Piezo · Nano · Positioning

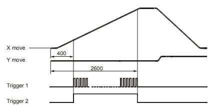
PZ 80E User Manual

E-710 Digital PZT Controller

Release: 5.3.9 Date: 2007-01-12

This document describes the following product(s):

■ E-710.3CD	Digital Piezo Controller, 3 Axes, Sub-D Connector
■ E-710.A3D	Digital Piezo Controller, 3 Axes, Sub-D Connector, Analog Input
■ E-710.APD	Digital Piezo Controller, 3 Axes, Sub-D Connector, Analog Input, PIO Interface
■ E-710.APS(0)	Digital Piezo Controller, 3 Axes, Sub-D Connector, Analog Input, PIO Interface, Digital Signal Transmission
■ E-710.4CD	Digital Piezo Controller, 4 Axes, Sub-D Connectors
■ E-710.4CL	Digital Piezo Controller, 4 Axes, Lemo Connectors
■ E-710.C4D	Digital Piezo Controller, 4 Axes, Sub-D Connectors
■ E-710.P3D	Digital Piezo Controller, 4 Axes, PIO Interface and Sub-D Connectors
■ E-710.P4D	Digital Piezo Controller, 4 Axes, PIO Interface and Sub-D Connectors
■ E-710.P4L	Digital Piezo Controller, 4 Axes, PIO Interface and Lemo Connectors
■ E-710.6CD	Digital Piezo Controller, 6 Axes, Sub-D Connectors
■ E-710.6SD(0)	Digital Piezo Controller, 6 Axes, Sub-D Connectors, Digital Signal Transmission



Auf der Römerstr. 1 · 76228 Karlsruhe, Germany
Tel. +49-721-4846-0 · Fax: +49-721-4846-299
info@pi.ws · www.pi.ws

Table of Contents

1 Introduction 7
1.1 Product Description 7
1.2 Prescribed Use 7
1.3 Safety and Warning Notices 8
1.4 Model Survey 9
1.5 Software Interfaces 10
1.6 Unpacking 12
1.7 Terminology 12

2 Starting Operation 13
2.1 Connecting Controller & Piezo Stage(s) 13
2.2 Installing Software on Host Computer 14
2.3 Connecting Controller & Host Computer 14
2.4 Controller Power On 15
2.5 Start Software on Host Computer 15
2.6 Set Servo-Loop State 16
2.7 Perform AutoZero 16
2.8 First Measurements and Motions 17
2.9 Set Operating Modes 17
2.10 Adjustments for Load Changes 18
2.11 Replacement procedure for stages with ID-Chip 19

3 Functionality and Block Diagrams 21
3.1 Axis and Channel Definitions 21
3.2 E-710 System Overview 21
3.3 Sensor Processing 22
3.4 Digital Servo-Controller 25
3.5 Drive Unit 26
3.6 Calculation of Target Positions 26

4 Hardware Basics 28
4.1 Controller Timing Basics 28
4.2 Piezo Dynamics Basics 28
4.3 Power Requirements and Open-Loop Frequency Response 28

5 Communication 30
5.1 RS-232 30
5.2 IEEE 488 30
5.3 Digital Trigger Signals (Digital I/O Lines) 31

5.4 Analog Input ... 32
5.5 Synchronization of Multiple E-710s ... 32
5.6 Optional PIO Interface (3- & 4-Axis Versions Only) ... 32
5.7 Communication Problems ... 33

6 System Flags and Settings ... 34
6.1 State and Error Flags ... 34
6.2 Axis Properties ... 35

7 Using the Analog Input ... 36
7.1 Basics ... 36
7.2 Parameters ... 36
7.3 Examples ... 38

8 Auto Focussing ... 44
8.1 Identifying the Focus Position ... 44
8.2 Holding the Focus Position ... 46

9 External-Axis Flatness Compensation ... 48
9.1 Concept ... 48
9.2 Implementation ... 48
9.3 Example ... 48

10 Wave Generator ... 50
10.1 Symmetrical Scanning Waveform Definition Example ... 51
10.2 Unidirectional Scanning Waveform Example ... 53
10.3 Programmable Output Trigger Signals ... 57
10.4 Input Trigger Signals ... 58
10.5 Data Recording ... 59
10.6 Wave Parameters and Commands ... 60

11 Dynamic Digital Linearization (DDL) ... 61
11.1 Introduction ... 61
11.2 Working Principle ... 61
11.3 Using DDL ... 65
11.4 Parameters ... 69
11.5 Advanced Tuning ... 70

12 Configuration Examples ... 72
12.1 No Servo-Control, Sensor Evaluation Only ... 72
12.2 Servo-Control with 1 to 6 Independent Channels ... 72
12.3 Servo-Control with 1 to 8 Coupled Channels ... 73

13 Servo-Controller Dynamic Calibration ... 74
13.1 Overview ... 74
13.2 Adjustment Procedures ... 75
13.3 Setting Servo Parameters ... 77
13.4 AutoZero ... 79

14 Command Format ... 82
14.1 Native ASCII Commands ... 82
14.2 PIO Commands ... 84

15 Command Survey ... 87
15.1 Native ASCII Commands (by type) ... 87
15.2 PIO Commands ... 90

16 Native Command Reference (alphabetical) ... 91

17 Technical Data ... 148
17.1 Line Power and Fuses ... 149
17.2 3- and 4-Axis Model-Specific Hardware Details ... 149
17.3 6-Axis Model-Specific Hardware Details ... 159
17.4 ANALOG IN Connector ... 163
17.5 SYNC Connector and Synchronization Cable ... 164
17.6 IEEE (GPIB) Interface ... 165
17.7 RS-232 Interface ... 165

18 PIO Interface ... 166
18.1 Guidelines for PC Cards Communicating with the E-710 PIO ... 166
18.2 PIO Line-Timing Considerations ... 167

19 Terminology Table ... 170

Declaration of Conformity

according to ISO / IEC Guide 22 and EN 45014

Manufacturer:	**Physik Instrumente (PI) GmbH & Co. KG**	
Manufacturer´s Address:	Auf der Römerstrasse 1 D-76228 Karlsruhe, Germany	

The manufacturer hereby declares that the product

Product Name:	**Digital Piezo Controller**
Model Numbers:	**E-710**
Product Options:	**all model types**

conforms to the following EMC Standards and normative documents:

Electromagnetic Emission:	EN 61000-6-3, EN 55011
Electromagnetic Immunity:	EN 61000-6-1
Safety (Low Voltage Directive) :	EN 61010-1

August 24, 2004
Karlsruhe, Germany

[illegible]

Dr. Karl Spanner
President

First printing 2007-01-12
Document Number PZ 80E, Release 5.3.9
E-710_User_PZ80E.doc

Subject to change without notice. This manual is superseded by any new release. The newest release is available for download at www.pi.ws.

About this Document

Users of this Manual

This manual is designed to help the reader to install and operate the E-710 Digital Piezo Controller. It assumes that the reader has a fundamental understanding of basic servo systems, as well as motion control concepts and applicable safety procedures.
The manual describes the physical specifications and dimensions of the E-710 Digital Piezo Controller as well as the installation procedures which are required to put the associated motion system into operation.

Conventions

The notes 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.

DANGER

Indicates the presence of high voltage (> 50 V). Calls attention to a procedure, practice or condition which, if not correctly performed or adhered to, could result in injury or death.

CAUTION

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

NOTE

Provides additional information or application hints.

Related Documents

The software tools which might be mentioned within this documentation are described in their own manuals. All documents are available on the respective product CD. Current releases can be downloaded from the PI Website as PDF files (www.pi.ws), obtained from your Physik Instrumente sales engineer or from info@pi.ws

NanoCapture_SM071E
PZTControl_PZ145E
E710_GCSLabVIEW_PZ141E
E-710_GCSDLL_PZ147E
SM069E

1 Introduction

1.1 Product Description

E-710 Digital Piezo Controllers integrate up to 8 independent power amplifiers for low-voltage PZTs (6 W, -20 to 110 V) and conditioning electronics for up to 6 capacitive position sensors.

In contrast to analog multi-axis controllers, digital controllers like the E-710 have the advantage that sensor and output channels can be combined in a flexibly programmable internal coordinate transformation. This means that the sensors and actuator geometry is independent of the logical coordinate system used.

All positioning commands are in standard units such as micrometers, microradians, etc., for easy programming. In addition all control settings can be saved in non-volatile memory and user accessible.

- For Piezo Nanopositioners with Capacitive Feedback Sensors
- 3-, 4- & 6-Axis Versions
- 32-Bit Digital Filters
- Hardware and Firmware Linearization
- Coordinate Transformation for Parallel Kinematics / Parallel Metrology Systems
- Fully Programmable Low-Pass Sensor Filter
- Fully Programmable Notch Filters on Servo-Loop Output
- Automatic Zero Adjust after Every Power-Up
- Built-In Wave Generators, Ideal for Multi-Axis Scanning Applications
- ID Chip Support for Automatic Calibration
- GCS (General Command Set) Compatible
- Optional Dynamic Digital Linearization (Firmware Option) Improves Scanning Linearity
- Optional High-Speed Parallel I/O or DSP Interface
- Optional Analog Input for an additional external sensor signal or level (e.g. position-sensor or a focus signal) or analog control input
- Optional Digital Signal Transmission (DST) for applications where the distance between the mechanics and electronics is greater than 3 m

1.2 Prescribed Use

Based on their design and realization, E-710 digital piezo controllers are intended to drive capacitive loads, in the present case, piezoceramic actuators. E-710s must not be used for applications other than stated in this manual, especially not for driving ohmic (resistive) or inductive loads. E-710s can be operated in closed-loop mode using capacitive position sensors. Appropriate sensors are provided by PI and integrated in the mechanics according to the mechanics product specifications. Other sensors may be used with the additional analog input of some controller models according to the specifications of this input.

Safe operation of E-710s is provided for under normal ambient conditions in terms of DIN EN 61010.

1.3 Safety and Warning Notices

Carefully read also the documentation of the included software components and of the mechanics used.

Ignoring the warning notices in the instructions can cause bodily injury of the user or damage to equipment or loss of warranty.

CAUTION

Install and operate the E-710 digital piezo controller only when you have read the operating instruction. Keep the instruction readily available close to the device in a safe place. When the instruction is lost or has become unusable, ask the manufacturer for a new copy. Add all information given by the manufacturer to the instruction, e.g. supplements or Technical Notes.

DANGER

The E-710 digital piezo controller does not contain any user-serviceable parts. Never re-assemble the device. Hazardous voltage can be present inside the housing.

DANGER—HAZARDOUS VOLTAGE

The amplifiers used by the E-710 are high-voltage amplifiers capable of generating high output currents. They may cause serious or even lethal injury if used improperly. Working with high-voltage amplifiers requires adequately trained operating personnel. Strictly observe the following:

→ Never touch any part that might be connected to the high-voltage output.

→ Do not touch the pins of the Sub-D mix connectors / the LEMO connectors when the E-710 is turned on. The output values depend on the last commanded targets (positions or voltages), even if you have quit the terminal or the program from which the targets were commanded—up to 120 V can be present on the Sub-D mix connectors or on the LEMO connectors.

CAUTION

Most piezo actuators that can be connected to the E-710 can be destroyed by uncontrolled oscillation near the mechanical resonant frequency. If you observe resonance while configuring your system, switch off power to the actuators concerned immediately and check the settings and servo-control parameters.

CAUTION

If you change the supply power voltage setting from 115 V to 230 V or vice-versa, you must also replace the line fuses with fuses appropriate for the new voltage. See “Line Power and Fuses” p. 149.

WARNING

All motion of the connected mechanical stages is software controlled, and software may fail.

1.4 Model Survey

1.4.1 Hardware Models

Several 3-, 4- and 6-axis standard versions are available (versions with identical number of axes are highlighted with the same color). If you received this manual with a controller model not listed, see any accompanying Technical Notes.

Model	Axis / Channel Support			PZT/Sensor Connector Type*	PIO Interface	Analog Input	Digital Signal Transmission (DST)	Hardware-Specific Details
	Logical Axes	Sensors	PZTs					
E-710.3CD	3	3	4	1 x 22-pin sub-D special	no	no	no	p. 152
E-710.A3D					no	yes	no	p. 152
E-710.APD					yes	yes	no	p. 152
E-710.APS				1 x 22-pin sub-D special on remote sensor interface box	yes	yes	yes, cable included (16 m)	p. 152
E-710.APS0					yes	yes	yes, but cable must be ordered separately**	p. 152
E-710.C4D	4	4	4	1 x 22-pin sub-D special 1 x 5-pin sub-D special	no	no	no	p. 151
E-710.4CD				4 x 5-pin sub-D special	no			p. 150
E-710.4CL				3 x (4 x) Lemo	no			p. 149
E-710.P3D				1 x 22-pin sub-D special 1 x 5-pin sub-D special	yes			p. 151
E-710.P4D				4 x 5-pin sub-D special	yes			p. 150
E-710.P4L				3 x (4 x) Lemo	yes			p. 149
E-710.6CD	6	6	8	2 x 22-pin sub-D special	no	yes	no	p. 159
E-710.6SD				2 x 22-pin sub-D special on remote sensor interface box		yes, on remote sensor interface box	yes, cable included (16 m)	p. 160
E-710.6SD0							yes, but cable must be ordered separately***	p. 160

* See page 150 for adapters available.

** Various cable lengths available to match the customer requirements, order# E-710.APSx (length increases in 2 m-steps)

*** Various cable lengths available to match the customer requirements, order# E-710.DSTx (length increases in 2 m-steps)

The Dynamic Digital Linearization (DDL) feature is standard with the 6-axis versions. On the other versions, DDL is available as an option, where it can also be installed after purchase and without opening the device.

For further device specifications see Section 17 on p. 148.

1.4.2 Firmware Version Numbers

The firmware version can be read out with the GI command (p. 113). A firmware version can be replaced by a higher one, as long as the digits before the decimal point are the same. Firmware with different whole-number version numbers may differ in terms of the hardware on which they are designed to run and may thus be incompatible.

1.5 Software Interfaces

It is possible to operate the E-710 with two command sets: the native ASCII command and the PI General Command Set (GCS). Note that all examples given in this User Manual use native commands. For the description of the GCS commands and the associated windows DLL, see the E-710_GCS_DLL Manual.

1.5.1 Native Command Set

The native ASCII command set is understood by the E-710 firmware. It can be used with virtually any terminal-emulator software (such as *WinTerm32* on the E-710 product CD). In addition, the *NanoCapture*™ graphical user interface (see Section 1.3.2) provides a terminal for the native commands.

Native commands are always two-letter combinations. The syntax of the native commands is described in Section 14.1 beginning on p. 82, the command reference in alphabetical order is to be found in Section 16 beginning on p. 91.

1.5.2 GCS Command Set

The GCS is the PI standard command set. This command set ensures the compatibility between different controllers. It provides more sophisticated access to the controller functionality, supported by a Windows DLL which translates the GCS commands to the native commands understood by the E-710 firmware. GCS commands are three-letter combinations.

The Windows DLL which supports the GCS is provided for programmers wishing to access the E-710 from Windows programs without concerning themselves with the details of native ASCII command generation or interface-hardware access. All E-710 functions except of the PIO interface communication are accessible via the DLL, either as DLL functions in their own right, or using the special DLL function which sends a native command over the interface.

NOTES

Do not mix up the GCS command set and the native command set! GCS move commands do not work properly anymore after the position was changed by native commands.

Due to the emulation of the native command set, the execution of some DLL motion functions is noticeably slower than that of the appropriate native commands. The DLL provides special motion functions for the case that your application requires quickest possible response to motion commands.

The DLL can be used with models with the PIO feature, but does not include PIO support.

For details see the E-710_GCS_DLL Manual.

In addition, a number of different software packages to facilitate controlling the E-710 are included on the CD. They all access the interface to the E-710 using the DLL and are therefore GCS based:

- *NanoCapture™* A powerful graphical user interface which gives easy access to step response measurement or waveform motion, but also provides access the advanced functionality of the controller. Users do not have to know any commands to work with *NanoCapture™*. Included are:
  - Open- and closed-loop graphic motion display
  - Bode plots
  - Dynamics tuning
  - Controller configuration, with all parameter types
  - Auto-zero adjustment
  - Wave editor
  - ASCII command entry (both GCS and native)
- *PZTControl* A graphical user interface which is recommended for users who are already familiar with the GCS command set or for programmers which want to use the Windows DLL. *PZTControl* makes it possible to test the DLL functions in a convenient way. Included are:
  - GCS ASCII command entry
  - Control and display of positioning tasks
  - Interactive access to important GCS commands
  - Firmware update via interface
- LabView VIs For programmers using the LabView programming environment. Includes functions for communications handling, complete command libraries and high-level functions.

The DLL, *NanoCapture™*, the LabView VIs and *PZTControl* are described in separate manuals, which can be found on the E-710 product CD. See also the examples in the \Samples directory of the product CD.

1.5.3 Updates

Updated releases of software and manuals are available for download at www.pi.ws. While the manuals are freely accessible, you need a password for software download. This password is provided in the E-710 Releasenews PDF file which is on the E-710 CD in the \Manuals directory.

To download the latest software (complete CD mirror) from the PI Website, proceed as follows:

1. On the www.pi.ws front page, click on *Download/Support* in the *Service* section on the left
2. On the *Download/Support* page, click on *Manuals and Software*
3. On the *PI Download Server* page, enter the *Username* and the *Password* which is provided in the E-710 Releasenews xxxxx.pdf on the E-710 CD and click on *Login*
4. Click on *Download* in the navigation bar across the top
5. Click on the *E Piezo Drivers & Nanopositioning controllers* category
6. Click on *E-710*

7. Click on *Release* (if you click on *Documents* you will get the latest manuals)
8. Click the download button below the latest CD-Mirror

1.6 Unpacking

See packing list for the exact contents of your order. It may include:

- E-710 controller
- Communication cable for RS-232 interface (order# C-815.34)
- CD which contains
  - *NanoCapture™*
  - *PZTControl*
  - Windows DLL
  - LabView drivers
  - Sample programs
  - *WinTerm32* terminal software
  - Hardware and software documentation (PDF files)
- E-710 User Manual (this document, PZ 80E)
- DDL Licence document E-710.scn (with 3- and 4-axis versions optional, with 6-axis versions, DDL is activated as standard and no special licence is required)
- Power Cord (order# 3763)
- PIO interface versions only:

  Communication cable with Mini D Ribbon (MDR) 50-pin connector 50 and 100-pin SCSI-2-type connector (order# K040B0003)
- Versions with Analog Input only:

  Analog input cable with Lemo 4-pin and BNC connectors (order# P892B0006)
- Versions with Digital Signal Transmission (DST) only:

  Remote sensor interface box

  Connecting cable for controller and sensor interface box
  Cable must be ordered separately with E-710.APS0 and E-710.6SD0, see Section 1.4.1 on p. 9 for ordering information
- 6-axis versions only:

  26-conductor MDR connector (customizable, for testing the DIGITAL I/O interface; order# 4236)

1.7 Terminology

To facilitate comprehension of this rather complex product, a number of terminology changes have been made. A listing of new and old terms is provided on p. 170 at the end of this manual to aid in understanding any older E-710 or *NanoCapture™* documentation you may be confronted with.

2 Starting Operation

If PI has been given sufficient information about your application and/or the E-710 is ordered together with the PZT positioners, the system will be calibrated at the factory. This section provides the information you need to put such a system into operation.

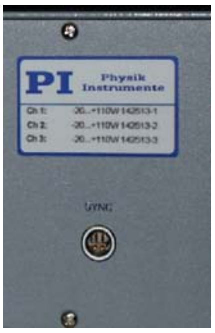


Fig. 1: Channel number and assigned piezo stage serial numbers

2.1 Connecting Controller & Piezo Stage(s)

A label on the rear panel of the controller indicates which PZT stage was assigned to which controller channel(s) during calibration (see example in Fig. 1). Be sure to respect these assignments when connecting the stage(s) to the controller.

The controller is available with different kinds of connectors (depending on model, see Section 1.5 on p. 9). If the connectors do not match those on the PZT stages, an appropriate adapter must be used. The front and rear panels of the different 3- and 4-axis controller models are illustrated in Section 17.2, p. 149 ff. The 6-axis version has all cable connections on the rear panel—see Fig. 66 on page 159.

2.1.1 Controller Types E-710.4CL and E-710.P4L

These controller types are illustrated in Fig. 58 on p. 149. The cables from the PZT stages are labeled with the type of cable, and, for multi-axis stages, with channel numbers.

Piezo Stage Connector Label	Controller Connector Label	
“Axis1”* or “PZT1”*	CH1	PZT
“P1”**	CH1	PROBE
“T1”**	CH1	TARGET
“Axis2”* or “PZT2”*	CH2	PZT
“P2”*	CH2	PROBE
“T2”*	CH2	TARGET
etc. to max. number of channels		

* Connectors on one-axis stages are labeled without channel number indication, e.g. “P” instead of “P1”.

The label on the E-710 rear panel shows which stage or stage channel has been assigned to (calibrated with) which controller channel (Fig. 1).

2.1.2 Controller Types E-710.4CD and E-710.P4D

These controller types are illustrated in Fig. 59, p. 150. The stages must be connected as indicated by the label on the rear of the controller.

2.1.3 Controller Types E-710.3CD, E-710.A3D, E-710.APD, E-710.C4D, E-710.P3D

These controller types are illustrated in Fig. 61 and Fig. 60, p. 152 and p. 151. With these models, the connector types determine the proper connections uniquely.

2.1.4 Controller Type E-710.6CD

This controller type is illustrated in Fig. 66, p. 159. It has all connectors on the rear panel. The stages must be connected to the channels with which they were calibrated (see the label on the back of the controller).

2.1.5 Controller Types E-710.APS(0) and E-710.6SD(0)

E-710.APS and E-710.6SD comprise a digital piezo controller and a separate interface box for digital sensor-signal transmission for applications where the distance between the mechanics and electronics is greater than 3 m. With these models, the piezo stages must be connected to the sensor interface box which is in turn connected to the controller.

E-710.APS (illustrated in Section 17.2.5 on p. 152)

The PZT&SENSOR socket on the sensor interface box is used to connect the piezo stage (carries both piezo voltage and sensor lines). The pinout is described on p. 156 (Sub-D Mix Connector with 3 coax lines and 22 single pins).
The connection between controller and sensor interface box is done via the PZT and SENSOR sockets and a connecting cable E-710.APSx (x encodes the cable length, see Fig. 64 on p. 154). Via this connection, also the supply power for the sensor interface box is provided by the controller.
The ground stud on the sensor interface box must be connected to a protective ground.

E-710.6SD (illustrated in Section 17.3.3 on p. 160):

The CH1/CH2/CH3 and CH4/CH5/CH6 sockets on the sensor interface box are used to connect the piezo stage (carry both piezo voltage and sensor lines). The pinout is described on p. 162 (2 Sub-D special sockets each with 3 coax lines and 22 pins).
The connection between controller and sensor interface box is done via the PIEZO and POWER / DATA sockets and a connecting cable E-710.DSTx (x encodes the cable length; see Fig. 74 on p. 162). Via this connection, also the supply power for the sensor interface box is provided by the controller.

2.2 Installing Software on Host Computer

To install the E-710 software on your host PC, proceed as follows:

1. Be sure to login as administrator and insert the E-710 CD in your host PC.
2. If the Setup Wizard does not open automatically, start it from the root directory of the CD with the icon.
3. Follow the on-screen instructions. You can choose between “typical” and “custom” installation. Typical components are LabView drivers, DLLs, *NanoCapture™*, *PZTControl*, *WinTerm32* and the manuals. “Typical” is recommended.

If your controller has a PIO interface (E-710.Pxx models only), and if that interface is also connected to the host computer, install the software you are going to use with the PIO (the E-710 software installed with the setup does not explicitly support the PIO, but will not interfere with its use. Sample programs that use the PIO are included on the CD in the \Samples\LabView and \Samples\C directories).

2.3 Connecting Controller & Host Computer

Two interface types are available for the ASCII command interface between the E-710 and the host computer: serial RS-232 and IEEE 488 (GPIB). If you want to use the IEEE 488 interface, a National Instruments GPIB card (or equivalent) must be installed on your host computer. The *NanoCapture*™ software works only with National Instruments GPIB cards or RS-232.

With the controller still powered down, connect the ASCII command interface cable (serial RS-232 or IEEE 488). The IEEE 488 is substantially faster than the RS-232 (the supplementary PIO interface can further increase the speed of operation).

If your E-710 has a PIO interface (Parallel Input/Output, p. 32) and you wish to use it, connect the PIO cable to the controller. The easiest way to interface to the PIO is to equip your host computer with an adequate PIO card (National Instruments PCI-96-DIO digital I/O card or equivalent). See page 90 for a discussion of the commands that can be sent over the PIO. The CD contains sample programs illustrating use of the PIO (in the \Samples\LabView and \Samples\C directories).

If you wish to use custom electronics with the PIO, you will need the PIO signal timing information starting on p. 166.

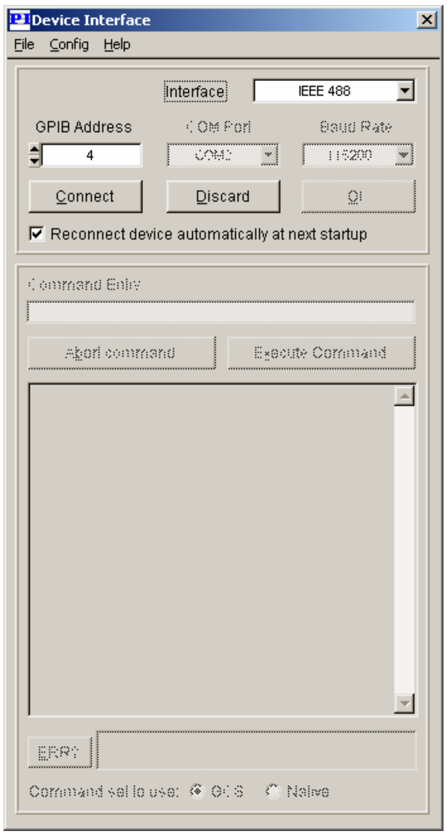
2.4 Controller Power On

1. Connect the power cable to the controller and to an appropriate power outlet.
2. Switch power on.
3. Wait until the power LED turns to green. This takes about 5 seconds.

2.5 Start Software on Host Computer

Starting the host software is exemplified using the *NanoCapture*™ software package because the E-710 is most conveniently controlled with *NanoCapture*™. To work with *NanoCapture*™, you do not need any knowledge of the E-710 command sets.

1. When *NanoCapture*™ is started for the first time, the *Device Interface* dialog will be displayed (Fig. 3).
   (Thereafter, the software can initialize the connection automatically with the last-used settings. These settings can be changed using the *Config* → *Device Interface* menu sequence.)
2. Select the type of *Interface*: RS-232 or IEEE 488 (GPIB).
3. If the RS-232 serial interface is used, select the desired baud rate and the host computer COM port to which the cable is connected. The software will automatically change the baud rate on both sides of the communication to the value specified.
   If the IEEE 488 (GPIB) interface is used, select the default controller Address of 4, unless your unit has been set to another address.
4. Press the *Connect* button. When the connection is successfully initialized, you will then be asked which of the axes supported by the controller are to be made available for use. When the connection to the controller is successfully established, the *OK* button and the Command Entry pane will no longer be grayed out.



Interface: IEEE 488
GPIB Address: 4
COM Port: COM1
Baud Rate: 115200
☑ Reconnect device automatically at next startup
Command Entry
Command set to use: ● GCS ○ Native

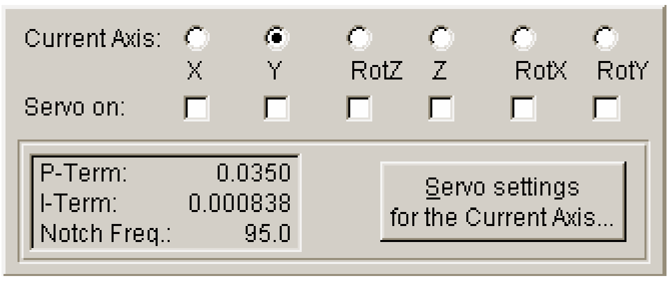
Fig. 2: NanoCapture™ Device Interface *window before connection*

5. Press the *OK* button (do not enter commands yet). Now *NanoCapture*™ starts to upload the current parameters from the controller. When the upload procedure is finished the *Device Interface* window will be closed automatically, and you can start working in the *NanoCapture*™ main window.
6. If the connection is not successful, see “Communication Problems,” p. 33 for troubleshooting the interface.

2.6 Set Servo-Loop State

Normally the servo loops for all logical axes are OFF when the controller is started for the first time. Set the servo-loop state according to your application. For wave generator output, for example, the servo-loop must be turned on.

In *NanoCapture*™, you can use the *Current Axis* pane of the *NanoCapture*™ main window (Fig. 4) to set the servo-loop state. The radio buttons in this pane permit selection of the axis that will be used by default for certain actions and measurements that *NanoCapture*™ can perform. All axes supported by the controller will have a radio button shown. Up to 6 axes can be supported, depending on the controller used. Under each axis’ radio button is a check box that can be used to turn that axis’ servo-loop on or off. If servo-control is disabled for the axis, the checkbox is dimmed.



P-Term: 0.0350
I-Term: 0.000838
Notch Freq.: 95.0

Fig. 3: Axis-select pane of the NanoCapture™ window

2.7 Perform AutoZero

For detailed explanation of AutoZero please refer to “AutoZero” on page 79. Note that AutoZero is not effective on non-linear axes and requires special parameter adjustment when the appropriate axis is connected to the Analog Input for target generation (see p. 80 for details).

2.7.1 Decide Whether Running AutoZero Is Necessary

- If the stage and controller are not well known, performing AutoZero is best.
- If the stage is being operated in the current environment for the first time, AutoZero is necessary.
- If only relative movement is important, performing AutoZero after every power up is recommended.

2.7.2 If AutoZero Is Necessary

With *NanoCapture*™, proceed as follows to carry out the AutoZero procedure:

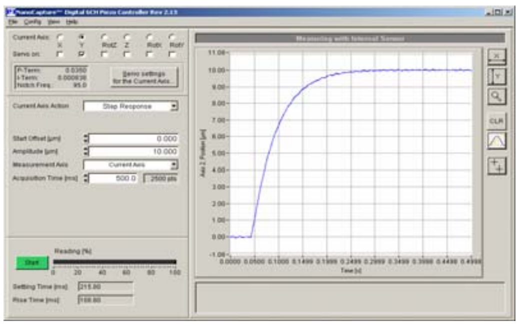
1. First open the *Device Interface* window (see Fig. 3) with the *Config* → *Device Interface* menu sequence to get the *Command Entry* pane.
2. In the *Device Interface* window send an AutoZero command by typing the appropriate AZ command or commands in the *Command Entry* pane. The process takes about 5 seconds. Note that AZ is not effective on non-linear axes.

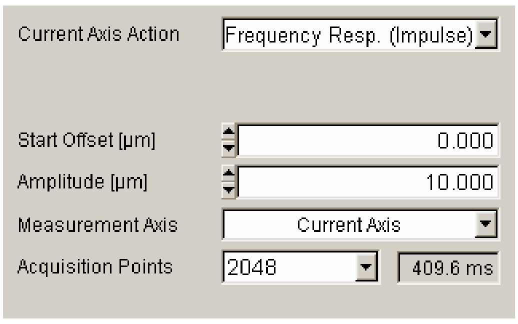
It is also possible to save the resulting offset value so that AutoZero does not have to be rerun at every power-up (see item 8 in the AutoZero procedure on p. 92 for details).

It is possible to perform an AutoZero automatically after every power up. See "AutoZero" on page 79 for full information.

2.8 First Measurements and Motions

Performing first actions is exemplified using *NanoCapture™* to measure the step response and frequency response of the system. Those actions are done in the *Current Axis Action* pane of the *NanoCapture™* main window (see Figures below).





Current Axis Action	Frequency Resp. (Impulse)
Start Offset [µm]	0.000
Amplitude [µm]	10.000
Measurement Axis	Current Axis
Acquisition Points	2048 · 409.6 ms

Fig. 4: NanoCapture™ *Current Axis Action Pane*

The measurement parameters that can be set are:

- *Start Offset* is the value to which the current axis will be commanded before the Step or Impulse is started.
- *Amplitude* is the command amplitude for the Step or Impulse.
- *Measurement Axis* is the axis with which the measurement will be made

The number of acquisition points (position values to record) and the acquisition time for the measurement are directly linked and depend on the Table Rate time base factor parameter (see *NanoCapture™* Manual). For *Step Response* you can set the *Acquisition Time* and for the *Frequency Response,* the number of *Acquisition Points*.

The Step or Impulse will be performed on the *Current Axis* in closed- or open-loop depending on the servo state of the *Current Axis*. The results will be displayed in the graphic report pane of the *NanoCapture™* main window.

For more information see the *NanoCapture™* manual.

2.9 Set Operating Modes

Operating modes are stored in memory and set separately for each axis. They can be changed using *NanoCapture™* (*Config → Device Parameter* Configuration menu sequence). The appropriate native command sequences (DP-DW compound commands) are described on p. 95 *ff*. and in the configuration examples on p. 71. Values stored in non-volatile memory are power-up defaults, so that the system can be used in the desired way immediately. Be aware that the non-volatile memory can only take a limited number of write cycles (a few thousand).

The following table shows how to select the desired operating mode by changing the values of the parameters.

Parameter / Operating Mode	Servo-Control Enable	Servo-Control Autostart	Auto Zero Autostart
Sensor, evaluation only	0	0	X
Servo disabled (open-loop)	0	0	X*
Servo enabled (closed-loop), Servo-Controller turned ON by SL command	1	0	X
Servo enabled (position controlled), Servo-Controller started automatically at power-up	1	1	X
Position control enabled, AutoZero upon power up	1	X	1

The symbol "X" means that the setting of the corresponding parameter can vary in the corresponding operating mode.

Sensor, evaluation only: The position values of the sensors can be read out, but are not used for servo-control. The values can be read by the TP command (p. 142) or over the PIO (if installed).

Open-loop, servo-controller disabled: An axis can be controlled by specifying the output voltage (VS command, Voltage Set, p. 146). Coordinate transformation is still applied, so the axis direction is respected. The sensor values are also fed through coordinate transformation but not used internally. If sensors are enabled, they can be read with the TP command.

Closed-loop, servo-controller enabled: With servo-control enabled, the servo-controller can be started in either of two ways:

- Manually, with the SL command (p. 137, or with *NanoCapture*™)
- By setting the *Servo Control Autostart* parameter to ON: then servo-control is turned ON upon power-up

With servo-control enabled and turned on, the MA (p. 123) or MR (p. 128) commands can be used to specify the desired position. The controller will move the stage to this position.

2.10 Adjustments for Load Changes

If the stages were calibrated with a load different from that to be used in the current application, it is necessary to re-determine the resonant frequency with the current load. When that is done, the notch filter frequency and the controller parameters (loop gain and time constant) have to be adjusted.

2.10.1 Determining Resonant Frequencies

Resonant frequency determination is a multi-step procedure. The first steps affect all channels, then the remaining steps must be run through once for each axis. See "Suppressing Mechanical Resonance with Notch Filters," p. 74 for details.

2.10.2 Setting Notch Filter Parameters

The notch filter frequency for each axis is usually adjusted to a value equal to the resonant frequency. See "Notch Filters," p. 75 for details.

2.10.3 Setting Servo-Control Parameters

Similarly, whenever the loading of one or more axis is changed, certain servo parameters must be adjusted. This is again a multi-step process involving preparation affecting all axis (p. 77) followed by steps that must be repeated for each axis. The loop-gain servo-control parameter must be readjusted (see p. 74). Normally the time constant value is calculated from the resonant frequency, but sometimes it might be reasonable to modify this value (p. 74) in order to achieve further optimization.

2.11 Replacement procedure for stages with ID-Chip

The PZT stage which is connected to the E-710 may contain an ID-chip. The following data is stored in the ID-chip:

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

When you are using a PZT stage with ID-chip together with the E-710, the PZT stage can be easily exchanged due to the functionality of the ID-chip.

Consider the following when replacing stages with ID-chip:

- Normally, when you replace a PZT 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 up of the system, the firmware reads the stage type and serial number stored in the ID-chip and compares this data to the data stored in the controller. If there is a new stage connected to the controller, the rest of the ID-chip data will be read and the corresponding parameters set to these values.

  If you have optimized some parameters for your application, PI recommends that you repeat your optimization routine once again with the new stage, because there are variations, e.g. in the stiffness and natural frequency of the PZT stages.
- If you send your stage to PI for e.g. upgrade or repair, the calibration data stored in the ID-chip might be changed in the process. When you re-connect this stage to the controller to which it was connected before, the firmware will detect that type and serial number are unchanged and **will not** read any more ID-chip data.

  To force the controller to read the complete data of the ID-chip every time the controller is switched on, you have to set address 109 of the controller non-volatile memory to 1. This has to be done for each axis separately. Use the following compound commands to do this:

  ```
  1dp-1,109dw1   for axis 1
  2dp-1,109dw1   for axis 2, etc.
  ```

  Now switch the controller OFF and ON again. This time **all** data is read from the ID-chip and stored in the controller.

  To ensure that at next power on the controller will not read all data again and perhaps overwrite your optimized parameters, you will have to set address 109 back to 0, again for each axis separately.

1dp-1,109dw0	for axis 1,
2dp-1,109dw0	for axis 2, etc.

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

3 Functionality and Block Diagrams

3.1 Axis and Channel Definitions

In dealing with a digital controller, the terms “axis” and “channel” are not synonymous. The axes of the coordinate system to which the commands are referenced can be freely defined and need not coincide with either the sensor directions (sensor input channels) or the actuator directions (PZT output channels). Nor must the sensor and actuator directions coincide or form an orthogonal system. The user is thus free to command the motions required using the coordinate system that is the most convenient for the application.

Typical system configurations include:

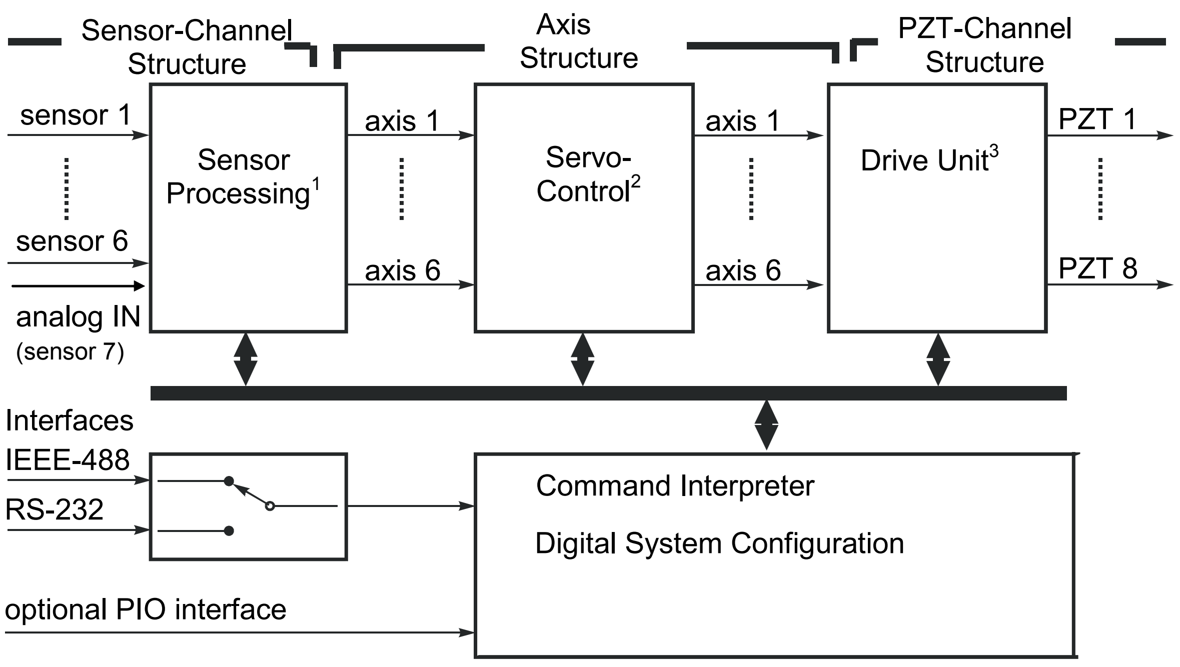
- Independent single-axis piezo actuators or stages. In this case the axes and channels correspond to one another. The axis names are arbitrary, examples are X, Y, Z, RotX, RotY or RotZ (where RotX means rotation around X). The user can even assign new axis names (maximum 4 characters).
- One multi-axis stage; Here the number of axes may be different from the number of channels. The number of PZT channels and sensor channels to and from the stage are also independent of each other.
- Analog-In line (not with all versions) configured as input from user-supplied external position sensor. The sensor direction is freely configurable in the same way as the capacitive-sensor sensor channels: the corresponding transformation matrix determines its relationship to the logical axes in the system. See p. 32 for details.

Typical axis definitions:

- Translation axes are typically named X, Y, and Z and are generally made to correspond to input and output channels 1, 2 and 3 of the stage actuators and sensors.
- Rotation axes are typically named RotX, RotY and RotZ and made to correspond to rotation about the corresponding axis. Some users may prefer to name the rotation axes U, V and W instead.
- The handedness of the axes can be set by the user, independent of that inherent in the hardware.

3.2 E-710 System Overview

The E-710 digital piezo controller consists of sensor processing electronics, digital proportional-integral (P-I) servo-controllers, power amplifiers, communication interfaces and a command interpreter. In the following sections, the design of the 6-axis version is explained. The 3- and 4-axis versions are identical, but have smaller transformation matrices, and some of them have no Analog Input.



[1] Sensor processing details, see Fig. 7;
[2] Servo-controller details, see Fig. 10;
[3] Drive Unit details, see Fig. 11

Fig. 5: E-710 System Overview

3.3 Sensor Processing

The E-710 is designed to work with highly accurate capacitive position sensors. Versions equipped with an Analog Input can also accept an externally conditioned sensor signal (0-10 V) from user-supplied electronics as the position value of one of the sensor channels. See Fig. 7 and p. 32.

The sensor processing section consists of

1. Analog sensor readout followed by analog to digital conversion
2. Digital sensor-value processing
3. Coordinate transformation (channel-to-axis conversion)

Fehler! Es ist nicht möglich, durch die Bearbeitung von Feldfunktionen Objekte zu erstellen.[1]Sensor digital processing details, see Fig. 8

Fig. 6: Sensor Processing

3.3.1 Sensor Analog/Digital Conversion Unit

The sensor and associated analog circuitry converts the mechanical distance change to an analog voltage change. In the standard configuration, the distance between the capacitor plates ranges from 50% to 150% of the mid-point distance. The analog voltage is proportional to the distance change. The mid-point distance is also equal to the measurement range.

The important related parameter is the:

➢ **Sensor Range Factor**, a factor which can be applied to the standard sensor range. A 100 µm sensor can also be used with a measurement range of 200 µm by specifying a factor of 2; at a range of 125 µm with a factor of 1.25 or at 300 µm with a factor of 3.

