PZ222E User Manual
E-709 Digital Piezo Controller, 1 Channel, -30 to 130 V
Release: 1.2.0 Date: 31.01.2019

This document describes the following products:

- E-709.SRG
  Digital Piezo Controller, 1 Channel, -30 to 130 V, SGS Sensor, Bench-Top
- E-709.SR
  Digital Piezo Controller, 1 Channel, -30 to 130 V, SGS Sensor, OEM Module
- E-709.PRG
  Digital Piezo Controller, 1 Channel, -30 to 130 V, Piezoresistive Sensors, Bench-Top
- E-709.PR
  Digital Piezo Controller, 1 Channel, -30 to 130 V, Piezoresistive Sensors, OEM Module
- E-709.CRG
  Digital Piezo Controller, 1 Channel, -30 to 130 V, Capacitive Sensor, Bench-Top, 10 W peak output power
- E-709.CR
  Digital Piezo Controller, 1 Channel, OEM Module, -30 to 130 V, Capacitive Sensor, 10 W peak output power
- E-709.CHG
  Digital Piezo Controller, 1 Channel, -30 to 130 V, Capacitive Sensor, Bench-Top, 50 W peak output power

Physik Instrumente (PI) GmbH & Co. KG, Auf der Roemerstrasse 1, 76228 Karlsruhe, Germany
Phone +49 721 4846-0, Fax +49 721 4846-1019, Email info@pi.ws, www.pi.ws

For the E-709, an EU Declaration of Conformity has been issued in accordance with the following European directives:

Low Voltage Directive
EMC Directive
RoHS Directive

The applied standards certifying the conformity are listed below.

Safety (Low Voltage Directive): EN 61010-1
EMC: EN 61326-1
RoHS: EN 50581

If an electrical operating device is designed to be integrated in another electrical operating device: The operator is responsible for a standards compliant integration of the electrical device into the overall system.

The patents owned by PI can be found in our patent list.

First printing 31.01.2019
Document Number PZ222E BRo, Release 1.2.0
E-709_User_PZ222E120.doc

Contents

1 About this Document

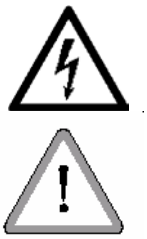
This user manual contains information on the intended use of the E-709. It assumes that the reader has a fundamental understanding of basic servo systems as well as motion control concepts and applicable safety procedures.

In this document, the E-709 Digital Piezo Controller, 1 Channel, -30 to 130 V is also referred to as "E-709" or "controller".

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

1.1 Symbols and Typographic Conventions

The notes and symbols used in this manual have the following meanings:



WARNING

Calls attention to a procedure, practice or condition which, if not correctly performed or adhered to, could result in injury or death.

NOTICE

Calls attention to a procedure, practice, or condition which, if not correctly performed or adhered to, could result in damage to equipment.

INFORMATION

Provides additional information or application hints.

Warning signs affixed to the product that refer to detailed information in this manual.

Symbol for the protective earth conductor, affixed to the product.

1.2 Other Applicable Documents

Description	Document
SPI interface of E-709 controller	E709T0002 Technical Note
Guide to Grounding and Shielding	A000T0074 Technical Note
GCS driver set for use with NI LabVIEW software	PZ225E Software Manual
Analog controller GCS driver set for use with NI LabVIEW software	PZ181E Software Manual
PI GCS 2.0 DLL	SM151E Software Manual
GCS array data format description	SM146E Software Manual
PIMikroMove	SM148E Software Manual
PI Update Finder: Search and download updates	A000T0028 Technical Note
PI Update Finder: Updating PC without Internet connection	A000T0032 Technical Note

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

1.3 Downloading Manuals

If a manual is missing or problems occur with downloading:

➔ Contact our customer service department (p. 246).

Access to the manuals is protected by a password. Protected contents are only displayed on the website after entering the access data.

1.3.1 Obtain 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 Find the access data for the download of protected contents in the section "User login for software download" in the *Release News*. Possibilities for the provision:

- Link to a registration page for request of access data
- Direct display of user name and password

5 If the access data have to be requested via a registration page:

5.1 Follow the link in the *Release News*.

5.2 Enter the information required for registration.

5.3 Click **Show login data**.

5.4 Find the user name and the password in the browser window.

1.3.2 Downloading Manuals

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

➔ Click the links on the registration page to move to the contents for your product, and log in there with the access data.

General procedure:

1 Open the website www.pi.ws.

2 If access to the manuals is protected by a password:

2.1 Click **Login**.

2.2 Log in with the user name and password.

3 Click **Search**.

4 Enter the product code up to the period (“E-709”) 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:

6.1 If necessary: Scroll down the list.

6.2 If necessary: **Click Load more results** at the end of the list.

6.3 Click the corresponding product in the list.

7 Click the **Downloads** tab.

The manuals are displayed under **Documentation**.

8 Click the desired manual and save it to the hard disk of your PC or to a data storage medium.

2 Safety

2.1 Intended Use

The E-709 is a laboratory device according to DIN EN 61010. It is intended to be used in interior spaces and in an environment which is free of dirt, oil and lubricants.

The E-709 is intended for driving capacitive loads (e. g. piezo ceramic actuators).

The E-709 must not be used for purposes other than those named in this user manual. In particular, the E-709 must not be used to drive ohmic or inductive loads.

The E-709 can be used for static as well as dynamic applications.

Depending on the model, capacitive sensors or strain gauge sensors or piezoresistive sensors (semiconductor strain gauge sensors) must be used for closed-loop operation. PI stages intended for closed-loop operation already have the corresponding sensors. Other sensors can only be used with PI approval.

2.2 General Safety Instructions

The E-709 is built according to state-of-the-art technology and recognized safety standards. Improper use can result in personal injury and/or damage to the E-709.

- ➔ Only use the E-709 for its intended purpose, and only use it if it is in a good working order.
- ➔ Read the user documentation (user manuals, Technical Notes).
- ➔ Immediately eliminate any faults and malfunctions that are likely to affect safety.

The operator is responsible for the correct installation and operation of the E-709.

- ➔ Install the E-709 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- ➔ Use the supplied components (adapter) to connect the E-709 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.
- ➔ Only use cables and connections that meet local safety regulations.

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the E-709 in the case of malfunction or failure of the system. If touch voltages exist, touching the E-709 can result in serious injury or death from electric shock.

- ➔ Connect the E-709 to a protective earth conductor before start-up (p. 60).
- ➔ Do not remove the protective earth conductor during operation.
- ➔ If the protective earth conductor has to be removed temporarily (e. g. in the case of modifications), reconnect the E-709 to the protective earth conductor before starting it up again.

If an E-709 OEM model is operated without a housing, live parts will be accessible. Touching the live parts can result in serious injury or death. Electrical, magnetic and electromagnetic fields emitted by live parts can disturb the E-709 OEM module and/or the environment.

- ➔ Only operate the E-709 OEM module when it is installed in a shielded housing that securely encloses all live parts and fulfills the requirements of electromagnetic compatibility.
- ➔ Since parts of the circuit will store charge, precautions must also be taken when the E-709 OEM module is not powered. After switching off the E-709 OEM module, wait a minute before you touch parts of the E-709 OEM module to be sure that any residual voltage has dissipated.

The E-709 OEM module heats up during operation.

- ➔ Do not touch the module during operation or immediately after operation.

The E-709 contains electrostatic-sensitive devices (ESD) that can be damaged if handled improperly.

- ➔ Avoid touching assemblies, pins and PCB traces.
- ➔ Before you touch the E-709, discharge yourself of any electric charges, e.g., by wearing an antistatic wrist strap.
- ➔ Only handle and store the E-709 in environments that dissipate existing static charges to earth in a controlled way and prevent electrostatic charges (ESD protected workstation or electrostatically protected area, in short EPA).
- ➔ Before cleaning, disconnect the E-709 from the power source by removing the mains plug.

2.3 Organizational Measures

User documentation (user manual, Technical Notes):

- ➔ Always keep this user documentation available by the E-709.
- ➔ The latest versions of the user documentation are available from PI.
- ➔ Add all information given by the manufacturer to the user documentation, for example supplements or Technical Notes.
- ➔ If you pass the E-709 on to other users, also turn over the user documentation as well as other relevant information provided by the manufacturer.
- ➔ Only use the device on the basis of the complete user documentation. Missing information due to an incomplete user documentation can result in serious or fatal injury as well as property damage.
- ➔ Only install and operate the E-709 after having read and understood this user manual.

2.4 Personnel Qualification

The E-709 may only be installed, started up, operated, maintained and cleaned by authorized and qualified staff.

3 Product Description

3.1 Model Overview

Order Number	Description
E-709.SRG	Digital Piezo Controller, 1 Channel, -30 to 130 V, SGS-Sensor, Bench-Top
E-709.SR	Digital Piezo Controller, 1 Channel, OEM Module, -30 to 130 V, SGS-Sensor
E-709.PRG	Digital Piezo Controller, 1 Channel, -30 to 130 V, Piezoresistive Sensors, Bench-Top
E-709.PR	Digital Piezo Controller, 1 Channel, OEM Module, -30 to 130 V, Piezoresistive Sensors
E-709.CRG	Digital Piezo Controller, 1 Channel, -30 to 130 V, Capacitive Sensor, Bench-Top, 10 W peak output power
E-709.CR	Digital Piezo Controller, 1 Channel, OEM Module, -30 to 130 V, Capacitive Sensor, 10 W peak output power
E-709.CHG	Digital Piezo Controller, 1 Channel, -30 to 130 V, Capacitive Sensor, Bench-Top, 50 W peak output power

3.2 Product View

3.2.1 E-709.SRG, .PRG, .SR, .PR

Figure 1: Operating elements of E-709.SRG and E-709.PRG

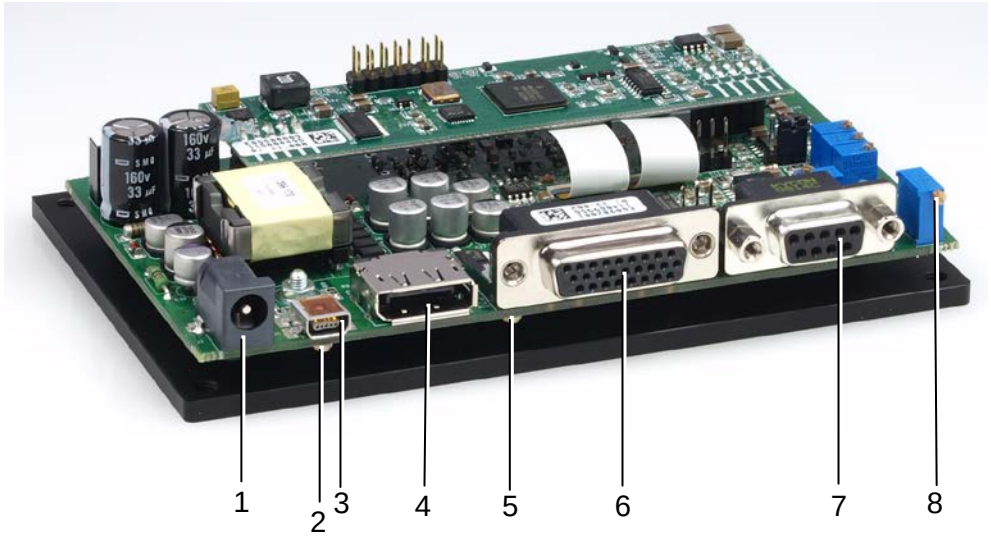


Figure 2: Operating elements of E-709.SR and E-709.PR

Name	Position with .SR and .PR	Function
Zero	8	Trim-pot for sensor zero-point adjustment. Do not use since adjustment is done before delivery. If readjustment of the sensor zero-point is necessary, use the AutoZero functionality provided by the E-709 firmware.

Name	Position with .SR and .PR	Function
PZT & Sensor -30 to 130 V 	7	Sub-D 9 (f) socket for piezo stage; carries the voltage for the piezo actuator (-30 to 130 V) and the signals of the SGS or piezoresistive sensor in the mechanics. See p. 257 for pinout. Stages equipped with Lemo connectors can be connected via the included E-709.03 Lemo to Sub-D 9 (m) adapter.
I/O	6	HD-Sub-D 26 (f) socket with lines for multiple purposes: ■ RS-232 serial connection to host PC ■ Analog input (connect a control-signal source or an external sensor) ■ Analog output (can be used to monitor the axis position or for controlling an external amplifier) ■ Monitor of piezo output voltage ■ Digital I/O lines ■ Status lines (on target, overflow, servo) ■ "ENA" enable signal (switches E-709 on/off) See "I/O Socket of E-709.SRG, .SR, .PRG, .PR, .CRG, .CR" (p. 260) for pinout. The RS-232 lines are accessible via the included E709B0002 RS-232 adapter. The analog input and output lines are accessible via the included E-709.04 adapter cable. To make the other lines accessible, you can order accessories (p. 22).
OFL LED (dark/red)	5	Overflow indicator, for closed-loop operation (servo ON) only. Red light indicates that the power amplifier is so near the end of its range that it is no longer able to follow input changes properly. In this case, readjustment of the sensor zero-point is necessary, using the AutoZero functionality provided by the E-709 firmware.
SPI	4	SPI interface (slave) for fast communication; primarily designed for transferring actual values from and control values to the E-709. See the E709T0002 Technical Note for a detailed description.
 / USB	3	Universal Serial Bus (USB Mini-B (m) socket) for connection to host PC. See "USB Connection" (p. 63) for more information.

Name	Position with .SR and .PR	Function
Status LED (green/red)	2	Power-on and error indicator: ■ Green light indicates that the device is powered on. ■ Red light indicates error (i.e. non-zero error code). After error code was queried and cleared using the ERR? command (p. 151), the LED color changes back to green.
24 V DC	1	Connector for power supply. See "24 V DC Socket" (p. 266) for pinout. Because grounding is not assured over the power connection, the E-709 must be connected to a protective earth conductor as described in "Installing the E-709" (p. 60).

3.2.2 E-709.CRG and .CR

Figure 3: Operating elements of E-709.CRG

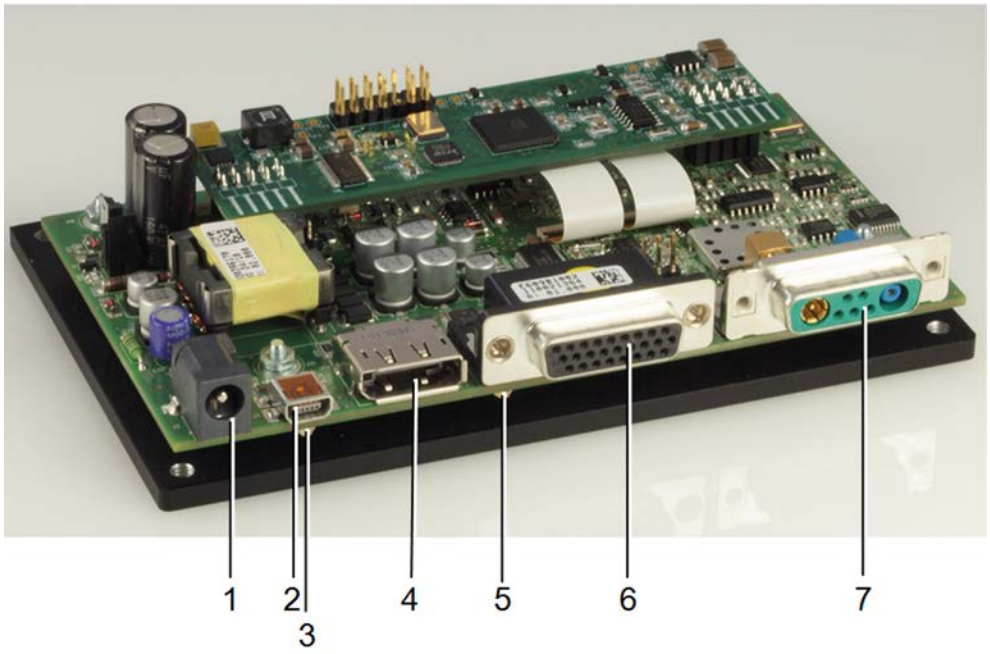


Figure 4: Operating elements of E-709.CR

Name with .CRG	Position with .CR	Function
PZT & Sensor -30 to 130 V	7	Sub-D mix 7W2 socket for piezo stage; carries the voltage for the piezo actuator (-30 to 130 V) and the signals of the capacitive sensor in the mechanics. See p. 258 for pinout.

Name with .CRG	Position with .CR	Function
I/O	6	HD-Sub-D 26 (f) socket with lines for multiple purposes: ■ RS-232 serial connection to host PC ■ Analog input (connect a control-signal source or an external sensor) ■ Analog output (can be used to monitor the axis position or for controlling an external amplifier) ■ Monitor of piezo output voltage ■ Digital I/O lines ■ Status lines (on target, overflow, servo) ■ "ENA" enable signal (switches E-709 on/off) See "I/O Socket of E-709.SRG, .SR, .PRG, .PR, .CRG, .CR" (p. 260) for pinout. The RS-232 lines are accessible via the included E709B0002 RS-232 adapter. The analog input and output lines are accessible via the included E-709.04 adapter cable. To make the other lines accessible, you can order accessories (p. 22).
OFL	5	LED (dark/red) Overflow indicator, for closed-loop operation (servo ON) only. Red light indicates that the power amplifier is so near the end of its range that it is no longer able to follow input changes properly. In this case, readjustment of the sensor zero-point is necessary, using the AutoZero functionality provided by the E-709 firmware.
SPI	4	SPI interface (slave) for fast communication; primarily designed for transferring actual values from and control values to the E-709. See the E709T0002 Technical Note for a detailed description.
Status	3	LED (green/red) Power-on and error indicator: ■ Green light indicates that the device is powered on. ■ Red light indicates error (i.e. non-zero error code). After error code was queried and cleared using the ERR? command, the LED color changes back to green.
	2	Universal Serial Bus (USB Mini-B (m) socket) for connection to host PC. See "USB Connection" (p. 63) for more information.

Name with .CRG	Position with .CR	Function
24 V DC	1	Connector for power supply. See "24 V DC Socket" (p. 266) for pinout. Because grounding is not assured over the power connection, the E-709 must be connected to a protective earth conductor as described in "Installing the E-709" (p. 60).

3.2.3 E-709.CHG

Figure 5: Operating elements of E-709.CHG

Name	Function
Sync In	Lemo socket for synchronization of multiple E-709.CHGs. Connects to previous E-709 ("Synchronizing Multiple E-709.CHGs", p. 52). See p. 258 for pinout.
Sync Out	Lemo socket for synchronization of multiple E-709.CHGs. Connects to next E-709 ("Synchronizing Multiple E-709.CHGs", p. 52). See p. 258 for pinout.
Analog In	SMB socket for analog control input, 0 to 10 V (in open-loop and closed-loop operation; handled by E-709 as input signal channel 2). The E-692.SMB adapter cable SMB/BNC can be ordered as an accessory (p. 22). The signal is also available via pin 19 of the I/O socket (pinout on p. 261). **Either** connect the signal via the SMB socket **or** via pin 19 of the I/O socket. Do not connect signals to both lines.
Monitor Out	SMB socket for analog output, 0 to 10 V. Can be used to monitor the axis position (default) or for controlling an external amplifier. The E-692.SMB adapter cable SMB/BNC can be ordered as an accessory (p. 22). The signal is also available on pin 20 of the I/O socket. See pinout of I/O socket for further details (p. 261).

Name	Function
PZT & Sensor -30 to 130 V 	Sub-D special socket 7W2 for piezo stage; carries the voltage for the piezo actuator (-30 to 130 V) and the signals of the capacitive sensor in the mechanics. See p. 258 for pinout.
OFL	LED (dark/red) Overflow indicator, for closed-loop operation (servo ON) only. Red light indicates that the power amplifier is so near the end of its range that it is no longer able to follow input changes properly. In this case, readjustment of the sensor zero-point is necessary, using the AutoZero functionality provided by the E-709 firmware.
SPI	SPI interface for fast communication; primarily designed for transferring actual values from and control values to the E-709. See the E709T0002 Technical Note for a detailed description.
Status	LED (green/red) Power-on and error indicator: ■ Green light indicates that the device is powered on. ■ Red light indicates error (i.e. non-zero error code). After error code was queried and cleared using the ERR? command, the LED color changes back to green.
	Universal Serial Bus (USB Mini-B (m) socket) for connection to host PC. See "USB Connection" (p. 63) for more information.
I/O	HD-Sub-D 26 (f) socket with lines for multiple purposes: ■ Analog input; also available via the Analog In SMB socket (see above). **Either** connect the signal via the SMB socket **or** via the I/O socket. Do not connect signals to both lines. ■ Digital I/O lines ■ Status lines (on target, overflow, servo) ■ Monitor of piezo output voltage ■ Analog output; also available via the Monitor Out SMB socket (see above). ■ "ENA" enable signal (switches E-709 on/off) See p. 261 for pinout and detailed descriptions.
RS-232	Sub-D 9 (m) panel plug for connection to host PC. See p. 259 for pinout and "RS-232 Serial Connection" (p. 65) for communication details.
24 V DC	Connector for power supply. See "24 V DC Socket" (p. 266) for pinout. Because grounding is not assured over the power connection, the E-709 must be connected to a protective earth conductor as described in "Installing the E-709" (p. 60).

3.3 Scope of Delivery

Unpack the E-709 Digital Piezo Controller, 1 Channel, -30 to 130 V with care. Compare the contents against the items covered by the contract and against the packing list.

The following components are included in the scope of delivery:

- E-709 Digital Piezo Controller, 1 Channel, -30 to 130 V, according to your order
- Power supply—with bench-top models only:
  - With E-709.xRG:
    C-501.24050H wide-range, 24 V power supply, 50 W, with line cord
  - With E-709.CHG:
    000023194 wide-range, 24 V power supply, 120 W, with line cord
- 000036360 USB cable (3 m, USB-A (m)/USB Mini-B (m)) for PC connection
- C-815.34 RS-232 cable for connecting controller and host PC (null-modem cable)
- With E-709.xRG and .xR models only:
  E709B0002 RS-232 adapter HD-Sub-D 26 (m) to Sub-D 9 (m) for PC connection via RS-232
- With E-709.SRG, .SR, .PRG and .PR models only
  E-709.03 adapter LEMO to Sub-D 9 (m) to connect mechanics equipped with LEMO connectors
- With E-709.xRG and .xR models only
  E-709.04 adapter cable HD-Sub-D 26 (m) to 2 x BNC (m) for analog input and analog output (adapter cable SMB/BNC available as an optional accessory (p. 22))
- E-709.CD distribution CD, containing host software (see "Software Description" (p. 39)) and all manuals as PDF files
- E-709 User Manual for E-709 in printed form (PZ222E, this document)

If parts are missing or you notice signs of damage, contact your PI representative or write to info@pi.ws immediately.

Save all packing materials in case the product needs to be shipped again.

If controller and mechanics were ordered together, make sure a label with the serial number of the mechanics is affixed to the controller housing.

3.4 Accessories

Contact your PI representative or write an e-mail to info@pi.ws if you need the following additional components:

Order Number	Description
E-709.01	Adapter HD-Sub-D 26 (m) to Sub-D 9 (m), screwless mini block and solder pins, 0.5 m cable included. Splits up the lines of the "I/O" socket into analog input and output, digital I/O, status and enable signals and a standard RS-232 interface; see "E-709.01 Adapter for "I/O" Socket" (p. 263) for further information.
E-709.02	Adapter Cable HD-Sub-D 26 (m) to open leads, 1 m. Makes the lines of the "I/O" socket available separately; see "E-709.02 Adapter Cable for "I/O" Socket" (p. 265) for wire assignment and "I/O Socket" (p. 260 or p. 261) for pinout.
E-709.03	Adapter Lemo to Sub-D 9 (m). Required to connect mechanics equipped with LEMO connectors to the "PZT & Sensor" socket (p. 257) of E-709 models for SGS and piezoresistive sensors.
E-709.04	Adapter cable HD-Sub-D 26 (m) to 2 x BNC (m), 1 m. Makes the analog input and output lines of the "I/O" socket (p. 260 or p. 261) available separately via two BNC connectors (male). With E-709.xRG and .xR models, the RS-232 interface of the E-709 cannot be used when using this adapter cable.
E-692.SMB	Adapter cable SMB to BNC (m), 1.5 m. Can be used with the E-709.CHG model to connect to the analog input and output lines. Can also be used with the E-709.04 adapter cable. In this case, for every BNC connection a standard BNC female to female adapter is required (not included).
K040B0151	Sync cable. If multiple E-709.CHGs are used, their sensor signals can be synchronized. To synchronize n+1 E-709.CHGs you need n synchronization cables. See "Synchronizing Multiple E-709.CHGs" (p. 52) for more information.
C-501.24050H	Wide-range, 24 V power supply, 50 W, can be used with line voltages from 100 VAC to 240 VAC at 50 or 60 Hz

Order Number	Description
E-500.ACD	CD with LabVIEW analog driver set, available free of charge upon request. The E-709 can be controlled via a signal connected to its analog input line (see "Using the Analog Input" (p. 67) for more information). This signal can, for example, be generated using a D/A board in a PC. PI offers a LabVIEW driver set which can be used with certain D/A boards. It is compatible with the PI General Command Set (GCS) LabVIEW driver set available for all newer controllers from PI. The PI LabVIEW drivers support all D/A converter boards from National Instruments that are compatible with DAQmx8.3. LabVIEW compatibility is given from version 7.1 upwards.

3.5 Axes, Channels, Functional Elements

The following list contains the items that can be accessed with commands of the PI General Command Set (GCS).

- Logical axis: one axis, the default identifier is 1.
  A logical axis is an axis of a linear, orthogonal coordinate system and represents a basic direction of motion in the E-709 firmware. All motion of the mechanics is commanded for logical axes.
  The axis identifier can be changed using the SAI command (p. 167). It can consist of up to 4 characters; valid characters are 123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ
  SAI changes the value of the Axis Name parameter, ID 0x07000600, in volatile memory (RAM). To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 202). Changes not saved with WPA will be lost when the controller is powered down. You can ask with SAI? (p. 168) for the current valid axis identifier.
  The number of axes is given by the Number Of System Axes parameter, ID 0x0E000B02.
  See "Processing Steps" (p. 63) for more information regarding the interrelation of logical axes and input / output signal channels.

- Input signal channels: two channels, the identifiers are 1 and 2.
  1 identifies the line for the sensor integrated in the mechanics (connects to the "PZT & Sensor" socket (p. 257 or p. 258)).
  2 identifies the analog input line (connects to pin 19 of the "I/O" socket (p. 260 or p. 261) or, with E-709.CHG models, also to the "Analog In" SMB connection (p. 19)). This line can be used for control value generation in analog command mode or for an external sensor (see "Using the Analog Input" (p. 67) for more information).
  The number of input signal channels is given by the Number Of Input Channels parameter, ID 0x0E000B00. The Number Of Sensor Channels parameter, ID 0x0E000B03, gives the number of sensors

which are directly integrated in the mechanics (i.e. line 1 with the E-709) and hence is always less than or equal to the number of input signal channels.

- Output signal channel: two channels, the identifiers are 1 and 2.
  1 identifies the line that carries the voltage for the piezo actuator in the mechanics (connects to the "PZT & Sensor" socket (p. 257 or p. 258)).
  2 identifies the analog output line (connects to pin 20 of the "I/O" socket (p. 260 or p. 261) or, with E-709.CHG models, also to the "Monitor Out" SMB connection (p. 19)). This line can be used to monitor the axis position or for controlling an external amplifier (see "Using the Analog Output" (p. 67) for more information).
  The number of output signal channels is given by the Number Of Output Channels parameter, ID 0x0E000B01. The Number Of Piezo Channels parameter, ID 0x0E000B04, is always less than or equal to the number of output signal channels and gives the number of piezo voltage amplifiers dedicated to the actuators in the mechanics. For the E-709, the values of both parameters are identical because there are no output signal channels other than that for the piezo voltage of the mechanics.

- Digital output lines: two lines, the identifiers are 1 and 2.
  1 identifies the Digital_OUT_1 line (pin 11) and 2 the Digital_OUT_2 line (pin 12) of the "I/O Socket" (p. 260 or p. 261).
  The number of digital output lines is given by the Number Of Trigger Outputs parameter, ID 0x0E000B05 and can also be queried with the TIO? command (p. 181).
  See "External Triggering / Signaling" (p. 85) for more information.

- Digital input lines: one line, the identifier is 1 (Digital_IN_1, pin 10 of the "I/O Socket" (p. 260 or p. 261)).
  The number of digital input lines can be queried with the TIO? command (p. 181).
  See "External Triggering / Signaling" (p. 85) for more information.

- Wave generator: one generator, the identifier is 1
  The number of wave generators can be queried with the TWG? command (p. 186). See "Wave Generator" (p. 91) for more information.

- Wave tables (memory tables for waveform data): 6 tables with a total of 16306 points, the identifiers are 1 to 6
  The number of wave tables is given by the Number of Waves parameter, ID 0x1300010A. See "Wave Generator" (p. 91) for more information.

- Data recorder tables (memory tables for recorded data): up to 4 tables with a total of 4096 points, the identifiers start with 1 and

continue sequentially up to the number of tables.
The number of data recorder tables can be set via the Data Recorder Chan Number parameter, ID 0x16000300. The maximum number of tables is limited by the Max Number Of Data Recorder Channels parameter, ID 0x16000100.
See "Data Recording" (p. 74) for more information.

- Whole system: the E-709 as a whole, the identifier is 1

3.6 Important Components of the Firmware

The firmware comprises the ASCII command set and the controller parameters and also includes some special features. For version information and updates see "Firmware Update" (p. 63).

- ASCII Commands:

  The E-709 understands the PI General Command Set (GCS; version 2.0).
  The PI General Command Set (GCS) is supported by a wide range of PI systems. This command set is well-suited for positioning tasks with one or more axes. The command set itself is independent of the specific hardware (controller or attached stages).

  Commands are used, for example, to set the control mode, to initiate motion of the mechanics and to query system and motion values.
  See "GCS Commands" (p. 123) for more information.

- Controller Parameters:

  The key features of the controller are mirrored in parameters. They represent the hardware basics and the calibration setup of the system. Some of the parameters are protected so that their factory settings cannot be changed, other parameters can be modified by the user to adapt the system to the individual application. See "Controller Parameters" (p. 223) for more information.

  Note that PI records data files of every E-709 controller calibrated at the factory for easy restoration of original parameter settings after shipping.

- Command Levels:

  "Command levels" determine the availability of commands and the write access to the controller parameters. Changing the current active command level may require a password and can be done with the CCL command (p. 138).

- Special Features:

  Wave generator: The axis can be controlled by a "wave generator" which outputs user-specified patterns, so-called "waveforms". This feature is especially important in dynamic applications which require periodic, synchronous motion of the axis. See "Wave Generator" (p. 91) for more information.

  Data recorder: The E-709 comprises a real-time data recorder. It is able to record several input and output signals (e.g. current position, sensor input, output voltage) from different data sources (e.g. logical axes or input and output signal channels). See "Data Recording" (p. 74) for more information.

- Control Algorithm for Closed-Loop Operation:

  For better position accuracy and performance, the E-709 can be operated in closed-loop mode. A PID servo-control algorithm (with sensor feedback) will then apply corrections to the control value. See "Axis Motion" (p. 26) and "Servo-Controller Dynamic Calibration" (p. 108) for more information.

3.7 Axis Motion

The E-709 controls the motion of one logical axis of a mechanics.

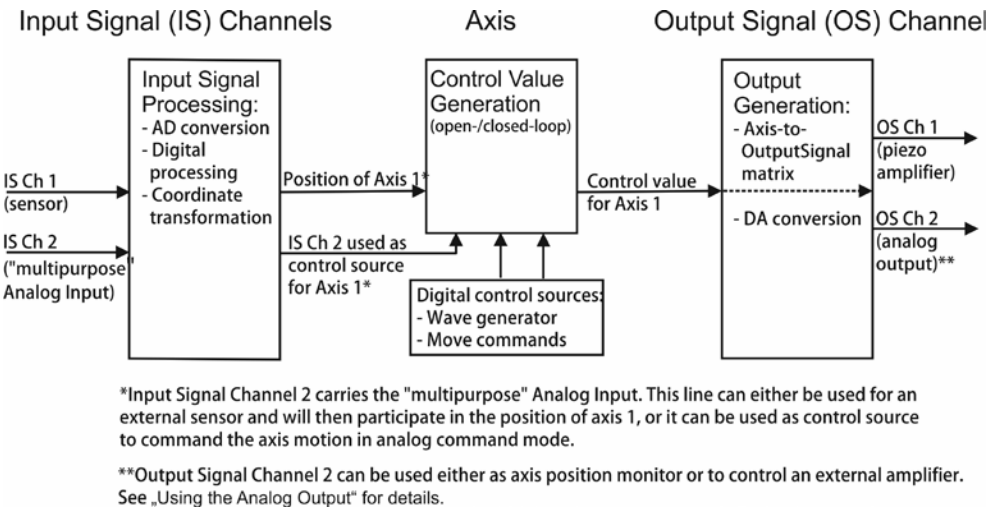


Figure 6: Processing Overview for E-709

The selected control mode (p. 29) determines how the E-709 interpretes the control value for the axis motion:

- Open-loop operation: The piezo output voltage is commanded.
- Closed-loop operation: The axis position is commanded.

The control value for axis motion can result from digital or analog control sources. The selection is made via the Current Command Mode parameter (ID 0x06000500): 0 = digital; 2 = analog. The parameter value can be set with SPA (p. 170). Find more information on how to change parameter values in "Parameter Handling" (p. 223).

3.7.1 Digital Command Mode

In digital command mode, the following control sources are available:

- Move commands: MOV (p. 161) and MVR (p. 162) in closed-loop operation; SVA (p. 176) and SVR (p. 180) in open-loop operation; IMP (p. 160) and STE (p. 175) for both control modes (see below for examples)
- Wave generator output for periodic motion in either control mode (see "Wave Generator" (p. 91) for more information and examples). Wave generator output will overwrite the control value given by a move command.

The following examples can be used in a terminal, e.g. in the *Command Entry* window of PIMikroMove or in the PI Terminal.

A first test that can be made after unpacking your new system: Install, interconnect and power on the system as described in this user manual. Then perform a first open-loop move in digital command mode and check the voltage and position values:

Command String to Send	**Action Performed**
SPA? 1 0x06000500	Check if the Current Command Mode parameter is set to "digital": the response must be 1 0X06000500=0 Note: If the response is "2" (= analog command mode), switch to digital command mode by sending SPA 1 0x06000500 0
SVO?	Check servo-control state. The response should be "1=0" which means that axis 1 is in open-loop operation, i.e. there is no correction of drift or other effects.

Command String to Send	Action Performed
VOL?	Check the current output voltage. The response should be 0 V unless otherwise preset in the system configuration.
POS?	Check the current position of axis 1. The current position value should be approximately 10 % of the travel range (in µm), due to the calibration settings of the system.
SVR 1 10	Send this command five times. With each command, the piezo output voltage for axis 1 increases relatively by 10 V.
VOL?	Check the current output voltage. It should be 50 V.
POS?	The current position value should be approximately half the maximum position.

If no load is applied to the piezo stage or if the system was calibrated at the factory with a load equal to the current one, you can perform a first closed-loop move:

Command String to Send	Action Performed
SVO 1 1	Set servo-control on (closed-loop operation) for axis 1; this also writes the current axis position to the target register, to avoid jumps of the mechanics.
POS?	Get current position of axis 1.
MOV 1 10	Axis 1 moves to an absolute position of 10 µm.
POS?	The current position of axis 1 should be exactly 10 µm.
MVR 1 14	Axis 1 moves relative by 14 µm.
POS?	The new position should be exactly 24 µm

3.7.2 Analog Command Mode

In analog command mode, axis motion is controlled via an analog input voltage. 0 to 10 V input correspond to the following target values:

- Open-loop operation: -30 to 130 V piezo output voltage
- Closed-loop operation: minimum position to maximum position of the axis (default setting)

See "How to work with the Analog Input" (p. 67) for more information.

3.8 Control Details

3.8.1 Control Modes

The E-709 provides the following control modes:

- Open-loop control (also referred to as "servo-off state" in this document): sensor feedback is not used
- Closed-loop control (also referred to as "servo-on state" in this document): sensor feedback participates in the control value generation.
  A PID servo controller (p. 32) is used to generate corrections to the control value.

The control mode can be selected with the SVO command (p. 178). By default, open-loop control is active after power-on. Using the Power Up Servo On Enable parameter (ID 0x07000800), you can set up the device to start with closed-loop control.

In open-loop operation, and in closed-loop operation, the two notch filters integrated in the E-709 are used (p. 33), and the velocity can be limited by a slew rate setting (closed-loop operation: ID 0x07000200; open-loop operation: ID 0x07000201).

Open-Loop Control:

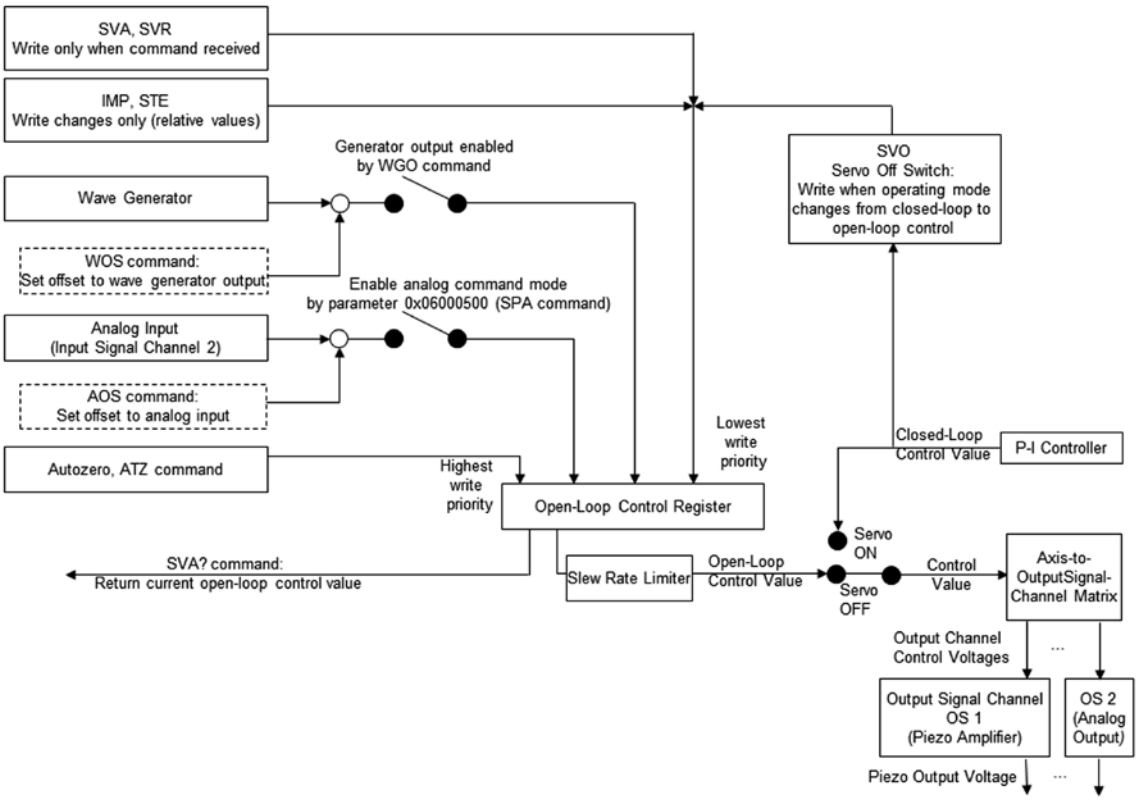


Figure 7: Control sources for the axis, in open-loop operation

Closed-Loop Control:

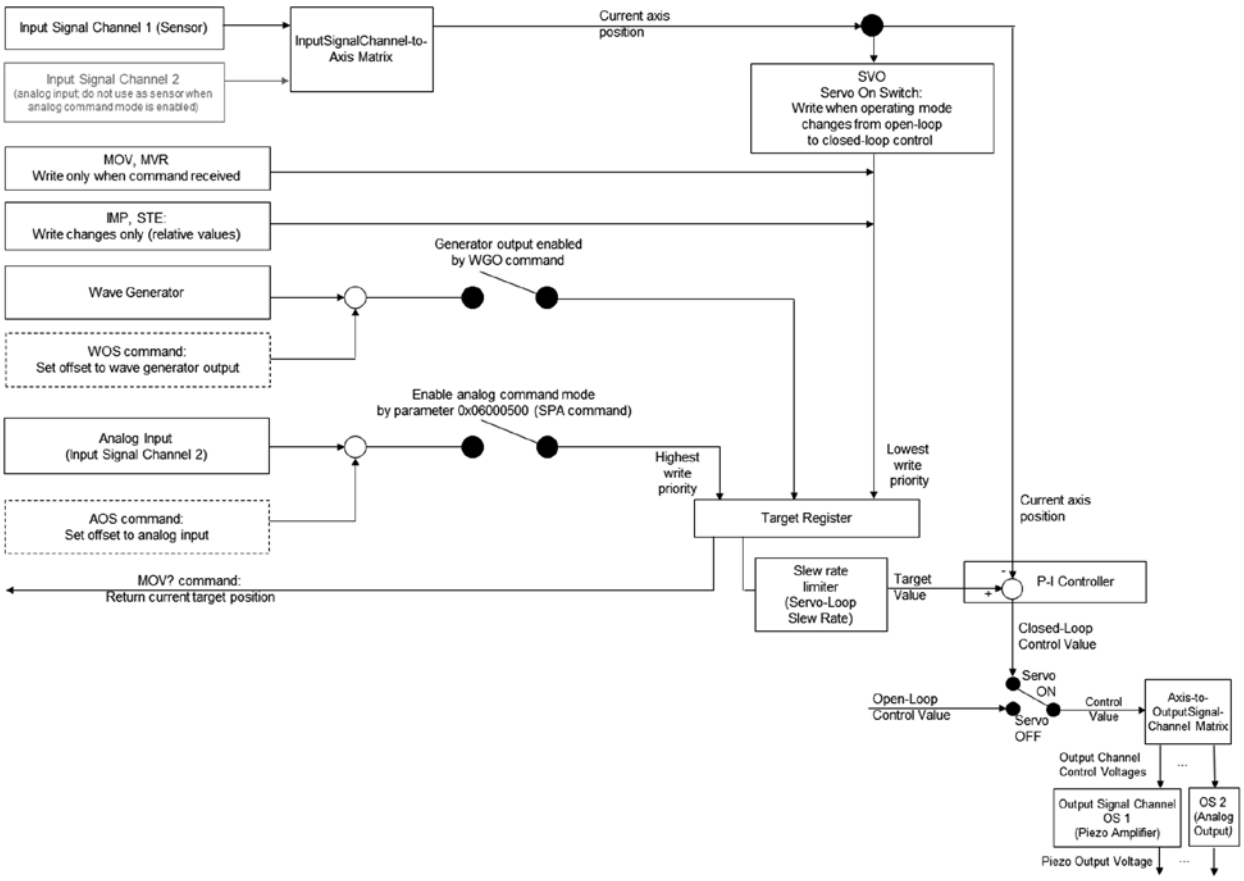


Figure 8: Control sources for the axis, in closed-loop operation

3.8.2 PID Algorithm for Closed-Loop Operation

A PID control algorithm is used in closed-loop operation.

The PID algorithm has the following structure:

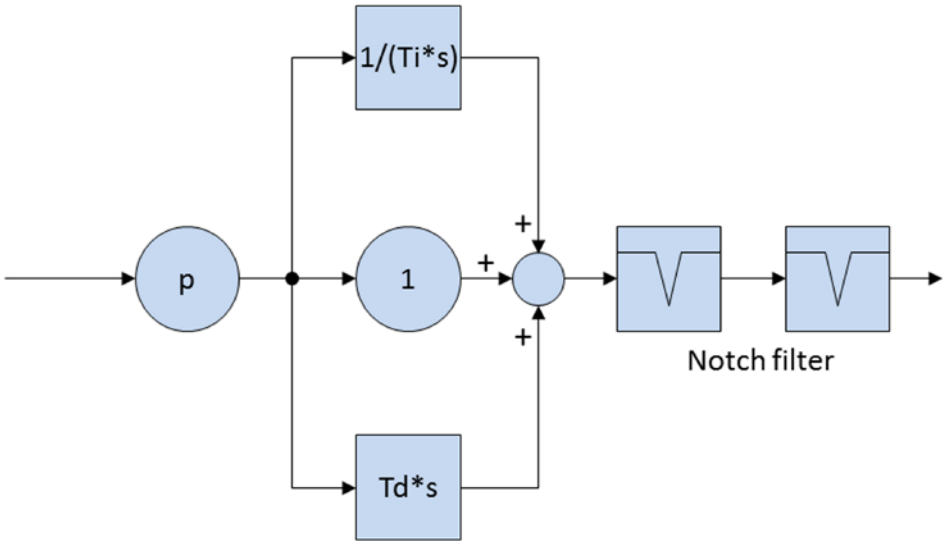


Figure 1: Structure of the PID algorithm; two notch filters are available

The PID algorithm can be configured with the following parameters:

Parameter	Notes
Servo-Loop P-Term, ID 0x07000300	p constant Must be > 0. For further details, see “Servo-Controller Dynamic Tuning” (p. 108).
Servo-Loop I-Term, ID 0x07000301	Integrator time constant Ti output = Ts / Ti · ∑ input where Ts is the servo update time (parameter 0x0E000200). When the time constant Ti is zero, then the integrator is turned off. For further details, see “Servo-Controller Dynamic Tuning” (p. 108).
Servo-Loop D-Term, ID 0x07000302	Differentiator time constant Td for position control output = Td / Ts · Δinput where Ts is the servo update time (parameter 0x0E000200). Must be > 0. For further details, see “Servo-Controller Dynamic Tuning” (p. 108).

3.8.3 Notch Filters

The E-709 provides two notch filters per axis. The corrections by a notch filter take place in closed-loop operation and in open-loop operation. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanics.

