for the local variable 0).
- As long as the macro is running, the values of the local variables can be queried with

```
VAR? 0
```

```
VAR? 1
```

```
VAR? 2
```

```
VAR? 3
```

VAR? 4

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

Global variables:

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

8.4 Command Overview

Command	Arguments	Description
#4		Request Status Register (p. 142)
#5		Request Motion Status (p. 144)
#7		Request Controller Ready Status (p. 144)
#8		Query If Macro Is Running (p. 144)
#24		Stop All Axes (p. 145)
*IDN?		Get Device Identification (p. 145)
ACC	{<AxisID> <Acceleration>}	Set Closed-Loop Acceleration (p. 146)
ACC?	[[<AxisID>]]	Get Closed-Loop Acceleration (p. 146)
ADD	<Variable> <FLOAT1> <FLOAT2>	Add and Save To Variable (p. 147)
BRA	{<AxisID> <BrakeState>}	Set Brake Activation State (p. 149)
BRA?	[[<AxisID>]]	Get Brake Activation State (p. 150)
CCL	<Level> [<PSWD>]	Set Command Level (p. 150)
CCL?		Get Command Level (p. 151)
CPY	<Variable> <CMD?>	Copy Into Variable (p. 151)
CST?	[[<AxisID>]]	Get Assignment Of Stages To Axes (p. 152)
CSV?		Get Current Syntax Version (p. 153)
CTO	{<TrigOutID> <CTOPam> <Value>}	Set Configuration Of Trigger Output (p. 153)

Command	Arguments	Description
CTO?	[[<TrigOutID> <CTOPam>]]	Get Configuration Of Trigger Output (p. 157)
DEC	{<AxisID> <Deceleration>}	Set Closed-Loop Deceleration (p. 158)
DEC?	[[<AxisID>]]	Get Closed-Loop Deceleration (p. 158)
DEL	<uint>	Delay The Command Interpreter (p. 159)
DFH	[[<AxisID>]]	Define Home Position (p. 159)
DFH?	[[<AxisID>]]	Get Home Position Definition (p. 161)
DIO	{<DIOID> <OutputOn>}	Set Digital Output Lines (p. 161)
DIO?	[[<DIOID>]]	Get Digital Input Lines (p. 162)
DRC	{<RecTableID> <Source> <RecOption>}	Set Data Recorder Configuration (p. 163)
DRC?	[[<RecTableID>]]	Get Data Recorder Configuration (p. 164)
DRL?	[[<RecTableID>]]	Get Number Of Recorded Points (p. 164)
DRR?	[<StartPoint> <NumberOfPoints> [[<RecTableID>]]]	Get Recorded Data Values (p. 165)
DRT	{<RecTableID> <TriggerSource> <Value>}	Set Data Recorder Trigger Source (p. 166)
DRT?	[[<RecTableID>]]	Get Data Recorder Trigger Source (p. 168)
ERR?		Get Error Number (p. 168)
FED	{<AxisID> <EdgeID> <Param>}	Find Edge (p. 169)
FRF	[[<AxisID>]]	Fast Reference Move To Reference Switch (p. 171)
FRF?	[[<AxisID>]]	Get Referencing Result (p. 171)
GOH	[[<AxisID>]]	Go To Home Position (p. 172)
HDR?		Get All Data Recorder Options (p. 173)
HDT	{<HIDeviceID> <HIDeviceAxis> <HIDTableID>}	Set HID Default Lookup Table (p. 174)
HDT?	[[<HIDeviceID> <HIDeviceAxis>]]	Get HID Default Lookup Table (p. 175)
HIA	{<AxisID> <MotionParam> <HIDeviceID> <HIDeviceAxis>}	Configure Control Done By HID Axis (p. 176)
HIA?	[[<AxisID> <MotionParam>]]	Get Configuration Of Control Done By HID Axis (p. 177)

Command	Arguments	Description
HIB?	[[<HIDeviceID> <HIDeviceButton>]]	Get State Of HID Button (p. 178)
HIE?	[[<HIDeviceID> <HIDeviceAxis>]]	Get Deflection Of HID Axis (p. 179)
HIL	{<HIDeviceID> <HIDeviceLED> <HIDLEDState>}	Set State Of HID LED (p. 180)
HIL?	[[<HIDeviceID> <HIDeviceLED>]]	Get State Of HID LED (p. 181)
HIN	{<AxisID> <HIDControlState>}	Set Activation State For HID Control (p. 182)
HIN?	[[<AxisID>]]	Get Activation State Of HID Control (p. 182)
HIS	{<HIDeviceID> <HIDItemID> <HIDPropID> <HIDPropValue>}	Configure HI Device (p. 183)
HIS?	[[<HIDeviceID> <HIDItemID> <HIDPropID>]]	Get Configuration Of HI Device (p. 184)
HIT	{<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fill HID Lookup Table (p. 188)
HIT?	[<StartPoint> [<NumberOfPoints> [[<HIDTableID>]]]]	Get HID Lookup Table Values (p. 190)
HLP?		Get List of Available Commands (p. 193)
HLT	[[<AxisID>]]	Halt Motion Smoothly (p. 193)
HPA?		Get List Of Available Parameters (p. 194)
HPV?		Get List Of Possible Parameter Values (p. 195)
IFC	{<InterfacePam> <PamValue>}	Set Interface Parameters Temporarily (p. 196)
IFC?	[[<InterfacePam>]]	Get Current Interface Parameters (p. 197)
IFS	<Pswd> {<InterfacePam> <PamValue>}	Set Interface Parameters As Default Values (p. 198)
IFS?	[[<InterfacePam>]]	Get Interface Parameters As Default Values (p. 200)
JRC	<Jump> <CMD?> <OP> <Value>	Jump Relatively Depending On Condition (p. 201)
LIM?	[[<AxisID>]]	Indicate Limit Switches (p. 202)

Command	Arguments	Description
MAC	<keyword> {<parameter>} BEG <macro> DEF <macro> DEF? DEL <macro> END ERR? NSTART <macro> <uint> [<String1> [<String2>]] START <macro> [<String1> [<String2>]]	Call Macro Function (p. 203)
MAC?	[<macroname>]	List Macros (p. 205)
MAN?	<CMD>	Get Help String For Command (p. 206)
MAT	<Variable> = <FLOAT1> <OP> <FLOAT2>	Calculate And Save To Variable (p. 207)
MEX	<CMD?> <OP> <Value>	Stop Macro Execution Due To Condition (p. 207)
MOV	{<AxisID> <Position>}	Set Target Position (p. 209)
MOV?	[[<AxisID>]]	Get Target Position (p. 210)
MVE	{<AxisID> <Position>}	Set Target Position For Vectorial Move (p. 211)
MVR	{<AxisID> <Distance>}	Set Target Relative To Current Position (p. 212)
ONT?	[[<AxisID>]]	Get On-Target State (p. 213)
POS	{<AxisID> <Position>}	Set Real Position (p. 213)
POS?	[[<AxisID>]]	Get Real Position (p. 214)
RBT		Reboot System (p. 215)
RMC?		List Running Macros (p. 215)
RON	{<AxisID> <ReferenceOn>}	Set Reference Mode (p. 215)
RON?	[[<AxisID>]]	Get Reference Mode (p. 216)
RPA	[[<ItemID> <PamID>]]	Reset Volatile Memory Parameters (p. 216)
RTR	<RecordTableRate>	Set Record Table Rate (p. 217)
RTR?		Get Record Table Rate (p. 218)
SAI	{<AxisID> <NewIdentifier>}	Set Current Axis Identifiers (p. 218)
SAI?	[ALL]	Get List Of Current Axis Identifiers (p. 219)

Command	Arguments	Description
SEP	<Pswd> {<ItemID> <PamID> <PamValue>}	Set Nonvolatile Memory Parameters (p. 219)
SEP?	{{<ItemID> <PamID>}}	Get Nonvolatile Memory Parameters (p. 221)
SMO	{<AxisID> <ControlValue>}	Set Open-Loop Control Value (p. 221)
SMO?	{{<AxisID>}}	Get Control Value (p. 222)
SPA	{<ItemID> <PamID> <PamValue>}	Set Volatile Memory Parameters (p. 223)
SPA?	{{<ItemID> <PamID>}}	Get Volatile Memory Parameters (p. 225)
SRG?	{<AxisID> <RegisterID>}	Query Status Register Value (p. 226)
STE	<AxisID> <Amplitude>	Start Step And Response Measurement (p. 227)
STP		Stop All Axes (p. 228)
SVO	{<AxisID> <ServoState>}	Set Servo Mode (p. 228)
SVO?	{{<AxisID>}}	Get Servo Mode (p. 230)
TAC?		Tell Number Of Analog Input Lines (p. 230)
TAV?	{{<AnalogInputID>}}	Get Analog Input Voltage (p. 230)
TCV?	{{AxisID}}	Get Commanded Closed-Loop Velocity (p. 231)
TGA	{<Trajectory> <Value>}	Append Value To Trajectory (p. 231)
TGC	{{<Trajectory>}}	Clear All Values In Trajectory (p. 233)
TGF	{{<Trajectory>}}	Finalize Trajectory (p. 234)
TGL?	{{<Trajectory>}}	Get Number Of Values In Trajectory (p. 234)
TGS	{{<Trajectory>}}	Start Trajectory (p. 235)
TGT	<NoOfServoCycles>	Set Trajectory Timing (p. 236)
TGT?		Get Trajectory Timing (p. 237)
TIM	[<Float>]	Set Timer Value (p. 238)
TIM?		Get Timer Value (p. 238)
TIO?		Tell Number Of Digital I/O Lines (p. 238)
TMN?	{{<AxisID>}}	Get Minimum Commandable Position (p. 239)

Command	Arguments	Description
TMX?	[[<AxisID>]]	Get Maximum Commandable Position (p. 239)
TNR?		Get Number Of Record Tables (p. 240)
TRO	{<TrigOutID> <TrigMode>}	Set Trigger Output State (p. 240)
TRO?	[[<TrigOutID>]]	Get Trigger Output State (p. 241)
TRS?	[[<AxisID>]]	Indicate Reference Switch (p. 241)
TVI?		Tell Valid Character Set For Axis Identifiers (p. 242)
VAR	<Variable> <String>	Set Variable Value (p. 242)
VAR?	[[<Variable>]]	Get Variable Value (p. 243)
VEL	{<AxisID> <Velocity>}	Set Closed-Loop Velocity (p. 244)
VEL?	[[<AxisID>]]	Get Closed-Loop Velocity (p. 245)
VER?		Get Versions Of Firmware And Drivers (p. 245)
WAC	<CMD?> <OP> <Value>	Wait For Condition (p. 245)
WPA	<Pswd> [[<ItemID> <PamID>]]	Save Parameters To Non-Volatile Memory (p. 247)

8.5 Command Descriptions for GCS 2.0

#4 (Request Status Register)

Description: Requests system status information.

Format: #4

Arguments: None

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

Notes: This command is identical in function to SRG? (p. 226), but only one character is sent via the interface. Therefore #4 can also be used while the controller is performing time-consuming tasks.

For multi-axis controllers, the response has the following format:

0x<status Axis1><status Axis2> etc.

Deactivated axes (p. 42) are not included in the response.

<Status Axis#> is the sum of the following codes in hexadecimal format:

Bit	Description
15	On-target state
14	Determines the reference value
13	In motion
12	Servo mode on
11	-
10	-
9	-
8	Error flag
7	Digital Input 4
6	Digital Input 3
5	Digital Input 2
4	Digital Input 1
3	-
2	Positive limit switch
1	Reference point switch
0	Negative limit switch

Example:

Response of a controller with two active axes:

0x90021102

The response means:

- For axis 1: 9002
The axis is on target (on-target state = true), servo mode is ON, no error has occurred, the states of digital input lines 1 to 4 are low, and the positioner axis is on the positive side of the reference point switch.
- For axis 2: 1102
The axis is not on target (on-target state = false), servo mode is ON, an error has occurred, the states of digital input lines 1 to 4 are low, and the positioner axis is on the positive side of the reference point switch.

#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

#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. 144) 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

#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. 228), but only one character is sent via the interface.

Format: #24

Arguments: None

Response: None

Notes: #24 stops all motion caused by motion commands (e.g., MOV (p. 209), MVR (p. 212), MVE (p. 211), GOH (p. 172), STE (p. 227), SMO (p. 221)), trajectory execution (TGS (p. 235)), the command for reference point definition (FRF (p. 171)), and macros (MAC (p. 203)). Also stops execution of macros but not execution of Python scripts.

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

HLT (p. 193) in contrast to #24 stops motion with given deceleration with regard to system inertia. Does not apply to trajectories.

*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

ACC (Set Closed-Loop Acceleration)

Description:	Sets acceleration of given axes. ACC can be changed while the axis is moving.
Format:	ACC {<AxisID> <Acceleration>}
Arguments:	<AxisID> is one axis of the controller <Acceleration> is the acceleration value in physical units/s ² .
Response:	None
Troubleshooting:	Illegal axis identifiers
Notes:	The ACC setting only takes effect when the specified axis is in closed-loop operation (servo mode ON). The lowest possible value for <Acceleration> is 0. ACC changes the value of the Closed-Loop Acceleration (Phys. Unit/s²) parameter (ID 0xB) in the volatile memory of the C-884. The parameter value can be stored as default with WPA (p. 247), for details see "Adapting Settings" (p. 271). The maximum value that can be set with the ACC command is specified by the Maximum Closed-Loop Acceleration (Phys. Unit/s²) parameter (ID 0x4A).

ACC? (Get Closed-Loop Acceleration)

Description:	Queries the acceleration value set with ACC (p. 146). If all arguments are omitted, gets the value of all axes set with ACC.
Format:	ACC? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>="<float> LF} where

<float> is the acceleration value set with ACC, in physical units/s².

ADD (Add and Save to Variable)

Description: Adds two values and saves the result to a variable (p. 135).

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

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

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

<FLOAT1> is the first summand.

<FLOAT2> is the second summand.

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

Response: None

Notes: Local variables can be set using ADD in macros only.

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

```
ADD C $A $B
```

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

```
Send: VAR?
```

```
Receive:
```

```
A=468
```

```
B=123
```

```
3Z=WORKS
```

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

```
Send: VAR?
```

```
Receive:
```

```
A=468
```

```
B=123
```

```
AWORKS=591
```

```
3Z=WORKS
```

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

Send: `VAR?`

Receive:

`A=468`

`B=123`

`AWORKS=591`

`WORKS=591`

`3Z=WORKS`

Example 3:

The macros below can be used to create a "flashing light" with LEDs that are connected to the digital output lines of the controller. \$1 and \$2 are values of local variables and must be given as arguments of the MAC START or MAC NSTART command when starting the macros (see below).

DIO 0 <bitmask>: Sets the output channels according to <bitmask>. For example, "DIO 0 5" activates channels 1 and 3 and deactivates all other channels (5 is 0000 0101 in binary notation).

To implement the "flashing light", perform the following steps:

1. Write the "STEPS" macro:

```
MAC BEG STEPS
DIO 0 $1
ADD 1 $1 1
DEL $2
JRC -3 VAR? 1 <= 15
ADD 1 $1 -1
DIO 0 $1
DEL $2
JRC -3 VAR? 1 > 0
MAC END
```

2. Write the "TEST" macro:

```
MAC BEG TEST
MAC START STEPS 0 $1
ADD 1 $1 10
JRC -2 VAR? 1 < 110
VAR 1 10
ADD 2 $2 -1
JRC -5 VAR? 2 > 0
MAC END
```

3. Start the TEST macro with arguments that define the variable values \$1 and \$2:

```
MAC START Test 10 50
```

Meaning of the variables here:

\$1: Delay in ms between each step in the STEPS macro. The value is incremented by 10 by the TEST macro until it reaches 110.

\$2: Number of repetitions of the whole "flashing light" procedure.

BRA (Set Brake Activation State)

Description: Activates/deactivates brake for given axes.

Format: BRA {<AxisID> <BrakeState>}

Arguments: <AxisID> is one axis of the controller

<BrakeState> can have the following values:

0 = Brake deactivated

1 = Brake activated

Response: None

Troubleshooting: Illegal axis identifier

Notes: The brake can only be used if parameter 0x1A (**Has Brake?**) has the value 1 ("yes").

If parameter 0x1A (**Has Brake?**) has the value 1 ("yes"), the following applies:

- The brake can be activated or deactivated with BRA only if the servo mode is switched off. Secure the positioner against unintentional motion before you deactivate the brake with BRA!
- Setting the servo mode with SVO (p. 228) influences the activation state of the brake:
 - Switching on the servo mode deactivates the brake.
 - Switching off the servo mode activates the brake.
- If a motion error occurs, the servo mode is switched off and the brake is activated.

If the integrated brake driver of the C-884 is to be used, parameter 0x3094 (**Internal Brake**) must also have the value 1. For more information, see the descriptions in "Parameter Overview" (p. 280).

BRA? (Get Brake Activation State)

Description:	Gets brake activation state of given axes. If all arguments are omitted, gets state of all axes.
Format:	BRA? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>=" "<BrakeState> LF} where <BrakeState> is the current brake activation state of the axis: 0 = Brake deactivated 1 = Brake activated
Troubleshooting:	Illegal axis identifier

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: With the C-884, the command levels only determine the write permission for the parameters. The availability of the commands of the C-884 is independent of the enabled command level.

HPA? (p. 194) lists the parameters including the information on which command level allows write access to them. For further information on using parameters, see "Adapting Settings" (p. 271).

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

CCL? (Get Command Level)

Description: Get the active "command level".

Format: CCL?

Arguments: none

Response: <Level> is the currently active command level; uint.

Notes: <Level> should be 0 or 1.

<Level> = 0 is the default setting, write access is given for level 0 parameters, read access is given for all parameters

<Level> = 1 allows write access for level 1 parameters (parameters from level 0 are included).

CPY (Copy Into Variable)

Description: Copies a command response to a variable (p. 135).

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

Format: CPY <Variable> <CMD?>

Arguments: <Variable> is the name of the variable to which the command response is to be copied.

<CMD?> is one query command in its usual notation. The response has to be a single value and not more.

Response: None

Notes: Local variables can be set using CPY in macros only.

Example 1: Using the following macro, it is possible to connect through the digital input and output lines of the controller. 1 is a local variable whose value must be given as argument of the MAC START or MAC NSTART command when starting the macro.

Write the "connect" macro:

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

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

```
CPY TARGET VAR? SOURCE
```

CST? (Get Assignment Of Stages To Axes)

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

Format: CST? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

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

Notes: The positioner name is read from the **Stage Name** parameter (ID 0x3C). If the parameter has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries). The identifier of a deactivated axis can only be queried with `SAI? ALL`. See also "Deactivation of Axes" (p. 42).
You can set the parameter value to the name of your

positioner with SPA (p. 223) or SEP (p. 219). You can find details in the parameter overview (p. 280).

CSV? (Get Current Syntax Version)

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

Format: CSV?

Arguments: None

Response: The current GCS syntax version

Notes: 1.0 (for GCS 1.0) or 2.0 (for GCS 2.0) are possible responses.

CTO (Set Configuration Of Trigger Output)

Description: Configures the trigger output conditions for the given digital output line.

Format: CTO {<TrigOutID> <CTOPam> <Value>}

Arguments: <TrigOutID> is one digital output line of the controller, see below for details.

<CTOPam> is the CTO parameter ID in decimal format, see below for the available IDs.

<Value> is the value to which the CTO parameter is set, see below.

Response: None

Notes: The trigger output conditions will become active when enabled with TRO (p. 240). Do not use DIO (p. 161) on digital output lines for which the trigger output is enabled with TRO.

The CTO settings are lost when the C-884 is switched off or rebooted. An easy way to keep them is to save them to a macro.

Output
lines and trigger
conditions
available:

<TrigOutID> corresponds to digital output lines 1 to 4, IDs = 1 to 4; see "I/O" (p. 309).

<CTOPam> parameter IDs available for C-884:

1 = TriggerStep
2 = Axis
3 = TriggerMode
7 = Polarity
8 = StartThreshold
9 = StopThreshold
10 = TriggerPosition
11 = PulseWidth

<Value> available for the appropriate <CTOPam> ID:

for TriggerStep (1): Distance

for Axis (2): The axis identifier that is to be connected to the digital output line. Irrelevant for the MotionError and HardwareTrigger trigger modes.

for TriggerMode (3): Default value is 0

- 0 = PositionDistance;
a trigger pulse is written whenever the axis has covered the TriggerStep distance (<CTOPam> ID 1). Optionally, values for StartThreshold and StopThreshold (<CTOPam> IDs 8 and 9) can be defined to enable the trigger output for a limited position range and a certain direction of motion only (negative or positive; Note: If the motion direction is reversed before the axis position has reached the stop threshold, trigger pulses will continue to be generated). When StartThreshold and StopThreshold are set to the same value, they will not be used.
- 2 = OnTarget;
the on-target state of the selected axis is transferred to the selected digital output line (this state can also be read with the ONT? command).
- 5 = MotionError;
the selected digital output line becomes active when a motion error occurs. The line will stay active until the error code is reset to 0 (by a query).
- 6 = InMotion;
the selected digital output line is active as long as the selected axis is in motion (the motion state can also be read with commands, e.g. SRG? or #5).

- 7 = Position+Offset;
the first trigger pulse is written when the axis has reached the position given by TriggerPosition (<CTOPam> ID 10). The next trigger pulses are written each time the axis position equals the sum of the last valid trigger position and the distance given by TriggerStep (<CTOPam> ID 1). Trigger output ends when the axis position exceeds the value given by StopThreshold (<CTOPam> ID 9). The sign of the TriggerStep value determines the direction of motion, for which trigger pulses are to be output. Trigger processing is done by the DSP of the C-884.
- 8 = SinglePosition;
the selected digital output line is active when the axis position has reached or exceeded the position given by TriggerPosition (<CTOPam> ID 10).
- 9 = HardwareTrigger;
basically corresponds to the Position+Offset trigger mode but is executed by the FPGA circuit of the C-884 (shorter processing time). Further differences to Position+Offset: Assignment of the axes to the digital output lines is fast (axis 1 to line 1, ... , axis 4 to line 4); HardwareTrigger functions only with A/B signals. The pulse width of the trigger pulses is determined by the PulseWidth factor (<CTOPam> ID 11).

for Polarity (7): Sets the signal polarity for the digital output line, default value is 1

0 = Active Low

1 = Active High

for StartThreshold (8)/StopThreshold (9): Position value;
if used for the PositionDistance trigger mode, both thresholds must be set in order to determine the position range and the direction of motion for the trigger output;
StopThreshold is used as the stop condition for Position+Offset and HardwareTrigger trigger modes

for TriggerPosition (10): Position value;
when used in the Position+Offset and HardwareTrigger trigger modes, the first trigger pulse is output at this position;
when used in the SinglePosition trigger mode, the output line is active when this position is reached or exceeded

for PulseWidth (11): Factor "n", which determines the pulse width for the HardwareTrigger trigger mode as follows:
Pulse width = $n * 33.3 \text{ ns}$

For application examples and further details see "Digital Output Signals" (p. 91) and the lines below.

Example 1:

A pulse is to be generated on digital output line 1 (ID 1) whenever axis 1 has covered a distance of 0.05 µm. The following parameters must be set:

```
TrigOutID = 1
Axis = 1
TriggerMode = 0
TriggerStep = 0.05
Send: CTO 1 2 1
Send: CTO 1 3 0
Send: CTO 1 1 0.00005
```

Example 2:

In this example, the digital output line 1 shall be set from low to high when axis A starts its motion. The following parameters must be set:

```
TrigOutID = 1
Axis = A (axis identifier was changed with SAI)
TriggerMode = 6
Polarity = Active High
So you have to send:
CTO 1 2 A
CTO 1 3 6
CTO 1 7 1
```

Example 3:

M-122.2DD is connected to axis 1. The reference position of M-122.2DD is 18.5 mm. Starting from its reference position, the axis is to be moved alternating forwards and backwards; trigger pulses are to be output for both directions of motion in a range of 1 mm using the Position+Offset trigger mode. For that purpose, two macros are written to the controller. Macro TRIGREF initializes the controller and could also be defined as startup macro, while macro TRIGGER starts motion and hence trigger output. Write the macros as shown below. Further details about macros see "Working with Macros" (p. 119).

Make sure that the velocity for the axis matches the CTO setting for the distance. Recommended value:
 $\text{maximum velocity} = \text{distance} * 20 \text{ kHz} / 2$
 where 20 kHz is the frequency of the C-884 servo cycle.