3.3.2 Sensor Digital Processing Unit

The sensor digital processing section consists of

1. Digital filters, one per sensor channel.
2. Digital compensation for non-linearities in the electronics (per channel)
3. Digital compensation for nonlinearities in the sensor mechanics (per sensor-channel basis)

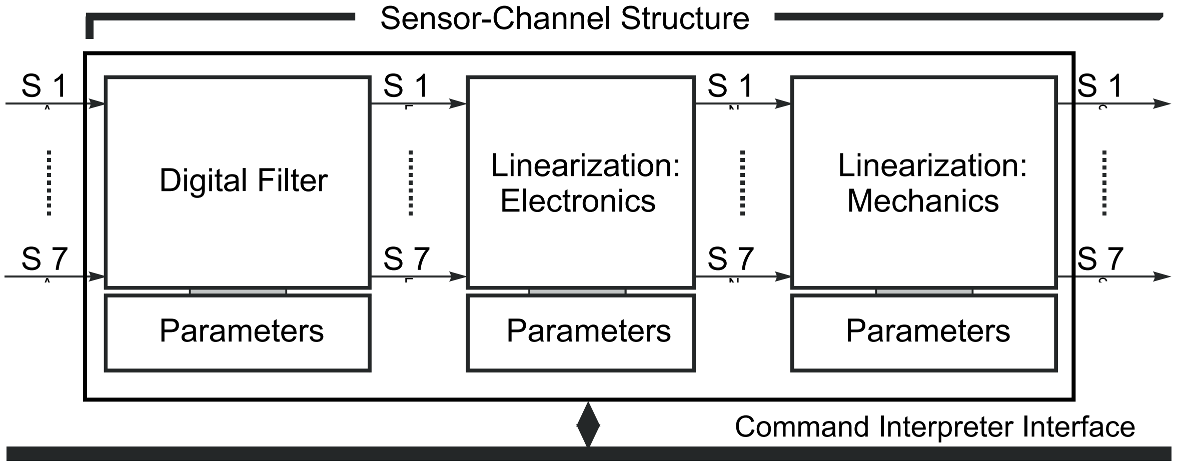


Fig. 7: Sensor Digital Processing

The important related parameters are:

- **Digital Filter Type.** There are three different types of digital filter. IIR filter, FIR filter and user FIR filter. The FIR filter is a simple moving-average filter.
- **Digital Filter Order.** The *order* of the digital filter is number of previous values used in determining the present output. The digital filter order is always 2 for IIR filter and maximum 1000 for FIR filter.
- **Digital Filter Bandwidth**, Only used if the IIR filter has been selected.
- **Polynomial-fit parameters for the electronics**: sensor offset, sensor gain, 2nd and 3rd order sensor correction. These parameters are independent of the connected mechanics. They are set by PI and may not be changed by the user.
- **Polynomial-fit parameters for the mechanics**: sensor offset, sensor gain, 2nd, 3rd and 4th order sensor correction. These parameters depend on the connected stage mechanics. If the connected stage has an ID-chip, the values will be read in from the ID-Chip upon power-up or when the stage is replaced.

The intermediate result values can be reported by special commands as follows:

- The TA command (p. 141) reports the value from the A/D converter (range 0 to 65535 or –32768 to +32767). Value is as same as the TN (p. 142) value.
- The TN (p. 142) command reports the result after the linearization for the electronics
  (-100 to 100).
- The TS (p. 143) command reports the result after the linearization for the mechanics (the unit is µm).

3.3.3 Sensor Coordinate Transformation

Matrix algebra is used for coordinate transformation. Normally fewer than maximum number of parameters is used for the transformation, but the full range is available. It can be used, for example, to compensate the cross-talk. Fig. 9 shows the principle block diagram for the calculation of one axis position from seven sensor values.

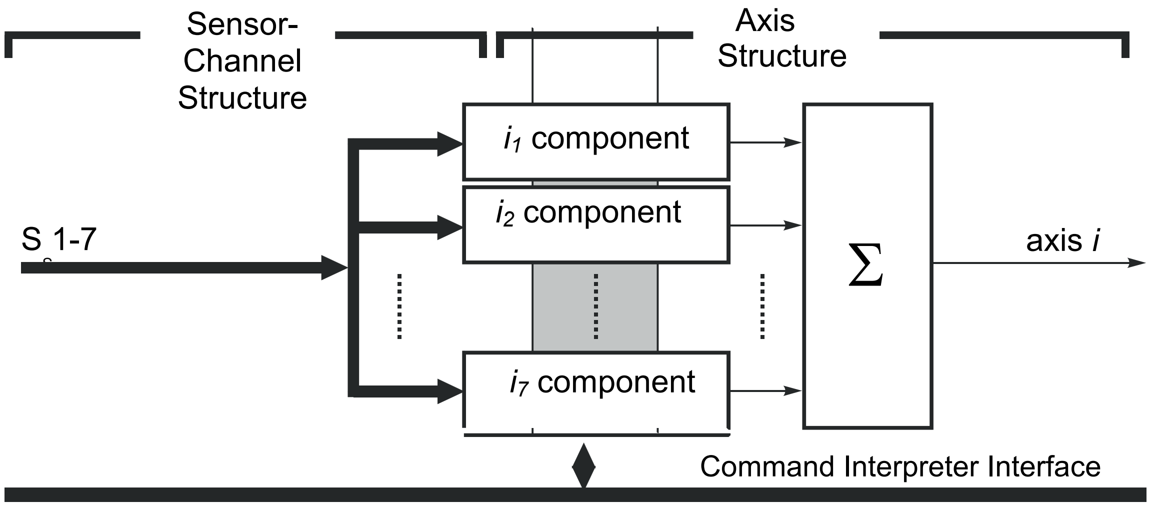


Fig. 8: Sensor-channel-to-axis coordinate transformation (axis i)

The important parameters are the components of the sensor matrix, which are defined as follows:

$$axis_i = a_{i1}S_1 + a_{i2}S_2 + a_{i3}S_3 + a_{i4}S_4 + a_{i5}S_5 + a_{i6}S_6 + a_{i7}S_7$$

The full transformation algorithm can be expressed in matrix form as follows (3- and 4-axis versions use a 4 x 4 matrix):

$$\begin{pmatrix} axis1 \\ axis2 \\ axis3 \\ axis4 \\ axis5 \\ axis6 \\ axis7* \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} & a_{67} \\ a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & a_{77} \end{pmatrix} \begin{pmatrix} S1 \\ S2 \\ S3 \\ S4 \\ S5 \\ S6 \\ S7 \end{pmatrix}$$

**axis7* resultant value is not used

Note that the sensor channel signals (*S1* to *S7*) can be read with the TS command (see previous section), while the *axis1* to *axis6* values can be read with the TP command

If your controller was ordered as part of a complete system, or if PI had sufficient knowledge of your setup, the coordinate transformation matrices will be filled in with the appropriate values during calibration before shipment. The 6-axis versions have commands for automatically updating the matrix to reflect measured crosstalk conditions (SO and OG) and to redefine the center of rotation for rotation axes (CO).

The matrix elements are accessible only with the DR, DP and DW memory commands (see the D... Set/Read Sensor Transformation Matrix Values command, p. 105) and only one at a time, making it quite cumbersome to manipulate them with terminal software. In *NanoCapture™* the values are accessible from the *Device Parameter Configuration* window in the *Servo 1* to *Servo 6* parameter groups. There, the element a_{ij} (i=1, ... , 7; *j* = 1, ... ,7) is referred to as the "[axis i] Position [component] from Sensor *j*". *NanoCapture™* requires you to find the password in the software manual before changing these important values.

The result of the transformation is single-column matrix, normally representing a position expressed in the logical-axis coordinate system (works for both linear and rotational axes). The elements of that matrix could, however, have independent special meanings or represent some measurement results on the inputs.

If the sensor and logical axes in the system correspond, then the values on the main diagonal of the transformation matrix are the scale factors and the rest will be close to zero. The phenomenon of sensor cross-talk between axes appears as (and can be compensated by) non-zero values for these coefficients. The cross-talk can be defined as $\delta_{ij} = \frac{\Delta_{ij}}{\Delta_i}$

where i is the commanded axis and *j* is the axis which has a cross-talk error of Δ_{ij}.

The motion of $axis_j$ is then given by:

$$\begin{aligned} axis_j &= a_{j1}S_{s1} + a_{j2}S_{s2} + a_{j3}S_{s3} + a_{j4}S_{s4} + a_{j5}S_{s5} + a_{j6}S_{s6} + a_{j7}S_{s7} \\ &+ \delta_{ij}(a_{i1}S_{s1} + a_{i2}S_{s2} + a_{i3}S_{s3} + a_{i4}S_{s4} + a_{i5}S_{s5} + a_{i6}S_{s6} + a_{i7}S_{s7}) \end{aligned}$$

If the controller and actuators are ordered together and/or PI has sufficient knowledge about your application, the system will be delivered preconfigured and precalibrated, so it should not be necessary to change these values.

3.4 Digital Servo-Controller

Servo-control is performed independently for each of the logical axes in the system (which is why the logical axes should be orthogonal).

The servo-controller is a proportional-integral (P-I) controller with a notch filter on the output. The target value is received from the command interpreter or the wave generator (which is, of course, under command-interpreter control). Motion commands are input in units like µm, µrad, etc.

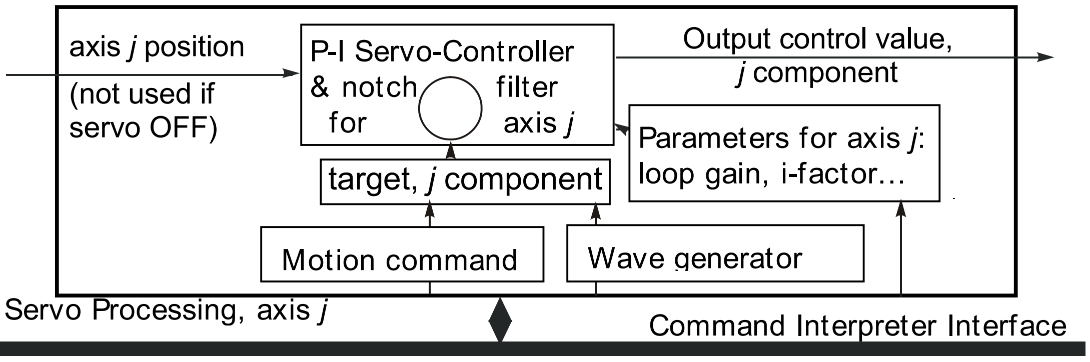

axis j position
(not used if servo OFF)
P-I Servo-Controller & notch filter for axis j
Output control value, j component
Parameters for axis j: loop gain, i-factor...
target, j component
Motion command
Wave generator
Servo Processing, axis j
Command Interpreter Interface

Fig. 9: Servo-control for axis j

3.5 Drive Unit

The drive unit receives the digital control signal from the servo-controller and includes most of the coordinate transformation from the logical axis structure to the PZT channel structure, the digital to analog converters that provide the power amplifier input signals and the power amplifiers which increase the voltage and current of the PZT control signals to the levels required to drive the PZTs.

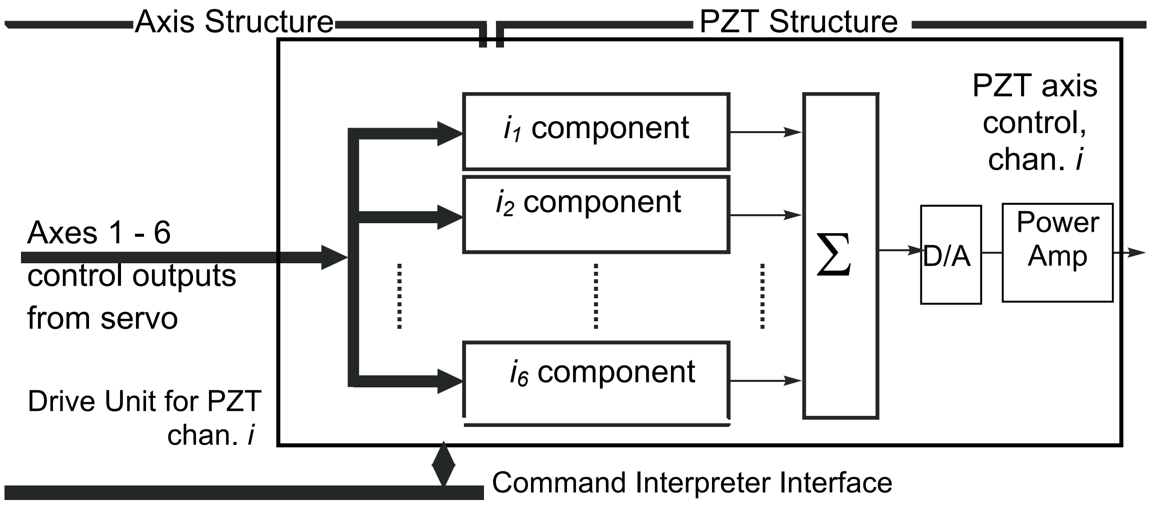

Axis Structure
PZT Structure
PZT axis control, chan. i
i1 component
i2 component
i6 component
Axes 1 - 6 control outputs from servo
Σ
D/A
Power Amp
Drive Unit for PZT chan. i
Command Interpreter Interface

Fig. 10: Drive unit for PZT channel i, i=1 to 8

The axis-to-PZT-channel transformation is also implemented in matrix multiplication. The 3- and 4-axis versions can both drive 4 PZTs and use a 4 x 4 matrix, the 6-axis version an 8 x 8 matrix. Input values of non-supported axes are always 0. In addition, the 6-axis versions provide for application of a correction delta to one axis output as a function of the values of 2 other axes. The function is implemented with a table of up to 625 value sets. This feature is typically used to correct flatness error in X-Y motion by compensating in Z. Note that the values are freely specified, so non-linearities can be corrected. See the NF and NP command descriptions for details.

3.6 Calculation of Target Positions

When servo-control is enabled and turned on, the target position which is feed into the servo-loop of an axis is the sum of the following components:

- So-called "baseline", i.e. the position that was set by a move command (MA p. 123, MR p. 128)

- Wave generator output, which is interpreted as relative position values (see MC (p. 124) and SC (p. 134) and Section "Wave Generator" beginning on p. 50 for more information)
- "Analog target", i.e. the analog input value (see Section "Generation of Target Positions/Offsets Using the Analog Input" on p. 40 for more information)

These commanding options can be combined in arbitrary configuration.

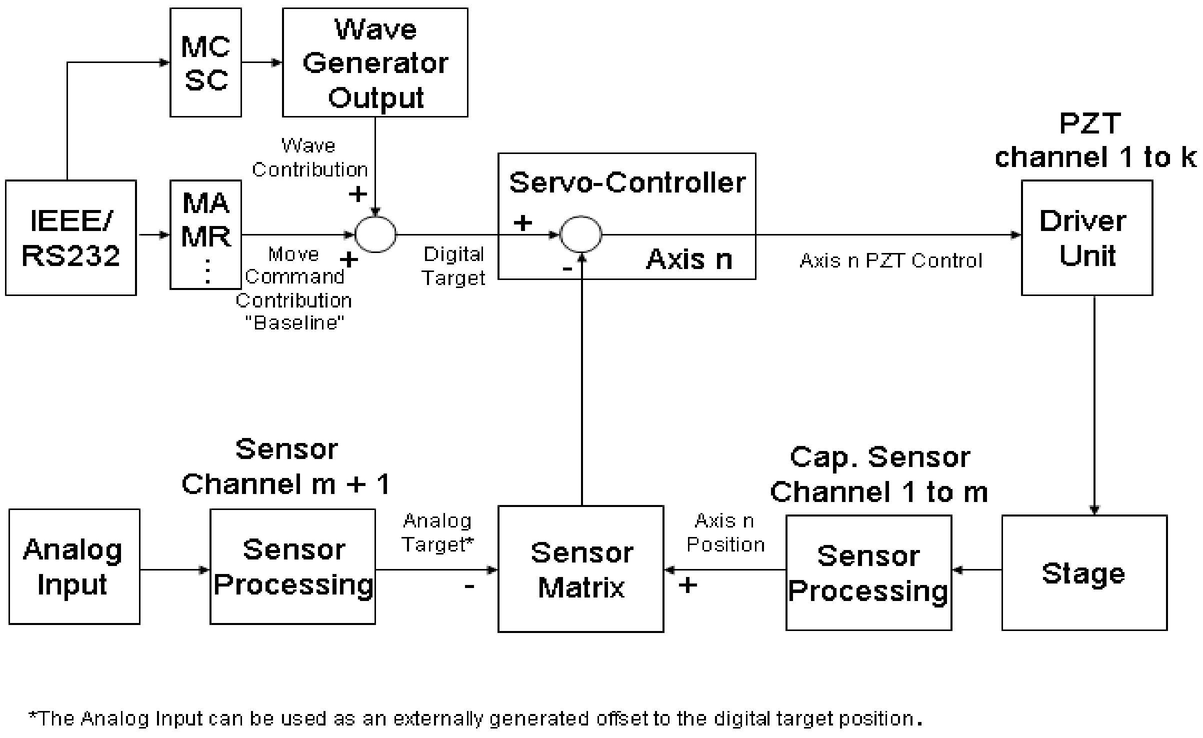


Fig. 11: Servo processing overview illustrated for Axis n

If you are working with the native commands in a terminal (i.e. the E-710 GCS DLL is not used), note the following:

Depending on the wave generator start option a baseline reset will be required after the wave generator run. See MD (p. 127) for more information.

When using the analog input for target generation of axis *n*, the *n*TP command (see p. 142) no longer gives the actual position of that axis, but the actual position minus the Analog Target. Calculate the actual position as follows:

Axis *n* Actual Position = *n*TP Report Value + (*m+1)*TS Report Value

4 Hardware Basics

4.1 Controller Timing Basics

Digital controller operation is based on internal sampling cycles. The E-710 uses a servo sampling rate of 5 kHz. New target positions are processed at most only once per servo sample period of 200 µs. Similarly, new sensor data is processed and can be made available also at most once per sample period. The sampling rate of the sensor data, which can be measured at the sensor cycle trigger output, is higher than the servo sampling rate. The 6-axis version has a sensor data sampling rate of 25 kHz. With the 3- and 4-axis versions it depends on the firmware version: with firmware version 4.xxx it is five times as high (25 kHz) and with firmware versions 5.xxx it is four times as high (20 kHz). If speed is a limiting factor for your 3- or 4-axis application, consider the optional PIO interface.

4.2 Piezo Dynamics Basics

Whenever a new target position is accepted, the stage is moved from the current position (as reported by the sensor) toward this target. On this level, the actual motion is highly dependent on mechanical and electrical characteristics, and the parameter settings. Target position processing and mechanics settling times are typically as follows:

- Target Position Processing < 200 µs (one sample period).
- Settling < 20 ms (can vary with range length or distance traveled).

4.3 Power Requirements and Open-Loop Frequency Response

In order to achieve minimum distortion of the output waveform, it is important to ensure that the control input amplitude is reduced in proportion to the fall-off of the output voltage at higher frequencies.

The frequency response of a given amplifier depends on the amplifier power, the amplifier design, and, of course the PZT capacitance. For dynamic applications, PZTs require high charge and discharge currents. Those requirements are best met by power amplifiers that can source and sink high peak currents. The average current is of secondary importance. For exact information on maximum operating frequency with a given PZT load, refer to the individual frequency response graphs in Fig. 12.

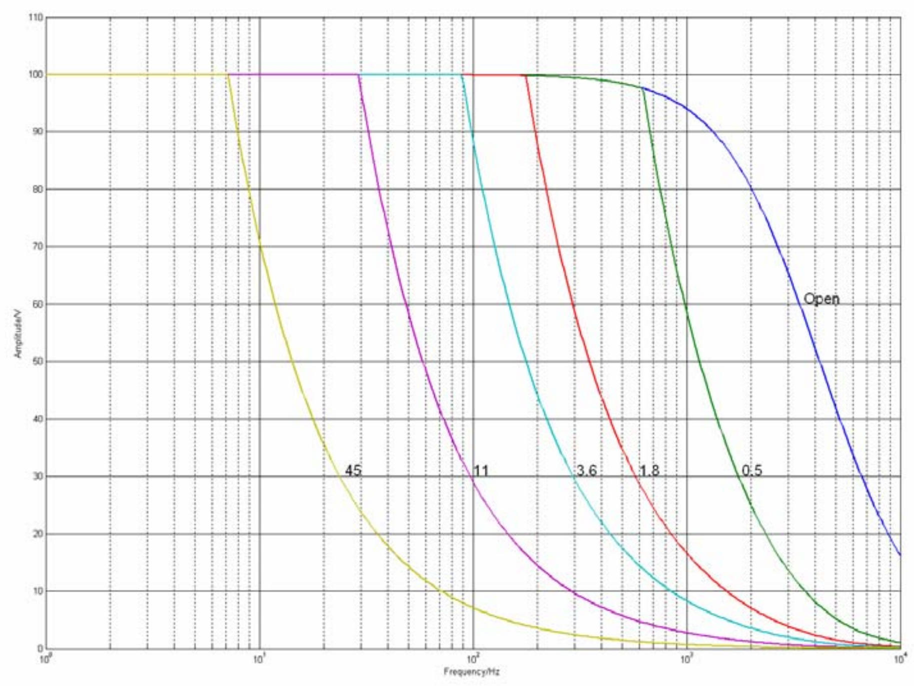


Fig. 12: E-710 frequency response with various PZT loads; capacitance values in µF

Open-loop frequency response data for all PI PZT Power Amplifiers were taken after 15 minutes of continuous operation (PZT and amplifier) at room temperature. Immediately after power-up (cold conditions) maximum operating frequency is higher. The indicated capacitance values are small-signal values for real PZTs (measured at 1 V, 1000 Hz, 20 °C, no load). The capacitance of PZT ceramics significantly changes with amplitude, temperature, and load—up to approximately 200% of the unloaded, small-signal capacitance at room temperature. Therefore the frequency response graphs actually reflect a higher load to the amplifier than the capacitance values indicate.

5 Communication

Communication with the E-710 must be established either through the serial RS-232 or the IEEE 488 interface. The E-710 activates whichever interface receives the first character after power-up. The other interface is disabled until the next power-up.

The PIO interface, if present, is a dedicated high-speed interface for real-time transfer of target and sensor data; it operates in addition to RS-232 / IEEE 488 and cannot carry the commands used with this serial interfaces.

After power up of the E-710, no communication between host computer and the controller is possible before the power LED turns green.

To test the communication using the ASCII commands described in this User Manual (native command set), you can simply start the *WinTerm32* terminal program. Note that ASCII commands are transferred as terminated by a line feed $\boxed{\text{LF}}$ character. The command is executed only after the $\boxed{\text{LF}}$ is received.

5.1 RS-232

Serial communication can be used for service tasks and low- and medium-performance communications (use the C-815.34 cable that comes with the controller).

The E-710 serial interface is set as factory default to: 9600 baud, 8 data bits, 1 stop bit, no parity

Hardware handshake (CTS) must be activated and properly implemented by the host PC.

The baud rate can be changed by the BR command (p. 93). The maximum baud rate is 115,200. Note that this command is sent automatically by PI communications software which changes the baud rate on both sides of the interface at once.

The baud rate can also be changed using the following DP (p. 95), DW (p. 97) command sequence

5DP-1,246DW*baudrate*

or using *NanoCapture*™ (up to rev. 3.5.1 in *System* parameter group, with rev. 4.0.2 and newer in *Interface parser* parameter group). In both cases the data will be written to non-volatile memory, and the new setting will be active not until the next controller power on. This prevents loss of communication during the transmission of modified communication parameters.

5.2 IEEE 488

Compared with the serial communication, the IEEE 488 interface allows faster communication and should be used if possible (factory default IEEE 488 address setting: 4). Use shielded cable.

5.2.1 IEEE Address Setting

No two devices on an IEEE 488 network may have the same address. The IEEE 488 address of the IEEE 488 (GPIB) interface on the E-710 can be changed from the factory default of 4 by using the GA command (see p. 110).

The device address can also be changed using the following DP (p. 95), DW (p. 97) command sequence

5DP-1,245DW*address*

or using *NanoCapture™* (up to rev. 3.5.1 in *System* parameter group, with rev. 4.0.2 and newer in *Interface parser* parameter group). In both cases the data will be written to non-volatile memory, and the new setting will be active not until the next controller power on. This prevents loss of communication during the transmission of modified communication parameters.

5.2.2 Serial Poll Response Byte

Currently only bit 4 (hex: 0x10) of the *Serial Poll Response Byte* is used. According to IEEE 488.2, bit number 4 is the *Message Available Bit* (MAV). It is set to 1 by the controller when it has data to send.

5.3 Digital Trigger Signals (Digital I/O Lines)

There are 4 types of trigger signals available on the Digital I/O connector. For pinout, see p. 157 (3- and 4-axis models) or page 163 (6-axis model).

- ➢ Sensor/Servo cycle trigger (OUT0), synchronized with the sample period of the sensor data servo-loop processing, shows the internal servo-loop processing state: HIGH for servo calculating and LOW: servo calculating finished (Fig. 13).
- ➢ Command cycle trigger (OUT1), is high from "command received" to "command processed". Typically the commands send/receive time (trigger low) is 0.5 ms for GPIB and 1.8 ms for RS-232 at 115200 baud (Fig. 13).
- ➢ Wave generator timing trigger (output). The trigger shape and position in the waveform is programmable (firmware Rev 5.010 or higher and all 6-axis versions). See p. 53 for details.
- ➢ Wave generator timing triggers (input). The wave generator can be configured to start output depending on this triggers, see p. 58 for details

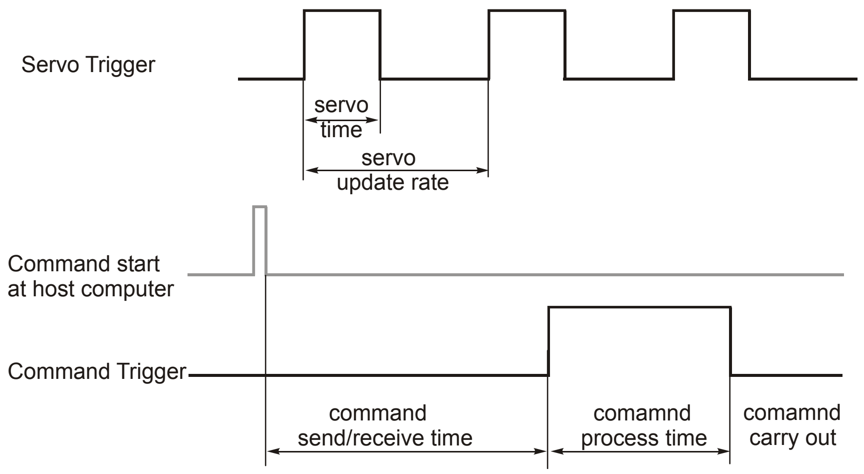


Fig. 13: Servo and command trigger shapes. Other trigger output shapes are programmable.

5.4 Analog Input

The Analog Input line (pinout p. 163) is available on all 6-axis versions and on the 3-axis versions E-710.A3D, E-710.APD and E-710.APS(0). It can be used to generate target positions or as an external sensor for position control, e.g. for auto focussing. For details see Section "Using the Analog Input" on p. 36 and Section "Auto Focussing" on p. 44.

5.5 Synchronization of Multiple E-710s

If only one E-710 is used, it will work as a master, i.e. it uses an internal 100 kHz signal.

Synchronization of multiple E-710s will be necessary to avoid interferences, when the connected mechanics and especially the capacitive sensors are installed close to each other. For that purpose, all E-710s are equipped with a Sync connector (see Sections 17.2 and 17.3 for the location on the device). Two E-710s can be synchronized via their Sync connectors using a special cable which can be obtained from PI, order# E71000207. Because the cable orientation determines the master-slave configuration, you have to respect the labelling on both ends of the cable when installing the connection: the "Master" side must connect to the device which is to be the master, and the "Slave" side to the slave device. See Section 17.5 on p. 164 for technical details.

NOTES

If two E-710s were ordered and calibrated together, normally the device with the axes X, Y and Θ_Z is the master.

If you want to synchronize more than two E-710s, there must be only one master; all others are slaves. Contact your Physik Instrumente sales engineer for an individual cabling solution.

5.6 Optional PIO Interface (3- & 4-Axis Versions Only)

The E-710.P3D, E-710.P4D, E-710.P4L, E-710.APD and E-710.APS(0) are equipped with a parallel input/output interface (PIO). The PIO equips the E-710 with high-speed data-exchange capability. Due to their serial nature and the command traffic they carry, the standard RS-232 and IEEE-488 interfaces always introduce speed limitations and processing-time uncertainties. The PIO was developed to allow fastest-possible and predictable access to crucial positioning data. It does not replace the ASCII command interface, but operates in addition to it. It supports “target write”, “sensor read” and “on-target signal read” operations on a by-axis basis.

A second reason for a PIO is to provide an isolated, dedicated data transmission path for highly sensitive precision applications.

The PIO can be connected to a special card in the host computer. The National Instruments PCI-96-DIO digital I/O card is recommended. Current versions of the *NanoCapture™* software do not recognize and cannot use the PIO link. Sample programsshowing how to address the PIO can be found on the included CD (in the \Samples\LabView and \Samples\C directories). See p. 84 for a description of PIO commands and p. 166 for signal-level and timing information.

NOTE

The E-710 GCS DLL can be used with models with the PIO feature, but does not include PIO support.

5.7 Communication Problems

If the ASCII connection fails, check the following points:

- ➢ Power LED of the controller green?
- ➢ Only one of the RS-232 or IEEE interfaces in use?
- ➢ RS-232 baud rate or IEEE address parameters set correctly?

If the current baud rate or address parameters are unknown, they can be reset to their default values (baud rate = 9600; IEEE address = 4) as follows.

1. Switch E-710 power off
2. Wait for 5 seconds
3. Switch power on
4. As soon as the power LED begins to flash, switch the power off again

The next time the power of controller is switched on, the communication parameters for both the IEEE and RS-232 interfaces will be set to the factory default values of 4 and 9600 respectively.

6 System Flags and Settings

6.1 State and Error Flags

All system flags can be read using ASCII commands. Each servo-controlled axis has a status word to indicate its current state. You can get the current state of the controller by sending the commands *a*GI1 and *a*GI8, where *a* is the number of the axis. For more information read the description of the GI command (p. 113). In PIO versions, some flags are also displayed on LEDs and can be read over the PIO interface (p. 85).

6.1.1 Flags Readable via ASCII Commands Only

- Servo on/off: If the servo loop is turned off the piezo can be controlled by the voltage set (VS, p. 146) command. If the servo loop is turned on the piezo can be controlled by the positioning commands move absolute (MA, p. 123) and move relative (MR, p. 128). The state of the servo-loop (on or off) can be changed by the SL command (p. 137).
- AutoZero running: This flag is set while AutoZero procedure is running
- Generator running: This flag is set while the wave generator is running (revision 5.xxx, 6.xxx, 7.xxx).
- Target position at low limit: This flag indicates that the target position which was set by the commands move absolute (MA, p. 123) or move relative (MR, p. 128) is lower than the value of the *Range limit min* parameter (at address 41). You will find this parameter on the *servo setup* pane for the respective axis in *NanoCapture*™.
- Target position at high limit: This flag indicates that the target position which was set by the commands move absolute (MA, p. 123) or move relative (MR, p. 128) is higher than the value of the *Range limit max* parameter at address 42. You will find this parameter on the *NanoCapture*™ *servo setup* configuration dialog for the corresponding axis.

6.1.2 Flags with Supplemental LED Display and PIO Readout

Units with the PIO interface have LEDs on the front panel that show the current on-target (position error) and overflow status of the individual axes. If servo-control is disabled or OFF, then the corresponding on-target and overflow flags are zero and the LEDs are dark.

These flags can also be read out electronically over the PIO (see p. 85 for details):

- On-Target (position error) flags (green LEDs): This flag is set (and the LED lit) if servo is ON and the position error is smaller than the tolerance setting. The *tolerance* parameter (stored at address 64 in non-volatile memory) can be modified by *NanoCapture*™. You will find this parameter in the *NanoCapture*™ *servo setup* configuration dialog for the corresponding axis.
- The On-Target4 line can be reconfigured as a “PIO data-valid” signal for use in triggering external electronics or as a “Servo All-On” indicator. The On-Target 4 LED appears to be ON when the line is configured to show data-valid. When in “Servo All-On” mode, the LED goes out if any of the axes has servo-control OFF. See the “D... Set/Read PIO ON-Target4 Line Function” command on p. 104 for configuring this line, and “PIO Read Operation Timing” p. 169 for signal details.
- Overflow (piezo voltage limits) flags (red LEDs): This flag is set (and the LED lit) if the voltage necessary to reach the target is outside the operating range of

the PZT. Overflow can occur if no stage is connected, or if there is a calibration/zero-point problem.

6.2 Axis Properties

The property settings described here affect the operation (or failure) of other commands, as noted in the respective command descriptions.

6.2.1 Curve Control

The various curve-control settings enable or disable an axis for participation in the following types of operations:

Waveform move	Motion controlled by the Wave Generator (see p. 50)
Synchronous move	Wave generator motion in synchronization with wave generator motion of another axis (allow/disallow, see SC and MC commands, p. 134, 124)
DDL	Enabling is ineffective without Dynamic Digital Linearization feature (see p. 61)
Synchronous DDL	Enabling is ineffective without Dynamic Digital Linearization feature (see p. 61)

See the D... Set/Read Curve Control command, p. 99 for a description of making these settings.

6.2.2 Proportional Gain

The proportional gain value is a factor applied to the value specified in a proportional move (SM) command to get the target position of the axis. With the SM command, it is possible to trace straight lines in space in any direction with a single command by setting the proportional gain values of the axes associated with X, Y and Z motion appropriately. See the SM command (p. 137) for an example of this type of motion.

See the D... Set/Read Proportional Gain command, p. 104 for setting the proportional gain of an axis (address 62). The factory default is 0.

6.2.3 Curve Moving Report Table Data Filtering

During wave generator motion, a table with actual positions, target positions and/or position errors can be built up (see the TL command on p. 141). If the DDL feature is present, DDL data can be included also. The table can be filled with either raw data or filtered data. Filtering can not only be enabled or disabled (addr. 158, see the D... Set/Read Filter Enable for Curve Moving Report command, p. 101) but the filter bandwidth can also be adjusted (addr. 159, see D... Set/Read Filter Bandwidth for Curve Moving Report, p. 100).

7 Using the Analog Input

7.1 Basics

E-710 6-axis versions and the E-710.Axx 3-axis versions are equipped with an ANALOG IN connector over which an analog input signal can be fed into the controller (for pinout see p. 163). For signal input, use the cable #P892B0006 which comes with the controller. Inside the controller, the analog input is treated like another sensor value and participates in the coordinate transformation. While with 6-axis versions the Analog Input is assigned to Sensor channel 7, it is handled as Sensor channel 4 with 3-axis models. By default, the Analog Input is not connected to any axis of the controller.

NOTE

The descriptions and examples in this document refer to the E-710 6-axis versions but can be adapted to the E-710.Axx 3-axis versions.

In general the Analog Input can be used in one of the following ways:

- As an externally generated offset to the target position (see p. 40)
- As an external servo-sensor, either of position or of some other position-related value, like image sharpness in an autofocus application (see p. 42)
- As an autofocus signal for use by the AutoFocus command outside the servo loop (see p. 107 and p. 44)

Like the sensor signals, the analog input undergoes digital filtering, polynomial linearization (including offset and gain adjustment, orders 0 and 1 respectively). The values at the different processing stages can be read out with the corresponding commands (in the 6-axis versions analog IN is referred to as Sensor channel 7).

Note: If the signal is being used as something other than a position sensor, then the meanings of the values will, of course, be different from that given in the readout command descriptions.

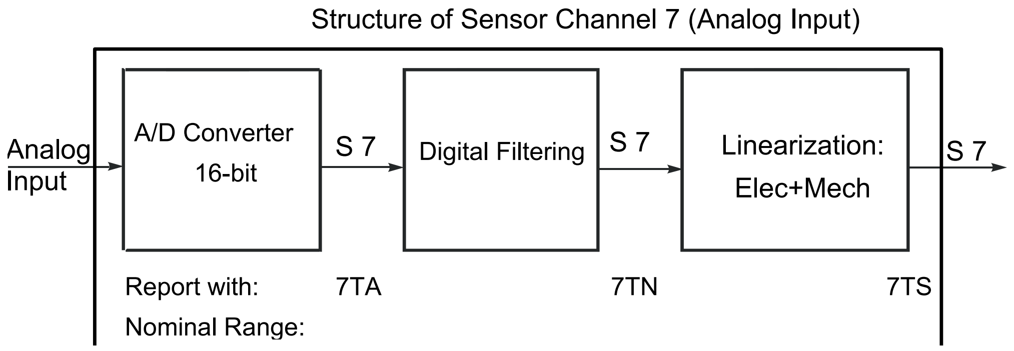


Fig. 14: Digital Processing of the Analog Input (Sensor Channel 7 in E-710 6-axis versions)

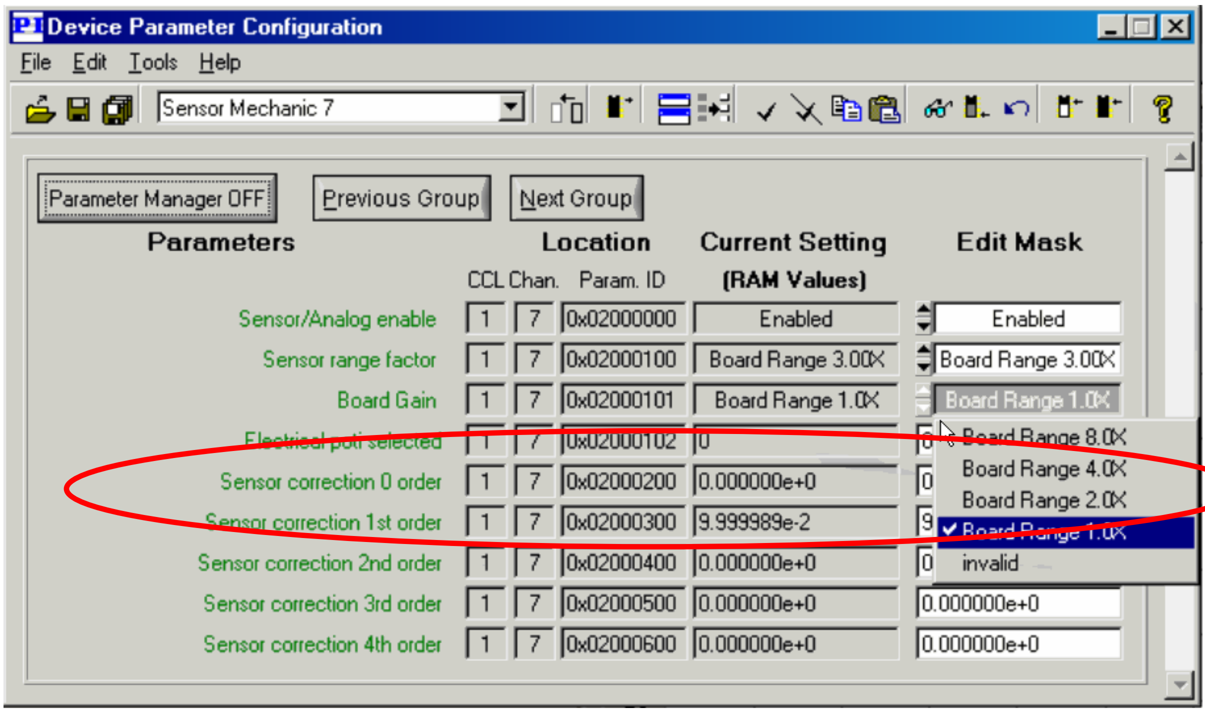
7.2 Parameters

Only parameters which apply to Analog In or differ from those of the sensor channels are discussed here.

7.2.1 Analog Input Voltage Range

The positive and negative Analog IN lines should both be kept within 10 V of the system ground to minimize common mode effects. The A/D converter input range is centered on 0 V differential, but its range is adjustable, with a maximum width of 20 V.

In the 6-axis controllers, the Sensor Mechanics 7 *Board Gain* parameter (see Fig. 16) can be used to amplify the Analog IN signal before digitalization so as to exploit the full 20-volt-wide range of the 16-bit A/D converter. Settings of 1x, 2x, 4x and 8x are available. On 4-axis controllers there is a factory setting available. There is no offset parameter, so unipolar signals sacrifice 8 bits resolution, or must be made bipolar by connecting a suitably stable, mid-range reference voltage in place of their GND (see p. 163).



Parameters	CCL	Chan.	Param. ID	Current Setting (RAM Values)	Edit Mask
Sensor/Analog enable	1	7	0x02000000	Enabled	Enabled
Sensor range factor	1	7	0x02000100	Board Range 3.00X	Board Range 3.00X
Board Gain	1	7	0x02000101	Board Range 1.0X	Board Range 1.0X
Electrical poti selected	1	7	0x02000102	0	0
Sensor correction 0 order	1	7	0x02000200	0.000000e+0	0
Sensor correction 1st order	1	7	0x02000300	9.999989e-2	9
Sensor correction 2nd order	1	7	0x02000400	0.000000e+0	0
Sensor correction 3rd order	1	7	0x02000500	0.000000e+0	0.000000e+0
Sensor correction 4th order	1	7	0x02000600	0.000000e+0	0.000000e+0

Board Range 8.0X
Board Range 4.0X
Board Range 2.0X
✓ Board Range 1.0X
invalid

Fig. 15: Parameter Configuration for the Analog Input (Sensor Channel 7)

7.2.2 Digital Range and Offset

Coefficients for 4 orders of sensor mechanics linearization are available to the user. The 0-order correction corresponds to a constant offset value, and the 1st order correction to a gain factor. In *NanoCapture™* they in the *Sensor Mechanics* group as “Sensor Correction...” The result of the linearization can be seen with the 7TS command. Because the corrections are applied to the digital data, resolution can be lost when large corrections are specified.

7.2.3 Configuration for Different Uses

The use of the Analog In signal is determined primarily by the entries in the Sensor Matrices for the various axes (and, of course, the digital target). After linearization, the resulting Analog IN value goes into the sensor coordinate transformation section, where its components in the different axis directions are calculated using the values in the matrices (see Section 3.3.3, p. 24). If, for example, Analog IN is to represent the Axis 3 target position, then Axis 3 must be configured with Sensor 7 having a non-zero negative contribution to its position. Other Examples are explained in detail below.

The Analog In contribution in the sensor-to-axis transformation is accessible in *NanoCapture™* in the *from Sensor 7* parameter of the *Servo 1* through *Servo 6* parameter groups (see .Fig. 18). If set to 0, Analog IN does not participate in the servo loop, but is still available to the FC command.

Note: If the analog input is being used for something other than an external position sensor, then the results of commands which normally report (or record) positions will be affected. For example, if the analog input is used to specify a 1 µm offset to the digital target for axis 1, then when the digital target is 0, the axis will be at +1 µm but the 1TP command will report 0 µm.

7.3 Examples

The following examples are constructed around a 6-axis E-710 with 6 capacitive sensors, an analog input and 8 piezo output channels connected to a host PC running *NanoCapture™*.

As an example, Fig. 17 shows how the servo-loop for Axis 4 works with no Analog IN component. The Axis 4 Digital Target might come from an MA or MR command, and the Axis 4 Servo-Controller sees the actual axis 4 position, which can also be read out with the 4TP command.