The transfer function of a notch filter is as follows:

$$G(s) = k^2 \times \frac{s^2 + 2 \times \omega \times r \times s + \omega^2}{s^2 + 2 \times \omega \times k \times s + \omega^2 \times k^2}$$

Where

G(s) is the transfer function of the notch filter
k is the bandwidth of the notch filter
s is the input signal
ω is the angular frequency, with $\omega = 2*\pi*f_0$, where f_0 is the notch filter frequency in Hz
r is the notch rejection

The notch filters can be configured per axis using the following parameters:

Parameter	Notes
Notch frequency 1, ID 0x08000100	Frequency f_0 of notch filter 1 and notch filter 2, in Hz. The maximum value is: $f_{0max} = 0.45*f_{sample}$ where f_{sample} is the servo rate in Hz (1/Servo Update Time (ID 0x0e000200)) Adjusting the notch filter frequency can be useful, particularly in the case of very high loads. For further details, see “Adjusting the Notch Filter(s) in Open-Loop Operation” (p. 110).
Notch frequency 2, ID 0x08000101	
Notch Rejection 1, ID 0x08000200	Notch rejection value r for notch filter 1 and notch filter 2. 0 to 0.98 Recommended value is 0.05. A notch rejection value of 1 deactivates the notch filter. The notch rejection value determines the filter width of the notch filter, i.e. it scales the damping done by the notch filter: The greater the rejection value, the wider the frequency spectrum of the damping, but the smaller the damping effect.
Notch Rejection 2, ID 0x08000201	

Parameter	Notes
Notch Bandwidth 1, ID 0x08000300	Bandwidth k of notch filter 1 and notch filter 2 ≥ 0.1
Notch Bandwidth 2, ID 0x08000301	The notch filter bandwidth determines the effect of the low-pass filtering: The smaller the bandwidth, the smaller the low-pass filter frequency.

3.9 Input Signal Processing

The E-709 provides two channels for analog input signals—the first channel is for the signal from the position sensor in the mechanics, the second channel carries a "multipurpose" analog input. The following processing is applied to both input signal channels:

- Analog to digital conversion
- Digital processing (filtering and linearization / scaling)
- Coordinate transformation to calculate the axis position from the input signal(s)

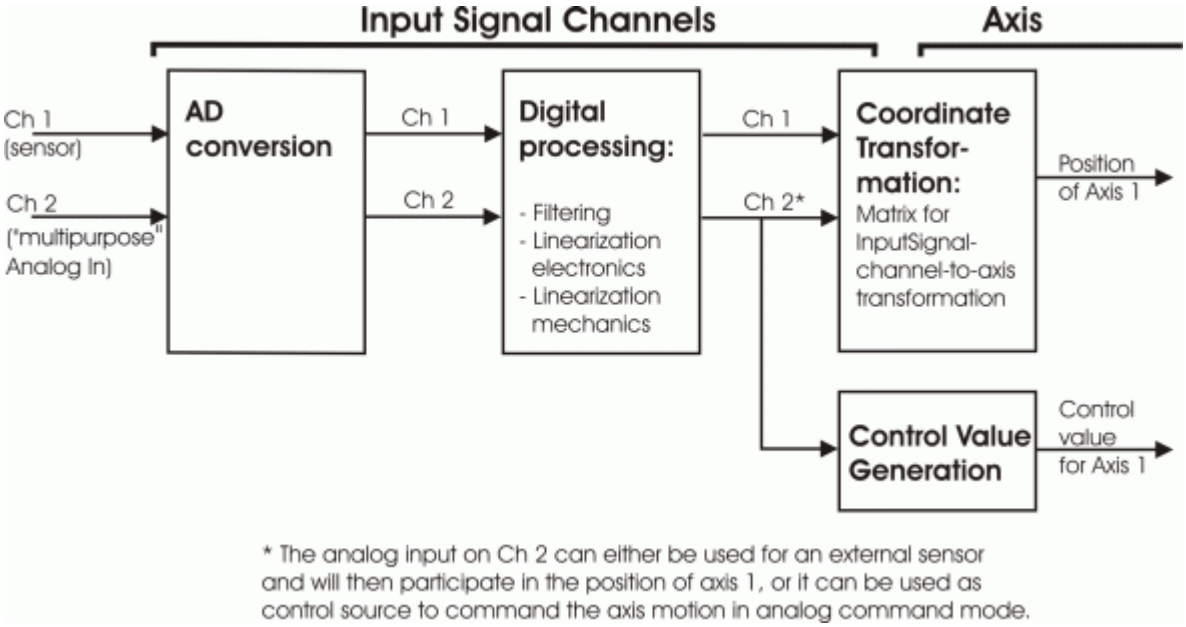


Figure 9: E-709 Input signal processing

Analog to digital conversion:

Note that in PIMikroMove, the parameters relating to the analog to digital conversion are available in the *Sensor Electronics* parameter groups in the *Device Parameter Configuration* window.

The result of the analog to digital conversion can be queried with the TAD? command (p. 180) for both channels.

Digital processing:

The digital processing of the input signals comprises the following steps:

- Digital filtering
- Electronics linearization
- Mechanics linearization

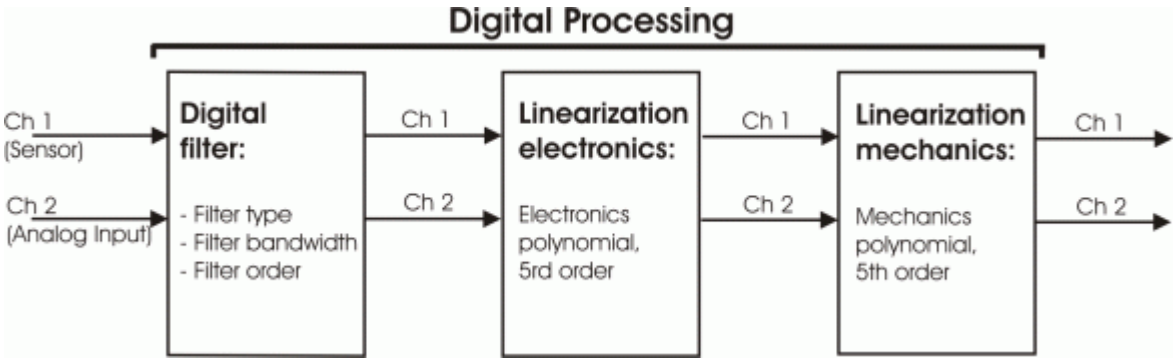


Figure 10: E-709 Digital processing of the input signals

The following parameters determine the digital filter settings:

- Digital Filter Type
  parameter ID 0x05000000
  0 = no filter
  1 = IIR low-pass filter, 2nd order
  2 = moving-average filter

- Digital Filter Bandwidth
  parameter ID 0x05000001
  Gives the frequency of the IIR low-pass filter. Only used if "Digital Filter Type" is set to "IIR low-pass filter, 2nd order".

- Digital Filter Order
  parameter ID 0x05000002
  Filter order of moving-average filter, gives the number of previous values used in determining the present output. Only used if "Digital Filter Type" is set to "moving-average filter" (for the IIR filter, the order is always 2).

In PIMikroMove, the digital filter parameters are available in the *Sensor Mechanics* parameter groups in the *Device Parameter Configuration* window.

Polynomial linearization is used to correct system performance. The form of the polynomials is as follows:

$$y = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4 + a_5 \cdot x^5$$

x – *filtered sensor ADC value*
y – *linearized sensor value*

To make the system components easily replaceable, sensor (i.e. mechanics) and electronics use separate polynomials. The coefficients of the polynomials are determined and set at the factory via the following parameters (some may presently be set to zero):

- Electronics linearization: parameters 0x03000100 to 0x03000603. They are independent of the connected mechanics and may not be changed by the user. In PIMikroMove, these parameters are available in the *Sensor Electronics* parameter groups in the *Device Parameter Configuration* window.

- Mechanics linearization: parameters 0x02000200 to 0x02000700. They depend on the connected mechanics. In PIMikroMove, these parameters are available in the *Sensor Mechanics* parameter groups in the *Device Parameter Configuration* window.
  For the sensor in the mechanics (input signal channel 1), the parameters should not be changed by the user. For the analog input line (input signal channel 2), changing the values for offset (0x02000200) and gain (0x02000300) is required to scale the analog input to suitable position values (see "Using the Analog Input" (p. 67) for more information and examples).

The TNS? command (p. 183) reports the result after the linearization for the electronics (normalized value, dimensionless), while the TSP? command (p. 185) reports the result after the linearization for the mechanics (scaled value, the unit is µm).

Coordinate transformation:

Up to two sensors can be used to monitor the position of axis 1: the sensor in the mechanics (input signal channel 1) and an additional, external sensor (input signal channel 2). The external sensor can optionally be connected to the analog input line (pin 19 of the "I/O" socket (p. 260 or p. 261) or, with E-709.CHG models, separate "Analog In" SMB connection (p. 19)). The axis position is calculated from the position values of the two input signal channels using a coordinate transformation matrix (InputSignalChannel-to-Axis matrix) which has 1 row and 2 columns:

$$(Axis1) = (a_{11} \quad a_{12}) \begin{pmatrix} InputCh1 \\ InputCh2 \end{pmatrix}$$

In equation form:

$Axis_1 = a_{11}InputCh_1 + a_{12}InputCh_2$

The matrix coefficients are given by the following parameters:

- a_{11} is given by Position From Sensor 1, parameter ID 0x07000500. This coefficient is for the sensor in the mechanics. It is determined during calibration at the factory and has a non-zero value. The preset value of the coefficient should not be changed unless the sensor is to be excluded from the position feedback of the axis, e.g. if an external sensor is connected to the analog input line (see "Using the Analog Input" (p. 67) for more information).

- a_{12} is given by Position From Sensor 2, parameter ID 0x07000501. This coefficient is for the analog input line (input signal channel 2). It should therefore be set to zero as long as no external sensor is connected or when the analog input is used for control value generation (see "Using the Analog Input" (p. 67) for more information).

In PIMikroMove, these parameters are available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

While TSP? (p. 185) reports the position values of the input signal channels, the POS? command (p. 164) reports the axis position after the matrix transformation (the unit is µm).

3.10 Output Generation

The control value for the axis is transformed to control voltage values for the output signal channels 1 and 2 via the Axis-to-OutputSignalChannel matrix. After the digital-to-analog conversion, the resulting control voltage

value of output signal channel 1 is sent to the piezo amplifier of the E-709 whose output drives the actuator in the mechanics.
The control voltage value of output signal channel 2 can be output by the analog output line (pin 20 of the "I/O" socket (p. 260 or p. 261) or, with E-709.CHG models, separate "Monitor Out" SMB connection (p. 19)) to drive an external amplifier (see "Use to Control External Amplifier" (p. 75) for more information).

Axis-to-OutputSignalChannel matrix:

$$\begin{pmatrix} OutputCh1 \\ OutputCh2 \end{pmatrix} = \begin{pmatrix} p11 \\ p21 \end{pmatrix} * (Axis1)$$

In equation form:

$OutputCh_1 = p_{11}Axis_1$
$OutputCh_2 = p_{21}Axis_1$

The matrix coefficients are given by parameters:

- p_{1i} = Driving Factor of Piezo 1, parameter ID 0x09000000,
  This coefficient is for the piezo amplifier in the E-709 which drives the piezo actuator in the mechanics.

- p_{2i} = Driving Factor of Piezo 2, parameter ID 0x09000001
  This coefficient is for the analog output line.

In PIMikroMove, these parameters are available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

INFORMATION

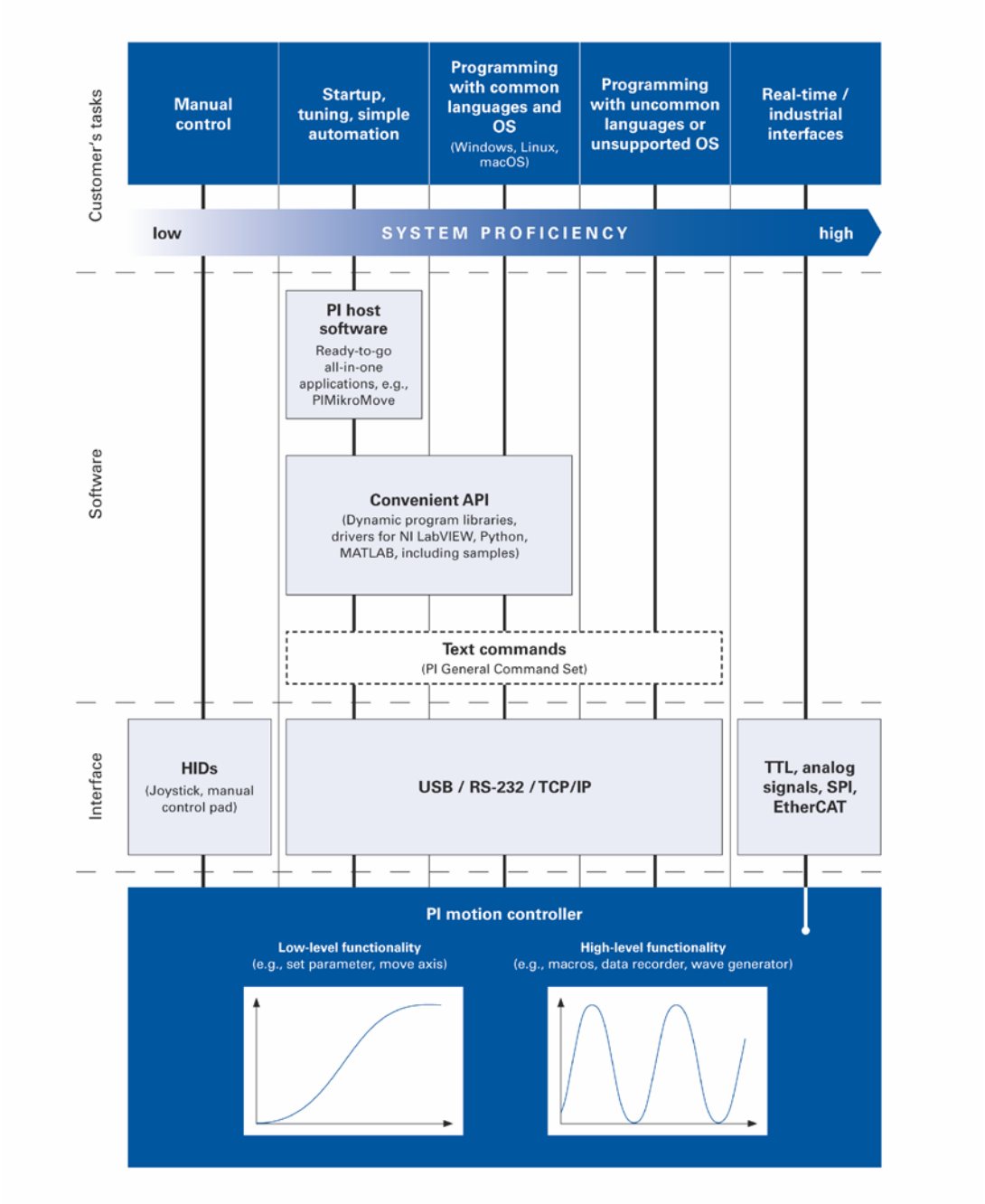
During calibration of conventional nanopositioning systems at the factory, the coefficients of the Axis-to-OutputSignalChannel matrix are set. You should not change the coefficient for the piezo amplifier channel.

If the connected mechanics has an ID-chip, the coefficients will be read in from the ID-chip (see "ID-Chip Support / Stage Replacement" (p. 120) for more information).

The VOL? command (p. 189) reports the current voltage output of the output signal channel (in volts).

3.11 Overview of PC Software

PI's systems can generally be controlled as follows:



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

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

Software Tool	Supported Operating System	Short Description	Recommended for
Dynamic program library for GCS	Windows, Linux	Allows program access to the E-709 from languages like C++. The functions in the library are based on the PI General Command Set (GCS). Windows operating systems: PI_GCS2_DLL; Linux operating systems: libpi_pi_gcs2.so.x.x.x and libpi_pi_gcs2-x.x.x.a where x.x.x gives the version of the library	Recommended for customers who want to use a library for their applications. Is required for PIMikroMove. Is required for the drivers for NI LabVIEW software.
Driver for use with NI LabVIEW software	Windows, Linux	NI LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments). The E-709 software is a collection of virtual instrument drivers (VI drivers) for the E-709. In addition to the product-specific drivers, the product CD also contains the Analog drivers, a collection of drivers for generating an analog control signal; see "Performing the Initial Installation" (p. 42). The drivers support the PI General Command Set.	For users who want to use NI LabVIEW to program their application.

Software Tool	Supported Operating System	Short Description	Recommended for
PIMikroMove	Windows	Graphic user interface for Windows with which the E-709 and other controllers from PI can be used: The system can be started without programming effort Graph of motions in open-loop and closed-loop operation Macro functionality for storing command sequences on the PC (host macros) Support of HID devices Complete environment for command entry, for trying out different commands No command knowledge is necessary to operate PIMikroMove. PIMikroMove uses the dynamic program library to supply commands to the controller. To provide the Device Parameter Configuration window, PIMikroMove requires the NI LabVIEW Run-Time Engine; see "Performing the Initial Installation" (p. 42).	For users who want to perform simple automation tasks or test their equipment before or instead of programming an application. A log window showing the commands sent makes it possible to learn how to use the commands.
PITerminal	Windows	Terminal program that can be used for nearly all PI controllers (see the description of the Command Entry window in the PIMikroMove user manual).	For users who want to send GCS commands directly to the controller.
PI Update Finder	Windows	Checks the PI software installed on the PC. If more current versions of the PC software are available on the PI server, downloading is offered.	For users who want to update the PC software.
PI Firmware Update Wizard	Windows	Program for user support when updating firmware of the E-727.	For users who want to update the firmware.
USB driver	Windows	Driver for the USB interface	For all users.

4 Installation

4.1 General Notes on Installation

➔ Install the E-709 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.

E-709 models for capacitive sensors:
➔ Electromagnetic signals in the range of the sensor frequency (100 kHz) can interfere with the sensor signal.
➔ Avoid electromagnetic signals in the range of 100 kHz.
➔ Keep in mind that low-frequency signals may have harmonics in the range of 100 kHz.
➔ If interfering signals in the range of 100 kHz cannot be avoided, take particular care to ensure suitable shielding and grounding. For more information, see “Troubleshooting” (p. 241).

4.2 Installing the PC Software

The communication between the E-709 and a PC is necessary to configure the E-709 and send motion commands using the commands of the GCS. Various PC software applications are available for this purpose.

4.2.1 Performing the Initial Installation

Accessories

- PC with a Windows operating system (7, 8, 10) or Linux operating system
- Product CD (included in the scope of delivery)

Important information on the procedure for installation on Windows

➔ Before you start installing the PC software on a PC with a Windows operating system (p. 44), read the following information.

When PIMikroMove **is installed (default installation):**

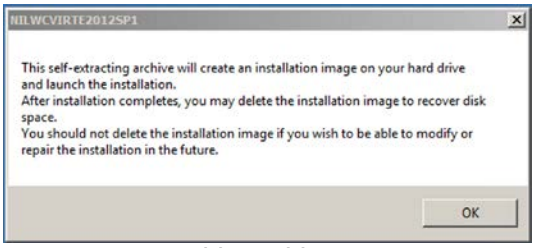
To provide the ***Device Parameter Configuration*** window, PIMikroMove requires the NI LabVIEW Run-Time Engine. The installation of PIMikroMove therefore includes the installation of the NI LabVIEW Run-Time Engine. A separate window opens for the installation of the NI LabVIEW Run-Time Engine in addition to the ***InstallShield Wizard*** window.

The InstallShield Wizard interrupts the installation of the PC software for the E-709 until the installation of the NI LabVIEW Run-Time Engine

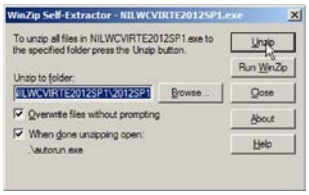
is started in the separate window.

➔ Note that the separate window can be covered by the InstallShield Wizard window on the screen. If necessary, display the separate window (e.g. by moving the InstallShield Wizard window).

➔ Follow the instructions for installing the NI LabVIEW Run-Time Engine that appear in the separate window (see figures below): Note that the files needed for installation have to be unpacked first. This does not complete the installation though; you have to continue according to the instructions in the separate window.



Agree to unpacking with **OK**



Start unpacking with **Unzip**

Finish unpacking with **OK**

Start the installation of the NI LabVIEW Run-Time Engine with **Install NI LabWindows/CVI Run-Time Engine 2012 SP1**

➔ Note that the installation of the NI LabVIEW Run-Time Engine can take some time.

➔ If you accidently close the separate window before the NI LabVIEW Run-Time Engine has been successfully installed: Go to the \SingleSetups directory on the product CD and start the installation by double-clicking the NI_LabWindows-CVI-RTE_2012_SP1_Setup.exe file.

Installing the PC software on Windows

1 Read "Important information on the procedure for installation on Windows" (p. 42).

2 Start the installation wizard by double-clicking the PI_E-709.CD_Setup.exe file in the installation directory (main directory of the CD).

The InstallShield Wizard window for the installation of programs and manuals for the E-709 opens.

3 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:

- Driver for use with NI LabVIEW software
  Exception: The Analog drivers component is only available through user-defined installation.
- Dynamic program library for GCS
- PIMikroMove
- PC software for updating the firmware of the E-709
- PI Update Finder for updating the PC software
- USB drivers

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

Installing the PC software on Linux

1 Unpack the tar archive from the /linux directory of the product CD to a directory on your PC.

2 Open a terminal and go to the directory to which you have unpacked the tar archive.

3 Log on as a superuser (root rights).

4 Enter ./INSTALL to start the installation.
Pay attention to lower and upper case when entering commands.

5 Follow the instructions on the screen.

You can select individual components for installation.

4.2.2 Installing Updates

➔ Always install the latest version of the PC software.

Prerequisite

- Active connection to the Internet.
- If your PC uses a Windows operating system:
  - You have installed the PI Update Finder from th product CD (p. 42).
  - You have the A000T0028 Technical Note for the PI Update Finder at hand. You can find the document on the product CD.
  - If the PC to be updated is not directly connected to the Internet:
    You have the A000T0032 Technical Note for the PI Update Finder at hand. You can find the document on the product CD.
- If your PC uses a Linux operating system:
  - You have the user name and password for the E-709 at hand. See the instructions on p. 7 for how to obtain the access data.

Updating the PC software on Windows

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

Updating the PC software on Linux

1 Open the website www.pi.ws.

2 Log in with the access data for E-709 (user name, password).

3 Click **Search**.

4 Enter the product code up to the period (“E-709”) 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:

6.1 If necessary: Scroll down the list.

6.2 If necessary: Click **Load more results** at the end of the list.

6.3 Click the corresponding product in the list.

7 Scroll down to the **Downloads** section on the product detail page.

The “CD Mirror” archive file is displayed under **Software Files**.

8 Copy the “CD Mirror” archive file to your PC.

9 Unpack the archive file to a separate installation directory.

10 In the directory with the unpacked files, go to the linux subdirectory.

11 Unpack the archive file in the linux directory by entering the command tar -xvpf <name of the archive file> on the console.

12 Read the accompanying information on the software update (readme file and/or "xxx_Releasenews.pdf" file) and decide whether the update makes sense for your application.

- If no: Stop the update procedure.
- If yes: Perform the following steps.

13 Log onto the PC as a superuser (root rights).

14 Install the update.

4.3 Installing the E-709

4.3.1 Ensuring Ventilation

➔ Place the system in a location with adequate ventilation to prevent internal heat build-up. Allow at least 10 cm (4 inches) clearance from the top and 5 cm (2 inches) from each side of the unit.

4.3.2 Installing E-709.xRG and .CHG Bench-Top Devices

E-709.xRG and .CHG bench-top units can be used as desktop device or mounted on a base in any orientation. If you want to mount the E-709.xRG or .CHG on a base, see "Dimensions" (p. 254) for the mounting hole locations.

Connecting E-709.xRG and .CHG Bench-Top Devices to the Protective Earth Conductor

To connect E-709.xRG and .CHG bench-top units to a protective earth conductor, use the intended screw attached to the mounting flange on the rear panel. See figure below.

Figure 11: Where to connect the protective earth conductor (here: E-709.xRG model; E-709.CHG models have the connection on the left side of the housing)

Prerequisite

- The E-709 is switched off, i. e. the power supply is not connected to the power socket via the power cord.

Tools and accessories

- Suitable protective earth conductor:
  - Cable cross-section ≥0.75 mm^2
  - Contact resistance < 0.1 ohm at 25 A at all connection points relevant for mounting the protective earth conductor
- Fastening material for the protective earth conductor, sits on the protective earth connector (threaded bolt) in the following order upon delivery of the E-709, starting from the housing:
  - Safety washer
  - Nut
  - Flat washer
  - Toothed washer
  - Nut
- Suitable wrench

Proceed as follows:

1 If necessary, fasten a suitable cable lug to the protective earth conductor.

2 Remove the outer nut from the protective earth connector on the rear panel of the E-709 (threaded bolt marked with ⏚).

3 Connect the protective earth conductor:

a) Push the cable lug of the protective earth conductor onto the threaded bolt.

b) Screw the nut onto the threaded bolt. In this way, the cable lug of the protective earth conductor is wedged between the toothed washer and the nut.

c) Tighten the nut with at least three rotations and a torque of 1.2 Nm to 1.5 Nm.

4.3.3 Installing E-709.xR OEM Modules

WARNING

Before you install E-709.xR OEM modules, diconnect the system from the supply voltage completely by removing the power plug from the wall socket. Otherwise voltages between -30 and 130 V can be exposed.

Since parts of the circuit will store charge, precautions must also be taken when an E-709 OEM module is not powered. After switching off the E-709 OEM module, wait a minute before you touch parts of the E-709 OEM module to be sure that any residual voltage has dissipated.

The E-709 OEM module heats up during operation. Do not touch the module during operation or immediately after operation.

The E-709 is an ESD-sensitive (electrostatic discharge sensitive) device. Observe all precautions against static charge buildup before handling the device.

E-709.xR OEM modules have to be installed in a suitable shielded enclosure to make sure that they conform to EMC standards and that live parts are suitably encased.

E-709.xR OEM modules can be mounted on a base in any orientation. This base must be made of material with high thermal conductivity. See "Dimensions" (p. 254) for the locations of the four M4 mounting holes in the E-709.xR cooling plate.

Connecting E-709.xR OEM Modules to the Protective Earth Conductor

To connect E-709.xR OEM modules to a protective earth conductor, use the protective earth connection of the module. See figure below.

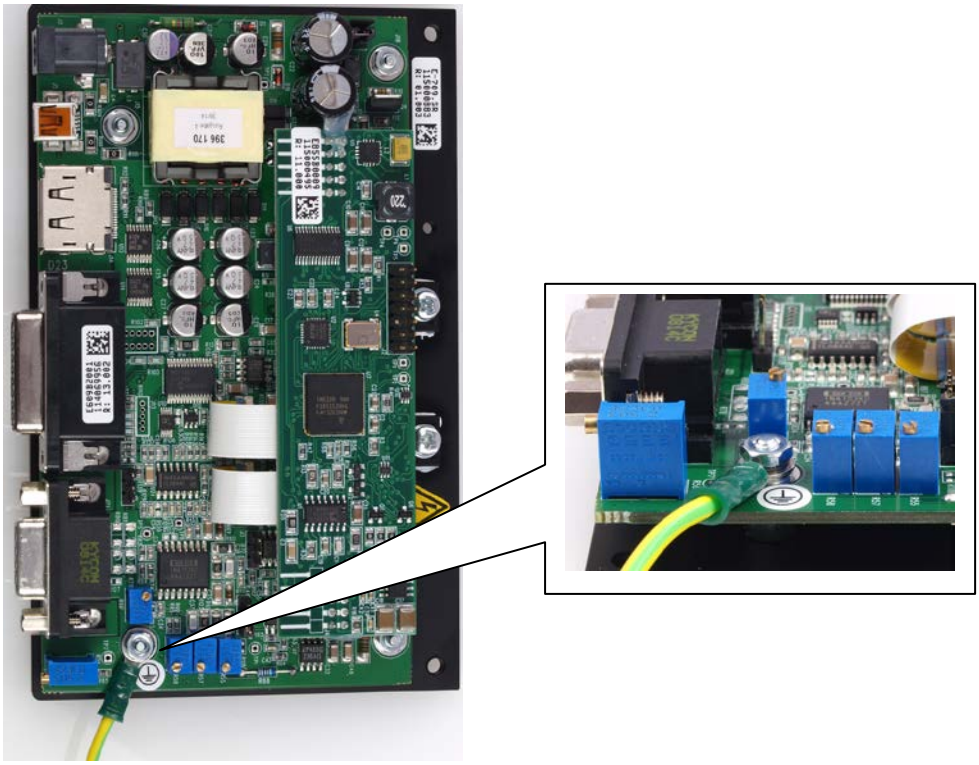


Figure 12: Protective earth connection of an E-709 OEM module (here: E-709.SR model)

Prerequisite

- The E-709 is switched off, i. e. the power supply is **not** connected to the power socket via the power cord.

Tools and accessories

- Suitable shielded enclosure: Metal housing that is connected to a protective earth conductor.
- Suitable conductor for connection between the E-709 OEM module and the metal housing:
  - Cable cross-section ≥0.5 mm^2
  - Cable lug suitable for M3 screws is present on the cable end which connects to the E-709

- Contact resistance < 0.1 ohm at 25 A at all connection points relevant for mounting the protective earth conductor

■ Fastening material for the conductor (not included):

- 2 flat washers suitable for M3 screws
- 1 spring washer suitable for M3 nuts

■ Suitable wrench

Proceed as follows:

1. Remove the nut from the protective earth connection on the E-709 OEM module (marked with ⏚).
2. Push the following items on the protective earth connection of the E-709 in the given order:
   a) Flat washer
   b) Cable lug of the conductor
   c) Flat washer
   d) Spring washer
3. Screw the nut which was removed in step 1 onto the protective earth connection of the E-709. In this way, the cable lug of the conductor is wedged between the washers.
4. Tighten the nut.
5. Connect the free end of the conductor to the metal housing in a suitable way.

4.4 How to Interconnect the System

Prerequisite

■ The E-709 is switched off, i. e. the power supply is **not** connected to the power socket via the power cord.

Proceed as follows:

1. If you want to use one or more of the lines listed below, connect the appropriate inputs/devices to the E-709 (depending on the E-709 model, the lines are present on the "I/O" socket and/or on separate "Analog In" and "Monitor Out" connections, see "Product View" (p. 14) for more information):

--> Analog input for control value generation or external sensor, see "How to work with the Analog Input" (p. 67) for more information.

--> Two digital output lines to trigger external devices and one digital input line to trigger items/events in the E-709, see "External Triggering / Signaling" (p. 85) for more information.

--> Analog output as axis position monitor or to control an external amplifier

--> Monitor line for piezo output voltage

--> On-target, overflow and servo state lines (LVTTL signals)

--> "ENA" enable line for switching the E-709 on or off (TTL, on = high (default), off = low; i.e. connecting this line to ground switches the E-709 off)

To facilitate connecting, PI offers appropriate accessories (p. 22).

2 Connect the E-709 to the host PC via a USB-A/USB-B cable or a null-modem RS-232 cable.

The RS-232 null-modem cable of E-709.xRG and E-709.xR models must be connected to the "I/O" socket (p. 260) of the E-709 via a suitable adapter (see p. 21 and p. 22).

See "Communication" (p. 63) for more information.

3 Connect the piezo stage to the "PZT & Sensor" socket (p. 257 or p. 258) of the E-709:

A label on the E-709 indicates the piezo stage with which the controller was calibrated. Be sure to respect this assignment when connecting the stage to the controller. When you are using a piezo stage with ID-chip together with the E-709, piezo stages can be easily exchanged because the calibration data is in the ID-chip; for details see "E-709.Cxx Models Only: ID-Chip Support / Stage Replacement" (p. 120).

4 Make sure that the E-709 is connected to a protective earth conductor. See "Installing the E-709" (p. 60) for more information.

5 Connect the "24 VDC" socket of the E-709 to a suitable power supply whose AC power cord is **not yet** connected to the wall socket.

E-709.xRG and E-709.CHG bench-top devices come with a 24 V wide-range-input power supply that can be used with line voltages from 100 VAC to 240 VAC at 50 or 60 Hz. With E-709 OEM modules, no power supply is included. You can order it separately, order number C-501.24050H (p. 22).

4.5 Synchronizing Multiple E-709.CHGs

If multiple E-709.CHGs are used, their sensor signals can be synchronized.

To synchronize *n+1* E-709.CHGs, you need *n* special synchronization cables which can be obtained from PI (order# K040B0151). Connect the "Sync Out" socket of the first device to the "Sync In" socket of the second device, the "Sync Out" socket of the second device to the "Sync In" socket of the third device and so on (see p. 258 for pinout and specifications).

Each connector of a synchronization cable matches to only one of the sockets on the E-709.CHG, so do not apply force when connecting them.

5 Start-Up

This chapter is intended to enable you to start initial test motions of a stage that is connected to an E-709 in the PIMikroMove PC software.
The start-up should comprise the following steps in the given order:

- Starting the system in PIMikroMove (p. 55): Installation, power-on, communication between E-727 and PC in PIMikroMove, configuration of PIMikroMove
- Creating backup file for controller parameters (p. 58)
- Executing test motions in open-loop operation (p. 59): First test of the function

5.1 General Notes on Start-Up and Operation

WARNING

If an E-709 OEM model is operated without a housing, live parts will be accessible. Touching the live parts can result in serious injury or death. Electrical, magnetic and electromagnetic fields emitted by live parts can disturb the E-709 OEM module and/or the environment.

- Only operate the E-709 OEM module when it is installed in a shielded housing that securely encloses all live parts and fulfills the requirements of electromagnetic compatibility.
- Since parts of the circuit will store charge, precautions must also be taken when the E-709 OEM module is not powered. After switching off the E-709 OEM module, wait a minute before you touch parts of the E-709 OEM module to be sure that any residual voltage has dissipated.

WARNING

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the E-709 in the case of malfunction or failure of the system. If touch voltages exist, touching the E-709 can result in serious injury or death from electric shock.

- Connect the E-709 to a protective earth conductor before start-up (p. 60).
- Do not remove the protective earth conductor during operation.
- If the protective earth conductor has to be removed temporarily (e. g. in the case of modifications or repair), reconnect the E-709 to the protective earth conductor before starting it up again.

NOTICE

Unsuitable settings of the notch filter and the servo-control parameters of the E-709 can cause the stage to oscillate. Oscillations can damage the stage and/or the load affixed to it.

→ If the stage is oscillating (unusual operating noise), immediately switch off the servo mode or disconnect the E-709 from the power source.

→ Only switch on the servo mode after you have modified the settings of the notch filter and the servo-control parameters of the E-709; see „Adjusting the Notch Filter(s) in Open-Loop Operation“ (p. 110) and "Checking and Optimizing the Servo-Control Parameters" (p. 115).

INFORMATION

→ Make sure that the system is interconnected completely before powering up the E-709.

→ Do not pull out any connector of the system during operation.

→ Secure the connections with the integrated screws against accidental disconnection.

INFORMATION

The "ENA" line (pin 4) of the "I/O" socket (p. 260 or p. 261) can be used to switch the E-709 on or off. Connecting this line to ground switches the E-709 off (NPN input, on = open (default), off = GND/low).

INFORMATION

The E-709 performance can be reduced directly after power on due to thermal instability.

→ Switch the E-709 on at least one hour before starting work.

→ If the E-709 is not used, but should remain switched on to ensure the temperature stability: Make sure that the servo mode is switched off (open-loop operation) and the piezo output voltage is set to 0 V. To set the piezo output voltage to 0 V, use the SVA command.

5.2 Starting the System in PIMikroMove

Proceed as follows to start the E-709 with the stage in PIMikroMove:

1 Read “General Notes on Start-Up and Operation” (p. 53) and the documentation of the stage.

2 Install the following on the PC:

- The PC software and the USB drivers from the product CD
- Updates for PC software

Details see "Installing the PC Software" (p. 42).

3 Install the E-709, for details, see p. 46.

4 Interconnect the system, for details, see p. 50.

5 Switch on the E-709: Connect the power cord of the wide-range-input power supply to the wall socket.

The E-709 is powered up immediately when the AC power cord of the external power supply is connected to the power socket. Green light of the "Status" LED indicates that the device is powered and ready for operation.

On power-on or reboot (with the RBT command (p. 165)), the E-709 performs firmware verification and copies information from non-volatile memory to volatile memory.

6 Start PIMikroMove on the PC.

7 Establish communication between the E-709 and the PC in PIMikroMove via RS-232 or USB:

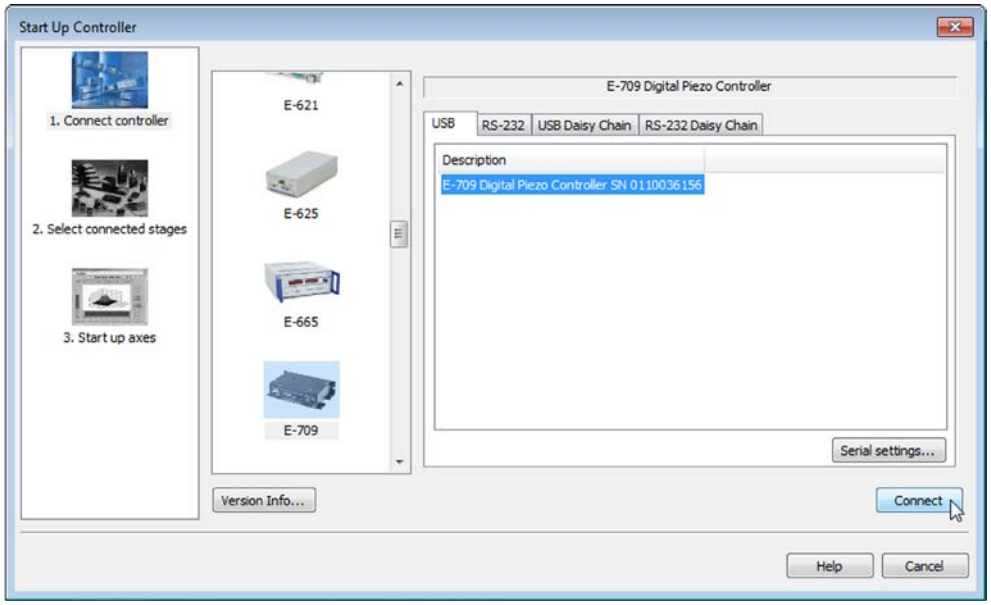
7.1 Select E-709 in the field for controller selection.

7.2 Select the tab card that matches the interface used (USB in the example shown below).

7.3 Depending on the interface used, select your controller on the tab card (USB) or select the interface settings (RS-232).

7.4 Click *Connect*.

For further details, see "Communication" (p. 63).



8 If necessary, execute the AutoZero procedure in the *Start up axes step* in PIMikroMove.

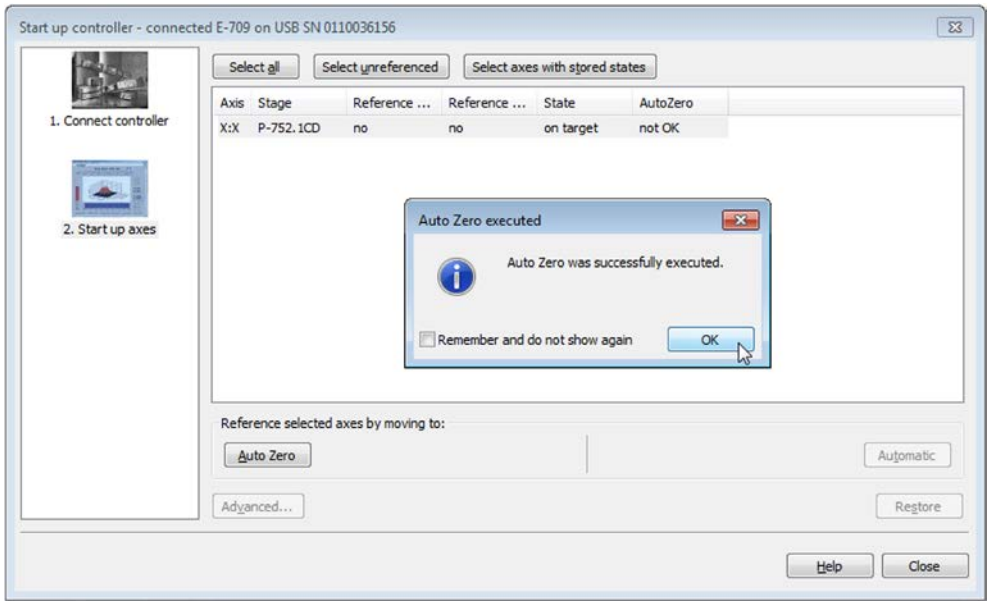
8.1 Mark the axis in the list.

8.2 Click *Auto Zero*. The *Auto Zero* dialog opens.

8.3 In the *Auto Zero* dialog, start the AutoZero procedure by clicking *Start*.

8.4 Click *OK* in the *Auto Zero executed* dialog.

For further details, see "AutoZero Procedure" (p. 61).

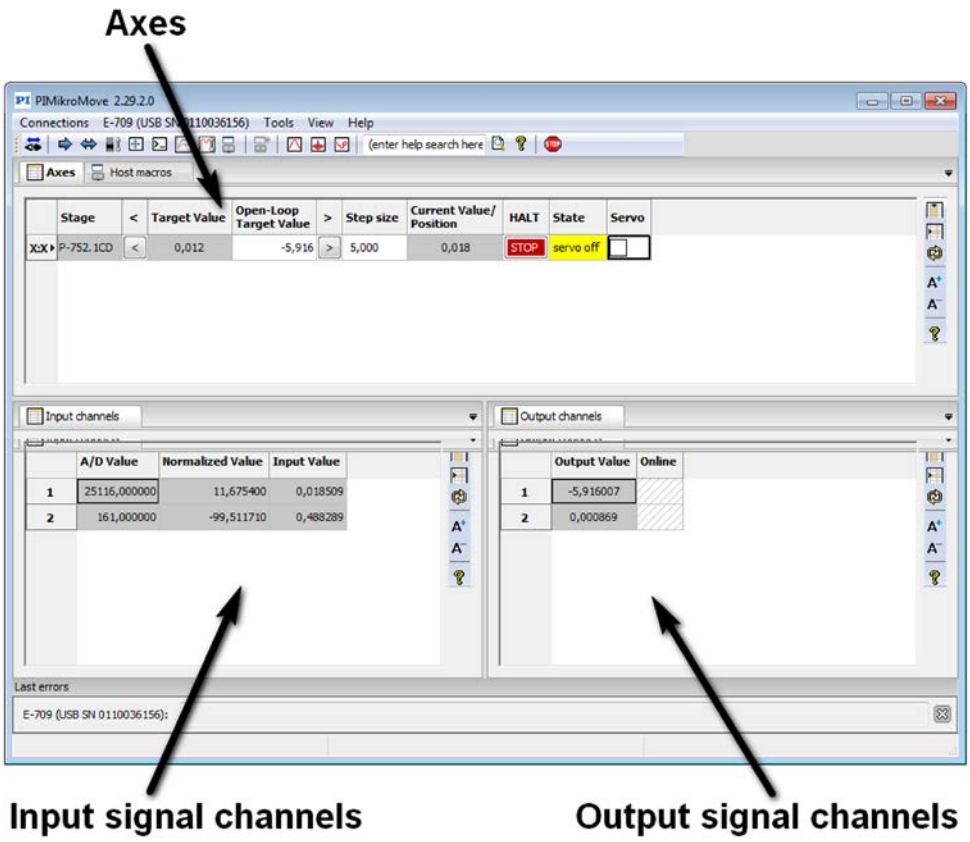


9 In the *Start up controller* window, click *Close*.
The main window of PIMikroMove opens.

10 Optionally: Configure the PIMikroMove main window.

It is recommended to see the tab cards for axes, input signal channels and output signal channels (see figure below). You can arrange them by dragging them with the left mouse button pressed so that they become docked e.g. to the bottom border of the window.

On the *Axes* tab card, amongst others you can start axis motion. The channel tab cards show the current values of the input signal channels (sensors) and output signal channels (output voltages for piezo actuators).



Note that the input and output signal channels of the E-709 are allocated to the logical axis via matrices, for details, see “Input Signal Processing” (p. 33) and “Output Generation” (p. 37).

5.3 Creating Backup File for Controller Parameters

The properties of the E-709 and the connected stage are stored in the E-709 as parameter values.

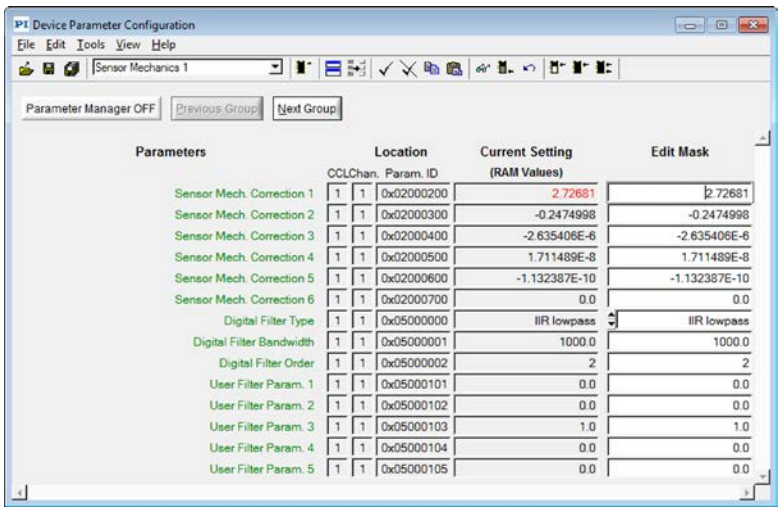
- ➔ Create a backup copy on the PC before changing the parameter values of the E-709. You can then restore the original settings at any time.
- ➔ Create an additional backup copy with a new file name each time after you optimize the parameter values.

To save the parameter values and to load them back to the E-709, use the *Device Parameter Configuration* window provided by PIMikroMove.

Proceed as follows to create a parameter file:

1 In the main window of PIMikroMove, open the *Device Parameter Configuration* window via the *E-709…* ⇒ *Parameter Configuration* menu item.

In the figure below, the *Device Parameter Configuration* window shows the *Sensor Mechanics 1* parameter group.



2 Save the parameter values from the *Edit Mask* column of the *Device Parameter Configuration* window in a parameter file (file extension .pam) on your PC. Use one of the following options:

- *File* > *Save Edit Values* or *File* > *Save Edit Values As* menu item
- (Save) or (Save As) button in the icon bar

5.4 Executing Test Motions in Open-Loop Operation

The first moves should be made in open-loop operation. With the factory default settings of the E-709, open-loop commanding means to specify the piezo output voltage.

1 In the main window of PIMikroMove, make some test moves with the axis using the controls on the *Axes* tab card. During the test moves, observe the position display for the axis (in the *Current Value / Position* field) and the current output voltage for the piezo actuator in the stage (in the *Output Value* fields of the *Output Channels* tab card).