A trigger signal frequency of 1 kHz results at a velocity of 20 mm/s.

- Record a macro named TRIGREF with the following contents:

```
CTO 1 3 7
SVO 1 1
FRF
TRO 1 1
MAC START TRIGGER
```

- Record a macro named TRIGGER with the following contents:

```
CTO 1 1 0.02
CTO 1 9 20.5
CTO 1 10 19.5
DEL 1000
DRT 0 2 0
MOV 1 20.51
WAC POS? 1 > 20.3
MEX CTO? 1 10 < 19.4
CTO 1 1 -0.02
CTO 1 9 19.5
CTO 1 10 20.5
DEL 1000
MOV 1 19.49
WAC POS? 1 < 19.5
MEX CTO? 1 10 > 19.6
MAC START TRIGGER
```

CTO? (Get Configuration Of Trigger Output)

Description: Gets the values set for specified trigger output lines and parameters.

Format: CTO? [{<TrigOutID> <CTOPam>}]

Arguments: <TrigOutID>: is a digital output line of the controller; see CTO.

<CTOPam>: parameter ID; see CTO.

If all arguments are omitted, the response contains the values for all parameters and all output lines.

Response: {<TrigOutID> <CTOPam>="<Value> LF}

For <Value> see CTO.

DEC (Set Closed-Loop Deceleration)

Description: Sets deceleration of given axes.

DEC can be changed while the axis is in motion.

Format: DEC {<AxisID> <Deceleration>}

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

<Deceleration> is the deceleration value in physical units/s².

Response: None

Troubleshooting: Illegal axis identifiers

Notes: The DEC setting only takes effect when the specified axis is in closed-loop operation (servo mode ON).

The lowest possible value for <Deceleration> is 0.

DEC changes the value of the **Closed Loop Deceleration (Phys. Unit/s²)** parameter (ID 0xC) in the volatile memory of the C-884. The parameter value can be stored as default with WPA (p. 247), for details see "Adapting Settings" (p. 271).

The maximum value that can be set with the DEC command is specified by the **Maximum Closed-Loop Deceleration (Phys. Unit/s²)** parameter (ID 0x4B).

DEC? (Get Closed-Loop Deceleration)

Description: Queries the deceleration value set with DEC (p. 158).

If all arguments are omitted, gets the value of all axes set with DEC.

Format: DEC? [{<AxisID>}]

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

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

where

<float> is the deceleration value set with DEC, in physical units/s².

DEL (Delay the Command Interpreter)

Description: Delays <uint> milliseconds.

Format: DEL <uint>

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

Response: None

Notes: DEL can only be used in macros. Do not mistake MAC DEL (deletes macros) for DEL (delays). Further information can be found in the description of the MAC command (p. 203) and in the "Controller Macros" (p. 117) section.

DFH (Define Home Position)

Description: Redefines the zero position of the given axis by setting the position value to zero at the current position.

If all arguments are omitted, DFH defines the zero position of all axes.

Format: DFH [{<AxisID>}]

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

Response: none

Troubleshooting: Illegal axis identifier

Notes: DFH sets the current position of the axis to zero and saves the position value which was valid when the command was called as offset in the volatile memory. By adding this offset to the response, the output values of the following commands are adapted to the new zero position:

- POS? (p. 214) (query of the current position)
- TMN? (p. 239) (query of the minimum commandable position)
- TMX? (p. 239) (query of the maximum commandable position)

DFH does **not** change the values of the parameters for the definition of travel range and soft limits (p. 34).

The offset is reset to zero in the following cases:

- When switching on and rebooting the C-884:
For all axes
- During reference point definition:
For the axis in question

Example:

```
Send: MOV 1 9.87
Send: POS? 1
Receive: 1=9.8700005
Send: DFH? 1
Receive: 1=0.0000000
Send: TMN? 1
Receive: 1=0.0000000
Send: TMX? 1
Receive: 1=14.9999982
```

Note: Axis 1 is moved to absolute position 9.87 mm. Then the current axis position (with POS?), the current offset value (with DFH?) and the minimum and maximum commandable position (with TMN? and TMX?) are queried.

```
Send: DFH 1
Send: POS? 1
Receive: 1=0.0000000
Send: DFH? 1
Receive: 1=9.8700005
Send: TMN? 1
Receive: 1=-9.8700005
Send: TMX? 1
Receive: 1=5.1299978
```

Note: The axis has not moved. The current axis position was defined as the new zero position using DFH. As a

result, the offset value for axis 1 is now 9.87 mm. The values for the minimum and maximum commandable position were adapted to the new zero position by adding the offset.

DFH? (Get Home Position Definition)

Description:	Queries the position value that is currently used as the offset for the specified axis to move the zero position.
	If all arguments are omitted, queries the position value of all axes.
Format:	DFH? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>=" "<PositionOffset> LF}
	where
	<PositionOffset> is the axis position that was valid at the time the last DFH command was processed. This position value is used internally as offset for the calculation of the current axis position.
Troubleshooting:	Illegal axis identifier
Notes:	<p>The axis position that was valid when the last DFH command was processed, is available in the volatile memory as an offset. The offset is reset to zero in the following cases:</p> <ul style="list-style-type: none"> ▪ When switching on and rebooting the C-884: For all axes ▪ During reference point definition: For the affected axis <p>See DFH for an example.</p>

DIO (Set Digital Output Line)

Description:	Switches the specified digital output line(s) to specified state(s).
	Use TIO? (p. 238) to get the number of installed digital I/O lines.
Format:	DIO {<DIOID> <OutputOn>}

Arguments:	<p><DIOID> is one digital output line of the controller, see below for details.</p> <p><OutputOn> is the state of the digital output line, see below for details.</p>
Response:	none
Notes:	<p>Using the DIO command, you can activate/deactivate digital output lines 1 to 4, which are located on the I/O socket (p. 309).</p> <p>The <DIOID> identifiers to use for the lines are 1 to 4. With the identifier 0, all lines are set according to a bit pattern given by <OutputOn>.</p> <p>If <OutputOn>=1 the line is set to HIGH/ON, if <OutputOn>=0 it is set to LOW/OFF.</p> <p>Do not use DIO on output lines for which the trigger output is enabled with TRO (p. 240).</p>

DIO? (Get Digital Input Lines)

Description:	<p>Gets the states of the specified digital input lines.</p> <p>Use TIO? (p. 238) to get the number of available digital I/O lines.</p>
Format:	DIO? [{<DIOID>}]
Arguments:	<DIOID> is the identifier of the digital input line, see below for details.
Response:	<p>{<DIOID>="<InputOn> LF}</p> <p>where</p> <p><InputOn> gives the state of the digital input line, see below for details.</p>
Notes:	You can use the DIO? command to directly read the digital input lines 1 to 4 that are located on the I/O socket (p. 309).

The <DIOID> identifiers to use for the lines are 1 to 4. If the identifier is omitted or 0, all lines are queried.

If <InputOn>=0, the digital input is LOW/OFF; if <InputOn>=1, the digital input is HIGH/ON. If <DIOID> is 0, <InputOn> is a bit pattern which gives the states of all lines in hexadecimal format.

DRC (Set Data Recorder Configuration)

Description:	Determines the data source to be used and the type of data to be recorded (record option) for the data recorder table given.
Format:	DRC {<RecTableID> <Source> <RecOption>}
Arguments:	<p><RecTableID> is one data recorder table of the controller, see below.</p> <p><Source> is the ID of the data source, for example, an axis or channel of the controller. The required source depends on the selected record option.</p> <p><RecOption> is the type of data to be recorded (record option).</p> <p>For details see the following list of the available record options and the corresponding data sources.</p>
Response:	none
Notes:	<p>The C-884 has 8 data recorder tables with 8192 points per table.</p> <p>With HDR? (p. 173), you will obtain a list of all available record and trigger options and additional information on the data recording. The number of available data recorder tables can be read with TNR? (p. 240).</p> <p>For further information, see "Data Recorder" (p. 89).</p>
Recording options available with the corresponding data sources:	<p>0=Nothing is recorded</p> <p>1=Commanded position of axis</p> <p>2=Actual position of axis</p> <p>3=Position error of axis</p> <p>44=Timestamp (TIM?)</p> <p>70=Commanded velocity of axis</p> <p>71=Commanded acceleration of axis</p>

73=Motor output of axis
 74=Kp of axis
 75=Ki of axis
 76=Kd of axis
 80=Signal status register of axis
 81=Analog input (channel = 1 - 4)
 86=Number of fifo values (for future use)
 87=Interpolation data (for future use)
 91=Motor current

Note: The input channels for the record option 81 are analog input lines 1 to 4 of the **I/O** socket (p. 309).

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. 173).

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 <RecTableID> is omitted, the data from all tables with nonzero record option is read.

With HDR? (p. 173) you will obtain a list of available record and trigger options and additional information about data recording.

Further information, see the description of the DRC (p. 163) command as well as "Data Recorder" (p. 89).

Example:

```
rtr?
10
drr? 1 20
# REM C-884
#
# VERSION = 1
```

```
# TYPE = 1
# SEPARATOR = 32
# DIM = 8
# SAMPLE_TIME = 0.0001000
# NDATA = 20
<<#
<<# NAME0 = Actual Position of Axis
AXIS:1
<<# NAME1 = Actual Position of Axis
AXIS:2
<<# NAME2 = Actual Position of Axis
AXIS:3
<<# NAME3 = Actual Position of Axis
AXIS:4
<<#
<<# END_HEADER
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<12.9999989 0.0000000 0.00020 -0.00010
<<13.0000055 0.0000000 0.00020 -0.00010
<<13.0000192 0.0000000 0.00020 -0.00010
<<13.0000330 0.0000000 0.00020 -0.00010
<<13.0000536 0.0000000 0.00020 -0.00010
<<13.0000810 0.0000000 0.00020 -0.00010
<<13.0001154 0.0000000 0.00020 -0.00010
<<13.0001497 0.0000000 0.00020 -0.00010
<<13.0001909 0.0000000 0.00020 -0.00010
<<13.0002390 0.0000000 0.00020 -0.00010
<<13.0002870 0.0000000 0.00020 -0.00010
<<13.0003419 0.0000000 0.00020 -0.00010
```

DRT (Set Data Recorder Trigger Source)

Description: Defines a trigger source for the given data recorder table.

Format: DRT <RecTableID> <TriggerSource> <Value>

Arguments:	<p><RecTableID> is one data recorder table of the controller. See below for details.</p> <p><TriggerSource> ID of the trigger source, see below for a list of available options.</p> <p><Value> depends on the trigger source, can be a dummy, see below.</p>
Response:	none
Notes:	<p>At present, only 0 is valid for <RecTableID>; this means that the given trigger source is set for all data recorder tables that have a record option that is not zero.</p> <p>Irrespective of the trigger option set, the data recording is always triggered when a step response measurement is carried out with STE (p. 227).</p> <p>With HDR? (p. 173), you will obtain a list of available record and trigger options and additional information on data recording.</p> <p>For further information, see the description of the DRC command (p. 163) as well as "Data Recorder" (p. 89).</p>
Available trigger options:	<p>0 = default setting; data recording is triggered by STE; <Value> must be a dummy.</p> <p>1 = any command changing the target position (e.g., MVR (p. 212), MOV (p. 209), MVE (p. 211)); <Value> must be a dummy.</p> <p>2 = next command, resets trigger after execution; <Value> must be a dummy.</p> <p>3 = external trigger; data recording is started with the digital input line whose ID is given by <Value> (see "I/O" for available input lines).</p> <p>6 = any command changing target position (e.g., MVR, MOV, MVE); resets Trigger after execution; <Value> must be a dummy.</p> <p>7 = SMO command, resets trigger after execution; <Value> must be a dummy.</p>

Example: The recording is to be triggered for all data recorder tables via digital input line 1. Send:

```
DRT 0 3 1
```

The servo cycle time of the C-884 is 50 μ s. For this reason, the trigger signal should have a frequency ≤ 10 kHz. With RTR you can set the record table rate, i.e., the number of servo cycles to be used in data recording operations.

DRT? (Get Data Recorder Trigger Source)

Description: 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. 166).

Notes: Since all data recorder tables of the C-884 have the same trigger source, the DRT? response is given as a single line of the form

0=<TriggerSource> <Value>

ERR? (Get Error Number)

Description: Get error code <int> of the last occurred error and reset the error to 0.

Only the last error is buffered. You should therefore call ERR? after each command.

The error codes and their descriptions are listed in "Error Codes" (p. 248).

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). <p>If the cause of an error continues, the corresponding error code is immediately set again after a query with ERR?.</p>

FED (Find Edge)

Description:	<p>Moves the given axis to a given signal edge.</p> <p>FED does not set a certain position value at the selected edge (in contrast to the FRF (p. 171) command for reference point definition), i.e., the axis is not referenced after FED is used.</p> <p>If multiple axes are given in the command, they are moved synchronously.</p>
Format:	FED {<AxisID> <EdgeID> <Param>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><EdgeID> is the type of edge the axis has to move to. See below for available edge types.</p> <p><Param> depends on the selected edge and qualifies it. See below for details.</p>
Response:	None
Troubleshooting:	<ul style="list-style-type: none"> ▪ Illegal axis identifier ▪ Limit switches and/or reference point switches are disabled ▪ Servo mode is switched off.

Notes: Servo mode must be switched on with SVO (p. 228) for the commanded axis prior to using this command (closed-loop operation).

The firmware of the C-884 determines the following based on parameters:

- Is a reference point switch present (parameter 0x14)?
- Are limit switches present (parameter 0x32)?
- If the reference point switch is represented by an index pulse: How should the move to the index pulse take place (parameters 0x70, 0x78, 0x79)?

According to the values of those parameters, the C-884 activates or deactivates FED motion to the appropriate signal edges. Adapt the parameter values to your hardware using SPA (p. 223) or SEP (p. 219). Further information can be found in the "Adapting Settings" section (p. 271).

You can use the digital input lines instead of the switches as source of the switch signals for FED. For further information, see "Digital Input Signals" (p. 101).

FED can be used to measure the physical travel range of a new mechanical system and therefore determine the values for the corresponding parameters:

- Distance from the negative to the positive limit switch
- Distance between the negative limit switch and the reference point switch (parameter ID 0x17)
- Distance between the reference point switch and the positive limit switch (parameter ID 0x2F).

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

The motion can be stopped by #24 (p. 145), STP (p. 228) and HLT (p. 193).

Motion commands such as FED are not allowed when the HID control is enabled for the axis. For further information, see "Control with an HID Device" (p. 106).

Available edge types and parameters:

The following edge types with their parameter settings are available:

- 1 = negative limit switch, <Param> must be 0
- 2 = positive limit switch, <Param> must be 0
- 3 = reference point switch, <Param> must be 0

FRF (Fast Reference Move To Reference Switch)

Description:	<p>Performs a reference move.</p> <p>Moves the specified 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 started simultaneously.</p>
Format:	FRF [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller, if omitted, all axes are affected.
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>Servo mode must be switched on with SVO (p. 228) for the commanded axis prior to using this command (closed-loop operation).</p> <p>If the reference move was successful, absolute motion will then be possible in closed-loop operation.</p> <p>The Reference Signal Type parameter (0x70) is evaluated for the reference move. Further information, see "Reference Point Definition" (p. 38).</p> <p>The value of the Value At Reference Position parameter (0x16) is set as the current position when the axis is at the reference position.</p> <p>You can use a digital input instead of the reference point switch as source of the reference signal for the FRF command. For further information, see "Digital Input Signals" (p. 101).</p> <p>The motion can be stopped by #24 (p. 145), STP (p. 228), and HLT (p. 193).</p> <p>Use FRF? (p. 171) to check whether the reference move was successful.</p>

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

GOH (Go To Home Position)

Description: Moves the given axis to the zero position.

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

The motion can be stopped by #24 (p. 145), STP (p. 228), and HLT (p. 193).

Format: GOH [{<AxisID>}]

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

Response: None

Troubleshooting: Illegal axis identifier

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

Motion commands such as GOH are not permitted when HID control is enabled for the axis. For further information, see "Control with a Human Interface Device" (p. 106).

The execution of motion commands can be configured with the **Inhibit Motion Commands** parameter (ID0x130); for more information, see "Motion Triggering" (p. 23).

The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 25).

HDR? (Get All Data Recorder Options)

Description: Lists a help string which contains all information available on data recording (record options and trigger options, information on additional parameters and commands concerned with data recording).

Format: HDR?

Arguments: None

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

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

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

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

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

end of help

Example: For the C-884, the response to HDR? reads as follows:

```
HDR?
#RecordOptions
0=Nothing is recorded
1=Commanded Position of Axis
2=Actual Position of Axis
3=Position Error of Axis
44=Timestamp (TIM?)
70=Commanded Velocity of Axis
71=Commanded Acceleration of Axis
73=Motor Output of Axis
74=Kp of Axis
75=Ki of Axis
76=Kd of Axis
80=Signal Status Register of Axis
81=Analog input (Channel = 1 - 4)
86=Number of fifo values
87=Interpolation data
```

```

91=Motor Current
#TriggerOptions
0=default setting
1=any command changing position (e.g.,
MOV)
2=next command
3=external trigger
6=any command changing position (e.g.,
MOV), reset trigger after execution
7=with SMO command, reset trigger after
execution
#Additional information
8 record tables
8192 datapoints per table
end of help

```

Notes:

TriggerOptions = 0 (default) means that recording is triggered by the STE command (p. 227).

HDT (Set HID Default Lookup Table)

Description: Assigns a lookup table to the given axis of the given human interface device.

Lookup tables are used during HID control of several motion parameters of the axes of the C-884; for details, see HIA (p. 176). A lookup table maps the displacement of the axis of a human interface device to the controlled motion parameter (for further details, see HIE? (p. 179)).

Format: HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}

Arguments: <HIDDeviceID> is one human interface device connected to the controller; see below for details.

<HIDDeviceAxis> is one axis of the human interface device; for more details, see below.

<HIDTableID> is one lookup table of the controller; for more details, see below.

Response: None

Notes: Lookup tables are only assigned with HDT in the volatile memory (RAM) of the C-884. With the WPA command (p.

247), the currently valid assignment can be saved in the nonvolatile memory of the C-884.

Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5. Information on the supported axes of the HID devices can be queried with the HIS? command. Further information see "Commandable Items" (p. 18).

Available lookup tables:

The C-884 provides the following lookup tables with 256 points each:

Identifier	Type
1	linear
2	parabolic (default)
101	User-defined
102	User-defined
103	User-defined
104	User-defined

With the HIT command (p. 188), user-defined lookup tables can be filled with values.

HDT? (Get HID Default Lookup Table)

Description: Gets the currently assigned lookup table for the given axis of the given human interface device.

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

Arguments: <HIDeviceID> is one human interface device connected to the controller; for more details, see HDT.

<HIDeviceAxis> is one axis of the human interface device; for more details, see HDT.

Response: {<HIDeviceID> <HIDeviceAxis>="<HIDTableID>LF}

where

<HIDTableID> is one lookup table of the controller; for more details, see HDT.

HIA (Configure Control Done By HID Axis)

Description:	<p>Configures the control of axes of the C-884 by axes of human interface devices ("HID control").</p> <p>Assigns an axis of a human interface device to the given motion parameter of the given axis of the C-884.</p> <p>HID control is enabled or disabled with the HIN command (p. 182). HIA can only be used when HID control is disabled for the affected axis of the C-884.</p>
Format:	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><MotionParam> is one motion parameter of the controller axis; for more details, see below.</p> <p><HIDDeviceID> is one human interface device connected to the controller; see below for details.</p> <p><HIDDeviceAxis> is one axis of the human interface device; for more details, see below.</p>
Response:	None
Notes:	<p>The HID control is only configured with HIA in the volatile memory (RAM) of the C-884. With the WPA command (p. 247), the currently valid configuration can be saved in the nonvolatile memory of the C-884.</p> <p>Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5. Information on the identifiers of the supported axes of the HID devices can be queried with the HIS? command (p. 184). Further information see "Commandable Items" (p. 18).</p> <p><MotionParam> gives the motion parameter to be controlled and can take on the following values:</p> <p>Value Motion parameter</p>

- 0 Deletes the current configuration of the HID control.
Can be sent in abbreviated notation without specification of <HIDDeviceID> and <HIDDeviceAxis> as follows:
HIA <AxisID> 0
- 3 Velocity of the controller axis.
Product of the lookup table value that corresponds to the current displacement of the axis of the HID device and the currently valid maximum velocity of the controller axis. The currently valid maximum velocity is specified by one of the following sources:
 - Value of the parameter 0x74
 - Value of the parameter 0x49 when the value of the parameter 0x74 is zero
 - Displacement of an axis of an HID device, see below.
- 4 Maximum velocity of the controller axis.
Product of the lookup table value that corresponds to the current displacement of the axis of the HID device and the value of the **Closed-Loop Velocity For HI Control** parameter (ID 0x74). If the parameter 0x74 has the value zero, the lookup table value is multiplied by the value of the parameter 0x49.

If the HID control is enabled with the HIN command, it will remain without effect in the following cases:

- <MotionParam> has the value zero, i. e. no function to be controlled is selected for the axis of the C-884.
- <HIDDeviceID> has the value zero, i. e. no HID device is selected for the HID control.
- <HIDDeviceAxis> has the value zero, i. e. no axis of the HID device is selected for the HID control.

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

Description: Gets the current control configuration for the given motion parameter of the given axis of the C-884, i.e., the currently assigned axis of a human interface device.

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

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

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

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

 where

 <HIDeviceID> is one human interface device connected to the controller; for more details, see HIA.

 <HIDeviceAxis> is one axis of the human interface device; for more details, see HIA.

HIB? (Get State Of HID Button)

Description: Gets the current state of the given button of the given human interface device.

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

Arguments: <HIDeviceID> is one human interface device connected to the controller; see below for details.

 <HIDeviceButton> is one button of the human interface device; for more details, see below.

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

 where

 <HIDButtonState> indicates the state of the button as an integer value:

 The possible values depend on the button type. The value range for the individual buttons can be queried with the HIS? command (p. 184). When only the values 0 and 1 are allowed, they have the following meaning:
 0 = Button not pressed, 1 = Button pressed
 The meaning of values > 1 depends on the human interface device.

Notes: Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5. Information on the identifiers of the supported buttons of the HID devices can be queried with the HIS? command (p. 184). Further information see "Commandable Items" (p. 18).

HIE? (Get Deflection Of HID Axis)

Description: Gets the current displacement of the given axis of the given human interface device.

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

Arguments: <HIDDeviceID> is one human interface device connected to the controller; see below for details.

<HIDDeviceAxis> is one axis of the human interface device; for more details, see below.

Response: {<HIDDeviceID> <HIDDeviceAxis> "="<HIDDeflection>}

where

<HIDDeflection> indicates the current displacement of the axis of the human interface device; for more details, see below.

Notes: Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5. Information on the identifiers of the supported axes of the HID devices can be queried with the HIS? command (p. 184). Further information see "Commandable Items" (p. 18).

<HIDDeflection> indicates the current displacement of the axis of the human interface device as a floating-point number in the range from -1.0 to 1.0.

For human interface device axes with hard stops, the value -1.0 corresponds to the maximum displacement in the negative direction and the value 1.0 corresponds to the maximum displacement in the positive direction.

The C-884 processes the information received by the human interface device so that 256 different displacement

values can be shown. When HID control takes place for a motion parameter on the basis of lookup tables, exactly one point in the currently assigned lookup table is assigned to each of these displacement values (for more details, see HDT (p. 174) and HIT (p. 188)).

Example:

Send: `HIE? 1 1 1 2`

Receive: `1 1=0.02`

`1 2=-0.7`

Note: Displacement of axes 1 and 2 of human interface device 1:
Axis 1 has the value 0.02, which corresponds to approximately the middle position.
Axis 2 has the value -0.7, i. e. it is displaced in the negative direction by around 2/3.

HIL (Set State Of HID LED)

Description: Sets the current state of the given output unit or characteristic ("LED") of the given HID device.

Format: `HIL {<HIDDeviceID> <HIDDeviceLED> <HIDLEDState>}`

Arguments: <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDDeviceLED> is one output unit or characteristic of the HID device; for more details, see below.

<HIDLEDState> is the state of the given output unit or characteristic of the HID device; for more details, see below.

Response: None

Notes: Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5. Information on the identifiers of the supported output units or characteristics of the HID devices can be queried with the HIS? command (p. 184). Further information see "Commandable Items" (p. 18).

The meaning of <HIDDeviceLED> and the possible values for <HIDLEDState> depend on the connected HID device and can be queried with the HIS? command.