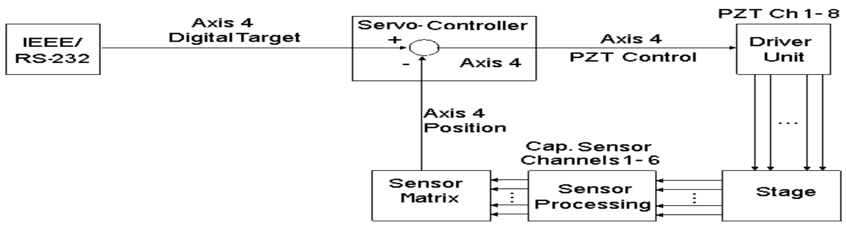
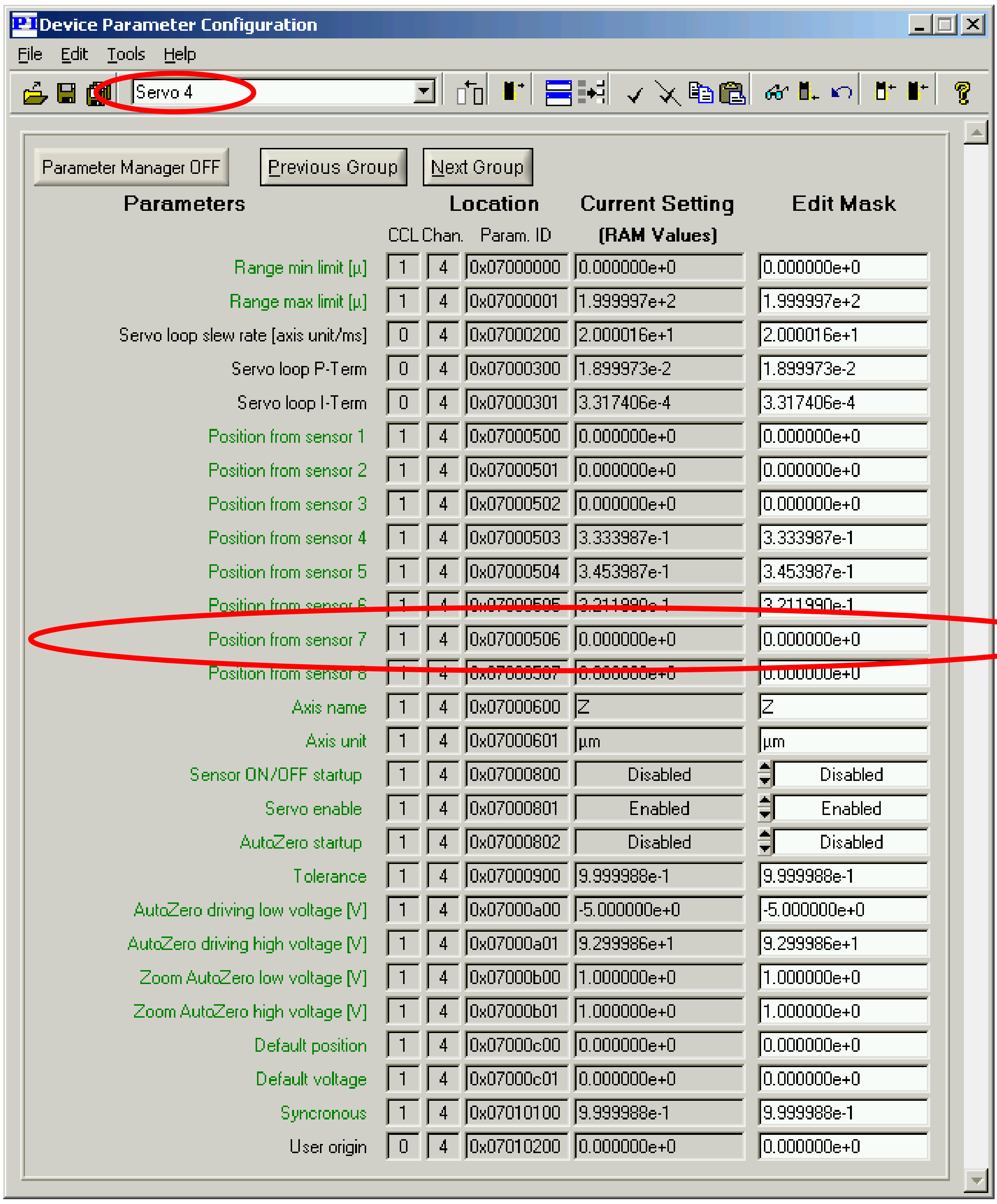


Fig. 16: Servo-loop for axis 4 with no analog input contribution

Eliminating the influence of Analog IN means setting the *from Sensor 7* parameter of *Servo 4* to 0, as shown in Fig. 18.



Parameters	Location CCL	Location Chan.	Location Param. ID	Current Setting (RAM Values)	Edit Mask
Range min limit [µ]	1	4	0x07000000	0.000000e+0	0.000000e+0
Range max limit [µ]	1	4	0x07000001	1.999997e+2	1.999997e+2
Servo loop slew rate [axis unit/ms]	0	4	0x07000200	2.000016e+1	2.000016e+1
Servo loop P-Term	0	4	0x07000300	1.899973e-2	1.899973e-2
Servo loop I-Term	0	4	0x07000301	3.317406e-4	3.317406e-4
Position from sensor 1	1	4	0x07000500	0.000000e+0	0.000000e+0
Position from sensor 2	1	4	0x07000501	0.000000e+0	0.000000e+0
Position from sensor 3	1	4	0x07000502	0.000000e+0	0.000000e+0
Position from sensor 4	1	4	0x07000503	3.333987e-1	3.333987e-1
Position from sensor 5	1	4	0x07000504	3.453987e-1	3.453987e-1
Position from sensor 6	1	4	0x07000505	3.211990e-1	3.211990e-1
Position from sensor 7	1	4	0x07000506	0.000000e+0	0.000000e+0
Position from sensor 8	1	4	0x07000507	0.000000e+0	0.000000e+0
Axis name	1	4	0x07000600	Z	Z
Axis unit	1	4	0x07000601	µm	µm
Sensor ON/OFF startup	1	4	0x07000800	Disabled	Disabled
Servo enable	1	4	0x07000801	Enabled	Enabled
AutoZero startup	1	4	0x07000802	Disabled	Disabled
Tolerance	1	4	0x07000900	9.999988e-1	9.999988e-1
AutoZero driving low voltage [V]	1	4	0x07000a00	-5.000000e+0	-5.000000e+0
AutoZero driving high voltage [V]	1	4	0x07000a01	9.299986e+1	9.299986e+1
Zoom AutoZero low voltage [V]	1	4	0x07000b00	1.000000e+0	1.000000e+0
Zoom AutoZero high voltage [V]	1	4	0x07000b01	1.000000e+0	1.000000e+0
Default position	1	4	0x07000c00	0.000000e+0	0.000000e+0
Default voltage	1	4	0x07000c01	0.000000e+0	0.000000e+0
Syncronous	1	4	0x07010100	9.999988e-1	9.999988e-1
User origin	0	4	0x07010200	0.000000e+0	0.000000e+0

Fig. 17: Servo Parameter Configuration for Axis 4; Analog Input not used

7.3.1 Generation of Target Positions/Offsets Using the Analog Input

The algorithm in the controller is the same, but now the Axis 4 contribution of Analog IN (Sensor 7) has been made -1. As a result, the servo-controller sees the Axis 4 Position minus the Analog Target (see Fig. 19) in place of the Axis 4 Position of Fig. 17. The Servo-Controller moves the axis to make this value equal to the Digital Target, meaning that the actual position is offset from the Digital Target by an amount equal to the Analog Target:

Effective Target = Analog Target + Digital Target

The Analog Target value can be read out with the 7TS command;

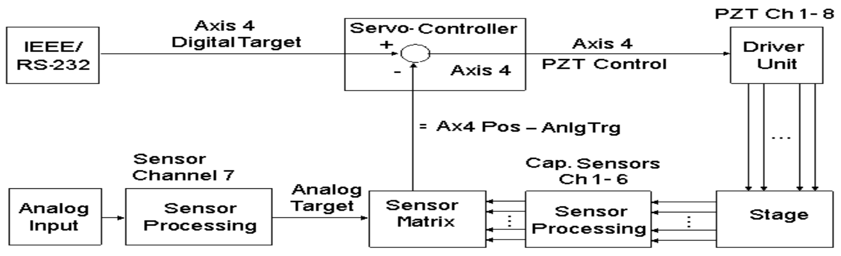


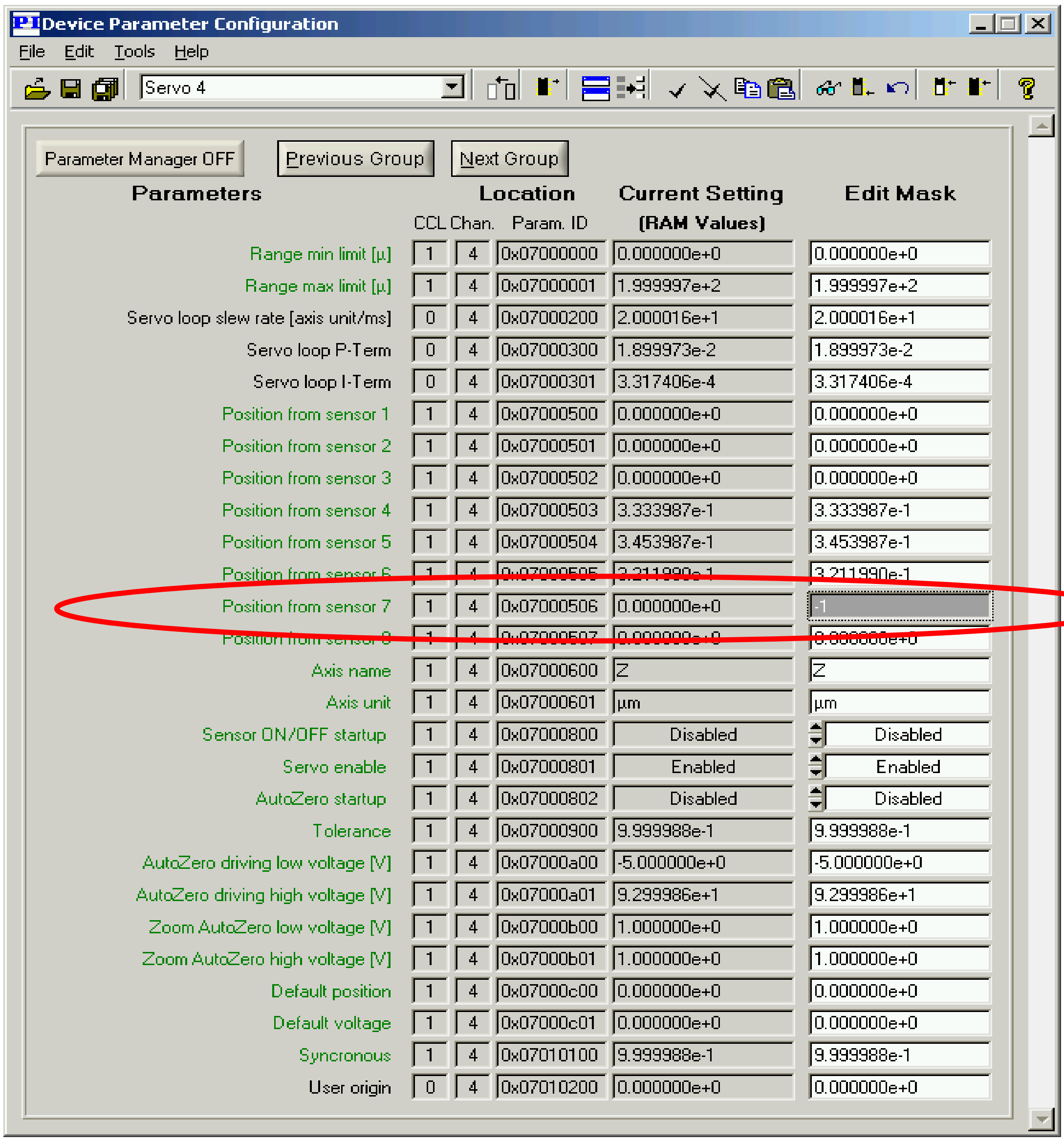
Fig. 18: Servo-loop for axis 4; Analog Input used for target generation

In *NanoCapture™*, the settings required for this situation (see Fig. 20) are characterized by a negative value in the *Position from sensor 7* field and the normal, positive, values for the sensors measuring axis 4 motion.

Note

Because Analog IN is not being used as a position sensor (but to specify an offset), the verbal descriptions of the values reported by some commands do not apply. For example, the 4TP command no longer gives the actual position, but the actual position minus the Analog Target. Calculate the actual position as follows:

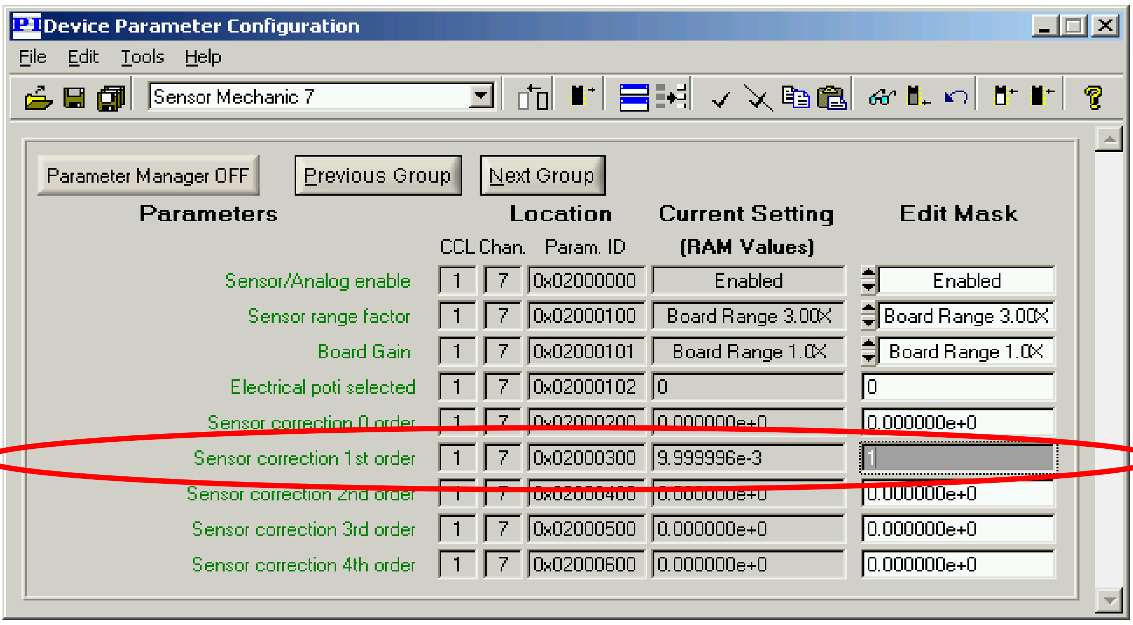
Axis 4 Actual Position = 4TP Report Value + 7TS Report Value



Parameters	Location CCL	Chan.	Param. ID	Current Setting (RAM Values)	Edit Mask
Range min limit [µ]	1	4	0x07000000	0.000000e+0	0.000000e+0
Range max limit [µ]	1	4	0x07000001	1.999997e+2	1.999997e+2
Servo loop slew rate [axis unit/ms]	0	4	0x07000200	2.000016e+1	2.000016e+1
Servo loop P-Term	0	4	0x07000300	1.899973e-2	1.899973e-2
Servo loop I-Term	0	4	0x07000301	3.317406e-4	3.317406e-4
Position from sensor 1	1	4	0x07000500	0.000000e+0	0.000000e+0
Position from sensor 2	1	4	0x07000501	0.000000e+0	0.000000e+0
Position from sensor 3	1	4	0x07000502	0.000000e+0	0.000000e+0
Position from sensor 4	1	4	0x07000503	3.333987e-1	3.333987e-1
Position from sensor 5	1	4	0x07000504	3.453987e-1	3.453987e-1
Position from sensor 6	1	4	0x07000505	3.211990e-1	3.211990e-1
Position from sensor 7	1	4	0x07000506	0.000000e+0	-1
Position from sensor 8	1	4	0x07000507	0.000000e+0	0.000000e+0
Axis name	1	4	0x07000600	Z	Z
Axis unit	1	4	0x07000601	µm	µm
Sensor ON/OFF startup	1	4	0x07000800	Disabled	Disabled
Servo enable	1	4	0x07000801	Enabled	Enabled
AutoZero startup	1	4	0x07000802	Disabled	Disabled
Tolerance	1	4	0x07000900	9.999988e-1	9.999988e-1
AutoZero driving low voltage [V]	1	4	0x07000a00	-5.000000e+0	-5.000000e+0
AutoZero driving high voltage [V]	1	4	0x07000a01	9.299986e+1	9.299986e+1
Zoom AutoZero low voltage [V]	1	4	0x07000b00	1.000000e+0	1.000000e+0
Zoom AutoZero high voltage [V]	1	4	0x07000b01	1.000000e+0	1.000000e+0
Default position	1	4	0x07000c00	0.000000e+0	0.000000e+0
Default voltage	1	4	0x07000c01	0.000000e+0	0.000000e+0
Syncronous	1	4	0x07010100	9.999988e-1	9.999988e-1
User origin	0	4	0x07010200	0.000000e+0	0.000000e+0

Fig. 19: Servo Parameter Configuration for axis 4; Analog IN used for target

The target value or offset can be scaled to match the values from the other sensors on the axis. Assume, for example Axis 4 is linear with a travel range of 200.µm (to be read out as -100 µm to +100µm) and Analog IN ranges from -10 V to +10 V. If the digital target is 0 and Analog IN in +10 V then the *Sensor correction 1st order* factor for *Sensor Mechanics 7* should be changed to 1 as shown in Fig. 21.



Parameters	Location CCL	Location Chan.	Location Param. ID	Current Setting (RAM Values)	Edit Mask
Sensor/Analog enable	1	7	0x02000000	Enabled	Enabled
Sensor range factor	1	7	0x02000100	Board Range 3.00X	Board Range 3.00X
Board Gain	1	7	0x02000101	Board Range 1.0X	Board Range 1.0X
Electrical poti selected	1	7	0x02000102	0	0
Sensor correction 0 order	1	7	0x02000200	0.000000e+0	0.000000e+0
Sensor correction 1st order	1	7	0x02000300	9.999996e-3	1
Sensor correction 2nd order	1	7	0x02000400	0.000000e+0	0.000000e+0
Sensor correction 3rd order	1	7	0x02000500	0.000000e+0	0.000000e+0
Sensor correction 4th order	1	7	0x02000600	0.000000e+0	0.000000e+0

Fig. 20: Parameter Configuration for the Analog Input (Sensor Channel 7); sensor position range is -100 to +100 µm

7.3.2 External Position Sensor

If the analog input is derived from an external position control sensor, it becomes an active part of the servo-control loop.

If the external position sensor is parallel to a logical axis, the *Position from Sensor 7* parameter will be non-zero for that axis and 0 for all other axes. Similarly, if no other sensors contribute to the determination of that axis' position, the *Position from Sensor 7* parameter will be the only non-zero *Position from* parameter for that axis.

Of course, more complicated geometries are possible in which the external sensor has components in various axis directions and no sensor alone measures any axis position.

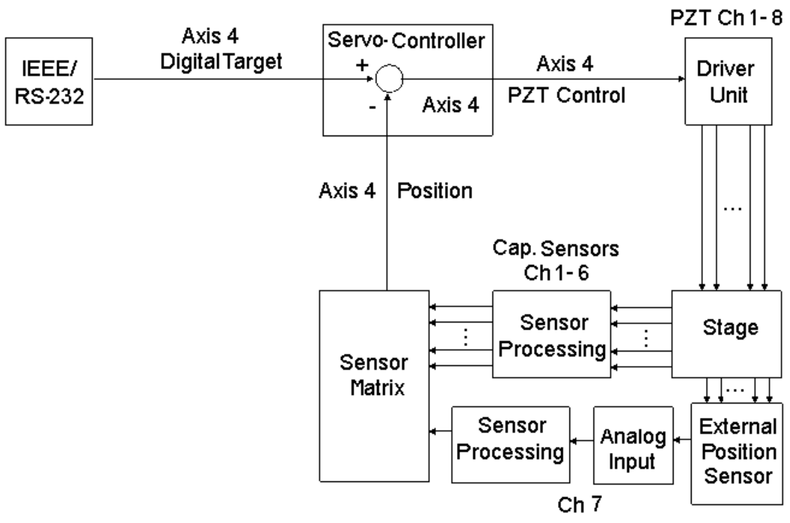
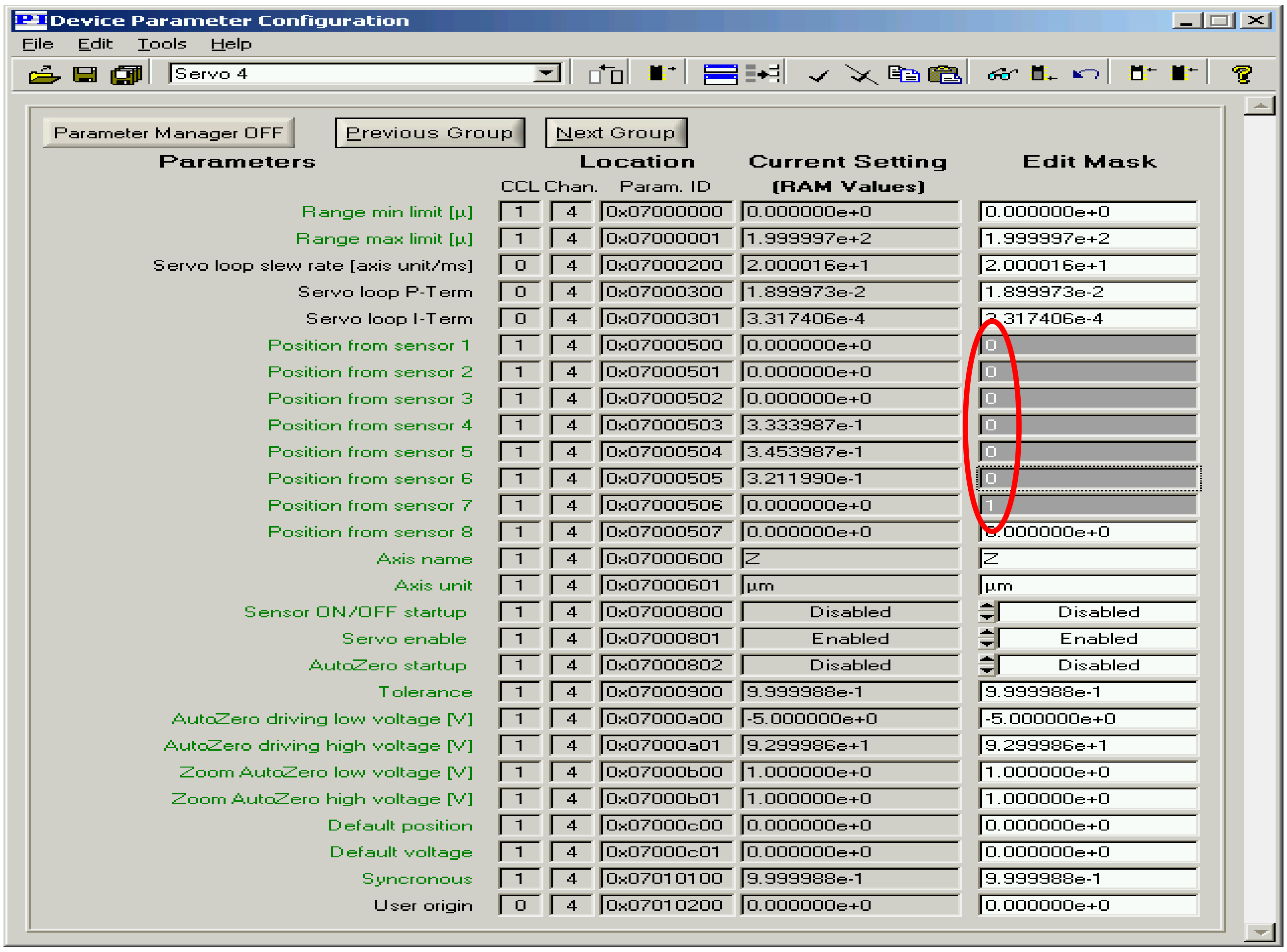


Fig. 21: Servo-loop for axis 4; Analog Input used with external position sensor

Again using Axis 4 as example, the contribution of the external sensor in the axis 4 direction will normally be positive. In the simplest case, where the external sensor is alone in measuring Axis 4, the contribution of the other sensors will be 0. In *NanoCapture*™ this situation is illustrated in Fig. 23



Parameters	Location CCL	Location Chan.	Location Param. ID	Current Setting (RAM Values)	Edit Mask
Range min limit [µ]	1	4	0x07000000	0.000000e+0	0.000000e+0
Range max limit [µ]	1	4	0x07000001	1.999997e+2	1.999997e+2
Servo loop slew rate [axis unit/ms]	0	4	0x07000200	2.000016e+1	2.000016e+1
Servo loop P-Term	0	4	0x07000300	1.899973e-2	1.899973e-2
Servo loop I-Term	0	4	0x07000301	3.317406e-4	3.317406e-4
Position from sensor 1	1	4	0x07000500	0.000000e+0	0
Position from sensor 2	1	4	0x07000501	0.000000e+0	0
Position from sensor 3	1	4	0x07000502	0.000000e+0	0
Position from sensor 4	1	4	0x07000503	3.333987e-1	0
Position from sensor 5	1	4	0x07000504	3.453987e-1	0
Position from sensor 6	1	4	0x07000505	3.211990e-1	0
Position from sensor 7	1	4	0x07000506	0.000000e+0	1
Position from sensor 8	1	4	0x07000507	0.000000e+0	0.000000e+0
Axis name	1	4	0x07000600	Z	Z
Axis unit	1	4	0x07000601	µm	µm
Sensor ON/OFF startup	1	4	0x07000800	Disabled	Disabled
Servo enable	1	4	0x07000801	Enabled	Enabled
AutoZero startup	1	4	0x07000802	Disabled	Disabled
Tolerance	1	4	0x07000900	9.999988e-1	9.999988e-1
AutoZero driving low voltage [V]	1	4	0x07000a00	-5.000000e+0	-5.000000e+0
AutoZero driving high voltage [V]	1	4	0x07000a01	9.299986e+1	9.299986e+1
Zoom AutoZero low voltage [V]	1	4	0x07000b00	1.000000e+0	1.000000e+0
Zoom AutoZero high voltage [V]	1	4	0x07000b01	1.000000e+0	1.000000e+0
Default position	1	4	0x07000c00	0.000000e+0	0.000000e+0
Default voltage	1	4	0x07000c01	0.000000e+0	0.000000e+0
Syncronous	1	4	0x07010100	9.999988e-1	9.999988e-1
User origin	0	4	0x07010200	0.000000e+0	0.000000e+0

Fig. 22: Servo 4 parameter configuration with Analog Input used as external position sensor on Axis 4.

The same scaling considerations apply as with analog signal targets (see p. 40).

7.3.3 Full-Time Autofocus

Full-time autofocus involves using the servo-control loop to adjust the position so that Analog IN (the focus signal) holds a constant value, typically 0. Before invoking full-time autofocus it is usually necessary to use a search algorithm to approach the desired position. The search can be programmed with move commands (read sensor channel 7 to monitor the focus level), or the FC command (see “Auto Focussing” p. 44 *ff.*) can be used (6-axis controllers only) to invoke an automated search.

With Analog IN close to the desired value, the servo-loop can be used to hold that value. This means setting letting Analog IN alone determine the motion of the axis or axes which affect the focus level (set the contributions from other sensor channels to 0). If Analog IN = 0 indicates the in-focus state, then the digital target must be set to 0 also.

Note that if the focus signal has a negative slope at the in-focus point, the respective sensor-channel 7 contribution must have a negative sign and vice versa. If the slope is zero, then additional external signal processing is required.

8 Auto Focussing

The goal of autofocussing is to position an axis so that the position-dependent signal from an external sensor—a focus signal fed into the analog input (see Section 7)—is kept constant on a particular value, typically 0.

Autofocussing is a two-step process:

1. Identifying the focus position, i.e. the position for which the analog input has the desired value (Section 8.1)
2. Keeping the focus position, i.e. servo-controlling the axis to keep the analog input signal at the desired value (Section 8.2)

Note: Since there is only one analog input, the sum signal of the focus signal from laser diodes can not be subjected to auto focus processing, the dark zones (where the differential focus signal shows also zero) can not be identified.

8.1 Identifying the Focus Position

The focus position can be found using move commands and monitoring the focus signal. Six-axis E-710s, however, have the FC command (p. 107) which automates such a search (provided only a single logical axis is involved).

With the FC command there are two different algorithms that can be used: the zero-crossing binary method (Section 8.1.1) and the random method (Section 8.1.2). See the D... Set/Read AutoFocus Parameters command, p. 98 for switching between the two different modes. The parameters used by the FC algorithm are described in Section 8.1.3.

While searching, either with move commands or with FC, the analog input signal (Sensor 7 in the 6-axis E-710s) should not contribute to the “position” values seen by the servo-controller (*Position from Sensor 7* servo parameters = 0).

8.1.1 Zero-Crossing Binary Method

This method monitors the analog input signal and first moves the specified axis one step of a specified size. The next step will use half the step-size of the previous step, etc. When the Analog-IN signal crosses the zero level, the direction of the next step is reversed. The number of steps to take is set using the D... Set/Read AutoFocus Parameters command (p. 98).

If the motion is that of a PIFOC microscope objective nanopositioner, and Analog In represents the change in (first derivative of) the sharpness of an edge in the field of view, the position found will be that where the specimen is in the focal plane.

8.1.2 Random Method

Sometimes the settling time of the stage is too long to find the best focus position within the desired time using the zero-crossing method. In this case the random method can be used. This method uses the same procedure as above, but takes note of the position where the first zero-crossing occurs and moves the axis to that position.

8.1.3 Parameters for Focus Identification

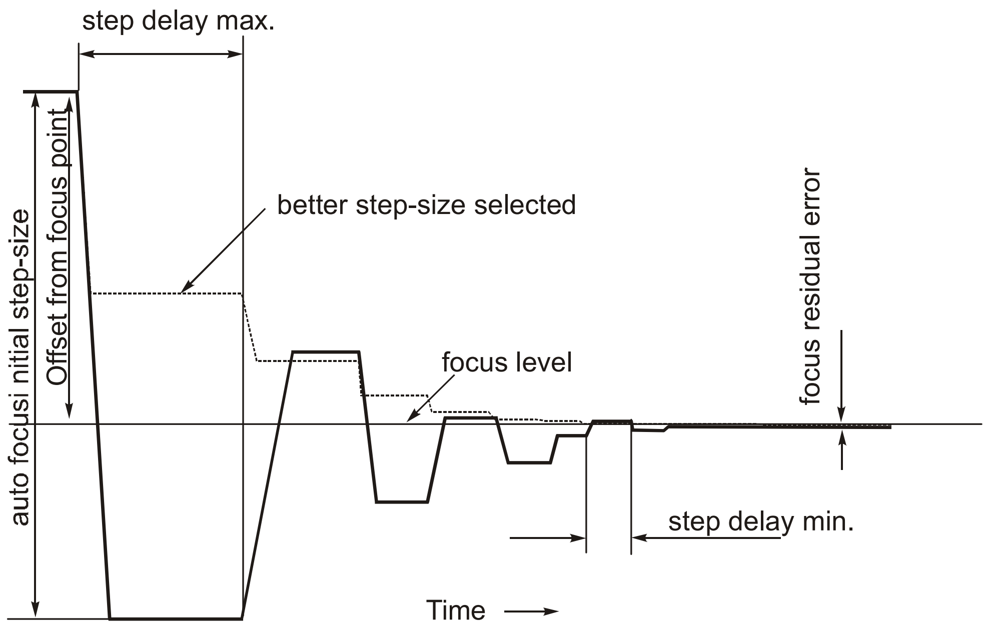


Fig. 23: Zero-crossing method: —large step-size; ---smaller (better) step-size

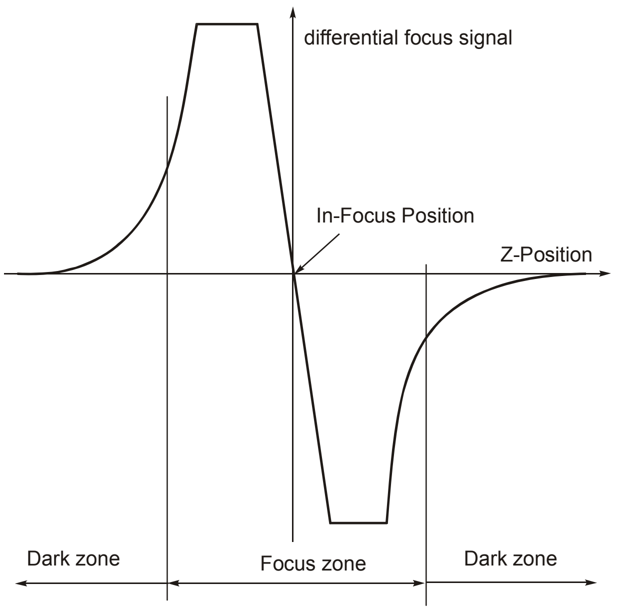


Fig. 24: Autofocussing close to in-focus point

The *step-size* argument to the FC command (p. 107) is the size of first step made. The *step-size* should be set so that

2 x step-size $\geq$ largest possible focus offset

The number of steps is limited, so that the focus position can be found more precisely when the step-size is not too large.

If the amplitude is too small, however, the focus error after FC may still be large (see Fig. 26).

There are a number of adjustment parameters for the FC processing. *Step Delay Max* is used to calculate the delay time after every step. It is used directly after the first step: After the second step, half that time is used, etc. until *Step Delay Min* is reached. These values can be changed by command and, if desired, saved in EEPROM as default values; see the D... Set/Read AutoFocus Parameters command, p. 98 for details.

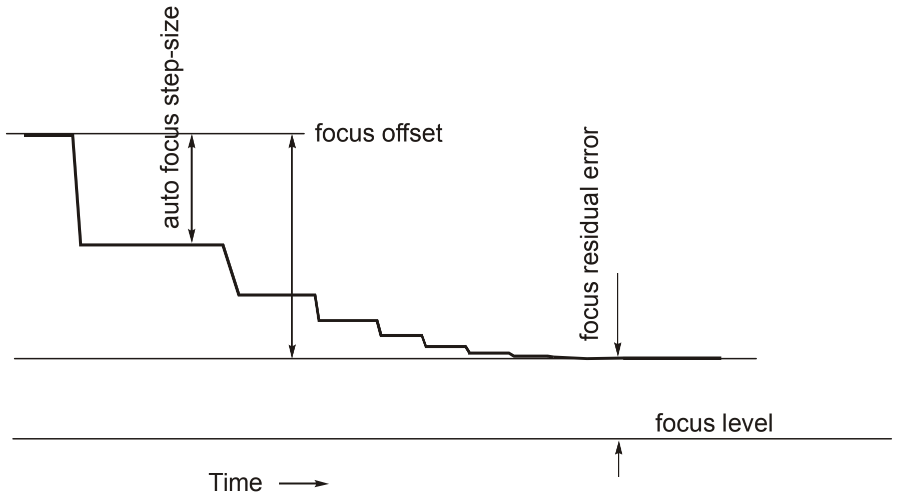


Fig. 25: Initial step-size too small

8.2 Holding the Focus Position

It is possible to use the controller's servo-loop to hold the in-focus position determined in 8.1. For that purpose the servo parameters of the involved axis are changed so that all internal sensors are disregarded in the sensor-channel-to-axis matrix, and the external sensor alone has a non-zero contribution. See Section "Sensor Coordinate Transformation" on p. 24 for more information about the matrix and the corresponding parameters. The following description is based on *NanoCapture*™ since changing the servo parameters can be most conveniently done using this program (see the *NanoCapture*™ Manual for more information).

To use the servo-loop to hold the focus position, proceed as follows:

1. Establish communication with *NanoCapture*™ and click OK in the *Device Interface* window, if you have not already done so.
2. From the *Config* menu, open the *Device Interface* window and make sure the *Native* command set is selected.
3. Make sure that servo-control is ON and that current position is very near the focus point (e.g. using the procedure described in Section 8.1)
4. Set servo OFF for the affected axis.
5. Configure the servo parameters for the axis (the required password is "E-710.3CD.4CL"):
   Use the *Config → Device Parameter Configuration* menu item to open the *Device Parameter Configuration* window. In the list that opens, select the "Servo *n*" entry for the appropriate axis (*n* = axis number). Edit the *Position from sensor* … fields so that all entries are 0 except that for sensor 7 (the analog input), which must have a non-zero value. If the voltage on the analog input increases when the axis is moved in

positive direction, use +1, otherwise -1. (If the focus point is at minimum or maximum of the signal, then it cannot be used here.)

6. Set servo ON for the axis and command it to target position 0 (e.g. with
   *n*MA0
   where *n* = axis number.

See Section 7.3.3 “Full-Time Autofocus” for more information.

9 External-Axis Flatness Compensation

9.1 Concept

The external-axis flatness compensation feature is present on the 6-axis versions only. It was conceived for users who have part or all of their PZTs mounted on a moving platform whose motion is not seen by the E-710. It is further assumed that the motion of this platform is controlled in a maximum of two directions (external axes 1 and 2) and that its position in a third direction (the “flatness compensated” direction) is a function of its position in the other two. This corresponds to the situation of an X-Y stage with a repeatable flatness error (Z runout).

9.2 Implementation

The flatness compensation feature works on the basis of a table-lookup, so the function is completely arbitrary. There are E-710 commands for:

- Specifying the size of the external-axis grid over which flatness compensation is to be defined (see EC, p. 106)
- Specifying the resolution of the external-axis grid (see NP, p. 129)
- Associating the 3rd external axis (the flatness-compensated axis) with one of the E-710 logical axes (see D... Set/Read Flatness-Compensated Axis, p. 100)
- Entering the correction values (deltas) for each grid point (see DZ, p. 105)
- Applying a correction delta (adding a specified delta to E-710 target of the flatness-compensated axis (see EC, p. 106)

9.3 Example

A PZT Z-stage is mounted on a motorized X-Y micropositioning stage with a travel range of 12 X 12 mm. Because of bumpiness in the X-Y plane, X and/or Y motion causes small, repeatable errors in Z-direction. These errors are outside of the servo-loop because the Z-sensor is mounted inside the Z-stage.

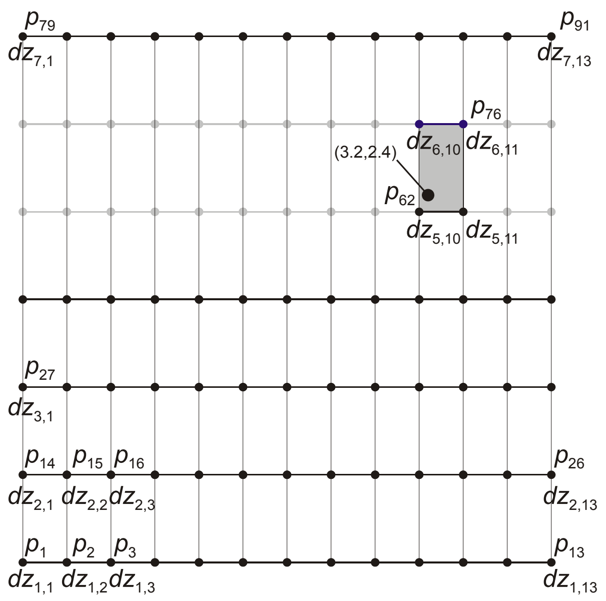


Fig. 26: Flatness compensation grid points and correction deltas

Suppose, for example, we take 13 points in the X-direction and 7 points in the Y-direction, forming a grid 1 mm in X by 2 mm in Y.

First enter the size and number of points that make up the grid in each direction. Then move the external X and Y axes successively to each grid point (p_1 ... p_{91})

and measure the flatness error at each point with an external measuring instrument. Designate the errors as {$dz_{1,1}, dz_{1,2}, \ldots dz_{i,j} \ldots dz_{7,13}$} as shown below.

Enter these errors (deltas) so that they can then be compensated on command.

The commands necessary are as follows:

101**EC**-6	Set the lower limit for external axis 1
201**EC**6	Set the upper limit for external axis 1
102**EC**-6	Set the lower limit for external axis 2
202**EC**6	Set the upper limit for external axis 2
1**NP**13	Set the number of grid points on external axis 1
2**NP**7	Set the number of grid points on external axis 2
3**DP**0,163**DW**1.0	Specify the E-710 logical axis to be corrected as axis 3
3**DP**-1,163**DW**1.0	Alternative: Specify the E-710 logical axis to be corrected and save specification as power up default

The controller sets up an internal table with 13 x 7 = 91 points to store the correction deltas. The correction deltas are numbered 1-91 in the order $dz_{1,1}$, $dz_{1,2}$... and assigned to grid points in the order (X_1, Y_1), (X_2,Y_1), etc. The values are entered into the controller with a series of DZ commands:

1**DZ**$dz_{1,1}$	Save correction delta for first point. The internal counter is set to 1
1**DZ**$dz_{1,2}$	save next point; (the internal counter increments)
1**DZ**$dz_{1,3}$	save next point.
...	...
1**DZ**$dz_{4,6}$	save next point
...	...
1**DZ**$dz_{7,12}$	save next point.
1**DZ**$dz_{7,13}$	save last point.

Now the correction deltas are all stored in the controller.

The spacing, $grid_X$ in X-direction and $grid_Y$ in Y-direction, are calculated. Any point (x,y) within the specified range is assigned the correction delta for point (x, y), where $x = N*grid_X + X_{min}$ and $y = M*grid_Y + Y_{min}$, is saved at point number M*13+N+1.

If the external axes are moved to position X = 3.2 mm, Y = 2.4 mm and the E-710 sent following commands:

1EC**3.2**

2**EC**2.4

the amount $dz_{5,10}$ will be added/subtracted from the Z-position target of the E-710-controlled Z-stage as soon as the 2**EC**xxx command is executed.

The input coordinate must be in the predefined range for compensation to be applied.

10 Wave Generator

The E-710 has the capacity for creating and storing arbitrary waveforms. These waveforms can then be output by up to two wave generators on one axis each. A waveform can be output once, a fixed number of times, or repeated indefinitely. Wave definition and output is controlled by the command interpreter, but the each wave generator outputs target values directly to the servo-controller(s) for the axis to which it has been “connected”. Note that these target values are relative positions, i.e. they are added to any other current target contributions coming from move commands and / or from the analog input (for details see Section "Calculation of Target Positions" beginning on p. 26).

There are also a number of digital signals for use in conjunction with the wave generator (see Digital Trigger Signals on p. 31, under “Communication” for details).

The different E-710 software interfaces also support use of the Wave Generator. In addition, waveforms can be defined, stored and displayed in and by the software, making a more user-friendly Wave Generator architecture possible. If using the Wave Generator with the GCS DLL, *PZTControl, NanoCapture*™ or LabView, read the descriptions in the associated software manual first and see also the examples in the \Samples directory of the product CD (download the latest versions of software and manuals from www.pi.ws).

Using the wave generators with the native commands over the ASCII interface is a lengthy multistep process:

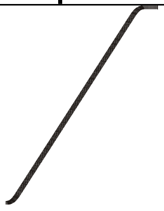
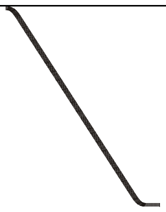
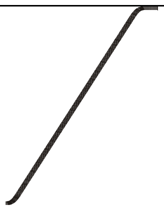
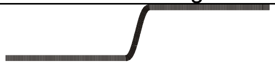
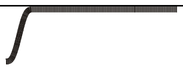
1. Fill the datapoint memory with values. A maximum of 62464 points can be accommodated (63488 points in 3- and 4-channel versions). To aid in filling the table, commands are provided to divide point storage into *segments* and to place *curves* (generic wave shapes like ramps, sinewaves...) appropriately in the respective *segments.* The curve starting point and the length of speed-up/slow-done zones before and after the curve can be specified, as can the position of the curve centerpoint. Values can also be downloaded one at a time, permitting fully arbitrary definitions.
2. Group the data points into *waveforms*. A *waveform* is made up of neighboring points, just like a *segment*. In fact, if desired, the *segments* defined in step 1 can be used as *waveforms*: *segment1* will be interpreted as *waveform1,* etc. A maximum of 8 *waveforms* can be defined. The *waveforms* are contiguous and may not overlap, but need not include all of the data points (run to the end of memory).
3. Assign a *waveform* to a *wave generator*. This is a one-to-one assignment, but can be changed without redefining the *waveforms*.
4. Set the various wave output parameters (triggers, ...).
5. Assign the wave generator to an axis (if synchronous move is enabled, then up to two axes will move together).
6. Make sure the axes to be moved have their curve-control properties set so as to be curve moveable (see the D... Set/Read Curve Control command, p. 99).
7. Issue a wave generator move command (MC for multiple cycles – see p. 124 – or SC for one cycle – see p. 134).

You should also consider the DDL (digital dynamic linearization) feature which is standard on the 6-axis versions and available as option E-710.SCN on the other versions. It permits enhanced output precision in conjunction with the wave generator’s periodic motion. DDL works by monitoring the motion during an initialization phase and refining the output voltages so as improve the resulting motion. This feature makes possible ultra-precise scanning operations. DDL is described in Section 11, beginning on page 61.

10.1 Symmetrical Scanning Waveform Definition Example

In this example, the commands necessary to load datapoint memory and define two waveforms will be explained. If the two axes are in the XY plane, the resulting motion will be that of a "back-and-forth" XY scan.