Proceed as follows for the axis:

1.1 Make sure that the *Servo* box is unchecked.

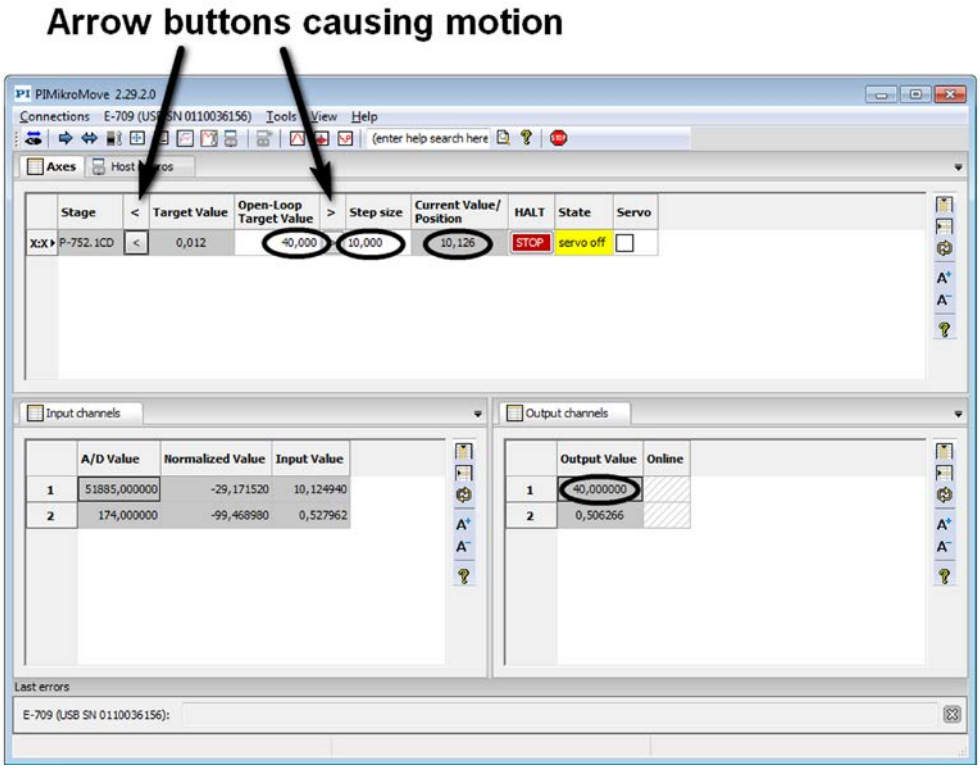
1.2 Command a piezo output voltage of 0 V by entering 0 in the *Open-Loop Target Value* field of the axis and pressing Enter on your keyboard.

1.3 Enter the value 10 (V) in the *Step size* field of the axis and press Enter.

1.4 Use the > button next to the *Open-Loop Target Value* field to increment the commanded value by the value given in the *Step size* field (10). Increment the piezo output voltage this way step by step until the axis is at its upper travel range limit.

1.5 Use the < button next to the *Open-Loop Target Value* field to decrement the commanded value by the value given in the *Step size* field (10). Decrement the piezo output voltage this way step by step back to zero.

The values for position and output voltage should follow the commanded open-loop values.



In the example shown in the figure above, the piezo output voltage for axis 1 was increased to 40 V by clicking the > button four times (step size value is 10). Since the piezo actuator of the axis is driven by output signal channel 1, the *Output Value* field of channel 1 shows 40 V. The current position has changed accordingly (10.126 µm).

2 Make open-loop frequency response measurements in the *Piezo Dynamic Tuner* window of PIMikroMove to determine the resonant frequencies of the axis. If there are resonances which are intolerable in your application, adjust the notch filter settings for the axis before you switch to closed-loop operation for the first time (servo on). Furthermore, it might be necessary to readjust the preset servo parameters for the axis. See "Servo Controller Dynamic Tuning" (p. 108) for more information.

6 AutoZero Procedure

The AutoZero procedure performs automatic zero point adjustment of the sensors.

INFORMATION

During the AutoZero procedure, the axis will move, and the motion can cover the whole travel range.

AutoZero is to be performed with linear axes only. Starting AutoZero for rotation axes will fail and cause the error code 74 („No sensor channel or no piezo channel connected to selected axis (sensor and piezo matrix)“).

AutoZero changes the mechanical zero position of the piezo stage.

The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished.

AutoZero works in open-loop operation only. If servo-control is on, it will be switched off automatically at the start of the AutoZero procedure and switched on again when it is finished.

Objective of AutoZero:

- Make the entire travel range available:
  Changes in temperature or changes in the mechanical load can cause small deviations of the sensor zero point. When the sensor zero-point is set correctly, the complete output voltage range of the amplifier can be used in closed-loop operation.

- Prevent the piezo actuators from damage:
  In open-loop operation, the stage displacement with 0 V piezo voltage should already be about 10 % of the travel range. Then the average applied voltage is reduced which lengthens the lifetime of the piezo actuator in the stage without reducing the nominal travel range.

Prerequisites for AutoZero:

- LowVoltage < HighVoltage
  LowVoltage is given by the value of the AutoZero Low Voltage parameter (ID 0x07000a00);

HighVoltage is given by the value of the AutoZero High Voltage parameter (ID 0x07000a01)

- The value of the AutoZero High Voltage parameter (ID 0x07000a01) should be identical with the piezo voltage that is required for maximum displacement of the axis.

Settings Changed by AutoZero:

The AutoZero procedure changes the value of the parameter Sensor Mech. Correction 1 (ID 0x02000200).

Starting AutoZero via Command Entry:

Via command entry, you have the following options to start AutoZero:

- Use the ATZ command to perform the AutoZero procedure once (see p. 135 for details). Afterwards save the values of the parameters Sensor Mech. Correction 1 (ID 0x02000200) and Sensor Offset factor (ID 0x02000102) to non-volatile memory.
- Send the ATZ command after every start or reboot of the E-709.
- Set the value of the Power Up AutoZero Enable parameter (ID 0x07000802) to 1 for the axis so that the AutoZero procedure is performed automatically with every start or reboot of the E-709.

Starting AutoZero in PIMikroMove:

See “Starting the System in PIMikroMove” (p. 55) for how to perform the AutoZero procedure in PIMikroMove.

INFORMATION

If present, do not use the "Zero" trim pot for sensor zero-point adjustment. It is adjusted before delivery.

7 Communication

7.1 Interfaces Available

The E-709 can be controlled from a host computer (not included) with ASCII commands sent via:

- USB
- RS-232 serial connection

In addition, the E-709 can be controlled by an SPI master, see the E709T0002 Technical Note for details.

The PC interfaces and the SPI interface are active simultaneously. The commands from the interfaces are queued in the order the completed command lines are received.

7.2 USB Connection

The USB interface is available on the front panel of the E-709 via the type B USB socket. Use the included USB cable (USB-A/USB-B) to connect the E-709 to the host PC.

The USB drivers are installed automatically with the PC software (p. 42).

In the PC software (e.g. PIMikroMove, PITerminal or drivers for use with NI LabView software) all E-709 which are connected to the USB sockets of the PC are listed. In PIMikroMove you have, for example, to click on the controller type (1). Then select the *USB* tab card (2). On the interface tab card, click on the E-709 to which you want to connect (3). To establish the connection, click *Connect* (4).

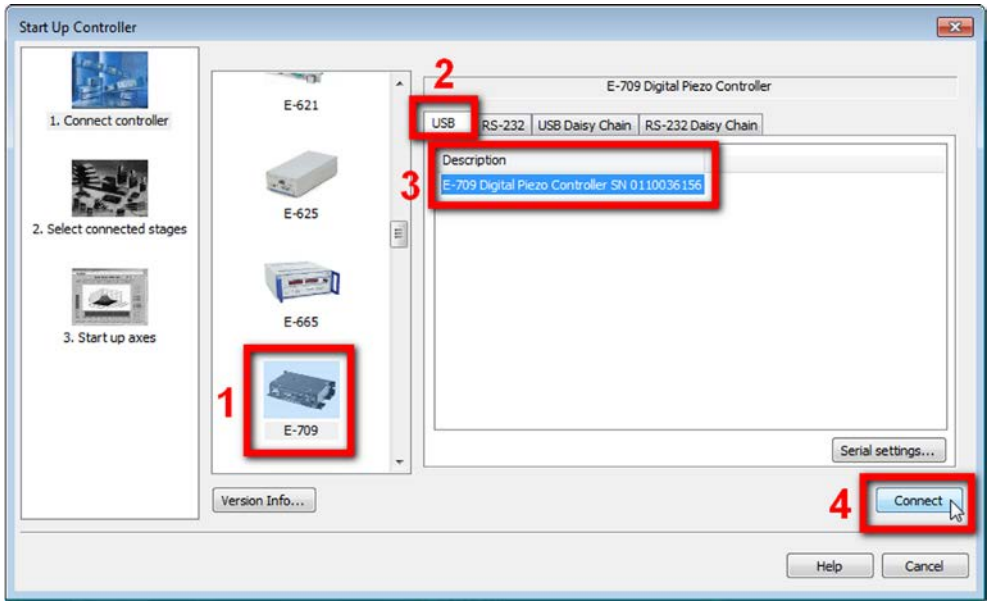


Figure 13: The USB connection dialog in PIMikroMove

INFORMATION

With USB connections, communication cannot be maintained after the E-709 is power-cycled or rebooted. The USB connection must then be closed and reopened.

7.3 RS-232 Serial Connection

The RS-232 serial communications port is accessed as follows:

- E-709.xRG and .xR models: via the "I/O" socket (p. 260) on the E-709 front panel and a suitable adapter. This is either the included E709B0002 RS-232 adapter (see figure below) or the E-709.01 adapter which must be ordered separately (see "Additional Components" (p. 22) for details).

Figure 14: E709B0002 RS-232 adapter, included with E-709.xRG and .xR models

- E-709.CHG models: via the "RS-232" socket (p. 259) on the E-709 front panel.

Connect the E-709 to a COM port of the host PC via the included null-modem cable.

The serial port on the E-709 is preset as follows:
57,600 baud, 8 bits, no parity, RTS/CTS

INFORMATION

The following commands are available for the baud rate of the E-709:

Value in the nonvolatile memory:

Get with IFS?

Set with IFS

Value in the volatile memory:

Get with IFC?

Set with IFC

For querying and setting the baud rate, it is recommended to use the *Configure Interface* window in PIMikroMove. For details, see the PIMikroMove manual.

In the connection dialog of the PC software (e.g. PIMikroMove, PITerminal or drivers for use with NI LabView software), you make the settings on the PC side. In PIMikroMove you have, for example, to click on the controller

type (1). Then select the *RS-232* tab card (2). On the interface tab card, select the correct COM port and baud rate of the PC (3). To establish the connection, click *Connect* (4).

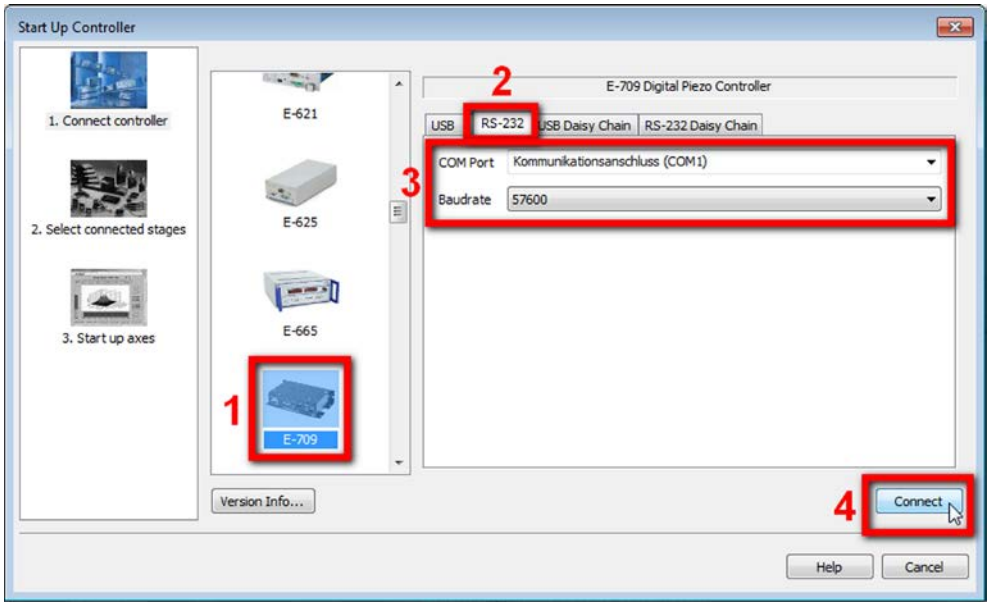


Figure 15: RS-232 configuration of the host PC side in PIMikroMove

8 Using the Analog Input

8.1 How to Work with the Analog Input - Overview

The E-709 provides an analog input line on pin 19 of the "I/O" socket (p. 260 or p. 261). With E-709.CHG models, the analog input line is also available on a separate "Analog In" SMB connection (p. 19).

For highest resolution, it is recommended to use the full input range of 0 to 10 V.

You can use the analog input line as follows:

- Connect a source for control value generation
- Connect an external sensor

Irrespective of the intended usage, the analog input values must first be scaled to suitable position values (see "Scaling the Analog Input" (p. 68)). Furthermore, it is necessary to change certain controller parameters to determine the usage of the analog input. See "Use as Control Value Generation Source" (p. 71) or "Use as External Sensor Input" (p. 72) for details.

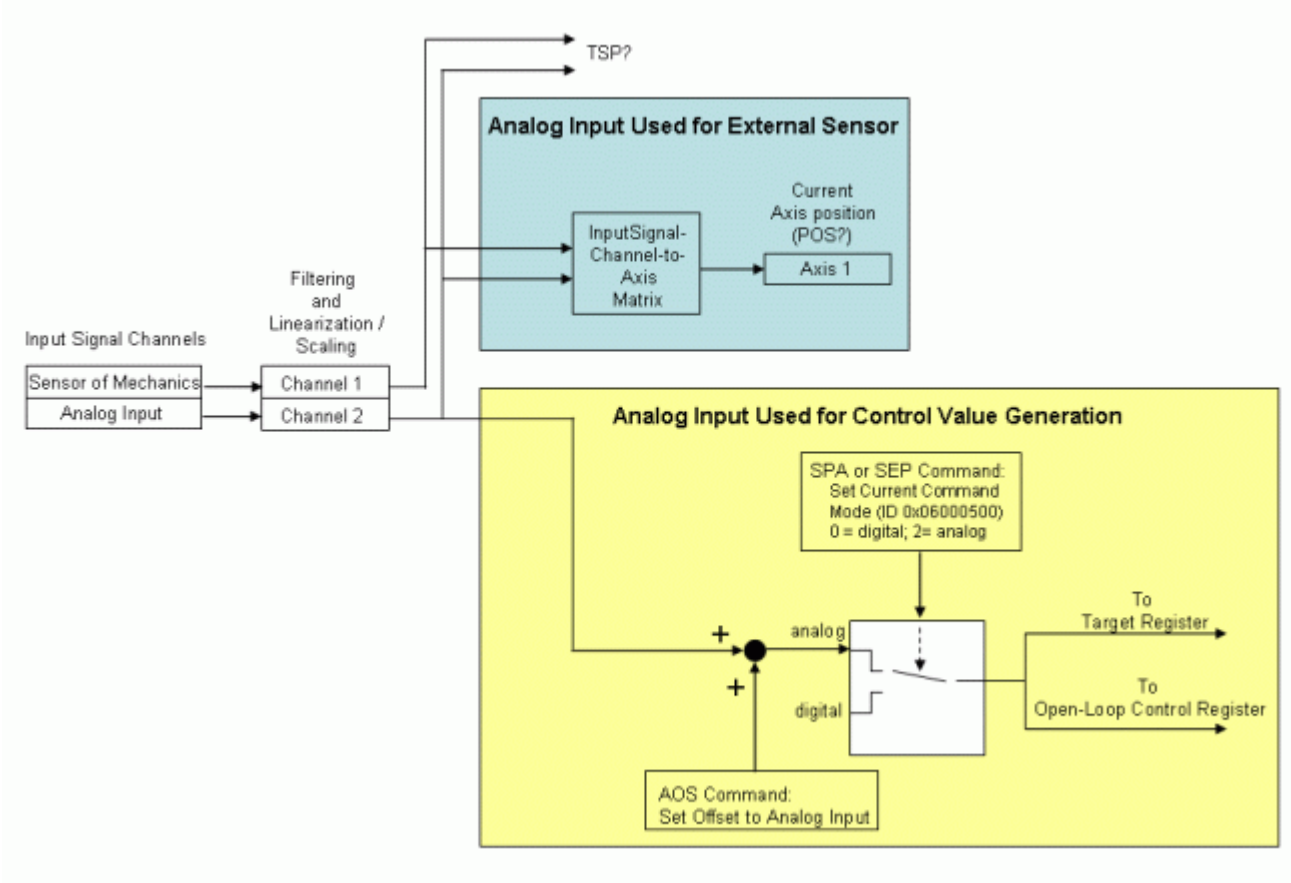
INFORMATION

It is strongly recommended to save the parameter values of the E-709 to a file on the host PC before you make any changes. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-709, use the *Device Parameter Configuration* window provided by PIMikroMove. See "Creating Backup File for Controller Parameters" (p. 58) for more information.

Wherever changing parameter values is mentioned, you can do this using SPA (p. 170) (volatile memory) or SEP (p. 168) (non-volatile memory). Furthermore, you can use WPA (p. 202) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. To have write access to certain parameter(s), it might be necessary to switch to a higher command level using CCL (p. 138). To read parameter values, query with the SPA? (p. 173) or SEP? (p. 169) commands.

PIMikroMove gives access to parameter values in a more convenient way. The program provides the *Device Parameter Configuration* window where you can check/edit the individual parameters. See the PIMikroMove manual for more information.

The analog input line is represented in the controller firmware as input signal channel 2 (see "Axes, Channels, Functional Elements" (p. 23)) for more information).



8.2 Scaling the Analog Input

Before the analog input line can be used with an external sensor or with a control-signal source, the input levels must be associated with suitable position values. To do this, adjust the OFFSET (parameter ID 0x02000200) and the GAIN (parameter ID 0x02000300) of the Mechanics linearization polynomial according to the travel range of the axis and the input signal range. See below for details. The TSP? command (p. 185) reports the analog input values after the scaling as position values in µm.
In addition, the digital filter parameters can be adjusted. See "Digital processing" for details.

How to adjust OFFSET and GAIN to map the analog input voltage to a suitably scaled position value:

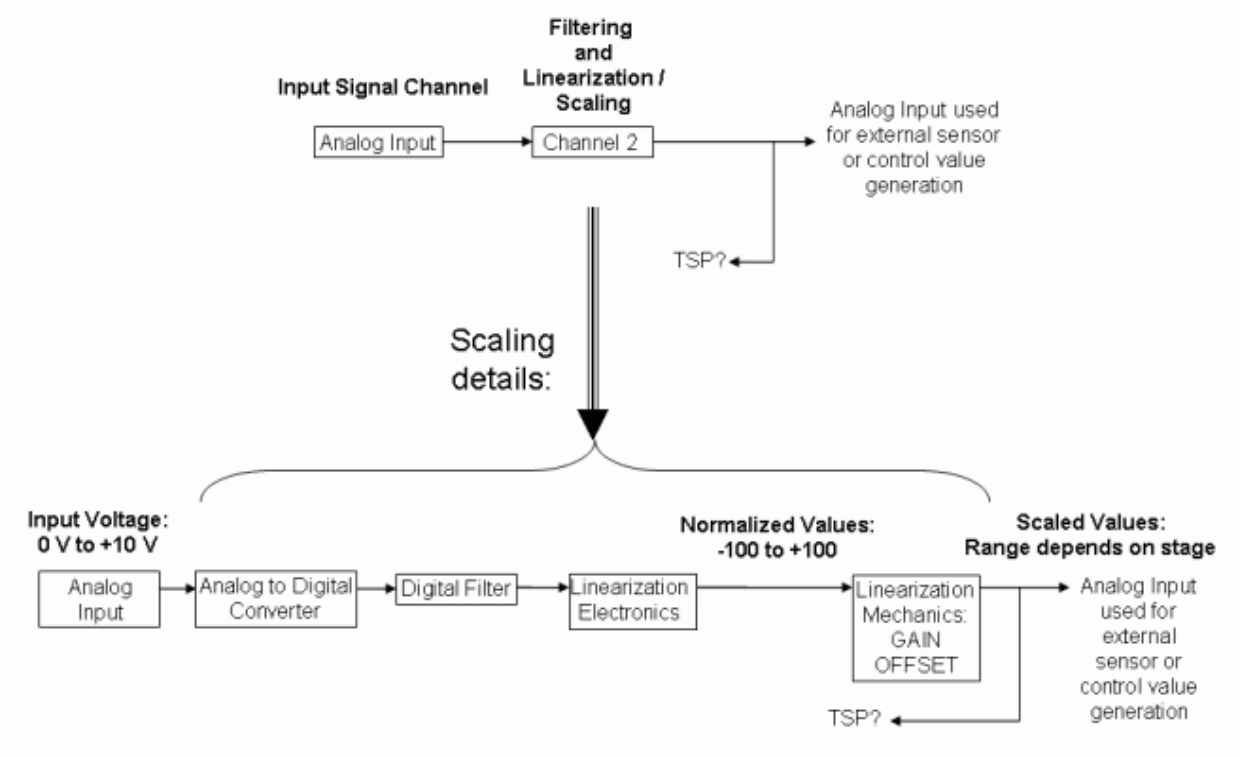


Figure 16: Processing of the analog input signal, detail from the overview figure above

Input Voltage:
the maximum range is 0 to +10 V

Normalized Value:
The polynomial used for electronics linearization (see "Digital processing" for details) converts the analog input voltage to a number in the range of -100 to +100. An input voltage value of 0 V always corresponds to -100, and +10 V corresponds to +100 respectively.

Scaled Value:
The range depends on the stage and can be set by the coefficients of the polynomial used for Mechanics linearization (see "Digital processing" for details):

ScaledValue = OFFSET + GAIN * NormalizedValue

where
OFFSET corresponds to the Sensor Mech. Correction 1 parameter, 'ID 0x02000200
GAIN corresponds to the Sensor Mech. Correction 2 parameter, ID 0x02000300
If no linearization is necessary, the other coefficients of the Mechanics linearization polynomial can be set to zero (parameter IDs 0x02000400, 0x02000500, 0x02000600, 0x02000700).

Note that in PIMikroMove, these parameters for the input signal channel 2 (= the analog input line) are available in the *Sensor Mechanics 2* parameter group in the *Device Parameter Configuration* window.

How to calculate the values to set for OFFSET and GAIN:
GAIN = (MaxScaledValue - MinScaledValue) / (MaxNormalizedValue - MinNormalizedValue)
OFFSET = MaxScaledValue - GAIN * MaxNormalizedValue

The values of "MinScaledValue" and "MaxScaledValue" depend on the travel range of the stage:
"MinScaledValue" is given by the TMN? (p. 182) answer (is defined by the Range Limit min parameter, ID = 0x07000000), and "MaxScaledValue" is given by the TMX? (p. 182) answer (is defined by the Range Limit max parameter, ID = 0x07000001).

The values of "MinNormalizedValue" and "MaxNormalizedValue" depend on the range of the external signal applied to the analog input line. See the examples below. For all examples, it is assumed that the stage has the following travel range:
MinScaledValue = -20 µm
MaxScaledValue = +120 µm

Example 1:
The full range of 0 V to +10 V is to be used (this is recommended for highest resolution).
MinNormalizedValue = -100
MaxNormalizedValue = +100
GAIN = (120 - (-20)) / (100 - (-100)) = 0.7
OFFSET = 120 - 0.7 * 100 = 50
ScaledValue = 50 + 0.7 * NormalizedValue
So you have to send

```
SPA 2 0x02000200 50
SPA 2 0x02000300 0.7
```

to adjust the GAIN and OFFSET parameters for input signal channel 2 (= the analog input line) in the E-709.

Example 2:
Only half the input voltage range is to be used, i.e. the range is 5 V to +10 V.
MinNormalizedValue = 0
MaxNormalizedValue = +100
GAIN = (120 - (-20)) / (100 - 0) = 1.4
OFFSET = 120 - 1.4 * 100 = -20
ScaledValue = -20 + 1.4 * NormalizedValue
Send:

```
SPA 2 0x02000200 -20
SPA 2 0x02000300 1.4
```

8.3 Use as Control Value Generation Source

To enable the analog control input for an axis, analog command mode must be selected for that axis. This is done with the Current Command Mode parameter (ID 0x06000500): 0 = digital command mode; 2= analog command mode. If the appropriate setting is saved as the power-on default, the axis can be commanded via analog input immediately after controller start-up, and no host PC is required.
To enable analog command mode in volatile memory, send:
SPA 1 0x06000500 2
Note that in PIMikroMove, this parameter is available in the *Target Manipulation* parameter group in the *Device Parameter Configuration* window.

When analog command mode is enabled for the axis, then the analog input overwrites the values of all other control sources for the axis except those from the AutoZero procedure. The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished. See "Axis Motion" (p. 26) for more information.

An offset value can be added to the analog input scaled value using the AOS command (p. 132). This offset is not used in digital command mode.

Stopping axis motion with STP (p. 176) or #24 (p. 131) enables the digital command mode. To recommence commanding the axis via the analog input, you have to re-enable analog command mode for the axis.

INFORMATION

In open-loop operation, 0 to 10 V applied to the analog input line correspond to -30 to 130 V piezo output voltage. For closed-loop operation, the analog input values must be scaled to suitable position values. See "Scaling the Analog Input" (p. 68) for more information. With the default scaling settings, the 0 to 10 V input range corresponds to the nominal travel range (minimum position to maximum position) of the axis.
Make sure that the slew rate (Servo Loop Slew-Rate, 0x07000200, or Open Loop Slew-Rate, 0x07000201) is set to a suitable value. If the slew rate value is too low the axis will not be able to follow the analog control input.
Make sure that in the InputSignalChannel-to-Axis matrix, the coefficient of the analog input is set to zero. With the E-709, this is the value of the Position From Sensor 2 parameter (ID 0x07000501). In IMikroMove, this parameter is available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

8.4 Use as External Sensor Input

To let the sensor connected to the analog input line participate in the position signal of axis 1, set its coefficient in the InputSignalChannel-to-Axis matrix to 1. This coefficient is represented by the Position From Sensor 2 parameter (ID 0x07000501) for axis 1. Send:
SPA 1 0x07000501 1
to change the coefficient in volatile memory. In PIMikroMove, this parameter is available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

One possible application could be that only the external sensor on the analog input line is to be used for position control of axis 1. In this case, the signal of the sensor in the mechanics must be excluded from the position monitoring of axis 1. To do this, set the Position From Sensor 1 coefficient in the InputChannel-to-Axis matrix to zero by sending:
SPA 1 0x07000500 0
The position of axis 1 (i.e. the POS? response) will then be based on the external sensor only , but it is still possible to read the signals of both the internal and the external sensor using the TSP? command.

INFORMATION

The analog input values must be scaled to suitable position values. See "Scaling the Analog Input" (p. 68) for more information.

Make sure that the analog input line is not used for control value generation. This means that the value of the Current Command Mode parameter (ID 0x06000500) must be 0 (digital command mode enabled). In PIMikroMove, this parameter is available in the *Target Manipulation* parameter group in the *Device Parameter Configuration* window.

8.5 Analog-Input-Related Commands and Parameters

Command	Description	Notes
AOS (p. 132)	Set Analog Input Offset	Adds an offset value to the analog input scaled value (Analog Target Offset, ID 0x06000501). This offset is active as long as the analog command mode is enabled for this axis.
AOS? (p. 134)	Get Analog Input Offset	Reads the current value of Analog Target Offset, parameter ID 0x06000501, from volatile memory
SEP (p. 168)	Set Nonvolatile Memory Parameters	Can be used to set the power-on default configuration for analog input usage.
SEP? (p. 169)	Get Nonvolatile Memory Parameters	Reads the current parameter values from non-volatile memory

Command	Description	Notes
SPA (p. 170)	Set Temporary Memory Parameters	Can be used to set a temporary configuration for analog input usage.
SPA? (p. 173)	Get Temporary Memory Parameters	Reads the current parameter values from volatile memory (RAM)
TAD? (p. 180)	Get ADC Value Of Input Signal	Reports the current ADC value of the analog input, dimensionless
TNS? (p. 183)	Get Normalized Input Signal Value	Reports the resulting value for the analog input after the electronics linearization, dimensionless
TSP? (p. 185)	Get Input Signal Position Value	Reports the resulting value for the analog input after the mechanics linearization (scaling), the unit is µm
WPA (p. 202)	Save Parameters To Nonvolatile Memory	Can be used to save the currently active configuration (including analog input usage) to non-volatile memory, where it becomes the power-on default.

See "How to work with the Analog Input" (p. 67) for more information. For detailed command descriptions see "Command Reference" (p. 129). For the identifiers of the items which can be addressed with the commands see "Axes, Channels, Functional Elements" (p. 23).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x02000200	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 1 (Offset)
0x02000300	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 2 (Gain)
0x05000000	1	Input Signal Channel	2	INT	Digital Filter Type
0x05000001	1	Input Signal Channel	2	FLOAT	Digital Filter Bandwidth
0x05000002	1	Input Signal Channel	2	INT	Digital Filter Order
0x06000500	1	Logical Axis	1	INT	Current Command Mode; 0 = digital 2 = analog
0x06000501	1	Logical Axis	1	FLOAT	Analog Target Offset
0x07000501	1	Logical Axis	1	FLOAT	Position from Sensor 2

See "Controller Parameters" p. 223) for more information regarding the controller parameters and their handling.

9 Using the Analog Output

9.1 How to Work with the Analog Output - Overview

The E-709 provides an analog output line on pin 20 of the "I/O" socket (p. 260 or p. 261). With E-709.CHG models, the analog output line is also available on a separate "Monitor Out" SMB connection (p. 19).

The analog output can be addressed in the firmware of the E-709 as output signal channel 2 and is intended for the following types of use:

- Control an external amplifier
- Monitor the axis position

The usage of the analog output is set via the parameters Select Output Type (ID 0x0a000003) and Select Output Index (ID 0x0a000004). See "Use Analog Output to Control External Amplifier" (p. 75) or "Use Analog Output to Monitor Axis Position" (p. 76) for details. The VOL? command reports the output voltage on the analog output. If necessary, the digital/analog converter of the analog output can be adjusted (p. 78).

INFORMATION

It is strongly recommended to save the parameter values of the E-709 to a file on the host PC before you make any changes. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-709, use the *Device Parameter Configuration* window provided by PIMikroMove. See "Creating Backup File for Controller Parameters" (p. 58) for more information.

Wherever changing parameter values is mentioned, you can do this using SPA (p. 170) (volatile memory) or SEP (p. 168) (non-volatile memory). Furthermore, you can use WPA (p. 202) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. To have write access to certain parameter(s), it might be necessary to switch to a higher command level using CCL (p. 138). To read parameter values, query with the SPA? (p. 173) or SEP? (p. 169) commands.

PIMikroMove gives access to parameter values in a more convenient way. The program provides the *Device Parameter Configuration* window where you can check/edit the individual parameters. See the PIMikroMove manual for more information.

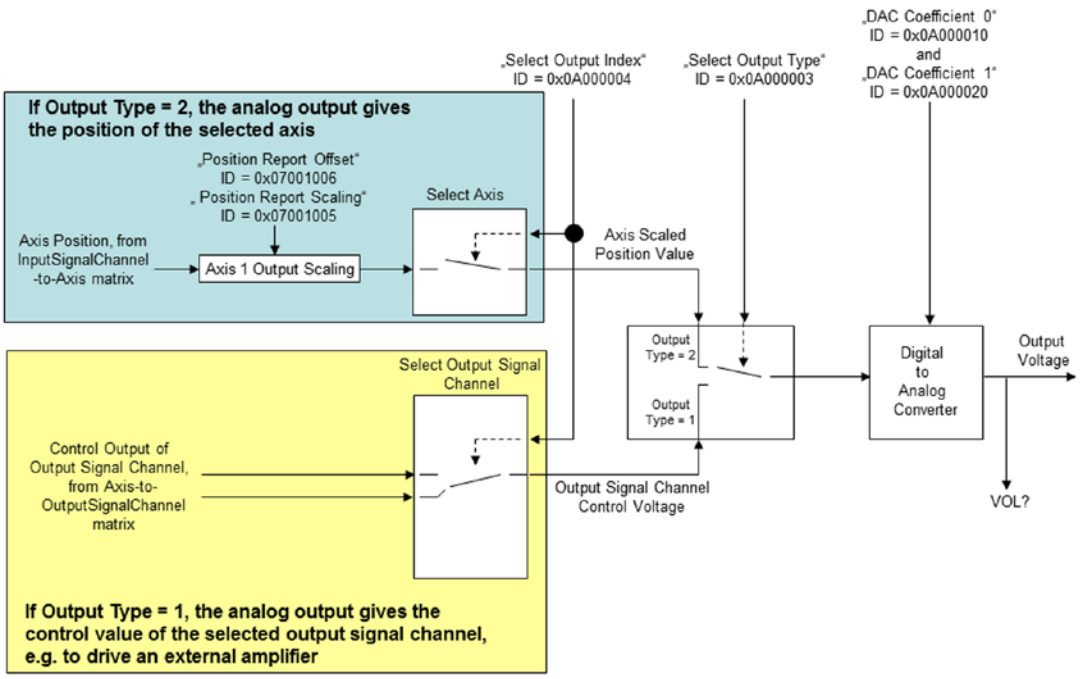


Figure 17: Overview over the usage of the analog output line of the E-709

9.2 Use to Control External Amplifier

Proceed as follows if you want to control an external amplifier via the analog output line:

1. Select output type 1 = "control voltage of output signal channel" for the analog output line using the Select Output Type parameter, ID 0x0A000003.

   Example: With an E-709.CR, the analog output line on pin 20 of the I/O socket (output signal channel 2) is to be used to control an external amplifier. To select the corresponding output type in volatile memory, send:
   SPA 2 0x0A000003 1

   Note that in PIMikroMove, this parameter is available in the *DAC 2* parameter group in the *Device Parameter Configuration* window.

2. Connect the output signal channel whose control value is to be output to the analog output line using the Select Output Index parameter, ID 0x0A000004.

Note:
The control value of an output signal channel results from the Axis-to-OutputSignalChannel matrix transformation, see "Output Generation" (p. 37) for more information.

In the example, the control value of output signal channel 1 (which is the internal piezo amplifier of the E-709.CR) is to be connected to the analog output line (output signal channel 2). Send
SPA 2 0x0A000004 1

In PIMikroMove, this parameter is also available in the *DAC 2* parameter group in the *Device Parameter Configuration* window. Note that for the piezo amplifier channel, the output index is fixed to output signal channel 1.

3 Check the Axis-to-OutputSignalChannel matrix (Driving Factor of Piezo parameters, IDs 0x09000000 and 0x09000001; in PIMikroMove available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window) for feasible settings.

9.3 Use to Monitor Axis Position

Proceed as follows if you want to output axis position values on the analog output line:

1 Select output type 2 = "current position of axis" for the analog output line using the Select Output Type parameter, ID 0x0A000003.

Example: With an E-709.CR, the analog output line on pin 20 of the I/O socket (output signal channel 2) is to be used to monitor the axis position. To select the corresponding output type in volatile memory, send:
SPA 2 0x0A000003 2

In PIMikroMove, this parameter is available in the *DAC 2* parameter group in the *Device Parameter Configuration* window. Note that for the piezo amplifier channel, the output type is fixed to "control voltage of output signal channel" (1).

2 Connect the axis (identifier is 1) to the analog output line using the Select Output Index parameter, ID 0x0A000004.
Note:
The axis position results from the InputSignalChannel-to-Axis matrix transformation, see "Input Signal Processing" (p. 33) for more information.

In the example, send
SPA 2 0x0A000004 1
to connect the axis position to the analog output line (output signal channel 2).

In PIMikroMove, this parameter is also available in the *DAC 2* parameter group in the *Device Parameter Configuration* window.

3 Scale the output value, i.e. associate the axis position values with suitable output levels (= scaled position values). To do this, set the Position Report Scaling parameter, ID 0x07001005, and the Position Report Offset parameter, ID 0x07001006 to suitable values for the axis.

ScaledPositionValue = PositionReportScaling * (PositionReportOffset + PositionValue)

Example:
The position range of the axis (axis identifier is 1) is given by the TMN? answer (is defined by the Range Limit min parameter, ID = 0x07000000) and by the TMX? answer (is defined by the Range Limit max parameter, ID = 0x07000001), it is -20 µm to +120 µm in the example. Furthermore, the full output range of 0 V to +10 V is to be used (this is recommended for highest resolution). The resulting parameter values for the axis position scaling are as follows:
Position Report Scaling = 0.0714
Position Report Offset = 1.432
i.e. you have to send:
SPA 1 0x07001005 0.0714
SPA 1 0x07001006 1.432

In PIMikroMove, these parameters for the axis are available in the *Servo* parameter group in the *Device Parameter Configuration* window.

9.4 Adjusting the D/A Converter

It is necessary to adjust the digital/analog converter of the analog output line when the measured output value deviates from the response to the VOL? command for output signal channel 2. During the adjustment, the offset and gain are set for the digital/analog converter.

1 Get the current value of output signal channel 2. Send
VOL? 2

2 Determine the actual output value at the analog output (pin 20 of the I/O socket or, with E-709.CHG, “Monitor Out” SMB connection) by measuring with a connected measuring device.

3 If the queried value deviates from the measured value:

Send
SPA 2 0x0A000010 *Offset*
where *Offset* specifies the offset value for the digital/analog converter of output signal channel 2.

Send
SPA 2 0x0A000020 *Gain*
where *Gain* specifies the gain value for the digital/analog converter of output signal channel 2.

In PIMikroMove, these parameters are available in the *DAC 2* parameter group in the *Device Parameter Configuration* window.

4 Repeat steps 1, 2 and 3 in this order until the queried and the measured value match.

9.5 Analog-Output-Related Commands and Parameters

Command	Description	Notes
SEP	Set Nonvolatile Memory Parameters	Can be used to set the power-on default configuration for analog output usage.
SEP?	Get Nonvolatile Memory Parameters	Reads the current parameter values from non-volatile memory
SPA	Set Temporary Memory Parameters	Can be used to set a temporary configuration for analog output usage.
SPA?	Get Temporary Memory Parameters	Reads the current parameter values from volatile memory (RAM)
VOL?	Get Voltage Of Output Signal Channel	Reads output voltage value of the given output signal channel

Command	Description	Notes
WPA	Save Parameters To Nonvolatile Memory	Can be used to save the currently active configuration (including analog output usage) to non-volatile memory, where it becomes the power-on default.

See "How to work with the Analog Output - Overview" (p. 74) for more information. For detailed command descriptions see "Command Reference" (p. 129). For the identifiers of the items which can be addressed with the commands see "Axes, Channels, Functional Elements" (p. 23).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x07001005	1	Logical Axis	1	FLOAT	Position Report Scaling, required if the axis position is to be output (output type = 2)
0x07001006	1	Logical Axis	1	FLOAT	Position Report Offset, required if the axis position is to be output (output type = 2)
0x09000000 and 0x09000001	1	Logical Axis	1	FLOAT	Driving Factor of Piezo 1 and Driving Factor of Piezo 2, give the Axis-to-OutputSignalChannel matrix
0x0A000003	1	Output Signal Channel	2	INT	Select Output Type; 1 = control voltage of output signal channel 2 = current position of axis
0x0A000004	1	Output Signal Channel	2	INT	Select Output Index; the selected object can be an output signal channel or an axis (depends on the selected output type)
0x0A000010	1	Output Signal Channel	2	FLOAT	Offset for the D/A converter; adjusts the measured output value of the output signal channel to the response to the VOL? command.
0x0A000020	1	Output Signal Channel	2	FLOAT	Gain for the D/A converter; adjusts the measured output value of the output signal channel to the response to the VOL? command.

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x0A000100	3	Output Signal Channel	2	FLOAT	DAC Bit Width
0x0e000b01	3	System	1	INT	Number of output channels
0x0e000b04	3	System	1	INT	Number of piezo channels

See "Controller Parameters" (p. 223) for more information regarding the controller parameters and their handling.

10 Data Recording

10.1 How to Use the Data Recorder

The E-709 includes a real-time data recorder. It is able to record several input and output signals (e.g. current position, sensor input, output voltage) from different data sources (e.g. controller axes or input and output channels). The gathered data is stored (temporarily) in "data recorder tables"—each table contains the signal from one data source. You can configure the data recorder flexibly, e.g. select the type of data and the data source. Furthermore, you can choose the number of record tables and hence influence their size.

For general information regarding the data recording you can send HDR? (p. 153), which lists available options, and gives information about additional parameters and commands concerned with data recording.

How to Define What to Record—Set Record Options

The data recorder configuration, i.e. the assignment of data sources and record options to the recorder tables, can be changed with DRC (p. 146), and the current configuration can be read with DRC? (p. 148). Data recorder tables with record option 0 are deactivated, i.e. nothing is recorded. The default data recorder configuration is as follows:

- Data recorder table 1: Current position of axis 1
- Data recorder table 2: Target position of axis 1
- Data recorder table 3: Open-loop control value of axis 1
- Data recorder table 4: Piezo output voltage of output signal channel 1

How to Start Recording—Set Trigger Options

Recording can be triggered in several ways. Ask with DRT? (p. 151) for the current trigger option and use DRT (p. 150) to change it. A trigger option set with DRT will become valid for all data recorder tables with non-zero record option. Irrespective of the DRT settings, data recording is always triggered by the following four commands:

- STE (p. 175) (step response measurement),
- IMP (p. 160) (impulse response measurement),
- WGO (p. 200) (wave generator start; note that recording does not take place when wave generator is started by external trigger)

- WGR (p. 200) (restarts recording when the wave generator is running).

Recording always takes place for all data recorder tables with non-zero record option and ends when the data recorder tables are completely filled.

How to Read Recorded Data

The last recorded data can be read with the DRR? command (p. 149). The data is reported in GCS array format. For details regarding GCS array see the separate manual (SM146E), which is provided on the E-709 CD. Reading out recorded data 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.

The number of points comprised by the last recording can be read with the DRL? command (p. 148). This can be useful, for example, if you restart recording with WGR and want to read data while recording is still in progress.

How to Configure Number of Tables and Sampling Period

The number of available data recorder tables can be read with the TNR? (p. 182) command. The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. You can change the parameter value to increase or decrease the number of data recorder tables. Note that changing the number of data recorder tables deletes the content of all tables. For E-709, the number of tables must be in the range of 1 to 4.

The total number of points available for data recording is given by the Data Recorder Max Points parameter, ID 0x16000200. The controller allocates these points in equal shares to the available tables (i.e. to the number of tables given in the TNR? answer). For E-709, the total number of points is 4096. If, for example, TNR? replies 4, each table comprises 1024 points.

The data recorder sampling period can be read with the RTR? command (p. 167). The answer gives the value of the Data Recorder Table Rate parameter (ID 0x16000000) whose default value is one servo cycle. You can cover longer periods by increasing this value. Use the RTR command (p. 166) or change the parameter value directly.

Wherever changing parameter values is mentioned, you can do this using SPA (p. 170) (volatile memory) or SEP (p. 168) (non-volatile memory). Furthermore, you can use WPA (p. 202) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. Read parameter values with SPA? (p. 173) or SEP? (p. 169).

When the controller is powered down, the contents of the data recorder tables, the configuration settings and all settings which were only made in

volatile memory are lost. On power on, all settings are reset to their power-on defaults.

10.2 Data-Recorder Related Commands and Parameters

Command	Description	Notes
DRC (p. 146)	Set Data Recorder Configuration	Assigns data sources and record options to data recorder tables. Settings will be lost on controller power down or reboot.
DRC? (p. 148)	Get Data Recorder Configuration	Reads current data recorder settings
DRL? (p. 148)	Get Number of Recorded Points	Reads the number of points comprised by the last recording.
DRR? (p. 149)	Get Recorded Data Values	Reading can take some time, depending on the number of points.
DRT (p. 150)	Set Data Recorder Trigger Source	Defines the trigger source for all data recorder tables with non-zero record option. Settings will be lost on controller power down or reboot.
DRT? (p. 151)	Get Data Recorder Trigger Source	Reads current trigger settings
HDR? (p. 153)	Get All Data Recorder Options	Lists available options, gives information about additional parameters and commands concerned with data recording
IMP (p. 160)	Start Impulse and Response Measurement	Triggers recording
RTR (p. 166)	Set Record Table Rate	Changes the data recorder table rate in volatile memory (Data Recorder Table Rate parameter, ID 0x16000000)
RTR? (p. 167)	Get Record Table Rate	Reads the current setting of the data recorder table rate (Data Recorder Table Rate parameter, ID 0x16000000)
STE (p. 175)	Start Step and Response Measurement	Triggers recording
TNR? (p. 182)	Get Number of Record Tables	Reads the number of available data recorder tables (Data Recorder Chan Number parameter, ID 0x16000300)
WGO (p. 200)	Set Wave Generator Start/Stop Mode	Triggers recording, except when wave generator is started by external trigger
WGR (p. 200)	Start Recording Synchronous to Wave Generator	Triggers recording

See "How to use the Data Recorder" (p. 81) for more information. For detailed command descriptions see "Command Reference" (p. 129). For the identifiers of the items which can be addressed with the commands see "Axes, Channels, Functional Elements" (p. 23).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x16000000	1	System	1	INT	Data Recorder Table Rate
0x16000100	3	System	1	INT	Max Number of Data Recorder Channels
0x16000200	3	System	1	INT	Data Recorder Max Points
0x16000300	1	System	1	INT	Data Recorder Chan Number; the available data recorder points are allocated in equal shares to the number of tables given by this parameter

See "Controller Parameters" (p. 223) for more information regarding the controller parameters and their handling.

11 External Triggering/Signaling

11.1 Using Trigger Input and Output

The digital input and output lines of the E-709 are present on the "I/O" socket (p. 260 or p. 261). Using the TIO? command (p. 181), you can query the number of I/O lines available on the E-709.

The Digital_IN_1 input line (LVTTL, active high) can be used to trigger several items/events. The trigger input can be configured using the CTI command in conjunction with parameter settings. See the description of the CTI command for details (p. 140). If the wave generator is to be triggered (default setting), in addition the "start via external trigger signal" start mode (bit 1) must be set with the WGO command (p. 197). See "Wave Generator Started by Trigger Input" (p. 104) for an example.
The trigger input functionality must be enabled using the TRI command (p. 184).
Using the DIO? command (p. 146), you can query the state of the digital input line.

You can program the Digital_OUT_1 and Digital_OUT_2 output lines (LVTTL, active high) to trigger other devices using CTO (configures trigger output, p. 142). If CTO is used in combination with TWS (sets trigger line action to waveform point, p. 187), the trigger output will be synchronized with the wave generator output. See "Configuring Trigger Output" (p. 85) and "Trigger Output Synchronized with Wave Generator" (p. 103) for examples.

11.2 Configuring Trigger Output

You can program the digital output lines of the E-709 to trigger other devices using the CTO command (p. 142).