Example: A gamepad is connected to the C-884 as an HID device. This gamepad has one LED and 2 motors for the vibration function. The flashing frequency of the LED as well as the vibration frequency of the motors can be adjusted. <HIDDeviceLED> and <HIDLEDState> therefore have the following values specifically for this gamepad (the values can be completely different for other HID devices).

Adjustable function	<HIDDeviceLED>	Possible values for <HIDLEDState>
Illumination state of the LED	1 = LED	0 = Off 1 = Continuously on 2 = Flashing
PWM control of the motors for the vibration function	2 = On time for slow vibration 3 = Off time for slow vibration 4 = On time for fast vibration 5 = Off time for fast vibration	Values between 0 and 255 determine the time duration each.

HIL? (Get State Of HID LED)

Description: Gets the current state of the given output unit or characteristic ("LED") of the given HID device.

Format: HIL? [{<HIDDeviceID> <HIDDeviceLED>}]

Arguments: <HIDDeviceID> is one HID device connected to the controller; for more details, see HIL.

<HIDDeviceLED> is one output unit or characteristic of the HID device; for more details, see HIL.

Response: {<HIDDeviceID> <HIDDeviceLED> "="<HIDLEDState>}

where

<HIDLEDState> indicates the current state of the output unit or characteristic as an integer value; for more details, see HIL.

HIN (Set Activation State For HID Control)

Description: Enables or disables the control by human interface devices ("HID control") that are connected to the controller for the given axis of the C-884.

HID control is configured with the HIA command (p. 176).

Format: HIN {<AxisID> <HIDControlState>}

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

<HIDControlState> is the activation state of HID control:
 0 = Control by human interface devices is disabled
 1 = Control by human interface devices is enabled

Response: None

Notes: Up to five human interface devices can be connected to the type A USB socket of the C-884. A USB hub is required to connect more than one human interface device. For further information, see "Connecting a Human Interface Device" (p. 48).

The enabled HID control will not have any effect if it has not been suitably configured with HIA.

During HID control, the target position of the controlled axis of the C-884 is set to the soft limit that is given by the parameter 0x15 or 0x30. Details on the parameters can be found in "Travel Range and Soft Limits" (p. 34). When HID control is disabled, the target position is set to the current position of the controlled axis.

No HID control is possible in open-loop operation (servo mode switched off).

Motion commands such as MOV (p. 209) and the execution of trajectories (TGS (p. 235)) are not permitted when HID control is enabled for the axis. For further information, see "Control with a Human Interface Device" (p. 106).

HIN? (Get Activation State Of HID Control)

Description: Gets the activation state of the control by human interface devices ("HID control") that are connected to the controller for the given axis of the C-884.

Format: HIN? [{<AxisID>}]

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

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

where

<HIDControlState> is the activation state of HID control:
 0 = Control by human interface devices is disabled
 1 = Control by human interface devices is enabled

HIS (Configure HI Device)

Description: Configures the given HID device.

Format: HIS {<HIDDeviceID> <HIDItemID> <HIDPropID>
 <HIDPropValue>}

Arguments: <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDItemID> is one operating element of the HID device: for more information, see below.

<HIDPropID> is one property of the operating element of the HID device; for more information, see below.

<HIDPropValue> is a string with the value to which the property of the operating element is set; for more information, see below.

Response: None

Note: For the C-884, the functionality of HIS corresponds to that of the HIL command (p. 180).

Supported operating elements and their properties: Up to five HID devices can be connected to the USB socket (type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5.

For <HIDItemID>, all supported operating elements of an HID device are consecutively numbered, starting with 1. For the connected HID devices, the values for <HIDItemID> and their assignment to the operating elements can be determined with the HIS? command (p. 184). <HIDItemID> may only take on values for HIS that

correspond to the output units or characteristics ("LEDs") of the HID device. Axes or buttons of HID devices cannot be configured with HIS.
Further information, see "Commandable Items" (p. 18).

Possible values for <HIDPropID>:
2 = Value for the current state of the output unit or characteristic.

Values for <HIDPropValue> when <HIDPropID> = 2:
The possible values for <HIDPropValue> and their meaning depend on the connected HID device; see also HIL.

HIS? (Get Configuration Of HI Device)

Description: Gets the given property for the given operating element of a human interface device.

Format: HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]

Arguments: <HIDDeviceID> is one human interface device connected to the controller; see below for details.

<HIDItemID> is one operating element of the human interface device; for more details, see below.

<HIDPropID> is one property of the operating element of the human interface device; for more details, see below.

Response: {<HIDDeviceID> <HIDItemID>
<HIDPropID>="<HIDPropValue>LF}

where

<HIDPropValue> is a string with the value to which the property of the operating element is set; for more details, see below.

Notes: After an HID device has been connected to the C-884 via the USB socket (type A), HIS? should first be sent without arguments. The response contains information on all supported operating elements of all connected HID devices. This information can be used to address specific operating elements of HID devices with the corresponding commands.

Supported Up to five HID devices can be connected to the USB socket

operating
elements and
their properties

(type A) of the C-884. A USB hub is required to connect more than one HID device. The identifiers of the HID devices are 1 to 5.

For <HIDItemID>, all supported operating elements of an HID device are consecutively numbered starting with 1, regardless of their type.

The following properties of an HID device are shown with HIS?:

For <HIDPropID> = 1:

<HIDPropValue> indicates the type and the identifier of the operating element. Possible types:

- "Axis" = Axis of an HID device, can be e. g. a joystick axis or a continuous slider
- "Button" = Button of an HID device, can be e. g. a pushbutton
- "Led" = Output unit or characteristic of the HID device, can be e. g. an LED or the on/off time of a motor for the vibration function

The type is followed by the identifier, separated by an underscore. For the identifier, all operating elements of the same type are consecutively numbered for each HID device. Except for in the HIS command (p. 183), the identifier must be used in all relevant commands, in order to specifically address the operating element.

For <HIDPropID> = 2:

<HIDPropValue> is the value for the current state of the operating element. The meaning of the value depends on the type of operating element:

- "Axis": Current displacement of the axis; for more information, see HIE? (p. 179)
- "Button": Current state of the button; for more information, see HIB? (p. 178)
- "Led": Meaning can be deduced from the name of the output unit or characteristic, i. e. the value for <HIDPropID> = 3, see below. For an example, see below and HIL (p. 180).

For <HIDPropID> = 3:

<HIDPropValue> is the name of an axis or an output unit or characteristic ("Led") of an HID device

For <HIDPropID> = 4:

<HIDPropValue> is the name of the HID device

For <HIDPropID> = 5:

<HIDPropValue> indicates the smallest possible value for the state of an operating element of the "button" or "LED" type.

For <HIDPropID> = 6:

<HIDPropValue> indicates the largest possible value for the state of an operating element of the "button" or "LED" type.

Example:

```
HIS?
1 1 1=Axis_1
1 1 2=-0.992
1 1 3=X
1 1 4=Colour Rumble Pad
1 2 1=Axis_2
1 2 2=-0.992
1 2 3=Y
1 2 4=Colour Rumble Pad
1 3 1=Axis_3
1 3 2=-0.992
1 3 3=Z
1 3 4=Colour Rumble Pad
1 4 1=Axis_4
1 4 2=-0.992
1 4 3=Rz
1 4 4=Colour Rumble Pad
1 5 1=Axis_5
1 5 2=0.008
1 5 3=Hat switch
1 5 4=Colour Rumble Pad
1 6 1=Button_1
1 6 2=0
1 6 4=Colour Rumble Pad
1 6 5=0
1 6 6=1
1 7 1=Button_2
1 7 2=0
1 7 4=Colour Rumble Pad
1 7 5=0
1 7 6=1
1 8 1=Button_3
1 8 2=0
1 8 4=Colour Rumble Pad
```



```
1 8 5=0
1 8 6=1
1 9 1=Button_4
1 9 2=0
1 9 4=Colour Rumble Pad
1 9 5=0
1 9 6=1
1 10 1=Button_5
1 10 2=0
1 10 4=Colour Rumble Pad
1 10 5=0
1 10 6=1
1 11 1=Button_6
1 11 2=0
1 11 4=Colour Rumble Pad
1 11 5=0
1 11 6=1
1 12 1=Button_7
1 12 2=0
1 12 4=Colour Rumble Pad
1 12 5=0
1 12 6=1
1 13 1=Button_8
1 13 2=0
1 13 4=Colour Rumble Pad
1 13 5=0
1 13 6=1
1 14 1=Button_9
1 14 2=0
1 14 4=Colour Rumble Pad
1 14 5=0
1 14 6=1
1 15 1=Button_10
1 15 2=0
1 15 4=Colour Rumble Pad
1 15 5=0
1 15 6=1
1 16 1=Button_11
1 16 2=0
1 16 4=Colour Rumble Pad
1 16 5=0
1 16 6=1
1 17 1=Button_12
```

```

1 17 2=0
1 17 4=Colour Rumble Pad
1 17 5=0
1 17 6=1
1 18 1=Led_1
1 18 2=0
1 18 3=Slow Blink On Time
1 18 4=Colour Rumble Pad
1 18 5=0
1 18 6=255
1 19 1=Led_2
1 19 2=0
1 19 3=Slow Blink Off Time
1 19 4=Colour Rumble Pad
1 19 5=0
1 19 6=255
1 20 1=Led_3
1 20 2=0
1 20 3=Fast Blink ON Time
1 20 4=Colour Rumble Pad
1 20 5=0
1 20 6=255
1 21 1=Led_4
1 21 2=0
1 21 3=Fast Blink OFF Time
1 21 4=Colour Rumble Pad
1 21 5=0
1 21 6=255

```

HIT (Fill HID Lookup Table)

Description: Fills the given lookup table with values.

Lookup tables are used during HID control of several motion parameters of the axes of the C-884; for details, see HIA (p. 176). A lookup table maps the displacement of the axis of a human interface device to the controlled motion parameter (for further details, see HIE? (p. 179)).

With the HDT command (p. 174), the lookup tables are assigned to the axes of human interface devices.

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

Arguments:	<p><HIDTableID> is one lookup table of the controller; for more information, see below.</p> <p><HIDTableAddr> is the index of a point in the lookup table, begins with 1, for number of points per table, see below.</p> <p><HIDTableValue> is the value of point n as a floating-point number in the range from -1.0 to 1.0; for further information, see below.</p>
Response:	None
Notes:	<p>The lookup tables are only filled by HIT in the volatile memory (RAM) of the C-884. With the WPA command (p. 247), the currently valid table contents can be saved in the nonvolatile memory of the C-884.</p> <p>The values of up to 4 points can be sent to the C-884 per HIT command.</p>
Available lookup tables:	The C-884 provides the following lookup tables with 256 points each:

Identifier	Type
1	linear
2	parabolic (default)
101	User-defined
102	User-defined
103	User-defined
104	User-defined

HIT can only be used to fill user-defined tables. Tables with a ≤ 100 identifier are predefined and write-protected.

The first point of a lookup table corresponds to the maximum axis displacement of the HID device in the negative direction, the 256th point to the maximum displacement in the positive direction. The values for points 1 to maximally 127 have a negative sign by default, while the remaining values have a positive sign. The sign of the values determines the direction of motion of the HID-controlled axis.

The ***Invert Direction Of Motion For Joystick-Controlled Axis?*** parameter (ID 0x61) can be used to reverse the direction assignment that was given by the values of the lookup table for an HID-controlled axis of the C-884.

HIT? (Get HID Lookup Table Values)

- Description:** Gets the values of the given points in the given lookup table.
- Format:** HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]
- Arguments:** <StartPoint> is the index of the first point to be queried in the lookup table, smallest possible value is 1.
- <NumberOfPoints> gives the number of the points to be queried per lookup table; for more details, see HIT.
- <HIDTableID> is one lookup table of the controller; for more details, see HIT.
- Response:** The lookup table content in GCS array format, see the separate manual for GCS array, SM 146E, and the example below.
- Example:** hit?

```
# TYPE = 1
#
# SEPARATOR = 32
# DIM = 6
# NDATA = 256
# NAME0 = Table 1
# NAME1 = Table 2
# NAME2 = Table 101
# NAME3 = Table 102
# NAME4 = Table 103
# NAME5 = Table 104
# END_HEADER
-1.0000 -1.0000 -1.0000 -1.0000 -1.0000
-1.0000
-0.9922 -0.9834 -1.0000 -1.0000 -0.9834
-0.9834
-0.9834 -0.9678 -0.9922 -0.9922 -0.9678
-0.9678
-0.9756 -0.9521 -0.9834 -0.9834 -0.9521
-0.9521
-0.9678 -0.9355 -0.9756 -0.9756 -0.9355
-0.9355
-0.9590 -0.9199 -0.9668 -0.9668 -0.9199
-0.9199
-0.9512 -0.9053 -0.9590 -0.9590 -0.9053
-0.9053
```

```

...
-0.5938 -0.3525 -0.5986 -0.5986 -0.3525
-0.3525
-0.5850 -0.3428 -0.5898 -0.5898 -0.3428
-0.3428
-0.5771 -0.3330 -0.5820 -0.5820 -0.3330
-0.3330
-0.5693 -0.3242 -0.5742 -0.5742 -0.3242
-0.3242
-0.5605 -0.3145 -0.5654 -0.5654 -0.3145
-0.3145
-0.5527 -0.3057 -0.5576 -0.5576 -0.3057
-0.3057
-0.5449 -0.2969 -0.5488 -0.5488 -0.2969
-0.2969
...
-0.1787 -0.0322 -0.1807 -0.1807 -0.0322
-0.0322
-0.1709 -0.0293 -0.1719 -0.1719 -0.0293
-0.0293
...
-0.0566 -0.0029 -0.0576 -0.0576 -0.0029
-0.0029
-0.0488 -0.0020 -0.0488 -0.0488 -0.0020
-0.0020
-0.0410 -0.0020 -0.0410 -0.0410 -0.0020
-0.0020
-0.0322 -0.0010 -0.0332 -0.0332 -0.0010
-0.0010
-0.0244 -0.0010 -0.0244 -0.0244 -0.0010
-0.0010
-0.0166 0.0000 -0.0166 -0.0166 0.0000
0.0000
-0.0078 0.0000 -0.0078 -0.0078 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
...
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000

```

```

0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000
...
0.2031 0.0410 0.2441 0.1836 0.0410
0.0410
0.2109 0.0449 0.2559 0.1914 0.0449
0.0449
0.2197 0.0479 0.2666 0.2002 0.0479
0.0479
0.2275 0.0518 0.2773 0.2080 0.0518
0.0518
...
0.3984 0.1592 0.5107 0.3838 0.1592
0.1592
0.4063 0.1650 0.5225 0.3916 0.1650
0.1650
0.4150 0.1719 0.5332 0.4004 0.1719
0.1719
...
0.6182 0.3818 0.8115 0.6084 0.3818
0.3818
0.6260 0.3916 0.8223 0.6162 0.3916
0.3916
0.6338 0.4023 0.8330 0.6250 0.4023
0.4023
0.6426 0.4121 0.8447 0.6338 0.4121
0.4121
0.6504 0.4229 0.8555 0.6416 0.4229
0.4229
0.6582 0.4336 0.8662 0.6504 0.4336
0.4336
...
0.6670 0.4443 0.8779 0.6582 0.4443
0.4443
0.6748 0.4551 0.8887 0.6670 0.4551
0.4551
0.6826 0.4668 0.9004 0.6748 0.4668
0.4668
...
0.9834 0.9678 1.0000 0.9834 0.9678

```

```

0.9678
0.9922 0.9834 1.0000 0.9912 0.9834
0.9834
1.0000 1.0000 1.0000 1.0000 1.0000
1.0000

```

HLP? (Get List Of Available Commands)

Description:	Lists a help string which contains all commands available.
Format:	HLP?
Arguments:	none
Response:	List of commands available
Troubleshooting:	Communication breakdown

HLT (Halt Motion Smoothly)

Description:	<p>Halts the motion of given axes smoothly. For further details, see the notes below.</p> <p>Error code 10 is set.</p> <p>#24 (p. 145) and STP (p. 228) in contrast abort current motion as fast as possible for the controller without taking care of maximum velocity and acceleration.</p>
Format:	HLT [{<AxisID>}]
Arguments:	<AxisID>: is one axis of the controller, if omitted all axes are halted
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	<p>HLT stops motion with given system deceleration with regard to system inertia. Does not apply to trajectories.</p> <p>HLT stops all motion caused by motion commands (e.g., MOV (p. 209), MVR (p. 212), MVE (p. 211), GOH (p. 172), STE (p. 227), SMO (p. 221)), trajectory execution (TGS (p. 235)), the command for reference point definition (FRF (p. 171)), and macros (MAC (p. 203)).</p>

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

HPA? (Get List Of Available Parameters)

Description: Responds with a help string which contains all available parameters with short descriptions. For further information, see "Parameter Overview" (p. 280).

Format: HPA?

Arguments: None

Response {<PamID>=" "<string> LF}

where

<PamID> is the ID of one parameter, hexadecimal format

<string> is a string which describes the corresponding parameter.

The string has the following format:

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

where

<CmdLevel> is the command level which allows write access to the parameter value.

<MaxItem> is the maximum number of items of the same type which are affected by the parameter. With the C-884, an "item" is an axis or the entire system.

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

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

<ParameterDescription> is the parameter name.

<PossibleValue> is one value from the allowed data range.

<ValueDescription> is the meaning of the corresponding value.

The parameters listed with HPA? can be changed and/or saved using the following commands:

SPA (p. 223) influences the parameter settings in volatile memory (RAM).

WPA (p. 247) copies parameter settings from volatile to nonvolatile memory.

SEP (p. 219) writes parameter settings directly into nonvolatile memory (without changing settings in volatile memory).

RPA (p. 216) resets volatile memory to the values from nonvolatile memory.

HPV? (Get Parameter Value Description)

Description: Responds with a help string which contains possible parameters values. Use HPA? instead to get a help string which contains all available parameters with short descriptions.

Format: HPV?

Arguments: None

Response: <string> has the following format:

```
"#Possible parameter values are:
{<PamID> <ItemID> "=" <ListType>
[ {TAB <PossibleValue> "=" <ValueDescription>} ] }
#CCL levels are:
{<PamID> <ItemID> "="<CmdLevel> }
#HPA_Category enabled
end of help"
```

where

<PamID> is the ID of one parameter, hexadecimal format

<ItemID> is one item (axis, channel, whole system) of the controller, if item=0 the description is valid for all items

<ListType> determines how the possible parameter values listed in the string have to be interpreted:

- 0 = parameter not applicable for this item
- 1 = enumeration
- 2 = min/max

<PossibleValue> is a value from the permissible data range

<ValueDescription> is the meaning of the corresponding value

Some parameters are write protected (by a command level > 1) for certain items. These parameters are listed below the "#CCL levels are" line.

<CmdLevel> is the command level that allows write access to the parameter value.

The "#HPA_Category enabled" line is evaluated by the PC software for display purposes.

IFC (Set Interface Parameters Temporarily)

Description: Configures interface parameters.

The new settings are enabled after IFC is sent. The PC interface configuration may have to be modified (closing of the current connection and re-opening with the new settings necessary).

Settings made with IFC are lost when the C-884 is switched off or rebooted. The defaults can be changed in the nonvolatile memory with the IFS command (p. 198).

Format: IFC {<InterfacePam> <PamValue>}

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

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

The following interface parameters can be set:

RSBAUD

<PamValue> indicates the baud rate to be used for RS-232

communication. Possible values are 9600, 19200, 38400, 57600 and 115200.

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 192.168.0.75:50000. Port 50000 cannot be changed.

Note: The C-884 will use the address given by IPADR only if IPSTART = 0 (see below).

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> specifies the subnet mask used for TCP/IP communication in the format uint.uint.uint.uint. Default is 255.255.255.0.

IPMAXCONN

<PamValue> specifies the number of permissible simultaneous TCP/IP connections to the C-884. The default setting is 1.

Response: None

Notes: The interface parameters can be changed in the nonvolatile memory with the IFS command (p. 198).

For more information, see "Establishing Communication via the TCP/IP Interface" (p. 62).

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 RSBAUD, IPADR, IPSTART, IPMASK or MACADR.

For <InterfacePam> = RSBAUD, the <PamValue> indicates the current baud rate of the RS-232 communication.

For <InterfacePam> = IPSTART, the <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, the <PamValue> indicates the currently set number of permissible simultaneous TCP/IP connections to the C-884.

For <InterfacePam> = MACADR, the <PamValue> indicates the unchangeable, unique address of the network hardware in the C-884.

IFS (Set Interface Parameters as Default Values)

Description: Stores interface parameters.

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

Format:	IFS <Pswd> {<InterfacePam> <PamValue>}
Arguments:	<p><Pswd> is the password for writing to the nonvolatile memory, default is "100"</p> <p><InterfacePam> is the interface parameter to be changed, see below</p> <p><PamValue> gives the value of the interface parameter, see below</p>
Response:	<p>None</p> <p>The following interface parameters can be set:</p> <p>RSBAUD <PamValue> indicates the baud rate to be used for RS-232 communication. Possible values are 9600, 19200, 38400, 57600 and 115200. Default is 115200.</p> <p>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 192.168.0.75:50000. Port 50000 cannot be changed. Note: The C-884 will use the address given by IPADR only if IPSTART = 0 (see below).</p> <p>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).</p> <p>IPMASK <PamValue> specifies the subnet mask used for TCP/IP communication in the format uint.uint.uint.uint. Default is 255.255.255.0.</p> <p>IPMAXCONN <PamValue> specifies the number of permissible simultaneous TCP/IP connections to the C-884. The default setting is 1.</p>
Response:	None

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

The interface parameters can be changed in the volatile memory with the IFC command (p. 196). Changes in the volatile memory become effective immediately.

For more information, see "Establishing Communication via the TCP/IP Interface" (p. 62).

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 RSBAUD, IPADR, IPSTART or IPMASK.

For <InterfacePam> = RSBAUD, <PamValue> indicates the baud rate for RS-232 communication.

For <InterfacePam> = IPSTART, <PamValue> indicates the setting of 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).

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

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

the format uint.uint.uint.uint.

For <InterfacePam> = IPMAXCONN, <PamValue> indicates the number of permissible simultaneous TCP/IP connections to the C-884.

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. 245). 1 jumps to the next line, making the command unnecessary, and 2 jumps over the next command. Only jumps within the current macro are allowed.

<CMD?> is one query command in its usual notation. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= <= < > >= !=

Important: There must be a blank space before and after the operator!

<Value> is the value to be compared with the response to <CMD?>.

Response: None

Troubleshooting: Check proper jump target

Example: Using the following macro, you can stop motion of axis "1" using a stop button connected to a digital input. The stop button is checked until the axis has reached the target position (query ONT?). When the stop button is pressed before the target position has been reached: The response to the POS? query is copied into the TARGET variable. This variable is then used as second argument for the MOV

command. Therefore, the positioner just stays where it was. To clean up, TARGET is defined as empty with the VAR command which deletes the variable.

Write the "stop" macro:

```
MAC BEG stop
MOV 1 20
JRC 2 DIO? 1 = 1
JRC -1 ONT? 1 = 0
CPY TARGET POS? 1
MOV 1 ${TARGET}
VAR TARGET
MAC END
```

LIM? (Indicate Limit Switches)

Description: Gets whether axes have limit switches.

Format: LIM? [{<AxisID>}]

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

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

where

<uint> indicates whether the axis has limit switches (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The C-884 firmware detects the presence or absence of limit switches using a parameter (ID 0x32). According to the value of this parameter, the C-884 enables or disables the stopping of the motion at the limit switches. Adapt the parameter value to your hardware using SPA (p. 223) or SEP (p. 219). For further information, see "Limit Switch Detection" (p. 33). You can use the digital input lines instead of the limit switches as source of the negative or positive limit switch signal. Further information see "Digital Input Signals" (p. 101).

MAC (Call Macro Function)

Description:	Calls a macro function. Permits recording, deleting and running macros on the controller.
Format:	<p>MAC <keyword> {<parameter>}</p> <p>in particular:</p> <p>MAC BEG <macroname> MAC DEF <macroname> MAC DEF? MAC DEL <macroname> MAC END MAC ERR? MAC NSTART <macroname> <uint> [<String1> [<String2> [<String3> [<String4>]]] MAC START <macroname> [<String1> [<String2> [<String3> [<String4>]]]</p>
Arguments:	<p><keyword> determines which macro function is called. The following keywords and parameters are used:</p> <p>MAC BEG <macroname> Starts recording a macro to be named <i>macroname</i> on the controller; may not be used in a macro; the commands that follow become the macro. End the recording with MAC END. Note that erroneous macro content cannot be detected by sending the ERR? command.</p> <p>MAC END Stops macro recording (cannot become part of a macro)</p> <p>MAC ERR? Reports the last error which occurred during macro execution.</p> <p>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.</p> <p>MAC DEF <macroname> Sets specified macro as startup macro. This macro will be automatically executed after the next switch-on or reboot of the controller. If <macroname> is omitted, the current startup macro selection is canceled.</p> <p>MAC DEF? Asks for the startup macro Response: <macroname></p>

If no startup macro is defined, the response is an empty string with the terminating character.