10.1.1 Define and Fill Segments

Command	Description	Results	Graph
1PT562 1CP512 1PC256 1PS50 1PA0 1FO0 1GL150	Set the total number of points in segment1 Set the curve points in segment1 Set the curve center point of segment 1 Set the speed-up and slow-down points Set the curve start point Set the curve offset Generate a single scan line	Segment 1, Memory Addr. 1– 562 Segment 1, offset = 0, Amplitude=150	 Segment1
2PT562 2CP512 2PC256 2PS50 2PA0 2FO150 2GL-150	Set the total number of points in segment 2 Set the curve points in segment 2 Set the curve center point of segment 2 Set the speed-up and slow-down points Set the curve start point Set the curve offset Generate a single scan line	Segment 2, Memory Addr. 563–1124 Segment 2, offset = 150, Amplitude=-150	 Segment2
3PT562 3CP512 3PC256 3PS50 3PA0 3FO0 3GL150	Set the total number of points in segment 3 Set the curve points in segment 3 Set the curve center point of segment 3 Set the speed-up and slow-down points Set the curve start point Set the curve offset Generate a single scan line	Segment 3, Memory Addr. 1125– 1686 Segment 3, offset = 0, Amplitude=150	 Segment3
4PT1024 4CP100 4PC0 4PS50 4PA462 4FO0 4GL20	Set the total number of points in segment 4 Set the curve points in segment 4 Set the curve center point of segment 4 Set the speed-up and slow-down points Set the curve start point Set the curve offset Generate a single scan line	Segment 4, Memory 1687-2711 Segment 4, offset = 0, Amplitude=20	 Segment4
5PT662 5CP100 5PC0 5PS50 5PA0 5FO20 5GL20	Set the total number of points in segment 5 Set the curve points in segment 5 Set the curve center point of segment 5 Set the speed-up and slow-down points Set the curve start point Set the curve offset Generate a single scan line	Segment 5, Memory 2712-3274 Segment 5, offset = 0, Amplitude=20	 Segment5

10.1.2 Define Waveforms

After generation of the 5 curve segments the *waveforms* for the wave generator(s) can be defined with following commands:

0PT0	Switch from segment definition mode to waveform definition mode
1PT1686	Define first waveform as first 1686 points (segments 1-3)
2PT1686	Define second waveform as next 1686 points (segments 4 and 5)

Note that the waveform points are taken in order (only the length is specified in the commands).

Both waveforms have the same length and, if run at the same time, will stay in phase with each other.

10.1.3 Resultant Segments and Waveforms

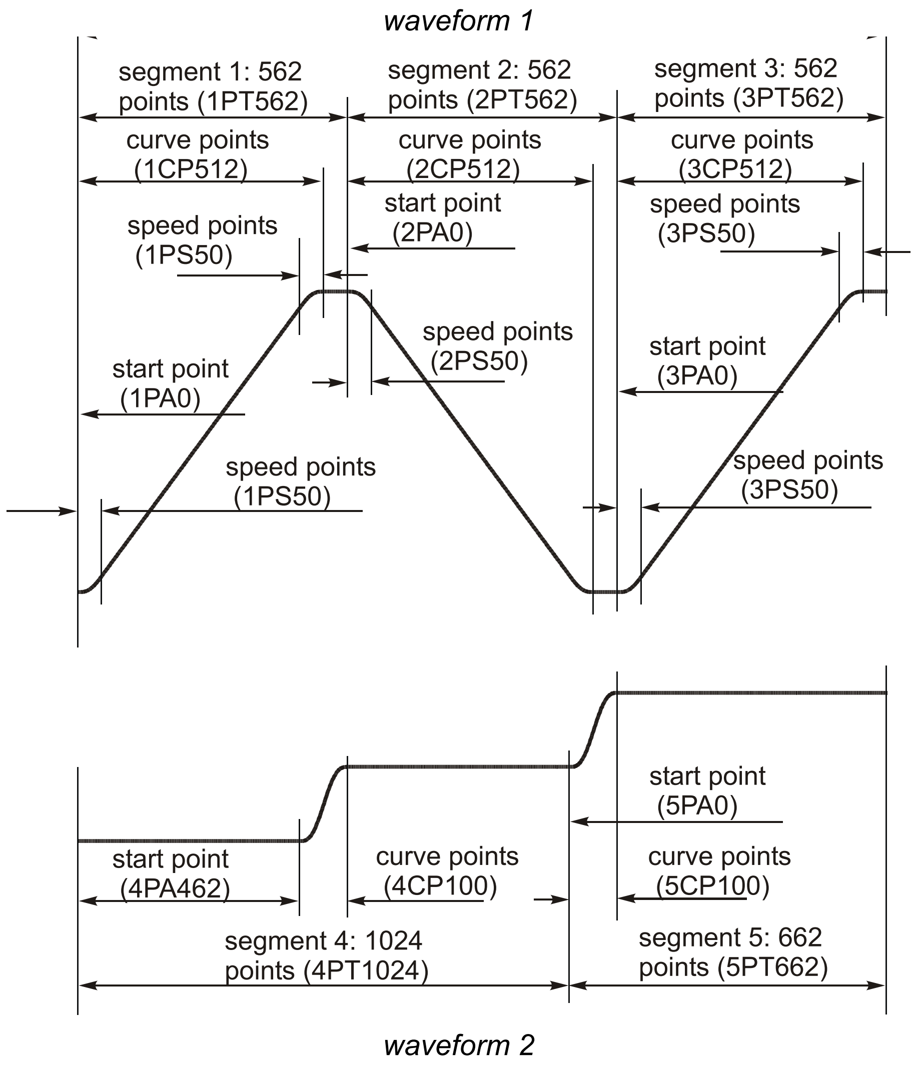


Fig. 27: Waveforms and segments in example

If *waveform 1* is applied to the X axis and *waveform 2* to the Y axis, the motion shown below would result:

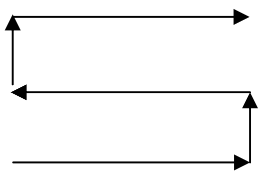


Fig. 28: "Back-and-forth" X-Y scan

10.2 Unidirectional Scanning Waveform Example

In this 90 x 90 x 90 µm scan the X-axis scans are all in the same direction (forward) with rapid returns and concurrent Y-steps interspersed. During the constant-speed phase of the scan, a trigger synchronized external measurement is effected (see Fig. 30).

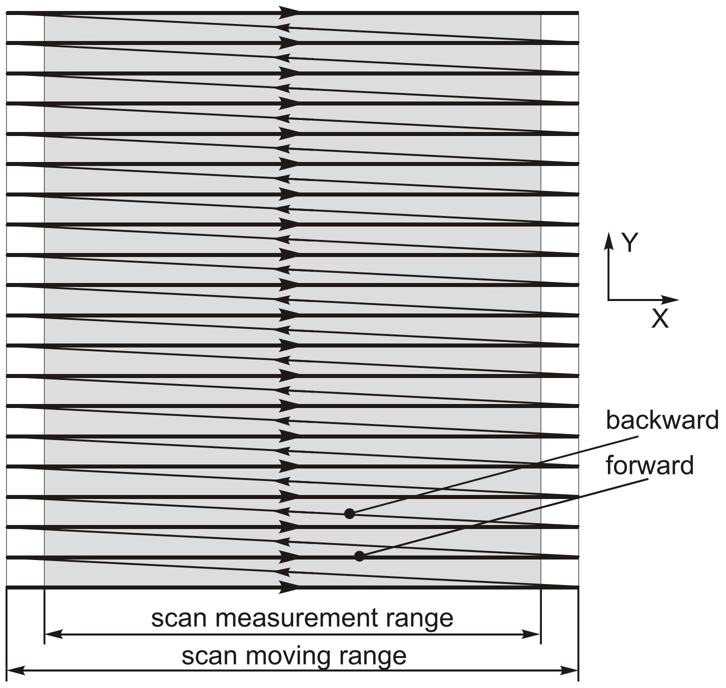


Fig. 29: Unidirectional scanning pattern, showing zone where external measurement is enabled

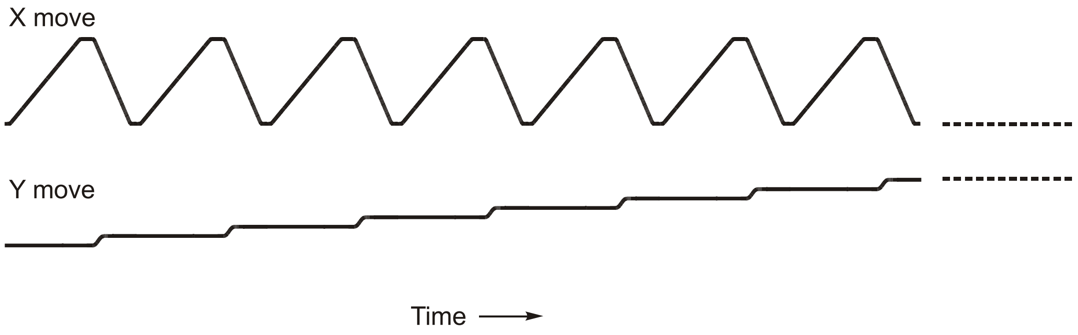


Fig. 30: Unidirectional scan X and Y move magnitudes

For this application, two different waveforms can be defined and the scan procedure can thus run automatically without a heavy data communication load. Trigger signals are used to synchronize the external sampling. The desired scan frequency is about 1 Hz (total 5000 points).

10.2.1 Define and Fill Segments

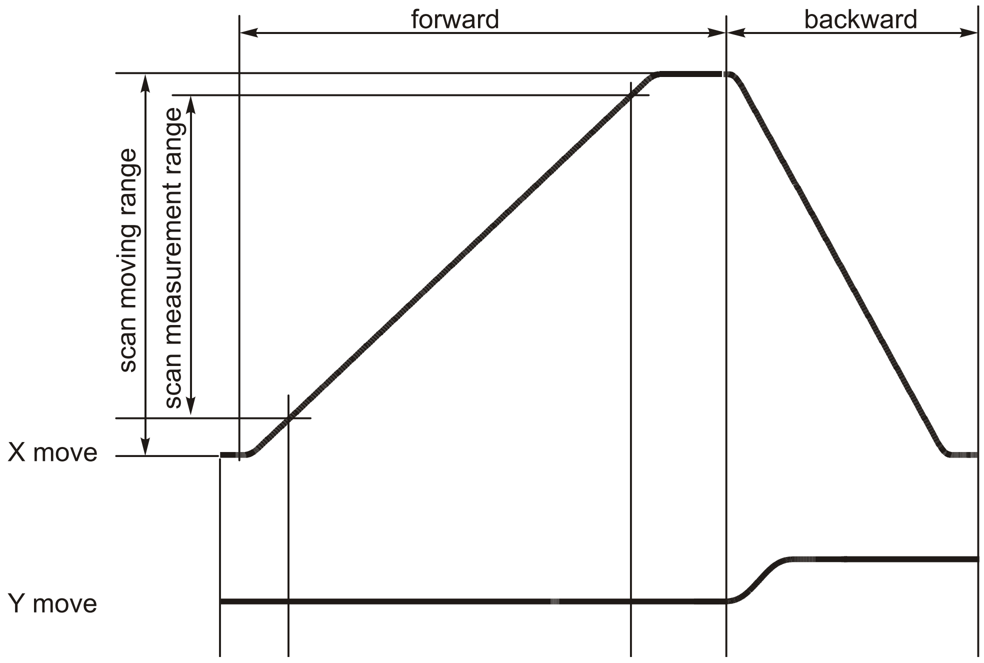
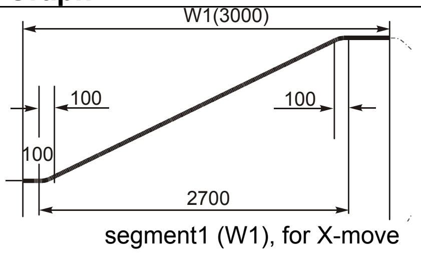
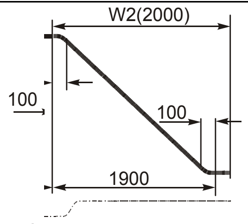
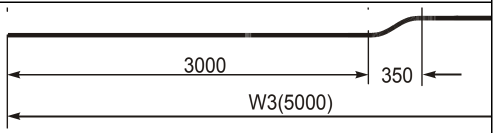


Fig. 31: X and Y waveforms

Command	Description	Results	Graph
0PT0 1PT3000 1CP2700 1PS100 1PA100 1FO5 1GL90	Initialize segment definition process Set total number of points in segment 1 (W1) Set number of curve points in segment 1 Set number of speed-up/slow-down points Set curve start point Set curve offset to 5 µm Generate curve (single scan line)	Segment 1 (W1), for X-move forward scan, Memory Addr. 1– 3000 Segment 1, offset = 5, Amplitude=90	 segment1 (W1), for X-move
2PT2000 2CP1900 2PS100 2PA0 2FO95 2GL-90	Set total number of points in segment 2 (W2) Set number of curve points in segment 2 Set number of speed-up/slow-down points Set curve start point Set curve offset to 95 µm = segment 1 end position Generate curve (single scan line, negative slope)	Segment 2 (W2), for X-move return, Memory Addr. 3001–5000 Segment 2, offset =95, Amplitude=-90	 Segment2 (W2), for X-move
3PT5000 3CP350 3PS100 3PA3000 3FO0 3GL1	Set total number of points in segment 3 (W3) Set number of curve points in segment 3 Set number of speed-up/slow-down points Set curve start point to coincide with end of X-move forward scan Set curve offset to 0 Generate curve (single scan line)	Segment 3 (W3) for Y-move (step), run concurrently with W1-W2, memory addr 5001-10000 Segment 3, offset=0, Amplitude=1 µm	 Segment3 (W3), for Y-move

10.2.2 Define Waveforms and Connect Generators and Axes

After generation of the 3 curve segments, the *waveforms* for the wave generator(s) can be defined; segments 1 and 2 will be combined to make waveform 1, segment 3 alone becomes waveform 2

0**PT**0	switch from segment definition mode to waveform definition mode.
1**PT**5000	define first 5000 memory points as waveform 1; this combines segment 1 (W1) and segment 2 (W2)
2**PT**5000	define the next 5000 memory points as waveform2; this is where W3 is stored.

Note that the waveform points are taken in order (only the length is specified in the commands). Both waveforms have the same length and, when run at the same time will stay in phase with each other.

1**SF**1	connects waveform 1 with wave generator 1
2**SF**2	connects waveform 2 with wave generator 2
1**CF**1	connects axis 1 to wave generator 1
2**CF**2	connects axis 2 to wave generator 2

10.2.3 Synchronize Triggers

There are 4 programmable output trigger signals (DIGITAL I/O: D_OUT2, D_OUT3, D_OUT4, D_OUT5) which can be used to synchronize the external measurement sampling. For more information about these trigger lines see "Programmable Output Trigger Signals" Section on p. 57 and the descriptions of the KT command (p. 120) and the FT command (p. 109).

In this example trigger signal 1 and 2 are used. Every sampling point will be started with a positive pulse trigger signal, the sampling will be started at point 400 and stopped at 2600, the range in which the scan speed is constant. The trigger 1 is set up to provide sampling pulse outputs, trigger 2 to provide a positive-level for the entire sampling period (see Fig. 33).

1**KT**1	trigger 1 is positive pulse.
2**KT**2	trigger 2 is positive level.
0**FT**0	clear all trigger points.
400**FT**3	trigger point 400 set to ACTIVE for 3 = 0011 binary (triggers 1 & 2).
2600**FT**259	trigger points 401 to 2600 set to ACTIVE for 259 = 256+3 = 1 0000 0011 binary (bit 9=1 for setting intermediate points, bits 1 & 2 for triggers 1 & 2).

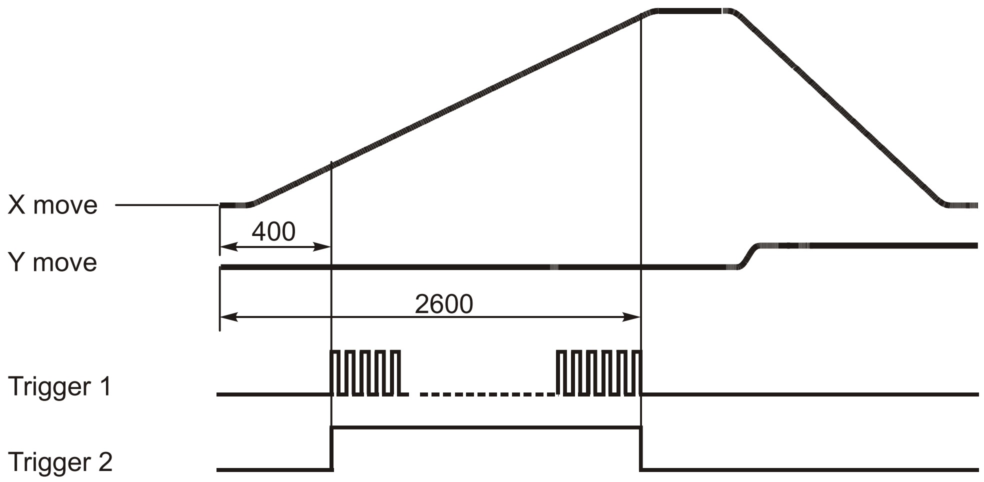


Fig. 32: Unidirectional scan, triggering external measurements during constant velocity phase

10.2.4 Running the Scan

For the XY scan, both waveforms must be output synchronously. The connected axes (here axis 1 and 2) must have "enabled for synchronous move" set in the axis property curve control.

Use D... Set/Read Curve Control if necessary (see page 99 for details).

To trace one scan line, send one of the following commands:

0**SC**32	for one scan line with reinitialize DDL off (32=10 0000 binary), or
0**SC**49	for one scan line with reinitialize DDL on (49 = 11 0001 binary)

The scan movement is: axis 1 (X-axis) start from 5 µm and move to 95 µm and come back to the start position. Axis 2 (Y-axis) makes first no movement (hold current position). As axis 1 starts to move back, axis 2 moves one small step of 1 µm and holds at that position, which would be is the starting position for the next scan line.

Repeat the **SC** command 90 times (settling time, here 1 sec., between every scan line has to be taken into account) to scan a 90 µm x 90 µm XY area.

It is possible to scan the entire 90 x 90 area with fewer commands:

0**SC**32,**WA**1000,**RP**90	where the RP90 command repeats the other commands in the compound command line 90 times

or,

0**SC**49,**WA**1000,**RP**90	where SC49 performs the move with DDL, if available

When RP is used at the end of a compound command, the controller will provide no interface responses for the duration of the scan (90.18 sec).

Another alternative is setting the repeat value first and then using the continuous run command (**MC,** p. 124**)**

(For 6-axis versions, firmware rev. 2.01 or higher required, for all other E-710s firmware rev. 5.011 or higher, except 6.010)

RN90	set repeat number
0**MC**32	start continuous move from both wave generators, without DDL

or

0**MC**49	start continuous move from both wave generators (with DDL if available)

With the RN-MC syntax, the controller can still communicate over the ASCII interface while the scan is running.

10.3 Programmable Output Trigger Signals

There are 4 programmable output trigger signals (DIGITAL I/O: D_OUT2, D_OUT3, D_OUT4, D_OUT5). These signals can only be used with the internal wave generator. The trigger position can be set to any point in the waveform. Each trigger output can be programmed as:

- Positive pulse,
- Negative pulse,
- Positive level
- Negative level

For more information about how to program the output trigger lines see the descriptions of the KT command (p. 120) and the FT command (p. 109). For the output trigger timing see Fig. 36 on p. 59.

Example:

The waveform is a ramp signal to be used for a scanning application. During the linear range (where the velocity is constant) an external sensing system is to be started to sample measurement results. To accomplish this, a "start pulse" (trigger 1) will be set to a point somewhere at the linear portion, and a "stop pulse" (also on trigger 1) set at a point near the end of the linear portion. Another trigger signal (trigger 2) will be programmed as a level trigger signal indicating the valid time for the external sampling. To show the various programming possibilities, trigger 3 and trigger 4 will be programmed the same as trigger 1 and trigger 2, but with inverted logic. The servo update rate (cycle time) is 200 µs.

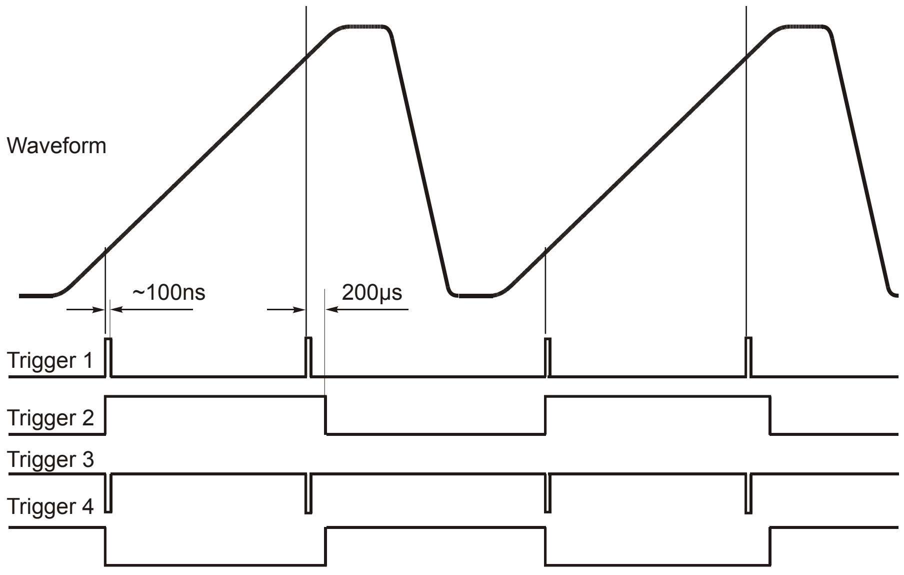


Fig. 33: Programmable output trigger example.

10.4 Input Trigger Signals

The following input trigger signals can be used to trigger the internal wave generator, depending on the run-mode specification in the SC or MC commands:

- Generator start (DIGITAL I/O: D_IN1):
  If enabled, the wave generator starts with the first curve point whenever a rising edge is detected on this line.
- Generator hold (DIGITAL I/O: D_IN2; 6-axis version: firmware rev.2.11 or newer, 3- and 4-axis version: firmware rev. 5.030 or higher, but not 6.0 to 6.029):
  If enabled, the generator stops at the current point when a falling edge is detected on this line and continues from that point when a rising edge is detected (optionally).

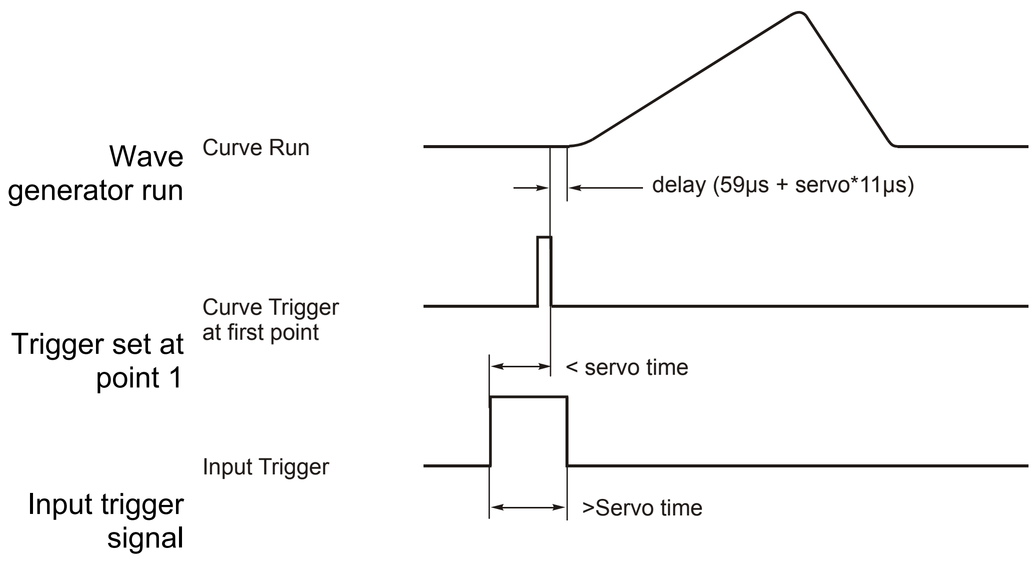


Fig. 34: Input trigger timing: servo cycle time is 200 µs

The input trigger signals are disabled by default. To enable the input trigger signal, write the proper value to memory address 231. Use the "D... Set/Read Input Trigger Mode" command, p. 101, to do this. If stored in non-volatile memory, it need be done only once.

Examples 1:

1DP0,231DW2	enable generator start line (D_IN1) for anytime
1DP0,231DW514	for one-time only (E-710.6CD only)

Example 2:

The resulting setup in Example 1 is in effect only until next power-down. To make the new values the new power-up defaults, use the DP-1 option to write them EEPROM:

1DP-1,231DW2	enable generator start line (D_IN1) for anytime
1DP-1,231DW514	for one-time only (E-710.6CD only)

Example 3:

1DP0,231DW4	enable generator hold line (D_IN2) for generator hold
1DP0,231DW6	enable generator hold line (D_IN2) for generator hold and generator start

10.5 Data Recording

During the wave generator output, various types of data can be saved internally to report tables, which can later be read out to check on performance (see the TT command, p. 144). Values are recorded in the report table with the same frequency as the output trigger signal. The report table can be programmed to collect values like current target position, real position value, position error, DDL data or PZT drive voltage.

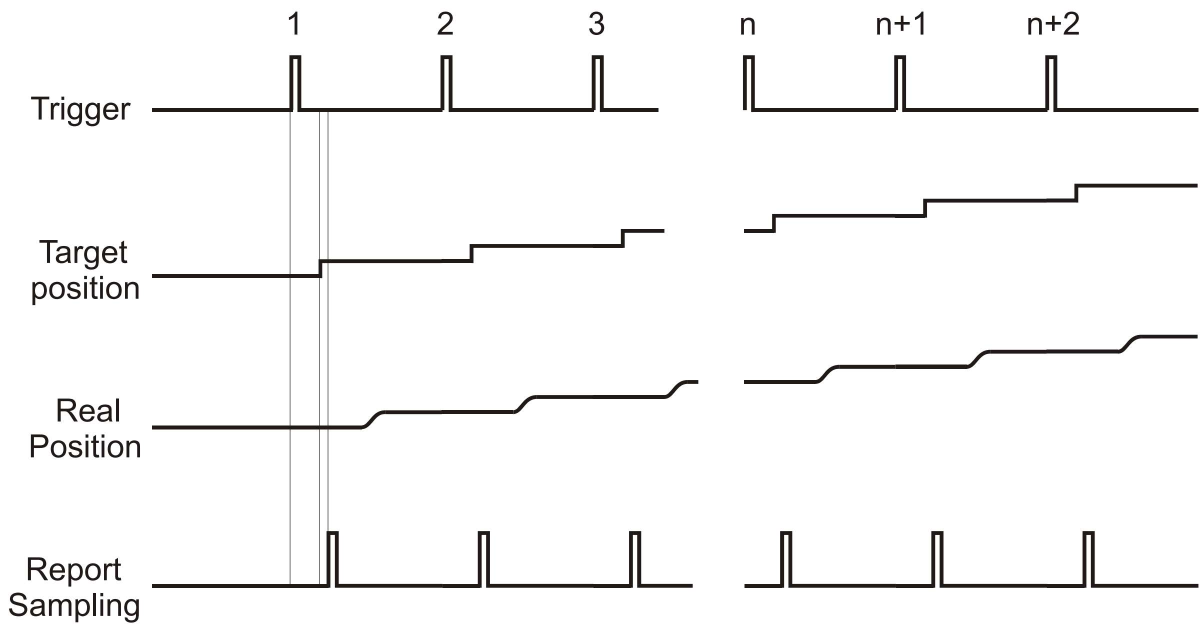


Fig. 35: Report table and output trigger timing

10.6 Wave Parameters and Commands

The following table summarizes the user-settable parameters and the commands used to set them:

Segment Parameters	**Command**	**Page**
Set total number of points in segment	PT	132
Set (amplitude) offset	FO	108
Download data points (fill segment)	FS	109
Curve Parameters		
Set number of points for curve	CP	95
Set curve relative start point	PA	131
Set curve center point	PC	131
Set speed-up / slow-down zone length	PS	132
Generate (up-down) ramp curve	GC	111
Generate single scan line	GL	115
Generate sine wave	GS	118
Waveform Parameters		
Set output triggers	FT	109
Specify length of next waveform (after finishing segment definitions)	PT	132
Axis Parameters		
Reset baseline position	MD	127
Configure output trigger type	KT	120
Set/read curve-control properties (e.g. synchronous move)	D...	99
Set/read proportional gain	D...	104
Set/read motion report value filtering	D...	101
Wave Generator Parameters		
Table rate *sampling interval* (affects all table-related commands)	TR	143
Repeat waveform cycles	RN	133
Stop wave generator output	RT	134
Output waveform once	SC	134
Assign waveform to wave generator	SF	136
Output waveform repeatedly	MC	124
Make motion report	TL	141
Configure input triggers	D…	101

11 Dynamic Digital Linearization (DDL)

11.1 Introduction

Dynamic Digital Linearization (DDL) is standard on 6-axis E-710s with firmware 2.0 and higher and a firmware option (order number E-710.SCN) available for 3- and 4-axis E-710 digital PZT controllers having firmware versions 5.017 and higher but not 6.0 to 6.016. This Section describes those functions related to DDL.

DDL makes it possible to achieve significantly better position accuracy for dynamic applications with periodic motion such as scanning. With DDL it is even possible to operate some PZTs at or near their mechanical resonant frequencies, although extreme care must be taken never to allow the PZT to resonate out of control. DDL is most effective when the operating conditions do not vary from one motion cycle to the next.

DDL works in conjunction with the E-710 Wave Generator (see Section 10 beginning on page 50). It works by “observing” axis motion over one or more wave generator output cycles (DDL initialization), then it uses the information gathered to refine the control output signals. During the DDL initialization phase, a *DDL table* is filled with data, which is then processed and can be used to condition future output.

The DDL feature is activated at the factory before shipment except on 3- and 4-axis versions which were not expressly ordered with the DDL option. In such case you can activate it after purchase and without opening the device (see Section 11.3.1, p. 66).

In any case you will be provided with a device-specific license number which is needed for activating the DDL option. For security reasons, this number will be communicated to you separately. If you do not feel the need to keep it confidential, write it in this manual.

11.2 Working Principle

The residual error of a standard PZT servo controller is practically zero at very low speeds but increases with the operating frequency.

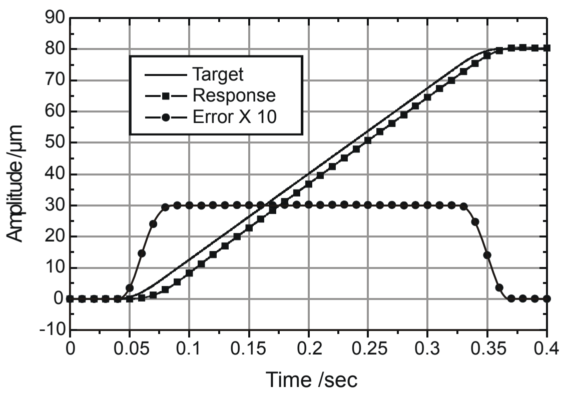


Fig. 36: Tracking error

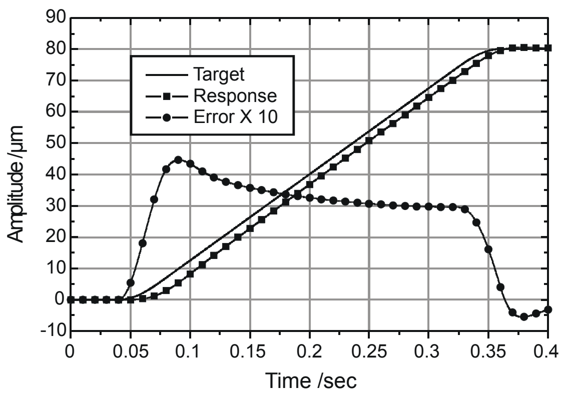


Fig. 37: Tracking error

In addition, for linear bi-directional scanning applications, the residual error is not a constant, as for typical linear PID servo systems (see Fig. 37), but changes with scan position as shown in Fig. 38. This is caused by the PZT's non-linearity and the servo's limited dynamic performance. As a result of these factors, the scan position is not exactly proportional to time but exhibits dynamic non-linearity.

For periodic scan functions it is possible to record the errors during one or more DDL initialization periods and then compensate for them in all subsequent periods. The principle of DDL is thus the inclusion of a regulating error-feedback loop in addition to the standard proportional-integral (P-I) servo-control loop. The error-compensation data is calculated with real-time error regulation when the DDL mode is enabled. The compensation output is so calculated that the actual response shows lower errors.

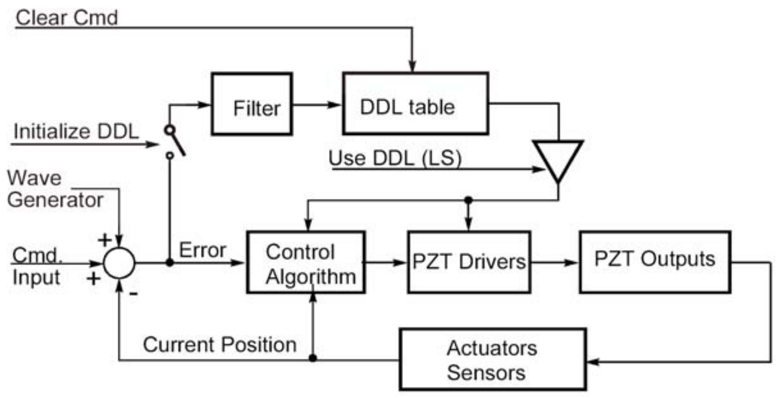


Fig. 38: DDL block diagram

Example:

A simple 400 µm forward and backward scan at 2.5 Hz. Even though the frequency is not high, there is still a static error of about ±10 µm and a dynamic error of about ±5 µm. The dynamic error indicates that the scan speed is not constant. This error will deteriorate the scanning accuracy.

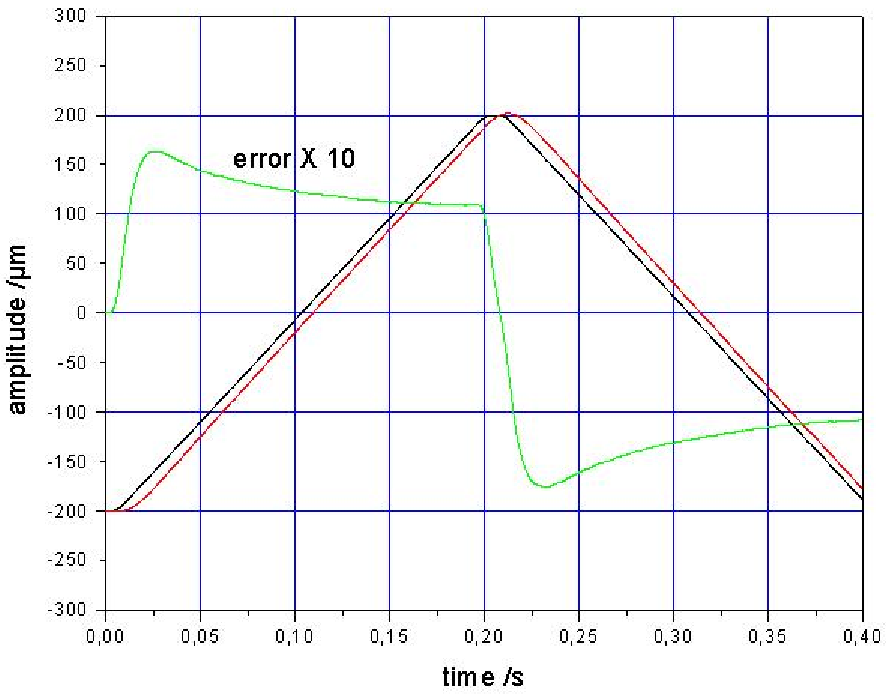


Fig. 39: Without DDL

With DDL, after one period the error for constant-speed motion will be reduced by a factor of about 100 and the dynamic error by a factor between 20 and 90.

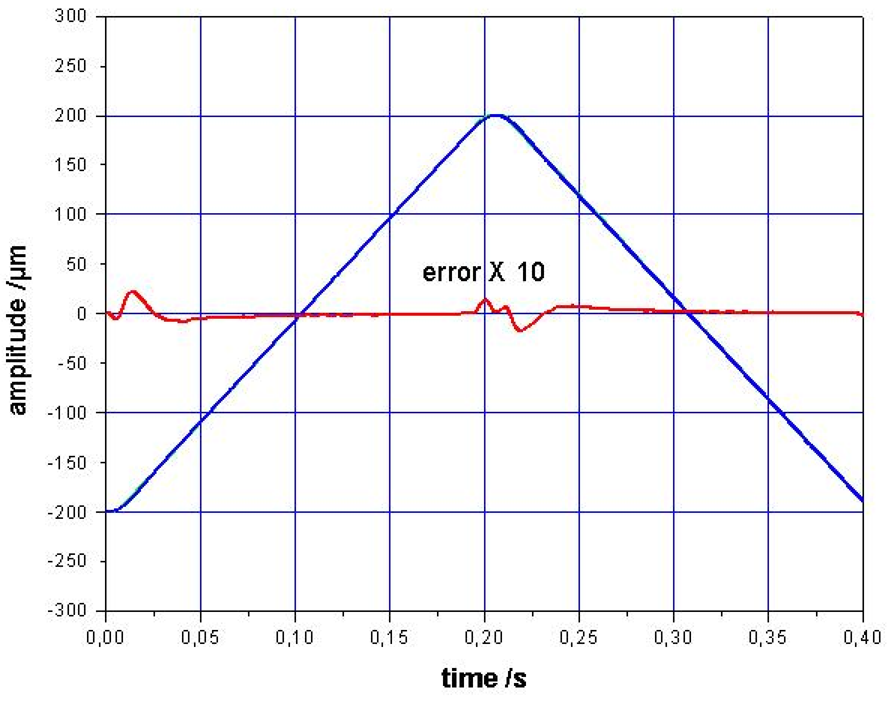


Fig. 40: With DDL, one initialization cycle

With more DDL initialization periods, the dynamic error can be reduced by a factor of many hundred.

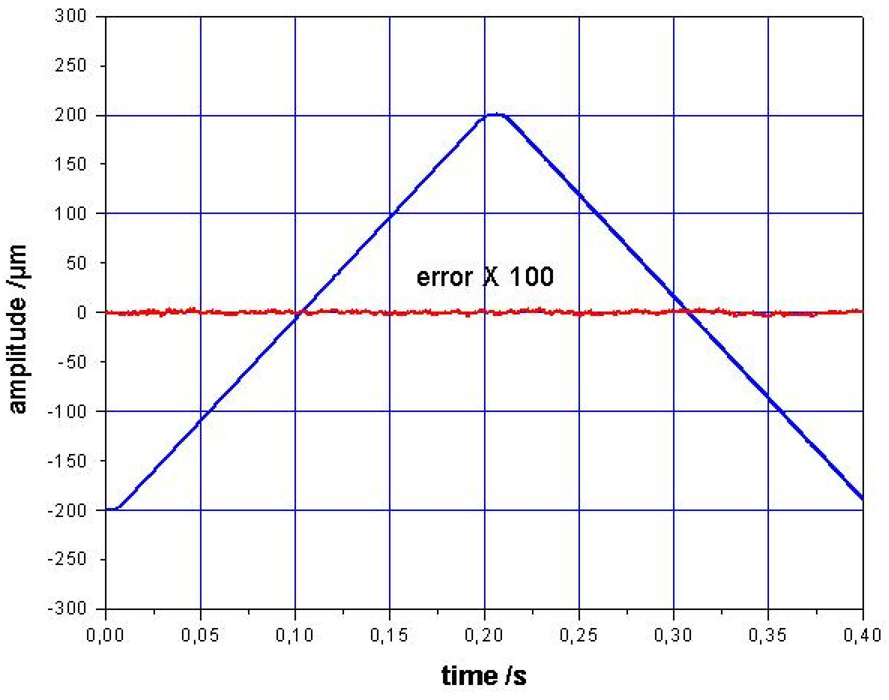


Fig. 41: With DDL, several DDL initialization cycles

The DDL feature can also be used when the motion is at the resonant frequency. At resonance, the P-I controller no longer has sufficient loop gain to keep the error small enough. On the contrary, the error can even be larger than the target value, as shown in Fig. 43.

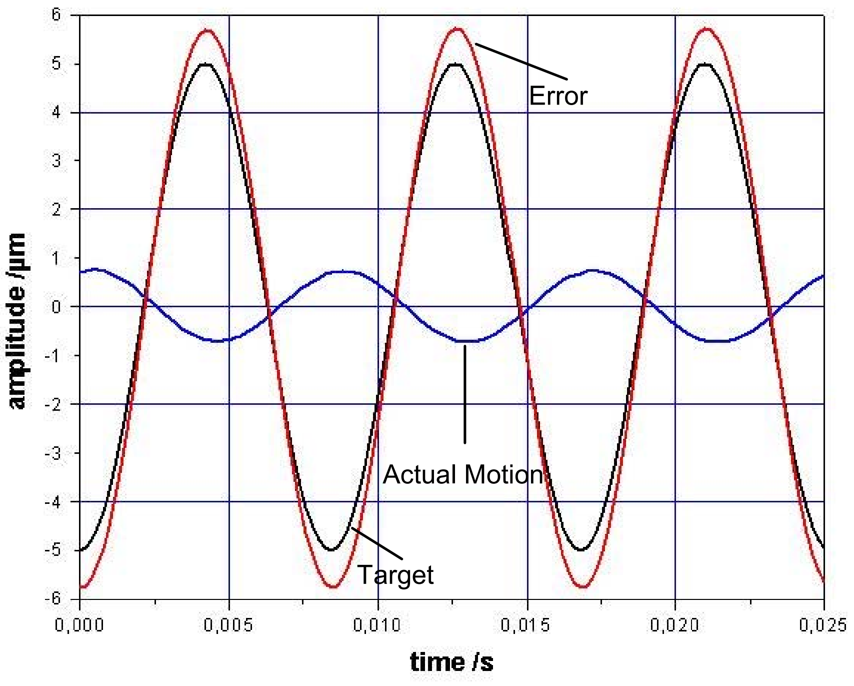


Fig. 42: High P-I error at resonant frequency

DDL, however, is still able to reduce the error.

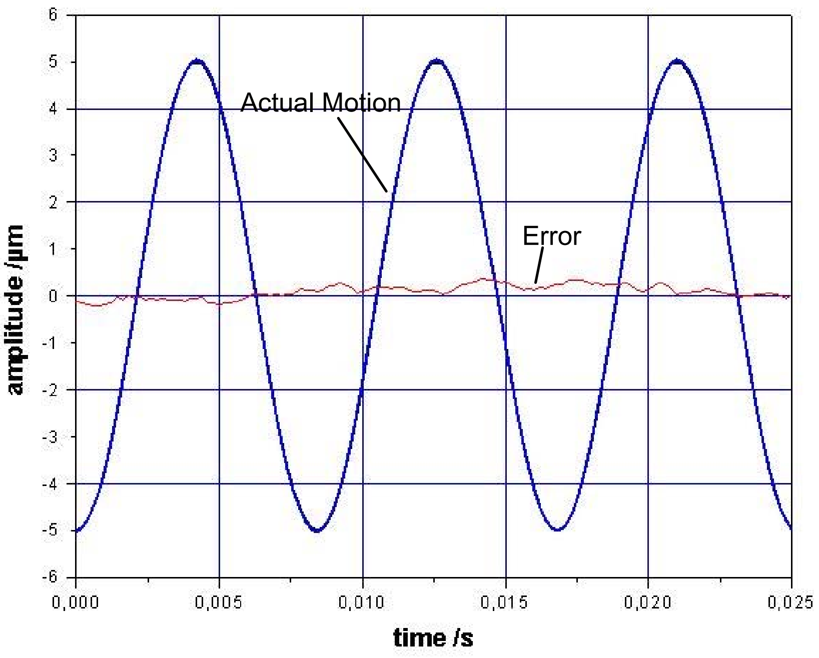


Fig. 43: Error at resonant frequency reduced by DDL

11.3 Using DDL

You can start and use DDL either using the E-710 ASCII commands directly as described in this Section or using the *NanoCapture*™ graphic user interface (release 3.11 or newer; see the *NanoCapture*™ User Manual).

Make sure that the load which will be used in your application is mounted on the stage.

CAUTION

Most piezo actuators that can be connected to the E-710 can be destroyed by uncontrolled oscillation near the mechanical resonant frequency. If you observe resonance while configuring your system, switch off power to the actuators concerned immediately and check the settings and servo-control parameters.

NOTE

Using the DDL could be critical if there are any residual oscillations in the system (e.g. there is a third resonant frequency which can not be suppressed by the two E-710 notch filters). If this is the case, using the DDL will cause the oscillations to build up—the more learning cycles, the stronger the effect. The easiest way to check if residual oscillations are present is to perform a step response measurement in *NanoCapture*™ (see p. 74 ff).