The format of the CTO command is as follows (i.e. one setting can be made per command):

CTO <TrigOutID> <CTOPam> <Value>

The following trigger modes are supported by the E-709:

- 0 = Position Distance; a trigger pulse is written whenever the axis has covered a given distance. 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). See "Example—"Position Distance" Trigger Mode" (p. 86)

- 2 = OnTarget; the on-target status of the selected axis is written to the selected trigger output line (this status can also be read with the ONT? command). See "Example—"On Target" Trigger Mode" (p. 87)

- 3 = MinMaxThreshold; values for MinThreshold and MaxThreshold must be defined. When the axis position of the selected axis is inside the band specified by the MinThreshold and MaxThreshold values, the selected trigger output line is set high, otherwise it is set low. See "Example—"MinMax Threshold" Trigger Mode" (p. 88)

- 4 = Generator Trigger; the trigger line action must be defined with TWS (p. 187). See "Example—"Generator Trigger" Mode" (p. 90) and "Trigger Output Synchronized with Wave Generator" (p. 103)

- 6 = InMotion; the selected trigger line is active as long as the selected axis is in motion. See "Example—"In Motion" Trigger Mode" (p. 90)

To select the mode, set <CTOPam> = 3 and <Value> to the code of the mode; by default InMotion (6) is selected for Digital_OUT_1 and On Target (2) for Digital_OUT_2.

The polarity (active high / active low) of the signal can be set at the digital output (<CTOPam> ID 7).

The following examples can be reproduced using the command entry facilities of PIMikroMove or PI Terminal.

11.2.1 Example—"Position Distance" Trigger Mode

The "Position Distance" trigger mode is designed for scanning applications. A trigger pulse is written whenever the axis has covered the distance given by the <TriggerStep> parameter of the CTO command. The pulse width is 1 µs.

The unit of <TriggerStep> is µm.

Send a sequence of the following commands for the digital output line (<TrigOutID>) which is to be used for trigger output (the order of the commands is irrelevant):

CTO <TrigOutID> 2 *Axis*
CTO <TrigOutID> 3 *Triggermode*
CTO <TrigOutID> 1 *Stepsize*

Example: A pulse on the digital output line 1 is to be generated whenever axis 1 has covered a distance of 0.1 µm. Send:

CTO 1 2 1
CTO 1 3 0

CTO 1 1 0.1

Optionally, you can define start and stop values for limiting the range and for specifying the motion direction of the axis (positive or negative). Send a sequence of the following commands for the digital output line (<TrigOutID>) which is to be used for trigger output (the order of the commands is irrelevant):

CTO <TrigOutID> 2 *Axis*
CTO <TrigOutID> 3 *Triggermode*
CTO <TrigOutID> 1 *Stepsize*
CTO <TrigOutID> 8 *StartValue*
CTO <TrigOutID> 9 *StopValue*

Note that 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.

Example: A pulse is to be output on digital output line 1 every time axis 1 has covered a distance of 0.1 µm, as long as axis 1 is moving in the 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 0.1
CTO 1 8 0.2
CTO 1 9 0.55

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

CTO 1 2 1
CTO 1 3 0
CTO 1 1 0.1
CTO 1 8 0.55
CTO 1 9 0.2

11.2.2 Example—"On Target" Trigger Mode

With the "On Target" trigger mode, the on-target status of the selected axis is written to the selected trigger line. It is the same on-target status flag which can also be read by the ONT? command (p. 163). The on-target status is detected only in closed-loop operation (servo ON) and is influenced by two parameters: settling window (On Target Tolerance, ID 0x07000900) and settling time (On Target Settling Time, ID 0x07000901). The on-target status is true when the current position is inside the settling

window and stays there for at least the settling time. The settling window is centered around the target position.

Send a sequence of the following commands for the digital output line (<TrigOutID>) which is to be used for trigger output (the order of the commands is irrelevant):

CTO <TrigOutID> 2 *Axis*
CTO <TrigOutID> 3 *Triggermode*

Example: The On-Target status flag of axis 1 is to be written to the digital output line 1. Send:

CTO 1 2 1
CTO 1 3 2

11.2.3 Example—"MinMax Threshold" Trigger Mode

With the "MinMax Threshold" trigger mode, a band is specified with MinThreshold and MaxThreshold (<CTOPam> IDs 5 and 6). When the axis position is inside the specified band then the trigger output line is set high, otherwise it is set low. Note that if the value of MinThreshold is larger than the value of MaxThreshold, then the trigger output line will never be set high.

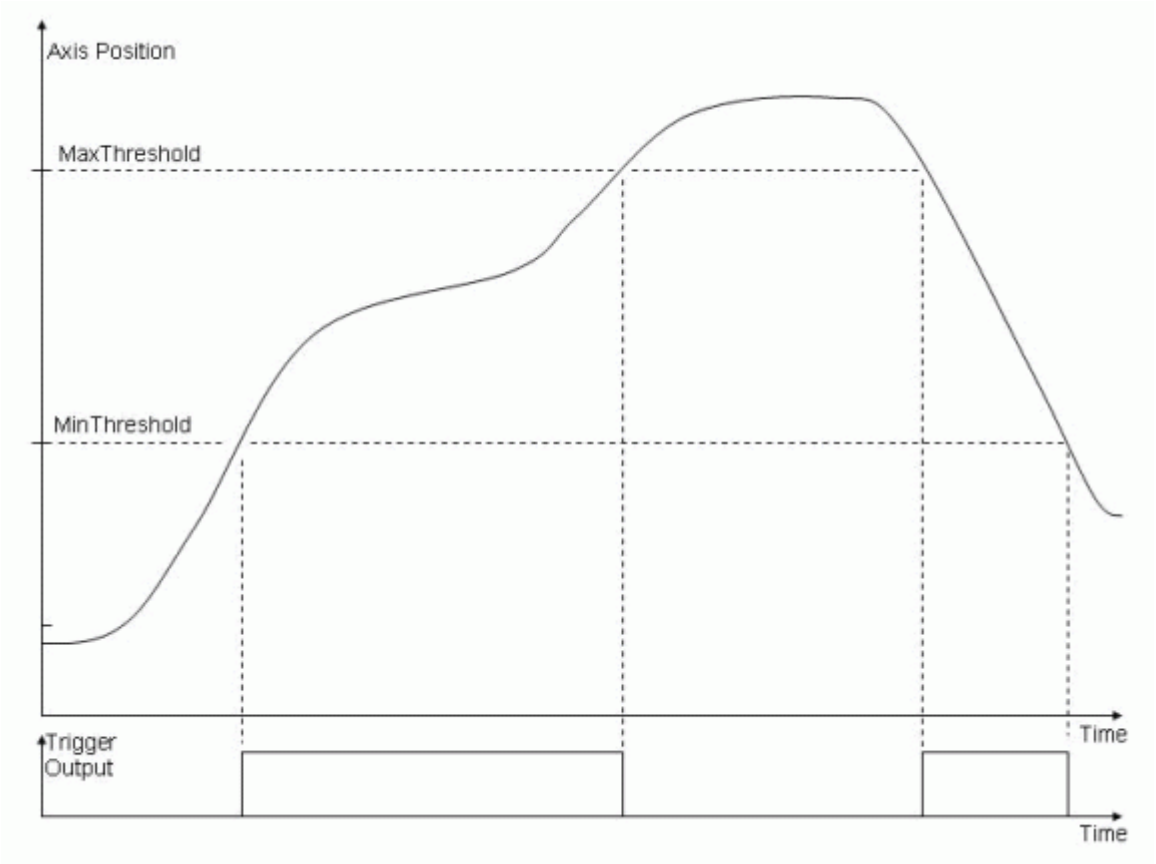


Figure 18: "MinMax Threshold" Trigger Mode

Send a sequence of the following commands for the digital output line (<TrigOutID>) which is to be used for trigger output (the order of the commands is irrelevant):

CTO <TrigOutID> 2 *Axis*
CTO <TrigOutID> 3 *Triggermode*
CTO <TrigOutID> 5 *min.pos*.
CTO <TrigOutID> 6 *max.pos.*

Example: The digital output line 1 is to be set high whenever the axis position of axis 1 is higher than 0.3 µm and lower than 0.6 µm. Send:

CTO 1 2 1
CTO 1 3 3
CTO 1 5 0.3
CTO 1 6 0.6

11.2.4 Example "Generator Trigger" Mode

With the "Generator Trigger" mode, the trigger output will be synchronized with the wave generator output, and CTO must be used in combination with TWS (p. 187).

Send the following command for the digital output line (<TrigOutID>) which is to be used for trigger output:

CTO <TrigOutID> 3 *Triggermode*

where *Triggermode* must be 4

See "Trigger Output Synchronized with Wave Generator" (p. 103) for a detailed example.

11.2.5 Example—"In Motion" Trigger Mode

With the "In Motion" trigger mode, the selected trigger line is active as long as the selected axis is in motion.

An axis is considered as to be "in motion" when

- AutoZero procedure is running
- Wave generator is running
- In closed-loop operation only: the current position is outside the settling window (On Target Tolerance, ID 0x07000900)

You can use #5 (p. 129) to check if an axis is in motion.

Send a sequence of the following commands for the digital output line (<TrigOutID>) which is to be used for trigger output (the order of the commands is irrelevant):

CTO <TrigOutID> 2 *Axis*
CTO <TrigOutID> 3 *Triggermode*

Example: The digital output line 1 is to be active as long as axis 1 is in motion. Send:

CTO 1 2 1
CTO 1 3 6

12 Wave Generator

The axis can be controlled by a "wave generator" which outputs user-specified patterns, so-called "waveforms". This feature is especially important in dynamic applications which require periodic, synchronous motion of the axis. The waveforms to be output are stored in "wave tables" in the controllers non-volatile flash memory—one waveform per wave table. Waveforms can be created based on predefined "curve" shapes. This can be sine, ramp or single scan line curves. Programmable trigger inputs and outputs facilitate synchronization of external events.

In "How to Work with the Wave Generator" (p. 91) and "Wave Generator Examples" p. 98) you will learn how to use the wave generator, and "Wave-Generator-Related Commands and Parameters" (p. 105) gives an overview.

During the wave generator output, data is recorded in "record tables" on the controller. See "Data Recording" (p. 74) for more information.

12.1 How to Work with the Wave Generator

The following subsections describe the wave generator handling in detail. See also "Wave Generator Examples" (p. 98).

12.1.1 Basic Data

The number of wave tables can be queried using the SPA? command (p. 173), parameter ID 0x1300010A. The E-709 has 6 wave tables for creating and storing arbitrary waveforms (identifiers are 1 to 6).
To ask for the number of wave generators, use the TWG? command (p. 186). As a single-axis controller, the E-709 has only one wave generator (identifier is 1).
A certain amount of the controllers memory space is reserved for the waveform data (ask with the SPA? command (p. 173), parameter ID 0x13000004). E-709 provides 16306 data points for waveform definition. This memory space is allocated to the individual wave tables during the waveform definition.

12.1.2 Basic Operation

NOTICE

Deterioration of the storage integrity.

Write waveforms only when necessary.

The non-volatile flash memory has a finite number of erase-write cycles. It is guaranteed to withstand about 10,000 write-erase-cycles, before the wear begins to deteriorate the integrity of the storage.

1. Define the waveform segment-by-segment using the WAV command (p. 189). The waveform will be written to the selected wave table in non-volatile flash memory.
2. Connect the wave generator to the wave table using the WSL command (p. 204).
3. Start the wave generator output and hence the motion of the axis using the WGO command (p. 200). You can choose several start options (e.g. start/stop by external trigger; see the description of the WGO command for more information).
   When starting the wave generator, data recording is started automatically, unless the wave generator is started by an external trigger.
4. Stop the wave generator output with WGO or #24 (p. 131) or STP (p. 176).

A simple example for your first steps (using the command entry facilities of PIMikroMove or PITerminal):

Command String to Send	**Action Performed**
WAV 4 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 4; see WAV description for details
WSL 1 4	Connect the Wave Generator 1 (axis 1) to Wave Table 4
WGO 1 1	Start output of Wave Generator 1 immediately (synchronized by servo cycles)
WGO 1 0	Stop output of Wave Generator 1

12.1.3 Additional Steps and Settings

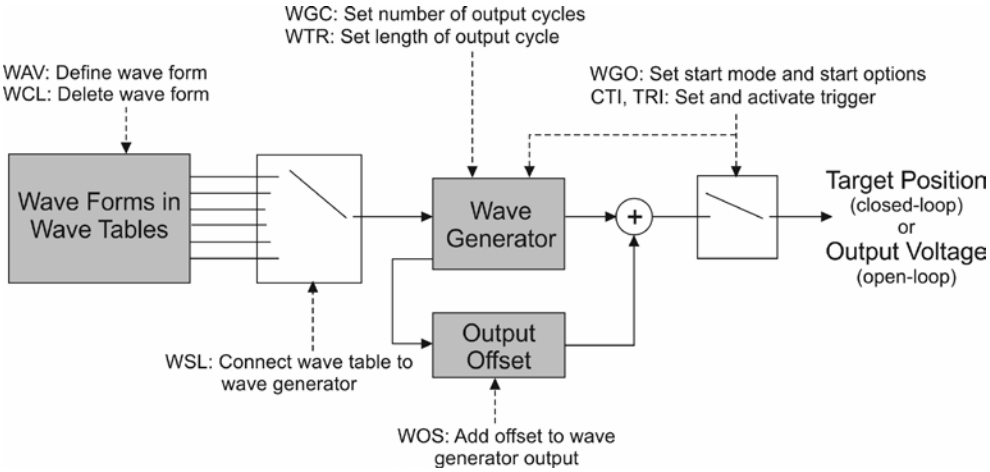


Figure 19: Wave generator block diagram

How to Check and Delete Wave Table Content

You can calculate the memory space remaining if you ask with WAV? (p. 194) for the current wave table length. To release memory space, delete the content of selected wave tables with the WCL command (p. 195).

After you have sent the waveform definition to the wave table (with WAV), it is always a good idea to check it by reading back the waveform sequence from the controller before actually outputting it. This can be done using the GWD? command (p. 152). Note that the response to GWD? does not contain any offset set with WOS (p. 201) to the wave generator output.

How to Add Offset to the Wave Generator Output

You can add an offset to the output of a wave generator using the WOS command (p. 201). Thereafter, the output of the specified wave generator is the sum of the offset value and the wave value:
Generator Output = Offset + Current Wave Value
WOS sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory. You can also change this parameter with SPA (p. 170) or SEP (p. 168) and save the value to non-volatile memory with WPA (p. 202). Deleting wave table content with WCL (p. 195) does not affect the WOS settings.

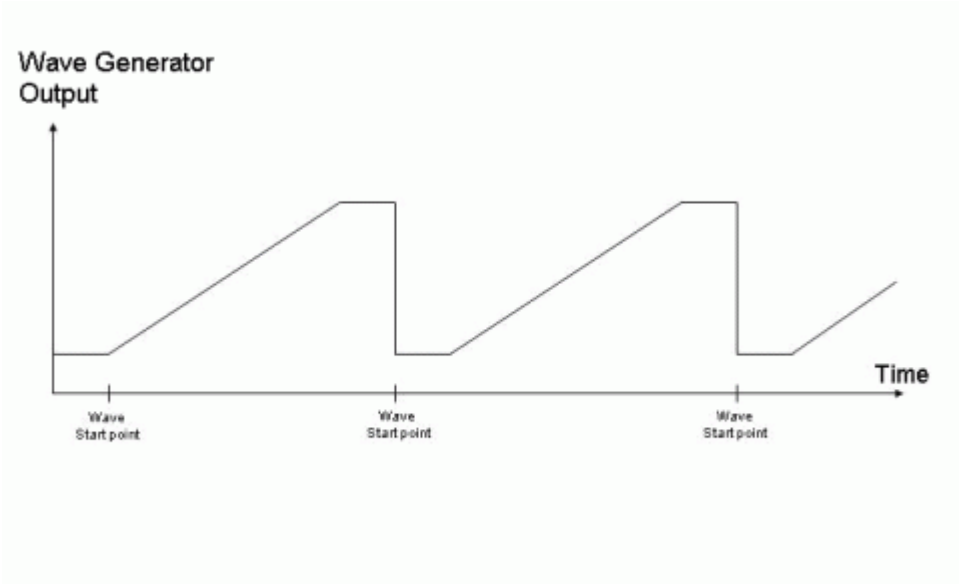


Figure 20: Wave generator started without "start at the endpoint of last cycle" (WGO bit 8 not set)

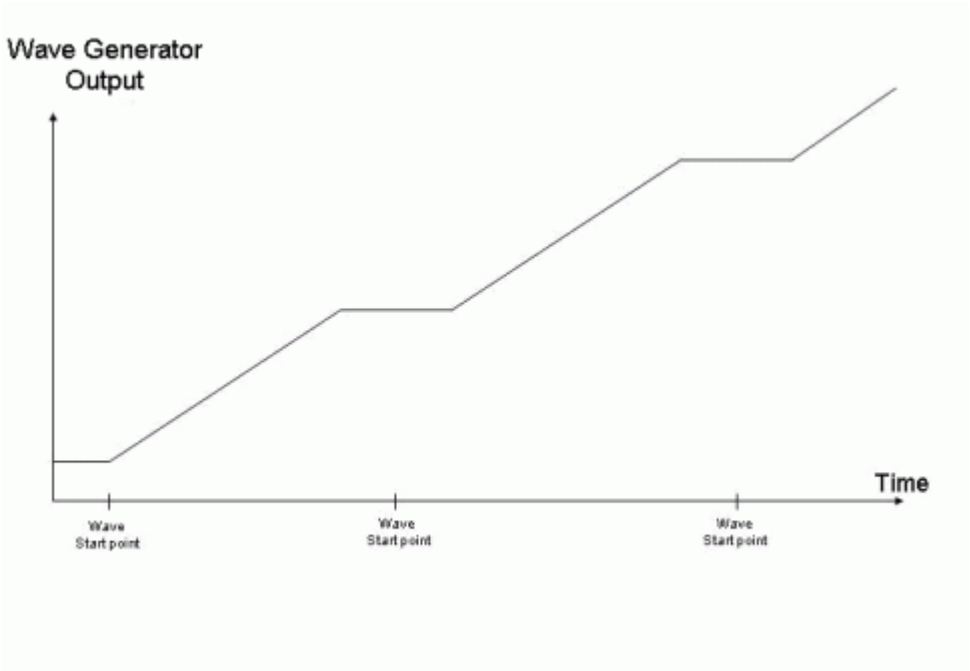


Figure 21: Wave generator started with "start at the endpoint of last cycle" (WGO bit 8)

How to Trigger External Devices While Wave Generator is Running

For triggering purposes, the wave generator output can be coupled with the digital output lines Digital_OUT_1 and Digital_OUT_2 of the controller (see "I/O" socket (p. 260 or p. 261)). You should first use TWC (p. 186) to set the

signal state of the output line to "low" for all waveform points ("low" is also the power-on default). Then use the TWS command (p. 187) to define the trigger line actions by setting the desired signal states of the output lines (high or low) for selected waveform points. At last, use the CTO command (p. 142) to activate the Generator Trigger mode for the output line. Note that the trigger line actions defined with TWS are valid for both digital output lines since the lines share a common definition table.

How to Check Activation State of Wave Generator

The #9 single-character command (p. 130) can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode. With WGO? (p. 200) you can ask for the last-commanded wave generator start options (WGO settings).

How to Define the Number of Output Cycles

You can limit the duration of the wave generator output by setting the number of output cycles with WGC (p. 195). The waveform itself remains unchanged.

How to Lengthen Output Cycles

Using the WTR command (p. 205), you can lengthen the individual output cycles of the waveform. The duration of one output cycle for the waveform can be calculated as follows:
Output Duration = Servo Update Time * WTR value * Number of Points
where
Servo Update Time is given in seconds by parameter 0x0E000200
WTR value gives the number of servo cycles the output of a waveform point lasts, default is 1
Number of Points is the length of the waveform (i.e. the length of the wave table)

WTR sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory. You can change this parameter also with SPA (p. 170) or SEP (p. 168) and save the value to non-volatile memory with WPA (p. 202). The value of the parameter in volatile memory can be read with the WTR? command (p. 205).

How to Restart Data Recording

With WGR (p. 200) you can restart data recording while the wave generator is running. The recorded data can be read with the DRR? command (p. 149). See "Data Recording" (p. 74) for more information.

12.1.4 Application Notes

Waveform Changes

Waveforms cannot be changed while they are being output by a wave generator. If you want to modify a waveform with WAV, first stop any wave generator output from the associated wave table.

Limited Frequency

The frequency of the wave generator output depends, among other factors, on the wave table length and on the wave generator table rate (WTR command). When you create waveforms, keep in mind that the usable frequency is limited by the available amplifier power. If the frequency is too high, a current limitation will be applied that cuts off the waveform amplitude.

Wave Generator Output Related to Other Control Sources

Wave generator output and analog control input:
It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In that case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be commanded by analog control input, you can stop the wave generator output, but not restart it.
Wave generator output and move commands:
When the wave generator output is active, move commands like MOV (p. 161) or SVA (p. 176) are not allowed for the associated axis.
See "Axis Motion" (p. 26) for details.

Interpretation of Wave Generator Output

The wave generator outputs absolute values. In closed-loop operation (servo ON), the output is interpreted as target positions. In open-loop operation (servo OFF), the output is interpreted as piezo output voltage values. Hence the axis can perform jumps when you switch the servo on or off (SVO (p. 178)) while the wave generator is running.

What can be Modified when the Wave Generator is Running

As long as a wave generator is running, it is not possible to change (WSL (p. 204)) or to delete (WCL (p. 195)) the connected wave table (i.e. the waveform). The wave generator table rate (WTR (p. 205)), the number of output cycles (WGC (p. 195)), the wave offset (WOS (p. 201)) and the output trigger settings (TWS (p. 187)) can be modified while a wave generator is running.

Starting and Stopping Wave Generator Output

When a wave generator finishes by running through a specified number of cycles completely, the final position will be the last point of the waveform, unless the option "start at the endpoint of the last cycle" was selected. In that case, the final position is the endpoint of the last output cycle. When the wave generator is stopped within an output cycle by command, the axis will remain at the last output position until a new position is commanded. If the wave generator is then restarted, it will normally continue with the first point of the waveform, unless started with the option "start wave generator output triggered by external signal".

By starting the wave generator with the option "start wave generator output triggered by external signal" (WGO bit 1), the Digital_IN_1 line (pin 10 of the "I/O" socket (p. 260 or p. 261)) is used for triggering, provided that the trigger functionality is enabled with TRI (p. 184). The trigger configuration is set with CTI and the Digital Trigger Input Usage parameter (ID 0x15000800; see p. 140). You can start data recording using WGR (p. 200) when the wave generator is running.

Wave generator output will continue even if the terminal or the program from which it was started is quit or if the high voltage output is deactivated.

See the WGO command (p. 200) for more information.

Temporary and Permanent Settings

The following settings are always lost when the controller is powered down or rebooted:

- Assignment of wave tables to wave generators (WSL (p. 204))
- Output trigger settings (TWS (p. 187))
- Number of cycles for wave generator output (WGC (p. 195))

The following settings can be saved with WPA (p. 202) to non-volatile memory, where they become the power-on defaults:

- Wave offset (WOS (p. 201))
- Wave generator table rate (WTR (p. 205))

Software Support for Wave Generator Handling

The different software interfaces provided for the controller also support use of the wave generator. Waveforms can be defined, stored and displayed in and by the software in a more user-friendly way than in a terminal using WAV and WGO. If you intend to use the wave generator with the GCS DLL, PIMikroMove, or LabView, read the descriptions in the associated software manual first.

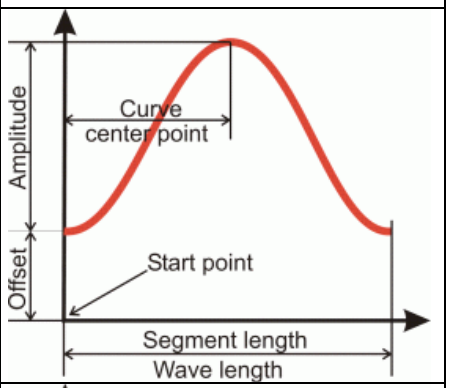
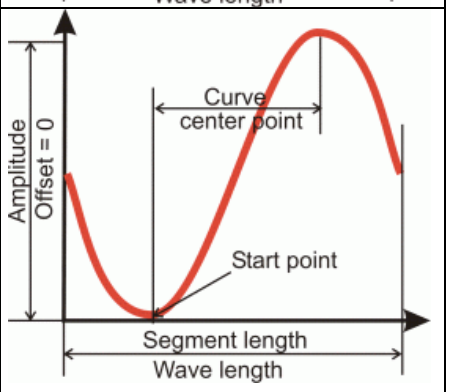
12.2 Wave Generator Examples

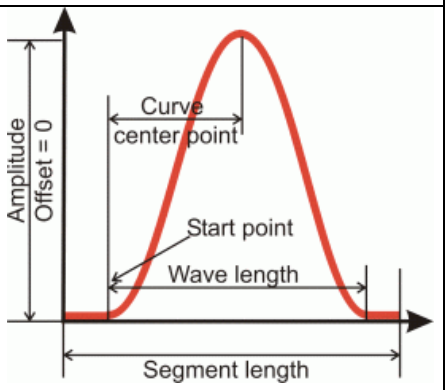
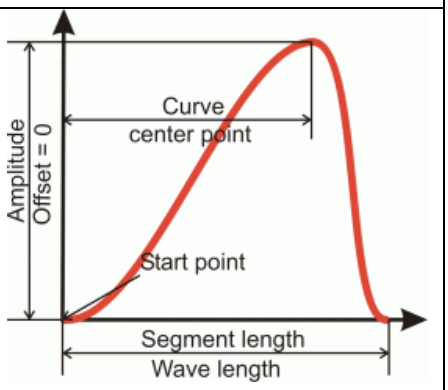
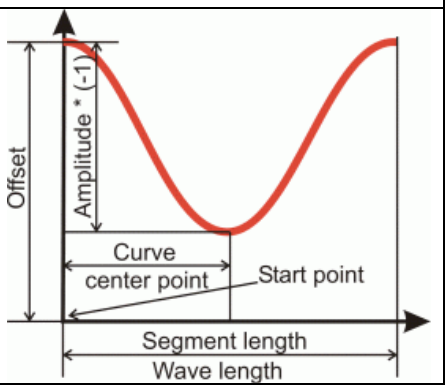
The following examples can be reproduced using the command entry facilities of PIMikroMove or PI Terminal. Note that it might be necessary to adapt them to your hardware configuration.

12.2.1 Defining Waveforms

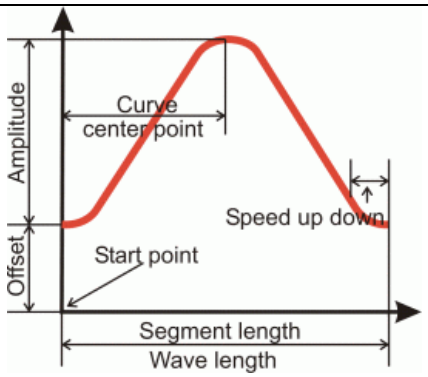
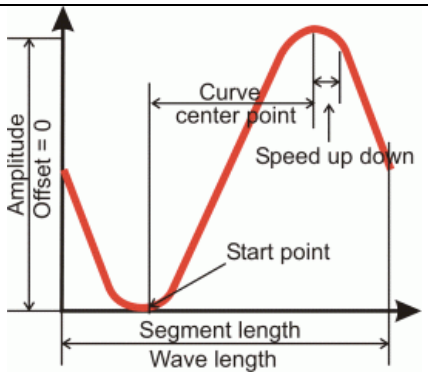
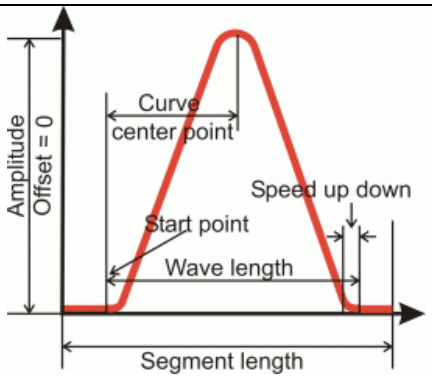
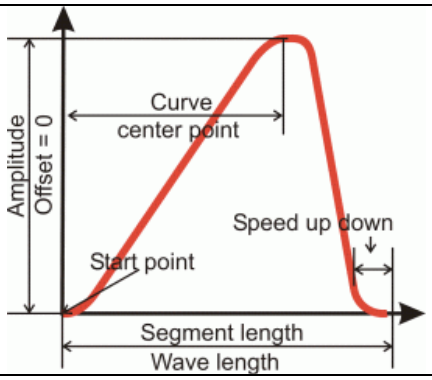
Examples for how to define waveform segments for the wave tables, based on predefined curve shapes (each WAV command defines a waveform segment which either replaces or is appended to the waveform in the specified wave table):

Inverted Cosine Curves

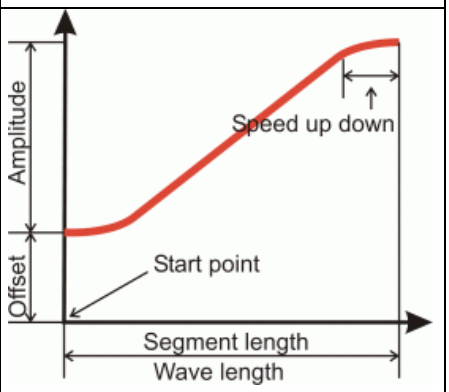
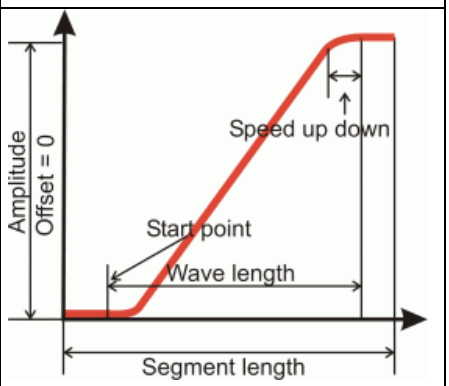
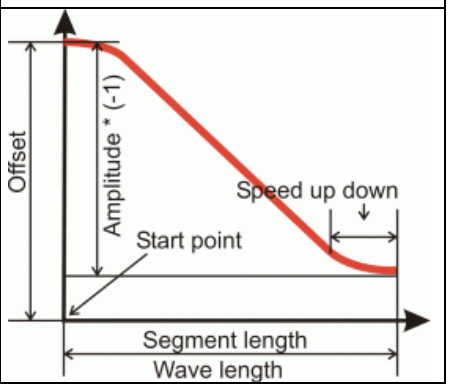
WAV command	Comments	Waveform Segment
WAV 2 X SIN_P 2000 20 10 2000 0 1000 <WaveTableID> = 2 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 20 <Offset> = 10 <WaveLength> = 2000 <StartPoint> = 0 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, waveform offset = 10 (Do not confuse with the wave generator output offset set with WOS!), symmetric curve	
WAV 2 X SIN_P 2000 30 0 2000 499 1000 <WaveTableID> = 2 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 30 <Offset> = 0 <WaveLength> = 2000 <StartPoint> = 499 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, symmetric curve	

WAV command	Comments	Waveform Segment
WAV 2 & SIN_P 2000 25 0 1800 110 900 <WaveTableID> = 2 <AppendWave> = & <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 25 <Offset> = 0 <WaveLength> = 1800 <StartPoint> = 110 <CurveCenterPoint> = 900	The defined segment will be appended to the existing wave table contents, symmetric curve	
WAV 3 X SIN_P 4000 20 0 4000 0 3100 <WaveTableID> = 3 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 4000 <Amp> = 20 <Offset> = 0 <WaveLength> = 4000 <StartPoint> = 0 <CurveCenterPoint> = 3100	The previous contents of the wave table are overwritten by the new segment, asymmetric curve	
WAV 1 X SIN_P 1000 -30 45 1000 0 500 <WaveTableID> = 1 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 1000 <Amp> = -30 <Offset> = 45 <WaveLength> = 1000 <StartPoint> = 0 <CurveCenterPoint> = 500	The previous contents of the wave table are overwritten by the new segment, negative-amplitude curve, symmetric curve	

Ramp Curves

WAV command	Comments	Waveform Segment
WAV 4 X RAMP 2000 20 10 2000 0 300 1000 <WaveTableID> = 4 <AppendWave> = X <WaveType> = RAMP <SegLength> = 2000 <Amp> = 20 <Offset> = 10 <WaveLength> = 2000 <StartPoint> = 0 <SpeedUpDown> = 300 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, waveform offset = 10 (Do not confuse with the wave generator output offset set with WOS!) symmetric curve	
WAV 4 X RAMP 2000 35 0 2000 499 300 1000 <WaveTableID> = 4 <AppendWave> = X <WaveType> = RAMP <SegLength> = 2000 <Amp> = 35 <Offset> = 0 <WaveLength> = 2000 <StartPoint> = 499 <SpeedUpDown> = 300 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, symmetric curve	
WAV 5 X RAMP 2000 15 0 1800 120 150 900 <WaveTableID> = 5 <AppendWave> = X <WaveType> = RAMP <SegLength> = 2000 <Amp> = 15 <Offset> = 0 <WaveLength> = 1800 <StartPoint> = 120 <SpeedUpDown> = 150 <CurveCenterPoint> = 900	The previous contents of the wave table are overwritten by the new segment, symmetric curve	
WAV 5 & RAMP 3000 35 0 3000 0 200 2250 <WaveTableID> = 5 <AppendWave> = & <WaveType> = RAMP <SegLength> = 3000 <Amp> = 35 <Offset> = 0 <WaveLength> = 3000 <StartPoint> = 0 <SpeedUpDown> = 200 <CurveCenterPoint> = 2250	The defined segment will be appended to the existing wave table contents, asymmetric curve	

Single Scan Line Curves

WAV command	Comments	Waveform Segment
WAV 1 X LIN 1500 30 15 1500 0 370 <WaveTableID> = 1 <AppendWave> = X <WaveType> = LIN <SegLength> = 1500 <Amp> = 30 <Offset> = 15 <WaveLength> = 1500 <StartPoint> = 0 <SpeedUpDown> = 370	The previous contents of the wave table are overwritten by the new segment, waveform offset = 15 (Do not confuse with the wave generator output offset set with WOS!)	
WAV 2 X LIN 1500 40 0 1100 210 180 <WaveTableID> = 2 <AppendWave> = X <WaveType> = LIN <SegLength> = 1500 <Amp> = 40 <Offset> = 0 <WaveLength> = 1100 <StartPoint> = 210 <SpeedUpDown> = 180	The previous contents of the wave table are overwritten by the new segment	
WAV 2 & LIN 3000 -40 50 3000 0 650 <WaveTableID> = 2 <AppendWave> = & <WaveType> = LIN <SegLength> = 3000 <Amp> = -40 <Offset> = 50 <WaveLength> = 3000 <StartPoint> = 0 <SpeedUpDown> = 650	The defined segment will be appended to the existing wave table contents, negative-amplitude curve	

12.2.2 Modifying the Wave Generator Table Rate

An example for how to modify the duration of the wave generator output using the wave table rate:

Command String to Send	**Action Performed**
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
SPA? 1 0x0E000200	Ask for the servo update time of the controller (reading the wave table for wave generator output is clocked by servo cycles). E-709 has as a servo update time of 100 µs.
WTR?	Ask for the current wave table rate setting, default is wave table rate = 1 (i.e. each wave table point will be output for a duration of one servo cycle). The duration of one wave generator output cycle will be: Servo Update Time (in s) * WTR value * Number of Points = Output Duration (in s) 0.0001 s * 1 * 2000 = 0.2 s
WTR 1 3 0	Set the wave table rate to 3, tripling the duration of one wave generator output cycle (each wave table point will now "occupy" 3 servo cycles). Duration of one output cycle will now be: 0.0001 s * 3 * 2000 = 0.6 s

12.2.3 Trigger Output Synchronized with Wave Generator

Using the digital output lines of the E-709, it is possible to trigger external devices. See "I/O" socket (p. 260 or p. 261) for the availability of the lines (pinout) and "Configuring Trigger Output" (p. 85) for trigger applications without wave generator usage.

An example for how to generate trigger pulses synchronized with the wave generator:

Command String to Send	**Action Performed**
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
TWC	Clears all output trigger settings related to the wave generator by switching the signal state for all points to "low" (the power-on default state is also "low"). It is recommended to use TWC before new trigger actions are defined.
TWS 1 500 1 TWS 1 1500 1 TWS 1 1900 1 TWS 1 2000 1	Set trigger actions for Digital_OUT_1 (identifier is 1): at the waveform points 500, 1500, 1900 and 2000 it is set high; at all other points the state of the line is low (due to the TWC usage).
CTO 1 3 4	Digital_OUT_1 is set to "Generator Trigger" mode.
WSL 1 2	Connect Wave Generator 1 (Axis 1) to Wave Table 2
WGO 1 1	Start output of Wave Generator 1 immediately (synchronized by servo cycle). Now the trigger output action will take place as specified.
WGO 1 0	Stop output of Wave Generator 1 and hence also the trigger output.

12.2.4 Wave Generator Started by Trigger Input

Using the Digital_IN_1 line (pin 10 of the "I/O" socket (p. 260 or p. 261)), it is possible to apply start/stop signals for the wave generator output.

An example for how to start / stop the wave generator by external trigger signals:

Command String to Send	**Action Performed**
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
WSL 1 2	Connect Wave Generator 1 (Axis 1) to Wave Table 2
SPA? 1 0x15000800	The response should be 1 which means that the digital input line starts/interrupts the wave generator output. If the response differs from 1, use SPA to set the parameter value to 1.
CTI?	Check the trigger configuration since the wave generator output depends on the selected trigger type (CTIPam 1). Possible types: “edge triggered” or “level triggered”. Edge triggered: Each activating state transition of the digital input line triggers the output of a point in the wave table. When an output rate > 1 is set with WTR, the corresponding number of activating state transitions is required to output a point. Level triggered (default): When the digital input line is in an active state, the wave generator outputs the points of the wave table. When the digital input line is in a non-active state, the wave generator output is interrupted.
TRI 1 1	Enable the trigger input functionality for the digital input line.
WGO 1 2	Start Wave Generator 1 triggered by external signal. To provide the external signal, use the Digital_IN_1 line. The generator output depends on the trigger configuration (level triggerd or edge triggered).

Command String to Send	Action Performed
WGO 1 0	Stop output of Wave Generator 1 (any further triggering will be ignored). You can also use #24 or STP.

12.3 Wave-Generator-Related Commands and Parameters

Command	Description	Notes
CTI (p. 140)	Set Configuration of Trigger Input	Configures the trigger input for the given digital input line; required when the wave generator output is to be started by an external trigger.
CTO (p. 142)	Set Configuration Of Trigger Output	Activates the Generator Trigger output mode which is required for the trigger line actions set with TWS.
DRR? (p. 149)	Get Recorded Data Values	Reads the last recorded data. Data recording is triggered by the WGO and WGR commands (among others).
GWD? (p. 152)	Get Wave Table Data	Should be used to check the waveform before the wave generator output is started.
TRI (p. 184)	Set Trigger Input State	Enables the trigger input functionality; required when the wave generator output is to be started by an external trigger.
TWC (p. 186)	Clear All Wave Related Triggers	Clears only the TWS settings, but not the CTO settings.
TWG? (p. 186)	Get Number Of Wave Generators	Number of wave generators = number of axes
TWS (p. 187)	Set TriggerLine Action To Waveform Point	In addition, the CTO command must be used to activate the Generator Trigger mode for the desired digital output line.
WAV (p. 189)	Set Waveform Definition	A waveform must be defined before the wave generator output can be started. Waveforms are stored in non-volatile flash memory which provides about 10,000 erase-write cycles!

Command	Description	Notes
WAV? (p. 194)	Get Waveform Definition	Reads the current wave table length.
WCL (p. 195)	Clear Wave Table Data	Clears the wave table content, but not the WSL and WOS settings.
WGC (p. 195)	Set Number Of Wave Generator Cycles	If WGC is not used, the wave generator must be stopped with WGO (p. 200), #24 (p. 131) or STP (p. 176).
WGC? (p. 195)	Get Number Of Wave Generator Cycles	
WGO (p. 197)	Set Wave Generator Start/Stop Mode	The WGO command starts the wave generator output. It provides several start options, e.g. "Start wave generator output triggered by external signal" or "Start at the endpoint of last cycle".
WGO? (p. 200)	Get Wave Generator Start/Stop Mode	Gets the last commanded start options, but not the activation status (use #9 instead)
WGR (p. 200)	Starts Recording in Sync with Wave Generator	Restarts data recording as long as a wave generator is running.
WOS (p. 201)	Set Wave Generator Output Offset	Sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory.
WOS? (p. 202)	Get Wave Generator Output Offset	Gets the value of the Wave Offset parameter, ID 0x1300010b, from volatile memory.
WSL (p. 204)	Set Connection Of Wave Table To Wave Generator	Must be set before the wave generator can be started.
WSL? (p. 204)	Get Connection Of Wave Table To Wave Generator	
WTR (p. 205)	Set Wave Generator Table Rate	Sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory.
WTR? (p. 206)	Get Wave Generator Table Rate	Gets the value of the Wave Generator Table Rate parameter (ID 0x13000109) from volatile memory.

Command	Description	Notes
#9 (p. 130)	Get Wave Generator Status	Gets the current activation status of the wave generator, but not the start options (use WGO? instead)

See "How to Work with the Wave Generator" (p. 91) for more information.
For detailed command descriptions see "Command Reference" (p. 129).
For the identifiers of the items which can be addressed with the commands see "Axes, Channels, Functional Elements" (p. 23).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x13000004	3	System	1	INT	Max Wave Points
0x13000109	0	System	1	INT	Wave Generator Table Rate
0x1300010a	3	System	1	INT	Number of Waves
0x1300010b	0	Logical Axis	1	FLOAT	Wave Offset
0x15000800	1	System	1	INT	Digital Trigger Input Usage

See "Controller Parameters" (p. 223) for more information regarding the controller parameters and their handling.

13 Servo-Controller Dynamic Tuning

If the controller and the attached piezo stages are ordered together and if PI has sufficient knowledge of your application, then the parameters of the closed-loop control algorithm (servo parameters) will be set to suitable values at the factory, and, if present, saved in the stage's ID-chip (p. 120). Modification of those parameters will, however, be necessary if the load applied to the piezo stage is changed.

13.1 Parameters to be Modified

The following parameters may need to be modified in the E-709:

- Settings for notch filters 1 and 2:
  Notch frequency 1 (parameter ID 0x08000100), notch frequency 2 (parameter ID 0x08000101)
  The notch filters are also active in open-loop operation.

  To determine the resonant frequencies and set the notch filters properly, observe the system response to an impulse in open-loop operation (p. 110).

- Servo-control parameters:
  P-term (parameter ID 0x07000300), I-term (parameter ID 0x07000301), servo-loop slew rate (parameter ID 0x07000200):

  Normally the proper P-term and I-term settings are found by observing the response of the axis to an abrupt change of the control value (step response) in closed-loop operation (p. 115).

If there was no parameter adjustment before shipment, the following default values are set:

- The two notch filters of the E-709 are deactivated (notch rejection parameters 0x08000200 and 0x08000201 are set to 1).
- The slew rate is not limited (servo loop slew rate 0x07000200, open-loop slew rate 0x07000201).
- P-term (ID 0x07000300) and I-term (ID 0x07000301) both are set to 0.01.

13.2 General Notes on Servo-Controller Dynamic Tuning

NOTICE

If the stage starts oscillating (humming noise):

In closed-loop operation, switch off the servo immediately. Switch the servo on only after you have adjusted the notch filter(s) and the servo-control parameters (P-term, I-term).

In open-loop operation, stop the axis motion immediately.

Otherwise the stage could be irreparable damaged.

- Before you change parameter values of the E-709, create a backup file. See "Creating Backup File for Controller Parameters" (p. 58) for more information.
- Enter the password “advanced” when prompted to change to command level 1.
- For stages with ID-chip, to make the optimized settings available in the future, the option "Power Up Read ID-Chip" must have "disabled" as its power-on default (value of parameter 0x0F000000 = 0 in non-volatile memory). See "ID-Chip Support / Stage Replacement" (p. 120) for more information.
- The settling behavior of the axis in closed-loop operation is influenced by the notch filter settings. Set the notch filter(s) **before** you optimize the servo-control parameters (p. 110).
- If you work with the *Piezo Dynamic Tuner* window of PIMikroMove:
  - If you change a parameter value of the E-709 by entering a corresponding value: The value is displayed in a blue font until you press Enter on your keyboard. Pressing Enter sends the value to the E-709 and changes the font color from blue to black. For fields highlighted by a red background, the parameter values in volatile and non-volatile memory of the E-709 differ.
  - When the *Notch Frequency 1* value is set in the *Parameter Settings* panel of the *Piezo Dynamic Tuner* window, the *Servo-Loop I-Term* value can be adjusted automatically in accordance. The adjustment depends on the selection in the *Automatic I-Term calculation* drop down menu.

    Default: The I-term is set to a “conservative” value which is calculated with the following formula:

$$I\,term_{conservative} = \frac{P\,term}{0.05 \times 4 \times \pi \times \text{Notch Frequency 1}}$$

Further options:

"Dynamic" I-term value, calculated with the following formula:

$$I\,term_{dynamic} = \frac{0.8 \times P\,term}{0.05 \times 4 \times \pi \times \text{Notch Frequency 1}}$$

"Off", i.e. no automatic I-term calculation.

- The settings for slew rate (*Slew Rate / Velocity* field) and record table rate (*Record Rate* field) can be changed in the *Piezo Dynamic Tuner* window. Entering new values in these fields changes the values of the corresponding parameters in volatile memory: Servo Loop Slew-Rate parameter (ID 0x07000200) or Open Loop Slew-Rate (ID 0x07000201), depending on the current operating mode (open-loop or closed-loop operation); Data Recorder Table Rate (ID 0x16000000). The values are **not** saved or reset when you use the *Save …* and *Reset …* buttons in the *Parameter Settings* panel of the *Piezo Dynamic Tuner* window.

13.3 Adjusting the Notch Filter(s) in Open-Loop Operation

The corrections by a notch filter take place in closed-loop operation and in open-loop operation. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanical system. Adjusting the notch filter frequency can be useful, particularly in the case of very high loads.

For further details, see "Notch Filters", p. 33.

INFORMATION

The notch rejection value, which scales the damping done by the notch filter, should always be 0.05. A notch rejection value of 1 deactivates the notch filter.

In addition to the measurement described below, you can create a Bode plot: In the PIMikroMove main window, open the *Data Recorder* window via the *E-709… ⇒ Show data recorder …* menu item. At the bottom of the *Data Recorder* window, enter the *Amplitude* value and click *Estimate* to start the frequency response.