MAC DEL <macroname>

Deletes specified macro.

MAC NSTART <macroname> <uint> [<String1> [<String2> [<String3> [<String4>]]]]

Repeats the specified macro <uint> times. Another execution is started when the last one is finished.

<STRING1> to <STRING4> are optional arguments which give the values for the local variables 1 to 4 used in the specified macro. <STRING1> to <STRING4> can be given directly or via the values of variables. The macro execution will fail if the macro contains local variables but <STRING1> and the other required arguments are omitted from the MAC NSTART command. See "Variables" (p. 135) for further details.

MAC START <macroname> [<String1> [<String2> [<String3> [<String4>]]]]

Starts one execution of the specified macro. <String1> to <String4> have the same function as with MAC NSTART.

Response: None

Troubleshooting: Macro recording is active (keywords BEG, DEL) or inactive (END)
Macro contains a disallowed MAC command

Notes: During macro recording no macro execution is allowed.

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

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

Macros can contain local and global variables. For more information, see "Variables" (p. 135).

A running macro sends no responses to any interface.

Depending on the value of parameter 0x72 (**Ignore Macro Error?**), the following possibilities exist when an error is caused by the running macro:

0 = Macro execution is aborted (default).

1 = The error is ignored and the macro execution is continued.

Irrespective of the parameter setting, MAC ERR? always reports the last error that occurred during a macro execution.

The following commands provided by the C-884 can only be used in macros:

DEL (p. 159), JRC (p. 201), MEX (p. 207) and WAC (p. 245).

A macro can start another macro. The maximum number of nesting levels is 5. A macro can call itself to form an infinite loop.

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

Macro execution can be stopped with #24 (p. 145) and STP (p. 228).

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

A running macro may not be deleted.

You can query with #8 (p. 144) if a macro is currently running on the controller.

Note: The number of write cycles in the nonvolatile memory is limited. Only record macros if it is necessary.

MAC? (List Macros)

Description:	Lists macros or content of a given macro.
Format:	MAC? [<macroname>]
Arguments	<macroname>: name of the macro whose content shall be listed; if omitted, the names of all stored macros are listed.
Response:	<string>
	If <macroname> was given, <string> is the content of this macro;
	If <macroname> was omitted, <string> is a list with the

names of all stored macros

Troubleshooting: Macro <macroname> not found

MAN? (Get Help String For Command)

Description: Shows a detailed help text for individual commands.

Format: MAN? <CMD>

Arguments: <CMD> is the command mnemonic of the command for which the help text is to be displayed (see below).

Response: A string that describes the command.

Notes: A detailed help text can be displayed for the following commands:
CTO, CTO?, HIA, HIA?, HIS, HIS?, HIT, HIT?, IFC, IFC?, WPA

Example: Send: MAN? CTO
Receive:
CTO {<TrigOutID> <CTOPam> <Value>} Set
Configuration Of Trigger Output
#AvailableCTOparameters
<CTOPam> <Description>
<CTOPam> (configuration parameter):
1 TriggerStep
2 Axis
3 TriggerMode
7 Polarity
8 StartThreshold
9 StopThreshold
10 TriggerPosition
11 PulseWidth
#AvailableTriggerModes
<Value> <Description>
0 PositionDistance
2 OnTarget
5 MotionError
6 InMotion
7 Position+Offset
8 SinglePosition
9 HardwareTrigger
#AvailablePolarities

```
<Value> <Description>
0 ActiveLow
1 ActiveHigh
end of help
```

MAT (Calculate And Save To Variable)

Description: Carries out a mathematical operation or bit operation and saves the result as a variable (p. 135).

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

Format: MAT <Variable> "=" <FLOAT1> <OP> <FLOAT2>

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

<FLOAT1> and <FLOAT2> are the values from which the result is to be calculated. They can be given directly or via the value of a variable.

<OP> is the operator to be used: The following operators are possible:

<OP>	Operation	Type
+	Addition	Mathematical operation
-	Subtraction	Mathematical operation
*	Multiplication	Mathematical operation
AND	UND	Bit operation
OR	ODER	Bit operation
XOR	XOR	Bit operation

Important: There must be a blank space before and after each "=" and the operator!

Response: None

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. 245).

Format: MEX <CMD?> <OP> <Value>

Arguments <CMD?> is one query command in its usual syntax. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= < > >= !=

Important: There must be a blank space before and after the operator!

<Value> is the value that is compared with the response to <CMD?>.

Response: None

Example: Send: MAC START LOOP

Note:

LOOP macro contains the following:

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

KEY1 macro contains the following:

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

KEY2 macro contains the following:

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

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

KEY1 checks the state of digital input line 1 (located on the I/O socket (p. 309)). If it is not set (0), the macro is aborted, otherwise the macro will move axis 1 by 1.0 in the positive direction (relative motion).

KEY2 checks the state of digital input line 2 and moves axis 1 in the negative direction accordingly.

By connecting digital input lines 1, 2 and 4 with pushbuttons, it is possible to implement the interactive control of an axis without any software assistance. The delay (DEL 100) is required to avoid generation of multiple MVR commands while pressing the pushbutton for a short time.

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

MOV (Set Target Position)

Description: Sets an absolute target position for the given axis.

Format: MOV {<AxisID> <Position>}

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

<Position> is the absolute target position in physical units.

Response: none

Notes: The servo mode must be switched on when this command is used (closed-loop operation).

The target position must be within the soft limits. Use TMN? (p. 239) and TMX? (p. 239) to query the current valid soft limits.

The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 25).

The motion can be stopped by #24 (p. 145), STP (p. 228), and HLT (p. 193).

During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: Motion commands can be

sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other.

The execution of motion commands can be configured with the **Inhibit Motion Commands** parameter (ID 0x130); for more information, see "Motion Triggering" (p. 23).

Example 1: Send: `MOV 1 10`
Note: Axis 1 moves to 10 (target position in mm)

Example 2: Send: `MOV 1 243`
Send: `ERR?`
Receive: `7`
Note: The axis does not move. The error code "7" in the reply to the ERR? command (p. 168) indicates that the target position given in the motion command is out of limits.

MOV? (Get Target Position)

Description: Returns last valid commanded target position.

Format: `MOV? [{<AxisID>}]`

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the last commanded target position in physical units

Troubleshooting: Illegal axis identifier

Notes: The target position can be changed by commands that cause motion (e.g., MOV (p. 209), MVR (p. 212), MVE (p. 211), GOH (p. 227), STE (p. 172)) or by HID control (when HID control is disabled, the target position is set to the current position for HID-controlled axes in closed-loop operation).

MOV? gets the commanded positions. Use POS? (p. 214) to get the current positions.

MVE (Set Target Position For Vectorial Move)

Description:	<p>Sets new absolute target positions for given axes. This command is intended for the simultaneous "vectorial" motion of coupled axes that results in a straight line.</p> <p>Servo mode must be switched on for all commanded axes prior to using this command (closed-loop operation). When the servo mode is switched off or a motion error occurs during the motion, all axes are stopped.</p>
Format:	MVE {<AxisID> <Position>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><Position> is the new absolute target position in physical units.</p>
Response:	None
Notes:	<p>The axes move along a "vectorial path". Velocities, accelerations and decelerations will be calculated to ensure that all axes follow the path. The current settings for velocity, acceleration and deceleration define the maximum possible values, and the slowest axis determines the resulting velocities. Further information, see "Generation of Dynamics Profile" (p. 25).</p> <p>The target position must be within the soft limits. Use TMN? (p. 239) and TMX? (p. 239) to get the currently valid soft limits.</p> <p>The motion can be stopped by #24 (p. 145), STP (p. 228) and HLT (p. 193).</p> <p>No other motion commands (e. g. MOV, MVR, MVE) are allowed during vectorial motion.</p> <p>Motion commands such as MVE are not allowed when the HID control is enabled for the axis. Further information see "Control with HID Device" (p. 106).</p> <p>The execution of motion commands can be configured with the Inhibit Motion Commands parameter (ID 0x130); for more information, see "Motion Triggering" (p. 23).</p>

MVR (Set Target Relative To Current Position)

Description: Moves the given axis relative to the last commanded target position.

Format: MVR {<AxisID> <Distance>}

Arguments: <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

Notes: The servo mode must be switched on when this command is used (closed-loop operation).

The target position must be within the soft limits. Use TMN? (p. 239) and TMX? (p. 239) to get the currently valid soft limits, and MOV? (p. 210) to get the current target.

The motion is executed as a point-to-point motion with the dynamics profile generated by the profile generator (p. 25).

The motion can be stopped by #24 (p. 145), STP (p. 228), and HLT (p. 193).

During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: Motion commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other.

The execution of motion commands can be configured with the **Inhibit Motion Commands** parameter (ID 0x130); for more information, see "Motion Triggering" (p. 23).

Example:

Send: MOV 1 0.5

Note: This is an absolute motion.

Send: POS? 1

Receive: 1=0.500000

Send: MOV? 1

Receive: 1=0.500000

Send: MVR 1 2

Note: This is a relative motion.

```

Send: POS? 1
Receive: 1=2.500000
Send: MVR 1 2000
Note: New target position of axis 1 would exceed motion
range. Command is ignored, i.e., the target position
remains unchanged, and the axis does not move.
Send: MOV? 1
Receive: 1=2.500000
Send: POS? 1
Receive: 1=2.500000

```

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

Notes: The detection of the on-target state is only possible in closed-loop operation (servo mode ON).

The on-target state is influenced by the settings for the settling window (parameter 0x36) and the delay time (parameter 0x3F). Details see "On-Target State" (p. 31).

POS (Set Real Position)

Description: Sets the current position of the axis (does not cause motion).

Format: POS {<AxisID> <Position>}

Arguments:	<p><AxisID> is one axis of the controller.</p> <p><Position> is the new current position in physical units.</p>
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>It is only possible to set the current position with POS when the mode of reference point definition is set to "0", see RON (p. 215).</p> <p>An axis is considered to be "referenced" when the position has been set with POS (for more information, see "Reference Point Definition" (p. 38)).</p> <p>The minimum and maximum commandable positions (TMN? (p. 239), TMX? (p. 239)) are not adapted when a position is set with POS. This can result in target positions which are allowed by the C-884 but cannot be reached by the hardware. Target positions are also possible that can be reached by the hardware but are refused by the C-884. Furthermore, the zero position can be outside of the physical travel range after using POS.</p>

POS? (Get Real Position)

Description:	<p>Gets the current axis position.</p> <p>If all arguments are omitted, the current position of all axes is queried.</p>
Format:	POS? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.
Response:	<p>{<AxisID>="<float> LF}</p> <p>where</p> <p><float> is the current axis position in physical units.</p>
Troubleshooting:	Illegal axis identifier

RBT (Reboot System)

Description:	Reboots system. The controller behaves the same as after switching on.
Format:	RBT
Arguments:	none
Response:	none
Notes:	RBT cannot be used in macros. This is to avoid problems with startup macro execution.

RMC? (List Running Macros)

Description:	Lists macros which are currently running.
Format:	RMC?
Arguments:	None
Response:	{<macroname> LF}
	where
	<macroname> is the name of one macro which is saved on the controller and currently running. The response is an empty line when no macro is running.

RON (Set Reference Mode)

Description:	Sets mode of reference point definition of given axes.
Format:	RON {<AxisID> <ReferenceOn>}
Arguments:	<AxisID> is one axis of the controller. <ReferenceOn> can be 0 or 1. 1 is default. Details see below.
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<ReferenceOn> = 0: To define the reference point of the axis, an absolute position value can be assigned with POS (p. 213) or a reference move can be started with FRF (p.

171). Relative motion is possible with MVR (p. 212), even when the reference point for the axis has not been defined yet.

<ReferenceOn> = 1: For reference point definition of the axis, a reference move must be started with FRF. Using POS is not allowed. Motion in closed-loop operation is only possible when the reference point for the axis has been defined.

Further information, see "Reference Point Definition" (p. 38).

RON? (Get Reference Mode)

Description: Gets mode of reference point definition of given axes.

Format: RON? [{ <AxisID>}]

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

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

where

<ReferenceOn> is the currently set mode of reference point definition for the axis

Troubleshooting: Illegal axis identifier

Note: Further information can be found in the description of the RON command (p. 215).

RPA (Reset Volatile Memory Parameters)

Description: Resets the given parameter of the given item. The value from nonvolatile memory is written into volatile memory.

Related commands:

With HPA? (p. 194) you can obtain a list of the available parameters. SPA (p. 223) influences the parameter settings in volatile memory, WPA (p. 247) writes parameter settings from volatile to nonvolatile memory, and SEP (p. 219) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).

See SPA for an example.

Format:	RPA [{<ItemID> <PamID>}]
Arguments:	<p><ItemID> is the item for which a parameter is to be reset. See below for details.</p> <p><PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.</p>
Response:	none
Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	<p>The information from the positioner's ID chip and the positioner databases are loaded into the volatile memory of the C-884. The loaded data is overwritten by RPA. Only use RPA if you are sure that the C-884 functions correctly with the parameter values from the nonvolatile memory.</p> <p>With RPA, you can reset either all parameters or specific individual parameters for the C-884.</p> <p>DPA resets parameter values and parameter-independent settings to the default settings.</p>
Available item IDs and parameter IDs:	<p>An item is an axis (the identifier can be changed with SAI (p. 218)) or the entire system. For further information, see "Commandable Items" (p. 18).</p> <p>Valid parameter IDs are given in "Parameter Overview" (p. 280).</p>

RTR (Set Record Table Rate)

Description:	Sets the record table rate, i.e., the number of cycles to be used in data recording operations. Settings larger than 1 make it possible to cover longer time periods.
Format:	RTR <RecordTableRate>
Arguments:	<RecordTableRate> is the record table rate to be used for recording operations (unit: number of cycles), must be an integer value larger than zero.
Response:	None

Notes: The duration of the recording can be calculated as follows:

Rec. duration = cycle time of the servo loop * RTR value *
number of points

where

the cycle time of the servo loop for the C-884 is 100 μ s

the number of points for the C-884 is 8192 (length of data
recorder table)

For further information, see "Data Recorder" (p. 89).

The record table rate set with RTR is saved in volatile
memory (RAM) only.

RTR? (Get Record Table Rate)

Description: Gets the current record table rate, i.e., the number of
cycles used in data recording operations.

Format: RTR?

Arguments: None

Response: <RecordTableRate> is the table rate used for recording
operations (unit: number of cycles).

SAI (Set Current Axis Identifiers)

Description: Sets the axis identifiers for the given axes.

After it was set with SAI, the new axis identifier must be
used as <AxisID> in all axis-related commands.

Format: SAI {<AxisID> <NewIdentifier>}

Arguments: <AxisID> is one axis of the controller

<NewIdentifier> is the new identifier to use for the axis,
see below for details

Response: none

Notes: An axis could be identified with up to 8 characters. Use TVI? (p. 242) to ask for valid characters.
The new axis identifier is saved automatically and is therefore still available after rebooting or switching on the next time.

SAI? (Get List Of Current Axis Identifiers)

Description: Gets the axis identifiers.

See also "Commandable Items" (p. 18).

Format: SAI? [ALL]

Arguments: [ALL] is optional. For controllers which allow for axis deactivation, [ALL] ensures that the response also includes the axes which are "deactivated".

Response: {<AxisID> LF}

<AxisID> is one axis of the controller.

Notes: When the **Stage Name** parameter (0x3C) has the value "NOSTAGE", the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e. g. motion commands or position queries) and is only included in the response to SAI? ALL.

SEP (Set Non-Volatile Memory Parameters)

Description: Sets a parameter of a given item to a different value in nonvolatile memory, where it becomes the new default.

After parameters were set with SEP, you can use RPA (p. 216) to activate them (write them to volatile memory) without controller reboot.

Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Related commands:

HPA? (p. 194) returns a list of the available parameters.

SPA (p. 223) writes parameter settings into volatile memory (without changing the settings in nonvolatile memory).

WPA (p. 247) writes parameter settings from volatile to nonvolatile memory.

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

Arguments

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

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

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

<PamValue> is the value to which the given parameter of the given item is set.

Response: None

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

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

With the C-884, you can write up to four (C-884.4DC) or six (C-884.6DC) parameters per SEP command.

Available item IDs and parameter IDs: An item is an axis (the identifier can be changed with SAI (p. 218)) or the entire system. For further information, see "Commandable Items" (p. 18).

Valid parameter IDs are given in "Parameter Overview" (p. 280).

SEP? (Get Nonvolatile Memory Parameters)

Description:	Gets the value of a parameter of a given item from nonvolatile memory. With HPA? (p. 194) you can obtain a list of the available parameters and their IDs.
Format:	SEP? [{<ItemID> <PamID>}]
Arguments:	<ItemID> is the element for which a parameter value from nonvolatile memory is to be queried. See below for details. <PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.
Response:	{<ItemID> <PamID>="<PamValue> LF} where <PamValue> is the value of the specified parameter for the specified element
Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	With the C-884, you can query either all parameters or specific individual parameters with each SEP? command.
Available item IDs and parameter IDs:	An item is an axis (the identifier can be changed with SAI (p. 218)) or the entire system. For further information, see "Commandable Items" (p. 18). Valid parameter IDs are given in "Parameter Overview" (p. 280).

SMO (Set Open-Loop Control Value)

Description:	Sets control value directly to move the axis. Profile generator (if present), sensor feedback and servo algorithm are not taken into account. Servo mode must be switched off when using this command (open-loop operation).
--------------	---

Format:	SMO {<AxisID> <ControlValue>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><ControlValue> is the new control value (dimensionless). See below for details.</p>
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>Servo mode is switched on for one of the specified axes.</p> <p>NOTICE: In the case of large control values, the positioner can hit the hard stop despite the limit switch function. This can cause damage to equipment.</p> <p>The unsigned control value may not be greater than the value of the Maximum Motor Output parameter (0x9). When this parameter is set to its maximum (32767), <ControlValue> ranges from -32766 to 32766 (dimensionless). <ControlValue> controls the PWM signal for the axis. The sign of the value determines the direction of motion: -32766 corresponds to the maximum output voltage in negative direction of motion and 32766 to the maximum output voltage in positive direction of motion.</p> <p>The Range Limit Min (0x07000000) and Range Limit Max (0x07000001) parameters can be used as soft limits for motion in open-loop operation with SMO: When the current position reaches these values, the control value is set to zero and the motion is stopped. The axis can be moved again as soon as the value for the soft limit has been increased or decreased.</p> <p>The execution of motion commands can be configured with the Inhibit Motion Commands (ID 0x130) parameter; for more information, see "Motion Triggering" (p. 23).</p>
Example:	<p>Send: SMO 1 -16000</p> <p>Note: The control value is about half the maximum control value. The axis moves in negative direction.</p>

SMO? (Get Control Value)

Description: Gets last valid control value of given axis.

Format:	SMO? [{<AxisID>}]
Arguments	<AxisID> is one axis of the controller
Response:	{<AxisID>="<float> LF}
	where
	<float> is the last valid control value (dimensionless). For details see below.
Troubleshooting:	Illegal axis identifier
Notes:	<p>The control value which is returned by SMO? can be the result of the servo algorithm and other corrections, or it can be the value set by an SMO command (p. 221) in open-loop operation. See the block diagram (p. 18) for further information.</p> <p>The control value ranges from -32766 to 32766 (dimensionless) and controls the PWM signal for the axis. The sign of the value determines the direction of motion: -32766 corresponds to the maximum output voltage in negative direction of motion and 32766 to the maximum output voltage in positive direction of motion.</p>

SPA (Set Volatile Memory Parameters)

Description:	Sets a parameter of the given item in the volatile memory (RAM) to a specific value. Parameter changes are lost when the controller is switched off or rebooted.
Format:	SPA {<ItemID> <PamID> <PamValue>}
Arguments:	<p><ItemID> is the element for which a parameter is changed in volatile memory. See below for details.</p> <p><PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.</p> <p><PamValue> is the value to which the specified parameter of the specified element is set.</p>
Response:	<p>None</p> <p>Parameter changes are also lost when the parameters are reset to their default values with RPA (p. 216).</p>

Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Related commands:

HPA? (p. 194) returns a list of the available parameters.

SEP (p. 219) writes parameter settings directly into non-volatile memory (without changing the settings in volatile memory).

WPA (p. 247) writes parameter settings from volatile to non-volatile memory.

RPA resets volatile memory to the value in non-volatile memory.

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

Notes: With the C-884, you can write up to four (C-884.4DC) or six (C-884.6DC) parameters per SPA command.

Available item IDs and parameter IDs: An item is an axis (the identifier can be changed with SAI (p. 218)) or the entire system. For further information, see "Commandable Items" (p. 18).

Valid parameter IDs can be found in the parameter overview (p. 280).

Example 1: Send: SPA 1 0x1 100

Note: Sets the P term of the servo algorithm for axis 1 to 100; the parameter ID is written in hexadecimal format

Send: SPA 1 1 150

Note: Sets the P term of the servo algorithm for axis 1 to 150; the parameter ID is written in decimal format

Example 2: The P, I and D parameters of the servo algorithm must be adapted to a new load applied to the mechanical system connected.

Send: SPA 1 0x1 150 1 0x2 200 1 0x3 100

Note:

The P term is set to 150 for axis 1.
 The I term is set to 200 for axis 1.
 The D term is set to 100 for axis 1.
 These settings are only made in the volatile memory.

Now check the function of the system. If the closed-loop system performance proves satisfactory and you want to use this system configuration as default, save the parameter settings from volatile to nonvolatile memory.

Send: `WPA 101`

Note: See the command description for WPA (p. 247) for details on the extent of the saved settings.

SPA? (Get Volatile Memory Parameters)

Description:	Gets the value of a parameter of a specified element from volatile memory (RAM).
	With HPA? (p. 194) you can obtain a list of the available parameters.
Format:	SPA? [{<ItemID> <PamID>}]
Arguments:	<p><ItemID> is the element for which a parameter is to be queried in volatile memory. See below for details.</p> <p><PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details.</p>
Response:	<p>{<ItemID> <PamID>="<PamValue> LF}</p> <p>where</p> <p><PamValue> is the value of the specified parameter for the specified element</p>
Troubleshooting:	Illegal element identifier, wrong parameter ID
Notes:	With the C-884, you can either query all parameters or up to four (C-884.4DC) or six (C-884.6DC) individual parameters per SPA? command.

Available item IDs and parameter IDs: An item is an axis; the identifier can be changed with SAI (p. 218). For further information, see "Commandable Items" (p. 18).

Valid parameter IDs can be found in the parameter overview (p. 280).

SRG? (Query Status Register Value)

Description: Returns register values for queried items and registers.

Format: SRG? {<ItemID> <RegisterID>}

Arguments: <ItemID> is the item for which a register is to be queried. See below for details.

<RegisterID> is the ID of the specified register; for available registers, see below.

Response: {<ItemID><RegisterID>="<Value> LF}

where

<Value> is the value of the register; for more details, see below.

Note: This command is identical in function to #4 (p. 142) which should be preferred when the controller is performing time-consuming tasks.

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

<RegisterID> can be 1.

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

Bit	15	14	13	12	11	10	9	8
Description	On-target state	Determines the reference value	In motion	Servo mode on	-	-	-	Error flag

Bit	7	6	5	4	3	2	1	0
Description	Digital input line 4	Digital input line 3	Digital input line 2	Digital input line 1	-	Positive limit switch	Reference point switch	Negative limit switch

Example:

Send: `SRG? 1 1`

Receive: `1 1=0x9002`

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

STE (Start Step And Response Measurement)

Description: Starts performing a step and recording the step 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 (p. 163).

The recorded data can be read with the DRR? command (p. 165).

Format: STE <AxisID> <Amplitude>

Arguments: <AxisID> is one axis of the controller

<Amplitude> is the size of the step. See below for details.

Response: None

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

The target position must be within the soft limits. Use TMN? (p. 239) and TMX? (p. 239) to get the currently valid soft limits, and MOV? (p. 210) to get the current target.