The following command survey lists all DDL-related commands and some general commands with options relevant to DDL. For the complete command reference see Section “Native Command Reference (alphabetical)” beginning on page 91, where the DDL options of the general commands are emphasized with bold type.

Command	Description	Type	Page

D...	Set/Read Curve Control	Control/Report	99
D...	Set/Read DDL license Status	Control/Report	102
CL	Clear DDL data	DDL	95
FL	Download DDL Data or Fill DDL Table	DDL	108
LP	DDL Initialization Phase Correction	DDL	121
LS	DDL Setup	DDL	121
LT	Define DDL Data Table	DDL	122
LN	DDL Repeat Number and Response Value	DDL	120
MC	Output Waveform Continuously	Waveform Move	124
SC	Single Cycle Output	Waveform Move	134
TL	Test curve moving / DDL Report	Waveform Move	141

11.3.1 Activating the DDL license

When a 3- or 4-channel controller is upgraded with the DDL feature, the DDL license must be entered and the DDL feature must be activated. See the “D... Set/Read DDL license Status” command on p. 102 for how to proceed.

Note: Entering an incorrect license number may deactivate the DDL license.

11.3.2 Operating Dynamic Digital Linearization (DDL)

Make sure that the following servo-controller parameters are adjusted properly so that the step response of the stage does not show any overshoot or ringing:

- Notch frequency
- Servo-loop time constant
- Servo-loop slew rate limitation
- Servo-loop P-term (loop gain)

Whenever any of these values is changed you must recalculate the processing parameters used for DDL data,

Whenever any of these values is changed the LP command (see p. 121) must be executed in order to calculate the processing parameters for DDL data:

0**LP**0

In this example the LP command calculates the processing data for every axis. The calculated data is written into the volatile memory only. This means that the calculated data will be lost if the power of the device is switched off. In this case the LP command has to be executed after every power-up. This is useful if the servo-controller parameters changed very often.

If you are sure that the servo-controller parameters will not be changed for a long time then the LP command can be executed with the option that the parameter is written into the non-volatile memory so that the data will not be lost if the power is switched off. Read the description of the LP command for details (p. 121).

Define the desired waveforms, as described in Section “Wave Generator” beginning on p. 50. Two examples show how to start the generator with the DDL function running. In the first example only one generator and one axis are used. In the second example two synchronous running generators and two axes are used.

Example 1:

In this example generator number one will be used and it will be connected to axis number 3.

At first the desired waveform must be defined It is a sine function with an amplitude of 10 µm:

0**PT**0

1**PT**600

1**CP**600

1**PC**300

1**PA**0

1**PS**0

1**FO**0.0

1**GS**10.0

0**PT**0

1**PT**600

Select the waveform for generator 1:

1**SF**1

Connect the generator to the desired axis. In this example, generator 1 is connected to axis 3:

3**CF**1

Connect axis 3 to DDL table 1:

3**LT**1

Enable DDL usage for axis 3:

3**LS**1

Clear DDL table 1:

1**CL**0

Set axis 3 to servo-on mode (if axis 3 is in servo-off mode):

3**SL**1

Before wave generator output is started, the number of cycles can be limited to a number close to the value of the DDL repeat number. If the repeat number is set to 35, then the number of waveform periods to output can be limited to 50:

RN50

Check the curve control properties of axis 3 to make sure waveform move and DDL feature are enabled

3DP0,130DR

The response should be 240 (waveform move, synchronous move, DDL and synchronous DDL enabled).

Perform a DDL initialization run: start wave generator 1 with reinitialize DDL on for axis 3 (see MC command on p. 124):

1**MC**256 *switch* bit 8 ON, i.e. first bit for 3rd axis

The wave generator will stop after the number of cycles which were set by the RN command (see p. 133). The next time when the wave generator is started without reinitialize DDL on, but with the DDL data saved in DDL table 1, DDL will be in effect and the position error will be reduced.

Before the wave generator is started again, the number of cycles can be set to the value desired for the application. If the number of wave generator cycles is to be unlimited, use the following command:

RN0

Now start wave generator 1 without DDL reinitialization:

1**MC**0

Example 2:

In this example both generators are used. Axis 2 will be connected to generator 1 and axis 2 will be connected to generator 2.

At first the waveforms are defined. The first waveform is a sine wave and the second is a cosine wave. The main difference between the definition sequences of the sine and the cosine wave is the parameter of the PA command:

0**PT**0

1**PT**600

1**CP**600

1**PC**300

1**PA**150

1**PS**0

1**FO**0.0

1**GS**10.0

2**PT**600

2**CP**600

2**PC**300

2**PA**0

2**PS**0

2**FO**0.0

2**GS**10.0

0**PT**0

1**PT**600

2**PT**600

Select the waveform 1 for generator 1 and waveform 2 for generator 2:

1**SF**1

2**SF**2

Connect the generators to the desired axes. In this example generator 1 is connected to axis 2 and generator 2 is connected to axis 3:

2**CF**1

3**CF**2

Connect axis 2 to DDL table 1:

2**LT**1

Enable DDL usage for axis 2:

2**LS**1

Connect axis 3 to DDL table 2:

3**LT**2

Enable DDL usage for axis 3:

3**LS**1

Clear all tables:

0**CL**0

Set axis 2 and 3 to servo-on mode (if axis 2 or 3 are in servo-off mode):

2**SL**1

3**SL**1

The limitation of the generator cycle number is done in the same way as it was described in example 1:

RN50

Perform a DDL initialization run: start both wave generators with reinitialize DDL on for axis 2 and 3 (see MC command on p. 124):

0**MC**272 *switch* bits 4 (decimal 16) and bit 8 (decimal 256) ON, i.e. first bit for 2nd and 3rd axis

The wave generator will stop after the number of cycles which were set by the RN command (see p. 133).

The next time when the wave generator is started without DDL reinitialization active, but with the DDL data saved in DDL tables 1 and 2, DDL will be in effect and the position error will be reduced.

Before the wave generators are started again, the number of cycles can be set to the value desired for the application. If the number of wave generator cycles is to be unlimited, use the following command:

RN0

Now start both generators without DDL reinitialization active:

0**MC**0

11.4 Parameters

Many parameters are available to fine-tune the DDL (digital dynamic linearization) operation. The following parameters are preset at the factory and usually do not need to be adjusted:

Time-Delay-Change-Rule = -3 to –5
DDL-Zero-Gain-Number = 5
Autocal-Time-Delay-Factor = 1
Autocal-Min/Max-Time-Delay-Factor = 0.45
DDL-Gain-Constant = 1.0
DDL-Gain-Change= 0
DDL-Gain-Curve = 0
Final-DDL-Gain = 0
DDL Repeat Number = 35

These parameters are dependent on system performance, such as the servo update rate (5 kHz), servo-parameter optimization

(integration time constant = $1/2\pi f_{res}$) and mechanical load (symmetrical load).

Some of these parameters can also be tested and optimized by the user, making small step-by-step changes.

The *Autocal-Time-Delay-Factor* and *Autocal-Min/Max-Time-Delay- Factor* parameters are used for automatic calculations (see the LP command on p. 121) and can be adjusted by service personnel. The following parameters are readjusted automatically by LP:

Time-Delay-Max, Time-Delay-Min and *Final-Time-Delay.*

The auto calculation is carried out only when an **LP** command is sent to the controller. The power-up sequence only reads them from non-volatile parameter memory. These parameters are dependent on the servo parameters of the axis. If the servo parameters are changed, the **LP** command should be run to recalculate these parameters.

At any one time a maximum of 4 DDL tables can be selected. To use a total of 32768 points the command 0**PT***offset* must be used. That is, select an offset, use 4 waveforms and 4 DDL tables. The entries of the DDL tables must also be defined with the **LT** command (p. 122).

The number of initialization cycles is determined by the *DDL Repeat Number*

11.5 Advanced Tuning

When using the parameter settings described and/or after running the automatic DDL data parameter calculation (LP command, p. 121), good results can be obtained in most cases, presuming that the system has been previous adjusted to assure that no oscillation occurs (see Section “Servo-Controller Dynamic Calibration”, p. 74).

Behavior of the system with DDL depends on the dynamic characteristics of the system as a whole. These depend not only on the size of the mass being moved, but also on its shape. Because of this fact, it may in some cases be necessary to apply corrections to the values resulting from the LP command or the values suggested in order to obtain optimal operation.

The following parameters can be adjusted for optimization:

Time-Delay-Min at address 147

Time-Delay-Max at address 146

Time Delay Change Rule at address 148

DDL Repeat Number at address 141

11.5.1 Periodic Errors

If there is a residual periodic position error which could not be reduced, then the following parameters must be adjusted:

Time-Delay-Min at address 147

Time-Delay-Max at address 146

These parameters can be modified by using the *Device Parameter Configuration* window of *NanoCapture*™. You will find the parameters in the “DDL” parameter groups. In the parameter group labeled “DDL 1” in the current software version you will find the parameters for axis 1, in the “DDL 2” parameter group you will find the parameters for axis 2, and so on for up to the maximum number of axes.

Perform the following steps to adjust these parameters:

1. Measure the period of the position error for the axis.
2. Compare the current time-delay parameters with the period. The period value should be at the midpoint between Time-Delay-Min and Time-Delay-

Max. The parameter value which is closest to the period value should be modified.

Example:

The measured period is 11 ms. The value of Time-Delay-Min is 10 ms and the value of Time-Delay-Max is 16 ms. In this case the Time-Delay-Min should be modified so that difference the parameter and the period value increases. First use 8 ms instead of 10 ms. If a further reduction of the position error is necessary, the Time-Delay-Min value could be set to 6 ms.

Do not use the LP command after you adjusted the Time-Delay-Min and Time-Delay-Max parameters to obtain the minimum error. Otherwise, the manually changed values will be overwritten.

If further improvement is needed, the Time Delay Change Rule (address 148) parameter can also be changed. The default value is –3. Only negative values will be accepted for this parameter. Normally, improvement can be achieved by using more negative values.

11.5.2 DDL Auto Examine

The DDL Auto Examine feature is available with 6-axis versions and 3- and 4-axis versions with firmware rev.5.035 or higher but not 6.00 to 6.034.

The residual tracking error is also dependent on the behavior of PZT material, which can vary with time. To preserve optimum Digital Dynamic Linearization (DDL) performance, a DDL initialization phase must be repeated from time to time (always under the same conditions as operation in the application and with a scan repeat rate as close to the same as possible. With Auto Examine (see the **AE** and **ES** commands) the residual tracking error will be measured and if it is larger than the allowed amount, a new DDL initialization phase will be initiated at once.

To examine the DDL result, use the **ES** to define the start and end points between which the residual error will be calculated.

Take, for example, a waveform with a total of 500 points, and a portion from point 150 to 400 where the moving speed is constant and the residual error between target and actual position should be very small. Generator 1 is connected to this wave and to axis 1. The commands to use would then be:

1**ES**150	define the start point 150 for axis 1
101**ES**400	define the end point 400 for axis 1
ES1	enable the auto examine

Thereafter, drive generator 1 with the **SC** or **MC** command several times (at least three times, under the same conditions as in the application—same waveform, same delay, etc.). The generator must be started in a mode that records the position error. For example:

1**MC**12	Record position error (*switch* = 12).

The residual error of last run will be saved internally and is available for examination with the **AE** command:

1**AE**1	read the average residual error value.
1**AE**2	read the residual error standard deviation.

The residual error is the difference between target value and actual value for the portion between points 150 to 400. If the error is too large,

1**AE**16	will make the compensation based on the current error data.

12 Configuration Examples

The following sections give some typical configuration examples and the corresponding memory value settings.

12.1 No Servo-Control, Sensor Evaluation Only

This configuration allows reading and processing from one to four sensor signals. Any sensor signal can be used with its linearization and coordinate transformation and optional digital filtering. Sensor-channel-to-axis coordinate transformation is handled based on the product of a unitary matrix (no cross-coupling) and a diagonal matrix (allowing any desired conversions with linear coefficients)

Typical non-volatile memory entries (channel/axis specific) are:

Memory Address	**Value**	**Function**
2	0	Servo-Control enable OFF
3	0	Servo-Control Autostart OFF
4	0	Auto Zero Autostart OFF

Use the following command sequence to write the parameter values into the memory. This sequence shown is for axis 2, it must be repeated for each axis:

2DP-1	prepare to change non-volatile memory parameters for axis 2
2DW0	disable servo-control for axis 2
3DW0	do not have axis 2 servo-control turned ON automatically at power-up
4DW0	do not run AutoZero automatically on axis 2 at power-up

12.2 Servo-Control with 1 to 6 Independent Channels

To enable the servo-controlled mode, first the sensors, then the servo-controllers have to be activated. Auto-zero mode should be enabled for the linear axes. For independent channels (max. 6 independent PZT channels!), both the sensor transformation and the PZT coordinate transformations must be diagonal matrices (all zeros except on main diagonal).

Memory Address	**Value**	**Function**
2	1	Servo-Control enabled
3	1	Servo-Control Autostart ON
4	1	Auto Zero Autostart ON

In the factory configuration the *Servo-Control Autostart* and *AutoZero Autostart* options are turned off, while the *Sensor enable* and *Servo-Controller enable* options are turned on.

Use the following command sequence to write the parameter values into the memory:

*a*DP-1	where *a* is the axis: repeat the entire sequence for each axis.

2DW1

3DW1

4DW1

12.3 Servo-Control with 1 to 8 Coupled Channels

A coupled sensor or PZT channel is a channel whose direction has components in more than one logical axis. The same settings as for independent channels can be used. Because of the coupling, the coordinate transformation matrices must contain the necessary coupling factors.

13 Servo-Controller Dynamic Calibration

If the controller and the attached actuators are ordered together and/or if PI has sufficient knowledge of your application, then the system will be delivered preconfigured and precalibrated. Because of drift and aging considerations, it will be necessary to carry out some calibration steps from time to time.

If the mass which is mounted on the piezo stage is changed, a modification of the controller parameters will be necessary. In this case, for each servo-controlled axis the following parameters may need to be modified

- ➢ Notch filter frequencies
- ➢ Servo-loop time constant
- ➢ Servo-loop P-term (loop gain)
- ➢ Slew rate

It is most convenient to use the *NanoCapture*™ software to change these parameters over the RS-232 or IEEE 488 interface.

13.1 Overview

13.1.1 Suppressing Mechanical Resonance with Notch Filters

Mechanical resonances of the system exaggerate the response to certain frequencies in the drive signal. Six-axis controllers have an anti-vibration (notch) filter on the *input* to the servo-loop designed to keep the user from exciting vibration of the entire stage in its mounting. This is especially important if the piezo stage is mounted on another stage. That filter is set with the “D... Set/Read Antivibration Filter ” command, p. 97, or the “Target Manipulation” parameter group in *NanoCapture*™ and will not be discussed further here.

The controller also has two notch filters for each axis on the *output* of the servo-control loop to compensate for resonances in the platform and attached application by reducing the corresponding frequency components in the control signal. To determine the resonant frequencies and set the notch filters properly, the system response to an open-loop impulse is observed.

13.1.2 Servo-Control Loop Settings

The servo-loop settings can be used to optimize settling time and control overshoot. The parameters which can be set for each axis include the loop gain (P-factor) and the time constant (I-factor). To set these parameters, the system response to a closed-loop step move is observed. The maximum slew rate (velocity) can also be set: it depends on the size of the actuators and the robustness of the attached mechanics.

13.1.2.1 Loop Gain (P-term)

This parameter also affects the speed of the stage. It is used to optimize the dynamic precision of the stage. Normally the proper loop gain setting is found by observing the response of the stage to an abrupt change in the drive voltage (step response). *NanoCapture*™ can display the step response of the stage in graph form.

13.1.2.2 Time Constant

The value of the time constant depends on the notch frequency. It is therefore adjusted automatically whenever the notch frequency is modified. The value of the time constant is calculated with the following formula:

$$TimeConstont = \frac{1}{4 \cdot \pi \cdot NotchFrequency}$$

If desired, the time constant can then be modified in the *Dynamic Tuner* window explicitly.

13.1.2.3 Slew Rate

This parameter limits the maximum speed at which the axis will be commanded to move from the current position to a new target.

In order to adjust the slew rate setting with *NanoCapture*™ use the *Dynamic Tuner* window.

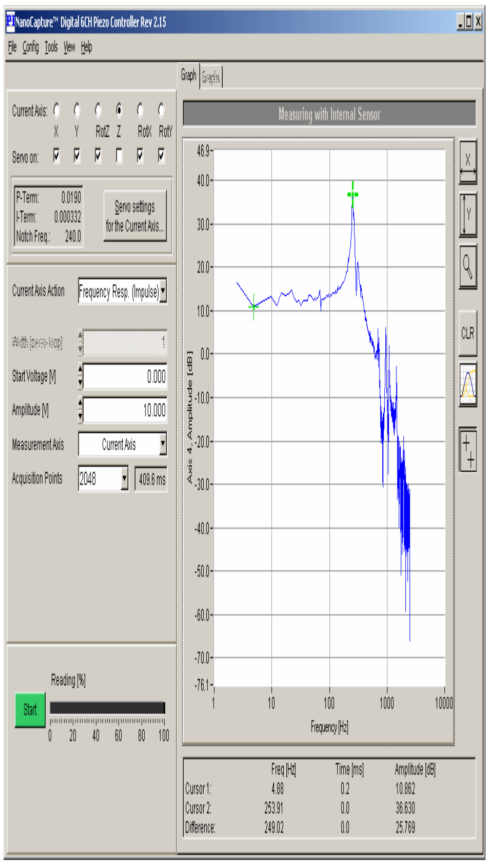
Slew rate limitation is only effective when servo-control is enabled and servo mode turned ON for the axis in question.

13.2 Adjustment Procedures

13.2.1 Setting Notch Filters

To determine the resonant frequencies and set the notch filters accordingly, use the *NanoCapture*™ software and proceed as follows:

1. Make sure the mechanics are mounted and connected to the controller in exactly the same way as in the application. The load on the stage is especially important.
2. Start *NanoCapture*™ on a host PC connected to the E-710 (see the *NanoCapture*™ software manual on the included CD for details).
3. In the *Current Axis Action* field, select *Frequency Response* (Fig. 44). *Frequency response* involves exciting an axis with an impulse move (IP command, p. 119) and recording/displaying the resulting motion.
4. Set *Measurement Axis* to *Current Axis* (i.e. measure and display the axis which receives the impulse)
5. The measurement must be done in open-loop mode. Uncheck the *Servo ON* boxes for all axes to be measured (Fig. 45). Axis numbers start with 1 on the left.
6. Select the axis to receive the impulse using the *Current Axis* radio



NanoCapture™ Digital 6CH Piezo Controller Rev 2.15

Current Axis: X ( ) Y ( ) RotZ ( ) Z (●) RotX ( ) RotY ( )

Servo on: X ✓ Y ✓ RotZ ✓ Z ☐ RotX ✓ RotY ✓

P-Term: 0.0190
I-Term: 0.000332
Notch Freq.: 240.0

Current Axis Action: Frequency Resp. (Impulse)
Start Voltage [V]: 0.000
Amplitude [V]: 10.000
Measurement Axis: Current Axis
Acquisition Points: 2048 409.6 ms

Measuring with Internal Sensor

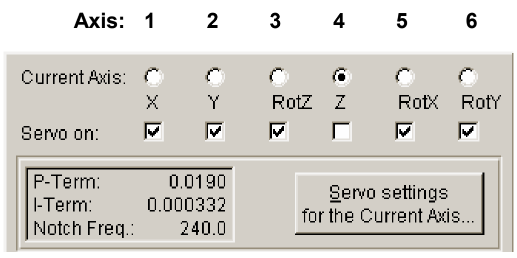
	Freq [Hz]	Time [ms]	Amplitude [dB]
Cursor 1:	4.88	0.2	10.862
Cursor 2:	253.91	0.0	36.630
Difference:	249.02	0.0	25.769

Fig. 44: Select Frequency Response (Impulse)

buttons (in Fig. 45, Axis 1).

7. Set *Start Offset* to 0, *Amplitude* to about 15% of the travel

8. Start the measurement by clicking on the *Start* button. An impulse move is commanded, position data recorded, analyzed and displayed as a Bode frequency response diagram (Fig. 46).

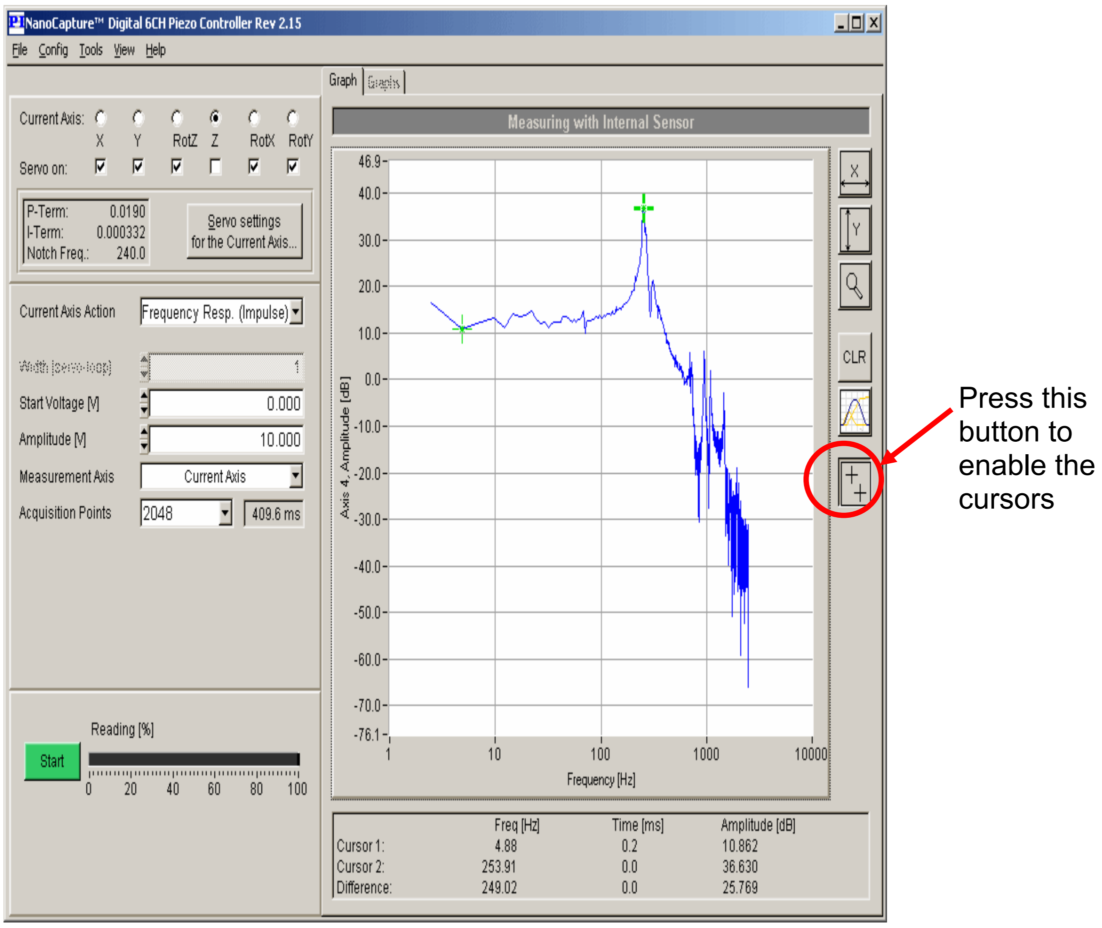


Current Axis: X Y RotZ [✓ Z] RotX RotY
Servo on: ✓ ✓ ✓ ☐ ✓ ✓

P-Term:	0.0190
I-Term:	0.000332
Notch Freq.:	240.0

Fig. 45: Servo ON/OFF and Current Axis selection

9. On the Bode plot, identify the resonance peak. To do that, you can place a cursor on the peak and read out the cursor value which is displayed below the graph (see Fig. 46).
   It is possible to adapt the notch filter setting automatically to the measured resonance peak (see step 11 below)—if you want to do that, the two cursors must be placed as follows:
   With the mouse, move either one of the cursors to the resonance peak. Make sure the other cursor points to a lower frequency, i.e. that it is to the left of the cursor pointing to the resonance peak.


NanoCapture™ Digital 6CH Piezo Controller Rev 2.15
File Config Tools View Help
Graph Graphs
Servo settings
for the Current Axis...
Measuring with Internal Sensor
Axis 4, Amplitude [dB]
Frequency [Hz]
Reading [%]
Start
CLR
Press this button to enable the cursors

Current Axis: X Y RotZ [✓ Z] RotX RotY
Servo on: ✓ ✓ ✓ ☐ ✓ ✓

P-Term:	0.0190
I-Term:	0.000332
Notch Freq.:	240.0

Current Axis Action	Frequency Resp. (Impulse)
Width (servo-loop)	1
Start Voltage [V]	0.000
Amplitude [V]	10.000
Measurement Axis	Current Axis
Acquisition Points	2048 / 409.6 ms

	Freq [Hz]	Time [ms]	Amplitude [dB]
Cursor 1:	4.88	0.2	10.862
Cursor 2:	253.91	0.0	36.630
Difference:	249.02	0.0	25.769

Fig. 46: Bode frequency response diagram; resonant frequency marked with rightmost cursor

10. Now open the *Dynamic Tuner* window by pressing F3 or using the *View* → *Dynamic Tuner* menu sequence. If necessary click *More* to see all the parameters. This window has sliders to adjust the dynamic parameters and buttons to save the settings (Fig. 47). If the controller only has one notch filter, the *2nd Notch Filter* pane will be grayed out.

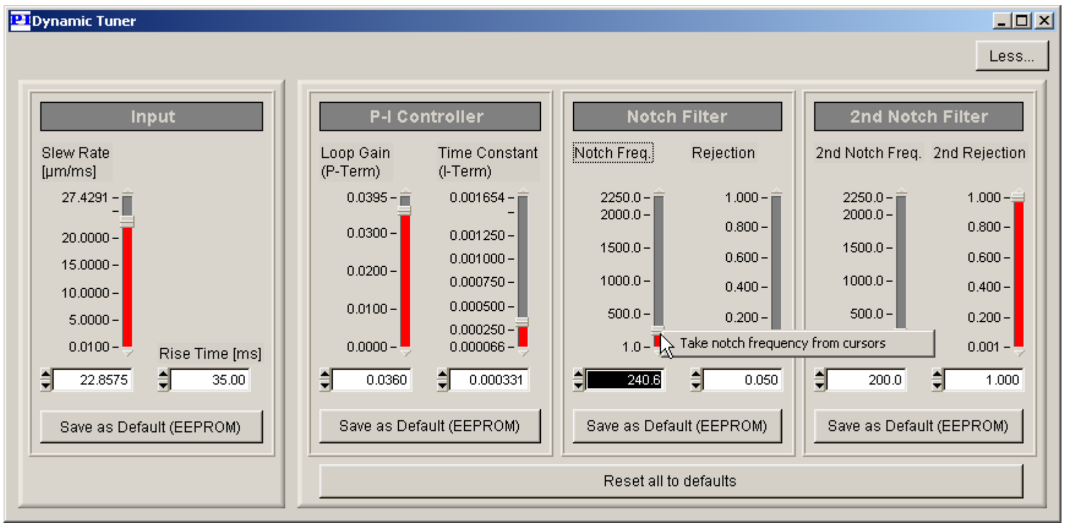
11. Move the *Notch Freq.* slider to adjust the notch frequency (Fig. 47) or right-click on the slider and click on the *Take values from cursors* item on the context menu that appears. The notch filter frequency will than be taken from the cursor which has the higher frequency value. If desired, repeat the procedure for the second notch filter, if available: mark the next higher resonance with the cursor (see step 9) and use the *2nd Notch Filter* slider.

NOTES

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

When the *Notch Freq.* value is set, the *Time Constant* servo parameter in the *P-I Controller* pane is adjusted automatically in accordance.

12. If you have finished the settings, you can press the *Save as Default* buttons to save the settings in the controller as power-up defaults (the other settings in this window will be discussed later).
13. Repeat steps 6 through 12 for each axis which needs to be retuned.



Slew Rate [µm/ms]: 22.8575
Rise Time [ms]: 35.00
Loop Gain (P-Term): 0.0360
Time Constant (I-Term): 0.000331
Notch Freq.: 240.6
Rejection: 0.050
2nd Notch Freq.: 200.0
2nd Rejection: 1.000

Fig. 47: Dynamic Tuner window, extended

13.3 Setting Servo Parameters

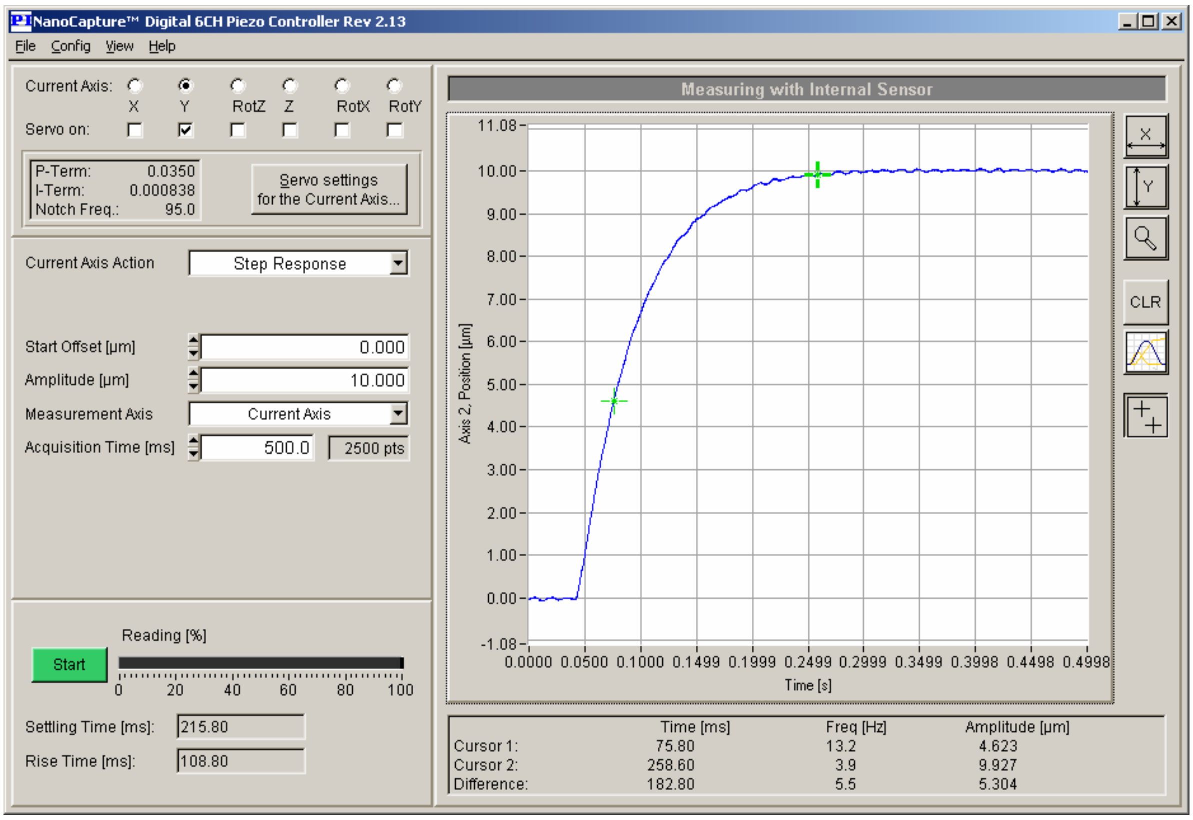
The servo parameters described above are optimized empirically by observing the effect of different values on a closed-loop Step Response.

1. Select *Step Response* in the *Current Axis Action* field
2. Set *Start Offset* to 0, *Amplitude* to about 15% of the travel range, and *Measurement Axis* to *Current Axis*, (so that moved axis and measured axis are the same).
3. Set servo-control ON for all affected axes (see Fig. 45).
4. Use the *Current Axis* radio buttons to select the axis to set.
5. Open the Dynamic Tuner window by pressing $\boxed{\text{F3}}$ or using the *View* → *Dynamic Tuner* menu sequence. If necessary click *More* to see all the parameters. (Fig. 47).
6. Change the P-term, slew rate and/or I-term slightly as desired.
7. Click *Start* in the main window to perform a step.

8. Observe the results, comparing with the examples shown (Fig. 48 to 50).

   - If the *loop gain* value is very low (as in Fig. 48) then the rise rate of the stage response curve will also very low.
   - If the *loop gain* value is increased (Fig. 49) then the rise rate also rises, but the overshoot will also increase. The figure shows a step response with a small overshoot.
   - If *loop gain* is further increased (Fig. 50) then the rise rate will not rise significantly, but the overshoot will increase.

   The stability of the servo-loop also depends on the *loop gain*. If the value is too high, then the controller will oscillate.


NanoCapture™ Digital 6CH Piezo Controller Rev 2.13
File Config View Help
Current Axis: X Y RotZ Z RotX RotY
Servo on:
Servo settings for the Current Axis...
Start
Reading [%]
0 20 40 60 80 100
Measuring with Internal Sensor
Axis 2, Position [µm]
Time [s]
CLR

P-Term: 0.0350
I-Term: 0.000838
Notch Freq.: 95.0

Current Axis Action: Step Response
Start Offset [µm]: 0.000
Amplitude [µm]: 10.000
Measurement Axis: Current Axis
Acquisition Time [ms]: 500.0 2500 pts

Settling Time [ms]: 215.80
Rise Time [ms]: 108.80

	Time [ms]	Freq [Hz]	Amplitude [µm]
Cursor 1:	75.80	13.2	4.623
Cursor 2:	258.60	3.9	9.927
Difference:	182.80	5.5	5.304

Fig. 48: Closed-loop step response; low loop gain, low speed, no overshoot

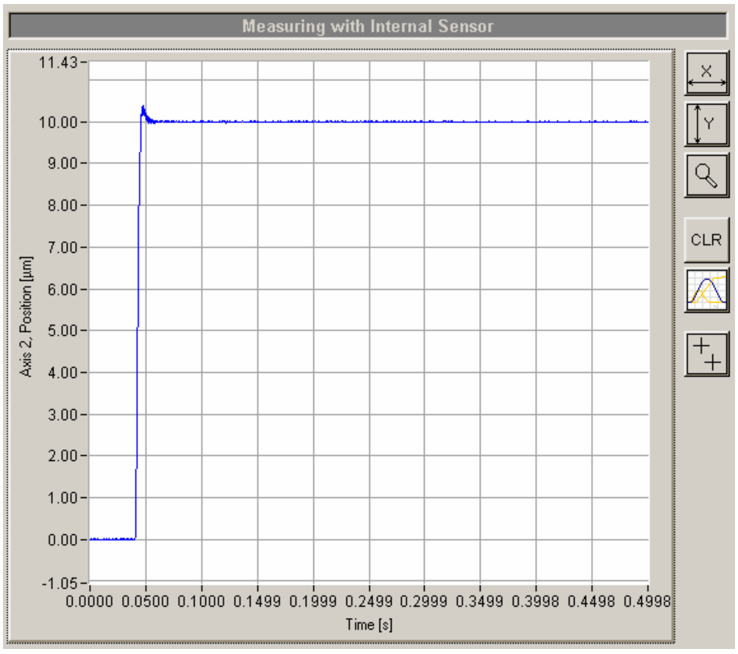


Fig. 49: Step response; high speed, low overshoot

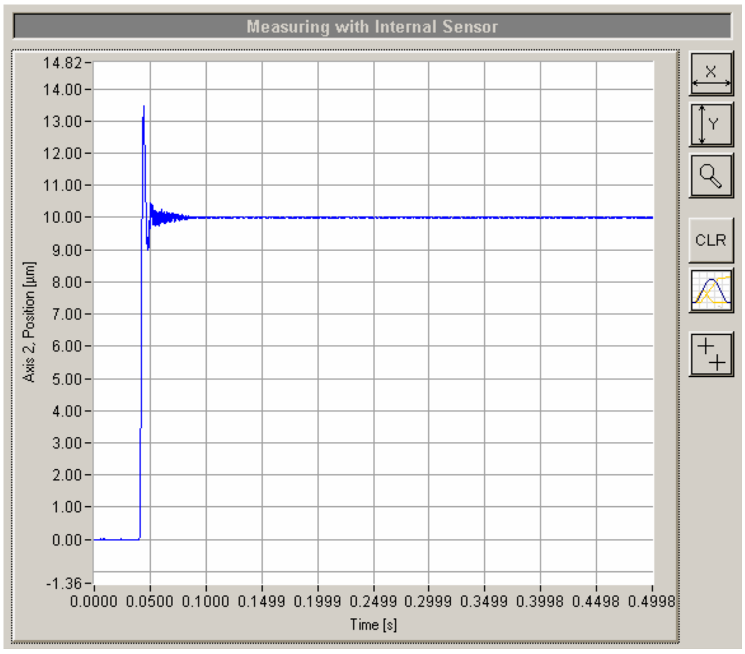


Fig. 50: Step response; higher speed, high overshoot

9. Return to the *Dynamic Tuner* window, refine the settings and repeat until satisfied.
10. Press *Save as Default* for each of the changed parameter groups.
11. Repeat steps 4 onwards for the remaining axes to tune.

13.4 AutoZero

Both the range of axis position values (from the sensors, after coordinate transformation) and the range of axis motion control signals are limited. If there is an offset between the input and output ranges caused by a mechanical drift of the piezo stage, then the usable range will be reduced. Such an offset can be compensated by the AutoZero function, as explained with the following example:

Take a one-dimensional piezo stage and its sensor with a position range from 0 µm to 200 µm. The controller has an output voltage range from –10 V to +110 V. The piezo stage has a sensitivity of about 2 µm/V. That means that the piezo stage would be displaced by 200 µm if the output voltage is 100 V. In this case the controller has a voltage reserve of about 10 V in each direction. Ideally the position of the piezo stage should be 0 µm if the servo loop is switched off and the output voltage of the controller is set to 0 V. In fact there will be a position offset because of some nonideal properties of the piezo. If the position offset is not larger than 10% of the position range (20 µm) this offset can be compensated by the controller when the servo-loop is activated, using the 10% voltage reserve. If the position offset is greater than 20 µm, it will no longer be possible to move the stage to the zero position. This situation can be corrected by using the AutoZero procedure

The position offset of the piezo stage is compensated by adding an offset constant value to the sensor value. The mechanical zero position of the piezo stage will change when AutoZero is executed. But after AutoZero the full travel range can be used.

13.4.1 When Should AutoZero Be Executed?

- If the stage and controller are not well known, executing AutoZero is the best way to begin.
- When the stage is first integrated into the application environment, AutoZero must be run.
- If only relative movement is important, executing AutoZero after every power up is recommended.

If absolute moves are needed, AutoZero should **not** be executed during normal operation because AutoZero changes the mechanical zero position of the piezo stage. AutoZero should be executed only when the stage is initially integrated into the application environment. In that case AutoZero should be started by the AZ command with automatic saving of the offset constant selected. Read the description of the AZ command for details (see p. 91).

AutoZero is also recommended if the system is subjected to temperature changes.

Note that AutoZero is not effective on non-linear axes.

13.4.2 How Can AutoZero Be Started?

CAUTION—AutoZero and Analog Input

Optionally, the Analog Input can be used to generate target positions / offsets for an axis (see p. 40). The Sensor Matrix for the axis is then configured accordingly: the Sensor channel representing the Analog Input has a negative contribution to the axis position.

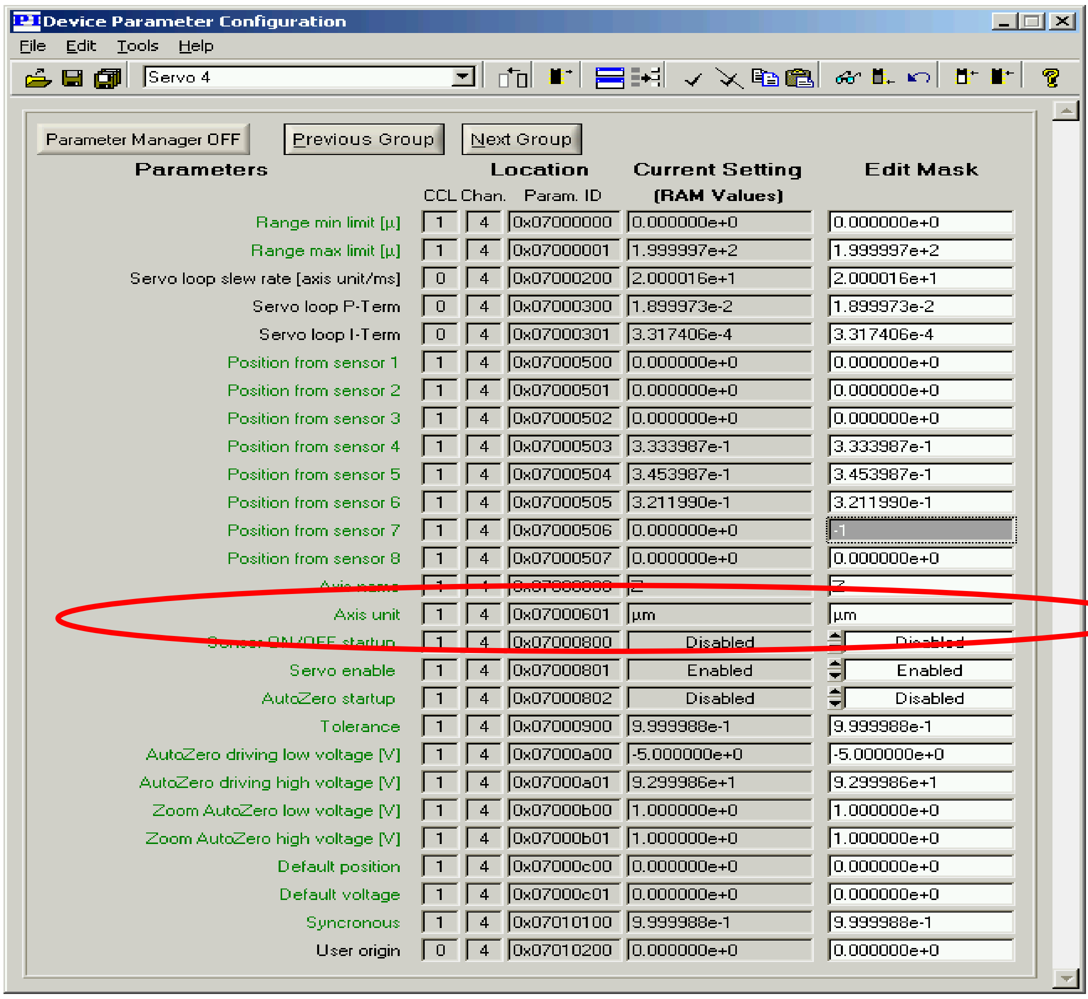
Axes with negative sensor channel contribution to the axis position are automatically considered as non-linear (rotational) axes on which AutoZero is not effective. To perform AutoZero for axes which are connected to the Analog Input for target generation, you have to change the sensor-to-axis matrix temporarily. Sensor-to-axis transformation matrices are accessible in *NanoCapture™* via the *Position from Sensor…* parameters of the *Servo* 1 through *Servo n* parameter groups. See below for details.

- To start AutoZero manually, execute the AZ command. You can specify a single axis in the command, or you can specify that autozero is to be run on all (linear) axes. With firmware 5.011 and higher (except 6.010), the AZ command allows saving the calculated offset value(s) automatically. Read the command description (p. 91) for details.

➢ AutoZero is started automatically after power up if the *autozero enable* configuration option is turned ON for one or more axes. The default value of this option is OFF. To turn this parameter ON you must write the appropriate configuration option values to non-volatile memory (see D... Set/Read AutoZero Enable Status, p. 98) with the following compound command:
*a*DP-1, 4DW1
where *a* is the axis designation. Repeat for each axis to be autozeroed upon power-up. If *a* is 0, all (linear) axes are autozeroed.

➢ AutoZero when the Analog Input is used for the axis target generation:

In the example shown in Fig. 51, the Analog IN determines the Axis 4 target position. Sensor 7 represents the Analog IN and therefore has a negative contribution to the position of axis 4.