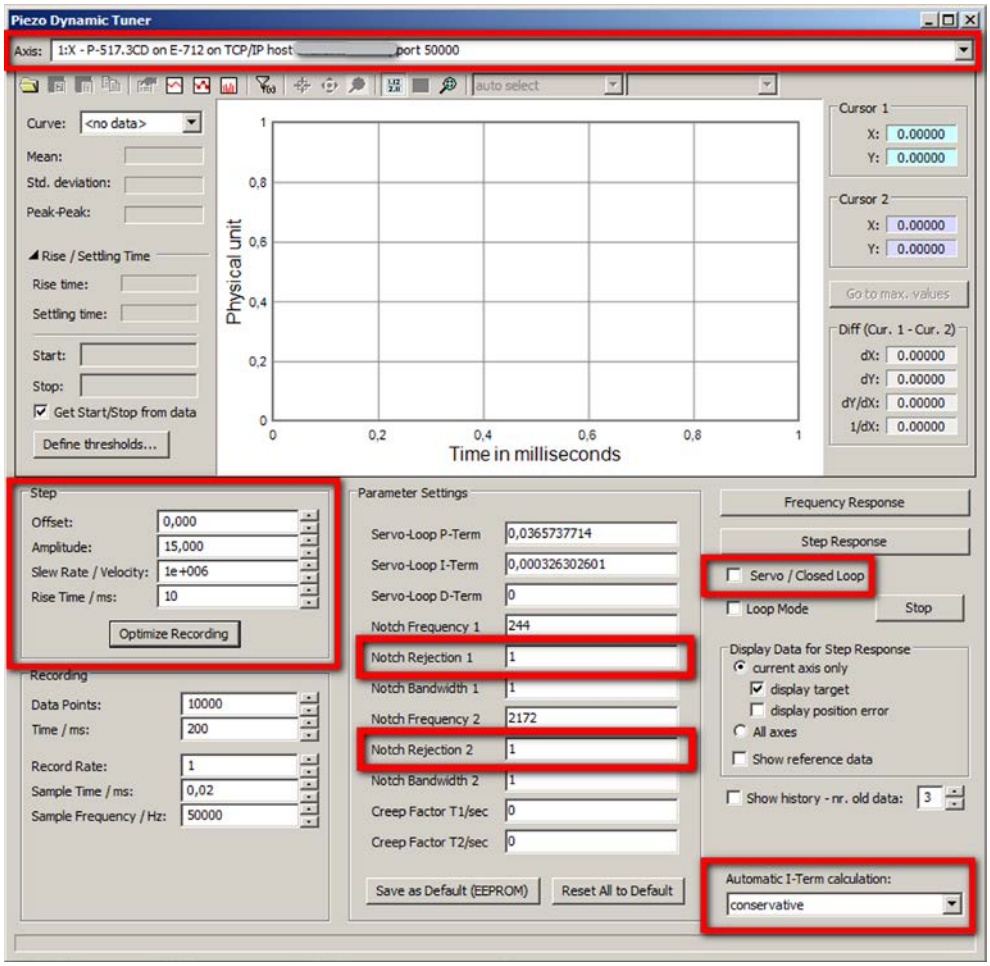
INFORMATION—Screenshots

The screenshots in the following instructions were created with an E-712 digital multi-axis controller. With E-709, some values may differ, and the E-709 does not support the *Creep Factor* parameters, but the procedure outlined in the screenshots is as with E-712.

To measure the resonant frequency and adjust the notch filter(s), a frequency response (axis response to an impulse) is recorded in open-loop operation.

Proceed as follows:

1. Read the "General Notes on Servo-Controller Dynamic Tuning" (p. 109).
2. Make sure the stage is mounted in exactly the same way as in the application. The load on the stage is especially important.
3. In the main window of PIMikroMove, open the *Piezo Dynamic Tuner* window via the *E-709… ⇒ Dynamic Tuner …* menu item.
4. Configure the frequency response in the *Piezo Dynamic Tuner* window:
   - 4.1 Make sure that the correct axis is selected (*Axis* drop down list).
   - 4.2 Enter suitable values for the start value (*Offset*:) and the amplitude (*Amplitude*:) of the impulse in the *Step* panel. The start value should be 0, and the amplitude should be about 10 % of the axis travel range.
   - 4.3 Make sure that the maximum permissible velocity of the axis is set high enough (*Slew Rate / Velocity:*).
   - 4.4 Make sure that the notch filters are deactivated, i.e., that *Notch Rejection 1* and *Notch Rejection 2* both have the value 1.
   - 4.5 Make sure that the axis is in open-loop operation *(Servo / Closed Loop* box is **not** checked).
   - 4.6 Select if and how the *Servo-Loop I-Term* is to be adjusted automatically when *Notch Frequency 1* is changed (*Automatic I-Term calculation*).

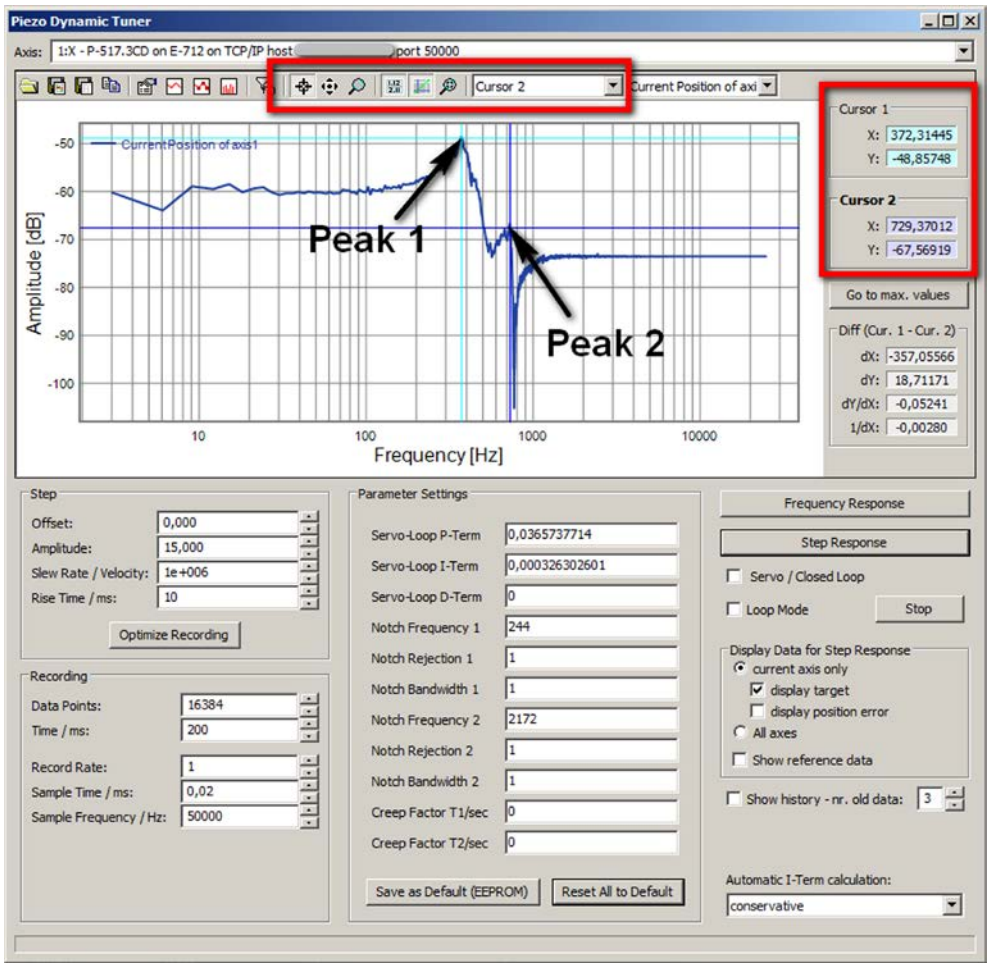


5 Perform the frequency response measurement by clicking the *Frequency Response* button in the *Piezo Dynamic Tuner* window.

6 Identify the resonant frequency in the *Piezo Dynamic Tuner* window:

- Identify the resonance peak(s) in the FFT display. To do so, place a cursor on the peak and read out the cursor value which is displayed on the right hand side of the graph. If there is more than one resonance peak, peak 1 is always the one with the lowest frequency.

- In the figure below, cursor 1 is at the first resonance peak (372.31445 Hz), and cursor 2 is at the second (next higher) resonance peak (729.37012 Hz).



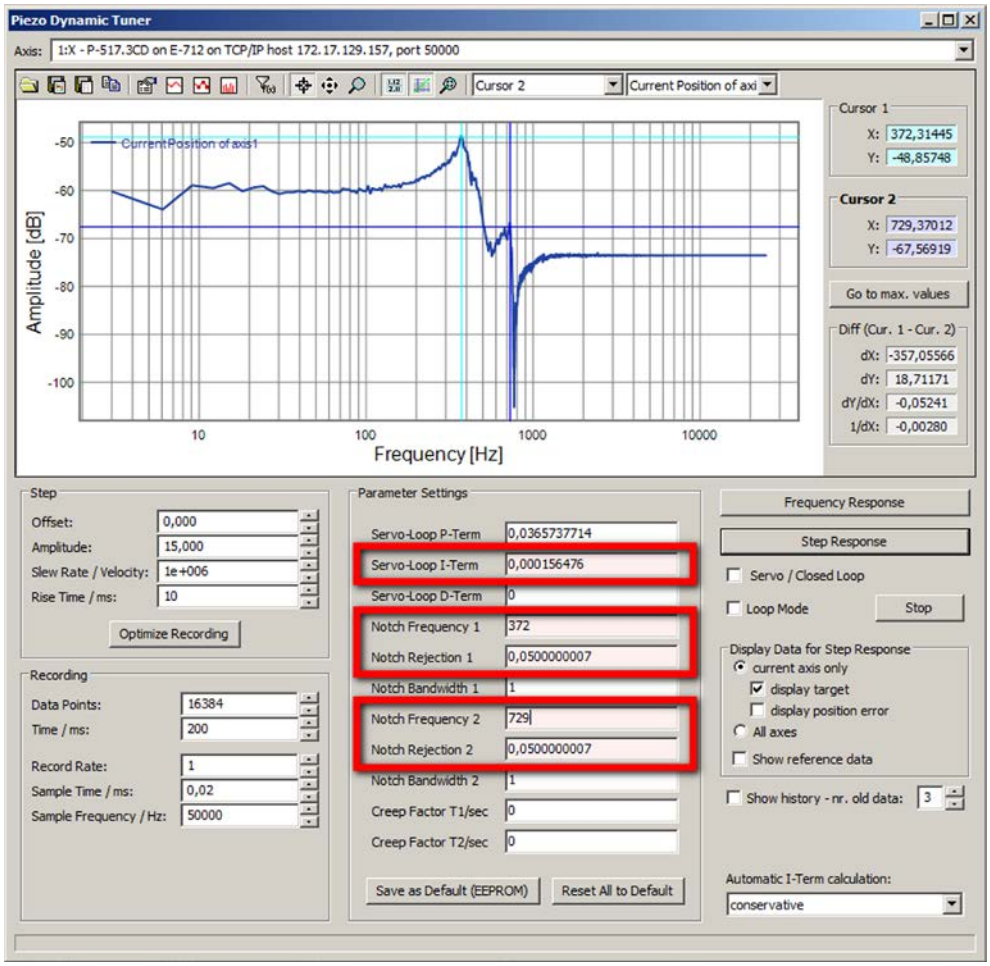
7 If necessary, adjust the notch filter settings in the *Parameter Settings* panel to the measured resonant frequencies (adjustment is necessary if the values significantly differ).

7.1 Enter the frequency value of the first resonance peak in the *Notch Frequency 1* field (in Hz). You can either right-click the field with the mouse and select the value from a menu, or type the value in the field.

NOte that depending on the selection for *Automatic I-Term calculation*, the *Servo-Loop I-Term* value is changed too automatically when you change the *Notch Frequency 1* value (for details, see p. 109).

7.2 To activate the first notch filter, enter the value 0.05 in the *Notch Rejection 1* field.

7.3 If you have measured a second resonance peak, enter the frequency value of the second resonance peak in the *Notch Frequency 2* field (in Hz), and change the rejection value to 0.05 in the *Notch Rejection 2* field.



8 Save or discard the new settings in the *Parameter Settings* panel:

- If you want to keep the new settings, save them to the non-volatile memory of the E-709 by clicking the *Save as Default (EEPROM)* button.
- If you want to discard the new settings and reset the parameter values to their defaults (i.e. to their values from non-volatile memory), click the *Reset All to Default* button.

13.4 Checking and Optimizing the Servo-Control Parameters

Adjusting the servo-control parameters (P-term, I-term) optimizes the dynamic properties of the system (overshoot and settling time). The optimum settings depends on your application and your requirements.

For further details regarding the control algorithm, see “PID Algorithm for Closed-Loop Operation”, p. 32.

The optimization of the servo-control parameters is typically done empirically: The response of the axis to a step (“step response”) is analyzed under various values in closed-loop operation.

INFORMATION—Screenshots

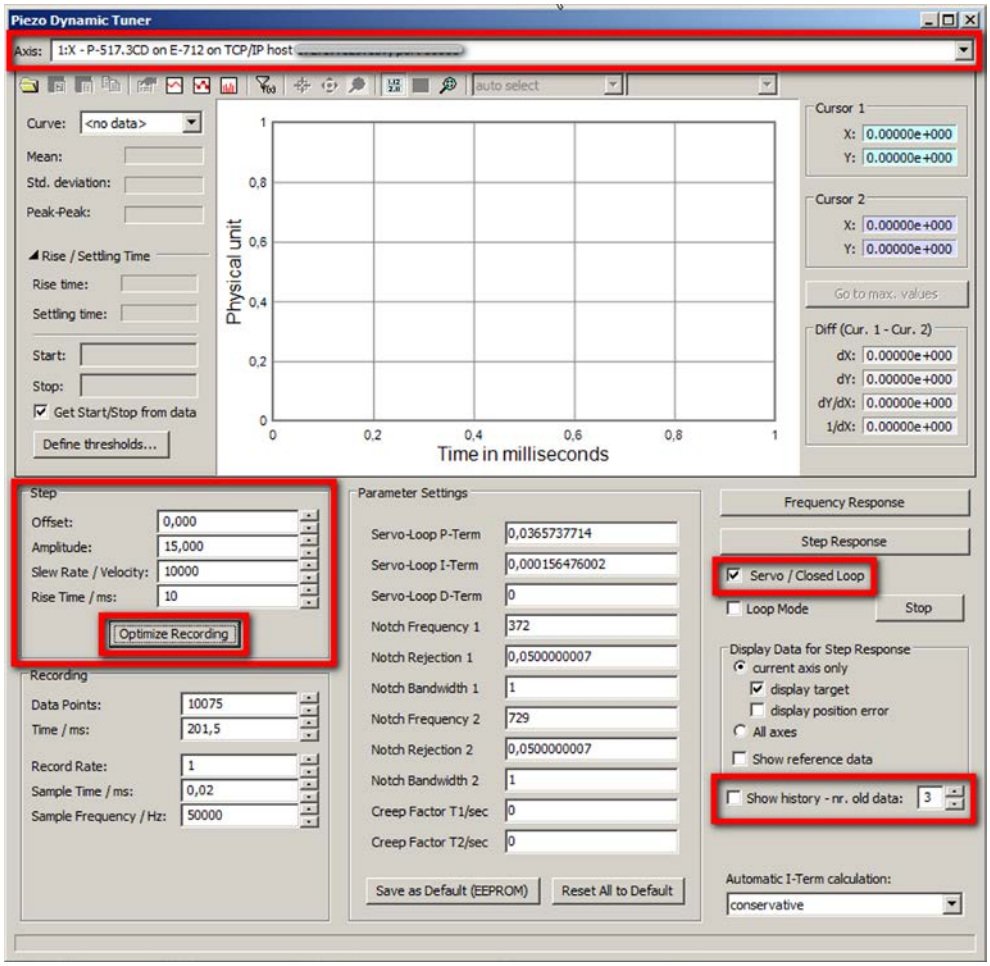
The screenshots in the following instructions were created with an E-712 digital multi-axis controller. With E-709, some values may differ, and the E-709 does not support the *Creep Factor* parameters, but the procedure outlined in the screenshots is as with E-712.

Proceed as follows for the axis:

1. Read the “General Notes on Servo-Controller Dynamic Tuning” (p. 109).
2. Make sure the stage is mounted in exactly the same way as in the application. The load on the stage is especially important.
3. In the main window of PIMikroMove, open the *Piezo Dynamic Tuner* window via the *E-709… ⇒ Dynamic Tuner …* menu item.
4. Make sure that the notch filter(s) are properly adjusted. For details, see “Adjusting the Notch Filter(s) in Open-Loop Operation” (p. 110).
5. Configure the step response in the *Piezo Dynamic Tuner* window:
   - 5.1 Make sure that the correct axis is selected (*Axis* drop down list).
   - 5.2 Make sure that the axis is in closed-loop operation (*Servo / Closed Loop* box is checked).
   - 5.3 Enter suitable values for the start value (*Offset*:) and the amplitude (*Amplitude*:) of the step in the *Step* panel. The start value should be 0, and the amplitude should be about 10 % of the axis travel range.

5.4 By clicking the *Optimize Recording* button in the *Step* panel, optimize the number of data recorder points that will be read from the controller when the step response has been performed.

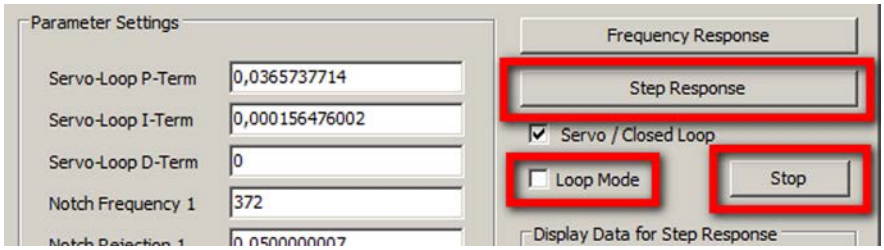
5.5 If you want to compare the results of multiple step resonse measurements, check the *Show history* box and select the number of old recordings to be displayed.



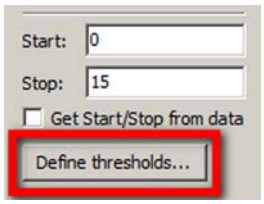
6 Perform and analyze the step response measurement in the *Piezo Dynamic Tuner* window:

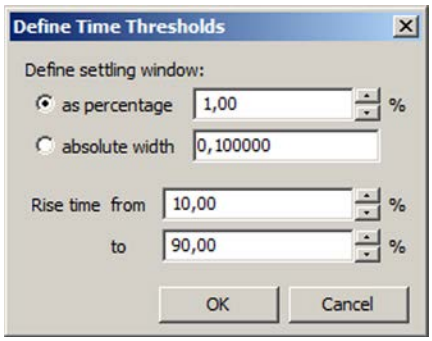
6.1 Optional: *Check the Loop Mode* box to move the axis in a permanent loop.
The loop mode is useful if you want to do the adjustment of the servo-control parameters during the motion. (The loop motion can be stopped at any time by clicking the *Stop* button.)

6.2 Start the step response by clicking the *Step Response* button.



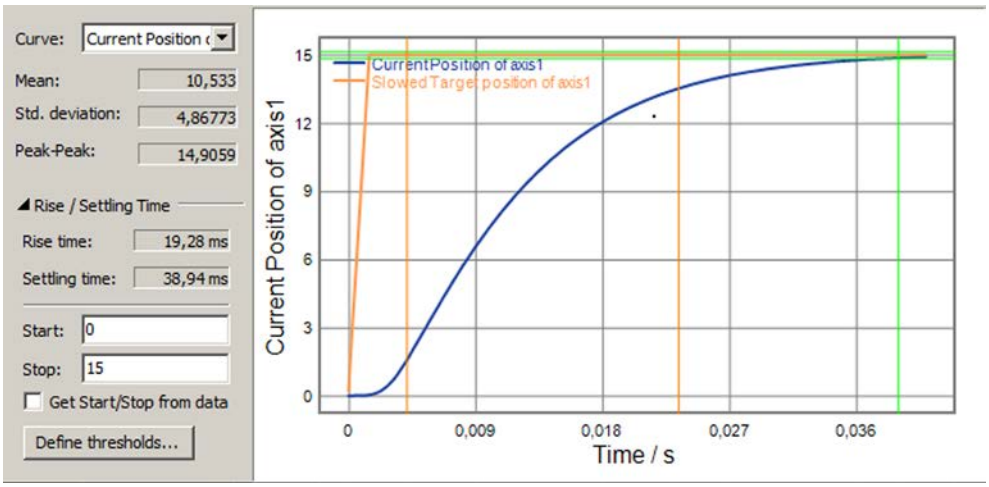
6.3 Optional: Click the *Define thresholds…* button to open the Define Time Thresholds window. In the *Define Time Thresholds* window, you can adjust the thresholds which are used by the *Piezo Dynamic Tuner* window to calculate and display the rise time and settling time of the axis, based on the recorded step response measurement.



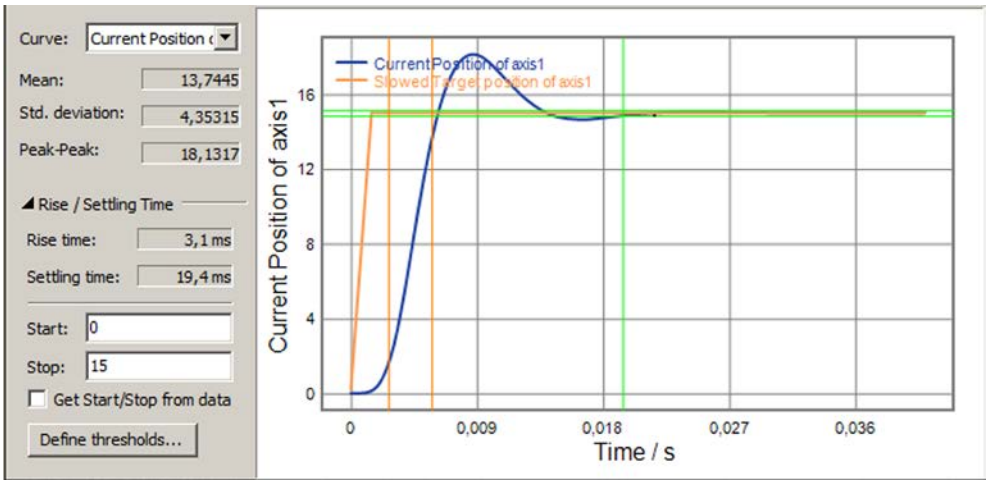


6.4 Check the step response result and compare it with the examples shown in the figures below. Tip: If the piezo stage starts oscillating (humming noise), reduce the P term and increase the I term.

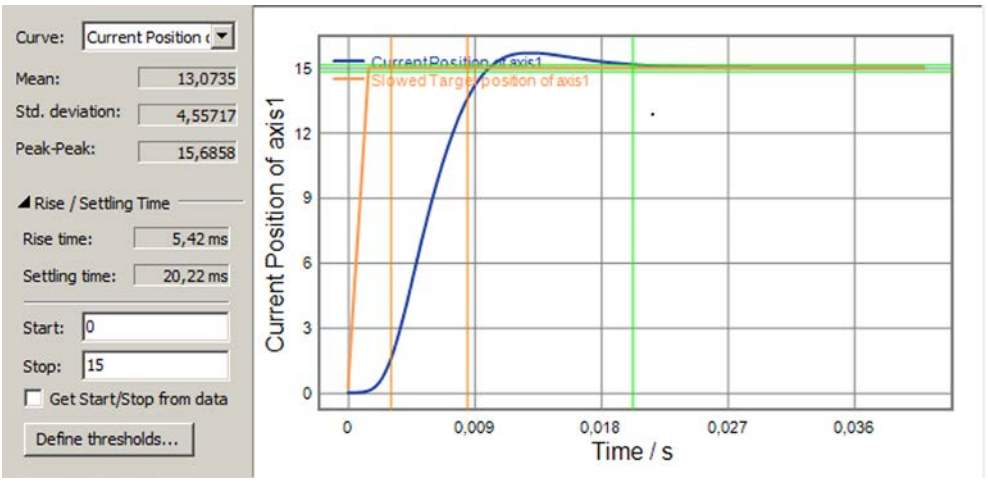
The rise rate of the step response is very low in the figure below. This means that the P term is too low and has to be increased.



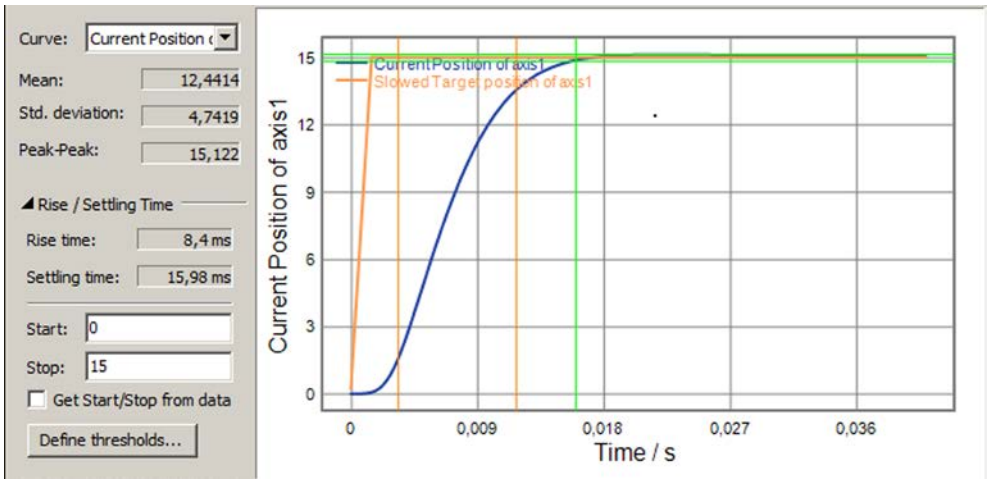
The figure below shows a step response with a high overshoot which means that the P term is too high and has to be decreased.



The figure below shows a step response with a small overshoot which means that the P term is still too high and has to be decreased.



The result of the step response is satisfactory when there is minimum overshoot, and the settling time is not too long, as in the figure below. No changes are required for the servo-control parameters.



7 Save or discard the new settings in the *Parameter Settings* panel:

- If you want to keep the new settings, save them to the non-volatile memory of the E-709 by clicking the *Save as Default (EEPROM)* button.
- If you want to discard the new settings and reset the parameter values to their defaults (i.e. to their values from non-volatile memory), click the *Reset All to Default* button.

14 E-709.Cxx Models Only: ID-Chip Support / Stage Replacement

The piezo stage which is connected to an E-709.Cxx model for capacitive sensors (.Cxx = .CRG or .CHG or .CR) may contain an ID-chip (located in the stage connector). The following data is stored in the ID-chip (and cannot be modified there by the customer):

- Stage type
- Serial number of the stage
- Calibration data
- Servo-control data (dynamic tuning, load dependent)

The parameters which are usually stored in ID-chips are marked in the table in "Parameter Overview" (p. 225), but the list can differ slightly among the different mechanics provided by PI.

When a stage with ID-chip is connected to the controller for the first time, the stage parameters from the ID-chip will be written to non-volatile and volatile memory upon controller power-on or reboot. Afterwards, the complete set of ID-chip parameters will be overwritten on power-on or reboot only if the "Power Up Read ID-Chip" option is enabled. By default, this option is disabled to facilitate maintaining optimized parameter settings on the controller.

INFORMATION

When you connect a stage when the controller is powered on, the ID-chip of the stage is not read by the controller. To read the ID-chip data, the controller must be power-cycled or rebooted using the RBT command or the corresponding host software functions.

A piezo stage can be easily exchanged due to the functionality of the ID-chip.

Consider the following when replacing stages with ID-chips.

"Simple" Replacement

Normally, when you replace a piezo stage with a new unit and you are using standard factory settings for all parameters, you do not have to adjust anything. The ID-chip holds all information needed. At power-on of the

system, the firmware reads the stage type and serial number stored in the ID-chip and compares this data to the data from the last connected stage, stored in the controller:

- If there is a new stage type connected to the controller, all the data in the ID-chip will be read and the corresponding parameters in the controller overwritten.

- If there is a stage of the same type but with a different serial number connected to the controller, the calibration data from the ID-chip will be read and only the corresponding parameters overwritten. The servo-control data will not be read, so those parameters will remain unchanged in the controller.

If you have optimized some parameters for your application, PI recommends that you repeat your optimization routine with any new stage, because there are variations, e.g. in the stiffness and natural frequency, of piezo stages.

Upgrade or Repair of Stages

If you send your stage to PI, e.g. for upgrade or repair, the calibration data stored in the ID-chip might be changed in the process. However, when you reconnect this stage to the controller to which it was connected before, the firmware will detect that the type and serial number are unchanged and will not read the new ID-chip data.

To force the controller to read the complete data of the ID-chip of the connected stage when the controller is switched on, you can enable the "Power Up Read ID-Chip" option (parameter ID 0X0F000000). This has to be done for the first input signal channel. Note that it might be necessary to switch to a higher command level to have write access to that parameter (use CCL or the appropriate facilities of PIMikroMove). Proceed as follows:

1. In PIMikroMove, open the *Device Parameter Configuration* window (*E-709* ⇒ *Parameter Configuration*) and select the *System Mechanics 1* group where you can enable the option. When this is done for input signal channel 1, press the "Write selected edit values to default settings" button in the icon bar of the *Device Parameter Configuration* window.

   Alternatively you can use the following command in a terminal to enable the option:
   SEP 100 1 0X0F000000 1 for input signal channel 1

2. Now reboot the controller by typing the RBT command in the terminal (alternatively you can power-cycle the controller). This time all data is read from the ID-chip and stored on the controller.

3 To ensure that at next power-on or reboot the controller will not read all data again and overwrite parameters you may have optimized, you will have to disable the "Power Up Read ID-Chip" option, again for input signal channel 1.

   In PIMikroMove, proceed as described above for enabling but make sure that the parameter now has the value "disabled".

   Alternatively you can use the following command in a terminal to disable the option:
   SEP 100 1 0X0F000000 0 for input signal channel 1

If you had optimized parameters before the repair/upgrade, PI recommends you to repeat your optimization routine when the stage is returned.

15 GCS Commands

The PI General Command Set (GCS) is supported by a wide range of PI systems. This command set is well-suited for positioning tasks with one or more axes. The command set itself is independent of the specific hardware (controller or attached stages).
Commands are used to set the control mode, initiate axis motion and to query system and motion values. Because of the variety of functions and parameters, a sequence of commands must often be transferred in order to achieve a desired system action.
You can type commands, for example, in the *Command Entry* window of PIMikroMove, or in the PITerminal.

15.1 Format

15.1.1 Notation

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

<...>	Angle brackets indicate an argument of a command, can be an item identifier (p. 63) or a command-specific parameter
[…]	Square brackets indicate an optional entry
{…}	Braces indicate a repetition of entries, i.e. that it is possible to access more than one item (e.g. several axes) in one command line.
LF	LineFeed (ASCII char #10), is the default termination character
SP	Space (ASCII char #32)
“…”	Quotation marks indicate that the characters enclosed are returned or to be entered.

15.1.2 GCS Syntax

Except as listed below, a GCS command consists of 3 characters, e.g. CMD. The corresponding query command has a "?" appended, e.g. CMD?.
Command mnemonic:

CMD ::= character1 character2 character3 [?]

Exceptions:

- Special commands, e.g. fast polling commands, consist only of one character. The 24th ASCII character e.g. is called #24. Note that these commands are not followed by a termination character (but the responses to them are).

- *IDN? (for GPIB compatibility).

The command mnemonic is not case-sensitive.

General:

CMD[{{SP}<argument>}]LF

That means the command mnemonic and all arguments (e.g. axis IDs, channel IDs, parameters, etc.) must be separated from each other by one space.

Example:
Send: MOVSP1SP10.0LF
to move Axis 1 to position 10.0 (the unit depends on the controller, can be µm or mm, for example)

More than one command mnemonic per line is not allowed. Several groups of arguments following a command mnemonic are allowed, e.g.
MOVSP1SP17.3SP2SP2.05LF
if there were 2 axes. The command line ends with the termination character (LF).

If 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. For example,
RPALF
resets all parameters in volatile memory.

The <AxisID> argument is used for the logical axes of the controller. Depending on the controller, an axis could be identified with up to 16 characters—all alphanumeric characters and the underscore are allowed. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers supported by the E-709.

Definitions for query commands (report commands):

CMD?[{{SP}<argument>}]LF

When all arguments are optional and are omitted, all possible values are reported. For example,
POS?
queries the position of all axes.

Reply syntax:

[<argument>[{SP<argument>}]"="]<value>LF

Multi-line reply syntax:
{[<argument>[{SP<argument>}]"="]<value>SP LF}
[<argument>[{SP<argument>}]"="]<value>LF for the last line!

The command
CMD?SP<arg3>SP <arg1>SP <arg2>LF
replies in the same order:
<arg3>"="<value3>SP LF
<arg1>"="<value1>SP LF
<arg2>"="<value2> LF

Example:
Send: TSP?SP2SP1
Report: 2=-1158.4405SP LF
1=+0000.0000LF

INFORMATION

With the E-709, you can address only one single item (e.g. axis or channel) per command line, or, if the command supports this, address all items by omitting the item identifier.

Example:
You can send
SEP 100 1 0x06000500 2
to select analog command mode as power-on default for axis 1.
But it is not possible to set analog command mode and analog offset value in the same command. Hence sending
SEP 100 1 0x06000500 2 1 0x06000501 20
will provoke an error.

15.2 Command Survey

Command	Format	Short Description	Details see
#5	#5	Request Motion Status	p. 129
#7	#7	Request Controller Ready Status	p. 130
#9	#9	Get Wave Generator Status	p. 130
#24	#24	Stop All Axes	p. 131
*IDN?	*IDN?	Get Device Identification	p. 131
AOS	AOS {<AxisID> <Offset>}	Set Analog Input Offset	p. 132
AOS?	AOS? [{<AxisID>}]	Get Analog Input Offset	p. 134
ATZ	ATZ [{<AxisID> <LowValue>}]	Set Automatic Zero Point Calibration	p. 135
ATZ?	ATZ? [{<AxisID>}]	Get State Of Automatic Zero Point Calibration	p. 138
CCL	CCL <Level> [<PSWD>]	Set Command Level	p. 138
CCL?	CCL?	Get Command Level	p. 139
CST?	CST? [{<AxisID>}]	Get Assignment Of Stages To Axes	p. 139
CSV?	CSV?	Get Current Syntax Version	p. 140
CTI	CTI {<TrigInID> <CTIPam> <Value>}	Set Configuration of Trigger Input	p. 140
CTI?	CTI? [{<TrigInID> <CTIPam>}]	Get Configuration of Trigger Input	p. 142
CTO	CTO {<TrigOutID> <CTOPam> <Value>}	Set Configuration Of Trigger Output	p. 142
CTO?	CTO? [{<TrigOutID> <CTOPam>}]	Get Configuration Of Trigger Output	p. 145
DIO?	DIO? [{<DIOID>}]	Get Digital Input Lines	p. 146
DRC	DRC {<RecTableID> <Source> <RecOption>}	Set Data Recorder Configuration	p. 146
DRC?	DRC? [{<RecTableID>}]	Get Data Recorder Configuration	p. 148
DRL?	DRL? [{<RecTableID>}]	Get Number Of Recorded Points	p. 148
DRR?	DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]	Get Recorded Data Values	p. 149
DRT	DRT {<RecTableID> <TriggerSource> <Value>}	Set Data Recorder Trigger Source	p. 150
DRT?	DRT? [{<RecTableID>}]	Get Data Recorder Trigger Source	p. 151
ERR?	ERR?	Get Error Number	p. 151
GWD?	GWD? [<StartPoint> <NumberOfPoints> [{<WaveTableID>}]]	Get Wave Table Data	p. 152
HDR?	HDR?	Get All Data Recorder Options	p. 153

Command	Format	Short Description	Details see
HLP?	HLP?	Get List Of Available Commands	p. 154
HPA?	HPA?	Get List Of Available Parameters	p. 154
HPV?	HPV?	Get Parameter Value Description	p. 155
IDN?	IDN?	Get Device Identification	p. 156
IFC	IFC {<InterfacePam> <PamValue>}	Set Interface Parameters Temporarily	p. 157
IFC?	IFC? [{<InterfacePam>}]	Get Current Interface Parameters	p. 158
IFS	IFS <Pswd> {<InterfacePam> <PamValue>}	Set Interface Parameters As Default Values	p. 158
IFS?	IFS? [{<InterfacePam>}]	Get Interface Parameters As Default Values	p. 159
IMP	IMP <AxisID> <Amplitude>	Start Impulse And Response Measurement	p. 160
IMP?	IMP? [{<AxisID>}]	Get IMP Settings	p. 160
MOV	MOV {<AxisID> <Position>}	Set Target Position	p. 161
MOV?	MOV? [{<AxisID>}]	Get Target Position	p. 162
MVR	MVR {<AxisID> <Distance>}	Set Target Relative To Current Position	p. 162
ONT?	ONT? [{<AxisID>}]	Get On-Target State	p. 163
OVF?	OVF? [{<AxisID>}]	Get Overflow State	p. 164
POS?	POS? [{<AxisID>}]	Get Real Position	p. 164
RBT	RBT	Reboot System	p. 165
RPA	RPA [{<ItemID> <PamID>}]	Reset Volatile Memory Parameters	p. 165
RTR	RTR <RecordTableRate>	Set Record Table Rate	p. 166
RTR?	RTR?	Get Record Table Rate	p. 167
SAI	SAI {<AxisID> <NewIdentifier>}	Set Current Axis Identifiers	p. 167
SAI?	SAI? [ALL]	Get List Of Current Axis Identifiers	p. 168
SEP	SEP <Pswd> {<ItemID> <PamID> <PamValue>}	Set Nonvolatile Memory Parameters	p. 168
SEP?	SEP? [{<ItemID> <PamID>}]	Get Nonvolatile Memory Parameters	p. 169
SPA	SPA {<ItemID> <PamID> <PamValue>}	Set Volatile Memory Parameters	p. 170
SPA?	SPA? [{<ItemID> <PamID>}]	Get Volatile Memory Parameters	p. 173
SSN?	SSN?	Get Device Serial Number	p. 174
STE	STE <AxisID> <Amplitude>	Start Step And Response Measurement	p. 175
STE?	STE? [{<AxisID>}]	Get STE Settings	p. 175

Command	Format	Short Description	Details see
STP	STP	Stop All Axes	p. 176
SVA	SVA {<AxisID> <Amplitude>}	Set Open-Loop Axis Value	p. 176
SVA?	SVA? [{<AxisID>}]	Get Open-Loop Axis Value	p. 178
SVO	SVO {<AxisID> <ServoState>}	Set Servo Mode	p. 178
SVO?	SVO? [{<AxisID>}]	Get Servo Mode	p. 179
SVR	SVR {<AxisID> <Difference>}	Set Relative Open-Loop Axis Value	p. 180
TAD?	TAD? [{<InputSignalID>}]	Get ADC Value Of Input Signal	p. 180
TIO?	TIO?	Tell Digital I/O Lines	p. 181
TMN?	TMN? [{<AxisID>}]	Get Minimum Commandable Position	p. 182
TMX?	TMX? [{<AxisID>}]	Get Maximum Commandable Position	p. 182
TNR?	TNR?	Get Number Of Record Tables	p. 182
TNS?	TNS? [{<InputSignalID>}]	Get Normalized Input Signal Value	p. 183
TPC?	TPC?	Get Number of Output Signal Channels	p. 183
TRI	TRI {<TrigInID> <TrigInMode>}	Set Trigger Input State	p. 184
TRI?	TRI? [{<TrigInID>}]	Get Trigger Input State	p. 184
TSC?	TSC?	Get Number of Input Signal Channels	p. 185
TSP?	TSP? [{<InputSignalID>}]	Get Input Signal Value	p. 185
TWC	TWC	Clear All Wave Related Triggers	p. 186
TWG?	TWG?	Get Number of Wave Generators	p. 186
TWS	TWS {<TrigOutID> <PointNumber> <Switch>}	Set Trigger Line Action To Waveform Point	p. 187
VEL	VEL {<AxisID> <Velocity>}	Set Closed-Loop Velocity	p. 188
VEL?	VEL? [{<AxisID>}]	Get Closed-Loop Velocity	p. 188
VOL?	VOL? [{<OutputSignalID>}]	Get Voltage Of Output Signal Channel	p. 189
WAV	WAV <WaveTableID> <AppendWave> <WaveType> <WaveTypeParameters>	Set Waveform Definition	p. 189
WAV?	WAV? [{<WaveTableID> <WaveParameterID>}]	Get Waveform Definition	p. 194
WCL	WCL {<WaveTableID>}	Clear Wave Table Data	p. 195
WGC	WGC {<WaveGenID> <Cycles>}	Set Number Of Wave Generator Cycles	p. 195
WGC?	WGC? [{<WaveGenID>}]	Get Number Of Wave Generator Cycles	p. 195
WGI?	WGI? [{<WaveGenID>}]	Get Index of Wave Table Point	p. 196

Command	Format	Short Description	Details see
WGN?	WGN? [{<WaveGenID>}]	Get Number of Completed Output Cycles	p. 196
WGO	WGO {<WaveGenID> <StartMode>}	Set Wave Generator Start/Stop Mode	p. 197
WGO?	WGO? [{<WaveGenID>}]	Get Wave Generator Start/Stop Mode	p. 200
WGR	WGR	Starts Recording In Sync With Wave Generator	p. 200
WOS	WOS {<WaveGenID> <Offset>}	Set Wave Generator Output Offset	p. 201
WOS?	WOS? [{<WaveGenID>}]	Get Wave Generator Output Offset	p. 202
WPA	WPA <Pswd> [{<ItemID> <PamID>}]	Save Parameters To Nonvolatile Memory	p. 202
WSL	WSL {<WaveGenID> <WaveTableID>}	Set Connection Of Wave Table To Wave Generator	p. 204
WSL?	WSL? [{<WaveGenID>}]	Get Connection Of Wave Table To Wave Generator	p. 204
WTR	WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}	Set Wave Generator Table Rate	p. 205
WTR?	WTR? [{<WaveGenID>}]	Get Wave Generator Table Rate	p. 206

15.3 Command Reference (alphabetical)

#5 (Request Motion Status)

Description: Requests motion status of the axes.

Format: #5 (single ASCII character number 5)

Arguments: none

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

1=first axis is moving
2=second axis is moving
4=third axis is moving
...

Examples: 0 indicates motion of all axes complete
3 indicates that the first and the second axis are moving

Notes: #5 is only effective in closed-loop operation (servo ON).

During an AutoZero procedure (see ATZ command

(p. 135)) and when the wave generator is running, the motion status can be queried with #5 irrespective of the current control mode (open-loop or closed-loop operation).

#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. 129) instead of #7 to verify if motion has finished.

Format: #7 (single ASCII character number 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 referencing command)

Troubleshooting The response characters may appear differently in non-Western character sets or other operating systems. They may be indistinguishable on the controller screen.

#9 (Get Wave Generator Status)

Description: Requests the status of the wave generator(s).

The #9 single-character command (p. 130) can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode. With WGO? you can ask for the last-commanded wave generator start options (WGO settings).

Format: #9 (single ASCII character number 9)

Arguments: none

Response: The answer <uint> is bit-mapped and returned as the hexadecimal sum of the following codes:
1 = Wave Generator 1 is running,
2 = Wave Generator 2 is running,
4 = Wave Generator 3 is running, etc.

Examples: 0 indicates that no wave generator is running
5 indicates that wave generators 1 and 3 are running

#24 (Stop All Axes)

Description: Stops all axes abruptly. For details see the notes below.

Sets error code to 10.

This command is identical in function to STP (p. 176), but only one character must be send via the interface. Therefore #24 can also be used while the controller is performing time-consuming tasks.

Format: #24 (ASCII character 24)

Arguments: none

Response: none

Notes: #24 stops motion of all axes caused by move commands (MOV (p. 161), MVR (p. 162), SVA (p. 176), SVR (p. 180)), by the wave generator (WGO (p. 200)), and by the autozero procedure (ATZ (p. 135)). If analog command mode is enabled, #24 (p. 131) enables the digital command mode. To recommence commanding the axis via the analog input, you have to re-enable analog command mode for the axis. See "How to work with the Analog Input" (p. 67) for more information.

After the axes are stopped, if servo is on their target positions are set to their current positions, or if servo is off, their open-loop control values are set to their last valid control values.

*IDN? (Get Device Identification)

Description: Reports the device identity number.

Format: *IDN?

Arguments: none

Response: One-line string terminated by line feed with controller name, serial number and firmware version

Notes: For E-709, *IDN? replies something like:

PHYSIK INSTRUMENTE,E-709,0109025544,1.000

*IDN? is identical in function with the IDN? command (p. 156).

AOS (Set Analog Input Offset)

Description:	Set an offset to be added to the analog input scaled value for the given axis (corresponding parameter is Analog Target Offset, ID 0x06000501). The AOS command changes the offset setting in volatile memory (RAM) only. On controller power-on or reboot, the offset value is loaded from the controllers non-volatile memory, and any changes made with AOS will be lost unless they have been saved. To save the currently valid AOS setting to non-volatile memory, where it becomes the power-on default, use WPA (p. 202).
Format:	AOS {<AxisID> <Offset>}
Arguments	<AxisID> is one axis of the controller <Offset> is the offset value, any floating point number. For details see below.
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	In closed-loop operation (servo ON), <Offset> is interpreted as axis position value. In open-loop operation (servo OFF), <Offset> is interpreted as piezo output voltage value. This offset is only effective when analog command mode is selected for the axis. This is done via the Current Command Mode parameter (ID 0x06000500) using SPA (p. 170) or SEP (p. 168) (0 = digital command mode; 2= analog command mode).

The control value for an axis which is connected to an input signal channel consists of:
Control Value = Analog Input Scaled Value of the Input Signal Channel + Offset
The resulting piezo output voltage is limited according to the valid limit values:
Closed-loop operation: travel range limit parameters 0x70000000 and 0x70000001
Open-loop operation: voltage limit parameters 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001

See also "Axis Motion" (p. 26) and "How to work with the Analog Input" (p. 67).

Example:

The E-709 is in closed-loop operation (servo on) in this example, i.e. the control sources write to the target register whose current value can be read with the MOV? command. In open-loop operation, you would use SVA? instead to ask for the current value of the open-loop control register.