Motion commands such as STE are not allowed when the HID control is enabled for the axis. Further information see "Control with HID Device" (p. 106).

The execution of motion commands can be configured with the **Inhibit Motion Commands** (ID 0x130) parameter; for more information, see "Motion Triggering" (p. 23).

Notes: A "step" consists of a motion with the specified amplitude which is performed relative to the current position.

STP (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 #24 (p. 145).

Format: STP

Arguments: None

Response: None

Troubleshooting: Communication breakdown

Notes: STP stops all motion caused by motion commands (e.g., MOV (p. 209), MVR (p. 212), MVE (p. 211), GOH (p. 172), STE (p. 227), SMO (p. 221)), trajectory execution (TGS (p. 235)), the command for reference point definition (FRF (p. 171)), and macros (MAC (p. 203)). Also stops execution of macros but not execution of Python scripts.

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

HLT (p. 193) in contrast to STP stops motion with given system deceleration with regard to system inertia. Does not apply to trajectories.

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>When switching from open-loop to closed-loop operation, the target is set to the current position to avoid jumps of the mechanics.</p> <p>The current state of the servo mode determines the applicable motion commands: Servo mode ON: Use MOV (p. 209), MVR (p. 212), MVE (p. 211), GOH (p. 172) (for point-to-point motion), TGS (p. 235) (for trajectory execution) or HID control (p. 106). Servo mode OFF: Use SMO (p. 221).</p> <p>Servo mode must be switched on before reference moves can be started with FRF (p. 171).</p> <p>When the servo mode is switched off while the axis is moving, the axis stops.</p> <p>Using a startup macro, you can configure the controller so that servo mode is automatically switched on when switching on or rebooting. For further information, see "Setting up a Startup Macro" (p. 125).</p> <p>If the axis has a brake, setting the servo mode with SVO influences the activation state of the brake:</p> <ul style="list-style-type: none"> ▪ Switching on the servo mode deactivates the brake. ▪ Switching off the servo mode activates the brake. When the servo mode is switched off, the brake can be activated or deactivated with BRA (p. 149). Secure the positioner against unintentional motion before you deactivate the brake with BRA! <p>If a motion error occurs, the servo mode is switched off and the brake is activated. For more information see "Motion Errors".</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

TAC? (Tell Analog Channels)

Description:	Gets the number of installed analog lines.
Format:	TAC?
Arguments:	None
Response:	<uint> indicates the total number of analog lines (inputs and outputs).
Notes:	Gets the number of analog input lines located on the I/O socket (p. 309) of the C-884. Further information see "Commandable Items" (p. 18).

TAV? (Get Analog Input Voltage)

Description:	Get voltage at analog input.
Format:	TAV? [{<AnalogInputID>}]
Arguments:	<AnalogInputID> is the identifier of the analog input channel; see below for details.

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

where

<float> is the current voltage at the analog input in volts

Notes: Using the TAV? command, you can directly read the analog input lines 1 to 4 on the **I/O** socket (p. 309) of the C-884. The identifiers of the lines are 1 to 4. See "Commandable Items" (p. 18) for more information.

You can record the values of the analog input lines using the DRC record option 81 (p. 163).

TCV? (Get Commanded Closed-Loop Velocity)

Description: Queries the current value of the velocity (value calculated by the profile generator).

Format: TCV? [{<AxisID>}]

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

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

where

<float> is the velocity value in physical units per second.

TGA (Append Value To Trajectory)

Description: Command for motion paths: Loads trajectory points to the buffer of the specified trajectory.

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

Format: TGA {<Trajectory> <Point>}

Arguments: <Trajectory>: Identifier of the trajectory
<Point>: Value in FLOAT format; indicates a trajectory point as the absolute position in physical units

Response: none

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 to axis 2
`TGA 2 3.4 1 5.6`
A trajectory point with the value 3.4 is added to the trajectory of axis 2; a trajectory point with the value 5.6 is added to the trajectory of axis 1

Troubleshooting:

- Invalid trajectory identifier
- Trajectory buffer full (> **Maximum Buffer Size**)

Notes: The timing for trajectories is set with the TGT (p. 236) command.

The TGS (p. 235) command starts the execution of a trajectory.

The TGF (p. 234) command properly completes the execution of a trajectory.

If the execution of a trajectory is cancelled after an error or stopped with STP (p. 228), #24 (p. 145), or HLT (p. 193), the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 234), deletion with TGC (p. 233)).

The C-884 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

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

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

- Therefore, pay attention to the following when working with trajectories:
 - The path that is specified by the trajectory points

must be continuously differentiable at least twice.

- During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
- When following the trajectory, an abrupt stop may not damage the load on the positioner.
- To generate the trajectory points and continuously transfer them to the C-884 during the trajectory execution, it is recommended to use a suitable program.

For further information, see "Trajectories for Motion Paths" (p. 85).

TGC (Clear All Values In Trajectory)

Description: Command for motion paths: Deletes the trajectory points in the buffer of the specified trajectory

If no argument is given, the points in the buffer of all trajectories are deleted.

Format: TGC [{<Trajectory>}]

Arguments: <Trajectory>: Identifier of the trajectory

Response: none

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 to axis 2
`TGC 2 1`
Points of the trajectories of axis 2 and axis 1 are deleted

Troubleshooting: Invalid trajectory identifier

Notes: If the execution of a trajectory is cancelled after an error or stopped with STP (p. 228), #24 (p. 145), or HLT (p. 193), the trajectory points that have not been processed by this time remain in the buffer. Therefore, before loading a new trajectory, make sure that there are no invalid trajectory points in the buffer.

The number of points in the buffer can be queried with the TGL? (p. 234) command.

For further information, see "Trajectories for Motion Paths" (p. 85).

TGF (Finalize Trajectory)

Description:	<p>Command for motion paths: Completes the execution of the specified trajectory</p> <p>TGF must be sent after the last trajectory point has been loaded. If the trajectory execution is not properly completed with TGF, an error will occur when the number of points in the buffer falls below the required minimum (4).</p> <p>For further information, see "Trajectories for Motion Paths" (p. 85).</p> <p>A trajectory will only be executed as long as there are at least 4 points in the trajectory buffer. For trajectories to be executed to the end, this command must be sent after all trajectory points have been loaded. It signals to the firmware that no more points will be supplied for the specified trajectory. In this case, the remaining trajectory points will be processed without an error occurring when the minimum number of points is no longer present.</p>
Format:	TGF [{<Trajectory>}]
Arguments:	<Trajectory>: Identifier of the trajectory
Response:	none
Example:	<p>Controller with 2 axes:</p> <p>Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2</p> <pre>TGF 2 1</pre> <p>Trajectories of axis 2 and axis 1 are completed</p>
Troubleshooting:	<ul style="list-style-type: none"> Invalid trajectory identifier Trajectory is not currently being executed

TGL? (Get Number Of Values In Trajectory)

Description:	<p>Command for motion paths: Queries the number of points in the buffer of the specified trajectory.</p> <p>If no argument is given, the points for all trajectories are queried.</p> <p>For further information, see "Trajectories for Motion Paths" (p. 85).</p>
--------------	--

Format: TGL? [{<Trajectory>}]

Arguments: <Trajectory>: Identifier of the trajectory

Response: <Trajectory>=<int>LF
where
<int> is the current number of points in the buffer of the trajectory

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2
Send: TGL? 2 1
Receive: 2=12
1=18
Trajectory buffer of axis 2 contains 12 points,
trajectory buffer of axis 1 contains 18 points

TGS (Start Trajectory)

Description: Command for motion paths: Starts the execution of the specified trajectory or trajectories

If no argument is given, the execution of all valid trajectories is started.

Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with TGA (p. 231). During the execution of a trajectory, the buffer must be refilled fast enough. The execution of a trajectory must be completed with TGF (p. 234).

Format: TGS [{<Trajectory>}]

Arguments: <Trajectory>: Identifier of the trajectory

Response: none

Example: Controller with 2 axes:
Trajectory 1 belongs to axis 1, trajectory 2 belongs to axis 2
TGS 2 1
Trajectories of axis 2 and axis 1 are started

Troubleshooting:

- Invalid trajectory identifier
- Trajectory buffer contains too few points (< 4)
- Servo mode switched off

- Notes:**
- The individual points of a trajectory are loaded to the buffer with the TGA (p. 231) command.
 - The timing for trajectories is set with the TGT (p. 236) command.
 - In addition to proper completion with TGF (p. 234), the execution of a trajectory can be stopped at any time with STP (p. 228), #24 (p. 145), or HLT (p. 193).
 - If an error occurs, the execution of all trajectories is cancelled.
 - After the execution has been stopped or cancelled, the unprocessed trajectory points remain in the buffer. Therefore, before executing a new trajectory, make sure that there are no invalid trajectory points in the buffer (query with TGL? (p. 234), deletion with TGC (p. 233)).
 - For further information, see "Trajectories for Motion Paths" (p. 85).

TGT (Set Trajectory Timing)

- Description:** Command for motion paths: Sets the timing for trajectories.
- The timing specifies the time interval at which the individual points are output during the execution of the trajectories.
- The specified value is valid for all trajectories.
- Format:** TGT <NoOfServoCycles>
- Arguments:** <NoOfServoCycles>: Time interval between the output of the individual points of a trajectory (unit: Number of servo cycles)
- Response:** None
- Notes:**
- The TGA (p. 231) command loads the points to the trajectory buffer.
 - The TGS (p. 235) command starts the execution of a trajectory.

The C-884 does **not** calculate a dynamics profile during the execution of a trajectory. After the last trajectory point has been reached, the motion of the axis is abruptly stopped. This holds true for the proper completion of trajectories as well as for their cancellation (e. g., by a stop command or error). Acceleration / deceleration, velocity, and steadiness of the motion therefore depend on the following factors during trajectory execution:

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

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

- Therefore, pay attention to the following when working with trajectories:
 - The path that is specified by the trajectory points must be continuously differentiable at least twice.
 - During the execution of the trajectory, the maximum permissible velocity and acceleration of the axis must **not** be exceeded.
 - When following the trajectory, an abrupt stop may not damage the load on the positioner.
 - To generate the trajectory points and continuously transfer them to the C-884 during the trajectory execution, it is recommended to use a suitable program.

For further information, see "Trajectories for Motion Paths" (p. 85).

TGT? (Get Trajectory Timing)

Description: Command for motion paths: Queries the timing for trajectories.

The returned value is valid for all trajectories.

For further information, see "Trajectories for Motion Paths" (p. 85).

Format: TGT?

Arguments: None

Response: <NoOfServoCycles>: Time interval between the output of the individual points of a trajectory (unit: Number of servo cycles)

TIM (Set Timer Value)

Description: Sets the timer to the given value. When the value is omitted, the timer is reset to zero.

The timer is incremented each servo cycle and can be used for time measurement. The incrementation starts at zero each time the C-884 is switched on or rebooted.

Format: TIM [<Float>]

Arguments: <Float> is the value to which the timer is set, in milliseconds. The minimum possible value is zero.

Response: None

Notes: The servo cycle time of the C-884 is 100 μ s (***Servo Update Time*** parameter, ID 0xE000200).

The current value of the timer can be recorded with the data recorder (see DRC (p. 163), record option 44).

TIM? (Get Timer Value)

Description: Gets the current value of the timer.

Format: TIM?

Arguments: None

Response: <Float> is the current value of the timer, in milliseconds. For details, see TIM (p. 238).

TIO? (Tell Digital I/O Lines)

Description: Tells number of installed digital I/O lines

Format: TIO?

Arguments: none

Response: I=<uint1>
O=<uint2>

where

<uint1> is the number of digital input lines.
<uint2> is the number of digital output lines.

Notes: The digital output lines reported by TIO? are the digital output lines 1 to 4. The state of these lines can be set using the DIO command (p. 161). Furthermore, you can program these lines using the CTO command (p. 153) (trigger configuration) and the TRO command (p. 240) (trigger enabling/disabling).

The digital input lines reported by TIO? are the digital input lines 1 to 4. They can be read with DIO? (p. 162), #4 (p. 142) and SRG? (p. 226).

All the lines are located on the **I/O** socket (p. 309) of the C-884.

TMN? (Get Minimum Commandable Position)

Description: Get the minimum commandable position in physical units.

Format: TMN? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the minimum commandable position in physical units

Note: The minimum commandable position is defined by the parameter 0x30. When redefining the zero position with the DFH (p. 159) command, the minimum commandable position is automatically adapted to the new zero position.

TMX? (Get Maximum Commandable Position)

Description: Get the maximum commandable position in physical units.

Format:	TMX? [{ <AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response	{<AxisID>="<float> LF}
	where
	<float> is the maximum commandable position in physical units
Note:	The maximum commandable position is defined by the parameter 0x15. When redefining the zero position with the DFH (p. 159) command, the maximum commandable position is automatically adapted to the new zero position.

TNR? (Get Number of Record Tables)

Description:	Gets the number of data recorder tables currently available on the controller.
Format:	TNR?
Arguments:	none
Response	<uint> is the number of data recorder tables which are currently available
Notes:	The C-884 has 8 data recorder tables with 8192 data points per table.
	For more information see "Data Recorder" (p. 89).

TRO (Set Trigger Output State)

Description:	Enables or disables the trigger output conditions which were set with CTO (p. 153) for the given digital output line.
Format:	TRO {<TrigOutID> <TrigMode>}
Arguments:	<TrigOutID> is a digital output line of the controller; see below for further details.
	<TrigMode> can have the following values: 0 = Trigger output disabled 1 = Trigger output enabled
Response:	None

Troubleshooting: Illegal identifier of the digital output line

Notes: <TrigOutID> corresponds to the digital output lines 1 to 4, IDs = 1 to 4; for further information, see "I/O" (p. 309).

Do not use DIO (p. 161) on digital output lines for which the trigger output is enabled with TRO.

TRO? (Get Trigger Output State)

Description: Returns if the trigger output configuration made with CTO (p. 153) is enabled or disabled for the given digital output line.

If all arguments are omitted, gets state of all digital output lines.

Format: TRO? [{<TrigOutID>}]

Arguments: <TrigOutID> is one digital output line of the controller, see TRO (p. 240) for more details.

Response: {<TrigOutID>="<TrigMode> LF}

where

<TrigMode> is the current state of the digital output line:
 0 = Trigger output disabled
 1 = Trigger output enabled

Troubleshooting: Illegal identifier of the digital output line

TRS? (Indicate Reference Switch)

Description: Indicates whether axes have a reference point switch with direction sensing.

Format: TRS? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<uint> indicates whether the axis has a direction-sensing reference point switch (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The C-884 firmware detects the presence or absence of a reference point switch via a parameter (ID 0x14). According to the value of this parameter, the C-884 enables or disables reference moves to the reference point switch (FRF command (p. 171)). Adapt the parameter value to your hardware using SPA (p. 223) or SEP (p. 219). For further information, see "Reference Point Switch Detection" (p. 32).

You can use a digital input line instead of the reference point switch as source of the reference point signal for the FRF command. For further information, see "Digital Input Signals" (p. 101).

TVI? (Tell Valid Character Set For Axis Identifiers)

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

Use SAI (p. 218) to change the axis identifiers and SAI? (p. 219) to ask for the current valid axis identifiers.

Format: TVI?

Arguments: None

Response: <string> is a list of characters

Notes: With the C-884, the string consists of 1234567890ABCDEFGHIJKLMNQRSTUWXYZ- _

VAR (Set Variable Value)

Description: Sets a variable to a certain value.

Local variables can be set using VAR in macros only. See "Variables" (p. 135) for more details on local and global variables.

The variable is present in RAM only.

Format: VAR <Variable> <String>

Arguments: <Variable> is the name of the variable whose value is to be set.

<String> is the value to which the variable is to be set. If omitted, the variable is deleted.

The value can be given directly or via the value of a variable.

See “Variables” (p. 135) for more details on conventions regarding variable names and values.

Response: None

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

```
VAR TARGET ${SOURCE}
```

Use braces if the name of the variable is longer than one character:

```
VAR A ONE
VAR VARB TWO
VAR $A 1
VAR ${VARB} 2
VAR $VARB 2 // this will result in an unwanted behavior
VAR?
A=ONE
VARB=TWO
ONE=1
TWO=2 // ${VARB} is replaced by its value “TWO”.
ARB=2 // $VARB: $V is replaced by its (empty) value.
```

See ADD (p. 147) for another example.

VAR? (Get Variable Values)

Description: Gets values of variables.

If VAR? is combined with CPY (p. 151), JRC (p. 201), MEX (p. 207) or WAC (p. 245), the response to VAR? has to be a single value and not more.

See “Variables” (p. 135) 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. 135) 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.
Notes:	Local variables can be queried using VAR? only when a macro with local variables is running. See "Variables" (p. 135) for details regarding local and global variables.
Example:	See ADD (p. 147) for an example.

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:	<p>The VEL setting only takes effect when the specified axis is in closed-loop operation (servo mode ON).</p> <p>The lowest possible value for <Velocity> is 0.</p> <p>The velocity can be changed with VEL while the axis is moving.</p> <p>VEL changes the value of the Closed-Loop Velocity (Phys. Unit/s) parameter (ID 0x49) in the volatile memory of C-884. The parameter value can be stored as default with WPA (p. 247), for details see "Adapting Settings" (p. 271).</p> <p>The maximum value that can be set with the VEL command is specified by the Maximum Closed-Loop</p>

Velocity (Phys. Unit/s), ID 0xA.

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.

Notes: VEL? queries the velocity value for closed-loop operation.

VER? (Get Versions Of Firmware And Drivers)

Description: Gets the versions of the firmware of the C-884 as well as of further components like, for example, drivers and libraries.

Format: VER?

Arguments: None

Response {<string1>":"<string2> [<string3>]LF}

where

<string1> is the name of the component;
 <string2> is the version information of the component
 <string1>;
 <string3> is an optional note.

WAC (Wait For Condition)

Description: Waits until a given condition of the following type occurs: a specified value is compared with a queried value according a specified rule.

Can only be used in macros.

See also the MEX command (p. 207).

Format: WAC <CMD?> <OP> <value>

Arguments <CMD?> is one query command in its usual notation. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= <= < > >= !=

Important: There must be a blank space before and after the operator!

<value> is the value to be compared with the response to <CMD?>.

Response: None

Example: Send:

```
MAC BEG LPMOTION
MVR 1 1
WAC ONT? 1 = 1
MVR 1 -1
WAC ONT? 1 = 1
MAC START LPMOTION
MAC END
MAC START LPMOTION
```

Note: Macro LPMOTION is first recorded and then started. WAC ONT? 1 = 1 waits until the response to ONT? 1 is 1=1. To form an infinite loop, the macro calls itself.

WPA (Save Parameters To Nonvolatile Memory)

Description: Writes the currently valid value of a parameter of a given item from volatile memory (RAM) to nonvolatile memory. The values saved this way become the default values.

Note: If the current parameter values are incorrect, this can cause a fault in the system. Make sure that the parameter settings are correct before you execute the WPA command.

RAM settings not saved with WPA will be lost when the controller is switched off or rebooted or when RPA (p. 216) is used to restore the parameters.

With HPA? (p. 194) you can obtain a list of all available parameters.

Use SPA? (p. 223) to check the current parameter settings in volatile memory.

See SPA (p. 223) for an example.

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

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

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

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

Response: None

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

Notes: Parameters can be changed in the volatile memory with SPA (p. 223), ACC (p. 146), DEC (p. 158) and VEL (p. 244).

WPA can also save parameter-independent settings that are set with the following commands:

HDT (p. 174) Assigns a lookup table to the axis of an HID device.

HIA (p. 176) Configures the HID control

HIT (p. 188) Fills lookup tables with values.

The password used determines what is saved with WPA, details see below.

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

Valid passwords for writing in the nonvolatile memory:	100	Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT
	101	Saves the currently valid values of all parameters
	HID	Saves the currently valid settings for HDT, HIA and HIT

Available item IDs and parameter IDs:	With the C-884, it is not possible to specifically select individual items and parameters for saving, i. e. <ItemID> and <PamID> are ignored.
---------------------------------------	---

8.6 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_	Open-loop commands (SVA,

	SERVO_ON	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 driver for use with NI LabVIEW reports error. See source control for details.
200	PI_CNTR_NO_AXIS	No stage connected to axis

201	PI_CNTR_NO_AXIS_PARAM_FILE	File with axis parameters not found
202	PI_CNTR_INVALID_AXIS_PARAM_FILE	Invalid axis parameter file
203	PI_CNTR_NO_AXIS_PARAM_BACKUP	Backup file with axis parameters not found
204	PI_CNTR_RESERVED_204	PI internal error code 204
205	PI_CNTR_SMO_WITH_SERVO_ON	SMO with servo on
206	PI_CNTR_UUDECODE_INCOMPLETE_HEADER	uudecode: incomplete header
207	PI_CNTR_UUDECODE_NOTHING_TO_DECODE	uudecode: nothing to decode
208	PI_CNTR_UUDECODE_ILLEGAL_FORMAT	uudecode: illegal UUE format
209	PI_CNTR_CRC32_ERROR	CRC32 error
210	PI_CNTR_ILLEGAL_FILENAME	Illegal file name (must be 8-0 format)
211	PI_CNTR_FILE_NOT_FOUND	File not found on controller
212	PI_CNTR_FILE_WRITE_ERROR	Error writing file on controller
213	PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE	VEL command not allowed in DTR command mode
214	PI_CNTR_POSITION_UNKNOWN	Position calculations failed
215	PI_CNTR_CONN_POSSIBLY_BROKEN	The connection between controller and stage may be broken
216	PI_CNTR_ON_LIMIT_SWITCH	The connected stage has driven into a limit switch, some controllers need CLR to resume operation
217	PI_CNTR_UNEXPECTED_STRUT_STOP	Strut test command failed because of an unexpected strut stop
218	PI_CNTR_POSITION_BASED_ON_ESTIMATION	While MOV! is running position can only be estimated!
219	PI_CNTR_POSITION_BASED_ON_INTERPOLATION	Position was calculated during MOV motion
220	PI_CNTR_INTERPOLATION_FIFO_UNDERRUN	FIFO buffer underrun during interpolation
221	PI_CNTR_INTERPOLATION_FIFO_OVERFLOW	FIFO buffer 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 response was received while executing WAC/MEX/JRC/...
1011	PI_CMD_NOT_VALID_IN_MACRO_MODE	Command not valid during macro execution
1024	PI_CNTR_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
1025	PI_CNTR_MAX_MOTOR_OUTPUT_REACHED	Maximum motor output reached
1063	PI_CNTR_EXT_PROFILE_UNALLOWED_COMMAND	User profile mode: command is not 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 cannot be defined as PPC axis
5003	PI_CNTR_PCC_SCAN_OUT_OF_RANGE	Defined scan area is larger than the travel range
5004	PI_CNTR_PCC_TYPE_NOT_EXISTING	Given PicoCompensation type is not defined

5005	PI_CNTR_PCC_PAM_ERROR	PicoCompensation parameter error
5006	PI_CNTR_PCC_TABLE_ARRAY_TOO_LARGE	PicoCompensation table is larger than maximum table length
5100	PI_CNTR_NEXLINE_ERROR	Common error in NEXLINE® firmware module
5101	PI_CNTR_CHANNEL_ALREADY_USED	Output channel for NEXLINE® cannot be redefined for other usage
5102	PI_CNTR_NEXLINE_TABLE_TOO_SMALL	Memory for NEXLINE® signals is too small
5103	PI_CNTR_RNP_WITH_SERVO_ON	RNP cannot be executed if axis is in closed loop
5104	PI_CNTR_RNP_NEEDED	Relax procedure (RNP) needed
5200	PI_CNTR_AXIS_NOT_CONFIGURED	Axis must be configured for this action
5300	PI_CNTR_FREQU_ANALYSIS_FAILED	Frequency analysis failed
5301	PI_CNTR_FREQU_ANALYSIS_RUNNING	Another frequency analysis is running
6000	PI_CNTR_SENSOR_ABS_INVALID_VALUE	Invalid preset value of absolute sensor
6001	PI_CNTR_SENSOR_ABS_WRITE_ERROR	Error while writing to sensor
6002	PI_CNTR_SENSOR_ABS_READ_ERROR	Error while reading from sensor
6003	PI_CNTR_SENSOR_ABS_CRC_ERROR	Checksum error of absolute sensor
6004	PI_CNTR_SENSOR_ABS_ERROR	General error of absolute sensor
6005	PI_CNTR_SENSOR_ABS_OVERFLOW	Overflow of absolute sensor position