Parameters	Location CCL	Location Chan.	Location Param. ID	Current Setting (RAM Values)	Edit Mask
Range min limit [µ]	1	4	0x07000000	0.000000e+0	0.000000e+0
Range max limit [µ]	1	4	0x07000001	1.999997e+2	1.999997e+2
Servo loop slew rate [axis unit/ms]	0	4	0x07000200	2.000016e+1	2.000016e+1
Servo loop P-Term	0	4	0x07000300	1.899973e-2	1.899973e-2
Servo loop I-Term	0	4	0x07000301	3.317406e-4	3.317406e-4
Position from sensor 1	1	4	0x07000500	0.000000e+0	0.000000e+0
Position from sensor 2	1	4	0x07000501	0.000000e+0	0.000000e+0
Position from sensor 3	1	4	0x07000502	0.000000e+0	0.000000e+0
Position from sensor 4	1	4	0x07000503	3.333987e-1	3.333987e-1
Position from sensor 5	1	4	0x07000504	3.453987e-1	3.453987e-1
Position from sensor 6	1	4	0x07000505	3.211990e-1	3.211990e-1
Position from sensor 7	1	4	0x07000506	0.000000e+0	-1
Position from sensor 8	1	4	0x07000507	0.000000e+0	0.000000e+0
Axis name	1	4	0x07000600	Z	Z
Axis unit	1	4	0x07000601	µm	µm
Sensor ON/OFF startup	1	4	0x07000800	Disabled	Disabled
Servo enable	1	4	0x07000801	Enabled	Enabled
AutoZero startup	1	4	0x07000802	Disabled	Disabled
Tolerance	1	4	0x07000900	9.999988e-1	9.999988e-1
AutoZero driving low voltage [V]	1	4	0x07000a00	-5.000000e+0	-5.000000e+0
AutoZero driving high voltage [V]	1	4	0x07000a01	9.299986e+1	9.299986e+1
Zoom AutoZero low voltage [V]	1	4	0x07000b00	1.000000e+0	1.000000e+0
Zoom AutoZero high voltage [V]	1	4	0x07000b01	1.000000e+0	1.000000e+0
Default position	1	4	0x07000c00	0.000000e+0	0.000000e+0
Default voltage	1	4	0x07000c01	0.000000e+0	0.000000e+0
Syncronous	1	4	0x07010100	9.999988e-1	9.999988e-1
User origin	0	4	0x07010200	0.000000e+0	0.000000e+0

Fig. 51: Servo Parameter Configuration for axis 4; Analog IN (represented by sensor channel 7) used for target generation

Before you can perform AutoZero for axis 4, the negative sensor contribution in the sensor-to-axis transformation matrix (see the marking in Fig. 51) must be temporarily replaced by 0 (see the *NanoCapture™* manual for how to change parameter values). Then perform the AutoZero procedure and afterwards restore the sensor-to-axis matrix with the original negative sensor contribution.

14 Command Format

The E-710 firmware understands native ASCII commands received over the RS-232 or IEEE 488 interface (whichever is activated) and a limited number of command-like operations signaled over the PIO interface.

14.1 Native ASCII Commands

For sending the native commands to the E-710, use a terminal program like *WinTerm32* or the *NanoCapture™* graphic user interface software which provides a terminal for the native commands.

14.1.1 Single Commands

A single command has the general format:

*Axis_identifier***Command***Parameter*$\boxed{\text{LF}}$

Axis_identifier:	Any number from 1 to 8 indicating the axis or channel to be addressed.
Command:	Command token (mnemonic) consisting of 2 characters.
parameter:	Parameter value in integer or floating point representation.
$\boxed{\text{LF}}$*:*	Line feed character (ASCII 10), terminates the command string.

Examples:

1ma50$\boxed{\text{LF}}$	Move axis #1 to absolute position 50.00 micrometers, parameter value in integer representation
1ma50.01$\boxed{\text{LF}}$	Move axis #1 to absolute position 50.01 micrometers, parameter value in floating point representation
1ma1.0031e2$\boxed{\text{LF}}$	Move axis #1 to absolute position 100.31 micrometers, parameter value in floating point representation
1tp$\boxed{\text{LF}}$	Report position of axis #1
he$\boxed{\text{LF}}$	Help command listing all commands available

Note that no spaces are allowed between the elements of a single command. A carriage return (ASCII 13) before the line feed will be ignored. The line feed characters will not be explicitly shown in the examples in the remainder of this manual.

14.1.2 Compound Commands

Compound commands are composed of single commands without terminators, separated by commas and sent in one command line. The termination character, $\boxed{\text{LF}}$, must be added as the last character in the string. Again, no spaces are allowed.

Example:

1ma75,wa100,1tp,1gh,wa300,rp5$\boxed{\text{LF}}$

This compound command first moves axis #1 to position 75 µm, then waits for 100 ms and reports the position before moving back to home. Then the PZT rests for 300 ms and starts all over again, repeating 5 times.

14.1.3 Command String Length

80 characters in one string;

last command ignored if the string is too long;

maximum 40 commands in one string

14.1.4 Report Formats

Query commands issue answers, called reports. Depending on the query command, the device answers with:

- Text
- Integer value
- Fixed-point number, format: xxx.xxxx, value in µm or µrad
- Floating point number
- Combination of text and numbers

Each report line is terminated by a $\boxed{\text{LF}}$ character. Every line of a report *except the last line* also has a space character, $\boxed{\text{SP}}$, ASCII 32, before the line feed. This makes it possible to recognize the end of a report.

14.1.5 Command Rate and Programming Hints

To optimize the data transfer rate to the controller, some aspects regarding the command set and the internal command processing have to be taken into account.

The general structure of the examples can be adapted to any IEEE 488 interface board and even to RS-232 interfaces, but in these cases, with significantly lower transfer rates.

There are many cases where sensor positions have to be read continuously to monitor the servo-control behavior of the system. Virtually continuous sensor data is available over the PIO interface (see p. 32). Otherwise, the "TP" command can be used in a program loop to read the sensor position repeatedly.

Example 1:

```
for(i=0;i<100;i++) {
     SendToE710("1TP");
     ReadE710(string);
     EvaluateAnswer(string);
}
```

where the functions referenced are defined by the user in accordance with the system and interfaces.

Alternating invocations of write and read procedures reduce the efficiency and limit the cycle rate. With the code in example 2, significantly higher data rates can be obtained:

Example 2:

```
SendToE710("1TP,WA10,RP100");
for(i=0;i<100;i++) {
     ReadE710(string);
     EvaluateAnswer(string);
}
```

Thus only one command is used to cause the controller to report the sensor position 100 times. The program loop reads the reports as fast as they come in. Any required synchronization must be done by the interface.

To use this technique successfully, some additional points must be kept in mind:

- The maximum number of cycles is limited to 10E6 (one million).
- The compound command must include a WAn command. Only then will the readings be taken at equal time intervals. The shortest allowable wait time is 1 ms (WA1).
- Up to 100 reports can be stored in the internal output buffer (with floating point values, the number depends on the number of characters). The buffer is organized FIFO. If the buffer overflows, which might happen if the receiving rate is too low, internal command execution is stopped, causing gaps in the sampling sequence.
- In order to get data at known points in time, the internal WAIT time should be set as long as required for a complete loop with read and detect procedures. All calculation and evaluation routines should be outside the sampling loop.

  Depending on the processor activity level, number of channels/axes, interfaces and programming techniques, the loop execution time will vary; the most appropriate value has to be found by experience.

Use the following example as a point of reference:

Controller E-710, 4 channels
Interface IEEE 488, NI-GPIB board
PC Pentium 166, Windows NT 4.0
Programming in LabWindows CVI
WAIT time WA2 (2 ms) gives a data rate of 500/s

14.2 PIO Commands

PIO commands and responses take the form of voltage levels on the various digital data, address and control lines of the PIO interface. The data transferred is on a per-axis basis and includes transmission of new target positions to the servo-controller, readout of current position values, on-target signals and system flags. Data can be transferred over the PIO many times faster than over the ASCII interface.

Refer to the \Samples\LabView or \Samples\C directories for examples on how to send the required signals over the interface. The current version of *NanoCapture*™ and the E-710 GCS DLL do not explicitly support the PIO.

14.2.1 Data Write (Set Target Position)

In the Set Target Position operation, a target value is input over the PIO and written to the servo-controller target register for the logical axis specified.

This operation is handled with a double-buffered latch structure and can be performed as follows:

1. Set target-position data word
2. Set address (= axis – 1)
3. On STRB and RW valid, data are stored into first-rank latch
4. Repeat operations 1-3 for all desired axes
5. Set LD to transfer data into second-level latch, which activates any new target on all axes simultaneously

The axes will only be moved to the new target positions when their servos are in closed loop mode. If one axis is not set into closed loop mode the PIO write command for this axis will be ignored.

14.2.1.1 Operating Notes

Set the targets on all axes once upon power-up. The first write operation after power-up should be performed as follows:

1. Set target position data word
2. Set address lines (= axis – 1)
3. On STRB and RW valid, data is stored into first-rank latch
4. Repeat operations 1-3 **for all axes**
5. Set LD to transfer data into second-level latch, which activates the new target

For further operation, it is sufficient to set only the axes which are to change position. All other axis targets will remain unchanged if not accessed.

It is necessary to wait a minimum amount of time—called TLDST and listed in table 1, p. 166—between two LD signals. Otherwise data may be lost.

The "address" value is equal to the logical (servo-controlled) axis number - 1.

If servo-control for the specified axis is disabled or turned OFF, the new target value will still be written to the register, but will have no immediate effect.

14.2.2 Data Read (Read Sensor or Flags)

The sensor-read operation can be performed at any time. It refers to the position values after sensor coordinate transformation, and is initiated by the following sequence:

1. Set RW and LD
2. Set address (address in 0-3 for axes 1-4; address = 4 for reading system flags)
3. Set STRB
4. Read data

If multiple axis positions are to be read, repeat steps 2-4 as desired. The completion of settling can be determined with such operations.

If the address is set to 4, system flags are read instead of a sensor value:

Data line	**Flag**
D0	On Target, Axis 1
D1	On Target, Axis 2
D2	On Target, Axis 3
D3	On Target, Axis 4
D4	Overflow, Axis 1
D5	Overflow, Axis 2
D6	Overflow, Axis 3
D7	Overflow, Axis 4

14.2.3 On-Target Signals

On-target signals are PIO outputs that can be used at any time. They are simple voltage-level hardware signals without any timing or readout constraints. Internal data update occurs once every 200 µs. They reference the servo-control axes.

14.2.4 PIO Data Format and Scaling

16-bit, integer, unsigned

Range: 0–65535

Computing data word to send as target position:

PIO_{data} = 65535 x (target– $PIO_{InputLowLimit}$) / ($PIO_{InputHighLimit}$ – $PIO_{InputLowLimit}$)

Computing sensor position from received data word:

current_position = (PIO_{data} / 65535) x ($PIO_{InputHighLimit}$ – $PIO_{InputLowLimit}$) + $PIO_{InputLowLimit}$

Values for *PIO_Input_Low_Limit* and *PIO_Input_High_Limit* are stored in E-710 parameter memory at addresses 116 and 117. See the D... Set/Read PIO High/Low Limit command, p. 103 for details on reading or writing these values. Note that these commands are sent over the RS-232 or IEEE 488 interface, whichever has been activated.

The default values of *PIO Input Low Limit* and *PIO Input High Limit* are the same as the values of the *Range Limit min/µ* and *Range Limit max/µ* parameters, which are stored in the same parameter group.

In order to increase the resolution of the PIO input, it is possible to increase the value of *PIO Input Low Limit* and/or decrease the value of *PIO Input High Limit*.

Example 1:

PIO Input Low Limit = 0 µm
PIO Input High Limit = 100 µm

PIO Limits 0 µm and 100 µm

Position in µm	**PIO data**
0	0
25	16384
50	32768
75	49152
100	65535

PIO Limits 0 µm and 50 µm

Position in µm	**PIO data**
0	0
12.5	16384
25	32768
37.5	49152
50	65535

15 Command Survey

15.1 Native ASCII Commands (by type)

Mne-monic	Description	Type of Function	Restriction	Page
BR	Baud Rate change	Communication / Report		93
GA	Change GPIB (IEEE 488) address	Communication / Report		110
AE	DDL Auto Examine	Control (R/W)		91
AZ	Auto-Zero	Control		91
D...[*]	Set/Read Antivibration Filter	Control/Report	6-axis versions only	97
D...*	Set/Read AutoFocus Parameters	Control/Report	6-axis versions only	98
D…*	Set/Read AutoZero Enable Status	Control/Report		98
D...**	Set/Read Configuration Options	Control/Report		99
D...*	Set/Read Curve Control	Control/Report		99
D...*	Set/Read Filter Bandwidth	Control/Report		100
D...*	Set/Read Filter Enable	Control/Report		101
D...*	Set/Read Flatness-Compensated Axis	Control/Report	6-axis versions only	100
D...*	Set/Read Input Trigger Mode	Control/Report		101
D…*	Set/Read PIO Axis Write Enable	Control/Report		103
D…*	Set/Read PIO High/Low Limit	Control/Report		103
D...*	Set/Read PIO ON-Target4 Line Function	Control/Report	PIO models only	104
D...*	Set/Read Proportional Gain	Control/Report		104
D...*	D... Set/Read Sensor Transformation Matrix Values	Control/Report		105
DP	Data Protection	Control		95
DW	Data Write	Control	if EEPROM selected by DP**	97

[*]"D..." commands are combinations made from DP, DR and DW commands. Type them very carefully, especially when writing values to EEPROM.

Mne-monic	Description	Type of Function	Restriction	Page
DZ	Specify a numbered correction delta value	Control (R/W)	6-axis versions only	105
EC	Enter Compensation Coordinates	Move/Control	6-axis versions only	106
ES	DDL Auto Examine Setup	Control (R/W)		107
MD	Reset baseline position	Control		127
NP	Specify Flatness-Compensation Number of Points	Control (R/W)	6-axis versions only	129
OG	Enter Crosstalk Error Factor	Control (R/W)	6-axis & some 4-axis versions[*]:	130
RP	Repeat	Control		133
SL	Servo-Loop	Control/Report		137
SO	Setup for Crosstalk Compensation	Control (R/W)	6-axis & some 4-axis versions*	138
SP	Set P-Term	Control		138
SS	Set Target for Move All Together	Control (R/W)	6-axis & some 4-axis versions[**]	138
TR	Table Rate	Control		143
WA	Wait	Control		147
FC	Auto-FoCus	Move	6-axis versions only	107
GH	Go Home	Move	only with Servo ON	112
IP	Impulse Response	Move		119
MA	Move Absolute	Move/Report	only with Servo ON	123
MR	Move Relative	Move	only with Servo ON	128
MS	Move All Together	Move	6-axis & some 4-axis versions**	129
SM	Proportional Move	Move		137
ST	Step Response	Move		139
SV	Set Velocity	Move		140
VR	Voltage Relative	Move	only with Servo OFF	145
VS	Voltage Set	Move	only with Servo OFF	146
DR	Data Read	Report		96

[*] 4-axis firmware versions 5.028 and higher but not 6.0 to 6.027

[**] 4-axis firmware versions 5.026 and higher but not 6.0 to 6.025

Mne-monic	Description	Type of Function	Restriction	Page
GI	Get Information	Report		113
HE	Help	Report		119
TA	Tell A/D Value	Report		141
TP	Tell Position	Report		142
TS	Tell Sensor Position	Report		143
TT	Tell report Table	Report		144
TV	Tell Velocity	Report		144
VT	Voltage Tell	Report		146
PN	PaNel Update	Service		131
TN	Tell Normalized sensor value	Service		142
UP	UPload from Controller	Service		145
CF	Connect to wave generator (R/W)	Wave generation		94
CP	Curve Points (R/W)	Wave generation		95
FO	Curve Function Offset (R/W)	Wave generation		108
FS	Download a waveform (Fill Segment)	Wave generation		109
FT	Wave generator Trigger Setting	Wave Generation Trigger		109
GC	Generate Ramp Curve	Wave generation		111
GL	Generate Single Scan Line	Wave generation		115
GS	Generate Sine Wave	Wave generation		118
KT	Kind of Trigger Signal	Wave Generation Trigger		120
MC	Output Waveform Continuously	Wave Generator Move		124
PA	Curve Start Point (R/W)	Wave generation		131
PC	Curve Centerpoint (R/W)	Wave generation		131
PS	Points for Speed Up/Slow-Down Zone (R/W)	Wave generation		132
PT	Waveform/Segment Points Total	Wave generation		132
RN	Repeat Number of Waveform cycles	Wave generator move		133
RT	Wave generator Run sTop	Wave generator move		134
SC	Single Cycle Output	Wave generator move		134
SF	Select Waveform for Wave Generator (R/W)	Wave generation		136

Mne-monic	Description	Type of Function	Restriction	Page
TL	Test curve moving/DDL Report	Wave generator move		141
LT	Define DDL Data Table	DDL	6-axis & some 4-axis versions*	122
LS	DDL Setup	DDL	6-axis & some 4-axis versions*	121
LN	DDL Repeat Number and Response Value	DDL	6-axis & some 4-axis versions*	120
LP	DDL Initialization Phase Correction	DDL	6-axis & some 4-axis versions*	121

* 4-axis firmware versions 5.028 and higher but not 6.0 to 6.027

15.2 PIO Commands

PIO commands take the form of electronic signals on the optional PIO interface. The following operations are supported, and offer a substantial speed advantage over their ASCII alternatives. The PIO is not used by current versions of *NanoCapture™*, nor is it available on 6-axis E-710s.

PIO Operation (see p. 85)	**ASCII Alternative**
Write Target	MA
Read Position	TP
Read On-Target Signal	GI (code 18)
Read Flags	GI

* 4-axis firmware versions 5.017 and higher but not 6.0 to 6.016

16 Native Command Reference (alphabetical)

AE DDL **A**uto **E**xamine (6-axis versions; 3- and 4-axis versions: firmware rev.5.035/6.035 or newer) **AE**

Command Type:	Control (R/W)
Description:	Checks residual (tracking) error, and if necessary performs a DDL initialization run.
Format:	[*axis*]**AE**[*switch*].
Arguments:	*axis* is the selected axis; it must be connected with one of the two wave generators.

switch has following functions

switch = 1:	the average residual error of the waveform is reported.
switch = 2:	the standard deviation value of the waveform is reported.
switch = 16:	compensated data will written to the DDL table and the value 1701736292 (string “done” in long format) reported after the process is finished

See also "DDL Auto Examine" p. 71.

AZ Auto Zero **AZ**

Command Type: Control Command

Description: Performs an automatic zero-point calibration for either a specified linear axis, or for all axes for which AutoZero Autostart is enabled (see D... Set/Read AutoZero Enable Status p. 98). This procedure lasts several seconds. Its completion state can be read with the GI command (p. 113), parameter 18. See “AutoZero,” p. 79 for information on why and when to perform an AutoZero.

When AutoZero is to be performed for an axis which is connected to the Analog Input for target generation, the sensor-to-axis matrix for that axis must be changed. See p. 80 for details.

Format: [*a*]**AZ**[*v*]

Arguments: *a:* axis_code

If *a* is zero all axes for which AutoZero Autostart is enabled are autozeroed.

If *a* is not zero, the axis it specifies is autozeroed whether it has AutoZero Autostart enabled or not.

With 6-axis versions or with firmware equal or newer than [5|6].011 (3- and 4-channel controllers), *a* = 100 + the axis specification (0-6), signifies that the new offset calibration(s) will be saved to EEPROM as the new default value(s). **Warning: The repeat write times of non-volatile memory is limited. Do not use the +100 option except when necessary.**

v: is “axis low voltage,” i.e. the axis component “voltage” which, when run through the coordinate transformation and the power amplifier, gives the voltage to be put on the PZT when the axis minimum position is commanded.

If the *v* parameter is omitted, the default *low_voltage* values (memory addr. 120) will be used (with *a* = 0 or 100, the *low_voltage* could be different for every axis).

Response:	none (the GI command, p. 113, parameter 18, can be used to determine if AZ has finished)
Troubleshooting:	if *a* $\geq$ 100 with firmware versions < [5|6].011, AZ will not run If AutoZero Autostart is enabled for one or more axes, the auto-zero procedure can be set to run automatically after power-up (see D... Set/Read AutoZero Enable Status, p. 98). If the changes are too large, the sensors may go out of range (overflow). In this case a *low_voltage* value must be so selected that it brings the sensor working range back to the normal range.

AutoZero Procedure Details:

Note that when AutoZero is to be performed for an axis which is connected to the Analog Input for target generation, the sensor-to-axis matrix for that axis must be changed first. See p. 80 for details.

1. All affected axes are moved to target position 0.
2. All affected axes are set to voltage-control mode (servo OFF).

Then the following steps are run through once for each affected linear axis (for non-linear axes, the sensors will be zeroed at zero voltage without that axis having been driven forward and backward):

3. All PZTs having a component in the selected-axis direction are driven slowly to the zero voltage.
4. All PZTs having a component in the selected-axis direction are moved slowly to the position corresponding to the axis *low_voltage* (either to the one in the command, or, if omitted, to the respective default value at memory addr. 120).
5. All PZTs having a component in the selected-axis direction are moved slowly to the position corresponding to the axis upper voltage, defined at memory addr. 121.
6. Now the operating voltage is decreased stepwise until the value corresponding to *low_voltage* is reached again.
7. Wait one second.
8. All sensor channels with components in the selected-axis direction get a new offset value, so that they show a value of zero. With firmware versions 5.011 and higher (except 6.010): If the axis specification was given as axis number plus 100, the new offset will be saved to non-volatile memory as the new default value.
9. Servo-control is turned back ON.

For environments with temperature changes, the original length of the PZTs changes too. In closed-loop mode the PZTs therefore need an offset voltage to hold the original zero position. Because of this offset voltage, the PZTs might not be able to cover their full nominal range. To enable covering the full range, after the auto-zero procedure the auto zero function changes the sensor offset value in such a way that the axis position shows zero.

It is recommended that AutoZero be run if the system is exposed temperature changes and the application requires the full movement range. If the user's system requires a fixed zero position, AutoZero should be carried out only once and the results saved as the new defaults during user's system initialization.

Warning! The repeat write times of non-volatile memory is limited. Do not use the +100 option (available only with firmware versions 5.011 and higher, but not 6.010) except when necessary.

Example:

After a new autozero (*low_voltage* = 0), the sensor overflows at the upper position (TA = +32767, +65535 or about TN =+100. Depending on controller type, the way to see sensor overflow is to make a number of TA reports in rapid succession: normally the readings will differ slightly; if the values are all the same, then the sensor is at its limit). To resolve this problem, perform AutoZero again with a new *low_voltage* (e.g. *v* = -10).

To set axes to “auto-zero enable” use the following command:

*axis*DP-1,4DW1

or disable it

*axis*DP-1,4DW0

Sample commands:

- To perform an AutoZero for all axes usingwith default *low_voltages*:
  0**AZ**
- To perform an AutoZero for all axes with default *low_voltages* and save them as default (only with firmware versions 5.011 and higher, except 6.010):
  100**AZ**
- To perform an AutoZero for axis 2 with default *low_voltage*:
  2**AZ**
- To perform an AutoZero for axis 2 with user-defined *low_voltage* –5–5 V:
  2**AZ**-5
- To perform an AutoZero for axis 2 with user defined *low_voltages* –5–5 V and save it as default (only with firmware versions 5.011 and higher except 6.010):
  102**AZ**-5

BR set Baud Rate **BR**

Command Type:	RS-232 Interface Configuration (R/W) / Report
Description:	Change RS-232 interface baud rate of the controller. **Note: Interface software provided by PI like *NanoCapture*™ and *WinTerm* and the GCS DLL sends this command automatically when the baud rate of the host computer is set or changed.**
Format:	[*switch*]**BR**[*baudrate*]
Arguments:	*switch* = 0 or omitted: change current baud rate but do not store it in non-volatile memory. *switch* = 1: change current baud rate and store it in non-volatile memory as power-up default.
Valid baud rates are:	9600, 19200, 38400, 57600, 115200. If communication is not perfect after the baud rate has been changed (the host

	computer's baud rate must be changed too), the controller must be turned OFF and ON to come back to the power-on default baud rate. If the power-on default is no longer known or unusable, the controller can be turned on, then off while the power LED is blinking then on again. After this sequence, the factory default of 9600 baud will be active. This will also set the GPIB address to the factory default value of 4. As an alternative, when the E-710 baud rate is unknown, the host computer baud rate can be varied until communication is established. **Interface software provided by PI, including the GCS DLL, uses this stategy when establishing communication.**
Response:	If the baudrate parameter is omitted, the command reports the current baud rate.
Troubleshooting:	Invalid baud rate specified, the controller will then use the default baud rate of 9600. PI interface software "Connection Failed" message: check cabling, retry at different baud rates, restart software.

Example (with non-PI interface program):

1. Turn on E-710.
2. Open host PC COM PORT with a baud rate of 9600.
3. Send: BR115200 followed by line feed.
4. Close the COM PORT.
5. Open COM PORT with the baud rate of 115200.
6. Send: GI followed by line feed (to evaluate communications).
7. Read two strings from COM PORT (example):
   " Digital Piezo Controller V5.015",
   " (c) 2002 Physik Instrumente GmbH"

   If OK then use this baud rate.
8. Send: "1BR115200" to store the baud rate in non-volatile memory as new power-on default.

CF Connect to Wave (Function) generator **CF**

Command Type:	Wave Generation (R/W)
Description:	connect a wave generator to an axis.
Format:	*axis*CF[*generator*]
Arguments:	*axis* is the selected axis or 0. When *axis* = 0, the wave generator is disconnected. *generator* (if present) = 1 or 2, for selecting one of two wave generators.
Response:	If *generator* is omitted, then the command reports the number of the wave generator connected to *axis* or 0 if none. If *generator* is a valid wave generator number, and if an axis is connected to that generator, the axis is immediately moved to a new position which is the sum of the axis

	baseline position value (see MD command, p. 127) and the offset value of the first segment in the selected waveform (see FO, p. 108 and SF, p. 136). If *generator* is 0, the axis is disconnected from the generator and the axis is moved back to its baseline value.
Troubleshooting:	Invalid axis, or axis not defined as curve moveable.

CL Clear DDL data **CL**

Command Type:	DDL (later than version 1.01)
Description:	clear the DDL data table.
Format:	*table***CL***switch*.
Arguments:	*table* is the selected DDL table, or *table*=0 to select all tables.
	switch = 0 : clear only DDL table.
	switch = 1*:* clear both DDL table and waveform
Response:	none.

CP Curve Points **CP**

Command Type:	Wave generation (R/W)
Description:	Define the number of points to use for generating a curve in the specified segment. The curve will begin at the point specified in the PA command (p. **131**), have its centerpoint at the specified in the PC (p. **131**) command, and have the size and shape specified in the GL (line, p. **115**), GC (ramp, p. 111) or GS (sine, p. 118) command.
Format:	*segment*CP[*curvepoints*]
Arguments:	*segment,* number of the segment being defined
	curvepoints is the number of points in the curve; it must be less than or equal to the total number of points in the segment, as specified in the corresponding PT command.
Response:	If *curvepoints* omitted reports the current *curvepoints* setting.

DP Data Protection **DP**

Command Type:	Control
Description:	This command selects the bank of memory (usually axis- or channel-related) to which the following read (DR) and write (DW) operations will refer. The DP command also selects the type of memory which is accessed by the read and write commands (non-volatile memory or RAM).
Format:	*a*DP*n*
Argument:	*a:* Axis or channel identifier *n:* code specifying RAM (0, default), EEPROM+RAM (-1)
Troubleshooting:	- Axis identifier not valid
	- Incorrect code

Details:

On power up the parameters are copied from non-volatile memory into RAM. During normal operation the E-710 uses the parameter values which are stored in RAM. Therefore the contents of the RAM will determine the behavior of the E-710. On the next power-up the data in RAM will be replaced by the default parameters from the non-volatile memory.

All parameters are stored in a non-volatile memory chip. On power up the parameters are copied into another memory which is called RAM. During normal operation the E-710 uses the parameter values which are stored in the RAM. Therefore the contents of the RAM will determine the behavior of the E-710. On the next power up the data in RAM will be replaced by the default parameters from the non-volatile memory.

Data in the non-volatile memory can be read any time to get the power-on defaults. **Except for the "D..." commands explicitly given, the contents of the non-volatile parameter memory should be modified by authorized persons and service engineers only.** Be aware that writing to the parameter memory will in most cases overwrite original factory-set data.

According to the non-volatile parameter memory chip manufacturer's specifications, only a few thousand write cycles can be executed successfully. Frequent re-configuring should therefore be avoided.

Intervening DP, DW or DR commands referring to other axes may change the DP code setting, so always issue a new DP command.

DR Data Read **DR**

Command Type:	Report
Description:	This command reads an address in the type of memory selected in the last DP command. The reported value is that associated with the axis/channel most recently selected in a DP command.
Format:	*a*DR
Argument:	*a*: Address, range 1-256
Report.	Contents of the indicated address, in either exponential format, (sign + 7-char mantissa + 2-char exponent) or integer format, depending on address.
Troubleshooting:	- Address not in the range 1-256 - invalid channel/axis selected (use DP to select channel/axis)

DW Data Write **DW**

Command Type:	Control
Description:	This command allows writing data to the memory specified in the previous DP command. If EEPROM+RAM was selected by the DP command the data is written into the non-volatile memory as well as to RAM. The most-recently sent DP command also determines the axis/channel for which the data is written.
Format:	*a*DW*v*
Argument:	*a:* Address to be written to *v:* data value to write Both parameters are required, Blanks are not allowed. *a* must be an integer from 1 to 256, *v* may be in integer or floating point representation.
Troubleshooting:	- Address not valid - Data value not valid - Axis specification in DP command was invalid

Details*:*

The DW command is used to configure and calibrate the system and, except for the examples given in this manual, may only be used by PI personnel.

Any write access can modify factory settings in the non-volatile memory or the volatile memory which is called RAM (see also the description of the DP command), if the values are in the allowed range. Most values in RAM will be updated immediately, some only at power-on.

D... Set/Read Antivibration Filter (6-axis versions only) D...

Command Type:	Control/Report
Description:	Set or read the parameters for the antivibration filter. This filter filters the (dynamic) target position just before it is fed into the axis' servo-controller. It is designed to prevent exciting vibration of the entire stage in its mounting. It operates on the same principle as the other notch filters. This is a compound command using the DP, DR and DW commands.
Format (to set):	*axis***DP***option***,***parameter***DW***value*
Format (to read):	*axis***DP***option***,***parameter***DR**
Arguments:	*axis:* axis identifier *option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results) *parameter:* 233 mid-frequency (notch frequency) 234 damping, range 0.02 to 1,0 235 bandwidth,, normally 1

value	setting for *parameter*

D... Set/Read AutoFocus Parameters (6-axis versions only) D...

Command Type: Control/Report

Description: Set or read the current AutoFocus parameters (see p. 44 for a description of the Auto Focus feature).

This is a compound command using the DP, DR and DW commands.

Format (to set): **7DP***option***,***parameter***DW***value*

Format (to read): **7DP***option***,***parameter***DR**

Arguments: *option:* if 0, set/read RAM only
if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results)

parameter:
62 Auto focus mode (*value* = 1 for random method, 0 for zero-crossing)
30 Resolution (*value*=resolution = $1/2^n$, where n is the number of steps
34 Step delay maximum
38 Step delay minimum

value parameter value set: interpretation depends on parameter

Response to the read format only: 0 for disabled, 1 for enabled

Example:

To set the number of steps to 8 and store as power-on default, send: 7DP-1,30DR0.0039, since $1/2^8$ = 1/256 = 0.0039

D... Set/Read AutoZero Enable Status D...

Command Type: Control/Report

Description: Set or read the current AutoZero enable status (see p. 79 for a description of AutoZero).

This is a compound command using the DP, DR and DW commands.

Format (to set): *axis***DP***option***,4DW***value*

Format (to read): *axis***DP***option***,DR**

Arguments: *axis:* axis identifier

option: if 0, set/read RAM only
if -1, set/read EEPROM, (power-up defaults) and RAM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results)

value 1 for enable, 0 for disable

Response	to the read format only: 0 for disabled, 1 for enabled

D... Set/Read Configuration Options D...

Command Type:	Control	
Description:	Set or read the current configuration options (see p. 71 for a description of the options concerned). This is a compound command using the DP, DR and DW commands.	
Format (to set):	*axis***DP-1,***number***DW***value*	
Format (to read):	*axis***DP-1,***number***DR**	
Arguments:	*axis:*	axis identifier
	number:	configuration option number:
	2	Servo-Control enable OFF
	3	Servo-Control Autostart OFF
	4	Auto Zero Autostart Enable OFF (also affects **0AZ** command, see p. 91) Auto Zero Autostart OFF
	value	1 for enable, 0 for disable
Response	to the read format only: 0 for disabled, 1 for enabled	

Since these options all concern the state at power-up, they are always stored in non-volatile memory as well as RAM. Type carefully, as improper changes can have unforeseeable results.

When setting or reading more than one parameter for the same axis, the "*axis*DP-1," portion of the second and subsequent commands can be omitted.

D... Set/Read Curve Control D...

Command Type: Control

Description: Set or read the current curve control parameters for the specified axis. These settings affect the operation or failure of other commands. If, for example, DDL is not enabled here, DDL commands will fail.

This is a compound command using the DP, DR and DW commands.

Format (to set): *axis***DP***option***,130DW***curvecontrol*

Format (to read): *axis***DP***option***,130DR**

Arguments: *axis:* axis identifier
option: if 0, set/read RAM only
if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results)

curvecontrol: bitmapped value interpreted as follows:

bit 0 to bit 3		currently unused.
bit 4	1	axis enabled for waveform move
bit 5	1	axis enabled for synchronous move

	bit 6	**1**	**axis enabled for DDL***
	bit 7	**1**	**axis enabled for synchronous DDL***

Response — If read format is used, reports the curve control setting as an integer from 0-255. Convert value to binary to see the bits (the Windows calculator in scientific view can be used). **DDL bit 6 contributes decimal 64, synchronous DDL, decimal 128**.

*DDL enable settings are allowable but ineffective without DDL. See D... Set/Read DDL license Status on p. 102.

D... Set/Read Flatness-Compensated Axis (6-axis versions only) D...

Command Type: Control

Description: Specify or report the E-710 logical axis to benefit from flatness compensation.

This command is used as part of a command series required for the external-axis flatness compensation feature. This series also comprises the EC (p. 106), NP (page 129), DZ (p. 105) commands. See p. 44 for complete description and an example.

Format (to set): *axis***DP***option***,163DW1.0**

Format (to read): *axis***DP***option***,163DR**

Arguments: *axis*: axis to which flatness correction deltas can be applied as a function of the positions of two external axes

option: if 0, set/read RAM only
if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results)

D... Set/Read Filter Bandwidth for Curve Moving Report D...

Command Type: Control

Description: Set or read bandwidth of the filter that can be enabled to filter values before storing them in a Curve Moving Report Table (see TL command, p. 141). This is a compound command using the DP, DR and DW commands.

Format (to set): *axis***DP***option***,159DW***bandwidth*

Format (to read): *axis***DP***option***,159DR**

Arguments: *axis:* axis identifier
option: if 0, set/read RAM only
if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM, Use sparingly as EEPROM lifetime write cycles are limited to a few thousand. Type carefully, as improper changes can have unforeseeable results!)

bandwidth: value is used internally as a factor applied to the control bandwidth. The factory default is 2.

Response — to the read format only, bandwidth setting

D... Set/Read Filter Enable for Curve Moving Report D...

Command Type:	Control
Description:	Set or read filter-enable status for filtering of values to be stored in a Curve Moving Report (see TL command, p. 141). This is a compound command using the DP, DR and DW commands.
Format (to set):	*axis***DP***option***,158DW***filter*
Format (to read):	*axis***DP***option***,158DR**
Arguments:	*axis:* axis identifier *option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM (Set sparingly, as EEPROM lifetime write cycles are limited to a few thousand. Type carefully, as improper changes can have unforeseeable results!) *filter:* 0 for disabled, 1 for enabled
Response	to the read format only: filter enable status setting

D... Set/Read Input Trigger Mode D...

Command Type:	Control
Description:	Set or read the input trigger mode. For a description of the input trigger usage see Section "Input Trigger Signals" on p. 58.
Format (to set):	1**DP***option***,231DW***enable_code*
Format (to read):	1**DP***option***231DR**
Arguments:	*option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results) *enable_code:* bitmapped value interpreted as follows: bit 0 = 0/1, D_IN0 disable/enable* bit 1 = 0/1, D_IN1 disable/enable bit 2 = 0/1, D_IN2 disable/enable ... bit 7 = 0/1, D_IN7 disable/enable* bit 8 = 0/1, D_IN0 anytime/one-time* (E-710.6CD only) bit 9 = 0/1, D_IN1 anytime/one-time (E-710.6CD only) ... bit 15 = 0/1, D_IN7 anytime/one-time* (E-710.6CD only)

*Only IN1 and IN2 are currently implemented.

The factory default mode of all input triggers is disabled.

Example:

To enable D_IN1 until the next power-down, send the command

1DP0,231DW2	for anytime
1DP0,231DW514	for one-time only (E-710.6CD only)

D... Set/Read DDL license Status D...

Command Type:	Control (R/W)
Description:	To activate the DDL feature on 3- and 4-axis versions, a *license* number is needed. DDL licenses are device-specific and can be purchased from PI. After enabling the DDL feature and entering the license with the command sequence listed below, the controller must be switched off and powered up again before DDL can be used. This is a compound command using the DP, DR and DW commands.
Format (to enable):	*axis***DP**-1,130**DW**240 (enter for all axes) 1**DP**-1,238**DW***license* Notes: The settings are saved to EEPROM and become the power-on defaults. The *axis***DP**-1**,130DW**240 sequences enable not only the DDL feature, but also waveform move, synchronous move and synchronous DDL for the individual axes to guarantee optimum performance (the DDL feature works only in conjunction with the wave generator). See also the D... Set/Read Curve Control on p. 99.
Format (to disable):	*axis***DP**-1,130**DW**0 (enter for all axes) 1**DP**-1,238**DW**0 Notes: The settings are saved to EEPROM and become the power-on defaults. The *axis***DP**-1**,130DW**0 sequences disable not only the DDL feature, but also waveform move, synchronous move and synchronous DDL for the individual axes. See also the D... Set/Read Curve Control on p. 99.
Format (to read):	*axis***DP**0,130**DR** to read the activation status of curve control properties 1**DP**-1,238**DR** to read the license number
Response:	To the read format only: If the license has been accepted by the controller and the DDL feature was successfully activated for the axes, the number 240 should be reported by *axis***DP**0**,130DR** commands. The license number is reported by the 1**DP-**1,238**DR** command.

D... Set/Read PIO Axis Write Enable D...

Command Type:	Control
Description:	Normally all four axes of the PIO input are updated at the same time, regardless of whether the user modifies the target values of all the four axes or only some of them. That means that a value which has previously been written through the serial interface (RS232) or IEEE488 (GPIB) is overwritten by the values on the PIO for the axis. To prevent this, the PIO write operation can be enabled for each axis separately. This command is available with firmware versions 5.021 or higher, except 6.00 - 6.020.. This is a compound command using the DP, DR and DW commands.
Format (to set):	*axis***DP***option***,***232***DW***value*
Format (to read):	*axis***DP***option***,***232***DR**
Arguments:	*option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results) *axis:* axis identifier *value* 1 for enable, 0 for disable
Response	to the read format only: 0 for disabled, 1 for enabled

Use sparingly, as EEPROM lifetime write cycles are limited to a few thousand. Type carefully, as improper changes can have unforeseeable results.

D... Set/Read PIO High/Low Limit D...

PIO option required. The default values of *PIO Input Low Limit* and *PIO Input High Limit* are the same as the values of the *Range Limit min/μ* and *Range Limit max/μ* parameters, which are stored in the same parameter group.

Description:	The PIO high and low limits, used in converting data on the PIO port to axis positions, can be read out or set. See PIO Data Format and Scaling, p. 86 for details. This is a compound command using the DP, DR and DW commands.
Format:(to set)	*axis***DP0,***limit***DW***value*
Format:(to read)	*axis***DP0,***limit***DR**
Arguments:	*axis:* axis identifier *limit:* 116 for PIO_Input_Low_Limit 117 for PIO_Input_High_Limit *value* new limit setting
Response	to the read format only: current limit setting.

Note: this command sequence sets the working value in RAM only. The values of *PIO Input High Limit* and *PIO Input Low Limit* can also be read by the *NanoCapture™* program. The values are stored in the *Axis 1 Servo Setup* to *Axis 6 Servo Setup* parameter groups.

D... Set/Read PIO ON-Target4 Line Function D...

PIO models only, firmware newer than 5.024/ or 6.022 required, 5.032/6.032 for Servo All-On indicator mode.

Description:	This setting determines the function of whether the On-Target4 line of the PIO connector and the corresponding LED. It can operate as the On-Target signal for axis 4, or whether it operates as an “output data-valid” signal indicating that the PIO output data has been refreshed, or as a “servo all-on” signal indicating that all axes have servo-control ON. See "Flags with Supplemental LED Display and PIO Readout", p. 34 for the on-target4 signal modes and "PIO Read Operation Timing", p. 169 for more information on the data-valid mode. This is a compound command using the DP, DR and DW commands.
Format (to set):	5**DP***option***,**233**DW***value*
Format (to read):	5**DP***option*,233**DR**
Arguments:	*option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. (Set sparingly as EEPROM lifetime write cycles are limited to several hundred. Type carefully, as improper changes can have unforeseeable results) *value* 0: for on-target function 4: Servo All-On indicator mode, 8: for data-valid indicator mode
Response	to the read format only: 0 for on-target mode, 8 for data-valid mode. Type carefully, as improper changes can have unforeseeable results.

D... Set/Read Proportional Gain D...

Command Type:	Control
Description:	The proportional gain value of an axis is a factor applied to the target in a proportional move (SM) command to obtain the target for the axis (p. 99). See the SM command (p. 137) for a proportional move example. This is a compound command using the DP, DR and DW commands.
Format (to set):	*axis***DP***option***,62DW***gain*
Format (to read):	*axis***DP***option***,62DR**
Arguments:	*axis:* axis identifier *option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and

	RAM, or read EEPROM (Set sparingly, as EEPROM lifetime write cycles are limited to a few thousand. Type carefully, as improper changes can have unforeseeable results!)
	gain: proportional gain value
Response	to the read format only: proportional gain value.

D... Set/Read Sensor Transformation Matrix Values D...

Command Type:	Control
Description:	The sensor-to-axis transformation matrix values can be set or read in RAM and EEPROM. This is a compound command using the DP, DR and DW commands.
Format (to set):	*axis***DP***option***,***SensorCode***DW***contribution*
Format (to read):	*axis***DP***option***,***SensorCode***DR**
Arguments:	*axis:* axis identifier *option:* if 0, set/read RAM only if -1, set EEPROM (power-up defaults) and RAM, or read EEPROM. Type carefully, as improper changes can have unforeseeable results!) *SensorCode*: code designating a sensor channel: 50 for sensor ch. 1 51 for sensor ch. 2 51 for sensor ch. 3 51 for sensor ch. 4 160 for sensor ch. 5 161 for sensor ch. 6 162 for sensor ch. 7 163 for sensor ch. 8 (some channels may not be implemented in hardware) *contribution*: component of specified sensor in the *axis* direction
Response	to the read format only: current *contribution* value.

DZ Specify a Flatness Correction Delta Value (6-axis versions only) **DZ**

Command Type:	Control (R/W)
Description:	Specify a numbered correction delta value. This command is used as part of the D... Set/Read Flatness-Compensated Axis (p. 100), EC (p. 106), NP (page 129), DZ (p. 105) command series of the external-axis flatness compensation feature. See p. 44 for complete description and an example.
Format:	[*index*]**DZ**[*value*].
Arguments:	*index*: index number of the correction delta to specify (1 to 625). If *index* = 0 or is omitted, a value 1 greater than that used for

the previous DZ command is used (if there was no previous DZ command, 1 is used).

value: new correction delta value (in the unit of the axis to which it will be applied). If omitted, current value is reported

Response: If *value* is omitted, the value associated with the current or specified *index* will be reported in fixed-point decimal format and axis units: ±xxxx.xxxx,

Note: The total number of correction deltas that need to be stored depends on the external axis compensation range limits and the number of grid points. These parameters are specified with the EC (p. 106) and **NP** commands (p. 129)

EC Enter **C**ompensation Coordinates (6-axis versions only) **EC**

Command Type: Control/Move/Report

Description: This command is used as part of the D... Set/Read Flatness-Compensated Axis, (p. 100), EC , NP (p. 129), DZ (p. 105) command series of the external-axis flatness compensation feature. See p. 44 for complete description and an example.