Send:	SPA 1 0x06000500 2
Note:	Select analog command mode for axis 1. Now the control value of axis 1 will result from the scaled value of the analog input line (p. 67) plus the offset.
Send	AOS 1 0.0
Note:	Set offset of axis 1 zero.
Send	TSP? 2
Receive	2=3.22
Note:	Request the filtered and scaled value of the analog input line (input signal channel 2). The current value is 3.22. This value plus the offset is the current target value of axis 1.
Send	MOV? 1
Receive	1=3.22
Note:	Request the current target position of axis 1. The target position and the scaled value of input signal channel 2 are the same because the offset is zero.
Send	AOS 1 1.50
Note:	Set offset of axis 1 to 1.5.
Send	TSP? 2
Receive	2=3.22

Send	MOV? 1
Receive	1=4.72
Note:	Now the target value of axis 1 is the scaled value of the analog input (input signal channel 2) plus the offset of axis 1.
Send	MOV 1 6.0
Send	ERR?
Receive	72
Note:	As long as analog command mode is selected for axis 1, it is not possible to set the target using the MOV command.
Send:	SPA 1 0x06000500 0
Note:	Switch to digital command mode for axis 1. Now its target position can be set by the MOV command. The AOS setting is no longer effective for the control value generation of axis 1.

AOS? (Get Analog Input Offset)

Description:	Get currently valid offset to the analog input scaled value for the given axis (Analog Target Offset parameter value in volatile memory (ID 0x06000501)). Get all axes when <AxisID>=""
Format:	AOS? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>"="<Offset> LF} where <Offset> is the offset value, see AOS (p. 132) for details
Troubleshooting:	Illegal axis identifier

ATZ (Set Automatic Zero Point Calibration)

Description: Automatic zero-point calibration. Sets the output voltage which is to be applied at the zero position of the axis and starts an appropriate calibration procedure.

This command can be interrupted by #24 (p. 131) or STP (p. 176).

ATZ works in open-loop operation (servo off). If the servo is on, it will be switched off automatically at the start of the ATZ procedure and switched on again when it is finished.

The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished.

ATZ is not effective on non-linear axes (rotation axes).

The success of the automatic zero-point calibration can be queried with the ATZ? command (p. 138). Note that starting the AutoZero procedure for an individual axis may influence the AutoZero results of other axes so that their success states are reset. For that reason it is recommended to start the AutoZero procedure for all axes at the same time (see below).

The automatic zero-point calibration can take several seconds. During this time, the controller is busy and only very limited able to execute or answer commands. Ask with #5 (p. 129) if the procedure is finished.

Format: ATZ [{<AxisID> <LowVoltage>}]

Arguments	<AxisID> is one axis of the controller <LowVoltage> gives the voltage value to be applied at the zero position of the axis; in volts; float. Can also be NaN ("not a number")—in this case the value of the Autozero Low Voltage parameter saved in the controller (ID 0x07000A00) will be used. If all arguments are omitted, ATZ will be carried out synchronously for all linear axes using their AutoZero Low Voltage parameter values. This is the recommended usage.
Response:	none
Troubleshooting:	ATZ will be not successful when an invalid axis identifier is used, e.g. ATZ 9 NAN or when NaN was omitted and no voltage value was given
Notes:	NOTICE: The ATZ procedure will move the axis, and the motion may cover the whole travel range. Make sure that it is safe for the stage to move. Procedure details: To match voltage and position as required, the axis is moved—the motion range is specified by the <LowVoltage> value given in the ATZ command (lower limit) and by the Autozero High Voltage parameter value saved in the controller (parameter ID 0x07000A01; upper limit). The final position is the zero position, with the given <LowVoltage> value applied.

There is no range check for the given <LowVoltage> value. Make sure that this value does not exceed the voltage limits of the amplifier(s) (Min Output Voltage of Amplifier, parameter ID 0x0B000007 and Max Output Voltage of Amplifier, parameter ID 0x0B000008). Otherwise the <LowVoltage> value will be set to the corresponding limit.

If NaN is entered for the <LowVoltage> value, the AutoZero Low Voltage parameter value saved in the controller will be used (parameter ID 0x07000A00). You can modify this parameter with SPA (p. 170) or SEP (p. 168).

The AutoZero procedure changes the offset of the polynomials used for mechanics linearization (Sensor Mech. Correction 1 parameter, ID 0x02000200).

To save the current valid values of the above-mentioned parameters to non-volatile memory, where they become the power-on defaults, use WPA (p. 202). To have write access to the parameters, it might be necessary to switch to a higher command level using CCL (p. 138).

See also "AutoZero Procedure" (p. 61).

Example 1:

Send: SEP? 1 0x07000A00
Receive: 1 0x7000a00=0.000000e+00
Note: The value of the AutoZero Low Voltage parameter saved in the controller is 0 V.
Send: ATZ 1 NaN
Note: Starts autozero for axis 1 with the value of the AutoZero Low Voltage parameter. Do not omit "NaN"!
Send: ATZ? 1
Receive: 1
Note: Autozero for axis 1 was successful

Example 2:

Send: ATZ 1 15.0
Note: Starts autozero for axis 1 with a voltage value of 15 V
Send: ATZ? 1
Receive: 0
Note: Autozero for axis 1 was not successful

ATZ? (Get Automatic Zero Point Calibration)

Description: Query success or failure of the automatic zero-point calibration (see ATZ (p. 135) for details).

Format: ATZ? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<uint> indicates whether the automatic zero-point calibration of the given axis was successful (= 1) or not (= 0).

Troubleshooting: Illegal axis identifier

CCL (Set Command Level)

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

Format: CCL <Level> [<PSWD>]

Arguments: <Level> is one 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 your Physik Instrumente Sales Engineer or write info@pi.ws if there seem to be problems with level 2 or higher parameters.

Response: none

Troubleshooting: Invalid password

Notes: HLP? (p. 154) lists all commands available in the current command level.

HPA? (p. 154) lists the parameters including the information about which command level allows write access to them. For more information about parameter handling see "Controller Parameters" (p. 223).

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

CCL? (Get Command Level)

Description: Get the active "command level".

Format: CCL?

Arguments: none

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

Notes: <Level> should be 0 or 1.

<Level> = 0 is the default setting, all commands provided for "normal" users are available, as is read access to all parameters

<Level> = 1 provides additional commands and write access to level-1 parameters (commands and parameters from Level 0 are included)

CST? (Get Assignment of Stages to Axes)

Description: Returns the name of the connected stage 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 stage assigned to the axis.

Notes: The stage name is read from the Stage Type parameter (ID 0x0F000100). Normally, the value of this parameter is written during the calibration at the factory.

You can change the parameter value using SPA (p. 170) or SEP (p. 168).

CSV? (Get Current Syntax Version)

Description: Get current GCS syntax version used in the firmware.

Format: CSV?

Arguments: none

Response: The current GCS syntax version, can be 1.0 (for GCS 1.0) or 2.0 (for GCS 2.0)

CTI (Set Configuration of Trigger Input)

Description: Configures the trigger input for the given digital input line.

Format: CTI {<TrigInID> <CTIPam> <Value>}

Arguments: <TrigInID> is one digital input line of the controller; for further information, see below.

<CTIPam> is the ID of the CTI parameter in decimal format, see below for available IDs.

<Value> is the value to which the CTI parameter is set, see below.

Response: None

Available input lines and configuration options:

<TrigInID>: 1 identifies the Digital_IN_1 line (pin 10 of the "I/O" socket (p. 260 or p. 261)).

<CTIPam> parameter IDs available for E-709:
1 = TriggerType
7 = Polarity

<Value> available for the appropriate <CTIPam> ID:

for TriggerType:
0 = Edge triggered; triggering upon state transition of the digital input line. The activating state transition can be low --> high or high --> low (depends on the signal polarity set (CTIPam 7)).
1 = Level triggered (default); triggering when the digital input line is in an active state (high or

low; depends on the signal polarity set (CTIPam 7)).

for Polarity: Sets the signal polarity for the digital input line:
0 = active low
1 = active high (default)

Notes:

The trigger functionality must be enabled with TRI (p. 184).

The item/event to be triggered by the digital input line is determined by the value of the Digital Trigger Input Usage parameter (ID 0x15000800). Possible values (1 is default):

0 = The digital input is not used for triggering.
1 = The digital input line starts/interrupts the wave generator output. The start mode "Start via external trigger signal" (bit 1) must be set for the wave generator with WGO (p. 197). The wave generator output depends on the selected trigger type (CTIPam 1):
Edge triggered: Each activating state transition of the digital input line triggers the output of a point in the wave table. When an output rate > 1 is set with WTR (p. 205), the corresponding number of activating state transitions is required to output a point.
Level triggered: When the digital input line is in an active state, the wave generator outputs the points of the wave table. When the digital input line is in a non-active state, the wave generator output is interrupted.
Regardless of the selected trigger type, the number of output cycles of the waveform can be limited with WGC (p. 195).
>1 = Reserved for special applications

The CTI and TRI settings are lost when the E-709 is switched off or rebooted.

CTI? (Get Configuration of Trigger Input)

Description: Gets the values set for specified trigger input lines and parameters.

Format: CTI? [{<TrigInID> <CTIPam>}]

Arguments: <TrigInID> is one digital input line of the controller, see CTI.

<CTIPam>: parameter ID; see CTI.

If all arguments are omitted, the response contains the values for all parameters and all input lines.

Response: {<TrigInID> <CTIPam>"="<Value> LF}

For <Value> see CTI.

Notes: If <TrigInID> and <CTIPam> are not omitted, only one parameter for one trigger line can be queried per CTI? command.

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

Note: The trigger output conditions will become active immediately.

Available output lines and trigger conditions:

<TrigOutID>: 1 identifies the Digital_OUT_1 line (pin 11) and 2 the Digital_OUT_2 line (pin 12) of the "I/O" socket (p. 260 or p. 261)

<CTOPam> parameter IDs available for E-709:
1 = TriggerStep
2 = Axis
3 = TriggerMode
5 = MinThreshold
6 = MaxThreshold
7 = Polarity
8 = StartThreshold
9 = StopThreshold

<Value> available for the appropriate <CTOPam> ID:

for TriggerStep: step size in physical units (default value is 0.1)

for Axis: the axis to connect to the trigger output line (axis 1).

for TriggerMode:
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: In case 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 (default for Digital_OUT_2); the on-target status of the selected axis is written to the selected trigger output line (this status can also be read with the ONT? command)
3 = MinMaxThreshold; values for MinThreshold and MaxThreshold (<CTOPam> IDs 5 and 6) must be defined. When the axis position of the selected axis is inside the band specified by the MinThreshold and MaxThreshold values, the selected trigger output line is set high, otherwise it is set low.

4 = Generator Trigger; the trigger line action must be defined with TWS (p. 187)
6 = InMotion (default for Digital_OUT_1); the selected trigger line is active as long as the selected axis is in motion (the in-motion state can also be read with #5).

for MinThreshold/MaxThreshold: position value in physical units; used for the MinMaxThreshold TriggerMode; both values must be set to form a band (default values are 0)

for Polarity: Sets the signal polarity for the digital output line
0 = Active Low
1 = Active High

for StartThreshold/StopThreshold: Position value; can be used for the PositionDistance trigger mode; both thresholds must be set to determine the position range and the direction of motion for the trigger output

For application examples see "Configuring Trigger Output" (p. 85) and the lines below.

Example 1: A pulse on Digital_OUT_2 is to be generated whenever the stage (i.e. axis 1) has covered a distance of 0.05 µm. The following parameters must be set:

TrigOutID = 2
TriggerMode = 0
TriggerStep = 0.05
Send: CTO 2 3 0
Send: CTO 2 1 0.05

Example 2: On Digital_OUT_2, pulses are to be generated at certain waveform points during the wave generator output, i.e. the trigger outputs are to be controlled by the wave generator. To do this, the trigger line must be programmed using the TWS and TWC commands, and the corresponding trigger mode is to be set by CTO.

Send: TWC
Note: Clears all trigger settings for the wave generator by switching the signal state for

all points to "low". It is recommended to do this before new trigger actions are defined.

Send: TWS 2 1 1

Send: TWS 2 2 0

Send: TWS 2 3 0

Note: Set trigger action for Digital_OUT_2, at waveform point 1 it is set high, points 2 and 3 are set low. Note that the trigger line actions defined with TWS are valid for Digital_OUT_1 and Digital_OUT_2 since the lines share a common definition table.

Send: CTO 2 3 4

Note: The TriggerMode for output line Digital_OUT_2 is set to "Generator Trigger". Now the wave generator can be started with WGO, and the trigger action will take place as specified. See also "Trigger Output Synchronized With Wave Generator" (p. 103).

CTO? (Get Configuration of Trigger Output)

Description: Replies with the values set for specified trigger output lines and parameters

Format: CTO? [{<TrigOutID> <CTOPam>}]

Arguments: <TrigOutID>: is one digital output line of the controller; see CTO

<CTOPam>: parameter ID; see CTO

If all arguments are omitted, the values for all parameters are given for all output lines.

Response: {<TrigOutID> <CTOPam>"="<Value> LF}

For <Value> see CTO.

Notes: If <TrigOutID> and <CTOPam> are not omitted, only one parameter for one trigger line can be queried per CTO? command.

DIO? (Get Digital Input Lines)

Description: Gets the states of the specified digital input lines.

Use TIO? (p. 181) to get the number of available digital I/O lines.

Format: DIO? [{<DIOID>}]

Arguments: <DIOID> is the identifier of the digital input line, see below for details.

Response: {<DIOID>"="<InputOn> LF}

where

<InputOn> gives the state of the digital input line, see below for details.

Notes: The DIO? command can be used to directly read digital input lines which are located on the "I/O" socket (p. 260 or p. 261).

If <DIOID> is omitted, all lines are queried.

If <InputOn>=0, the digital input signal is LOW/OFF; if <InputOn>=1, the digital input signal is HIGH/ON.

DRC (Set Data Recorder Configuration)

Description: Set data recorder configuration: determines the data source and the kind of data (RecordOption) used for the given data recorder table.

Format: DRC <RecTableID> <Source> <RecOption>

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

<Source>: is the data source, for example an axis, output signal channel or input signal channel of the controller. The required source depends on the selected record option.

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

See below for a list of the available record options and the corresponding data sources.

Response: none

Notes:

The number of available data recorder tables can be read with TNR? (p. 182). The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. Using SPA (p. 170) or SEP (p. 168) you can change the parameter value in the range of 1 to 4 to increase or decrease the number of data recorder tables.

The total number of points available for data recording is 4096 (Data Recorder Max Points, ID 0x16000200). These points are allocated in equal shares to the available tables (i.e. to the number of tables given in the answer to TNR?).

With HDR? (p. 153) you will obtain a list of available options and information about additional parameters and commands concerned with data recording.

For detailed information see "Data Recording" (p. 74).

Record options for the appropriate data sources:

<Source>	<RecOption>
Axis	0 = Nothing is recorded 1 = Target Position of axis (i.e. target value in closed-loop operation), corresponds to the MOV? response 2 = Current Position of axis, corresponds to the POS? response 3 = Position Error of axis 14 = Open-Loop Control Value of axis (i.e. open-loop control value), corresponds to the SVA? response 15 = Closed-Loop Control Value of axis (i.e. output of PI servo controller) 22 = Slowed Target of axis (in closed-loop operation), target position after slew rate limitation
Output Signal Channel	16 = Piezo Output Voltage of output signal channel
Input Signal Channel	17 = Normalized Value of input signal channel 18 = Filtered Value of input signal channel

See "Axis Motion" (p. 26) for more information.

Example:

Send DRC 4 1 2
to record the current position of axis 1 in record table 4.

DRC? (get Data Recorder Configuration)

Description: Returns settings made with DRC (p. 146).

Format: DRC? [{<RecTableID>}]

Arguments: <RecTableID>: is one data recorder table of the controller; if omitted settings for all tables are given.

Response: The current DRC settings:

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

where

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

<RecOption>: is the kind of data to be recorded

See DRC for a list of the available record options and the corresponding data sources.

Notes: If <RecTableID> is not omitted, only one single data recorder table can be queried per DRC? command.

DRL? (Get Number of Recorded Points)

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

Format: DRL? [{<RecTableID>}]

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

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

where

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

Notes: The number of points is reset to zero for a data recorder table when changing its configuration with DRC (p. 146). Changing the number of data recorder tables via parameter 0x16000300 deletes the content of all tables.

If <RecTableID> is not omitted, only one single data recorder table can be queried per DRL? command.

DRR? (Get Recorded Data Values)

Description: Reading of the last recorded Data Set.

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, 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: The recorded data in GCS array format, see the separate manual for GCS array, SM 146E, and the example below

Notes: If <DataRecorderTable> is omitted, the data from all available tables will be read.

With HDR? (p. 153) you will obtain a list of available record options and trigger options and information about additional parameters and commands concerned with data recording.

For detailed information see "Data Recording" (p. 74).

Example:

```
STE 1 50
DRR? 1 10 1 3 4
# REM Dataset sent by E-709, Serial Number:0109025544
# REM Content: 10 Record Table Data of Record Table
1, 3 and 4 from Start Point 1
# TYPE = 1
# SEPARATOR = 9
# DIM = 3
# SAMPLE_TIME = 9.999999E-5
# NDATA = 10
# NAME0 = Real Position
# NAME1 = Open Loop Control Input
# NAME2 = Piezo Output Voltage
# END_HEADER
-0.08142092    -8.0    -8.000152
-0.08282473    42.0    -8.000152
-0.08352664    42.0    -5.18761
-0.07019045    42.0    -2.92439
0.02456665     42.0    1.733883
0.2807654      42.0    6.926834
0.7461374      42.0    12.21255
```

```
1.451561        42.0    17.50804
2.385107        42.0    22.80842
3.487113        42.0    28.10879
```

DRT (Set Data Recorder Trigger Source)

Description: Defines a trigger source for the given data recorder table.

Format: DRT <RecTableID> <TriggerSource> <Value>

Arguments: <RecTableID> is one data recorder table of the controller. See below for details.

<TriggerSource> ID of the trigger source, see below for a list of available options

<Value> depends on the trigger source, can be a dummy, see below

Response: none

Notes: The number of available data recorder tables can be read with TNR? (p. 182). The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. Using SPA (p. 170) or SEP (p. 168) you can change the parameter value in the range of 1 to 4 to increase or decrease the number of data recorder tables.

At present, the specified trigger source is always set for all data recorder tables, irrespective of the <RecTableID> value given in the DRT command. For compatibility reasons, <RecTableID> can also be 0.

With HDR? (p. 153) you will obtain a list of available record options and trigger options and additional information about data recording.

For detailed information see "Data Recording" (p. 74).

Available trigger options:

0 = default setting; data recording is triggered with IMP (p. 160), STE (p. 175), WGO (p. 200), WGR (p. 200); <Value> must be a dummy

1 = any command changing target position or voltage (MVR (p. 162), MOV (p. 161), SVA (p. 176), SVR (p. 180); in addition to IMP, STE, WGO, WGR); <Value> must be a dummy

2 = next command, resets trigger after execution;
<Value> must be a dummy

DRT? (Get Data Recorder Trigger Source)

Description: Returns 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 ID of the trigger source, see DRT (p. 150) for details

<Value> depends on the trigger source, if 0 it is a dummy, see DRT for details

Notes: If <RecTableID> is not omitted, only one single data recorder table can be queried per DRT? command.

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. Therefore you should call ERR? after each command.

The error codes and their descriptions are fully listed in "Error Codes" (p. 206).

Format: ERR?

Arguments: none

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

Troubleshooting: Communication breakdown

GWD? (Get Wave Table Data)

Description: Query waveform shape for given wave table.

The response to GWD? does not contain any offset to the wave generator output set with WOS (p. 201).

Format: GWD? [<StartPoint> [<NumberOfPoints> [{<WaveTableID>}]]]

Arguments: <StartPoint> is the start point in the wave table, starts with index 1

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

<WaveTableID> is one wave table of the controller

Response: The wave table contents (waveform) in GCS array format (see the separate manual for the GCS array, SM 146E, and the example below)

Notes: Depending on the waveform definition with WAV (p. 189), the wave tables may have different lengths. Due to the GCS array response format definition, it is not possible to read from tables of different lengths with one command line.

If the length of the wave tables differs, only tables with identical length can be read with the same command with the following syntax:

GWD? <StartPoint> <NumberOfPoints> {<WaveTableID>}

Example:

```
gwd? 1 10 1 2
# REM Dataset sent by E-709, Serial Number:0109025544
# REM Content: 10 Wave Table Data of Wave Table 1 and
2 from Start Point 1
# TYPE = 1
# SEPARATOR = 9
# DIM = 2
# SAMPLE_TIME = 9.999999E-5
# NDATA = 10
# NAME0 = Data Wave Table 1
# NAME1 = Data Wave Table 2
# END_HEADER
0.9401634      0.6970518
0.92187 0.683153
0.9037383      0.6693896
0.8857673      0.6557635
0.8679612      0.6422737
0.8503187      0.6289223
0.8328403      0.6157072
0.8155273      0.6026295
0.7983788      0.5896882
0.7813981      0.5768851
```

HDR? (Get All Data Recorder Options)

Description: List a help string which contains all information available about data recording (record options and trigger options, information about additional parameters and commands concerned with data recording).

Format: HDR?

Arguments: none

Response

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

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

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

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

end of help

Example

```
hdr?
#RecordOptions
0=Nothing is recorded
1=Target Position of Axis
2=Current Position of Axis
3=Position Error of Axis
7=Piezo Output Voltage of Output Signal Channel
14=Open-Loop Control Value of Axis
15=Closed-Loop Control Value of Axis (output of PID
controller)
22=Target Position with Slew Rate Limitation of Axis
#TriggerOptions
0=Default:
Performing a step triggers data recording (STE
<AxisID> <Amplitude>)
Performing an impulse triggers data recording (IMP
<AxisID> <Amplitude>)
Starting wave generator triggers data recording (WGO
<WaveGenID> <StartMode>)
Restart data recorder when wave generator is running
(WGR)
1=Any Command Changing Position or Voltage (MOV, MVR,
SVA, SVR), in addition to default
2=Next Command
#Parameters to be set with SPA
0x16000000=Data Recorder Table Rate
0x16000300=Data Recorder Chan Number
#Additional information
Set Data Recorder Configuration (DRC <RecTableID>
```

```
<Source> <RecOption>)
Get Data Recorder Configuration (DRC? [<RecTableID>])
Reading of last recorded Data (DRR? [<StartPoint>
[<NumberOfPoints> [<RecTableID>]]])
Get current Recorder Table Length (DRL?
[<RecTableID>])
Set Recorder Trigger Source (DRT <RecTableID>
<TriggerSource> <Value>)
Get Recorder Trigger Source (DRT? [<RecTableID>])
Set Data Recorder Table Rate (RTR <RecordTableRate>)
Get current Data Recorder Table Rate (RTR?)
Tell Number of available Data Recorders (TNR?)
end of help
```

HLP? (Get List Of Available Commands)

Description: List a help string which contains all commands available.

Format: HLP?

Arguments: none

Response: List of commands available

Troubleshooting: Communication breakdown

Notes: The HLP? response contains the commands provided by the current command level. See CCL (p. 138) for more information.

HPA? (Get List Of Available Parameters)

Description: Responds with a help string which contains all available parameters with short descriptions. See "Controller Parameters" (p. 223) for further details.

The listed parameters can be changed and/or saved using the following commands:

SPA (p. 170) affects the parameter settings in volatile memory (RAM).

WPA (p. 202) copies parameter settings from RAM to non-volatile memory.

SEP (p. 168) writes parameter settings directly into non-volatile memory (without changing RAM settings).

RPA (p. 165) resets RAM to the values from non-volatile memory.

Format: HPA?

Arguments: none

Response {<PamID>"="<string> LF}

where

<PamID> is the ID of one parameter, hexadecimal format

<string> is a string which describes the corresponding parameter.

The string has following format:

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

where

<CmdLevel> is the command level which allows write access to the parameter value

<MaxItem> is the maximum number of items of the same type which are affected by the parameter (the meaning of "item" depends on the parameter, can be axis, output signal channel, input signal channel, the whole system or internal hardware modules)

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

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

<ParameterDescription> is the parameter name

<PossibleValue> is one value from the allowed data range

<ValueDescription> is the meaning of the corresponding value

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>

<string> has the following format:

“#Possible parameter values are:

{<PamID> <ItemID> “=” <ListType>

[{TAB <PossibleValue> “=” <ValueDescription>}] }

#CCL levels are:

{<PamID> <ItemID> “=”<CmdLevel> }

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 one value from the allowed 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 which allows write access to the parameter value.

IDN? (Get Device Identification)

Description: Reports the device identity number. Is identical in function with the *IDN? command (p. 131).

Format: IDN?

Arguments: none

Response: One-line string terminated by line feed with controller name, serial number and firmware version, see *IDN? for an example.

IFC (Set Interface Parameters Temporarily)

Description: Interface configuration.

The baud rate setting for the RS-232 serial interface is specified. After IFC is sent, the new setting becomes active and the host PC interface configuration may need to be changed to maintain communication (close the current connection and re-open it with the new baud rate, see "Communication" (p. 63) for an example).

Baud rate settings made with IFC are lost when the controller is powered down. To save settings to non-volatile memory and thus make them the power-on defaults, use IFS (p. 158) instead.

Alternatively, you can change the baud rate setting with SPA (p. 170) or SEP (p. 168) and save the current value with WPA (p. 202) to non-volatile memory (provided that the current command level provides write access to the parameter, see CCL (p. 138)). For the appropriate parameter ID, see below.

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:

For <InterfacePam> = RSBAUD,
<PamValue> gives the baud rate to be used for RS-232 communication;
is also accessible as parameter ID 0x11000400, Uart Baudrate

Response: None

Note: Default baud rate is 57600. Other possible values are 300, 1200, 2400, 4800, 9600, 14400, 19200, 38400 and 115200

IFC? (Get Current Interface Parameters)

Description: Get the interface configuration parameter values from volatile memory.

The values from volatile memory can also be queried with SPA? (p. 173), for the corresponding parameter IDs see below.

Format: IFC? [{<InterfacePam>}]

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

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

where

<PamValue> gives the value of the interface parameter from volatile memory.

For <InterfacePam> = RSBAUD, <PamValue> gives the current baud rate of the RS-232 communication; is also accessible as parameter ID 0x11000400, Uart Baudrate

IFS (Set Interface Parameters As Default Values)

Description: Interface parameter store.

The power-on default parameters for the interface are changed in non-volatile memory, but the current active parameters are not. Settings made with IFS become active with the next power-on or reboot.

To change the baud rate setting for the RS-232 serial connection immediately (but temporarily) use IFC (p. 157).

It is also possible to change the default settings in non-volatile memory with SEP (p. 168) and to read them with the SEP? (p. 169) command (provided that the current command level provides write access to the parameter, see CCL (p. 138)).

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

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

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

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

The following interface parameter can be set:

RSBAUD
<PamValue> gives the baud rate to be used for RS-232 communication;
is also accessible as parameter ID 0x11000400, Uart Baudrate;
Possible values are 300, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600 and 115200

Response: none

Notes: Warning: The number of write cycles of non-volatile memory is limited. Write default values only when necessary.

If you use RPA (p. 165) to activate the changed baud rate settings the host PC interface configuration will need to be changed.

IFS? (Get Interface Parameters As Default Values)

Description: Get the interface configuration parameter values stored in non-volatile memory (i.e. the current power-on default)

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 non-volatile memory.

<InterfacePam> can be RSBAUD.

IMP (Start Impulse And Response Measurement)

Description: Starts performing an impulse and recording the impulse response for the given axis.

The data recorder configuration, i.e. the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 146).

The recorded data can be read with the DRR? command (p. 149).

Format: IMP <AxisID> <Amplitude>

Arguments: <AxisID> is one axis of the controller

<Amplitude> is the height of the impulse. See below for details.

Response: none

Troubleshooting: The control value resulting from the specified impulse height is out of limits:

Open-loop operation: the amplitude limitation depends on the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001)
Closed-loop operation: use TMN? (p. 182) and TMX? (p. 182) to ask for the current valid travel range limits.

Motion commands like IMP are not allowed when analog control input or wave generator output are active. See "Axis Motion" (p. 26) for details.

Notes: An "impulse" consists of a relative move of the specified amplitude followed by an equal relative move in the opposite direction. The impulse is performed relative to the current position or to the current piezo output voltage: In closed-loop operation (servo ON), the given amplitude is interpreted as position variation. In open-loop operation (servo OFF), the amplitude is interpreted as piezo output voltage variation.

IMP? (Get IMP Settings)

Description: Get last sent IMP settings for the given axis.

Format: IMP? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<Amplitude> is the height of the last commanded impulse. See IMP (p. 160) for details.

MOV (Set Target Position)

Description: Set new absolute target position for given axis.

Servo must be enabled for the commanded axis prior to using this command (closed-loop operation).

Format: MOV {<AxisID> <Position>}

Arguments <AxisID> is one axis of the controller

<Position> is the new absolute target position in physical units.

Response: none

Troubleshooting: Target position out of limits. Use TMN? (p. 182) and TMX? (p. 182) to ask for the current valid travel range limits.

Illegal axis identifier

Servo is Off for one of the axes specified.

Motion commands like MOV are not allowed when analog control input or wave generator output are active on the axis. See "Axis Motion" (p. 26) for details.

Notes: During a move, a new move command resets the target to a new value and the old one may never be reached.

The MOV command can be interrupted by #24 (p. 131) and STP (p. 176).

Example 1: Send: MOV 1 10
Note: Axis 1 moves to 10 (target position in µm)

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. 151) indicates that the target position given in the move 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 various sources, e.g. by commands that cause motion (MOV (p. 161), MVR (p. 162), IMP (p. 160), STE (p. 175)), by the wave generator and by an analog input signal. See "Axis Motion" (p. 26) for details.

MOV? gets the commanded positions. Use POS? (p. 164) to get the current positions.

MVR (Set Target Relative To Current Position)

Description: Move given axes relative to the last commanded target position.

The new target position is calculated by adding the given value <Distance> to the last commanded target value.

Servo must be enabled for the commanded axis prior to using this command (closed-loop operation).

Format: MVR {<AxisID> <Distance>}

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

<Distance> gives the distance to move; the sum of the distance and the last commanded target position is set as new target position (in physical units).

Response: none

Troubleshooting: Target position out of limits. Use TMN? (p. 182) and TMX? (p. 182) to ask for the current valid travel range limits, and MOV? (p. 162) for the current target.

Illegal axis identifier

Servo is Off for one of the axes specified.

Motion commands like MVR are not allowed when analog control input or wave generator output are active on the axis. See "Axis Motion" (p. 26) for details.

Notes: The MVR command can be interrupted by #24 (p. 131) and STP (p. 176).

Example:

Send:	MOV 1 0.5
Note:	This is an absolute move.
Send:	POS? 1
Receive:	1=0.500000
Send:	MOV? 1
Receive:	1=0.500000
Send:	MVR 1 2
Note:	This is a relative move.
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: Get on-target status of given axis.

If all arguments are omitted, gets status of all axes.

Format: ONT? [{<AxisID>}]

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

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

where

<uint> = "1" when the specified axis is on-target, "0" otherwise.

Troubleshooting: Illegal axis identifier

Notes: The on-target status is detected only in closed-loop operation (servo ON).

The on-target status is influenced by two parameters: settling window (On Target Tolerance, ID 0x07000900) and settling time (On Target Settling Time, ID 0x07000901).

The on-target status is true when the current position is inside the settling window and stays there for at least the settling time. The settling window is centered around the target position.

OVF? (Get Overflow State)

Description: Get overflow status of given axis.

If all arguments are omitted, gets status of all axes.

Overflow means that the control variables are out of range (can only happen if controller is in closed-loop operation).

Format: OVF? [{<AxisID>}]

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

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

where

<uint> = "0" (axis is not in overflow) or "1" (axis is in overflow)

Troubleshooting: Illegal axis identifier

POS? (Get Real Position)

Description: Returns the current axis position.

If all arguments are omitted, gets current position of all axes.

Format: POS? [{<AxisID>}]

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

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

where

<float> is the current axis position in physical units

Troubleshooting: Illegal axis identifier

Note: To request the current position of input signal channels (sensors) in physical units, use the TSP? (p. 185) command instead.

RBT (Reboot System)

Description: Reboot system. Controller behaves just like after power-on.

Format: RBT

Arguments: none

Response: none

RPA (Reset Volatile Memory Parameters)

Description: Resets the given parameter of the given item. The value from non-volatile memory is written into volatile memory.

Related commands:

With HPA? (p. 154) you can obtain a list of the available parameters. SPA (p. 170) affects the parameter settings in volatile memory, WPA (p. 202) writes parameter settings from volatile to non-volatile memory, and SEP (p. 168) writes parameter settings directly into non-volatile memory (without changing the settings in volatile memory).

See SPA for an example.

Format: RPA [{<ItemID> <PamID>}]

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

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

Response: none

Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	This procedure can take a few seconds. You can reset either all parameters or one single parameter with RPA.
Available item IDs and parameter IDs:	The item can be an axis, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" (p. 225).

RTR (Set Record Table Rate)

Description:	Sets the record table rate, i.e. the number of servo-loop 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 table rate to be used for recording operations (unit: number of servo-loop cycles), must be an integer value larger than zero
Response:	None
Notes:	RTR affects the Data Recorder Table Rate parameter, ID 0x16000000. The duration of the recording can be calculated as follows: Rec. Duration = Servo Update Time * RTR value * Number of Points where Servo Update Time is given in seconds by parameter 0x0E000200 Number of Points is the length of the data recorder table For more information see "Data Recording" (p. 74).

The record table rate set with RTR is saved in volatile memory (RAM) only. To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 202). Changes not saved with WPA will be lost when the controller is powered down.

RTR? (Get Record Table Rate)

Description: Gets the current record table rate, i.e. the number of servo-loop cycles used in data recording operations.

Format: RTR?

Arguments: None

Response: <RecordTableRate> is the table rate used for recording operations (unit: number of servo-loop cycles)

Notes: Gets the Data Recorder Table Rate parameter value in volatile memory (ID 0x16000000).

For more information see "Data Recording" (p. 74).

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 4 characters. Valid characters are 123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ

SAI changes the value of the Axis Name parameter, ID 0x07000600, in volatile memory (RAM). To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 202). Changes not saved with WPA will be lost when the controller is powered down.

SAI? (Get List Of Current Axis Identifiers)

Description: Gets the axis identifiers.

See also "Axes, Channels, Functional Elements" (p. 23).

Format: SAI? [ALL]

Arguments: [ALL] is optional. For controllers which allow for axis deactivation, [ALL] ensures that the answer also includes the axes which are "deactivated".

Response: {<AxisID> LF}

<AxisID> is one axis of the controller.

Notes: Gets the Axis Name parameter value in volatile memory (ID 0x07000600).

SEP (Set Non-Volatile Memory Parameters)

Description: Set a parameter of a given item to a different value in non-volatile memory, where it becomes the new power-on default.

After parameters were set with SEP, you can use RPA (p. 165) to activate them (write them to volatile memory) without controller reboot.

NOTICE: This command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Related commands:

HPA? (p. 154) returns a list of the available parameters.

SPA (p. 170) writes parameter settings into volatile memory (without changing the settings in non-volatile memory).

WPA (p. 202) writes parameter settings from volatile to non-volatile memory.

See SPA for an example.

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

Arguments	<Pswd> is the password for writing to non-volatile memory, default is "100" <ItemID> is the item for which a parameter is to be changed in non-volatile memory. See below for details. <PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details. <PamValue> is the value to which the given parameter of the given item is set
Response:	none
Troubleshooting:	Illegal item identifier, wrong parameter ID, invalid password, command level too low for write access
Notes:	To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 138). You can write only one single parameter per SEP command. Warning: The number of write cycles of non-volatile memory is limited. Write default values only when necessary.
Available item IDs and parameter IDs:	The item can be an axis identifier, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" p. 225).

SEP? (Get Non-Volatile Memory Parameters)

Description:	Get the value of a parameter of a given item from non-volatile memory. With HPA? (p. 154) you can obtain a list of the available parameters and their IDs.
Format:	SEP? [{<ItemID> <PamID>}]

Arguments:	<ItemID> is the item for which a parameter value from non-volatile 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 given parameter for the given item
Troubleshooting:	Illegal item identifier, wrong parameter ID
Note:	You can query either all parameters or one single parameter per SEP? command.
Available item IDs and parameter IDs:	The item can be an axis identifier, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" (p. 225).

SPA (Set Volatile Memory Parameters)

Description:	Set a parameter of a given item to a value in volatile memory (RAM). Parameter changes will be lost when the controller is powered down or rebooted or when the parameters are restored with RPA (p. 165). NOTICE: This command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

	Related commands: HPA? (p. 154) returns a list of the available parameters. SEP (p. 168) writes parameter settings directly into non-volatile memory (without changing the settings in volatile memory). WPA (p. 202) writes parameter settings from volatile to non-volatile memory. RPA resets volatile memory to the value in non-volatile memory.
Format:	SPA {<ItemID> <PamID> <PamValue>}
Arguments	<ItemID> is the item for which a parameter is to be changed in volatile memory. See below for details. <PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details. <PamValue> is the value to which the given parameter of the given item is set
Response:	none
Troubleshooting:	Illegal item identifier, wrong parameter ID,value out of range, command level too low for write access
Notes:	You can write only one single parameter per SPA command. To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 138).
Available item IDs and parameter IDs:	The item can be an axis identifier, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" (p. 225).
Example 1:	Send: SPA 1 0x16000000 8 Note: Set the Data Recorder Table Rate for the

controller to 8, parameter ID written in hexadecimal format
Send: SPA 1 369098752 2
Note: Sets the Data Recorder Table Rate for the controller to 2, parameter ID written in decimal format

Example 2: When analog command mode is selected for the axis, the analog input line must not participate as sensor in the axis position calculation. Hence the coefficient of the analog input in the InputSignalChannel-to-Axis matrix (Position From Sensor 2 parameter, ID 0x07000501) must be set to zero.

Send: CCL 1 advanced

Note: Switch to command level 1 because this level is required for write access to the Position From Sensor 2 parameter.

Send: SPA 1 0x07000501 0

Note: The analog input line no longer participates in the position calculation of axis 1. The setting is made in volatile memory only.

Now make further configuration settings in volatile memory using SPA and then test the functioning of the system. See "Using the Analog Input" (p. 67) for more information. If everything is okay and you want to use this system configuration after the next power-on, save the parameter settings from volatile to non-volatile memory.

Send: WPA 100

Note: When WPA is used without specifying any parameters, all currently valid parameter values from volatile memory are saved.

Send: SEP? 1 0x07000501

Receive: 1 0x07000501=0.0

Note: Check the parameter settings in non-volatile memory.

Example 3: The task performed in example 2 can also be done in the following way, provided you are sure that the new system configuration will work:

Send: CCL 1 advanced

Note: Switch to command level 1 because this level is required for write access to the Position From Sensor 2 parameter.

Send: SEP 100 1 0x07000501 0

Note: The analog input line no longer participates in the position calculation of axis 1. The setting is made in non-volatile memory and hence is the new power-on default, but is not yet active.

Make further configuration settings in non-volatile memory using SEP. See "Using the Analog Input" (p. 67) for more information. To use the new settings immediately, you now have to load them to volatile memory (otherwise they would become active after the next power-on or reboot of the controller).

Send: RPA

Note: The new configuration is now active.

Send: SPA? 1 0x07000501

Receive: 1 0x07000501=0.0

Note: Check the parameter settings in volatile memory.

SPA? (Get Volatile Memory Parameters)

Description: Get the value of a parameter of a given item from volatile memory (RAM).

With HPA? (p. 154) you can obtain a list of the available parameters and their IDs.

Format: SPA? [{<ItemID> <PamID>}]

Arguments:	<ItemID> is the item for which a parameter is to be queried in volatile memory. See below for details. <PamID> is the parameter 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 given parameter for the given item
Troubleshooting:	Illegal item identifier, wrong parameter ID
Note:	You can query either all parameters or one single parameter per SPA? command.
Available item IDs and parameter IDs:	The item can be an axis identifier, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" (p. 225).

SSN? (Get Device Serial Number)

Description:	Gets the serial number of the E-709.
Format:	SSN?
Arguments:	none
Response:	<SerialNumber> is the serial number of the device.
Note:	SSN? queries the value of the Device S/N parameter (ID 0x0d000000).

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. 146).

The recorded data can be read with the DRR? (p. 149) command.

Format: STE <AxisID> <Amplitude>

Arguments: <AxisID> is one axis of the controller

<Amplitude> is the height of the step. See below for details.

Response: none

Troubleshooting: The control value resulting from the specified step height is out of limits:

Open-loop operation: the amplitude limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001)
Closed-loop operation: use TMN? (p. 182) and TMX? (p. 182) to ask for the current valid travel range limits.

Motion commands like STE are not allowed when analog control input or wave generator output are active. See "Axis Motion" (p. 26) for details.

Notes: A "step" consists of a relative move of the specified amplitude. The step is performed relative to the current position or to the current piezo output voltage: In closed-loop operation (servo ON), the given amplitude is interpreted as position variation. In open-loop operation (servo OFF), the amplitude is interpreted as piezo output voltage variation.

STE? (Get STE Settings)

Description: Get last sent STE settings for the given axis.

Format: STE? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<Amplitude> is the height of the last commanded step. See STE (p. 175) for details.

STP (Stop All Axes)

Description: Stops all motion abruptly. For details see the notes below.

Sets error code to 10.

This command is identical in function to #24 (p. 131) which should be preferred when the controller is performing time-consuming tasks.

Format: STP

Arguments: none

Response: none

Troubleshooting: Communication breakdown

Notes: STP stops motion of all axes caused by move commands (MOV (p. 161), MVR (p. 162), SVA (p. 176), SVR (p. 180)), by the wave generator (WGO (p. 200)), and by the autozero procedure (ATZ (p. 135)). If analog command mode is enabled, STP enables the digital command mode. To recommence commanding the axis via the analog input, you have to re-enable analog command mode for the axis. See "How to work with the Analog Input" (p. 67) for more information.

After the axes are stopped, if servo is on their target positions are set to their current positions, or if servo is off, their open-loop control values are set to their last valid control values.

SVA (Set Open-Loop Axis Value)

Description: Set absolute open-loop control value to move the axis.

Servo must be switched off (open-loop operation) when using this command.

Format: SVA {<AxisID> <Amplitude>}

Arguments	<AxisID> is one axis of the controller <Amplitude> is the new absolute open-loop control value. See below for details.
Response:	none
Troubleshooting:	The control value specified by the given amplitude is out of limits. The limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001) of the output signal channel. Illegal axis identifier Servo is On for one of the specified axes Motion commands like SVA are not allowed when analog control input or wave generator output are active. See "Axis Motion" (p. 26) for details.
Notes:	This command can be interrupted by #24 (p. 131) and STP (p. 176). <Amplitude> gives the piezo output voltage for the axis. The given value will be output by the output signal channel (piezo amplifier).
Example 1:	Send: SVA 1 10 Send: VOL? Receive: 1=10.00061 Send: POS? Receive: 1=2.771162 Note: The piezo output voltage is set to 10 V which moves the axis to position 2.771162. Since the axis is in open-loop operation, the position is not maintained and can change over time due to drift and other effects, while the piezo output voltage is kept constant.
Example 2:	Send: SVA 1 300 Send: ERR? Receive: 66 Note: The axis does not move. The error code "66" reported by the ERR? command (p. 151) indicates that the piezo output voltage value given by SVA is out of limits.

SVA? (Get Open-Loop Axis Value)

Description:	Returns last valid open-loop control value of given axis.
Format:	SVA? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>"="<float> LF} where <float> is the last commanded open-loop control value. See below for details.
Troubleshooting:	Illegal axis identifier
Notes:	The open-loop control value is changed by multiple sources, e.g. by commands that cause motion (SVA (p. 176), SVR (p. 180), IMP (p. 160), STE (p. 175)), by the wave generator and by an analog input signal. See "Axis Motion" (p. 26) for details. The response to SVA? gives the current valid value of the piezo output voltage for the axis, i.e. the current piezo amplifier output.

SVO (Set Servo State)

Description:	Sets servo-control state for given axes (open-loop or closed-loop operation).
Format:	SVO {<AxisID> <ServoState>}
Arguments:	<AxisID> is one axis of the controller <ServoState> can have the following values: 0 = servo off (open-loop operation) 1 = servo on (closed-loop operation)
Response:	none
Troubleshooting:	Illegal axis identifier

Notes: Whenever the servo state is changed, SVO writes a control value to the target register or the open-loop control register. See "Axis Motion" (p. 26) for more information.

The current servo state affects the applicable move commands:
servo-control off: use SVA (p. 176) and SVR (p. 180)
servo-control on: use MOV (p. 161) and MVR (p. 162)

Using the Power Up Servo On Enable parameter (ID 0x07000800), you can configure the controller so that servo is automatically switched on upon power-on or reboot. To do this, set the value of the parameter to 1 in non-volatile memory (using SEP (p. 168) or SPA (p. 170) + WPA (p. 202)). To have write access to the parameter, it is necessary to switch to command level 1 using CCL (p. 138).

SVO? (Get Servo State)

Description: Gets servo-control state of given axes.

If all arguments are omitted, gets status of all axes.

Format: SVO? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<ServoState> is the current servo state of the axis:
0 = servo off (open-loop operation)
1 = servo on (closed-loop operation)

Troubleshooting: Illegal axis identifier

SVR (Set Relative Open-Loop Axis Value)

Description: Set open-loop control value relative to the current open-loop control value to move the axis.

The new open-loop control value is calculated by adding the given value <Difference> to the last commanded open-loop control value.

Servo must be off when using this command (open-loop operation).

Format: SVR {<AxisID> <Difference>}

Arguments: <AxisID> is one axis of the controller

<Difference> is the value which is added to the current open-loop control value. See below for details.

Response: none

Troubleshooting: The specified control value is out of limits. The limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001) of the output signal channel.

Illegal axis identifier

Servo is On for one of the specified axes

Motion commands like SVR are not allowed when analog control input or wave generator output are active. See "Axis Motion" (p. 26) for details.

Notes: This command can be interrupted by #24 (p. 131) and STP (p. 176).

<Difference> gives a voltage value. This value is added to the current piezo output voltage for the axis. The resulting value will be output by the output signal channel (piezo amplifier).

TAD? (Get ADC Value Of Input Signal)

Description: Get the current value from the specified input signal channel's A/D converter. Using this command it is possible to check for sensor overflow.

Format: TAD? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<uint> is the current A/D value, dimensionless

Notes: The TAD? response represents the digitized signal value without filtering and linearization.

Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 63)). TAD? reads the values for the individual input signal channels, not for a logical axis.

If <InputSignalID> is not omitted, only one single channel can be queried per TAD? command.

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: All digital I/O lines are found on the "I/O" socket (p. 260 or p. 261).

The digital output lines reported by TIO? can be programmed using the CTO command (p. 142).

The digital input line reported by TIO? can be used to trigger several items/events, see the CTI command (p. 140) for details.
Using the DIO? command (p. 146), you can query the state of the digital input line.

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 Range Limit min parameter, ID 0x07000000.

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

Description: The maximum commandable position is defined by the Range Limit max parameter, ID 0x07000001.

TNR? (Get Number of Record Tables)

Description: Get 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 answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. You can change the parameter value to increase or decrease the number of data recorder tables.

For more information see "Data Recording" (p. 74).