Interface errors

0	COM_NO_ERROR	No error occurred during function call
-1	COM_ERROR	Error during com operation (could not be specified)
-2	SEND_ERROR	Error while sending data
-3	REC_ERROR	Error while receiving data

-4	NOT_CONNECTED_ERROR	Not connected (no port with given ID open)
-5	COM_BUFFER_OVERFLOW	Buffer overflow
-6	CONNECTION_FAILED	Error while opening port
-7	COM_TIMEOUT	Timeout error
-8	COM_MULTILINE_RESPONSE	There are more lines waiting in buffer
-9	COM_INVALID_ID	There is no interface or DLL handle with the given ID
-10	COM_NOTIFY_EVENT_ERROR	Event/message for notification could not be opened
-11	COM_NOT_IMPLEMENTED	Function not supported by this interface type
-12	COM_ECHO_ERROR	Error while sending "echoed" data
-13	COM_GPIB_EDVR	IEEE488: System error
-14	COM_GPIB_ECIC	IEEE488: Function requires GPIB board to be CIC
-15	COM_GPIB_ENOL	IEEE488: Write function detected no listeners
-16	COM_GPIB_EADR	IEEE488: Interface board not addressed correctly
-17	COM_GPIB_EARG	IEEE488: Invalid argument to function call
-18	COM_GPIB_ESAC	IEEE488: Function requires GPIB board to be SAC
-19	COM_GPIB_EABO	IEEE488: I/O operation aborted
-20	COM_GPIB_ENEB	IEEE488: Interface board not found
-21	COM_GPIB_EDMA	IEEE488: Error performing DMA
-22	COM_GPIB_EOIP	IEEE488: I/O operation started before previous operation completed
-23	COM_GPIB_ECAP	IEEE488: No capability for intended operation
-24	COM_GPIB_EFSO	IEEE488: File system operation error
-25	COM_GPIB_EBUS	IEEE488: Command error during device call
-26	COM_GPIB_ESTB	IEEE488: Serial poll-status

		byte lost
-27	COM_GPIB_ESRQ	IEEE488: SRQ remains asserted
-28	COM_GPIB_ETAB	IEEE488: Return buffer full
-29	COM_GPIB_ELCK	IEEE488: Address or board locked
-30	COM_RS_INVALID_DATA_BITS	RS-232: 5 data bits with 2 stop bits is an invalid combination, as is 6, 7, or 8 data bits with 1.5 stop bits
-31	COM_ERROR_RS_SETTINGS	RS-232: Error configuring the COM port
-32	COM_INTERNAL_RESOURCES_ERROR	Error dealing with internal system resources (events, threads, ...)
-33	COM_DLL_FUNC_ERROR	A DLL or one of the required functions could not be loaded
-34	COM_FTDIUSB_INVALID_HANDLE	FTDIUSB: invalid handle
-35	COM_FTDIUSB_DEVICE_NOT_FOUND	FTDIUSB: device not found
-36	COM_FTDIUSB_DEVICE_NOT_OPENED	FTDIUSB: device not opened
-37	COM_FTDIUSB_IO_ERROR	FTDIUSB: IO error
-38	COM_FTDIUSB_INSUFFICIENT_RESOURCES	FTDIUSB: insufficient resources
-39	COM_FTDIUSB_INVALID_PARAMETER	FTDIUSB: invalid parameter
-40	COM_FTDIUSB_INVALID_BAUD_RATE	FTDIUSB: invalid baud rate
-41	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_ERASE	FTDIUSB: device not opened for erase
-42	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_WRITE	FTDIUSB: device not opened for write
-43	COM_FTDIUSB_FAILED_TO_WRITE_DEVICE	FTDIUSB: failed to write device
-44	COM_FTDIUSB_EEPROM_READ_FAILED	FTDIUSB: EEPROM read failed
-45	COM_FTDIUSB_EEPROM_WRITE_FAILED	FTDIUSB: EEPROM write failed
-46	COM_FTDIUSB_EEPROM_ERASE_FAILED	FTDIUSB: EEPROM erase failed
-47	COM_FTDIUSB_EEPROM_NOT_PRESENT	FTDIUSB: EEPROM not present
-48	COM_FTDIUSB_EEPROM_NOT_PROGRAMMED	FTDIUSB: EEPROM not programmed
-49	COM_FTDIUSB_INVALID_ARGS	FTDIUSB: invalid arguments
-50	COM_FTDIUSB_NOT_SUPPORTED	FTDIUSB: not supported

-51	COM_FTDIUSB_OTHER_ERROR	FTDIUSB: other error
-52	COM_PORT_ALREADY_OPEN	Error while opening the COM port: was already open
-53	COM_PORT_CHECKSUM_ERROR	Checksum error in received data from COM port
-54	COM_SOCKET_NOT_READY	Socket not ready, you should call the function again
-55	COM_SOCKET_PORT_IN_USE	Port is used by another socket
-56	COM_SOCKET_NOT_CONNECTED	Socket not connected (or not valid)
-57	COM_SOCKET_TERMINATED	Connection terminated (by peer)
-58	COM_SOCKET_NO_RESPONSE	Can't connect to peer
-59	COM_SOCKET_INTERRUPTED	Operation was interrupted by a nonblocked signal
-60	COM_PCI_INVALID_ID	No device with this ID is present
-61	COM_PCI_ACCESS_DENIED	Driver could not be opened (on Vista: run as administrator!)
-62	COM_SOCKET_HOST_NOT_FOUND	Host not found
-63	COM_DEVICE_CONNECTED	Device already connected

DLL errors

-1001	PI_UNKNOWN_AXIS_IDENTIFIER	Unknown axis identifier
-1002	PI_NR_NAV_OUT_OF_RANGE	Number for NAV out of range--must be in [1.10000]
-1003	PI_INVALID_SGA	Invalid value for SGA--must be one of 1, 10, 100, 1000
-1004	PI_UNEXPECTED_RESPONSE	Controller sent unexpected response
-1005	PI_NO_MANUAL_PAD	No manual control pad installed, calls to SMA and related commands are not allowed
-1006	PI_INVALID_MANUAL_PAD_KNOB	Invalid number for manual control pad knob
-1007	PI_INVALID_MANUAL_PAD_AXIS	Axis not currently controlled by a manual control pad
-1008	PI_CONTROLLER_BUSY	Controller is busy with some lengthy operation (e.g.,

		reference move, fast scan algorithm)
-1009	PI_THREAD_ERROR	Internal error--could not start thread
-1010	PI_IN_MACRO_MODE	Controller is (already) in macro mode--command not valid in macro mode
-1011	PI_NOT_IN_MACRO_MODE	Controller not in macro mode--command not valid unless macro mode active
-1012	PI_MACRO_FILE_ERROR	Could not open file to write or read macro
-1013	PI_NO_MACRO_OR_EMPTY	No macro with given name on controller, or macro is empty
-1014	PI_MACRO_EDITOR_ERROR	Internal error in macro editor
-1015	PI_INVALID_ARGUMENT	One or more arguments given to function is invalid (empty string, index out of range, ...)
-1016	PI_AXIS_ALREADY_EXISTS	Axis identifier is already in use by a connected stage
-1017	PI_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
-1018	PI_COM_ARRAY_ERROR	Could not access array data in COM server
-1019	PI_COM_ARRAY_RANGE_ERROR	Range of array does not fit the number of parameters
-1020	PI_INVALID_SPA_CMD_ID	Invalid parameter ID given to SPA or SPA?
-1021	PI_NR_AVG_OUT_OF_RANGE	Number for AVG out of range--must be >0
-1022	PI_WAV_SAMPLES_OUT_OF_RANGE	Incorrect number of samples given to WAV
-1023	PI_WAV_FAILED	Generation of wave failed
-1024	PI_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
-1025	PI_RUNNING_MACRO	Controller is (already) running a macro
-1026	PI_PZT_CONFIG_FAILED	Configuration of PZT stage or amplifier failed
-1027	PI_PZT_CONFIG_INVALID_PARAMS	Current settings are not valid for desired configuration
-1028	PI_UNKNOWN_CHANNEL_IDENTIFIER	Unknown channel identifier
-1029	PI_WAVE_PARAM_FILE_ERROR	Error while reading/writing

		wave generator parameter file
-1030	PI_UNKNOWN_WAVE_SET	Could not find description of wave form. Maybe WG.INI is missing?
-1031	PI_WAVE_EDITOR_FUNC_NOT_LOADED	The WGWaveEditor DLL function was not found at startup
-1032	PI_USER_CANCELLED	The user cancelled a dialog
-1033	PI_C844_ERROR	Error from C-844 Controller
-1034	PI_DLL_NOT_LOADED	DLL necessary to call function not loaded, or function not found in DLL
-1035	PI_PARAMETER_FILE_PROTECTED	The open parameter file is protected and cannot be edited
-1036	PI_NO_PARAMETER_FILE_OPENED	There is no parameter file open
-1037	PI_STAGE_DOES_NOT_EXIST	Selected stage does not exist
-1038	PI_PARAMETER_FILE_ALREADY_OPENED	There is already a parameter file open. Close it before opening a new file
-1039	PI_PARAMETER_FILE_OPEN_ERROR	Could not open parameter file
-1040	PI_INVALID_CONTROLLER_VERSION	The version of the connected controller is invalid
-1041	PI_PARAM_SET_ERROR	Parameter could not be set with SPA--parameter not defined for this controller!
-1042	PI_NUMBER_OF_POSSIBLE_WAVES_EXCEEDED	The maximum number of wave definitions has been exceeded
-1043	PI_NUMBER_OF_POSSIBLE_GENERATORS_EXCEEDED	The maximum number of wave generators has been exceeded
-1044	PI_NO_WAVE_FOR_AXIS_DEFINED	No wave defined for specified axis
-1045	PI_CANT_STOP_OR_START_WAV	Wave output to axis already stopped/started
-1046	PI_REFERENCE_ERROR	Not all axes could be referenced
-1047	PI_REQUIRED_WAVE_NOT_FOUND	Could not find parameter set required by frequency relation
-1048	PI_INVALID_SPP_CMD_ID	Command ID given to SPP or

		SPP? is not valid
-1049	PI_STAGE_NAME_ISNT_UNIQUE	A stage name given to CST is not unique
-1050	PI_FILE_TRANSFER_BEGIN_MISSING	A uuencoded file transferred did not start with "begin" followed by the proper filename
-1051	PI_FILE_TRANSFER_ERROR_TEMP_FILE	Could not create/read file on host PC
-1052	PI_FILE_TRANSFER_CRC_ERROR	Checksum error when transferring a file to/from the controller
-1053	PI_COULDNT_FIND_PISTAGES_DAT	The PiStages.dat database could not be found. This file is required to connect a stage with the CST command
-1054	PI_NO_WAVE_RUNNING	No wave being output to specified axis
-1055	PI_INVALID_PASSWORD	Invalid password
-1056	PI_OPM_COM_ERROR	Error during communication with OPM (Optical Power Meter), maybe no OPM connected
-1057	PI_WAVE_EDITOR_WRONG_PARAMNUM	WaveEditor: Error during wave creation, incorrect number of parameters
-1058	PI_WAVE_EDITOR_FREQUENCY_OUT_OF_RANGE	WaveEditor: Frequency out of range
-1059	PI_WAVE_EDITOR_WRONG_IP_VALUE	WaveEditor: Error during wave creation, incorrect index for integer parameter
-1060	PI_WAVE_EDITOR_WRONG_DP_VALUE	WaveEditor: Error during wave creation, incorrect index for floating point parameter
-1061	PI_WAVE_EDITOR_WRONG_ITEM_VALUE	WaveEditor: Error during wave creation, could not calculate value
-1062	PI_WAVE_EDITOR_MISSING_GRAPH_COMPONENT	WaveEditor: Graph display component not installed
-1063	PI_EXT_PROFILE_UNALLOWED_CMD	User profile mode: command is not allowed, check for required preparatory commands
-1064	PI_EXT_PROFILE_EXPECTING_MOTION_ERROR	User profile mode: first target position in user profile is too

		far from current position
-1065	PI_EXT_PROFILE_ACTIVE	Controller is (already) in user profile mode
-1066	PI_EXT_PROFILE_INDEX_OUT_OF_RANGE	User profile mode: block or data set index out of allowed range
-1067	PI_PROFILE_GENERATOR_NO_PROFILE	ProfileGenerator: No profile has been created yet
-1068	PI_PROFILE_GENERATOR_OUT_OF_LIMITS	ProfileGenerator: Generated profile exceeds limits of one or both axes
-1069	PI_PROFILE_GENERATOR_UNKNOWN_PARAMETER	ProfileGenerator: Unknown parameter ID in Set/Get Parameter command
-1070	PI_PROFILE_GENERATOR_PAR_OUT_OF_RANGE	ProfileGenerator: Parameter out of allowed range
-1071	PI_EXT_PROFILE_OUT_OF_MEMORY	User profile mode: out of memory
-1072	PI_EXT_PROFILE_WRONG_CLUSTER	User profile mode: cluster is not assigned to this axis
-1073	PI_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
-1074	PI_INVALID_DEVICE_DRIVER_VERSION	The installed device driver doesn't match the required version. Please see the documentation to determine the required device driver version.
-1075	PI_INVALID_LIBRARY_VERSION	The library used doesn't match the required version. Please see the documentation to determine the required library version.
-1076	PI_INTERFACE_LOCKED	The interface is currently locked by another function. Please try again later.
-1077	PI_PARAM_DAT_FILE_INVALID_VERSION	Version of parameter DAT file does not match the required version. Current files are available at www.pi.ws .
-1078	PI_CANNOT_WRITE_TO_PARAM_DAT_FILE	Cannot write to parameter DAT file to store user defined stage type.
-1079	PI_CANNOT_CREATE_PARAM_DAT_FILE	Cannot create parameter DAT file to store user defined stage type.

-1080	PI_PARAM_DAT_FILE_INVALID_REVISION	Parameter DAT file does not have correct revision.
-1081	PI_USERSTAGES_DAT_FILE_INVALID_REVISION	User stages DAT file does not have correct revision.
-1082	PI_SOFTWARE_TIMEOUT	Timeout Error. Some lengthy operation did not finish within expected time.
-1083	PI_WRONG_DATA_TYPE	A function argument has an unexpected data type.
-1084	PI_DIFFERENT_ARRAY_SIZES	Length of data arrays is different.
-1085	PI_PARAM_NOT_FOUND_IN_PARAM_DAT_FILE	Parameter value not found in parameter DAT file.
-1086	PI_MACRO_RECORDING_NOT_ALLOWED_IN_THIS_MODE	Macro recording is not allowed in this mode of operation.
-1087	PI_USER_CANCELLED_COMMAND	Command cancelled by user input.
-1088	PI_TOO_FEW_GCS_DATA	Controller sent too few GCS data sets
-1089	PI_TOO_MANY_GCS_DATA	Controller sent too many GCS data sets
-1090	PI_GCS_DATA_READ_ERROR	Communication error while reading GCS data
-1091	PI_WRONG_NUMBER_OF_INPUT_ARGUMENTS	Wrong number of input arguments.
-1092	PI_FAILED_TO_CHANGE_CCL_LEVEL	Change of command level has failed.
-1093	PI_FAILED_TO_SWITCH_OFF_SERVO	Switching off the servo mode has failed.
-1094	PI_FAILED_TO_SET_SINGLE_PARAMETER_WHILE_PERFORMING_CST	A parameter could not be set while performing CST: CST was not performed (parameters remain unchanged).
-1095	PI_ERROR_CONTROLLER_REBOOT	Connection could not be reestablished after reboot.
-1096	PI_ERROR_AT_QHPA	Sending HPA? or receiving the response has failed.
-1097	PI_QHPA_NONCOMPLIANT_WITH_GCS	HPA? response does not comply with GCS2 syntax.
-1098	PI_FAILED_TO_READ_QSPA	Response to SPA? could not be received. Response to SPA? could not be received.

-1099	PI_PAM_FILE_WRONG_VERSION	Version of PAM file cannot be handled (too old or too new)
-1100	PI_PAM_FILE_INVALID_FORMAT	PAM file does not contain required data in PAM-file format
-1101	PI_INCOMPLETE_INFORMATION	Information does not contain all required data
-1102	PI_NO_VALUE_AVAILABLE	No value for parameter available
-1103	PI_NO_PAM_FILE_OPEN	No PAM file is open
-1104	PI_INVALID_VALUE	Invalid value
-1105	PI_UNKNOWN_PARAMETER	Unknown parameter
-1106	PI_RESPONSE_TO_QSEP_FAILED	Response to SEP? could not be received.
-1107	PI_RESPONSE_TO_QSPA_FAILED	Response to SPA? could not be received. 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_CONTANS_UN KNOWN_PAM_IDS	PI stage database: Response to HPA? contains unknown parameter IDs.
-10007	PI_PARAMETER_DB_AND_QHPA_ARE_INC ONSISTENT	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_PA RAMETERS_MISMATCH	Controller does not support all stage parameters stored in PI stage database. No parameters were set.
-10011	PI_PARAMETER_DB_DATABASE_IS_OUTD ATED	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_MISMATC H_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_MISMATC H_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_PAR AMETERS_CORRECTLY	One or more parameters could not be set correctly on the controller.
-10015	PI_PARAMETER_DB_MISSING_PARAMETE R_DEFINITIONS_IN_DATABASE	One or more parameter definitions are not present in stage database. Please update PISTAGES3.DB via PIUpdateFinder. Missing parameters were ignored.

9 Adapting Settings

In this Chapter

Settings of the C-884.....	271
Changing Parameter Values in the C-884	271
Creating or Modifying a Positioner Type	277
Parameter Overview	280

9.1 Settings of the C-884

The properties of the C-884 and the connected positioner are stored in the C-884 as parameter values (e.g., settings for the servo algorithm (p. 29)).

The parameters can be divided into the following categories:

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

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

Every parameter is in the volatile as well as in the nonvolatile memory of the C-884. The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-884. The values in the volatile memory determine the current behavior of the system.

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

9.2 Changing Parameter Values in the C-884

NOTICE



Unsuitable parameter settings!

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-884 and take effect immediately. Unsuitable parameter settings can cause damage to the connected positioner.

- Change parameter values only after careful consideration.
- Save the current parameter values to the PC (p. 273) before you make changes in the nonvolatile memory.

INFORMATION

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

- Overwrite the default values only when it is necessary.
- Save the current parameter values to the PC (p. 273) before you make changes in the nonvolatile memory.
- Contact our customer service department (p. 301), if the C-884 exhibits unexpected behavior.

INFORMATION

If the connected positioner has an ID chip (p. 15), the data is loaded from the ID chip into the volatile memory of the C-884 after switching on or rebooting the C-884.

The ID chip only contains some of the information that is required to operate the positioner with the C-884. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 15) into the volatile memory of the C-884. Parameters that are loaded from the ID chip or from a positioner database are marked in color in the parameter overview (p. 280).

9.2.1 General Commands for Parameters

The following general commands are available for parameters:

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

You can find details in the command descriptions (p. 142).

9.2.2 Commands for Fast Access to Individual Parameters

The following special commands only change the corresponding parameters in the volatile memory. When necessary, the changed values must be written to the nonvolatile memory with the `WPA` command (p. 247).

INFORMATION

The parameters listed below can also be changed with the general commands.

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

You can find details in the command descriptions (p. 142).

9.2.3 Saving Parameter Values in a Text File

INFORMATION

The C-884 is configured via parameters, e.g., to adapt the mechanics connected. Changing parameter values can cause undesirable results.

- Create a backup copy on the PC before changing the parameter settings of the C-884. You can then restore the original settings at any time.
- Create an additional backup copy with a new file name each time after optimizing the parameter values or adapting the C-884 to specific mechanics.

Prerequisites

You have read and understood the general notes on startup (p. 57).

- ✓ You have established communication between the C-884 and the PC with PIMikroMove or PITerminal via TCP/IP (p. 65), RS-232 (p. 59) 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 parameter values from which you want to create a backup copy.

- If you want to save the parameter values from the volatile memory of the C-884: Send the `SPA?` command.
 - If you want to save the parameter values from the nonvolatile memory of the C-884: Send the `SEP?` command.
3. Click on the **Save...** button.
The **Save content of terminal as textfile** window opens.
 4. In the **Save content of terminal as textfile** window, save the queried parameter values in a text file on your PC.

9.2.4 Changing Parameter Values: General Procedure

For working with parameters, you can use the general commands (p. 272) and the commands for quick access (p. 273).

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

NOTICE



Unsuitable parameter settings!

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the C-884 and take effect immediately. Unsuitable parameter settings can cause damage to the connected positioner.

- Change parameter values only after careful consideration.
- Save the current parameter values to the PC (p. 273) before you make changes in the nonvolatile memory.

INFORMATION

The following procedure is generally recommended for changing parameter values:

1. Change the parameter values in the volatile memory.
2. Check whether the C-884 works correctly with the changed parameter values.

If so:

- Write the changed parameter values into the nonvolatile memory.

If not:

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

INFORMATION

The write access for the parameters of the C-884 is defined by command levels. After the controller is switched on or rebooted, the active command level is always 0. On command levels > 1, write access is only available to PI service personnel.

- Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 301).

Requirements

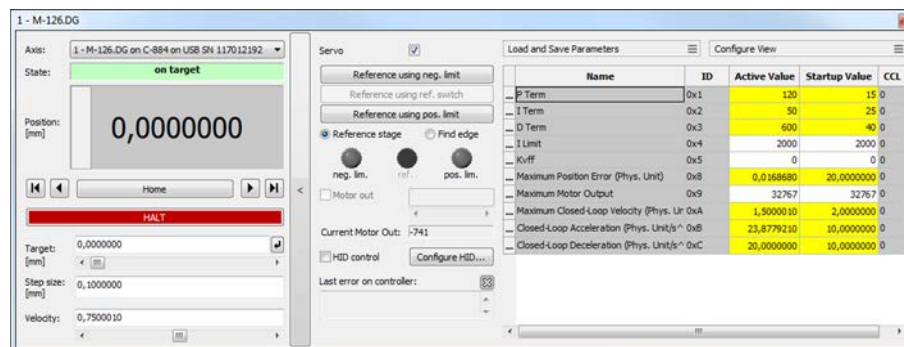
- ✓ If you want to change parameter values in the nonvolatile memory of the C-884: You have saved the parameter values of the C-884 in a text file on the PC (p. 273).
- ✓ You have established communication between the C-884 and the PC with PIMikroMove (p. 59).

Changing Parameter Values: General Procedure

1. Display the parameter list in PIMikroMove.

If you want to change the axis-related parameter of the C-884:

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



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

If you want to change the system-related parameters of the C-884:

- In the main window of PIMikroMove, open the window for the system-related parameters of the C-884 by selecting **C-884> Show system parameters** in the menu.

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

2. In the corresponding parameter list, change the desired parameter values in the volatile or nonvolatile memory of the C-884.

If you want to change parameter values in the volatile memory, you have the following options:

- Type the new parameter value into the corresponding input field in the **Active Value** column of the list. Press the **Enter** key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the volatile memory of the C-884.
- Click on **Load and Save Parameters -> Load all startup parameters of the axis / system from controller** in order to load the values of all axis-related / system-related parameters from the nonvolatile memory of the C-884.
- Click **Load and Save Parameters > Load parameters from stage database...** in the extended single-axis window to load a selected parameter set for the axis from the positioner database. You can use **Load and Save Parameters > Reload parameters from stage database...** to reload the currently loaded parameter set.

If you want to change parameter values in the nonvolatile memory, you have the following options:

- Type the new parameter value into the appropriate input field in the column **Startup Value** of the list. Press the **Enter** key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the nonvolatile memory of the C-884.
- Click on **Load and Save Parameters -> Save all currently active axis / system parameters as startup parameters to controller** to write the values of all axis-related / system-related parameters from the volatile to the nonvolatile memory of the C-884. You can skip parameters for which there is no write access on the current command level.

If a parameter value in the volatile memory (**Active Value** column) is different from the parameter value in the nonvolatile memory (**Startup Value** column), the line in the list is highlighted in color.

9.3 Creating or Modifying a Positioner Type

You can select a parameter set appropriate for your positioner from a positioner database in the PC software from PI. The software transfers the values of the selected parameter set to the volatile or nonvolatile memory of the controller. For further information, see "Positioner Databases" (p. 15).

You can create and edit new parameter records in the PISTages3 database. This can be required in the following cases, for example:

- You want to operate a positioner with different servo control parameter settings than the one from the default parameter set.
- You want to adapt the soft limits of the positioner to your application.
- You have a custom positioner.