The EC command has 3 distinct functions, depending on *code*

- Specify a travel limit for one of the external axes, OR
- Input the current position of external axis 1 OR
- Input the current position of external axis 2 AND execute the corresponding compensating move on the compensated E-710 axis.

Format: *code***EC***external_coordinate*

Arguments:

If	**Then:**
code = 1:	*external_coordinate* = current position of external axis 1
code = 2:	*external_coordinate* = current position of external axis 2
code = 101:	*external_coordinate* = the lower limit of the compensation area on external axis 1
code = 201:	*external_coordinate* = the upper limit of the compensation area on external axis 1
code = 102:	*external_coordinate* = the lower limit of the compensation area on external axis 2
code = 202:	*external_coordinate* = the upper limit of the compensation area on external axis 2

If *external_coordinate* is omitted

If *code* = 1 or 2, the start point of the compensation area is reported

If *code = 101 or 202,* the associated limit is reported

Response: If *external_coordinate* is omitted and *axis* = 101, 102, 201 or 202 the associated limit is reported

If *external_coordinate* is omitted and *axis* = 1 or 2, the reported value is the start point of the compensation area (see “DZ” on page 105).

ES DDL Auto Examine Setup[*] ES

Command Type:	DDL
Description:	This command defines the motion range for the DDL Auto Examine feature.
Format:	[*axis_code*]**ES**[*value*].
Arguments:	*axis_code* = 0, axis, or axis + 100.
axis_code = 0:	*value* is the axis to benefit from DDL Auto Examination; report table 1 is used in any case to save the error data, independent of the switch code in the **SC** or **MC** command. Note that the **TL** command can not save the error data correctly, and is not suitable for this test as the servo-control deviation is calculated differently. If *value* = 0, auto examination is disabled.
axis_code = axis:	*axis_code* is the selected axis, value is the lower limit (start point) of the range which will be examined. If *axis_code* is less than 1, the start point is automatically set to 1.
axis_code = axis+100:	the upper limit (end point) of the range which will be examined. If *axis_code* is less than the lower limit, the end point is automatically set to start point + 1.
value	the point number for the corresponding limit. If *axis code* ≠ 0 and *value* is omitted, the corresponding setting (point number) will be reported.
Response	see *value* above.

FC Auto-**F**o**C**us (6-axis versions only) FC

Command Type:	Move command (R/W)
Description:	Start the Auto-Focus algorithm, which tries to find the best focus position by moving the specified axis toward the point at which the Analog In (out-of-focus signal) is 0. To configure the Auto-Focus function, use the D... Set/Read AutoFocus Parameters command on p. 98. See "Auto Focussing" Section on p. 44 for a detailed description of the auto focus process. See the "Using the Analog Input" Section on p. 36 for details.
Format:	*axis***FC**[*stepsize*].
Arguments:	*axis* axis to move *stepsize* is the size of the first step in the auto focus process (see description p. 44).
Response:	If *stepsize* is omitted, the current level of Analog-In (out-of-focus signal) is reported.
Troubleshooting:	no signal on Analog IN; the axis is not in closed-loop.

[*] Six-axis versions and 4-axis versions with firmware rev.5.035 or higher, but not 6.00 to 6.034)

FL Download DDL Data or Fill DDL Table **FL**

Command Type:	DDL (R/W)
Description:	Download a DDL data value which has been previously generated by DDL (see **TL** command, p. 141) so as to output a waveform without a new initialization process. Or read a DDL data value An internal counter keeps track of the current position in the DDL data table. The counter is incremented each time a point is downloaded or read, and reset to the beginning if the FL command is issued with *DDL_table* = 0. Other commands may leave the counter in an unpredictable state.
Format:	*DDL_table***FL***DDL_table_value*.
Arguments:	*DDL_table* is the number of the DDL data table to be used (see **LT** command on p. 122). If *DDL_table* = 0 then the controller will reset the internal counter to the beginning position. *DDL_table_value* is the data which will be downloaded to the specified internal DDL data table. If *DDL_table_value* is omitted, the command reports a value from the table. The internal counter is incremented.
Response:	for read, a data point from the DDL data table will be reported.
Troubleshooting:	No DDL table has been defined (see **LT** command, p. 122.).
Examples:	
0FL	will reset the internal counter.
1FL0,RP1000	will set 1000 data points in DDL table 1 to 0.
0FL	will reset the internal counter again.
1FL,RP1000	will read all 1000 data points from DDL table 1.

FO Segment/Waveform Offset **FO**

Command Type:	Wave generation (R/W)
Description:	Specify or read the offset value of a segment or waveform (When waveforms are defined, each waveform takes on the offset of its first segment; the offsets can then be redefined using waveform numbers instead of segment numbers).
Format:	*segment***FO**[*offset*]
Arguments:	*segment,* number of the segment being defined *offset* is the start amplitude of the specified segment.
Response:	If *offset* is omitted, it reports the current offset value for *segment*.
Troubleshooting:	No valid curve-segment definition; For moving, the axis must be in closed-loop.

FS Download Data Point (Fill Segment) **FS**

Command Type:	Wave generation(R/W)
Description:	Load the next point of a user-defined segment to datapoint memory, or read the next point from a segment. An internal counter keeps track of the current position in the segment. The counter is incremented each time a point is loaded, and reset to the beginning if the FS command is issued with *segment* = 0. Other commands may leave the counter in an unpredictable state.
Format:	*segment*FS[*value*]
Arguments:	*segment* is the number of the curve segment. If *segment* = 0, the internal counter is reset to the beginning position. *value* is the value which will be stored in the next point of the specified segment.
Response:	If *value* is omitted, the command reports the value of the next datapoint of the segment and increments the internal counter automatically. Note that commands such as PT (p. 132) will reset the internal counter.
Troubleshooting:	- No valid segment definition; Internal pointer incorrectly set: to guarantee correct pointer values, it is important to initialize the pointer with 0FS first and then issue FS commands for all points in the segment sequentially. Intervening commands of other types or referring to other segments may destroy the pointer contents.

Examples:

0FS will reset the internal counter.

1FS0,rp1000 will set all 1000 data points of curve segment No. 1 to zero.

0FS will reset the internal counter once again.

1FS,rp1000 will read all 1000 data points from curve segment No. 1.

FT Wave Generator Trigger Setting **FT**

Command Type:	Trigger Configuration
Description:	Set trigger values for point(s) on the waveform.
Format:	*point***FT***setvalue*
Arguments:	*point* is the point within the waveform. It is independent of the wave generator. If the *point* = 0 then all points are selected and set to the value which is indicated by *setvalue*. The bits of *setvalue* have the following functions. Bit 0: trigger line 1: 0 not active, 1 active Bit 1: trigger line 2: 0 not active, 1 active Bit 2: trigger line 3: 0 not active, 1 active Bit 3: trigger line 4: 0 not active, 1 active Bit 8 if = 1, then the command sets the trigger value for all curve points between the last curve point

	changed by this command and *point* to the values specified in the corresponding bits of *setvalue.* The polarities of the trigger signals (active low or active high) are controlled by the KT command (see p. 120).
Response:	If *setvalue* is omitted, the command reports the current trigger line setting (3- and 4-axis versions firmware rev. 5.035/6.035 or newer)
Troubleshooting:	trigger signals are valid only while waveform movement is taking place (**SC**, **MC** or **LT** command). The trigger table is limited to 32768 points (6-axis version firmware rev. 2.14 or newer, 4-axis versions firmware rev. 5.035/6.035 or newer; 16384 points with older firmware versions).

Example 1: Waveform

0**FT**0	clears all waveform triggers
50**FT**1	activates trigger 1 (only) for point 50
1850**FT**1	activates trigger 1 (only) for point 1850

Example 2:

1**FT**1	activates trigger 1 (only) for curve point 1
2**FT**0	deactivates all triggers for curve point 2
120**FT**257 (bit 8 set)	activates trigger 1 (only) for curve points 3-120
300**FT**256 set)	deactivates all triggers for points 121-300 (bit 8

GA Change GPIB (IEEE 488) Address **GA**

Command Type:	GPIB (IEEE 488) Interface Configuration (R/W) or Report
Description:	Change GPIB interface Address.
Format:	[*switch*]GA[*Address*]
Arguments:	*switch* = 0 or omitted: changes current GPIB address but does not store it in non-volatile memory. *switch* = 1: change current GPIB address and store it in non-volatile memory.
Valid GPIB addresses:	1 to 15. If the power-on default GPIB address is no longer known or unusable, the controller can be turned on, then off while the power LED is blinking, then on again. After this sequence, the factory default of 4 baud will be restored. This will also set the RS-232 baud rate to 9600.
Response:	If *Address* parameter omitted, reports the current GPIB address.
Troubleshooting:	invalid GPIB Address, the controller will then use the factory default GPIB Address of 4, even if another default had been set.

GC Generate Ramp Curve **GC**

Command Type:	Wave generation
Description:	Generates a ramp curve in the specified segment. The ramp moves from the offset amplitude, set by the FO command (p. 108), to a maximum amplitude and back to the offset amplitude. Both slopes have a shape similar to that of a single scan line (GL command, below).This command clears all DDL data tables.
Format:	*segment***GC***amplitude*
Arguments:	*segment,* number of the segment being defined *amplitude* is the difference between starting amplitude and maximum amplitude.
Response:	None.
Troubleshooting:	No valid segment definition; For moving, the axis must be in closed-loop mode.

Example1: Command sequence for generating ramp curve in segment 2:

2**PT**2048	set total segment length
2**CP**1800	set number of points to use for curve
2**PC**900	set curve center point: half total for symmetrical
2**PA**100	set start point for curve
2**PS**50	set length of speed-up/slow-down zones
2**FO**0	set offset amplitude
2**GC**150	generate ramp 150 units high

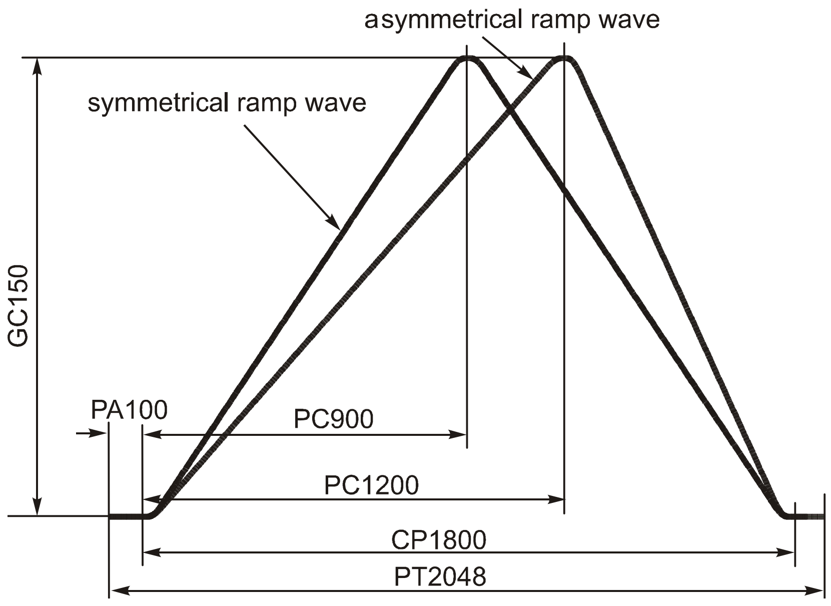


Fig. 52. Example 1: Ramp in Segment 2

Example 2:	When **CP** points plus **PA** points is larger than **PT** points, the curve exhibits a phase-shift:

2**PT**2048	set total segment length
2**CP**2048	set number of points to use for curve
2**PC**1024	set curve center point: half total for symmetrical
2**PA**100	set start point for curve
2**PS**2	(not required)
2**FO**0	set offset amplitude
2**GC**150	generate ramp 150 units high

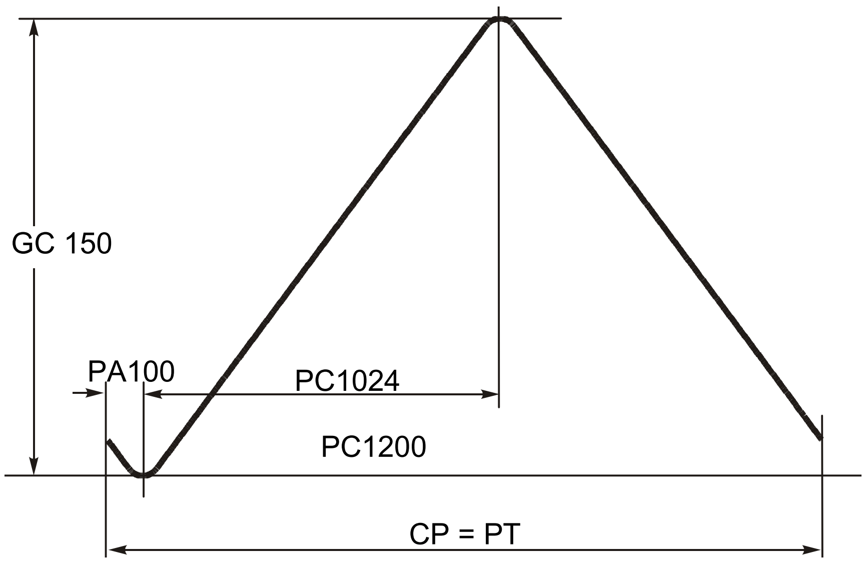


Fig. 53. Example 2: Ramp with Phase Shift

GH Go Home **GH**

Command Type:	Move Command
Description:	The selected axis gets zero as the new target position. The actuator moves with the programmed velocity towards it. If the previously commanded position has not yet been reached, the previous move is aborted and the new move begins immediately.
Format:	*a*GH
Arguments:	*a:* axis identifier The axis identifier is required and must be an integer.
Response:	none
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) servo-control disabled or OFF for this axis or sensor disabled Servo-control OFF or disabled for this axis (new target will not be accepted) Sensor not enabled If servo-control is enabled but turned OFF, issue an SL command to turn it ON New position outside of position limits

Position tolerance value too small (smaller than minimal incremental motion)

Incorrect control parameter, p-term zero or too small (see SP command)

Velocity too low or zero (see SV command)

GI Get Information GI

Command Type:	Report
Description:	This command reports the device status.
Format:	*a*GI[*v*]
Argument:	*a:* axis or channel identifier
	v: code for type of information requested (see table below)

Report Format:	Text Strings, length and contents depend on the code.
	OFF = 0
	ON = 1

	If all arguments are omitted, GI returns the following information (example for 6-axis version with firmware rev.2.13):
	Digital 6 CH Piezo Controller V2.13 \n
	(c) Copyright 2004 by Physik Instrumente (PI) GmbH & Co.\n

Details:

The GI function reports the system state. The information available is organized as follows:

Code *v*	**Bit**	**Meaning**
Omitted		Reports manufacturer, device, date and version of the firmware
1		Reports general configuration information: Servo-Loop OFF / ON Zoom Mode OFF / ON (not implemented) Sensor Enable (should always be ON) Control Enable OFF / ON (6-axis versions: Servo Enable OFF / ON) Servo-Loop Autostart OFF / ON Auto-Zero Enable OFF / ON
2		Reports sensor processing information Filter type (0:none, 1:IIR Bessel, 2:FIR, 99:IIR with own coefficients) Filter depth (2 @ IIR, 2-1000 @ FIR) Bandwidth of filter (unit Hz, only valid for type 1) Filter coefficient (only valid for type 99) Sensor sample period (unit s, required for IIR filter)
3		Reports Sensor Linearization Zoom Mode OFF / ON (not implemented) Parameter a0, automatically determined during zero-point calibration.

	Parameter a1, Sensor Gain Parameter a2-a4, Polynomial coefficients for 2nd to 4th degree linearization
4	Reports Servo-Controller Settings Zoom Mode OFF / ON (not implemented) Servo-Controller P-Term, value currently in use Servo-Controller Integration time Servo-Controller step size per ms (velocity) Notch filter center frequency Notch filter rejection Servo loop period (unit s, for I-Term and Notch filter)
5	Coordinate Transformation Sensor-to-axis transformation coefficients 3- and 4-axis:1-4 6-axis: 1-8 (ch. 8 may not be implemented in hardware) 3- and 4-axis: Reserved 1-4 Piezo transformation control matrix 3- and 4-axis:1-4 6-axis: 1-8
6	General Limits and Gains Zoom Mode OFF / ON (not implemented) Lower limit for Position Upper limit for Position Zoom Factor (factor for lower and upper limit) (Zoom mode not implemented) Minimum Piezo voltage Maximum Piezo voltage Lower Piezo voltage for AutoZero alignment Upper PZT voltage for AutoZero alignment Position tolerance for on-target indication
7	Password protection OFF / ON
8	Status word, depends on axis

Status Word
For 3- and 4-axis controllers with firmware revisions 4.xxx (except 4.019, 4.20 and 4.021) and for 5.018, the bits of the status word have the following meanings:

Bit 0:	0 = Servo on, 1 = Servo off
Bit 1:	0 = Piezo voltage inside limits,1 = Piezo voltage at limit
Bit 2:	0 = On Target: Position error smaller than tolerance 1 = Position error too large
Bit 3:	0 = Target position higher than low limit, 1 = Target Position at low limit
Bit 4:	0 = Target position smaller than high limit, 1 = Target position at high limit.
Bit 5:	0 = AutoZero is not running 1 = AutoZero is running (axis 1 only)
Bit 6:	not used
Bit 7:	0 = no error, 1 = last sent command not accepted, error during command processing. The bit is reset by this command.

For 3- and 4-axis controllers with firmware revisions 4.019, 4.020 and 5.xxx, 6.xxx (except 5.018 and 6.018) as well as all 6-axis versions, all bits are shifted 8 positions to the left

Bits 0 - 7:	Reserved
Bit 8:	0 = Servo on, 1 = Servo off
Bit 9:	0 = Piezo voltage inside limits, 1 = Piezo voltage at limit
Bit 10:	0 = On Target: Position error smaller than tolerance 1 = Position error too large
Bit 11:	0 = Target position higher than low limit, 1 = Target Position at low limit
Bit 12:	0 = Target position smaller than high limit, 1 = Target position at high limit.
Bit 13:	0 = AutoZero is not running, 1 = AutoZero is running (axis 1 only)
Bit 14:	0 = generator not running 1 = generator is running (axis 1 only)
Bit 15:	0 = no error 1 = last sent command not accepted, error during command processing. The bit is reset by this command

GL Generate Single Scan Line **GL**

Command Type:	Wave generation
Description:	Generates a single scan line which moves from the offset amplitude, set by the FO command (p. 108), to a final amplitude, with speed ramping up, holding constant and ramping down. This command clears all DDL data tables.
Format:	*segment***GL***amplitude*
Arguments:	*segment,* number of the segment being defined *amplitude* is the difference between starting amplitude and final amplitude.
Response:	None.
Troubleshooting:	No valid segment definition; For moving, the axis must be in closed-loop mode.

Example: Command sequence for generating single scan line in segment 3:

3**PT**2048 (p. **132**)	set total segment length
3**CP**1800 (p. 95)	set number of points to use for curve
3**PC**101 (p. **131**)	use at least 2x the size of the speed-up/slow-down zone (PS command)
3**PA**100 (p. **131**)	set start point for curve
3**PS**50 (p. **132**)	set speed-up / slow-down zone lengths
3**FO**0 (p. 108)	set offset amplitude
3**GL**150 (p. **115**)	generate scan line

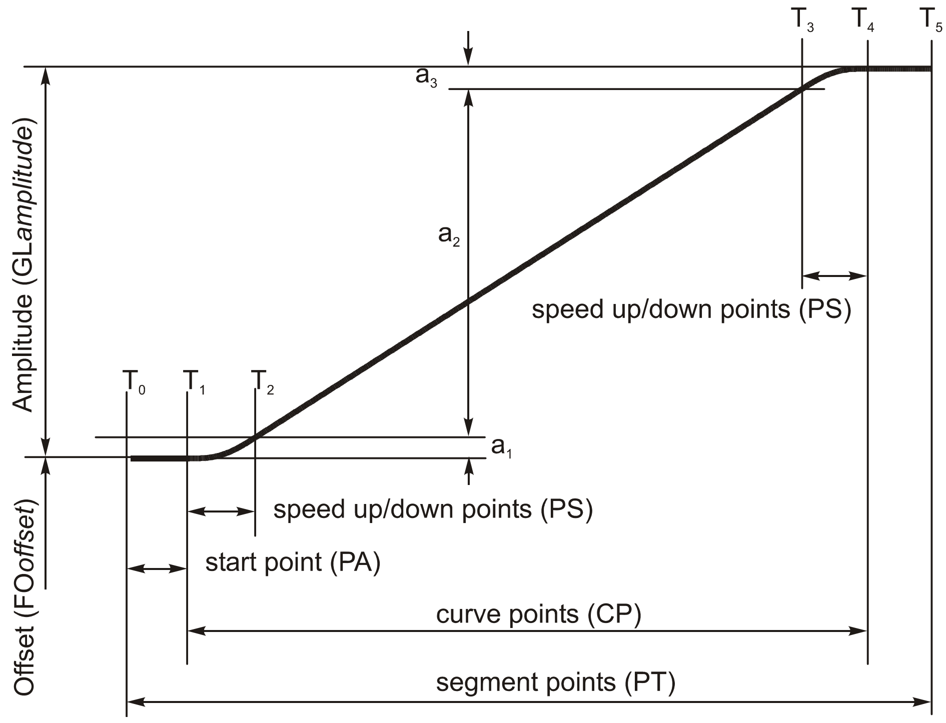


Fig. 54. Single scan line example

Definitions (from figure):

PA: curve starting point in current segment (T_0 to T_1) with amplitude = segment offset from FO command (p. 108).

CP = number of curve points

PS = number of speed-up / slow-down points

PT = segment points

T_{servo} = servo-cycle time, p. 28 (see PA command on p. 131).

Start, T_0 to T_1:

T_0 = 0 sec.

T_1 = (PA-1) * T_{servo}

$$n = 0 ... PA - 1$$

$$CurveForm[n] = FOoffset$$

Speed up zone, T_1 to T_2:

T_1 = PA * T_{servo}

T_2 = (PA+PS-1) * T_{servo}

$$n = 0 ... (PS - 1)$$

$$CurveForm[n + PA] = FOoffset + \frac{GLamplitude}{\frac{CP}{PS} - 1}\left(\frac{n}{2PS} - \frac{1}{2\pi}\sin\left(n\frac{\pi}{PS}\right)\right)$$

$$a_1 = \frac{GLamplitude}{2(\frac{CP}{PS} - 1)}$$

Constant speed zone, T_2 to T_3:

T_2 = (PA+PS) * T_{servo}

T_3 = (PA+CP-PS-1) * T_{servo}

$$n = 0...(CP - 2PS - 1)$$

$$CurveForm[n + PA + PS] = FOoffset + \frac{GLamplitude}{\frac{CP}{PS} - 1} * \left(\frac{1}{2} + \frac{n}{PS}\right)$$

$$Speed = \frac{2a_1}{T_{servo} * PS}$$

$$a_2 = GLamplitude * \frac{CP - 2PS}{CP - PS}$$

Speed down zone, T_3 to T_4:

T_3 = (PA+CP-PS) * T_{servo}

T_4 = (PA+CP-1) * T_{servo}

$$n = 0...(PS - 1)$$

$$CurveForm[n + PA + Cp - PS] = FOoffset + \frac{GLamplitude}{\frac{CP}{PS} - 1} * \left(\frac{CP - 2PS}{PS} + \frac{n + PS}{2PS} - \frac{1}{2\pi}\sin\left((n + PS)\frac{\pi}{PS}\right)\right)$$

$$a_3 = a_1 = \frac{GLamplitude}{2(\frac{CP}{PS} - 1)}$$

Final position, T_4 to T_5:

T_4 = (PA+CP) * T_{servo}

T_5 = (PT-1) * T_{servo}

$$n = (PA + CP)...(PT - 1)$$

$$CurveForm[n] = FOoffset$$

GS Generate Sine Wave **GS**

Command Type:	Wave generation
Description:	Generates a sine waveform in the current segment. Clears all DDL data tables.
Format:	*segment*GS*amplitude*
Arguments:	*segment,* number of the segment being defined *amplitude* is the difference between minimum and maximum amplitudes
Response:	none.
Troubleshooting:	No valid segment definition; For moving the axis must be in closed-loop.

Example 1: Command sequence for generating sine wave curve in segment 1:

1PT2048	set total segment length
1CP1800 (p. 95)	set number of points to use for curve
1PC900	set curve centerpoint: half total for symmetric
1PC1200	for asymmetric wave shown
1PA100 (p. **131**)	set start point for curve
1PS1	(not required)
1FO0	set offset amplitude
1GS150	generate sine wave

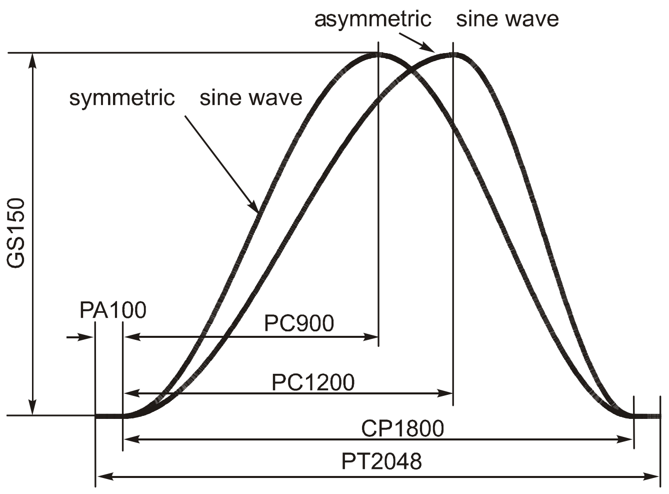


Fig. 55: Sine Wave Example 1

Example 2: If the CP value plus the PA value is larger than the PT value, the curve exhibits a phase-shift, just as with the ramp curve.

1PT2048
1CP2048 (instead of
1800 above)
1PC1024
1PA100
1PS1 (not required)
1FO0
1GS150

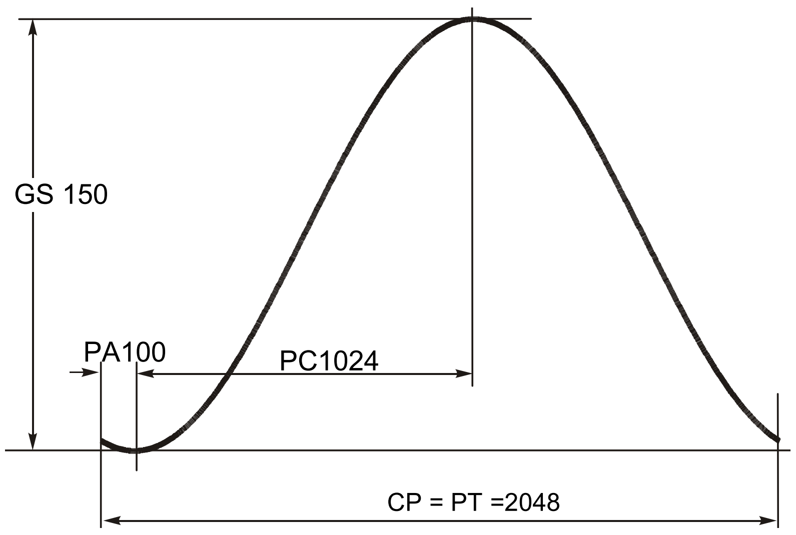


Fig. 56. Sine Wave with Phase Shift

HE Help **HE**

Command Type:	Report
Description:	Outputs a list of all commands available
Format:	HE
Argument:	none
Report Format:	text strings

IP Impulse Response **IP**

Command Type:	Move Control
Description:	Perform an impulse move and record the resulting motion in the internal report table. A position data point is recorded for each *n* servo-loop cycles, where *n* is the sampling interval set with the TR command. The width of the impulse is also *n* servo-loop cycles. The resultant motion profile can be read out with the TT command (page 144)
Format:	*axis***IP***amplitude*
Arguments:	*axis*: axis identifier *amplitude*: size of the impulse move, if servo ON, in axis units, if servo OFF, in volts (should not be over about 1/10 of max. voltage)
Response:	none
Troubleshooting:	Axis identifier not valid

KT Kind of Trigger Signal KT

Command Type:	Trigger Configuration
Description:	Adjust polarity and length of trigger pulse.
Format:	*triggerline***KT***triggermode*
Arguments:	*triggerline* specifies one of four output trigger lines.
	triggermode specifies polarity and length of trigger pulses as follows:

0 or +1	trigger active high, short pulse
+2	trigger active high, long pulse
-1	trigger active low, short pulse
-2	trigger active low, long pulse

Output of one curve point lasts approximately 200 microseconds. If "long pulse" is selected, the trigger at a curve point will be activated for 200 microseconds. If "short pulse" is selected then the trigger will be activated for a few microseconds only (much less than 200 microseconds).

Response: None

LN DDL Repeat Number and Response Value LN

Command Type:	DDL (R/W)
Description:	Set/read the DDL initialization phase repeat number (default 35) and DDL initialization phase response value
Format:	*axis***LN**[*number*]
Arguments:	*axis* is the desired axis;
	if *axis*=0, all axes will be affected
	if *number > 0*: *number* is assigned as the DDL initialization phase repeat number of the specified axis.
	if *number < 0 -number* is assigned as the DDL initialization phase response value of the specified axis
Response:	If *number* is omitted, reports two (for *axis* ≠ 0) or more (for *axis* = 0) integer values. The repeat values are all reported first, then the response values.
Troubleshooting:	Axis does not have DDL-enable curve-control property set to ON

Example:

1LN40	sets the DDL initialization phase repeat number to 40 for axis 1.
1LN-10	sets the DDL initialization phase response value to –10 for axis 1.
1LN	reports the values of the DDL initialization phase repeat number and DDL initialization phase response value:
	14
	-010.000
0LN	will send more strings (max. to 6 repeat numbers and 6 response values:
	160

20

20

60

-040.000

-015.000

-015.000

-015.000

LP DDL Initialization Phase Correction LP

Command Type:	DDL (R/W)
Description:	Calculate processing parameters for DDL data, or report current parameters.
Format:	*axis***LP**[*switch*]
Arguments:	*axis* is the desired axis if *axis*=0 do all DDL axes. *switch* = 0: Calculate the working parameters automatically. *switch* =-1: Calculate working parameters automatically and save as power-on defaults. Use this option sparingly, as the EEPROM lifetime write cycles are limited to a few hundred. *switch* > 0: For service personnel only.
Related Argument:	DDL Repeat Number (see **LN** command, p. 120), P-I controller parameters, piezo gain (set at factory).
Response:	if *switch* parameter omitted, reports one (if *axis* ≠ 0) or more (if *axis* = 0) floating point values (delay time in seconds).
Troubleshooting:	no valid axis with the DDL curve control property enabled.

Example:

1LP-1 will cause automatic calculation of DDL parameters and save them as power-on defaults.

1LP will report a string like “5.399997e-3” for axis1.

0LP will send more strings, one for each DDL axis:

5.399997e-3

6.295850e-3

8.686437e-3

3.681464e-3

LS DDL Setup LS

Command Type:	DDL (R/W)
Description:	Enable/disable use of DDL data on an axis, or report the current activation state
Format:	*axis***LS**[*DDL_active_gain*]

Arguments:	*axis* is the desired axis
	if *axis*=0, all axes will be affected
	DDL_active_gain > 0.0 enable DDL use on axis
	DDL_active_gain = 0.0 disable DDL use on axis
Response:	

If *DDL_active_gain* is omitted, one (if *axis* ≠ 0) or more (if *axis* = 0) floating point values will be reported.

Troubleshooting: Axis does not have axis curve control property so defined as to allow enable DDL

Example:

1LS1.0	will use DDL data for the axis 1.
1LS	will report a string like “1.000000e+0” for axis1.
0LS	will report one string for each axis:

```
1.00000e+0
1.00000e+0
0.000000e+0
...
```

LT Define DDL Data Table LT

Command Type:	DDL (R/W)
Description:	Connect axis to a DDL data table.
Format:	*axis***LT**[*DDL_table_num*].
Arguments:	*axis* is the axis to be run with DDL.
	DDL_table_num from 1 to 8.
Response:	If *DDL_table_num* is omitted, reports the number of the DDL table used for the axis
Troubleshooting:	Axis does not have the *DDL enable* curve control property ON.

Example:

1LT1 will connect *DDL table 1* to *axis 1*; *DDL table 1* could have been generated by running *waveform1* on a wave generator connected to axis 1.

The command 1LT4 could later connect DDL table 4 to axis 1; that DDL table could have been generated by running another waveform on axis 1.

MA Move Absolute **MA**

Command Type:	Move Command or Report Command
Description:	The selected axis gets a new target position, or the current target position is reported.
Notes:	The target is the sum of • the position given by MA (i.e. the baseline; see Section "Calculation of Target Positions" on p. 26 more information) • the wave generator output • the analog input value—if used for target generation (see Section "Calculation of Target Positions" on p. 26). When working with the wave generator (MC (p.124) or SC (p. 134)): If the start option "axis will return to the start position" was chosen, any offset (e.g. set with FO) will be permanently added to the baseline until you reset the baseline position using MD (p. 127). With the option "final position after one period is made the new start position for the next period motion" a baseline reset is already included, i.e. it is not necessary to use the MD command. If the new target is within the allowed position limits, the axis moves to it with the programmed velocity to (see also SV command, Set Velocity). If the new target is outside the allowed position limits, the target is set at the limit. If the previously commanded position has not yet been reached, the previous move is aborted and the new move begins immediately. If the current position differs from the baseline position, that difference will be maintained.
Format:	*a*MA[*v*]
Arguments:	*a:* axis identifier, *v:* position value Axis identifier required. If the position value *v* is omitted, the E-710 reports the current target value. The axis identifier must be an integer. Position values can be in integer or floating point representation. The unit is micrometer for linear axes and µrad for rotation axes. The name of the unit is stored in non-volatile memory at address 6.
Response:	none
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) Servo-control OFF or disabled for this axis (new target will not be accepted) Sensor not enabled If servo-control is enabled but turned OFF, issue an SL command to turn it ON

	Position tolerance value too small (smaller than minimal incremental motion or sensor resolution)
	Parameter out of limits
	Incorrect servo parameter, p-term zero or too small (see SP command)
	Velocity too low or zero (see SV command)

MC Output Waveform Continuously MC

Command Type:	Wave Generator Move
Description:	The waveform assigned for the specified generator is output continuously to the connected axis. It is possible to limit the number of cycles using the RN command.
Format:	*generator***MC***switch*
Arguments:	*generator* can be the number of the desired wave generator; if *generator*=0 both generators are affected. D... Set/Read Curve Control
	If *generator* has the value of the generator number plus 100, the generator will start or hold depending on the activation status of the input trigger lines on the digital I/O connector. With generator number plus 200, the generator will be started only once by the first rising edge which is detected.(4-axis versions firmware rev. 5032/6.032 and newer).
	Note: For the kinds of trigger signals currently available on the digital I/O connector see "Digital Trigger Signals" on p. 31). Before a trigger input line can be used it must be enabled by the "D... Set/Read Input Trigger Mode" command (p. 101). If stored in non-volatile memory, it need be done only once.
	The *switch* parameter is used to define the type of motion for axes 1 to 6 (maximum). A group of 4 bits is used for each axis, i.e.:
	Bits 0- 3 for axis 1, bits 4-7 for axis 2, bits 8-11 for axis 3, etc.

Byte								Byte								Byte							
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st
Axis 6				Axis 5				Axis 4				Axis 3				Axis 2				Axis 1			

Axis, bit, byte locations in *switch*

Within each group, the bits have the following definitions:

1st bit: = 0	***Reinitialize DDL* turned OFF**
1st bit: = 1	***Reinitialize DDL* turned ON (see Note 1 below)**
2nd bit: = 0	Axis returns to the waveform start position after every period so that the start position is always the same (start position = baseline position + offset; see the FO (p. 108) and CF (p. 94) commands). Note that there is no baseline reset, i.e. if an offset was set (e.g. with FO), it will be added

	to every subsequent MA command until you reset the baseline with MD (p. 127).
2nd bit: = 1	Final position after one period is made the new start position for the next period motion. This option includes a baseline reset, i.e. it is not necessary to use the MD command before the next MA command.

Bits 3 & 4 determine what position information will be saved during the move:

3rd+4th bits = 0:	Actual positions will be saved (see Note 2 below).
3rd+4th bits = 1:	Relative target positions will be saved (see Note 2 below).
3rd+4th bits = 2:	Actual positions will be saved (see Note 2 below).
3rd+4th bits = 3:	Position errors will be saved (see Note 2 below).

Note 1: The DDL feature must be enabled for the axis (see D... Set/Read DDL license Status, p. 102).

Note 2: The data table number in which the data is saved, and the number of points saved, is as follows:

- ➢ If bits 3 & 4 are 0 for all axes, up to 16384 actual position values will be saved for each of the two axes associated with a wave generator; the data table number used is the same as the wave generator number, not the axis number.
- ➢ If bit 3 or 4 of any axis is not 0, then the data is written to data tables 1 and 2 as follows: data for the axis given by the less significant bit is written to data table 1, and data for the axis given by the more significant bit is written to data table 2. A total of up to 32768 data values can be saved, if two wave generator axes are involved then up to 16384 values per axis.

Note 3: The length of time between data points saved can be set (in units of servo-loops) with the *sampling interval* parameter of the TR command. The default of 1 saves a data point for each servo-loop cycle.

Response: None

Troubleshooting: Invalid axis, or axis does not have curve-control properties defined to permit DDL or waveform motion. See p. 99 for discussion of curve control properties and how to set them.

Example:

Axis 1 and axis 2 are both enabled for waveform move and synchronous move.

Generator 1 has an inverted cosine wave and generator 2 the same waveform but with a 90-degree phase shift (i.e. a sine waveform). Axis 1 is connected to generator 1, axis 2 to generator 2.

1**MC**0	specifies a continuous periodic cosine-waveform motion for axis 1. The maximum number of points for the waveform is 62464.
2**MC**0	specifies continuous periodic sine waveform motion for axis 2. The maximum number of points for the waveform is 62464.
0**MC**0	will induce synchronized motion of both axes, i.e. continuous circular motion. The maximum number of points for the waveform is 31232 per axis.
1**MC**1	starts generator 1 with DDL reinitialization; 32768 points available for DDL data

0**MC**1	starts both generators with DDL reinitialization; 16384 points per axis available for DDL data

As long as *switch* is 0, the following data is saved and can be read out as follows:

1**MC**0	Data table 1 contains the actual position values of the axis which is connected to generator 1. Data can be read using the 1TT command (p. 144)
2**MC**0	Data table 2 contains the actual position values of the axis which is connected to generator 2. Data can be read using the 2TT command.
0**MC**0	Data table 1 contains the actual position values of the axis which is connected to generator 1; can be read using the 1TT command. Data table 2 contains the actual position values of the axis which is connected to generator 2; can be read using the 2TT command.

If *switch* has a value different from 0, this value determines where the data for which axis is saved (see Note 2 above):

1**MC**8	bit 3 (4th bit for axis 1) is set: actual position values of axis 1 are saved; data can be read with 1TT
1**MC**128	bit 7 (4th bit for axis 2) is set: actual position values of axis 2 are saved; data can be read with 1TT
1**MC**136	bit 3 (4th bit for axis 1) and bit 7 (4th bit for axis 2) are set: actual position values of axis 1 and axis 2 are saved; data can be read with 1TT for axis 1 and 2TT for axis 2.
1**MC**2048	bit 11 (4th bit for axis 3) is set: actual position values of axis 3 are saved; data can be read with 1TT
1**MC**2176	bit 7 (4th bit for axis 2) and bit 11 (4th bit for axis 3) are set: actual position values of axis 2 and axis 3 are saved; data can be read with 1TT for axis 2 and 2TT for axis 3

MD Reset Baseline Position MD

Command Type:	Control
Description:	After an **SC**, **MC** or **TL** move or when the wave generator output was stopped with **RT**, the final position is the final position of the waveform plus the baseline position (e.g. the position resulting from a previous **MA** movement). This command resets the baseline position to coincide with the actual position[1] and sets the wave contribution to the target position to zero. It should be run if the waveform move is to be followed by an MA command. It should not be run if the waveform move is followed by other waveform moves referring to the same baseline. Note that the SC and MC commands have options that also affect the baseline setting: There will be no baseline reset if the start option "axis will return to the start position" was chosen. In this case any offset (e.g. set with FO) will be permanently added to the baseline until you reset the baseline position using MD. The option "final position after one period is made the new start position for the next period motion" already includes a baseline reset, i.e. with this option it is not necessary to use MD. See Section "Calculation of Target Positions" on p. 26 for more information about the "baseline" term and the target generation.
Format:	*generator***MD***switch*
Arguments:	*generator* is the selected generator (1, 2). If *generator* = 0 both generators are selected. *switch* = 0 the current position is made the new baseline position. Otherwise, the baseline remains unchanged.
Response:	none.

[1] Note that any offset commanded by the analog input is not included in the actual position which is reported with TP (p. 142).

Examples: The configured waveform start position is 100 (FO) and its final position is 500:

Command Sequence	**Actual Pos.**	**Baseline Pos.**	**Comments**
1MA150	150	150	
1SC0	250	150	SC option: return to start
1MA0	100	0	
whereas:			
1MA150	150	150	
1SC0	250	150	SC option: return to start
1MD0	250	250	
1MA0	0	0	
and:			
1MA150	150	150	
1SC2	550	550	SC option: reset baseline
1MA0	0	0	

MR Move Relative **MR**

Command Type: Move Command

Description: The selected axis gets a new target position. The position given by this command is a relative position.

Notes: The target is the sum of

- the relative position given by MR plus the previous baseline (see Section "Calculation of Target Positions" on p. 26 more information)
- the wave generator output
- the analog input value—if used for target generation (see Section "Calculation of Target Positions" on p. 26).

When working with the wave generator (MC (p.124) or SC (p. 134)):
If the start option "axis will return to the start position" was chosen, any offset (e.g. set with FO) will be permanently added to the baseline until you reset the baseline position using MD (p. 127). With the option "final position after one period is made the new start position for the next period motion" a baseline reset is already included, i.e. it is not necessary to use the MD command.

If the new target is within the allowed position limits, the axis moves to it with the programmed velocity (see also SV command, Set Velocity).

If the new target is outside the allowed position limits, the target is set at the limit. If the previously commanded position has not yet been reached, the previous move is aborted and the new move begins immediately.

Format: *a*MR*v*

Arguments: *a:* axis identifier, *v:* position value
Axis identifier is required. A missing position value is interpreted as zero and the axis will not move. Blanks are

	not allowed. The axis identifier must be an integer. Position values can be in integer or floating point representation. The axis unit depends on its definition. Usually the axis unit is micrometer when the axis is a linear axis and the unit is µrad when the axis is a rotation axis. The name of the unit is stored in the non-volatile parameter memory at address 6.
Response:	none
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6)
	Servo-control OFF or disabled for this axis (new target will not be accepted)
	Sensor not enabled
	If servo-control is OFF, issue an SL command to turn it ON
	Resulting new position out of position limits
	Position tolerance value too small (smaller than minimal incremental motion or sensor resolution)
	Incorrect servo parameter, p-term zero or too small (see SP command)
	Velocity too low or zero (see SV command)
	Baseline not same as real position, see MD command p. 127

MS **M**ove All Together MS

Not available in 3- or 4-axis versions with firmware lower than 5.026 or from 6.00 to 6.025

Command Type:	Move
Description:	Permits moving multiple axes at the same time. Affects all axes which are in servo ON mode. The target for each affected axis must have been specified in a previous SS command (p. 138). If no target was specified with SS, the axis will be moved to an indeterminate position.
Format:	**MS**
Arguments:	none
Response:	none.
Remark	The final settling time is that of the participating axis which takes the longest time to settle.

NP Specify Flatness-Compensation **N**umber of **P**oints NP

(6-axis versions only)

Command Type:	Control
Description:	Specify or report the number of points on the flatness-correction grid in the direction of the specified external axis. This command is used as part of the D... Set/Read Flatness-Compensated Axis, (p. 100), EC (p. 106), NP (page 129), DZ (p. 105) command series

	of the external-axis flatness compensation feature. See p. 44 for complete description and an example.
Format:	[*external_axis*]**NP**[*number*]
Arguments:	*external_axis* = 1 or 2: identifier of external axis for which the number of points (grid divisions + 1) is being specified. *number:* number of grid points in the specified direction. The product of the numbers of points in both directions may not exceed 625.
Response:	if *number* is omitted, the current number-of-points specification in this direction will be reported.