TNS? (Get Normalized Input Signal Value)

Description: Get the normalized value for the given input signal channel. This value is internally the input for the mechanics linearization.

Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 63)). TNS? reads the values for the individual input signal channels, not for a logical axis.

Format: TNS? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<float> is the normalized value ranging from controller specific minimum to maximum (e.g. -100 to 100), dimensionless

Notes: If <InputSignalID> is not omitted, only one single channel can be queried per TNS? command.

TPC? (Get Number of Output Signal Channels)

Description: Get the number of output signal channels available on the controller.

Format: TPC?

Arguments: none

Response <uint> is the number of output signal channels which are available; the answer gives the value of the Number Of Output Channels parameter, ID 0x0E000B01

Notes: The output signal channels are comprised of the piezo channels and any additional analog output channels. The number of piezo channels can be queried with the Number Of Piezo Channels parameter, ID 0x0E000B04. See "Axes, Channels, Functional Elements" (p. 23) for more information.

TRI (Set Trigger Input State)

Description: Enables or disables the trigger functionality of the given digital input line.

Format: TRI {<TrigInID> <TrigInMode>}

Arguments: <TrigInID> is one digital input line of the controller; for further information, see below.

<TrigInMode> can take on the following values:
0 = trigger disabled
1 = trigger enabled

Response: none

Troubleshooting: Illegal identifier of the digital input line

Notes: <TrigInID> corresponds to the Digital_IN_1 line (pin 10 of the "I/O" socket (p. 260 or p. 261)).

For trigger configuration details see CTI (p. 140) and WGO (p. 197).

The CTI and TRI settings are lost when the E-709 is switched off or rebooted.

The status of the digital input line can be queried with DIO? (p. 146).

TRI? (Get Trigger Input State)

Description: Gets the activation state of the trigger functionality of the given digital input line.

If all arguments are omitted, the state of all digital input lines is queried.

Format: TRI? [{<TrigInID>}]

Arguments: <TrigInID> is a digital input line of the controller; see the description of the TRI command (p. 184) for more information.

Response: {<TrigInID>"="<TrigInMode> LF}

where

<TrigInMode> is the current state of the digital input line:
0 = trigger disabled
1 = trigger enabled

TSC? (Get Number of Input Signal Channels)

Description: Get the number of input signal channels available on the controller.

Format: TSC?

Arguments: none

Response: <uint> is the number of input signal channels which are available; the answer gives the value of the Number Of Input Channels parameter, ID 0x0E000B00

Notes: The input signal channels are comprised of the sensor channels and any additional analog input channels. The number of sensor channels can be queried with the Number Of Sensor Channels parameter, ID 0x0E000B03. See "Axes, Channels, Functional Elements" (p. 23) for more information.

TSP? (Get Input Signal Position Value)

Description: Requests the current position of the selected input signal channel in physical units (µm).

Format: TSP? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<float> is the current position of the input signal channel, in physical units

Notes: Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 63)). TSP? reads the position values for the individual input signal channels, not for a logical axis. To get the current position of an axis, use POS? (p. 164) instead.

If <InputSignalID> is not omitted, only one single channel can be queried per TSP? command.

TWC (Clear All Wave Related Triggers)

Description: Clears all output trigger settings for the wave generators (the settings made with TWS (p. 187)) by switching the signal state for all points to "low".

For a detailed description see "Wave Generator" (p. 91) and "Configuring Trigger Output" (p. 85).

Format: TWC
Arguments: none
Response: none

TWG? (Get Number of Wave Generators)

Description: Get the number of wave generators available on the controller.

Format: TWG?

Arguments: none

Response <uint> is the number of wave generators which are available

TWS (Set Trigger Line Action To Waveform Point)

Description: Associates output trigger line and trigger line action (signal state high or low) with waveform point.

The power-on default state of all points is low. Afterwards, the signal state of the trigger output line can be switched to "low" for all points using the TWC command (p. 186). It is recommended to use TWC before trigger actions are set with TWS.

Generator trigger mode must be activated for the selected trigger output line with the CTO command (p. 142).

See also "Wave Generator" (p. 91) and "Configuring Trigger Output" (p. 85).

Format: TWS {<TrigOutID> <PointNumber> <Switch>}

Arguments: <TrigOutID> is one digital output line of the controller, see below for details

<PointNumber> is one point in the waveform, starts with index 1, see below for the timing calculation

<Switch> is the signal state of the digital output line: 0 = low, 1 = high

Response: None

Notes: <TrigOutID> corresponds to the Digital_OUT_1 and Digital_OUT_2 lines available on pins 11 and 12 of the "I/O" socket (p. 260 or p. 261). The trigger line actions defined with TWS are valid for both digital output lines since the lines share a common definition table.

You can define only one point per TWS command.

Example:

Send: TWS 1 1 1

Send: TWS 1 2 0

Send: TWS 1 3 0

Note: Sets trigger actions for Digital_OUT_1 (identifier 1), at waveform point 1 it is set high, points 2 and 3 are set low. These settings are also valid for Digital_OUT_2 since there is only one common definition table.

VEL (Set Closed-Loop Velocity)

Description:	Set velocity of given axes. VEL can be changed while the axis is moving.
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, axis is under joystick control (via host PC)
Notes:	<Velocity> must be > 0. VEL concerns the value of the Servo Loop Slew-Rate parameter, ID 0x07000200. The velocity set with VEL is saved in volatile memory (RAM) only. To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 202). Changes not saved with WPA will be lost when the controller is powered down. To have write access to the parameter, it might be necessary to switch to a higher command level using CCL (p. 138).

VEL? (Get Closed-Loop Velocity)

Description:	Get the current velocity value. If all arguments are omitted, gets current value of all axes.
Format:	VEL? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>"="<float> LF} where <float> is the current active velocity value in physical units / s.
Note:	The current value of the closed-loop velocity is given by the Servo Loop Slew-Rate parameter, ID 0x07000200, in volatile memory.

VOL? (Get Voltage Of Output Signal Channel)

Description: Read the current voltage value of the given output signal channel.

Format: VOL? [{<OutputSignalID>}]

Arguments: <OutputSignalID> is one output signal channel of the controller

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

where

<float> is the current voltage value in V

WAV (Set Waveform Definition)

Description: Define waveform of given type for given wave table. The waveforms are written to non-volatile flash memory.

NOTICE

Deterioration of the storage integrity.

Write waveforms only when necessary.

The non-volatile flash memory has a finite number of erase-write cycles. It is guaranteed to withstand around 10,000 write-erase-cycles, before the wear begins to deteriorate the integrity of the storage.

To allow for flexible definition, a waveform (wave table contents) can be built up by adding "segments". Each segment is created with a separate WAV command. Use the <AppendWave> argument (see below) to define the segment handling.

A segment can be based on predefined "curve" shapes (see the <WaveType> argument below).

Waveforms cannot be changed while they are being output by a wave generator. If you want to modify a waveform with WAV, first stop any wave generator output from the associated wave table.

The waveform values are absolute values. They give target positions in closed-loop operation and piezo

output voltage values in open-loop operation.

As long as the wave generator output is synchronized by servo-cycles and not paused by an external signal (see WGO (p. 200) for details), the duration of one output cycle for the waveform can be calculated as follows:

Output Duration = Servo Update Time * WTR value * Number of Points

where

Servo Update Time in seconds is given by parameter 0x0E000200

WTR (wave table rate) value gives the number of servo cycles the output of a waveform point lasts, default is 1

Number of Points is the length of the wave table (which is the sum of the lengths of all segments in this table)

See "How to work with the Wave Generator" (p. 91) for more information.

Format: WAV <WaveTableID> <AppendWave> <WaveType> <WaveTypeParameters>

Arguments: <WaveTableID> is the wave table identifier.

<AppendWave> This can be "X", "&" or "+":
- "X" clears the wave table and starts writing with the first point in the table.
- "&" appends the defined segment to the already existing wave table contents (i.e. concatenates a segment to lengthen the waveform).
- "+" adds the content of the defined segment to the already existing wave table contents (i.e. the values of the defined points are added to the existing values of that points); the defined segment must not be larger than the already existing wave table content.

<WaveType> The type of curve used to define the segment. This can be one of
- "SIN_P"(inverted cosine curve)
- "RAMP" (ramp curve)

"LIN" (single scan line curve)

<WaveTypeParameters> stands for the parameters of the curve and can be as follows:

For "SIN_P":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint> <CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the sine curve.

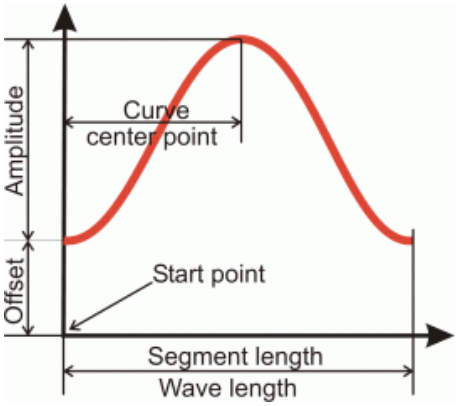
<Offset>: The offset of the sine curve.

<WaveLength>: The length of the sine curve in points (cycle duration).

<StartPoint>: The index of the starting point of the sine curve in the segment. Gives the phase shift. Lowest possible value is 0.

<CurveCenterPoint>: The index of the center point of the sine curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for more examples see "Defining Waveforms" (p. 98)):



For "RAMP":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint> <SpeedUpDown> <CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the ramp curve.

<Offset>: The offset of the ramp curve.

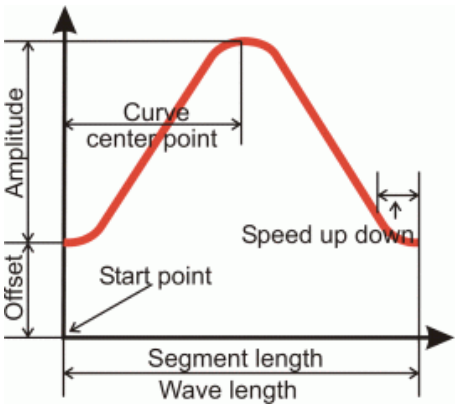
<WaveLength>: The length of the ramp curve in points (cycle duration).

<StartPoint>: The index of the starting point of the ramp curve in the segment. Gives the phase shift. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

<CurveCenterPoint>: The index of the center point of the ramp curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for more examples see "Defining Waveforms" (p. 98)):



For "LIN":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint> <SpeedUpDown>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table. If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

<Amp>: The amplitude of the scan line.

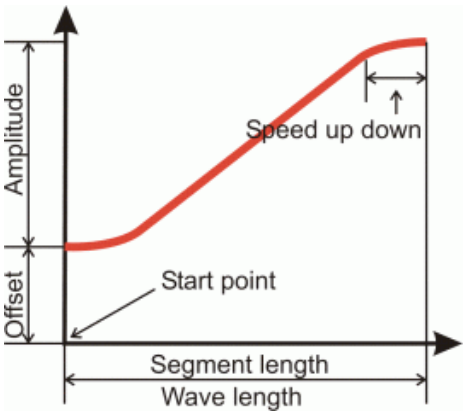
<Offset>: The offset of the scan line.

<WaveLength>: The length of the single scan line curve in points.

<StartPoint>: The index of the starting point of the scan line in the segment. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

Example (for more examples see "Waveform Definition" (p. 98)):



Response: None

Troubleshooting: Invalid wave table identifier

The total number of points for the waveform (which may consist of several segments) exceeds the available number of memory points.

Notes: The frequency of the wave generator output depends, among other factors, on the wave table length and on the wave generator table rate (WTR command). When you create waveforms, keep in mind that the usable frequency is limited by the available amplifier power. If the frequency is too high, a current limitation will be applied so that the waveform amplitude will be cut off.

WAV? (Get Waveform Definition)

Description: Get the value of a wave parameter for a given wave table.

See "How to work with the Wave Generator"s (p. 91) for more information.

Format: WAV? [{<WaveTableID> <WaveParameterID>}]

Arguments: <WaveTableID> is the wave table identifier.

<WaveParameterID> is the wave parameter ID, 1 = current wave table length in number of points; more parameters may be defined in the future

Response: {<WaveTableID> <WaveParameterID>"="<float> LF}

where

<float> depends on the <WaveParameterID>; gives the current number of waveform points in the wave table for <WaveParameterID> = 1

Troubleshooting: Invalid wave table identifier

Notes: If <WaveTableID> and <WaveParameterID> are not omitted, only one wave table can be queried per WAV? command.

WCL (Clear Wave Table Data)

Description: Clears the content of the given wave table.

As long as a wave generator is running, it is not possible to clear the connected wave table.

For a detailed description see "Wave Generator" (p. 91).

Format: WCL {<WaveTableID>}

Arguments: <WaveTableID> is the wave table identifier.

Response: none

Note: Only one wave table can be cleared per WCL command.

WGC (Set Number Of Wave Generator Cycles)

Description: Sets the number of output cycles for the given wave generator (the output itself is started with WGO (p. 200)).

For a detailed description see "Wave Generator" (p. 91).

Format: WGC {<WaveGenID> <Cycles>}

Arguments: <WaveGenID> is the wave generator identifier

<Cycles> is the number of wave generator output cycles. If cycles = 0 then the waveform is output without period limitation until it is stopped by WGO (p. 200) or #24 (p. 131) or STP (p. 176).

Response: None

WGC? (Get Number Of Wave Generator Cycles)

Description: Gets the number of output cycles set for the given wave generator.

For a detailed description see "Wave Generator" (p. 91).

Format: WGC? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<Cycles> LF}

where

<Cycles> is the number of wave generator output cycles set with WGC (p. 195).

WGI? (Get Index of Wave Table Point)

Description: Get the index of the wave table point which is currently output by the given wave generator.

Format: WGI? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<WavePointIndex> LF}

where

<WavePointIndex> is the index of the wave table point that is currently output by the wave generator, starts with 1.

Notes: If the wave generator was not started since the last power-on or reboot of the E-709, the response to WGI? is 1.

WGN? (Get Number of Completed Output Cycles)

Description: Get the number of output cycles that have been completed since the last start of the given wave generator.

Format: WGN? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<NumberOfOutputCycles> LF}

where

<NumberOfOutputCycles> is the current number of completed output cycles since the last start of the wave generator.

Notes: The cycle counter is resetted the next time the wave generator output is started with WGO bit 0 (“start output immediatley”, see p. 197).

The cycle counter is halted (but not resetted) when the wave generator output is stopped with WGO or #24 (p. 131) or STP (p. 176) or interrupted by trigger input (see CTI (p. 140)).

After 2^{32} cycles, a counter overflow occurs.

WGO (Set Wave Generator Start/Stop Mode)

Description: Start and stop the specified wave generator in the given mode. In addition, one data recording cycle is started, unless the wave generator was started by external trigger.

The number of output cycles can be limited by WGC (p. 195).

Using the WTR command (p. 205), you can lengthen the individual output cycles of the waveform.

The data recorder configuration can be made with DRC (p. 146). Recording can be restarted with WGR (p. 200).

Keep in mind that wave generator output will continue even if the terminal or the program from which it was started is quit.

The #9 single-character command (p. 130) can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode. With WGO? you can ask for the last-commanded wave generator start options (WGO settings).

For more information see "Wave Generator" (p. 91).

Format: WGO {<WaveGenID> <StartMode>}

Arguments: <WaveGenID> is the wave generator identifier

<StartMode> is the start mode for the specified wave generator.
In the WGO command, you supply the start mode in hex or decimal format. When no bits are set (<StartMode> = 0), there is no wave generator output for the associated axis.
Note that bit 8 (0x100 or 256) cannot start the wave generator output. It simply specifies a start option and must always be combined with one of the start modes specified in bit 0 (0x1 or 1) or bit 1 (0x2 or 2). See the examples below.

The start mode values in detail:

0: wave generator output is stopped. You can also use #24 (p. 131) or STP (p. 176) to stop the wave generator output, but WGO? (p. 200) will then still report the last commanded start mode.

bit 0 = 0x1 (hex format) or 1 (decimal format):
start wave generator output immediately, synchronized by servo cycle

bit 1 = 0x2 (hex format) or 2 (decimal format):
start wave generator output triggered by external signal, synchronized by servo cycle. The Digital_IN_1 line (pin 10 of the "I/O" socket (p. 260 or p. 261)) can be used to provide the external signal. The trigger functionality must be enabled with TRI (p. 184). The trigger configuration can be set with CTI and the Digital Trigger Input Usage parameter (ID 0x15000800; see p. 140). During the wave generator output, the data recording can be started with WGR (p. 200).

bit 8 = 0x100 (hex format) or 256 (decimal format):
wave generator started at the endpoint of the last cycle; start option.
The second and all subsequent output cycles each start at the endpoint of the preceding cycle which makes this start option appropriate to scanning applications. The final position is the endpoint of the last output cycle.

Response: None

Troubleshooting: Invalid wave generator identifier

There is no wave table connected to the wave generator. Use WSL (p. 204) to connect a wave table.

Wave generator output and analog control input:
It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In that case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be commanded by analog control input, you can stop the wave generator output, but not restart it.

Wave generator output and move commands:
When the wave generator output is active, move commands like MOV (p. 161) or SVA (p. 176) are not allowed for the associated axis.

See "Axis Motion" (p. 26) for details.

Example: Wave generator 1 is to be used with the "start at the endpoint of the last cycle" option, i.e. bit 8 on, contributing a value of 0x100 (dec.: 256) to <StartMode>. Because bit 8 is only a "start option" and does not actually start the wave generator output, a "start mode" ("immediately" or "triggered by external signal") must be chosen in addition. In this example, the wave generator is to be started by an external trigger signal, so bit 1 must be turned on, contributing 0x2 (dec.: 2), obtaining a <StartMode> value of 0x102 (dec.: 258).

Send the following WGO command, with the <StartMode> given in hex format:
WGO 1 0x102
The same command with <StartMode> given in decimal format:
WGO 1 258

Note that the trigger configuration must be enabled and set correctly for the digital input line (see TRI (p. 184) and CTI (p. 140)).

WGO? (Get Wave Generator Start/Stop Mode)

Description: Get the start/stop mode of the given wave generator.

The #9 single-character command (p. 130) can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode. With WGO? you can ask for the last-commanded wave generator start options (WGO settings).

Note that #24 (p. 131) or STP (p. 176) stop the wave generator output, but do not reset the start/stop mode settings so that WGO? will still report the start mode which was set by the last WGO command.

For more information see "Wave Generator" (p. 91).

Format: WGO? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<StartMode> LF}

where

<StartMode> is the last commanded start mode of the wave generator, in decimal format. The value may be the sum of several start options and one start mode. See the WGO command description (p. 200) for details.

WGR (Starts Recording In Sync With Wave Generator)

Description: Restarts recording when the wave generator is running (a first data recording cycle is started with the WGO command (p. 200) which starts the wave generator output).

The data recorder configuration can be made with DRC (p. 146). The recorded data can be read with the DRR? command (p. 149).

For more information see "Wave Generator" (p. 91) and "Data Recording" (p. 74).

Format: WGR

Arguments: None

Response: None

WOS (Set Wave Generator Output Offset)

Description: Sets an offset to the output of a wave generator. The current wave generator output is then created by adding the offset value to the current wave value:

Generator Output = Offset + Current Wave Value

Do not confuse the output-offset value set with WOS with the offset settings specified during waveform creation with WAV (p. 189). While the WAV offset affects only one segment (i.e. only one waveform), the WOS offset is added to all waveforms which are output by the given wave generator.

WOS sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory.

Deleting wave table content with WCL (p. 195) has no effect on the settings for the wave generator output offset.

For more information see "Wave Generator" (p. 91).

Format: WOS {<WaveGenID> <Offset>}

Arguments: <WaveGenID> is the wave generator identifier

<Offset> is the wave generator output offset, any float number. See below for details.

Response: None

Notes: You can change the Wave Offset parameter, ID 0x1300010b, also with SPA (p. 170) or SEP (p. 168) and save the value with WPA (p. 202) to non-volatile memory, where it becomes the power-on default.

In closed-loop operation (servo ON), the offset is interpreted as position value. In open-loop operation (servo OFF), the offset is interpreted as piezo output voltage value.

WOS? (Get Wave Generator Output Offset)

Description: Reads the current value of the offset which is added to the wave generator output.

For more information see also "Wave Generator" (p. 91).

Format: WOS? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<Offset> LF}

where

<Offset> is the current wave generator output offset. In closed-loop operation (servo ON), the offset is interpreted as position value. In open-loop operation (servo OFF), the offset is interpreted as piezo output voltage value.

Notes: The offset read by WOS? is the Wave Offset parameter value in volatile memory (ID 0x1300010b).

WPA (Save Parameters To Non-Volatile Memory)

Description: Write the currently valid value of a parameter of a given item from volatile memory (RAM) to non-volatile memory. The values saved this way become the power-on defaults.

NOTICE: If current parameter values are incorrect, the system may malfunction. Be sure that you have the correct parameter settings before using the WPA command.

RAM settings not saved with WPA will be lost when the controller is powered down or rebooted or when RPA (p. 165) is used to restore the parameters.

With HPA? (p. 154) you can obtain a list of all available parameters.

Use SPA? (p. 170) to check the current parameter settings in volatile memory.

See SPA (p. 170) for an example.

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

Arguments	<Pswd> is the password for writing to non-volatile memory. See below for details. <ItemID> is the item for which parameters are to be saved from volatile to non-volatile 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, command level too low for write access
Notes:	Parameters can be changed in volatile memory with SPA (p. 170), AOS (p. 132), ATZ (p. 135), IFC (p. 157), RTR (p. 166), SAI (p. 167), VEL (p. 188), WOS (p. 201) and WTR (p. 205). When WPA is used without specifying any arguments except of the password, all currently valid parameter values are saved. Otherwise only one single parameter can be saved per WPA command. To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 138). Warning: The number of write cycles of non-volatile memory is limited. Write default values only when necessary. NOTICE: Avoid powering down the E-709 during the WPA procedure.
Available passwords, item IDs and parameter IDs:	The password for writing to non-volatile memory is "100". <ItemID> can be an axis identifier, an input signal channel, an output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 225) for the item type concerned. See "Axes, Channels, Functional Elements" (p. 23) for the identifiers of the items. Valid parameter IDs are given in "Parameter Overview" (p. 225).

WSL (Set Connection Of Wave Table To Wave Generator)

Description: Wave table selection: connects a wave table to a wave generator or disconnects the selected generator from any wave table.

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

Deleting wave table content with WCL (p. 195) has no effect on the WSL settings.

As long as a wave generator is running, it is not possible to change its wave table connection.

For more information see "Wave Generator" (p. 91).

Format: WSL {<WaveGenID> <WaveTableID>}

Arguments: <WaveGenID> is the wave generator identifier

<WaveTableID> is the wave table identifier. If <WaveTableID> = 0, the selected generator is disconnected from any wave table.

Response: None

WSL? (Get Connection Of Wave Table To Wave Generator)

Description: Get current wave table connection settings for the specified wave generator.

For more information see "Wave Generator" (p. 91).

Format: WSL? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<WaveTableID> LF}

where

<WaveTableID> is the wave table identifier. If <WaveTableID> = 0, no wave table is connected to the wave generator.

WTR (Set Wave Generator Table Rate)

Description: Set wave generator table rate and interpolation type.

Format: WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}

Arguments: <WaveGenID> is the wave generator identifier. See below for details.

<WaveTableRate> is the table rate to be used for wave generator output (unit: number of servo-loop cycles), must be an integer value larger than zero

<InterpolationType> When a wave generator table rate higher than 1 is set, this option can be used to apply interpolation to the wave generator output between wave table points. For the available interpolation types see below.

Response: None

Notes: <WaveGenID> is 1

Using the WTR command, you can lengthen the individual output cycles of the waveform. The duration of one output cycle for the waveform can be calculated as follows:
Output Duration = Servo Update Time * WTR value * Number of Points
where
Servo Update Time is given in seconds by parameter 0x0E000200
WTR value gives the number of servo cycles the output of a waveform point lasts, default is 1
Number of Points is the length of the waveform (i.e. the length of the wave table)

WTR sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory. You can change this parameter also with SPA (p. 170) or SEP (p. 168) and save the value to non-volatile memory with WPA (p. 202). The value of the parameter in volatile memory can be read with the WTR? command (p. 205).

<InterpolationType> must be 0. The E-709 does not support any interpolation.

For more information see "Wave Generator" (p. 91). An application example can be found in "Modifying the Wave Generator Table Rate".

WTR? (Get Wave Generator Table Rate)

Description:	Gets the current wave generator table rate. Gets also the interpolation type used with table rate values > 1. For more information see "Wave Generator" (p. 91). An application example can be found in "Modifying the Wave Generator Table Rate".
Format:	WTR? [{<WaveGenID>}]
Arguments:	<WaveGenID> is the wave generator identifier
Response:	{<WaveGenID>"="<WaveTableRate> <InterpolationType> LF} where <WaveTableRate> is the table rate used for wave generator output (unit: number of servo-loop cycles) <InterpolationType> interpolation type applied to outputs between wave table points when a wave generator table rate higher than 1 is set. See below for available interpolation types.
Notes:	The wave table rate gives the number of servo-loop cycles used by the wave generator to output one waveform point. The wave table rate read by WTR? is the Wave Generator Table Rate parameter value in volatile memory (ID 0x13000109). <InterpolationType> is always 0. The E-709 does not support any interpolation.

15.4 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.

15.4.1 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 overran and command couldn't be received correctly
77	PI_CNTR_NOT_ENOUGH_RECORDED_DATA	Data record table does not hold enough recorded data
78	PI_CNTR_TABLE_DEACTIVATED	Data record table is not configured for recording
79	PI_CNTR_OPENLOOP_VALUE_SET_WHEN_SERVO_ON	Open-loop commands (SVA, SVR) are not allowed when servo is on
80	PI_CNTR_RAM_ERROR	Hardware error affecting RAM
81	PI_CNTR_MACRO_UNKNOWN_COMMAND	Not macro command
82	PI_CNTR_MACRO_PC_ERROR	Macro counter out of range
83	PI_CNTR_JOYSTICK_ACTIVE	Joystick is active
84	PI_CNTR_MOTOR_IS_OFF	Motor is off
85	PI_CNTR_ONLY_IN_MACRO	Macro-only command
86	PI_CNTR_JOYSTICK_UNKNOWN_AXIS	Invalid joystick axis
87	PI_CNTR_JOYSTICK_UNKNOWN_ID	Joystick unknown
88	PI_CNTR_REF_MODE_IS_ON	Move without referenced stage
89	PI_CNTR_NOT_ALLOWED_IN_CURRENT_MOTION_MODE	Command not allowed in current motion mode
90	PI_CNTR_DIO_AND_TRACING_NOT_POSSIBLE	No tracing possible while digital IOs are used on this HW revision. Reconnect to switch operation mode.
91	PI_CNTR_COLLISION	Move not possible, would cause collision
92	PI_CNTR_SLAVE_NOT_FAST_ENOUGH	Stage is not capable of following the master. Check the gear ratio.
93	PI_CNTR_CMD_NOT_ALLOWED_WHILE_AXIS_IN_MOTION	This command is not allowed while the affected axis or its master is in motion.
94	PI_CNTR_OPEN_LOOP_JOYSTICK_ENABLED	Servo cannot be switched on when open-loop joystick control is enabled.
95	PI_CNTR_INVALID_SERVO_STATE_FOR_PARAMETER	This parameter cannot be changed in current servo mode.
96	PI_CNTR_UNKNOWN_STAGE_NAME	Unknown stage name

97	PI_CNTR_INVALID_VALUE_LENGTH	Invalid length of value (too much characters)
98	PI_CNTR_AUTOZERO_FAILED	Autozero procedure was not successful
99	PI_CNTR_SENSOR_VOLTAGE_OFF	Sensor voltage is off
100	PI_LABVIEW_ERROR	PI 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_CMD	User profile mode: command is not allowed, check for required preparatory commands
1064	PI_CNTR_EXT_PROFILE_EXPECTING_MOTION_ERROR	User profile mode: first target position in user profile is too far from current position
1065	PI_CNTR_PROFILE_ACTIVE	Controller is (already) in user profile mode
1066	PI_CNTR_PROFILE_INDEX_OUT_OF_RANGE	User profile mode: block or data set index out of allowed range
1071	PI_CNTR_PROFILE_OUT_OF_MEMORY	User profile mode: out of memory
1072	PI_CNTR_PROFILE_WRONG_CLUSTER	User profile mode: cluster is not assigned to this axis
1073	PI_CNTR_PROFILE_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
1090	PI_CNTR_TOO_MANY_TCP_CONNECTIONS_OPEN	There are too many open tcpip connections
2000	PI_CNTR_ALREADY_HAS_SERIAL_NUMBER	Controller already has a serial number
4000	PI_CNTR_SECTOR_ERASE_FAILED	Sector erase failed
4001	PI_CNTR_FLASH_PROGRAM_FAILED	Flash program failed
4002	PI_CNTR_FLASH_READ_FAILED	Flash read failed
4003	PI_CNTR_HW_MATCHCODE_ERROR	HW match code missing/invalid
4004	PI_CNTR_FW_MATCHCODE_ERROR	FW match code missing/invalid
4005	PI_CNTR_HW_VERSION_ERROR	HW version missing/invalid
4006	PI_CNTR_FW_VERSION_ERROR	FW version missing/invalid
4007	PI_CNTR_FW_UPDATE_ERROR	FW update failed
4008	PI_CNTR_FW_CRC_PAR_ERROR	FW Parameter CRC wrong
4009	PI_CNTR_FW_CRC_FW_ERROR	FW CRC wrong
5000	PI_CNTR_INVALID_PCC_SCAN_DATA	PicoCompensation scan data is not valid
5001	PI_CNTR_PCC_SCAN_RUNNING	PicoCompensation is running, some actions cannot be performed during scanning/recording
5002	PI_CNTR_INVALID_PCC_AXIS	Given axis 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

15.4.2 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

15.4.3 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_UNKNOWN_PAM_IDS	PI stage database: Response to HPA? contains unknown parameter IDs.
-10007	PI_PARAMETER_DB_AND_QHPA_ARE_INCONSISTENT	PI stage database: Inconsistency between database and response to HPA?.
-10008	PI_PARAMETER_DB_SYSTEM_COULD_NOT_BE_ADDED	PI stage database: Stage has not been added.
-10009	PI_PARAMETER_DB_SYSTEM_COULD_NOT_BE_REMOVED	PI stage database: Stage has not been removed.
-10010	PI_PARAMETER_DB_CONTROLLER_DB_PARAMETERS_MISMATCH	Controller does not support all stage parameters stored in PI stage database. No parameters were set.
-10011	PI_PARAMETER_DB_DATABASE_IS_OUTDATED	The version of PISTAGES3.DB stage database is out of date. Please update via PIUpdateFinder. No parameters were set.
-10012	PI_PARAMETER_DB_AND_HPA_MISMATCH_STRICT	Mismatch between number of parameters present in stage database and available in controller interface. No parameters were set.
-10013	PI_PARAMETER_DB_AND_HPA_MISMATCH_LOOSE	Mismatch between number of parameters present in stage database and available in controller interface. Some parameters were ignored.
-10014	PI_PARAMETER_DB_FAILED_TO_SET_PARAMETERS_CORRECTLY	One or more parameters could not be set correctly on the controller.
-10015	PI_PARAMETER_DB_MISSING_PARAMETER_DEFINITIONS_IN_DATABASE	One or more parameter definitions are not present in stage database. Please update PISTAGES3.DB via PIUpdateFinder. Missing parameters were ignored.

16 Controller Parameters

16.1 Parameter Handling

To adapt the E-709 to your application, you can modify parameter values. The parameters available depend on the controller firmware. With HPA? (p. 154) you can obtain a list of all available parameters with information about each (e.g. short descriptions). The volatile and non-volatile memory parameter values can be read with the SPA? (p. 173) or SEP? (p. 169) commands, respectively.
Note that many parameters are "protected" by higher command levels, as indicated in the "Command Level" column in the "Parameter Overview" table (p. 225). By going to command level 1 using the CCL command (p. 138), it is possible to change level-1 parameters. Parameters with level 2 or higher are reserved for service personnel.

Using the "general" modification commands SPA, RPA, SEP and WPA, all parameters for which the currently active command level has write permission can be changed in volatile memory (SPA (p. 170), RPA (p. 165)) or in non-volatile memory (SEP (p. 168), WPA (p. 202)). It is recommended that any modifications be first made with SPA, and when the controller runs well, saved using WPA.

In addition to the "general" modification commands, there are commands which change certain specific parameters:

AOS (p. 132) (analog input offset)
ATZ (p. 135) (autozero result: offset of the polynomials used for mechanics linearization)
IFC (p. 157) (baud rate for RS-232 serial connection)
RTR (p. 166) (record table rate)
SAI (p. 167) (axis identifier)
VEL (p. 188) (servo loop slew-rate)
WOS (p. 201) (wave generator output offset)
WTR (p. 205) (wave table rate)

The commands listed above change the corresponding parameter value in volatile memory only, and WPA must be used to save changes to non-volatile memory.

You can use IFS (p. 158) to change and save the baud rate for RS-232 serial connections directly in non-volatile memory.

NOTICE

Incorrect parameter values may lead to improper operation or damage to your hardware. Be careful when changing parameters.

It is strongly recommended to save the parameter values of the E-709 to a file on the host PC before you make any changes. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-709, use the *Device Parameter Configuration* window provided by PIMikroMove. See "Creating Backup File for Controller Parameters" (p. 53) for more information.

INFORMATION

The *Device Parameter Configuration* window of PIMikroMove gives access to parameter values in a more convenient way. Use this window to check/edit the individual parameters. See the PIMikroMove manual for more information.

Each parameter refers to one of the following item types (see the "Item Type Concerned" column in the table below):

- Whole system
- Logical axes
- Input signal channels
- Output signal channels

The "Max. No. of Items" column shows the maximum number of items for which the parameter is used. Example: "2" for parameter 0x02000200 means that this parameter has different values for each of the 2 input signal channels. For parameters which refer to the whole system the maximum number of items is always 1. See "Axes, Channels, Functional Elements" (p. 23) for the item identifiers to use with SPA, SEP or WPA when changing/saving parameter values or when asking for parameter values with the SPA? or SEP? commands.

Values stored in non-volatile memory are power-on defaults, so that the system can be used in the desired way immediately. Note that PI records the data files of every E-709 controller calibrated at the factory for easy restoration of original settings should that ever be necessary.

When the stage is equipped with an ID-chip (located in the stage connector) and connected to the controller for the first time, the values for

stage-related parameters will be written from the ID-chip to the volatile and non-volatile memory of the E-709 upon controller power-on. You cannot overwrite the parameters in the ID-chip (this can only be done by PI). See "E-709.Cxx Models Only: ID-Chip Support / Stage Replacement" (p. 120) for more information. The parameters stored in the ID-chip are marked in the "Notes" column in the table below.

16.2 Parameter Overview

See "Parameter Handling" (p. 223) for the meaning of the individual columns.

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x02000100	1	Input Signal Channel	2	INT	Sensor Range Factor	ID-chip 1=3.12 2=2.10 3=1.20 4=1.005
0x02000101	1	Input Signal Channel	2	INT	Sensor Board Gain	ID-chip 1=1x 2=2x 3=3x
0x02000102	1	Input Signal Channel	2	INT	Sensor Offset Factor	ID-chip
0x02000200	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 1	ID-chip GAIN for analog input scaling
0x02000300	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 2	ID-chip OFFSET for analog input scaling
0x02000400	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 3	ID-chip
0x02000500	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 4	ID-chip
0x02000600	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 5	ID-chip
0x02000700	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 6	ID-chip

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x03000100	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 1	
0x03000101	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 1	
0x03000102	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 1	
0x03000103	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 1	
0x03000200	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 2	
0x03000201	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 2	
0x03000202	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 2	
0x03000203	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 2	
0x03000300	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 3	
0x03000301	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 3	
0x03000302	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 3	
0x03000303	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 3	
0x03000400	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 4	
0x03000401	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 4	
0x03000402	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 4	
0x03000403	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 4	
0x03000500	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 5	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x03000501	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 5	
0x03000502	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 5	
0x03000503	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 5	
0x03000600	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 6	
0x03000601	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 6	
0x03000602	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 6	
0x03000603	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 6	
0x03001000	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 1	
0x03001100	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 2	
0x03001200	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 3	
0x03001300	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 4	
0x03001400	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 5	
0x03001500	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 6	
0x03002000	2	Input Signal Channel	2	INT	Strain Gauge Bridge Configuration	0=not specified 1=full bridge 2=half bridge 3=quarter bridge
0x04000000	2	Input Signal Channel	2	FLOAT	PGA Correction Of Gain	ADC
0x04000001	2	Input Signal Channel	2	FLOAT	PGA Correction Of Gain	ADC

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x04000002	2	Input Signal Channel	2	FLOAT	PGA Correction Of Gain	ADC
0x04000003	2	Input Signal Channel	2	FLOAT	PGA Correction Of Gain	ADC
0x04000500	2	Input Signal Channel	2	FLOAT	Gain	ADC
0x04000600	2	Input Signal Channel	2	FLOAT	Offset	ADC
0x04000800	2	Input Signal Channel	2	FLOAT	HW_Offset	ADC
0x04000b00	3	Input Signal Channel	2	INT	ADC Bit Width	ADC
0x04000c00	3	Input Signal Channel	2	INT	Sensor Range Counter	ADC
0x05000000	1	Input Signal Channel	2	INT	Digital Filter Type	ID-chip 0=no filter 1=IIR lowpass 2=average filter 99=user filter
0x05000001	1	Input Signal Channel	2	FLOAT	Digital Filter Bandwidth	ID-chip
0x05000002	1	Input Signal Channel	2	INT	Digital Filter Order	ID-chip
0x05000101	1	Input Signal Channel	2	FLOAT	User Filter Param. 1	
0x05000102	1	Input Signal Channel	2	FLOAT	User Filter Param. 2	
0x05000103	1	Input Signal Channel	2	FLOAT	User Filter Param. 3	
0x05000104	1	Input Signal Channel	2	FLOAT	User Filter Param. 4	
0x05000105	1	Input Signal Channel	2	FLOAT	User Filter Param. 5	
0x06000500	0	Logical Axis	1	INT	Current Command Mode	0=digital 2=analog

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x06000501	0	Logical Axis	1	FLOAT	Analog Target Offset	
0x07000000	1	Logical Axis	1	FLOAT	Range Limit min	ID-chip
0x07000001	1	Logical Axis	1	FLOAT	Range Limit max	ID-chip
0x07000200	1	Logical Axis	1	FLOAT	Servo Loop Slew-Rate	ID-chip
0x07000201	1	Logical Axis	1	FLOAT	Open Loop Slew-Rate	ID-chip
0x07000300	1	Logical Axis	1	FLOAT	Servo-loop P-Term	ID-chip
0x07000301	1	Logical Axis	1	FLOAT	Servo-loop I-Term	ID-chip
0x07000302	1	Logical Axis	1	FLOAT	Servo-loop D-Term	ID-chip
0x07000500	1	Logical Axis	1	FLOAT	Position from Sensor 1	ID-chip
0x07000501	1	Logical Axis	1	FLOAT	Position from Sensor 2	ID-chip
0x07000600	0	Logical Axis	1	CHAR	Axis Name	ID-chip
0x07000800	1	Logical Axis	1	INT	Power Up Servo ON Enable	ID-chip
0x07000802	1	Logical Axis	1	INT	Power Up AutoZero Enable	ID-chip
0x07000803	1	Logical Axis	1	INT	Power Up Joystick Axis Enable	Reserved for special application
0x07000900	0	Logical Axis	1	FLOAT	ON Target Tolerance	ID-chip
0x07000901	0	Logical Axis	1	FLOAT	Settling Time	ID-chip
0x07000a00	1	Logical Axis	1	FLOAT	AutoZero Low Voltage	ID-chip
0x07000a01	1	Logical Axis	1	FLOAT	AutoZero High Voltage	ID-chip
0x07000c00	1	Logical Axis	1	FLOAT	Default Position	ID-chip
0x07000c01	1	Logical Axis	1	FLOAT	Default Voltage	ID-chip
0x07001005	1	Logical Axis	1	FLOAT	Position Report Scaling	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x07001006	1	Logical Axis	1	FLOAT	Position Report Offset	
0x08000100	1	Logical Axis	1	FLOAT	Notch frequency 1	ID-chip
0x08000101	1	Logical Axis	1	FLOAT	Notch frequency 2	ID-chip
0x08000200	1	Logical Axis	1	FLOAT	Notch Rejection 1	ID-chip
0x08000201	1	Logical Axis	1	FLOAT	Notch Rejection 2	ID-chip
0x08000300	1	Logical Axis	1	FLOAT	Notch Bandwidth 1	ID-chip
0x08000301	1	Logical Axis	1	FLOAT	Notch Bandwidth 2	ID-chip
0x09000000	1	Logical Axis	1	FLOAT	Driving Factor Of Piezo 1	ID-chip
0x09000001	1	Logical Axis	1	FLOAT	Driving Factor Of Piezo 2	ID-chip
0x0a000003	1	Output Signal Channel	1	INT	Select Output type	DAC
0x0a000004	1	Output Signal Channel	1	CHAR	Select Output index	DAC
0x0a000010	2	Output Signal Channel	1	FLOAT	DAC Coefficient 0	DAC
0x0a000020	2	Output Signal Channel	1	FLOAT	DAC Coefficient 1	DAC
0x0a000100	3	Output Signal Channel	1	FLOAT	DAC Bit Width	DAC
0x0b000007	2	Output Signal Channel	1	FLOAT	Min Output Voltage of Amplifier	
0x0b000008	2	Output Signal Channel	1	FLOAT	Max Output Voltage of Amplifier	
0x0c000000	1	Output Signal Channel	1	FLOAT	Soft Voltage Low Limit	ID-chip
0x0c000001	1	Output Signal Channel	1	FLOAT	Soft Voltage High Limit	ID-chip
0x0d000000	2	System	1	CHAR	Device S/N	
0x0d000700	3	System	1	CHAR	Hardware Name	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x0d000800	1	System	1	INT	Controller Address	Reserved for special application
0x0e000100	3	System	1	FLOAT	Sensor Sampling Time	in s
0x0e000200	3	System	1	FLOAT	Servo Update Time	in s
0x0e000b00	3	System	1	INT	Number of input channels	
0x0e000b01	3	System	1	INT	Number of output channels	
0x0e000b02	3	System	1	INT	Number of system axes	
0x0e000b03	3	System	1	INT	Number of sensor channels	
0x0e000b04	3	System	1	INT	Number of piezo channels	
0x0e000b05	3	System	1	INT	Number of trigger outputs	
0x0f000000	1	Input Signal Channel	1	INT	Power Up ID-Chip Enable	Only for the first channel
0x0f000100	1	Input Signal Channel	1	CHAR	Stage Type	ID-chip Only for the first channel
0x0f000200	1	Input Signal Channel	1	CHAR	Stage Serial Number	ID-chip Only for the first channel
0x10000500	0	System	1	INT	Fast IF Axis Input Usage	0=Disabled 1=Target
0x10000501	0	System	1	INT	Fast IF Data Type	0=32 bit float 1=16 bit uint 2=24 bit uint 3=32 bit uint
0x10000502	0	System	1	FLOAT	Fast IF Data Low Limit	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x10000503	0	System	1	FLOAT	Fast IF Data High Limit	
0x11000400	0	System	1	INT	Uart Baudrate	300, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, 115200
0x13000004	3	System	1	INT	Max Wave Points	
0x13000109	0	System	1	INT	Wave Generator Table Rate	
0x1300010a	3	System	1	INT	Number of Waves	
0x1300010b	0	Logical Axis	1	FLOAT	Wave Offset	
0x15000800	1	System	1	INT	Digital Trigger Input Usage	0=Not used 1=Wave generator > 1: Reserved for special applications
0x15001c00	1	System	1	FLOAT	Input Trigger Step Offset	Reserved for special application
0x15001c01	1	System	1	FLOAT	Input Trigger Step Value	Reserved for special application
0x16000000	0	System	1	INT	Data Recorder Table Rate	
0x16000100	3	System	1	INT	Max Number of Data Recorder Channels	
0x16000200	3	System	1	INT	Data Recorder Max Points	
0x16000300	0	System	1	INT	Data Recorder Chan Number	
0xffff0001	3	System	1	INT	Firmware valid/invalid Mark	
0xffff0002	3	System	1	INT	CRC-32 of Firmware Program Code	
0xffff0003	3	System	1	INT	CRC-32 of Firmware Description	
0xffff0004	3	System	1	INT	Version of Firmware Description	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0xffff0006	3	System	1	CHAR	Unique Firmware Name	
0xffff0007	3	System	1	CHAR	Unique Board Name	
0xffff0008	3	System	1	INT	Version of Firmware	
0xffff000b	3	System	1	INT	Maximal Size of Flash	
0xffff000c	3	System	1	CHAR	Logical Device	
0xffff000d	3	System	1	CHAR	Description of Firmware	
0xffff000e	3	System	1	CHAR	Date of Firmware Development	
0xffff000f	3	System	1	CHAR	Name of Firmware Developer	
0xffff0010	3	System	1	INT	Length of Firmware	
0xffff0011	3	System	1	INT	Firmware Compatibility Index	
0xffff0012	3	System	1	INT	Relative Address from FW-Description to FW-Start	
0xffff0013	3	System	1	CHAR	Logical Device Type	
0xffff0014	3	System	1	INT	Hardware Revision of Board	inactive, does not show the current hardware revision
0xffff0015	3	System	1	INT	Execution Address of Firmware	
0xffff0016	3	System	1	INT	Configuration Options	

17 Maintenance

17.1 Updating Firmware

The current firmware revision of your E-709 can be identified in the answer of the *IDN? command (p. 156). Example of a response of the E-709:

PHYSIK INSTRUMENTE,E-709,0110036156,5.001

- 0110036156: Serial number of the E-709
- 5.001: Firmware version

To update the firmware of E-709, the USB interface must be used with the PI Firmware Update Wizard PC software.