INFORMATION

Possibilities for creating and editing parameter sets in the PISTAGES3.DB database:

- You can create a new positioner type easily by modifying an existing positioner type in PIMikroMove and saving it under a new name.
- With the PISTages3Editor program, which is included on the product CD, you can open and edit the positioner database directly. For more information on working with the PISTages3Editor, see the program manual.

PIMikroMove is used in the following for creating a new positioner type and for modifying an existing positioner type.

Requirements

- ✓ You have installed the latest version of the PISTAGES3.DB database onto your PC (p. 51).
- ✓ If PI provided a custom positioner database for your positioner, the dataset was imported (p. 54) into PISTages3.
- ✓ You have established communication with PIMikroMove between the C-884 and the PC (p. 59).

Create a positioner type in the positioner database

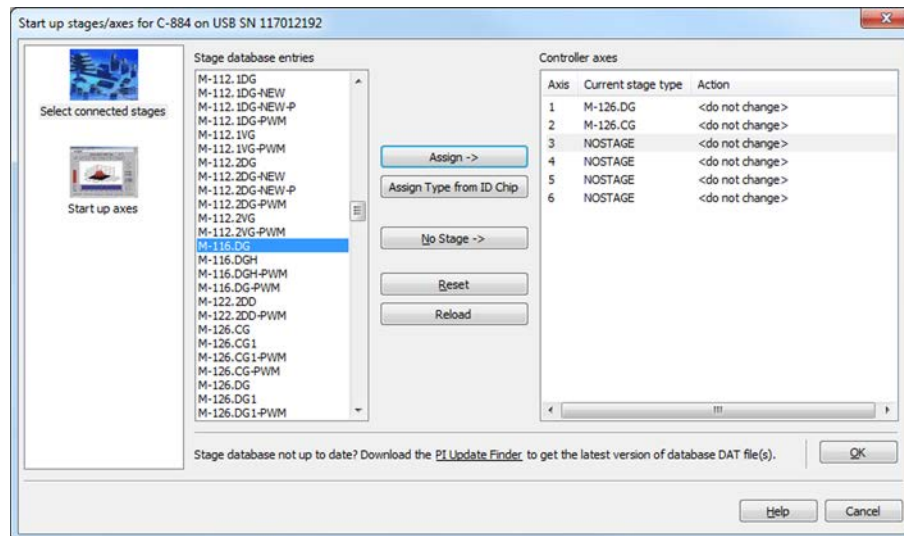
1. In the main window of PIMikroMove, select the **C-884 > Select connected stages...** menu item.

The **Start up stages/axes for C-884** window opens, the step **Select connected stages** is active.

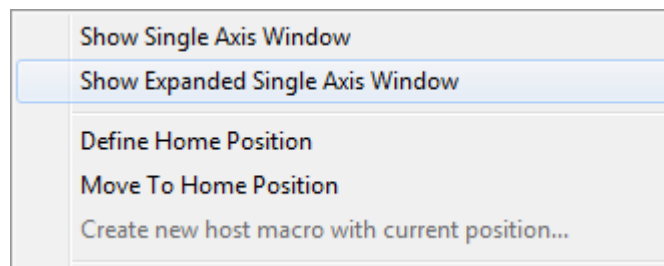
2. Select an appropriate type of positioner during the **Select connected stages** step:
 - Click on **Assign Type from ID Chip**.

or

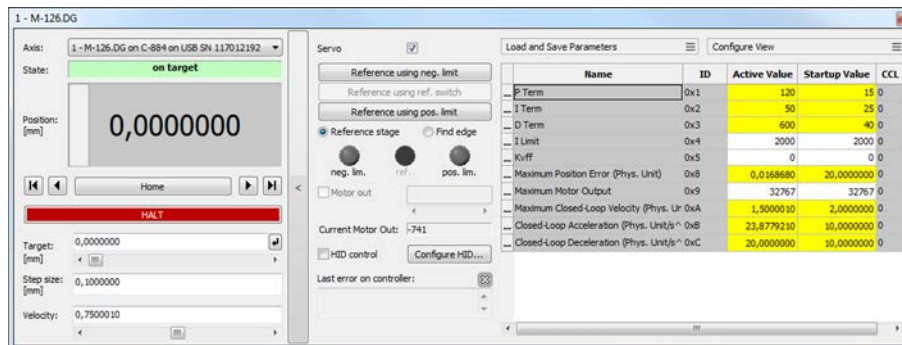
- a) Mark the positioner type in the **Stage database entries** list.
- b) Click **Assign**.



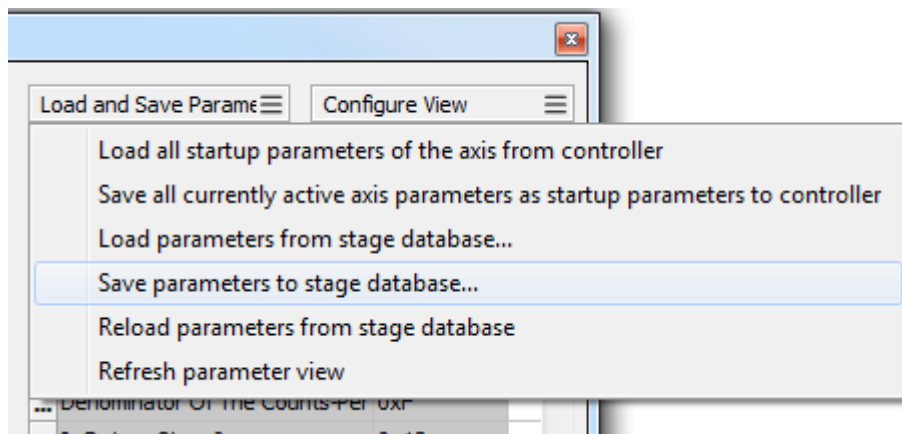
- c) Confirm the selection with **OK**.
 3. In the **Save all changes permanently** dialog, click **Keep the changes temporarily** to load the parameter settings into the volatile memory of the C-884.
- The **Start up stages/axes** window changes to the step **Start up axes**.
4. In the step **Start up axes** click on **Close** to close the **Start up stages/axes** window.
 5. Open the expanded single axis window for the selected positioner in the main window of PIMikroMove by clicking the right mouse button on the corresponding line of the **Axes** tab and selecting **Show Expanded Single Axis Window** in the context menu.



6. Enter new values for the parameters to be changed:



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



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

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

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

the main window of PIMikroMove. The new positioner type is also available immediately for selection in the **Select connected stages** step.

Changing a positioner type in the positioner database

1. In the main window of PIMikroMove, select the **C-884 > Select connected stages...** menu item.

The **Start up stages/axes for C-884** window opens, the step **Select connected stages** is active.

2. Select a positioner type that you created during the **Select connected stages** step as described above (p. 277). Proceed with the selection as described in step 2 of the **Creating a positioner type in a positioner database** instruction.
3. Proceed with steps 3 to 7 in **Creating a positioner type in the positioner database**.
4. Save the modified parameter values of the positioner type in the **Save Parameters as User Stage Type** dialog:
 - a) Leave the entry in the **Parameters of axis** field unchanged.
 - b) Leave the entry in the **Save as** field unchanged.
 - c) Click **OK**.
 - d) In the **Stage type already defined** dialog, click on **Change settings**. The **Save Parameters as User Stage Type** dialog closes automatically after a short time.

The parameter values of the positioner type have been updated in the PISTAGES3.DB positioner database and in the main window of PIMikroMove.

9.4 Parameter Overview

INFORMATION

The write access for the parameters of the C-884 is defined by command levels. After the controller is switched on or rebooted, the active command level is always level 0. For particular parameters, write access is only allowed on command level 1. On command levels > 1, write access is only available to PI service personnel.

The C-884 ignores the active command level in the following cases:

- The C-884 reads parameter values from the ID chip of the positioner.
 - The positioner type is selected in the PC software.
 - The current parameter values are written from the volatile to the nonvolatile memory (directly with WPA or in the PC software).
- If necessary, send the **CCL 1 advanced** command or enter the password **advanced** to change to command level 1.
 - Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 301).

INFORMATION

To save parameter values in the nonvolatile memory, it is necessary to enter a password.
Usable passwords:

- 100 Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT
 Use with the WPA and SEP commands
- 101 Saves the currently valid values of all parameters
 Use with the WPA command

Meaning of the color highlight in the parameter table:

Dark gray:	The value of the parameter is loaded from the ID chip of the positioner (p. 15).
Light gray:	The value of the parameter can be loaded from a positioner database (p. 15).
Colorless:	The value of the parameter can only be changed via command (SPA, SEP) or by using corresponding operating elements of the PC software (p. 274).

Designations in the header of the following table:

- ID = Parameter ID, hexadecimal format
- Type = Data type:
 - INT = Integer value, including Boolean values
 - FLOAT = Floating point number
 - CHAR = String format
- CL = Command Level for write access
- Element = Element type to which the parameter refers, for more information, see "Commandable Items" (p. 18)
- Parameter name = Name of the parameter
- Description = Explanation of the parameter

ID	Type	CL	Element	Parameter name	Description
0x1	INT	0	Axis	P term	Proportional constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).

ID	Type	CL	Element	Parameter name	Description
0x2	INT	0	Axis	I term	Integration constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x3	INT	0	Axis	D term	Differential constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x4	INT	0	Axis	I limit	Limiting the integration constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x5	INT	0	Axis	Kvff	Feed-forward control of the commanded velocity For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x8	FLOAT	0	Axis	Maximum Position Error (Phys. Unit)	Maximum position error Is used for detecting motion errors. For details, see "Behavior with Motion Error".
0x9	INT	0	Axis	Maximum Motor Output	Maximum control value for driving the axis (dimensionless) 0 to 32767
0xA	FLOAT	0	Axis	Maximum Closed-Loop Velocity (Phys. Unit/s)	Maximum velocity in closed-loop operation with dynamics profile Specifies the maximum value for parameter 0x49. For details, see "Generation of Dynamics Profile" (p. 25).
0xB	FLOAT	0	Axis	Closed-Loop Acceleration (Phys. Unit/s ²)	Acceleration in closed-loop operation with dynamics profile Is limited by parameter 0x4A. For details, see "Generation of Dynamics Profile" (p. 25).
0xC	FLOAT	0	Axis	Closed-Loop Deceleration (Phys. Unit/s ²)	Deceleration in closed-loop operation with dynamics profile Is limited by parameter 0x4B. For details, see "Generation of Dynamics Profile" (p. 25).

ID	Type	CL	Element	Parameter name	Description
0xE	INT	0	Axis	Numerator Of The Counts-Per-Physical-Unit Factor	Numerator and denominator of the factor for counts per physical length unit For details, see "Physical Units" (p. 22).
0xF	INT	0	Axis	Denominator Of The Counts-Per-Physical-Unit Factor	
0x10	INT	0	Axis	Output Mode	Present for compatibility reasons only.
0x13	INT	0	Axis	Is Rotary Stage?	Is this a rotation stage? 0 = Not a rotation stage 1 = Rotation stage No evaluation by the C-884, but only by the PC software: PIMikroMove determines which motion is permissible on the basis of this value.
0x14	INT	0	Axis	Has Reference?	Does the positioner have a reference point switch? For details, see "Reference Point Switch Detection" (p. 32).
0x15	FLOAT	0	Axis	Maximum Travel In Positive Direction (Phys. Unit)	Soft limit in positive direction See examples in "Travel Range and Soft Limits" (p. 35).
0x16	FLOAT	0	Axis	Value At Reference Position (Phys. Unit)	Position value at the reference point switch See examples in "Travel Range and Soft Limits" (p. 35).
0x17	FLOAT	0	Axis	Distance From Negative Limit To Reference Position (Phys. Unit)	Distance between reference point switch and negative limit switch See examples in "Travel Range and Soft Limits" (p. 35).
0x18	INT	0	Axis	Limit Mode	Signal logic of the limit switches For details, see "Limit Switch Detection" (p. 33).
0x1A	INT	0	Axis	Has Brake?	Does the positioner have a brake? 0 = No brake present 1 = Brake present. In this case, switching the servo mode on/off and activating/deactivating the brake are coupled to each other, see BRA (p. 149) and SVO (p. 228).

ID	Type	CL	Element	Parameter name	Description
0x2F	FLOAT	0	Axis	Distance From Reference Position To Positive Limit (Phys. Unit)	Distance between reference point switch and positive limit switch See examples in "Travel Range and Soft Limits" (p. 35).
0x30	FLOAT	0	Axis	Maximum Travel In Negative Direction (Phys. Unit)	Soft limit in negative direction See examples in "Travel Range and Soft Limits" (p. 35).
0x31	INT	0	Axis	Invert Reference?	Should the reference signal be inverted? For details, see "Reference Point Switch Detection" (p. 32).
0x32	INT	0	Axis	Has No Limit Switches?	Does the positioner have limit switches? For details, see "Limit Switch Detection" (p. 33).
0x33	INT	0	Axis	Motor Offset Positive	Offset for the positive direction of motion For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x34	INT	0	Axis	Motor Offset Negative	Offset for the negative direction of motion For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x36	INT	0	Axis	Settling Window (encoder counts)	Settling window around the target position For details, see "On-Target State" (p. 31).
0x3C	CHAR	0	Axis	Stage Name	Positioner name Maximum of 20 characters; default value: NOSTAGE See also "Deactivation of Axes" (p. 42).
0x3F	FLOAT	0	Axis	Settling Time (s)	Delay time for setting the on-target state. For details, see "On-Target State" (p. 31).
0x47	INT	0	Axis	Reference Travel Direction	Default direction for the reference move For details, see "Reference Point Definition" (p. 38).

ID	Type	CL	Element	Parameter name	Description
0x48	INT	0	Axis	Motor Drive Offset	Velocity-dependent offset For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x49	FLOAT	0	Axis	Closed-Loop Velocity (Phys. Unit/s)	Velocity in closed-loop operation with dynamics profile Is limited by parameter 0xA For details, see "Generation of Dynamics Profile" (p. 25).
0x4A	FLOAT	0	Axis	Maximum Closed-Loop Acceleration (Phys. Unit/s ²)	Maximum acceleration in closed-loop operation with dynamics profile Specifies the maximum value for parameter 0xB. For details, see "Generation of Dynamics Profile" (p. 25).
0x4B	FLOAT	0	Axis	Maximum Closed-Loop Deceleration (Phys. Unit/s ²)	Maximum deceleration in closed-loop operation with dynamics profile Specifies the maximum value for parameter 0xC. For details, see "Generation of Dynamics Profile" (p. 25).
0x50	FLOAT	0	Axis	Velocity For Reference Moves (Phys. Unit/s)	Maximum velocity for reference move For details, see "Reference Point Definition" (p. 38).
0x5A	INT	0	Axis	Numerator Of The Servo-Loop Input Factor	Numerator and denominator of the servo-loop input factor For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x5B	INT	0	Axis	Denominator Of The Servo-Loop Input Factor	
0x5C	INT	0	Axis	Source Of Reference Signal	Reference signal source for axis motion to the reference point switch For details, see "Commands and Parameters for Digital Inputs" (p. 101) and "Using Digital Input Signals as Switch Signals" (p. 103).
0x5D	INT	0	Axis	Source Of Negative Limit Signal	Reference signal source for axis motion to the negative travel range limit For details, see "Commands and Parameters for Digital Inputs" (p. 101) and "Using Digital Input Signals as Switch Signals" (p. 103).

ID	Type	CL	Element	Parameter name	Description
0x5E	INT	0	Axis	Source Of Positive Limit Signal	Reference signal source for axis motion to the positive travel range limit For details, see "Commands and Parameters for Digital Inputs" (p. 101) and "Using Digital Input Signals as Switch Signals" (p. 103).
0x5F	INT	0	Axis	Invert Digital Input Used For Negative Limit	Inverts the polarity of the digital inputs that are used as the source of the negative limit switch signal. For details, see "Commands and Parameters for Digital Inputs" (p. 101) and "Using Digital Input Signals as Switch Signals" (p. 103).
0x60	INT	0	Axis	Invert Digital Input Used For Positive Limit	Inverts the polarity of the digital inputs that are used as the source of the positive limit switch signal. For details, see "Commands and Parameters for Digital Inputs" (p. 101) and "Using Digital Input Signals as Switch Signals" (p. 103).
0x61	INT	0	Axis	Invert Direction Of Motion For Joystick-Controlled Axis?	Should the direction of motion for HID-controlled axes be inverted? For details, see "Commands and Parameters for Human Interface Devices" (p. 107).
0x63	FLOAT	0	Axis	Distance Between Limit And Hard Stop (Phys. Unit)	Distance between the built-in limit switch and the hard stop For details, see "Reference Point Definition" (p. 38).
0x70	INT	0	Axis	Reference Signal Type	Reference signal type For details, see "Reference Point Switch Detection" (p. 32).
0x71	INT	0	Axis	D Term Delay (No. Of Servo Cycles)	D term delay For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x72	INT	0	System	Ignore Macro Error?	Ignore macro error? For details, see "Commands and Parameters for Macros" (p. 117).
0x74	FLOAT	0	Axis	Closed-Loop Velocity For HI Control (Phys. Unit/s)	Maximum velocity during HID control For details, see "Commands and Parameters for Human Interface

ID	Type	CL	Element	Parameter name	Description
					Devices" (p. 107).
0x75	FLOAT	0	Axis	Closed-Loop Acceleration For HI Control (Phys. Unit/s ²)	Maximum acceleration during HID control For details, see "Commands and Parameters for Human Interface Devices" (p. 107).
0x76	FLOAT	0	Axis	Closed-Loop Deceleration For HI Control (Phys. Unit/s ²)	Maximum deceleration during HID control For details, see "Commands and Parameters for Human Interface Devices" (p. 107).
0x77	INT	0	Axis	Use Limit Switches Only For Reference Moves?	Should the limit switches only be used for reference moves? For details, see "Limit Switch Detection" (p. 33).
0x78	FLOAT	0	Axis	Distance From Limit To Start Of Ref Search (Phys. Unit)	Distance between the limit switch and the starting position for the reference move to the index pulse For details, see "Reference Point Definition" (p. 38).
0x79	FLOAT	0	Axis	Distance For Reference Search (Phys. Unit)	Maximum distance for the reference move to the index pulse For details, see "Reference Point Definition" (p. 38).
0x7C	FLOAT	0	Axis	Maximum Motor Output (V)	Maximum permissible voltage to be output to the motor 0 to 24
0x94	FLOAT	0	Axis	Notch Filter Frequency 1 (Hz)	Frequency of the first notch filter For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x95	FLOAT	0	Axis	Notch Filter Edge 1	Rise of the edge of the first notch filter For details, see "Servo Algorithm and Other Control Value Corrections" (p. 29).
0x130	INT	0	Axis	Inhibit Motion Commands	Configures the triggering of motion via motion commands. For details, see "Motion Triggering" (p. 23).

ID	Type	CL	Element	Parameter name	Description
0x3003320	INT	2	Axis	Sensor Signal Type	Signal type output by the position sensor 0 = No sensor 1 = A/B (default setting) 3 = BiSS relative (32 bits) 4 = BiSS absolute (32 bits) 5 = BiSS 24 bits relative and 32 bits absolute 6 = BiSS 32 bits absolute and 24 bits relative 7 = BiSS relative (24 bits) 8 = BiSS absolute (24 bits)
0x3003330	FLOAT	2	Axis	Abs sensor offset	Offset for correction of the signals of an absolute sensor
0x7000000	FLOAT	0	Axis	Range Limit Min	Additional soft limit for the negative direction of motion (physical unit) For details, see "travel range and Soft Limits" (p. 34).
0x7000001	FLOAT	0	Axis	Range Limit Max	Additional soft limit for the positive direction of motion (physical unit) For details, see "travel range and Soft Limits" (p. 34).
0x7000601	CHAR	0	Axis	Axis Unit	Unit symbol For details, see "Physical Units" (p. 22).
0xE000200	FLOAT	2	System	Servo Update Time	Servo cycle time in seconds
0x0F000100	CHAR	2	Axis	Stage Type	Positioner type Format for standard positioner: x-xxx Format for customized positioners: x-xxxKxxx
0x0F000200	CHAR	2	Axis	Stage Serial Number	Serial number of the positioner 9-digit number
0x0F000300	CHAR	2	Axis	Stage Assembly Date	Date of manufacture of the positioner Date format: DDMMYY
0x0F000400	INT	2	Axis	Stage HW Version	Version number of the positioner hardware

ID	Type	CL	Element	Parameter name	Description
0x16000001	INT	0	System	Recorded Points Per Trigger	Number of data points to be recorded per trigger For details, see "Data Recorder" (p. 89).
0x16000002	INT	0	System	Clearing Of RecTable On Trigger	Write mode during the recording For details, see "Data Recorder" (p. 89).
0x16000003	INT	0	System	Data Recorder Buffer Mode	Behavior with full data recorder tables For details, see "Data Recorder" (p. 89).
0x16000004	INT	3	System	Data Recorder Buffer Overflow	Buffer overflow counter of the data recorder For details, see "Data Recorder" (p. 89).
0x16000200	INT	1	System	Data Recorder Max. Points	Maximum number of all points of the data recorder tables For details, see "Data Recorder" (p. 89).
0x22000020	INT	2	System	Maximum Buffer Size	Maximum number of trajectory points in the memory For details, see "Trajectories for Motion Paths" (p. 85).

10 Maintenance

In this Chapter

Cleaning the C-884 291

Updating Firmware 291

10.1 Cleaning the C-884

NOTICE



Short circuits or flashovers!

The C-884 contains electrostatic-sensitive devices that can be damaged by short circuits or flashovers when cleaning fluids penetrate the housing.

Before cleaning, disconnect the C-884 from the power source by removing the mains plug.

- Prevent cleaning fluid from penetrating the housing

- When necessary, clean the C-884 case surface with a cloth lightly dampened with a mild cleanser or disinfectant.

10.2 Updating Firmware

INFORMATION

When updating the firmware, the macros saved on the C-884 are retained.

INFORMATION

The firmware of the C-884 consists of several components that can be updated separately.

INFORMATION

To update the firmware, the C-884 and PC must communicate via the TCP/IP interface.

INFORMATION

The `*IDN?` command reads the device identification string of the C-884, which also contains the Linux firmware component version.

Example: `(c)2018 Physik Instrumente (PI) GmbH & Co. KG, C-884.4DC, 123456789, 1.0.0.0`

- `C-884.4DC` device name
- `123456789` serial number of the C-884, contains the encrypted date of manufacture
- `1.0.0.0` Version of the Linux firmware component

The versions of further firmware components can be read out with the `VER?` command.

Example:

- `FW_ARM: V1.0.0.0`: Version of the Linux firmware component
- `FW_DSP: V0015`: Version of the DSP firmware
- `FW_FPGA: V0110`: Version of the FPGA firmware

Requirements

- ✓ You have connected the C-884 to the network via the RJ45 Ethernet socket on its front panel or directly to the PC (p. 55).
- ✓ You have read and understood the documentation that you have received from our customer service department together with the current firmware files.
- ✓ You have made all necessary preparations for communication via the TCP/IP interface, see "Establishing Communication via TCP/IP in the PC-Software" (p. 65).
- ✓ The C-884 is switched on and the starting procedure of the C-884 has finished (p. 58).
- ✓ The PC is switched on.

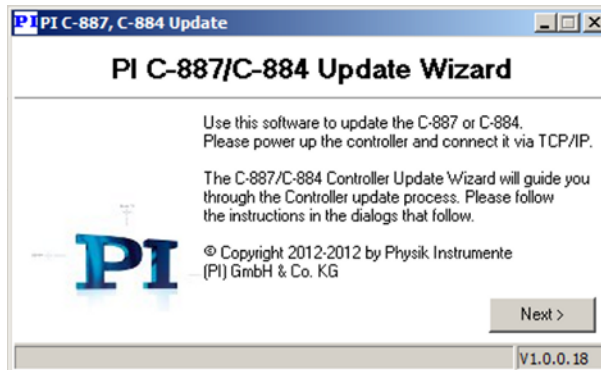
Tools and accessories

- PC with Windows operating system that has been prepared as follows:
 - The **C-887/C-884 Controller Update Wizard** program from the product CD (in the scope of delivery) is installed.
 - The current files of the firmware components that you have received from our customer service department (file types: IPK or JED) are located in a directory on the PC.

Updating firmware components of the C-884

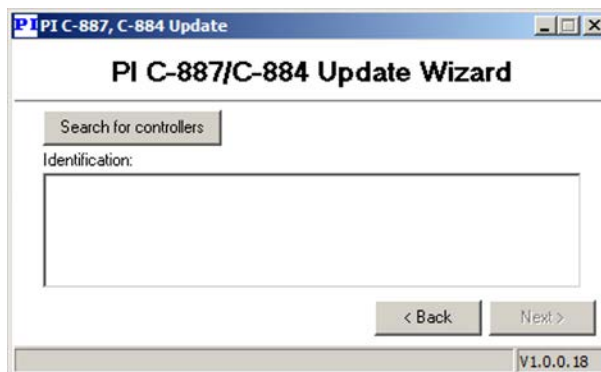
1. Start the **C-887/C-884 Controller Update Wizard** program on the PC via the **All Programs > PI > C-884 > Firmware Update Tool** start menu item.

The **PI C-887, C-884 Update** window opens.



2. Click the **Next >** button.

The **Search for controllers** button is displayed.

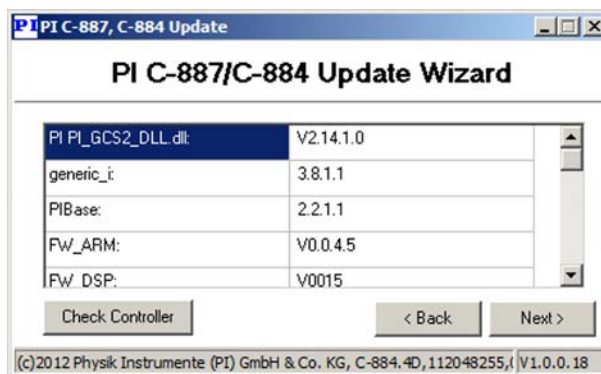


3. Click the **Search for controllers** button.

The found C-884 is displayed in the **Identification:** field (serial number, port, IP address).

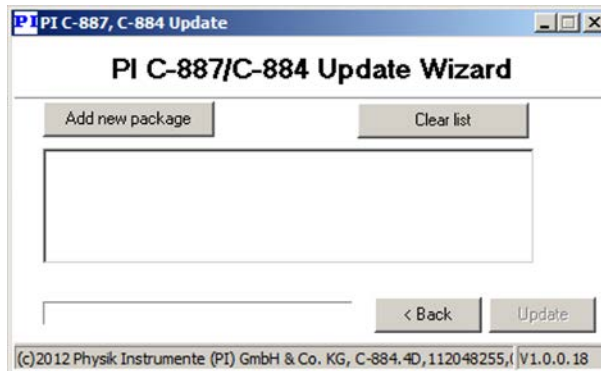
4. Mark the line of the C-884 in the **Identification:** field.
5. Click the **Next >** button.

The currently active firmware components of the C-884 as well as the dynamic program library PI_GCS2_DLL installed on the PC are listed with their version numbers.



6. Click the **Next >** button.

The **Add new package** button for selecting the firmware components to be updated is displayed.



7. 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. Examples of filenames:
 - Linux firmware component and DSP firmware: adding-file_1.1.1_armel.ipk
 - FPGA firmware: C884_4dc_01010.jed
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-884.

After the firmware has been transferred to the controller, the **Information** window is displayed.



12. Click the **OK** button.
The C-884 reboots.
The updating of the firmware components is finished.

11 Troubleshooting

Fault: Positioner does not move	
Possible causes	Remedial measures
Cable not connected correctly	➤ Check the cable connections.
Positioner or positioner cable is defective	➤ If available, replace the defective positioner with another positioner and test the new combination.
Positioner not connected to power adapter	➤ Connect the positioner to a suitable power adapter and make sure that the power adapter is functioning properly.
Limit switch signals not compatible with the C-884	Positioners from third-party suppliers may use unsuitable limit switch signals. ➤ Contact the customer service department (p. 301) and the manufacturer of the positioner.
The positioner has been connected to the switched-on C-884	The sensor electronics in the positioner have not been initialized and the ID chip of the sensor (p. 15) has not been read out. ➤ Switch the C-884 off and on again, or reboot the C-884 with the RBT command or with the corresponding functions of the PC software.
Unsuitable positioner cable used	If unsuitable cables are used, interference can occur in the signal transmission between the positioner and the C-884. ➤ Only use genuine PI parts to connect the positioner to the C-884. ➤ If you need extension cables, contact our customer service department (p. 301).
Incorrect configuration	➤ Check the parameter settings of the C-884 with the SPA? (volatile memory) and SEP? (nonvolatile memory) commands; see "Adapting Settings" (p. 271).
Incorrect command or incorrect syntax	➤ Send the ERR? command and check the error code that is returned.
Incorrect axis commanded	➤ Make sure that the correct axis identifier is used and that the commanded axis belongs to the correct positioner.
HID control is enabled	Motion commands and the execution of trajectories are not allowed when the HID control is enabled for the axis. ➤ Disable HID control with the HIN command.

Fault: Positioner does not move	
Possible causes	Remedial measures
When executing a trajectory: Trajectory buffer empty	<p>Before a trajectory is executed, at least 4 points must be loaded to the trajectory buffer with TGA (p. 231). During the execution of a trajectory, the buffer must be refilled fast enough. The execution of a trajectory must be completed with TGF (p. 234).</p> <ul style="list-style-type: none"> ➤ Make sure that a sufficient number of trajectory points is always in the buffer.

Fault: Positioner performs unintentional motion	
Possible causes	Remedial measures
Human interface device is not connected but HID control is activated in the C-884	<ul style="list-style-type: none"> ➤ Activate HID control only when the C-884 is actually connected to a human interface device.
HID axis/axes not calibrated	<ul style="list-style-type: none"> ➤ Calibrate the axis/axes of the human interface device (p. 110).
Startup macro is run	<ul style="list-style-type: none"> ➤ Check whether a macro is specified as the startup macro and cancel the selection of the startup macro if necessary (p. 119).
Positioner brake is deactivated	<ul style="list-style-type: none"> ➤ Switch off the servo mode and activate the brake with the BRA (p. 149) command. ➤ Secure the positioner against unintentional motion before you deactivate the brake by command!
Wrong trajectory executed	<ul style="list-style-type: none"> ➤ Start the desired trajectory for the desired axis.

Fault: Positioner is oscillating or positions inaccurately	
Possible causes	Remedial measures
The load was changed.	<ul style="list-style-type: none"> ➤ Readjust the system according to the changed load (p. 79).
When executing a trajectory: Unsuitable trajectory design	<p>Following an unsuitable trajectory can cause the positioner to oscillate or stop motion abruptly. Oscillation or stopping abruptly can damage the positioner and/or the load fixed to it.</p> <ul style="list-style-type: none"> ➤ Make sure that the path that is specified by the trajectory points is continuously differentiable at least twice. ➤ Design the trajectory (trajectory points and timing) so that the maximum permissible velocity and acceleration of the axis is not exceeded.

Fault: Positioner is already oscillating during the reference move	
Possible causes	Remedial measures
Very high load on the positioner	<p>In case of a very high load, proceed with PIMikroMove during the reference move as follows:</p> <ol style="list-style-type: none"> 1. Do not start the reference move in the Start up axes step, but click on Close to close the Start up controller window instead. 2. Open the single axis window for the connected positioner in the main window by selecting the positioner in the View > Single Axis Window menu. 3. Expand the view of the single axis window by clicking on the > button at the right edge of the window. 4. With the Servo check box, make sure that the servo mode is switched on. 5. Start the reference move by clicking on one of the Reference... buttons. 6. If the positioner is oscillating: Stop the reference move immediately in the Reference Axes dialog, close the dialog and switch off the servo mode by unchecking the respective check box in the single axis window. 7. Enter suitable values for the settings of the notch filter, see "Setting the Notch Filter" (p. 73). 8. Restart the reference move. 9. If the positioner continues to oscillate, repeat steps 6 to 8 until the reference move has completed successfully without oscillation.

Fault: There is no communication between the controller and the PC	
Possible causes	Remedial measures
The wrong communication cable is used or it is defective	<ul style="list-style-type: none"> ➤ Use the following cables for TCP/IP connections (p. 55): <ul style="list-style-type: none"> – TCP/IP direct connection with the PC: crossover network cable – TCP/IP network: Straight-through network cable ➤ Use a null modem cable for the RS-232 connection (p. 55). ➤ If necessary, check whether the cable works on a fault-free system.
Communication interface is not correctly configured	<p>When using the RS-232 interface:</p> <ul style="list-style-type: none"> ➤ Check the port settings, the baud rate and the handshake setting of the PC. <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. 62). ➤ Make sure that the network is not blocked for unknown devices. ➤ Make sure that several PC software applications cannot access the C-884 at the same time. ➤ Make sure that you have selected the correct C-884 when establishing communication. ➤ If you cannot solve the problems, consult your network administrator if necessary.
The start procedure of the C-884 firmware has not finished yet	<ol style="list-style-type: none"> 1. After switching on or rebooting the C-884, wait until the PWR and STA LEDs light up continuously. 2. Try to establish communication.
Another program is accessing the interface.	<ul style="list-style-type: none"> ➤ Close the other program.
Problems with special software	<ul style="list-style-type: none"> ➤ Check whether the system works with other software, such as a terminal program or a development environment. You can test communication by starting a terminal program (such as PITerminal for example) and entering <code>*IDN?</code> or <code>HLP?</code>. ➤ Make sure that you end the commands with an LF (line feed). <p>A command is only executed when LF has been received.</p>

Fault: The customer software does not function with the PI drivers	
Possible causes	Remedial measures
Incorrect combination of driver routines/VIs	<ul style="list-style-type: none"> ➤ Check whether the system functions with a terminal program (e.g., PITerminal). <p>If so:</p> <ul style="list-style-type: none"> ➤ Read the information in the corresponding software manual and compare the sample code on the product CD with your program code.

Fault: Controller does not send an error code in the case of incorrect system behavior	
Possible causes	Remedial measures
Error code was already queried by another instance	<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 <code>ERR?</code> command. The error code is reset to 0 during the query.</p> <ul style="list-style-type: none"> ➤ If possible, access the controller with one instance only. ➤ Check whether the error code is regularly queried in the background by a macro, a script or PC software (e.g., PIMikroMove).

If the problem that occurred with your system is not in the list above or cannot be solved as described, contact our customer service department (p. 301).

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 available)
 - Version of the driver or the software (if available)
 - Operating system on the PC (if available)
- If possible: Take photographs or make videos of your system that can be sent to our customer service department if requested.

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

13 Technical Data

In this Chapter

Specifications	303
System Requirements	305
Dimensions	307
Pin Assignment	308

13.1 Specifications

13.1.1 Data Table

	C-884.4DC / C-884.6DC
Function	Position control for closed-loop DC motors
Processor	Dual-core architecture. Controller on a DSP core, with extendable command interpreter in an ARM core under Linux
Axes	C-884.4DC: 4 / C-884.6DC: 6
Supported functions	Linear vector motion. Point-to-point motion. User-definable trajectories. Startup macro. PI Python. Data recorder for recording operating data such as motor voltage, velocity, position or position error. ID chip detection.




Motion and servo controller	C-884.4DC / C-884.6DC
Controller type	PID controller, changing a parameter during operation
Servo cycle time	100 μ s
Profile generator	Trapezoidal velocity profile
Encoder input	A/B quadrature (TTL differential according to RS-422), 50 MHz; BiSS interface
Stall detection	Automatic motor stop when a programmable position error is exceeded
Limit switches	2 \times TTL per axis (programmable polarity)
Reference point switch	1 \times TTL per axis
Motor brake	1 \times TTL per axis, can be switched by software




Electrical properties	C-884.4DC / C-884.6DC
Max. output voltage*	24 V
Max. output power	240 W
Current limitation	2.5 A per axis
Interfaces and operation	C-884.4DC / C-884.6DC
Communication interfaces	TCP/IP: RJ45/Ethernet; USB: Mini-USB type B; RS-232: Sub-D 9 (m); SPI: DisplayPort
Motor connector	Sub-D 15 (f)
I/O lines	4 analog inputs (-10 to 10 V), resolution: 10-bit 4 digital inputs (5 V TTL) 4 digital outputs (5 V TTL)
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	API for C / C++ / C# / VB.NET / MATLAB / Python, drivers for NI LabVIEW
Manual control	USB interface for HID-compliant devices
Miscellaneous	C-884.4DC / C-884.6DC
Operating voltage	External power adapter 24 V / 5 A (120 W) in the scope of delivery
Max. current consumption	C-884.4DC: 11 A / C-884.6DC: 16 A
Current consumption without load	500 mA
Operating temperature range	5 to 50 °C
Mass	C-884.4DC: 1.77 kg / C-884.6DC: 1.97 kg
Dimensions	312 mm × 153.4 mm × 59.2 mm (incl. mounting rails)

* Depending on the power supply used

13.1.2 Maximum Ratings

The C-884 is designed for the following operating data:

Input on:	Maximum Operating Voltage	Operating Frequency	Maximum Current Consumption
			
Sub-D 2W2C	24 V	—	11 A

Output on:	Maximum Output Voltage 	Maximum Output Current 	Maximum Output Frequency 
Sub-D 15-pin (f): pins 2 and 9	= operating voltage	2.5 A	36 kHz (PWM)
Sub-D 15-pin (f): pins 3 and 11	5 V TTL	10 mA	36 kHz (PWM)

13.1.3 Ambient Conditions and Classifications

The following ambient conditions and classifications for the C-884 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 C-884:

- PC with Windows (7, 8, 10; 32 bit, 64 bit) or Linux operating system and at least 30 MB free memory
- Communication interface to the PC:
 - Free COM port on the PC
 - or -

- USB A socket on the PC
- or -
- RJ45 Ethernet socket on the PC
- or -
- Free access point to the network, to which the PC is connected
- C-884 with power adapter
- Mechanical system (positioner) with DC motor or voice coil drive
- Cable for connecting the controller to the PC:
 - RS-232 null modem cable, USB cable or crossover network cable for connecting the controller to the PC
 - or -
 - Straight-through network cable for connecting the controller to the PC via a TCP/IP network
- Product CD with PC software

13.3 Dimensions

Dimensions in mm. Note that the decimal places are separated by a comma in the drawings.

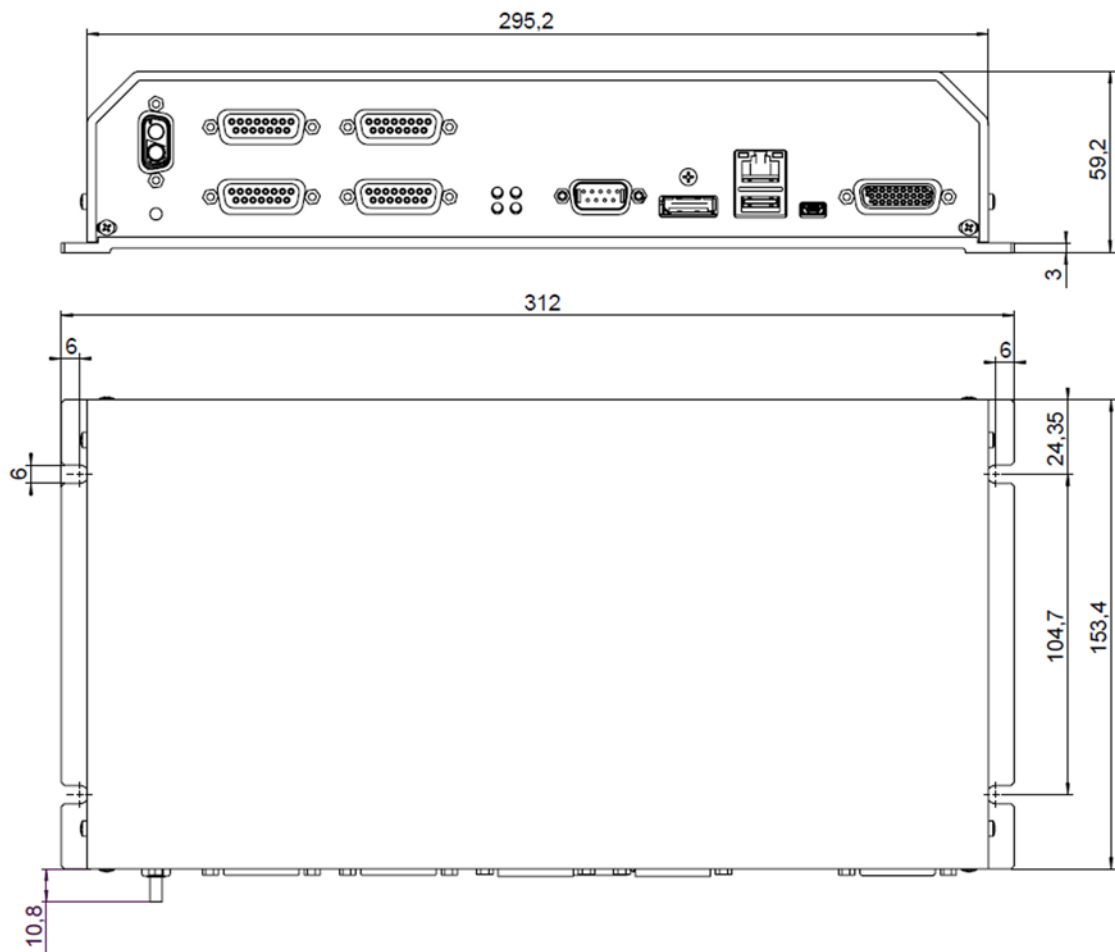


Figure 13: C-884.4DC: Dimensions

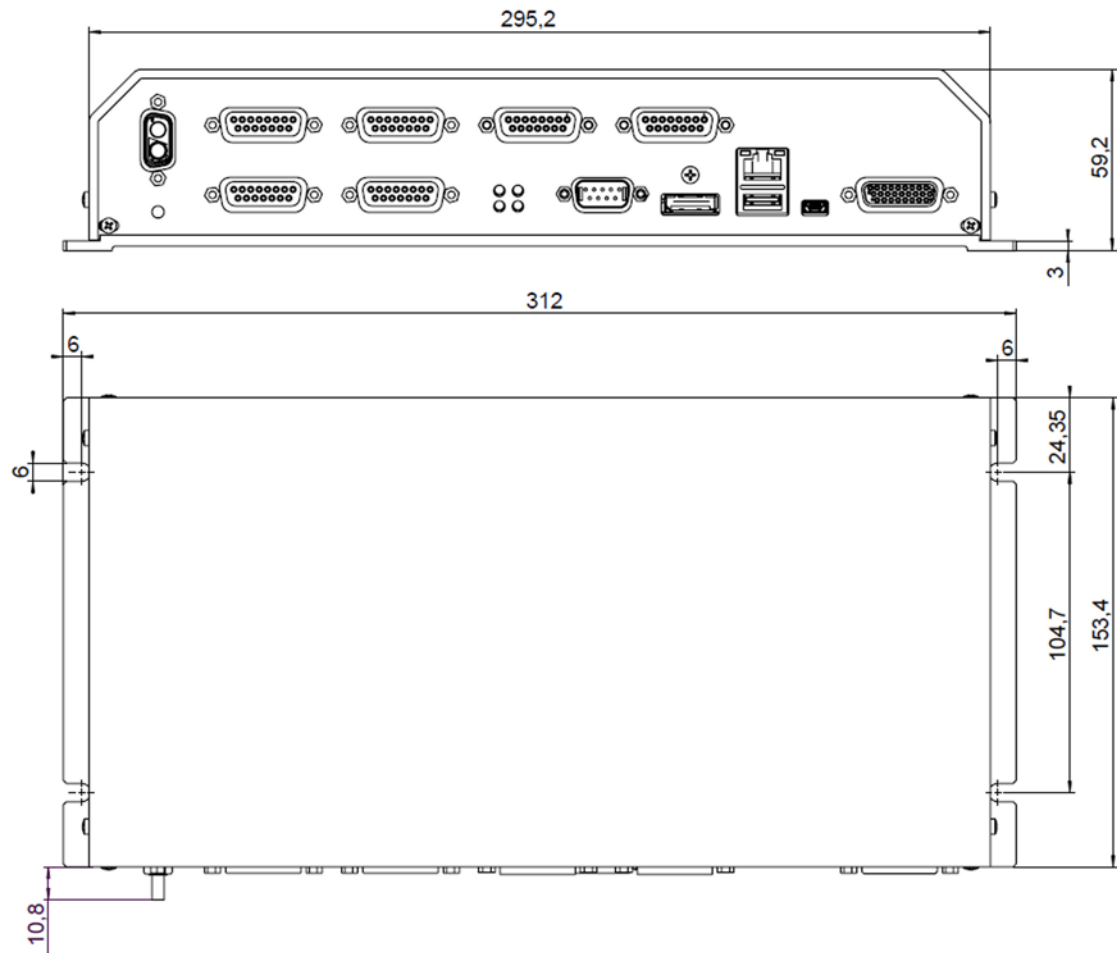


Figure 14: C-884.6DC: Dimensions

13.4 Pin Assignment

13.4.1 DC motor

Sub-D socket, 15-pin, female

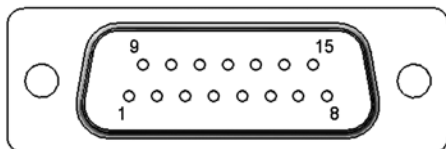


Figure 15: Front view of the socket

Pin	Signal	Function		
1	Output	Motor brake (0 or + 5 V)		
2	Output	Motor + (differential; power PWM); for positioners without PWM amplifier		
3	Output	PWM magnitude (TTL); for stages with PWM amplifier		
4	Output	+5 V		
5	Input	Positive limit switch		
6	Bidirectional	Data line for ID chip		
7	Input	Encoder:	A/B: A inverted, RS-422	BiSS: MA-
8	Input	Encoder:	A/B: B inverted, RS-422	BiSS: SL-
9	Output	Motor + (differential; power PWM); for positioners without PWM amplifier		
10	GND	Ground		
11	Output	PWM sign (TTL); for positioners with PWM amplifier		
12	Input	Negative limit switch		
13	Input	Reference point switches		
14	Input	Encoder:	A/B: A, RS-422	BiSS: MA+
15	Input	Encoder:	A/B: B, RS-422	BiSS: SL+

13.4.2 I/O

HD Sub-D socket, 26-pin, female

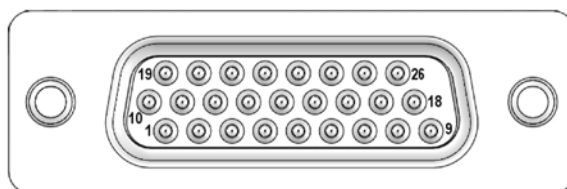


Figure 16: Front view of the socket

Pin	Function
1	Analog input 2 (-10 V to 10 V)
2	GND (analog)
3	Reserved

Pin	Function
4	Reserved
5	Reserved
6	Vcc (+5 V)
7	Digital input 3 (TTL)
8	Digital output 4 (TTL)
9	Digital output 1 (TTL)
10	Analog input 1 (-10 V to 10 V)
11	Analog input 4 (-10 V to 10 V)
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Digital input 4 (TTL)
17	Digital input 1 (TTL)
18	Digital output 2 (TTL)
19	Analog input 3 (-10 V to 10 V)
20	GND
21	Reserved
22	GND
23	Reserved
24	GND
25	Digital input 2 (TTL)
26	Digital output 3 (TTL)

13.4.3 RS-232

RS-232: Sub-D 9-pin panel plug, male

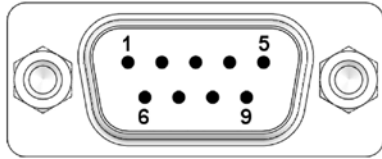


Figure 17: Front view of the panel plug

Pin	Function
1	Not connected
2	RxD (PC to controller)
3	TxD (controller to PC)
4	Not connected
5	GND
6	Not connected
7	Not connected
8	Not connected
9	Not connected

13.4.4 Power supply connection 24 V DC

Sub-D 2W2C power supply connection



Figure 18: Front view of the power supply connector

Pin	Function
Socket contact (left)	GND (power)
Pin contact (right)	Input: 24 V DC

14 Old Equipment Disposal

In accordance with EU law, electrical and electronic equipment may not be disposed of in EU member states via the municipal residual waste.

Dispose of your old equipment according to international, national, and local rules and regulations.

In order to fulfil its responsibility as the product manufacturer, Physik Instrumente (PI) GmbH & Co. KG undertakes environmentally correct disposal of all old PI equipment made available on the market after 13 August 2005 without charge.

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

Physik Instrumente (PI) GmbH & Co. KG
Auf der Roemerstr. 1
D-76228 Karlsruhe, Germany



15 EU Declaration of Conformity

For the C-884, 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