Requirements

- You have installed the PI Firmware Update Wizard on the PC (p. 42).
- You have obtained the current firmware file from our customer service department (p. 246) and copied the file to a directory on the PC. Make sure that this directory only contains the current firmware file.
  Example for a file name: E855F0013_HW12000_FW05001.hex

Updating the firmware

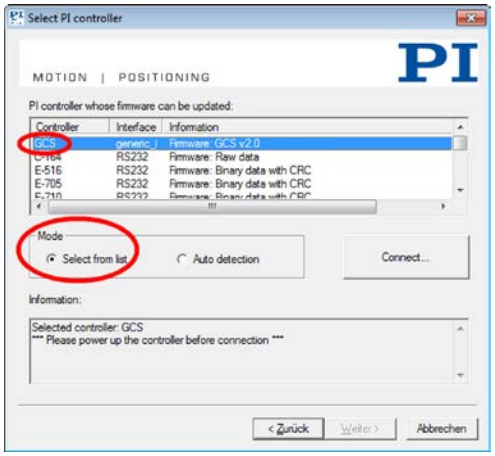
Proceed as follows to update the firmware:

1. Connect the E-709 to the host PC via an USB-A/USB-B cable.
2. Switch on the E-709: Connect the "24 VDC" socket of the E-709 to a suitable power supply, and connect the power cord of the power supply to the wall socket.
3. Start the PI Firmware Update Wizard on the PC.
   - Use Start > All programs > PI > PIFirmwareWizard, or navigate to the folder, where the PIFirmwareWizard is installed on your PC, and start it by double-clicking on PIFirmwareWizard.exe

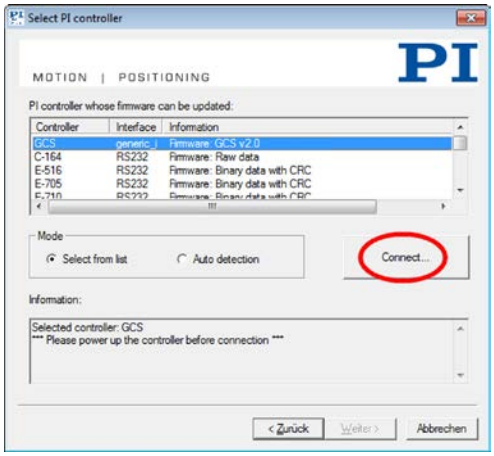
4 In the *Welcome* window, click *Next* (“Weiter” means “Next”, “Abbrechen” means “Abort”).

The *Select PI controller* window will open.

5 Make sure that *Select from list* is selected. Then select *GCS* as controller from the list:

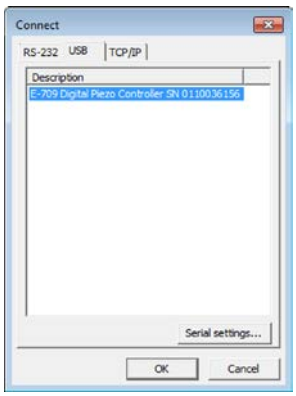


6 Click *Connect* ….



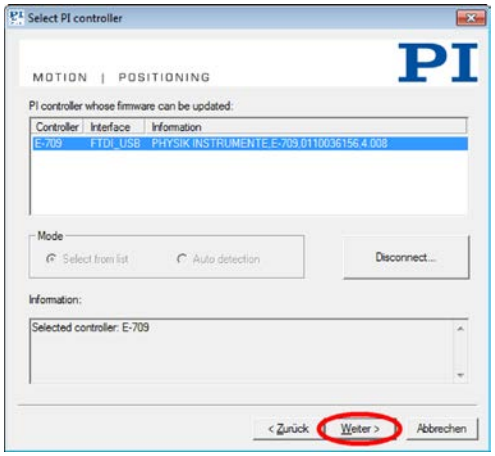
The *Connect* window will open.

7 Select the *USB* tab. When the controller description is displayed as in the example, click *OK* to establish the connection.



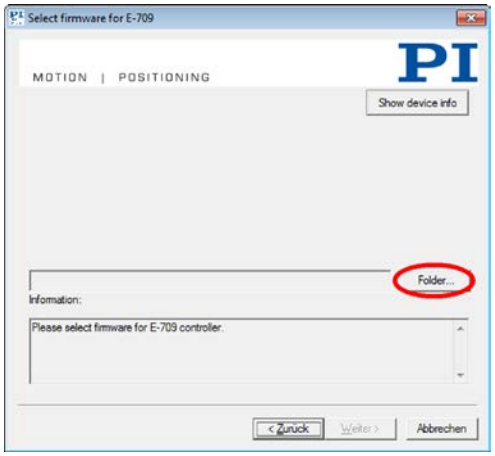
The *Connect* window will be closed. In the *Select PI controller* window, now information on the connected E-709 should be displayed as shown below.

8 Click *Next* (“Weiter” means “Next”, “Abbrechen” means “Abort”, “Zurück” means “Back”).



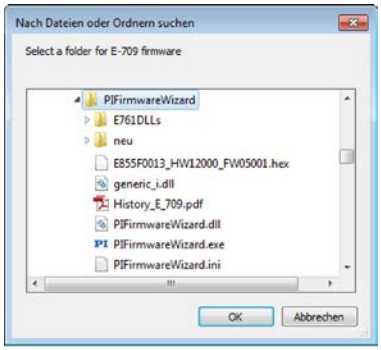
The window changes to the *Select firmware for E-709* step.

9 Click *Folder...* to select the directory where you have saved the firmware file on the PC:



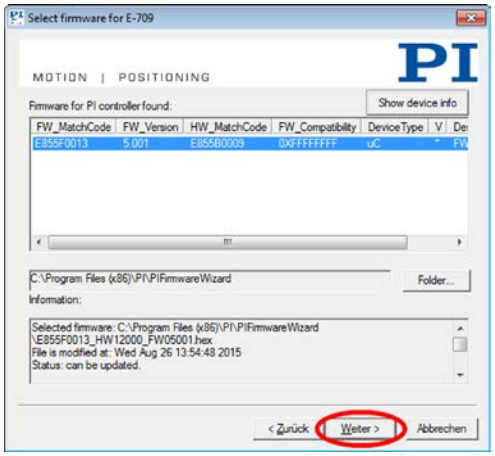
A selection window opens.

10 Select the directory, where the file is located (do not select the file!), and click *OK*:



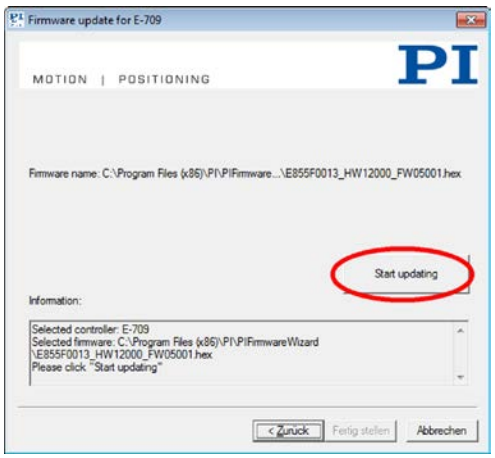
The selection window will be closed. In the *Select firmware for E-709* window, now the firmware is displayed as shown below.

11 Click *Next* (“Weiter” means “Next”):

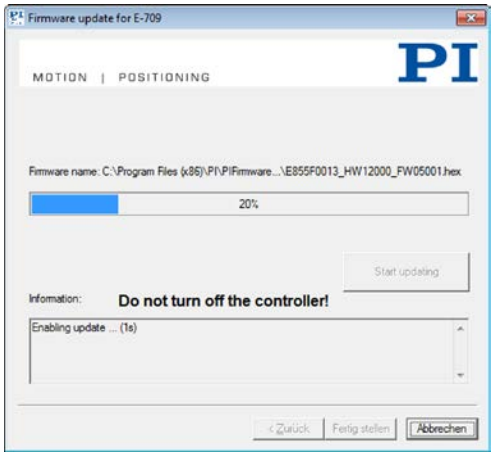


The window changes to the *Firmware update for E-709* step.

12 Click *Start updating* to start the firmware update.

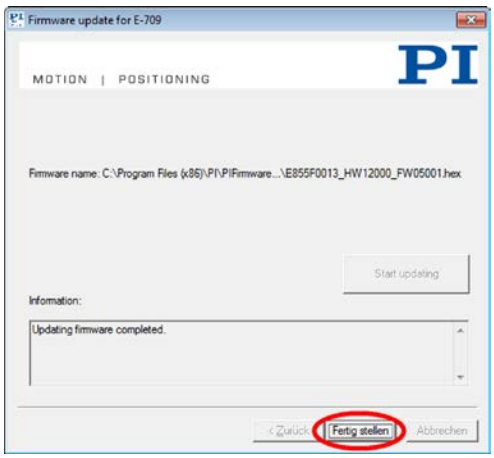


The update progress will be displayed in the window (see below). Do not turn off the E-709 while the firmware update is in progress.



When the firmware update was successful, the E-709 will be rebooted automatically with the new firmware. After the reboot, the *Firmware update for E-709* window should display a “completed” message as shown below.

13 Click *Finish* to close the PI Firmware Update Wizard (“Fertig stellen” means “Finish”).



17.2 Cleaning the E-709

NOTICE

The E-709 contains electrostatic sensitive devices that can be damaged by short circuits or flashovers when cleaning fluids penetrate the housing.

→ Before cleaning, disconnect the E-709 from the power source by removing the mains plug.

→ Prevent cleaning fluid from penetrating the housing.

E-709 bench-top units only:

1. Disconnect the E-709 from the power source.
2. Wait a minute to be sure that any residual voltage has dissipated.
3. Clean the housing surfaces of the E-709 using mild detergents or disinfectant solutions.

18 Troubleshooting

If the problem that occurred with your system is not listed below or cannot be solved as described, contact our customer service department (p. 246).

Communication with controller does not work

Communication cable is wrong or defective

⇒ Check cable. Does it work properly with another device?

For RS-232, a null-modem cable must be used.

The interface is not configured correctly

⇒ With the RS-232 interface, check port and baud rate (depending on your controller, the baud rate may be set via DIP switches on the front panel or via a controller parameter). It is recommended that the host PC have a "genuine" RS-232 interface on board. If the host PC uses a USB-to-serial adapter instead, data loss could occur during communication, especially when transferring large amounts of data.

⇒ The USB interface is dominant. This means that when RS-232 and USB interfaces are connected to the host PC via the corresponding cables, communication is possible only via the USB interface.

Another program is using the interface

⇒ Close the other program.

Specific software has problems

⇒ See if the system works with some other software, e.g. a terminal or development environment. You can, for example, test the communication by simply starting a terminal program, e.g. PI Terminal, and entering commands like *IDN? or HLP?. Note that multi-character commands are transferred as terminated by a LF (line feed) character and are executed only after the LF is received.

Stage does not move

Cable not connected properly

⇒ Check the connecting cable(s)

Stage or stage cable is defective

⇒ Exchange stage with a working stage of the same type to test a new combination of controller and stage:
Adjust the sensor zero point for the new stage by running an AutoZero procedure. You can use the ATZ command (p. 135) or the AutoZero functionality of PIMikroMove. Do not use the "Zero" trim pot for sensor zero-point adjustment. It is adjusted before delivery. If the system performance is still unsatisfactory, send the system back to PI for a new calibration.

Wrong command or wrong syntax

⇒ Check the error code with the ERR? command (p. 151). "Error Codes" (p. 206) gives the complete error reference.

Wrong axis commanded

⇒ Check if the correct axis identifier is used and if the commanded axis is that of the desired stage (axis identifier also required with single-axis systems!)

Move commands or wave generator output provoke errors and are ignored

⇒ The axis motion can result from multiple control sources (see "Axis Motion" (p. 26) for details). The sources have different write priorities:

Motion commands like MOV, MVR, SVA, SVR, IMP and STE are not allowed (will cause an error) when analog control input or wave generator output are active.

It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In this case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be

commanded by analog control input, you can stop the wave generator output, but not restart it.

The analog control input is ignored

⇒ When the analog input is used as control source and the axis motion is stopped with STP (p. 176) or #24 (p. 131), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis. See "How to work with the Analog Input" (p. 67) for more information.

Sensor zero point is not set correctly so that the commanded control value cannot be realized

⇒ For systems with linear piezo actuators, both the range of sensor position values and the range of the output drive voltages are limited. If mechanical drift of the piezo actuator causes too great a shift in the relation between these ranges, then the usable closed-loop travel range will be reduced. Such an offset can be compensated by the AutoZero function. If AutoZero is compatible with your application, use the ATZ command (p. 135) or the AutoZero functionality of PIMikroMove.

Incorrect configuration

⇒ Check the parameter settings on the E-709 with the SPA? (p. 173) and SEP? (p. 169) commands.

Unsatisfactory system performance

The sensor values are not reliable, and the whole system is instable.

⇒ Only thermally stable systems can have the best performance. For a thermally stable system, power on the E-709 at least one hour before you start working with it.

⇒ If the E-709 is not used, but should remain switched on to ensure the temperature stability: Make sure that the servo mode is switched off (open-loop operation) and the piezo output voltage is set to 0 V.

Stage is oscillating or positions inaccurately

The load was changed. Unsuitable settings of the notch filter and the servo-control parameters of the E-709 can cause the stage to oscillate or to position inaccurately. Oscillations can damage the stage and/or the load affixed to it.

⇒ If the stage is oscillating (unusual operating noise), immediately switch off the servo mode or switch off the E-709.

⇒ Only switch on the servo mode after you have modified the settings of the notch filter and the servo-control parameters of the E-709; see „Adjusting the Notch Filter(s) in Open-Loop Operation“ (p. 110) and "Checking and Optimizing the Servo-Control Parameters" (p. 115).

Electromagnetic signal causes noise of the sensor signal.

⇒ Check the sensor signal.

If the sensor signal seems to be abnormal:

⇒ Avoid interfering signals.

⇒ Take particular care to ensure suitable shielding and grounding. For more information, download the “Guide to Grounding and Shielding” from our website:

1. Open the website www.pi.ws.
2. Search for A000T0074.
3. In the search results, click the *Downloads* tab.
4. Download the Technical Note A000T0074 “Guide to Grounding and Shielding” file.

Custom software accessing PI drivers does not run.

Wrong combination of driver routines/VIs

⇒ Check if system runs with Terminal program. If yes read the software manual and compare sample code from the E-709 CD to check the necessary driver routines.

Device Parameter Configuration window is not available in PIMikroMove.

NI LabVIEW Run-Time Engine has not been installed

⇒ Install the NI LabVIEW Run-Time Engine, see "Performing the Initial Installation" (p. 42).

19 Customer Service

For inquiries and orders, contact your PI sales engineer or send us an email (mailto:service@pi.de).

➔ If you have any questions concerning your system, provide the following information:
- Product and serial numbers of all products in the system
- Firmware version of the controller (if applicable)
- Version of the driver or the software (if applicable)
- Operating system on the PC (if applicable)

➔ If possible: Take photographs or make videos of your system that can be sent to our customer service department if requested.

Only PI service personnel must repair the E-709.

The latest versions of the user manuals are available for download (p. 8) on our website.

20 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

21 Technical Data

21.1 Specifications

	E-709.SR, E-709.SRG, E-709.PR, E-709.PRG
Function	Digital controller for single-axis piezo nanopositioning systems (.SR, .PR: OEM module)
Channels	1
Processor	DSP 32-bit floating point, 150 MHz
Servo characteristics	PID, two notch filters, sensor linearization
Sampling rate	10 kHz
Sampling rate, sensor	10 kHz
Sensor	
Sensor type	Metal foil strain gauge sensors (.SR, .SRG) Piezoresistive sensors (.PR, .PRG)
Linearization	5th order polynomials
Sensor bandwidth	5 kHz
Sensor resolution	16 bit
Ext. synchronization	No
Amplifier	
Output voltage	-30 V to +130 V
Peak output power	10 W (<5 ms)
Average output power	5 W (>5 ms)
Peak current	100 mA (<5 ms)
Average current	50 mA (>5 ms)
Current limitation	Short-circuit-proof
Resolution DAC	17 bit

Interfaces and operation	
Communication interfaces	USB, RS-232, SPI
Piezo / sensor connector	Sub-D 9-pin
I/O connector	HD-Sub-D 26-pin 1x analog input 0 to 10 V (external sensor or control input) 1x analog output 0 to 10 V (position monitor or control voltage) 1x monitor of amplifier output -0.3 to 1.3 V 1x digital input (LVTTL, programmable) 5x digital output (LVTTL, 3x predefined, 2x programmable)
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Software drivers	LabVIEW drivers, shared libraries for Windows and Linux, MATLAB, MetaMorph, µManager, Andor iQ
Supported functionality	Wave generator, data recorder, auto zero, trigger I/O
Display	Status LED, overflow LED
Miscellaneous	
Operating temperature range	5 to 50 °C (over 40 °C, max. av. power derated)
Mass	E-709.xR: 0.26 kg E-709.xRG: 0.47 kg
Operating voltage	24 V DC (for benchtop devices in the scope of delivery: external power adapter)
Max. power consumption	24 W
Current consumption without load (typ.)	275 mA at 24 V operating voltage

	E-709.CRG, .CR	E-709.CHG
Function	Digital controller for single-axis piezo nanopositioning systems (.CR: OEM module)	
Channels	1	
Processor	DSP 32-bit floating point, 150 MHz	
Servo characteristics	PID, two notch filters, sensor linearization	
Sampling rate	10 kHz	
Sampling rate, sensor	10 kHz	
Sensor		
Sensor type	Capacitive	
Linearization	5th order polynomials	
Sensor bandwidth	5 kHz	
Sensor resolution	16 bit	
Ext. synchronization	not available	yes
Amplifier		
Output voltage	-30 V to +130 V	
Peak output power, < 2ms	10 W	50 W
Average output power	5 W	15 W
Peak current, < 2ms	100 mA	500 mA
Average current	50 mA	160 mA
Current limitation	Short-circuit-proof	
DAC Resolution	17 bit	

	E-709.CRG, .CR	E-709.CHG
Interfaces and operation		
Communication interfaces	USB, RS-232, SPI	
Piezo / sensor connector	Sub-D special connector	
I/O connector	HD-Sub-D 26-pin: 1x analog input 0 to 10 V (external sensor or control input) 1x analog output 0 to 10 V (position monitor or control voltage) 1x monitor of amplifier output -0.3 to 1.3 V 1x digital input (LVTTL, programmable) 5x digital output (LVTTL, 3x predefined, 2x programmable)	
Command set	PI General Command Set (GCS)	
User software	PIMikroMove	
Software drivers	LabVIEW drivers, shared libraries for Windows and Linux, MATLAB, MetaMorph, µManager, Andor iQ	
Supported functionality	Wave generator, data recorder, auto zero, trigger I/O	
Display	Status LED, overflow LED	
Miscellaneous		
Operating temperature range	5 to 50 °C, above 40°C power derated	5 to 50 °C
Mass	E-709.CRG: 0.47 kg E-709.CR: 0.26 kg	2.5 kg
Operating voltage	24 V DC (for benchtop devices in the scope of delivery: external power adapter)	
Max. power consumption	24 W	45 W
Current consumption without load (typ.)	320 mA at 24 V operating voltage	240 mA at 24 V operating voltage

21.2 Maximum Ratings

The E-709 is designed for the following operating data:

Input on:	**Maximum Operating Voltage** 	**Operating Frequency** 	**Maximum Power Consumption** 
Barrel connector	24 V	⎓	E-709.xRG, .xR: 24 W E-709.CHG: 45 W

21.3 Ambient Conditions and Classifications

The following ambient conditions and classifications must be observed for the E-709:

Area of application	For indoor use only
Maximum altitude	2000 m
Relative humidity	Highest relative humidity 80% for temperatures up to 31°C Decreasing linearly to 50% relative humidity at 40°C
Storage temperature	0°C to 70°C
Transport temperature	–25°C to +85°C
Overvoltage category	II
Protection class	I
Degree of pollution	2
Degree of protection according to IEC 60529	IP20

21.4 System Requirements

The following system requirements must be met to operate the E-709:

- A PC with Windows operating system (Windows 7, 8 and 10 (32 bit, 64 bit)) or Linux operating system (kernel 2.6, GTK 2.0, from glibc 2.15). Note that not all software components are available for Linux PCs. See "Overview of PC Software" (p. 39) for more information.
- E-709 CD with host software
- Communication interface to the PC:
  RS-232: a free COM port on the PC, a null-modem cable, and with E-709.xRG and .xR models, the E709B0002 RS-232 adapter
  USB: a free USB port, USB cable, USB drivers (installed on the PC with the software, see p. 42)
- The mechanics (piezo stage) the controller was calibrated with
- If you intend to use the analog and digital lines provided by the "I/O" socket of the E-709 (analog input and output, digital I/O, status and enable signals), an adapter is required (see "Additional Components" for available adapters (p. 22)).
  Note that with E-709.CHG models, the lines for analog input and output are available on separate SMB connections.

21.5 Dimensions

21.5.1 E-709.xRG Bench-Top Devices

Dimensions in millimeters, decimal places separated by commas.

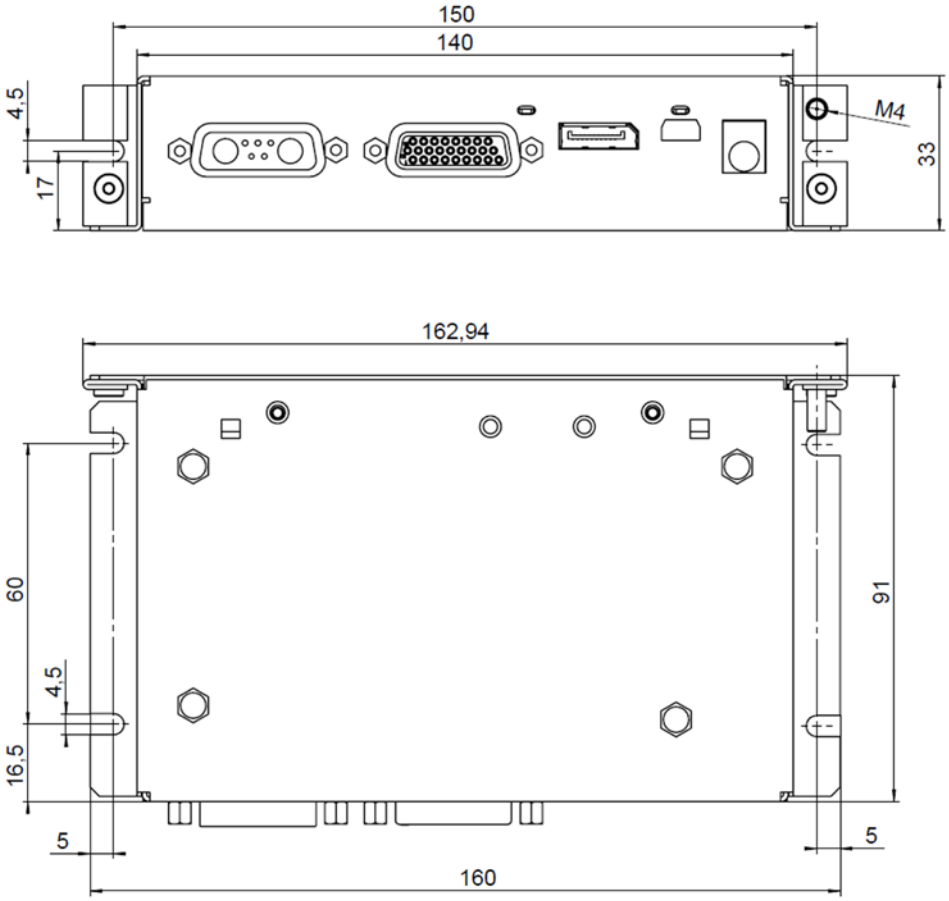


Figure 22: E-709.xRG dimensions

21.5.2 E-709.CHG Bench-Top Devices

Dimensions in millimeters, decimal places separated by commas.

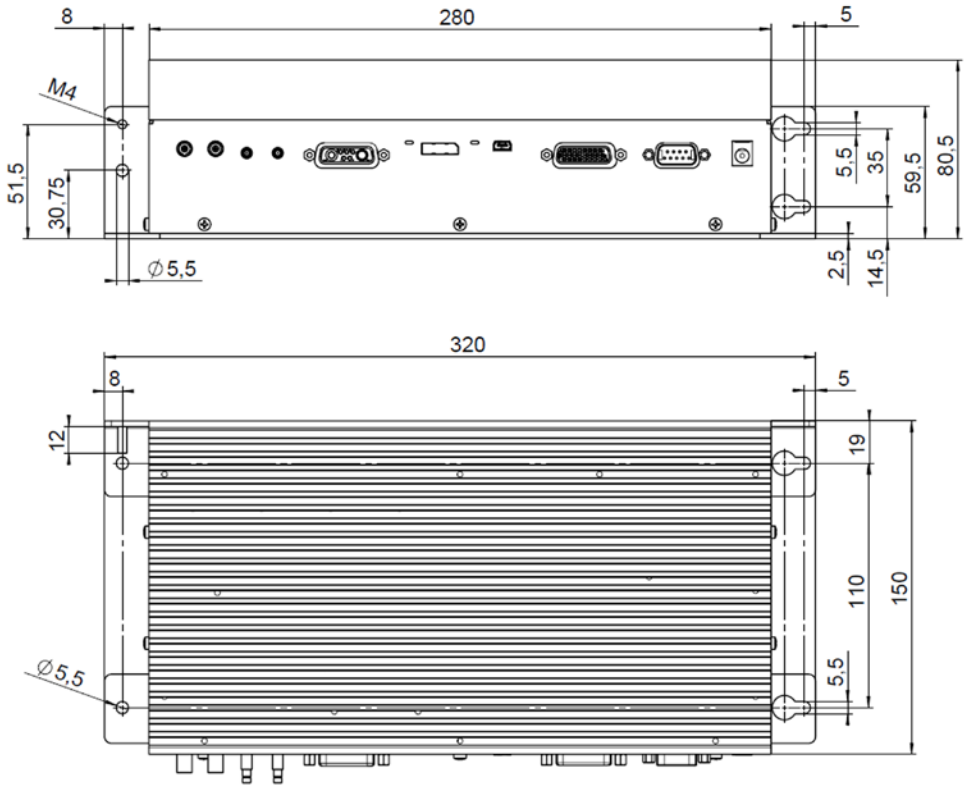


Figure 23: E-709.CHG dimensions

21.5.3 E-709.xR OEM Modules

Dimensions in millimeters, decimal places separated by commas.

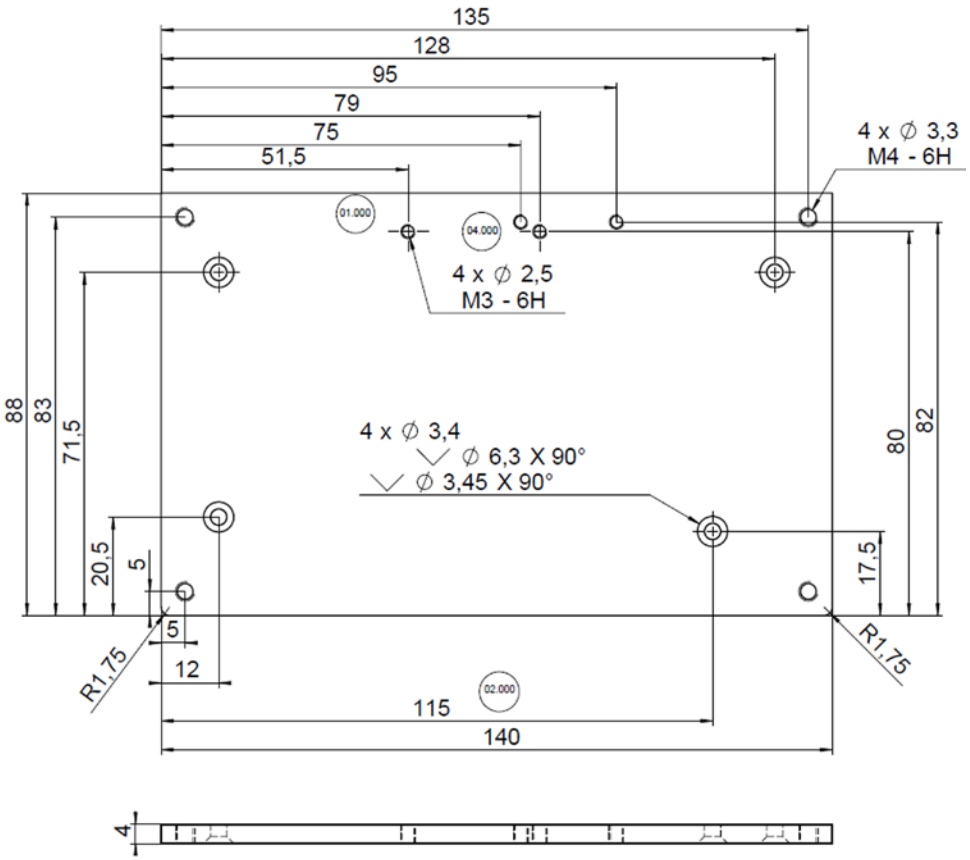


Figure 24: E-709.xR cooling plate dimensions

21.6 Pin Assignments

21.6.1 "PZT & Sensor" Socket of E-709.SRG, .PRG, .SR, .PR

Connector type: Sub-D 9 (f)

Pin	Signal
1	PZT Out (-30 V to +130 V)
2	Internal use
3	+15V, internal use
4	Sensor + INPUT (SGS or piezoresistive)
5	AGND
6	PGND
7	-15V, internal use
8	Sensor REF (SGS or piezoresistive)
9	Sensor – INPUT (SGS or piezoresistive)

21.6.2 "PZT & Sensor" Socket of E-709.CRG, .CR, .CHG

Connector type: Sub-D special connector 7W2 for 2 coax lines and 5 single pins

Pin	Signal	Function
Coax inner lines:		
A1	output	PZTOUT
A2	input	Sensor Probe
Standard pins:		
1	bidirectional	ID-Chip
2	GND	AGND
3	GND	AGND
4	GND	CGND
5	output	Sensor Target

Note:

Probe and Target are the connections of the capacitive sensor in the mechanics.

The PZTOUT line carries the piezo voltage for the actuator in the mechanics, up to 130 V.

21.6.3 "Sync" Sockets of E-709.CHG

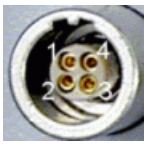
"**Sync In**" **Socket**

Connector type: LEMO EPG.00.304.NLN

Pin	Function
1	SYNCBASE_I_P (LVDS, 4.8 MHz)
2	SYNCBASE_I_N (LVDS, 4.8 MHz)
3	SYNC100_I_P (LVDS, 100 kHz)
4	SYNC100_I_N (LVDS, 100 kHz)

"**Sync Out**" **Socket**

Connector type: LEMO EPA.00.304.NLN



Pin	Function
1	SYNCBASE_O_P (LVDS, 4.8 MHz)
2	SYNCBASE_O_N (LVDS, 4.8 MHz)
3	SYNC100_O_P (LVDS, 100 kHz)
4	SYNC100_O_N (LVDS, 100 kHz)

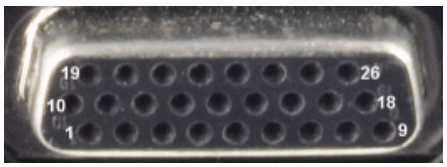
21.6.4 RS-232 Panel Plug of E-709.CHG

Connector type: Sub-D 9 pin (m)

Pin	Function
1	nc
2	RXD receive data
3	TXD send data
4	nc
5	DGND ground
6	nc
7	RTS Hardware handshake, output
8	CTS Hardware handshake, input
9	nc

21.6.5 I/O Socket of E-709.SRG, .SR, .PRG, .PR, .CRG, .CR

Connector type: HD-Sub-D 26 (f)

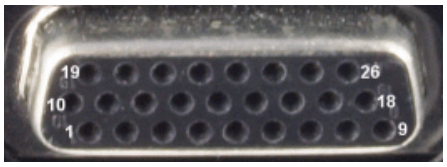


Pin	Signal	Function	Corresponding GCS Command
1	Servo On/Off	Servo monitor (LVTTL; on = low, off = high)	SVO?
2	ONT	On target state (LVTTL; on target = low, otherwise high)	ONT?
3	OVL	Overflow state (LVTTL; overflow = low; otherwise high)	OVF?
4	ENA	Enable device; NPN input, on = open (default), off = GND/low; i.e. connecting this pin to ground switches the E-709 off	-
5	Internal use	Do not connect	-
6	Internal use	Do not connect	-
7	Internal use	Do not connect	-
8	Internal use	Do not connect	-
9	RS232_GND	GND	-
10	Digital_IN_1	Digital input 1, can be configured for triggerings tasks (LVTTL, active high)	CTI, TRI, WGO, DIO?
11	Digital_OUT_1	Digital output 1, can be configured for triggering tasks (LVTTL, active high)	CTO, TWS
12	Digital_OUT_2	Digital output 2, can be configured for triggering tasks (LVTTL, active high; shares the TWS trigger table with Digital_OUT_1)	CTO, TWS
13	Internal use	Do not connect	-
14	DGND	GND	-
15	DGND	GND	-
16	Internal use	Reserved for future applications	-
17	RS232_RTS	RS-232 communication	-
18	RS232_CTS	RS-232 communication	-

Pin	Signal	Function	Corresponding GCS Command
19	Analog_Input	0 to 10 V Can be used to connect a control-signal source or an external sensor (handled by E-709 as input signal channel 2)	TAD?, TNS?, TSP?
20	Analog_Output	0 to 10 V Can be used to monitor the axis position or for controlling an external driver (handled by E-709 as output signal channel 2)	VOL?
21	Amplifier_Out_ Monitor	Monitor output of the piezo output voltage present on the "PZT & Sensor" socket, -0.3 to 1.3 V (piezo output voltage divided by 100; the piezo output voltage itself is handled by E-709 as output signal channel 1)	VOL? for the piezo output voltage
22	AGND	GND	-
23	Internal use	Reserved for future applications	-
24	Internal use	Reserved for future applications	-
25	RS232_RX	RS-232 communication	-
26	RS232_TX	RS-232 communication	-

21.6.6 I/O Socket of E-709.CHG

Connector type: HD-Sub-D 26 (f)



Pin	Signal	Function	Corresponding GCS Command
1	Servo On/Off	Servo monitor (LVTTL; on = low, off = high)	SVO?
2	ONT	On target state (LVTTL; on target = low, otherwise high)	ONT?
3	OVL	Overflow state (LVTTL; overflow = low; otherwise high)	OVF?

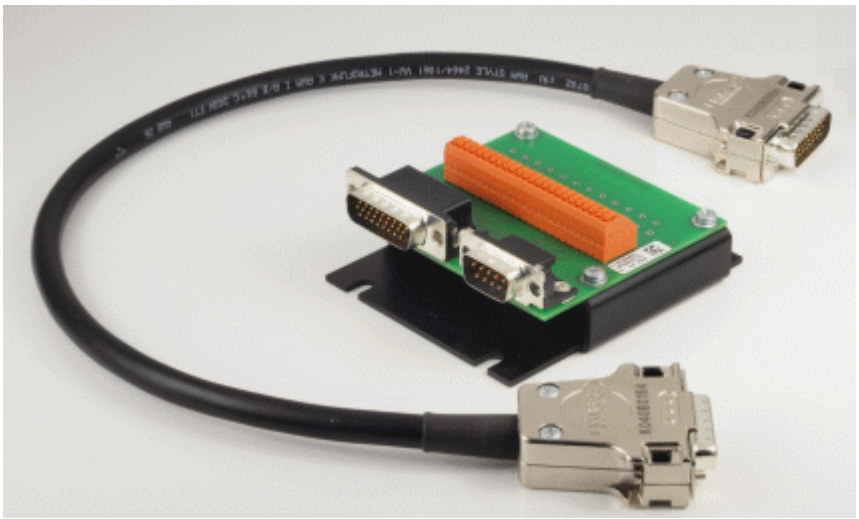
Pin	Signal	Function	Corresponding GCS Command
4	ENA	Enable device; NPN input, on = open (default), off = GND/low; i.e. connecting this pin to ground switches the E-709 off	-
5	Internal use	Do not connect	-
6	Internal use	Do not connect	-
7	Internal use	Do not connect	-
8	Internal use	Do not connect	-
9	GND	GND	-
10	Digital_IN_1	Digital input 1, can be configured for triggerings tasks (LVTTL, active high)	CTI, TRI, WGO, DIO?
11	Digital_OUT_1	Digital output 1, can be configured for triggering tasks (LVTTL, active high)	CTO, TWS
12	Digital_OUT_2	Digital output 2, can be configured for triggering tasks (LVTTL, active high; shares the TWS trigger table with Digital_OUT_1)	CTO, TWS
13	Internal use	Do not connect	-
14	GND	GND	-
15	GND	GND	-
16	Internal use	Reserved for future applications	-
17	nc	Not connected	-
18	nc	Not connected	-
19	Analog_Input	0 to 10 V Can be used to connect a control-signal source or an external sensor (handled by E-709 as input signal channel 2) Also available via the "Analog In" SMB socket on the front panel (p. 19). **Either** connect the signal via the SMB socket **or** via pin 19 of the "I/O" socket. Do not connect signals to both lines.	TAD?, TNS?, TSP?
20	Analog_Output	0 to 10 V Can be used to monitor the axis position or for controlling an external driver (handled by E-709 as output signal channel 2) Also available via the "Monitor Out" SMB socket on the front panel (p. 19).	VOL?

Pin	Signal	Function	Corresponding GCS Command
21	V-Monitor	Monitor output of the piezo output voltage present on the "PZT & Sensor" socket, -0.3 to 1.3 V (piezo output voltage divided by 100; the piezo output voltage itself is handled by E-709 as output signal channel 1)	VOL? for the piezo output voltage
22	GND	GND	-
23	Internal use	Reserved for future applications	-
24	Internal use	Reserved for future applications	-
25	nc	Not connected	-
26	nc	Not connected	-

21.6.7 E-709.01 Adapter for "I/O" Socket

The E-709.01 adapter splits the lines of the "I/O" HD-Sub-D 26 (f) socket on the E-709 up into:

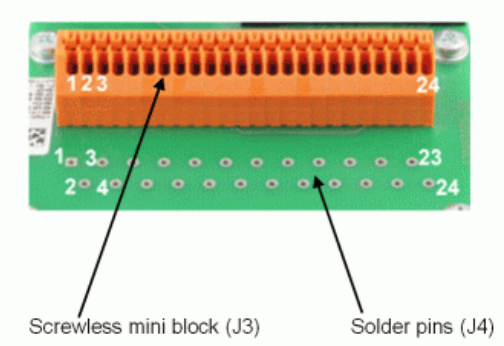
- Screwless mini block
- Solder pins
- RS-232 interface (Sub-D 9 (m))



The K040B0164 cable (1:1, 0.5 m) for connection to the E-709 "I/O" socket is included.

Signals on Screwless Mini Block and Solder Pins

Screwless mini block and solder pins carry the same signals (connected in parallel).



Pin on J3 and J4	Signal	Function
1	GND	GND
2	Digital_IN_1	Corresponds to pin 10 of "I/O" socket on E-709
3	Digital_OUT_1	Corresponds to pin 11 of "I/O" socket on E-709
4	Digital_OUT_2	Corresponds to pin 12 of "I/O" socket on E-709
5	ENA	Corresponds to pin 4 of "I/O" socket on E-709
6	Internal use	Do not connect
7	Internal use	Do not connect
8	Internal use	Do not connect
9	Internal use	Do not connect
10	AGND	GND
11	Analog_Input	Corresponds to pin 19 of "I/O" socket on E-709
12	Servo On/Off	Corresponds to pin 1 of "I/O" socket on E-709
13	ONT	Corresponds to pin 2 of "I/O" socket on E-709
14	SGS_Sensor_Monitor	Corresponds to pin 20 of "I/O" socket on E-709
15	OVL	Corresponds to pin 3 of "I/O" socket on E-709
16	Amplifier_Out_Monitor	Corresponds to pin 21 of "I/O" socket on the E-709
17	Internal use	Reserved for future applications
18	Internal use	Reserved for future applications
19	Internal use	Reserved for future applications
20	GND	-
21	GND	-
22	n.c.	-

Pin on J3 and J4	Signal	Function
23	GND	-
24	GND	-

Depending on your E-709 model, see p. 260 or p. 261 for pinout of the "I/O" HD-Sub-D 26 (f) socket on the E-709.

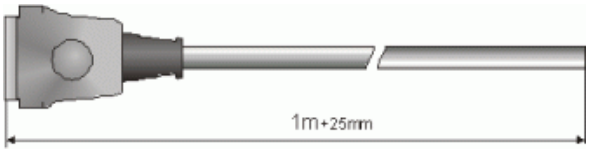
RS-232 Interface

Connector type: Sub-D 9 pin (m)

Pin	Function
1	nc
2	RXD receive data
3	TXD send data
4	nc
5	DGND ground
6	nc
7	RTS Hardware handshake, output
8	CTS Hardware handshake, input
9	nc

21.6.8 E-709.02 Adapter Cable for "I/O" Socket

HD-Sub-D 26 (m) to open leads



Depending on your E-709 model, see p. 260 or p. 261 for pinout of the "I/O" HD-Sub-D 26 (f) socket on the E-709.

Pin	Wire Color	Pin	Wire Color
1	black	14	white-orange
2	brown	15	white-yellow
3	red	16	white-green
4	orange	17	white-blue
5	yellow	18	white-purple
6	green	19	white-gray

Pin	Wire Color	Pin	Wire Color
7	blue	20	brown-black
8	purple	21	brown-red
9	gray	22	brown-orange
10	white	23	brown-yellow
11	white-black	24	brown-green
12	white-brown	25	brown-blue
13	white-red	26	brown-purple

Shield connected to connector housing

21.6.9 24 V DC Socket

Connector type: barrel connector

Pin	Function
Center	+24 VDC

21.7 Operating Limits

In order to achieve minimum distortion of the output waveform, it is important to ensure that the amplitude of higher-frequency control input is reduced in proportion to the fall-off of the output voltage at these frequencies. For exact information on maximum operating frequency with a given piezo load (capacitance), refer to the individual operating limit graphs in the figure below.

Note that the operating limits of a given piezo amplifier depends on the amplifier power, the amplifier design, and, of course the capacitance of the piezo actuator. The capacitance of piezo ceramics changes significantly with amplitude, temperature, and load-up to approximately 200% of the unloaded, small-signal capacitance at room temperature.

The following equations describe the relationship between (reactive) drive power, actuator capacitance, operating frequency and drive voltage.

The average power that a piezo amplifier has to be able to provide for sinusoidal operation is given by:

$Pa \approx C \cdot Umax \cdot Up\text{-}p \cdot f$

Peak power for sinusoidal operation is:

$P_{max} \approx \pi \cdot C \cdot U_{max} \cdot U_{p-p} \cdot f$

Where:

Pa = average power [W]
Pmax = peak power [W]
C = PZT actuator capacitance [farad (As/v)]
f = operating frequency [Hz]
Umax = nominal voltage of the amplifier [V]
Up-p = peak-peak drive voltage [V]

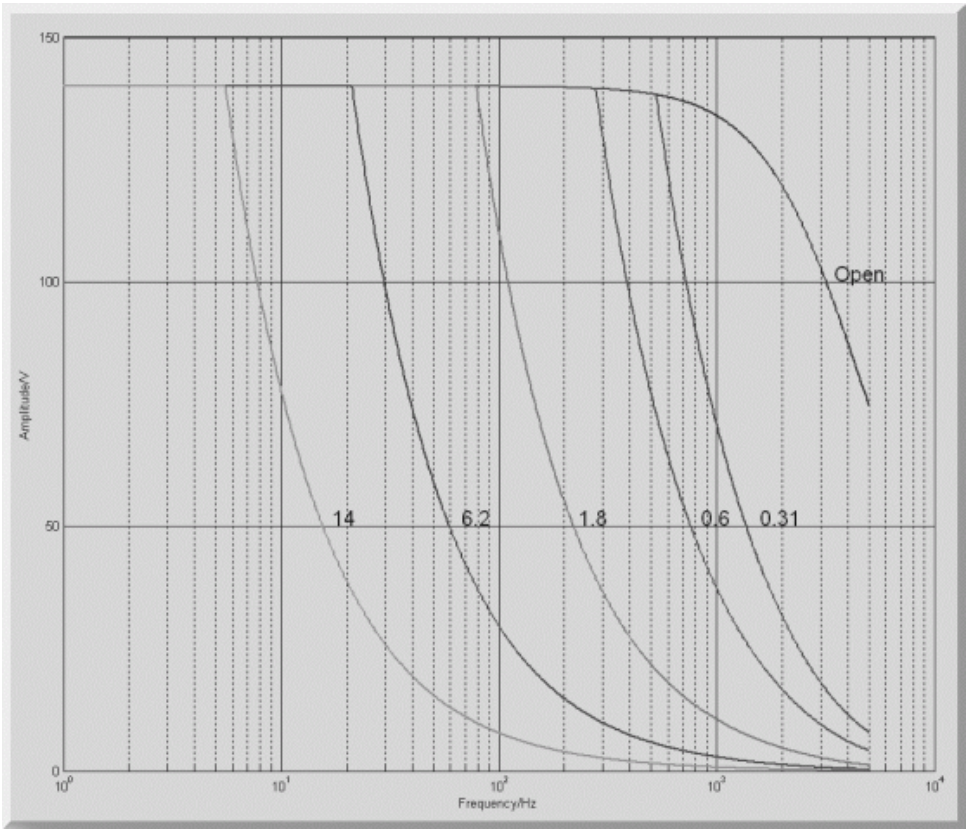


Figure 25: E-709.xR and .xRG operating limits with various piezo loads. Capacitance values in µF.

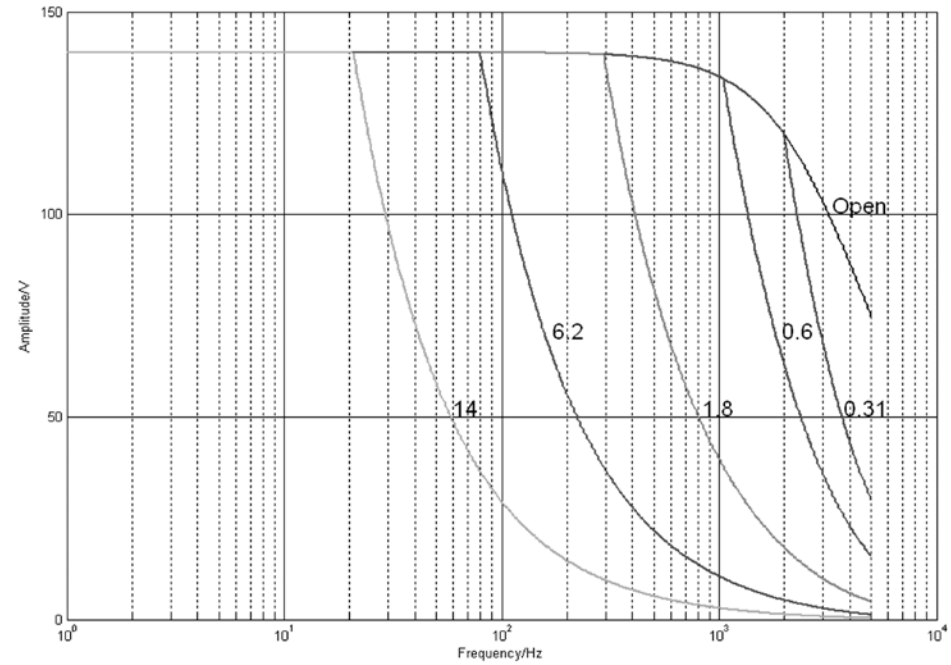


Figure 26: E-709.CHG operating limits with various piezo loads. Capacitance values in µF.