OG Enter Crosstalk Error Factor **OG**

Not available in 3- and 4-axis versions with firmware older than.5.028

Command Type:	Service
Description:	Permits entry of an “axis to correct” and an externally measured crosstalk error value. Requires that the axis causing the crosstalk and the motion range are specified in a previous SO command (p. 138). The axis to correct must have been in open-loop when the external measurement was made.
Format:	*corrected_axis***OG***measured_error*
Arguments:	*corrected_axis* is the axis to benefit from the correction *measured_error* is the externally measured error in the *corrected_axis* direction caused by crosstalk from the axis and range specified in the previous SO command. If *corrected_axis* = 0 no error factors are saved and the function depends on *measured_error* as follows: *measured_error* = 0; the crosstalk error factor will be replaced with the power-up value gain). *measured_error* = 1; the current error factor will be saved to EEPROM becomingthe new power-up default value (there is then no way to recover the previous value!). *measured_error* has the same units as *corrected_axis*
Response:	none
Troubleshooting:	*corrected_axis* not valid
Remark	The cross-talk error correction feature is especially useful for improving dynamic performance. In static operation, any crosstalk occuring inside the servo-loop will be eliminated by the servo-controller whether it is compensated or not.
Example:	the Axis1 is the X-Axis and Axis 6 is the rotational axis ϕ_z-Axis around Z-Axis. The X-Axis range is 100 µm. After the move the error in axis 6 is (rotational error around Z-Axis) is measured at 0.050 mrad. To enter the crosstalk correction, send following commands:

1**SO**100	define a 100 unit move on axis 1
6**OG**0.05	tell the controller that that move caused a 0.50 unit error on axis 6

After checking the results very carefully, they can be made the new power-up default, send:

0**OG**1

after this the old values cannot be restored any more.

PA Curve Start Point **PA**

Command Type:	Wave generation (R/W)
Description:	Set point number in specified segment where curve generation is to begin. See the GL (p. 115), GC (p. 111) and GS (p. 118) commands for details on curve generation.
Format:	*segment***PA**[*startpoint*]
Arguments:	*segment,* number of the segment being defined *startpoint,* must be ≥ 0 and less than the *totalpoints* parameter in the preceding PT (p. 132) command.
Response:	If *startpoint* is omitted, reports the current setting for the segment.

PC Curve Centerpoint **PC**

Command Type:	Wave generation (R/W)
Description:	Specify the centerpoint of the curve to be generated. See the GL (p. 115), GC (p. 111) and GS (p. 118) commands for details on curve generation
Format:	*segment***PC**[*centerpoint*]
Arguments:	*segment,* number of the segment being defined *centerpoint* must be less than the value for *curvepoints* specified in the CP (p. 95) command and at least twice as large as *speedpoints* in the PS command (see p. 132), as *speedpoints* would otherwise be limited by *centerpoint*.
Response:	If *centerpoints* is omitted, reports the current centerpoint setting for the specified segment.

PN Report for Display PaNel **PN**

Command Type:	Service
Description:	Report axis positions, sensor values and PZT voltages of all possible channels/axes.
Format:	PN
Arguments:	none.
Response:	3- and 4-channel versions report 4 axis positions, 4 sensor channel values and 4 PZT channel voltages; 6-channel versions report 8 axis positions, 8 sensor channel values, and 8 PZT channel voltages. Values for items not implemented in hardware should be ignored.
Example:	After sending a PN command 12 floating point numbers will be sent back. Each value has a format like

"-0012.5678 SP LF" except the one in the last line, which has no space before the LF. Some firmware versions may not send the spaces.

PS Points for Speed-Up / Slow-Down **PS**

Command Type:	Wave generation (R/W)
Description:	Specify the number of points in the speed-up / slow-down zones of a single-line or ramp curve to be generated. See the GL (p. 115) and GC (p. 111) commands for details on generation of these curves.
Format:	*segment***PS**[*speedpoints*]
Arguments:	*segment,* number of the segment being defined *speedpoints* is limited by the values specified for *centerpoint* in the PC command and *curvepoints* in the CP (p. 95) command: *speedpoints < centerpoints/2* *speedpoints < (curvepoints - centerpoints)/2*
Response:	If *speedpoints* is omitted, reports the current *speedpoints* value for the segment.

PT Waveform or Segment Points Total **PT**

Command Type:	Wave generation (R/W)
Description:	During segment definition: specify the total number of points in the specified segment. The segments must be defined in ascending numerical order. During waveform definition: specify the total number of points in the specified waveform. Waveforms must be defined in ascending numerical order.
Format:	*number***PT**[*totalpoints*]
Arguments:	*number,* the number of the segment or waveform being defined; if *number*=0, then the process of segment or waveform definition is reinitialized. *totalpoints* is the total number of points in the segment or a waveform. For waveform definitions, segment boundaries must be respected.
Response:	If *totalpoints* is omitted, reports the number of points in the segment or waveform.
Troubleshooting:	The **PT** commands must be issued in a specific order: first 0PT0,1PT*totalpoints*$_1$*,* then generate the curve in segment 1 (see GC (p. 111), GL (p. 115) or GS (p. 118) command), 2PT*totalpoints*$_2$, then generate the curve in segment 2, etc. until all desired segments have been defined, then 0PT0 to switch to waveform definitions, then 1PT*totalpoints*$_1$, 2PT*totalpoints*$_2$*,* ... to define all desired waveforms.

The maximum total number of points that may be defined on 3- and 4-channel versions is 63488, on 6-channel versions 62464.

A maximum of 8 waveforms may be defined.

A maximum number of 128 segments may be defined.

See the wave generation example starting on p. 51

RN Repeat Number of waveform cycles **RN**

Command Type:	Wave Generation (R/W)
Description:	Specifies the number of waveform cycles to be carried out when wave generator output is started by an MC command (see p. 124). The RN command can be sent to the controller before or after the MC command. If it is sent after the MC command, the wave generator starts counting at that point and stops after the specified number of (additional) cycles have been performed. Note that RN applies to both wave generators.
Format:	**RN**[*cycles*]
Arguments:	*cycles* is the number of waveform cycles which are to be performed. If *cycles* is equal to or less than zero, the number of waveform cycles is unlimited.
Response:	If *cycles* is omitted, reports the current *cycles* setting.
Example 1:	The number of waveform cycles is set to 1000 by the RN command. When the generator is started by the MC command, it will run 1000 times and then stop.

RN1000	set number of waveform cycles to 1000
0**MC**0	start wave generators and run both for 1000 cycles

Example 2: Here the RN command is sent after the MC command

0**MC**0	Start wave output
RN50	Begin counting cycles and stop after 50

RP Repeat **RP**

Command Type:	Control Command
Description:	This command allows repetition of all commands in a compound command. RP must be the last command on a compound command line. By including a WAIT in the compound command, fixed-period cyclic events can be executed.
Format:	RP*n*
Arguments:	*n:* number of repetitions of the command line
Troubleshooting:	RP can only be used at the end of a compound command line.

Examples:	Have the controller send the current position of axis one 100 times. Note that both the TP command and the RP command are written on one single line (otherwise the TP would be executed only once): `1TP,RP100` If the position values are to be taken at equal time intervals, then a wait command must be included in the compound command: `1TP,WA10,RP100`

RT Wave generator RunsTop **RT**

Command Type:	Wave Generator Move
Description:	Stop wave generator output Note that if the final position after a waveform move is not identical with the baseline position, the resulting offset will be permanently added to every subsequent MA command until you reset the baseline position using MD (p. 127). See Section "Calculation of Target Positions" on p. 26 for more information.
Format:	*generator***RT**
Arguments:	*generator* = 0, 1 or 2. If *generator* = 0, both generators are stopped.
Response:	none

SC Single Cycle Output **SC**

Command Type:	Wave generator Move
Description:	Output the configured waveform to the connected axis once. The new actual position is then the baseline position plus the final position of waveform.
Format:	*generator***SC***switch*
Arguments:	*generator* can be the number of the desired wave generator; if *generator=0* both generators are affected. *If* the connected axes have curve-control properties set to enable synchronous motion (see the D... Set/Read Curve ControlIf *generator* has the value of the generator number plus 100, the generator will start or hold depending on the activation status of the input trigger lines on the digital I/O connector. With generator number plus 200, the generator will be started only once by the first rising edge which is detected.(4-axis versions firmware rev. 5032/6.032 and newer). Note: For the kinds of trigger signals currently available on the digital I/O connector see “Digital Trigger Signals” on p. 29). Before a trigger input line can be used it must be enabled by the “D... Set/Read Input Trigger Mode”

command (p. 100). If stored in non-volatile memory, it need be done only once.

The switch parameter is used to define the type of motion for axes 1 to 6 (maximum). A group of 4 bits is used for each axis, i.e.:

Bits 0- 3 for axis 1, bits 4-7 for axis 2, bits 8-11 for axis 3, etc.

Byte								Byte								Byte							
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st	4th	3rd	2nd	1st
Axis 6				Axis 5				Axis 4				Axis 3				Axis 2				Axis 1			

Axis, bit, byte locations in *switch*

Within each group, the bits have the following definitions:

1st bit: = 0	***Reinitialize DDL* turned OFF**
1st bit: = 1	***Reinitialize DDL* turned ON (see Note 1 below)**
2nd bit: = 0	Axis returns to the waveform start position after every period so that the start position is always the same (start position = baseline position + offset; see the FO (p. 108) and CF (p. 93) commands). Note that there is no baseline reset, i.e. if an offset was set (e.g. with FO), it will be added to every subsequent MA command until you reset the baseline with MD (p. 126).
2nd bit: = 1	Final position after one period is made the new start position for the next period motion. This option includes a baseline reset, i.e. it is not necessary to use the MD command before the next MA command.

Bits 3 & 4 determine what position information will be saved during the move:

3rd+4th bits = 0:	Actual positions will be saved (see Note 2 below).
3rd+4th bits = 1:	Relative target positions will be saved (see Note 2 below).
3rd+4th bits = 2:	Actual positions will be saved (see Note 2 below).
3rd+4th bits = 3:	Position errors will be saved (see Note 2 below).

Note 1: The DDL feature must be enabled for the axis (see D... Set/Read DDL license Status, p. 101).

Note 2: The data table number in which the data is saved, and the number of points saved, is as follows:

- If bits 3 & 4 are 0 for all axes, up to 16384 actual position values will be saved for each of the two axes associated with a wave generator; the data table number used is the same as the wave generator number, not the axis number.
- If bit 3 or 4 of any axis is not 0, then the data is written to data tables 1 and 2 as follows: data for the axis given by the less significant bit is written to data table 1, and data for the axis given by the more significant bit is written to data table 2. A total of up to 32768 data values can be saved, if two wave generator axes are involved then up to 16384 values per axis.

Note 3: The length of time between data points saved can be set (in units of servo-loops) with the *sampling interval* parameter of the TR command. The default of 1 saves a data point for each servo-loop cycle.

Response:	None
Troubleshooting:	Invalid axis, or axis does not have curve-control properties defined to permit DDL or waveform motion. See p. 99 for discussion of curve control properties and how to set them.

Examples:

Axis1 and axis 2 are both enabled for waveform move and synchronous move.

Generator 1 has an inverted cosine wave and generator 2 the same waveform but with a 90-degree phase shift (i.e. a sine waveform). Axis1 is connected to generator 1, axis 2 to generator 2.

1**SC**0	specifies a cosine-waveform motion for axis 1
2**SC**0	specifies sine waveform motion for axis 2
0**SC**0	will induce synchronized motion of both axes, i.e. a single complete circle
1**SC**1	starts generator 1 with DDL reinitialization (1 cycle for axis 1). This command must be repeated until the required number of cycles (parameter 141: DDL repeat number or RN command) is reached or the position error is reduced to the desired value. As long as the DDL initialization is not finished, do not execute one of the following commands because they would cause an initialization restart: SF, CF, LT, LS, GS, GC, GL, CL.

SF Select a waveform for a generator **SF**

Command Type:	Wave Generation (R/W)
Description:	Select a waveform for specified wave generator.
Format:	*generator***SF**[*waveform*]
Arguments:	*generator* = 1 or 2
	waveform = 1 to 8.
Response:	If *waveform* is omitted, reports the current waveform number selected for the generator.

Example:

1SF1	select the waveform No. 1 for the generator 1
2SF2	select the waveform No. 2 for the generator 2
1SF3	change the waveform to No. 3 for the generator 1
2SF4	change the waveform to No. 4 for the generator 2
1SF	answer is: 3
2SF	answer is: 4

SL Servo-Loop ON/OFF **SL**

Command Type:	Control Command / Report Command
Description:	This command turns the servo-controller for the specified axis ON or OFF. For servo-control to actually be active, servo-control must also be enabled for the respective axis. Each axis can be configured so that the servo-control is enabled and turned ON automatically on power up. If servo autostart is ON it is not necessary to send the SL command in order to turn servo-control ON. As factory default, all axes are set to servo-OFF on power up. The power-up defaults can be changed by changing non-volatile memory (see p. 17).
Format:	*a*SL*p*
Arguments:	*a:* axis identifier (integer, range 1-6) *p:* value parameter, can be 0 or 1. value = 0 set servo OFF, value = 1 set servo ON If the parameter is omitted, the E-710 reports the current state.
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) Servo-control OFF or disabled for this axis Sensor not enabled

SM Proportional Move **SM**

Command Type:	Move command
Description:	Move all axes which have non-zero proportional gain to new position. Operates in the same way as MA (p. 123), except that all axes are affected and the axis parameter “proportional gain” is multiplied in when the target for the axis is calculated. Note that, as with MA, if the current position of any axis differs from the baseline position, that difference will be retained.
Format:	SM*position*
Arguments:	*position* is the amplitude of proportional move. The distance an axis is moved is *position* times the *proportional-gain* axis-property value.
Response:	none.

Example 1:

If the starting positions of all 4 axes in a 4-axis system are 0, then after the following commands:

1DP0,62DW1	Set proportional gain of axis 1 to 1
2DP0,62DW2	Set proportional gain of axis 2 to 2
3DP0,62DW0	Set proportional gain of axis 3 to 0
4DP0,62DW1	Set proportional gain of axis 4 to 1

SM100	Perform proportional move of 100

the positions of the axes will be:

axis1:	100
axis2:	200
axis3:	0
axis4:	100

SO **S**etup for Cr**o**sstalk Compensation SO

6-axis versions; 4-axis version: firmware rev.5.028 or newer

Command Type:	Service (R/W)
Description:	Permits entry of an axis and a displacement amount to be used by a subsequent OG command (OG, p. 130). Use of this command pair requires that you make the specified move and measure the unwanted motion in the other axis (crosstalk) with an external measuring device. That error will then be entered as a parameter in the OG command.
Format:	*crosstalk_axis***SO**[*range*]
Arguments:	*crosstalk_axis* is the axis whose motion over *range* causes the crosstalk whose amount and axis will be entered in the OG command. *range* is the displacement used when measuring the crosstalk. The axis to benefit from the correction must be in open-loop during the measurement; *range* has the same units as *crosstalk_axis*
Response:	none

SS **S**et Target for Move All Together SS

Not available on 3- or 4-axis versions with firmware lower than 5.026 or between 6.00 and 6.025

Command Type:	Move Control (R/W)
Description:	Specify an axis and a target position for subsequent MS commands (p. 129).
Format:	*axis***SS**[*ms_target*]
Arguments:	*axis* is the axis to which target applies *ms_target* is the position to move the axis to when the next MS command is issued.
Response:	If *ms_target* is omitted, the current *ms_target* for *axis* is reported. If *axis* = 0, the *ms_target* of all axes are reported.

SP Set p-Term SP

Command Type:	Control Command
Description:	This command sets the proportional servo-control term to be used for the specified axis. The proportional term can be set within the range of 0 to a maximum value defined as 1.5

	times of the factory-default value. The new p-value is valid until it is overwritten by another SP command or the E-710 is shut down.
	If the new SP value is within the allowed range, it becomes effective immediately, otherwise the factory-set default value is used instead of the specified value.
	The power-up -default value is stored in non-volatile memory. This value was optimized by factory for a default mass. The user should optimize this value for his or her application.
Format:	*a*SP*p*
Arguments:	*a:* axis identifier (integer, range 1-6) *p:* p-term of digital servo-controller If the p-term value is omitted, the p-term is set to zero (see troubleshooting). Blanks are not allowed. p-term values can be in integer or floating point representation, but must be positive.
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) When servo-controller disabled or OFF for this axis or sensor not enabled, this value has no effect.
	If the p-term parameter is very low or zero, then the stage can not by moved with MA (p. 123), MR (128) or GH (p. 112) commands.
	If the p-term parameter is out of the allowed range from 0 to 1.5 x default, then the default value is used.
	If the servo-control loop becomes unstable, try a smaller SP value or switch power to the E-710 device off and on again to restore the power-on default.

ST Step Response **ST**

Command Type:	Move
Description:	Make a step move and fill the internal report table. A position data point is recorded for each *n* servo-loop cycles, where *n* is the sampling interval set with the TR command. The start position is the current position and the step amplitude is defined with command parameter. 8192 data points are recorded. The resultant motion profile can be read out with the TT command (page 144).
Format:	*axis***ST***amplitude*
Arguments:	*axis*: axis indentifier
	amplitude: size of step to use, if servo ON, in axis units, if servo OFF, in volts (should not over about 1/10 of max. voltage).
Response:	none
Troubleshooting:	Axis identifier not valid:
	out of allowed range (1-6),

	servo-control disabled or OFF for this axis or sensor disabled *amplitude* too small (smaller than resolution) Incorrect servo-control parameters, e.g. p-term zero or too small. Velocity too small.

SV Set Velocity **SV**

Command Type:	Configuration Command
Description:	This command sets a new velocity for motion of the specified axis. The unit of velocity is axis_unit/ms. The velocity setting is applied in servo-mode only. The new velocity setting takes effect immediately if the commanded value does not exceed the velocity limit. The maximum allowed velocity (velocity limit) is stored in non-volatile memory and is not modified by the SV command. As factory default, the maximum velocity is used.
Format:	*a*SV*p*
Arguments:	*a:* axis identifier *p:* velocity parameter In practice, both parameters are required, because if the velocity value is omitted, the velocity is set to zero (see troubleshooting). Blanks are not allowed. The axis identifier must be an integer, velocity values can be in integer or floating point representation, but must be positive. Units of velocity are µm/ms or µrad/ms.
Troubleshooting:	Axis identifier not valid or out of allowed range (1-6) Servo-control disabled or turned OFF for this axis If servo-control is OFF, issue an SL command (see p. 137) to turn it ON. Velocity parameter out of allowed range, negative or beyond upper limits. The axis can not be moved by the MA or MR command when the velocity is too low or zero. The minimum allowable velocity depends on the absolute value of the current position: Velocity [axis_unit / ms] > |Current position [axis_unit]| / 10^6

TA Tell ADC Value **TA**

Command Type:	Report
Description:	This command returns the most-recently read value from the A/D converter. It is an unsigned 16-bit value representing the digitized sensor value without filtering, linearization or transformation.
Format:	*a*TA
Arguments:	*a:* sensor-channel identifier The sensor-channel identifier is required and must be an integer.
Response:	Digitized sensor value as an integer in the range of 0-65535
Troubleshooting:	Channel identifier not valid: out of allowed range (1-7) Sensor, cable or sensor module failure or improper configuration Missing or invalid synchronization signal on external trigger

TL Test curve moving/DDL Report **TL**

Command Type:	Waveform Move
Description:	Make Curve Moving Report. This command starts data recording. Data for all axes is recorded simultaneously; the data can later be read with the TT command. With this command the motion of the axis associated with a wave generator can recorded in a report table for upload to the host computer (see the TT command description). The length of time between recorded data points (in servo-loop cycles) is determined by the sampling interval set with the TR (table rate) command (see p. 143). The maximum number of points per axis is 8192 for 3- and 4-axis units, 4096 for 6-axis units.
Format:	*generator***TL***switch*
Arguments:	*generator* is the desired (running) generator. *switch* = 0 record no data, *switch* = 1 record target data *switch* = 2 record actual position *switch* = 3 record position-error data *switch* = 4 record DDL data (DDL required) *switch* = 5 record output from servo-control loop (see Fig. 6).
Response:	Number of wave generator cycles completed since the last start

The report table can be filled with either original data or filtered data.

See the D... Set/Read Filter Enable for Curve Moving Report command, p. 101, and the D... Set/Read Filter Bandwidth for Curve Moving Report command, p. 100 for controlling the filter.

TN Tell Normalized Sensor Value **TN**

Command Type:	Service
Description:	Report the normalized sensor value. This value is used internally as the input for the sensor position linearization section (see diagram, p. 23).
Format:	*c*TN
Arguments:	*c:* sensor channel (or a comparable analog channel).
Response:	report normalized sensor value in a format like: 0000.0001.
Troubleshooting:	Invalid sensor channel
Example:	3TN response = -0087.3248 (no unit) For comparison, read sensor channel number 3, raw value. command: 3TS response: -0003.0004 (unit is µm or µrad)

TP Tell Position **TP**

Command Type:	Report
Description:	This command reports the current position of the specified axis. The reported value for the axis is calculated from the last sensor reading(s), processed by the digital filter and corrected in accordance with the offset, gain and linearization settings. The sensor-channel-to-axis coordinate transformation matrix applied to the sensor value(s) should mirror the cross-talk behavior of the mechanics. Due to mechanical interaction, the coordinate transformation applied may induce more or less crosstalk than appropriate. The mechanical system must therefore be taken into account when interpreting the reported value. Note that when using the analog input for target generation of an axis, the TP command no longer gives the actual position of that axis, but the actual position minus the Analog Target. In this case, calculate the actual position as follows: Axis *n* Actual Position = *n*TP Report Value + *analog_input*TS Report Value See Section "Using the Analog Input" on p. 36 for more information.
Format:	*a*TP
Arguments:	*a:* axis identifier The axis identifier is required and must be an integer.
Response:	Position value in fixed decimal format: +xxx.xxxx, units are µm or µrad.
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) Servo-control OFF or disabled for this axis

	Sensor not enabled Incorrect sensor parameters. See block structure (p. 22) and GI command (p. 113).

TR Table Rate **TR**

Command Type:	Control (R/W)
Description:	Sets or reads the number of servo-loop cycles to be used by certain other commands. Settings other than 1 make it possible to cover longer time periods with a limited number of points. This command affects data recording (see TT, p. 144) and wave generator.
Format:	[*option*]**TR**[*sampling_interval*]
Arguments:	*sampling_ interval*: an integer as new update time factor in servo-loop cycles; if omitted current setting is given as response *option*: (only 3-and 4-axis versions with firmware equal or newer than 5.040 / 6.040 and 6-axis versions with firmware equal or newer than 2.17) if *option* = 0 or omitted (and *sampling_interval* > 1), then interpolated target values are inserted in the target value output of the wave generator to avoid jumps of the mechanics and to keep the DDL working correctly. if *option* = 1, no interpolation is done. *option* values different from 1 or 0 are ignored, and the last valid setting remains active. On controller power-up, the interpolation is activated (if *sampling_interval* > 1).
Response:	If *sampling_ interval* is omitted, current setting in number of servo-loop cycles

TS Tell Sensor Position **TS**

Command Type:	Report
Description:	This command returns the current position of the specified sensor channel, in µm.
Format:	*channel*TP
Arguments:	*channel:* channel number Sensor channel number (or corresponding analog input channel)
Response:	Position value in fixed decimal format: +xxx.xxxx, units are µm.
Troubleshooting:	Channel number out of allowed range (1-7 Sensor parameters set incorrectly. See block structure (p. 22) and GI3 command (p. 113).

TT Tell report Table **TT**

Command Type:	Report
Description:	Report a value from the report table. There are several commands which can be used to fill the report table: ST (step response), IP (impulse move), MC and SC (Single and multiple waveform moves) with position saving ON, TL (Test curve moving/DDL Report).
Format:	*a***TT[***s*]
Arguments:	*a:* is the related report axis. If axis = 0, then currently selected report function axis is used. There are two types of responses, depending on the value of the switch, *s*, used as an index to the table. *s* = 0 or omitted: report data value at the current index counter and increment the index counter automatically. *s* > 0: Report data at index *s* and set the current index counter to *s* +1. Commands such as ST, IP, SC and TL will reset the index counter.
Response:	Data value.
Troubleshooting:	axis identifier not valid. Data counter overflow.
Example 1:	Read 800 data items, from index 1 to 800 for axis 1. 1TT, RP800 response: 800 values
Example 2:	Read 800 data items from index 200 to 1000 for axis 1. 1TT200 response: 1 value 1TT,RP799response: 799 values

TV Tell Velocity **TV**

Command Type:	Report
Description:	Reports the current velocity setting for the specified axis in µm or µrad per ms. This setting can be changed with the SV command. The axis unit depends on its definition. Usually the axis unit is micrometer when the axis is a linear axis and µrad when it is a rotation axis. The name of the unit is stored in the non-volatile memory at address 6.
Format:	*a*TV
Argument:	*a:* axis identifier (integer) The axis identifier is required.
Report Format:	Exponential format: sign + 7-digit mantissa + 2-digit exponent
Troubleshooting:	Axis identifier not valid: out of allowed range (1-6) When servo-control disabled for this axis the value has no effect.

UP Upload Controller **UP**

Command Type:	Service
Description:	Report parameters stored in the specified bank of internal non-volatile memory.
Format:	*c***UP***n*
Arguments:	*c:* Memory bank (usually axis/channel number) in non-volatile memory *n:* number of parameters to report
Response:	Report containing the specified number of parameters with channel/axis and location (memory address) prefix.
Troubleshooting:	invalid memory bank (channel/axis) or *n* too large
Example:	to read parameters stored in locations 1 to 10 for channel 3

Command: 3UP10.

The report will resemble the following[*]:

```
A3L1= 1 SP  LF
A3L2= 1 SP  LF
A3L3= 0 SP  LF
A3L4= 0 SP  LF
A3L5= 1818324545 SP  LF
A3L6= 86 SP  LF
A3L7= 100 SP  LF
A3L8= 20 SP  LF
A3L9= -2.400000e+1 SP  LF
A3L10= 1.000000e-2 LF
```

where “A” and “L” are literals contained in the response

VR Voltage Relative **VR**

Command Type:	Move
Description:	The VR command allows use of individual axes in servo-OFF mode.
Format:	*a*VR*n*
Argument:	*a:* Axis identifier *n:* voltage value

Details:

The voltage value parameter represents a change in voltage that is distributed by a control matrix to one or more PZTs resulting in the desired motion relative to the

[*] Some firmware versions may not send the space characters SP as they should; As a result some versions of host or application software may hold back response lines and show them as responses to later commands. Try sending blank commands as a work-around.

current position without position-control feedback. The command also makes it possible to find defects or incorrect settings.

In this operating mode, ramps and slopes are not limited, so mechanical noise may be generated by the stages.

Troubleshooting:	Axis identifier not valid
	Servo-control enabled and servo ON for this axis
	Resultant PZT voltage above limits. The voltage limits are stored in the parameter memory at the addresses 110 (low limit) and 111 (high limit).
	Depending on the control matrix, another axis may contribute to the output voltage of the output channels.
	Required PZT output-channel voltage exceeds amplifier range.

VS Voltage Set **VS**

Command Type:	Move
Description:	Allows use of individual axes in servo-OFF mode.
Format:	*a*VS*n*
Argument:	a: Axis identifier *n:* voltage value

Details:

The specified voltage is applied to the PZT channel which is most closely coupled with the specified axis. The other affected axes, if any, see voltage contributions in the proportions specified by the matrix, resulting in (open-loop) motion in the direction of the specified axis. This command can also be used to find defects or improper settings.

In this operating mode, velocities and slew rates are not limited, so mechanical noise may be generated by the stages.

Troubleshooting:	Axis identifier not valid
	Servo-control ON and enabled for this axis
	The voltage value is not within the allowable limits. The voltage limits are stored in the parameter memory at the addresses 110 (low limit) and 111 (high limit).
	Depending on the transformation matrix, the specified axis may contribute to the output voltage to more than one PZT output channel.

VT Voltage Tell **VT**

Command Type:	Report
Description:	This command reports the output voltages of up to 8 PZT output channels.
Format:	VT
Argument:	none
Report Format:	8 output voltages are reported:

```
PZT 1  +110.0000
PZT 2  -018.0000
PZT 3  +000.0000
PZT 4  +011.1111
PZT 5  +087.0500
PZT 6  -098.0960
PZT 7  +000.0000
PZT 8  +000.0000
```

Details:

The reported output voltages may not correspond with the voltages set by the VS command, because the PZT transformation matrix may distribute the voltages set there to different output channels. The VT command can be used whether the axes are in Servo-ON or Servo-OFF modes. In Servo-ON mode, it can be used to check the relation between the working range and the voltage range.

WA Wait **WA**

Command Type:	Control Command
Description:	This command is used within a compound command to add a delay time between two commands. In conjunction with RP (see p. 133), fixed-period cyclic events can be generated.
Format:	WA*n*
Argument:	*n*: delay time in ms, integer from 1 to 100000.

17 Technical Data

For the availability of PIO interface, Analog Input and Digital Signal Transmission (DST) see the table in Section 1.4.1 on p. 9.

Model	**E-710.3CD / E-710.A3D / E-710.APD / E-710.APS / E-710.APS0**	**E-710.4CD / E-710.4CL / E-710.P3D / E-710.P4D / E-710.P4L / E-710.C3D / E-710.C4D**	**E-710.6CD / E-710.6SD / E-710.6SD0**
Function	Digital NanoAutomation® PZT controller and power amplifier	Digital NanoAutomation® PZT controller and power amplifier	Digital NanoAutomation® PZT controller and power amplifier
Sensor channels	3 cap. sensors	4 cap. sensors	6 cap. sensors
Logical axes	3	4	6
PZT channels	4	4	8
Processor	DSP 32-bit floating point, 33 MHz	DSP 32-bit floating point, 33 MHz	2 x DSP 32-bit floating point, 33 MHz
Sampling interval	50 µs (sensor), 200 µs (servo-loop, 3 axes)	50 µs (sensor), 200 µs (servo loop, 4 axes)	40 µs (sensor), 200 µs (servo-loop, 6 axes)
Effective Resolution DAC	20-bit	20-bit	20-bit
Maximum output power	25 W / channel	25 W / channel	25 W / channel
Average output power	6 W / channel	6 W / channel	6 W / channel
Peak output current < 20 ms	200 mA / channel	200 mA / channel	200 mA / channel
Average output current > 20 ms	50 mA / channel	50 mA / channel	50 mA / channel
Current limitation	Short-circuit proof	Short-circuit proof	Short-circuit proof
Output voltage	-20 to +110 V	-20 to +110 V	-20 to +110 V
PZT voltage output sockets	Sub-D special	Sub-D, special (models E-710.xxD only), LEMO ERN.00.250.CTL (E-710.xxL only)	Sub-D special
Sensor sockets	Sub-D, special	Sub-D, special (models E-710.xxD only), LEMO PSA.00.250.CTAC22 (E-710.xxL only)	Sub-D, special
Interfaces	RS-232 and IEEE 488 (GPIB)	RS-232 and IEEE 488 (GPIB)	RS-232 and IEEE 488 (GPIB)
Dimensions	450 x 88 x 390 mm Controller 200 x 40 x 180 mm Sensor Interface Box (with E-710.APS and E-710.APS0 only)	450 x 88 x 390 mm	450 x 88 x 390 mm Controller 214 x 89 x 260 mm Sensor Interface Box (with E-710.6SD and E-710.6SD0 only)
Weight	7 kg Controller 0.485 kg Sensor Interface Box (with E-710.APS and E-710.APS0 only)	7 kg	7 kg Controller 2.35 kg Sensor Interface Box (with E-710.6SD and E-710.6SD0 only)
Power consumption (max)	60 W	60 W	60 W
Operating voltage and line fuses	115 or 230 VAC, 50-60 Hz	115 or 230 VAC, 50-60 Hz	115 or 230 VAC, 50-60 Hz

17.1 Line Power and Fuses

The power connection, input voltage selection and line fuses are located on the rear panel. Unless you request a specific setting, the unit will be set up for the line voltage we believe predominant in your country.

New Fuses Required when Changing Supply Voltage

To connect the product to a different supply voltage, it is necessary not only to reorient the fuse carrier so that the desired voltage is visible in the window, but also to replace both the line fuses with the value appropriate for the new voltage. Lower supply voltage requires higher current.

To access the fuse carrier, remove the power cord, and pry open the door that covers the carrier. The carrier can then be pried out, exposing the two line fuses (see illustration at right).

After replacing the fuses, reinstall the carrier in the new orientation; check that the correct voltage setting is visible through the window in the door when it is reclosed.

Always use fuses with a rating suitable for your line voltage:

230 V 1 A*

115 V 2 A*

* slow blow fuses required

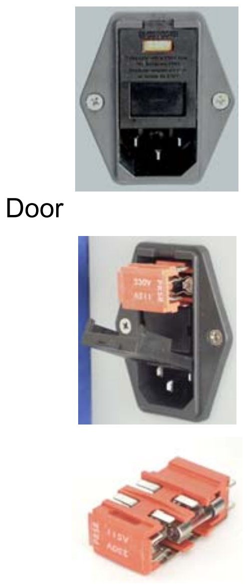


Fig. 57. Fuse location (1 of 2 fuses visible) and line voltage setting

17.2 3- and 4-Axis Model-Specific Hardware Details

The following sections show the front panels of the various models with their connector types. The rear panels are described on page 154.

17.2.1 E-710.4CL and E-710.P4L Front Panel

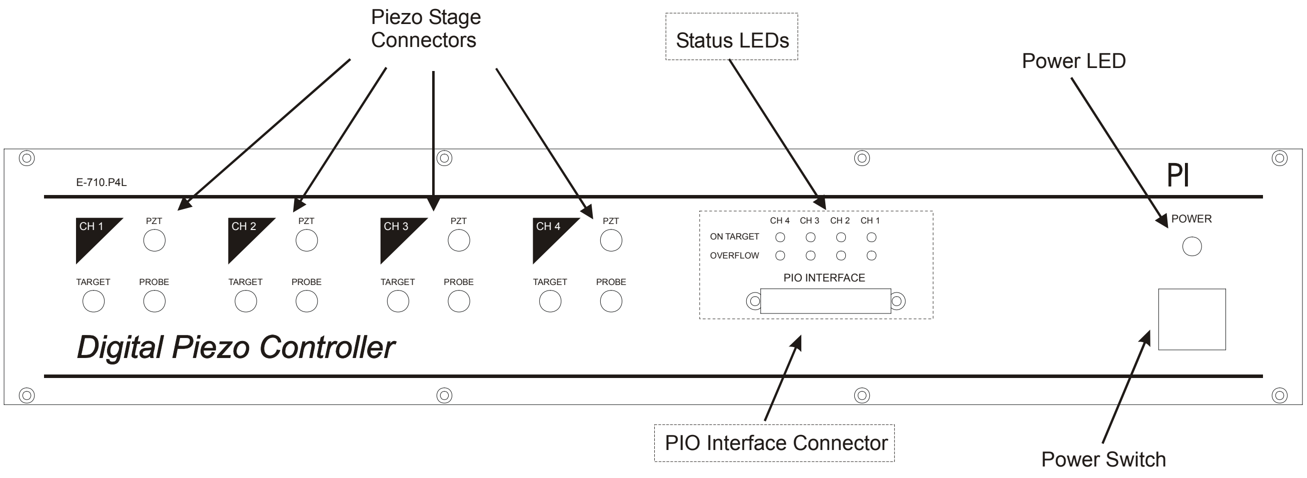


Fig. 58: E-710.4CL and E-710.P4L front panel. The two models differ in the absence/presence of the elements in dashed boxes (PIO interface and associated status LEDs).

Front panel connectors*	For connection to	Details and pinouts, page
1 LEMO connector per channel with PZT drive voltage and ground.	PZT Drives	155
2 LEMO connectors per channel: TARGET (Sensor Target) and shield PROBE (Sensor Probe) and shield	Sensor	155
PIO Interface connector (E-710.P4L only)	PIO command source	158

*Note: Stages with sub-D connectors can be connected using the following sub-D to LEMO adapters: E-710.3L0, 3 channels, 0.3 m and/or E-710.1L0, 1 channel, 0.3 m. Use of ID-chip functionality may be impaired by adapter cable.

For rear panel layout and connectors, see p. 154.

17.2.2 E-710.4CD and E-710.P4D Front Panel

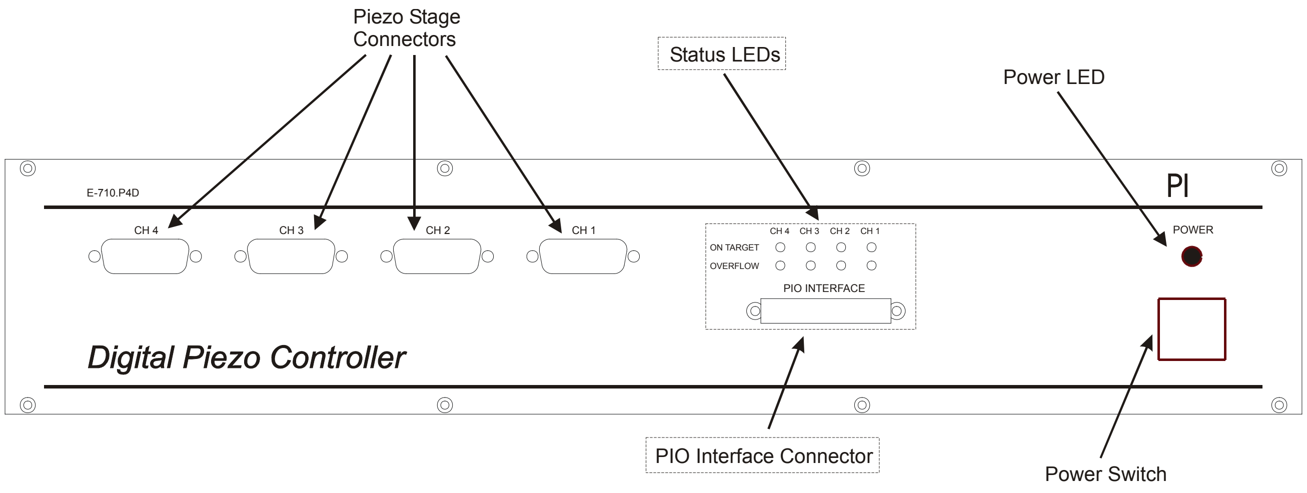


Fig. 59: E-710.4CD and E-710.P4D front panel. The two models differ in the absence/presence of the elements in dashed boxes (PIO interface and associated status LEDs).

Front panel connectors	For connection to	Details and pinouts, page
4 sub-D special connectors with 2 coax lines and 5 pins, each for: 1 x PZT drive voltage and ground 1 x TARGET (Sensor Target) and shield 1 x PROBE (Sensor Probe) and shield	PZT Stage (PZT Drive and Sensor)	156
PIO Interface connector (E-710P4D only)	PIO command source	158

For rear panel layout and connectors, see p. 154.

17.2.3 E-710.P3D and E-710.C4D Front Panel

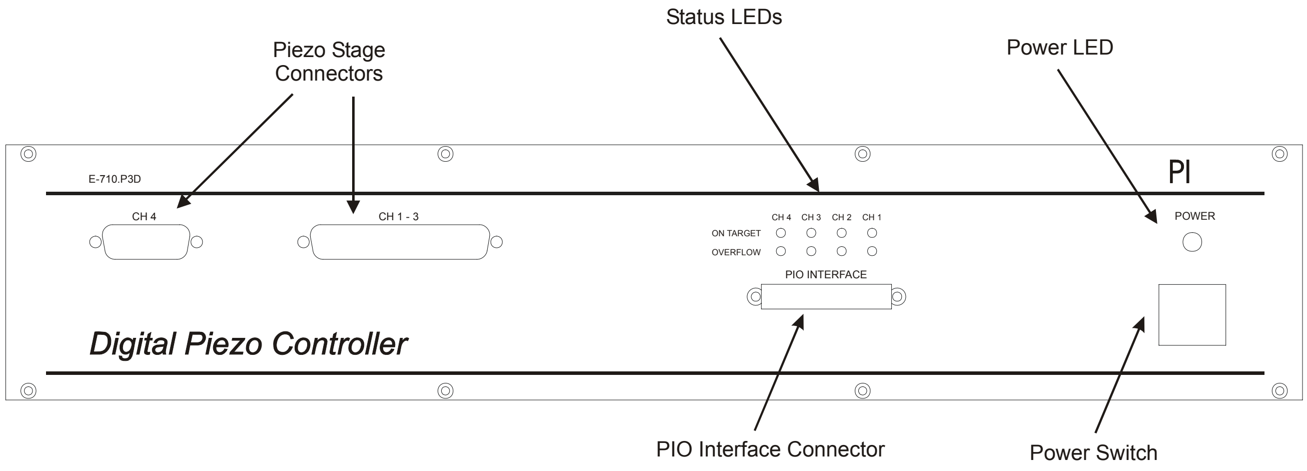


Fig. 60: E-710.P3D (4-axis) front panel, E-710.C4D front panel is identical but has no status LEDs and no PIO Interface Connector

Front panel connectors	For connection to	Details and pinouts, page
1 Sub-D special connector with 3 coax lines and 22 pins for: 3 x PZT drive voltage and ground 3 x TARGET (Sensor Target) and shield 3 x PROBE (Sensor Probe) and shield	PZT Stage (PZT Drive and Sensor)	156
1 Sub-D special connector with 2 coax lines and 5 pins for: 1 x PZT drive voltage and ground 1 x TARGET (Sensor Target) and shield 1 x PROBE (Sensor Probe) and shield	PZT Stage (PZT Drive and Sensor)	156
PIO Interface connector, only with E-710.P3D models, not on E-710.C4D	PIO command source	158

For rear panel layout and connectors, see page 154.

17.2.4 E-710.3CD, E-710.A3D and E-710.APD Front Panel

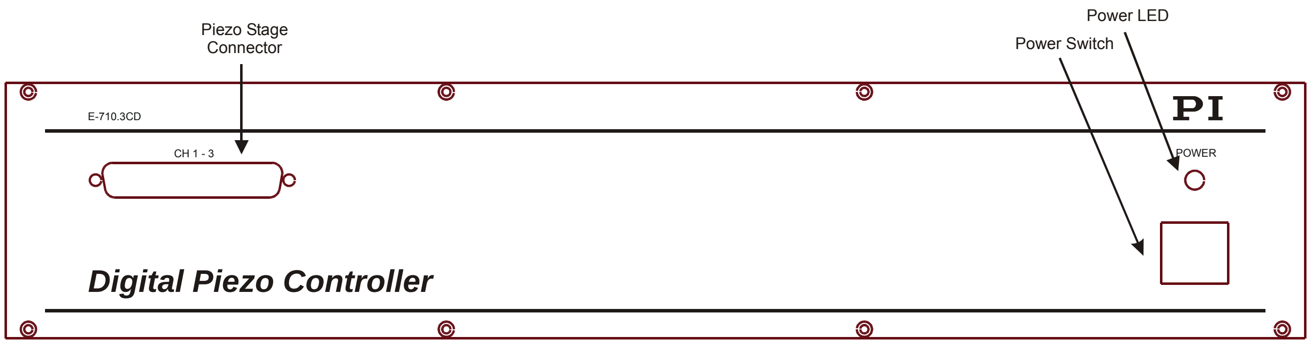


Fig. 61: E-710.3CD and E-710.A3D Front Panel, E-710.APD front panel is identical but has in addition the status LEDs and the PIO Interface Connector which are shown in Fig. 58, Fig. 59 and Fig. 60

Front panel connectors	For connection to	Details and pinouts, page
1 Sub-D special connector with 3 coax lines and 22 pins for: 3 x PZT drive voltage and ground 3 x TARGET (Sensor Target) and shield 3 x PROBE (Sensor Probe) and shield	PZT Stage (PZT drives and sensors)	156
PIO Interface connector, only with E-710.APD models, not on E-710.3CD and E-710.A3D	PIO command source	158

For rear panel layout and connectors, see p. 154 and p. 155.

17.2.5 E-710.APS and E-710.APS0 Components

E-710.APS and E-710.APS0 comprise a digital piezo controller and a separate interface box for digital sensor-signal transmission for applications where the distance between the mechanics and electronics is greater than 3 m. The connection between controller and sensor interface box is done via the connecting cable E-710.APSx (x encodes the cable length; see Fig. 64 on p. 154).

Controller:

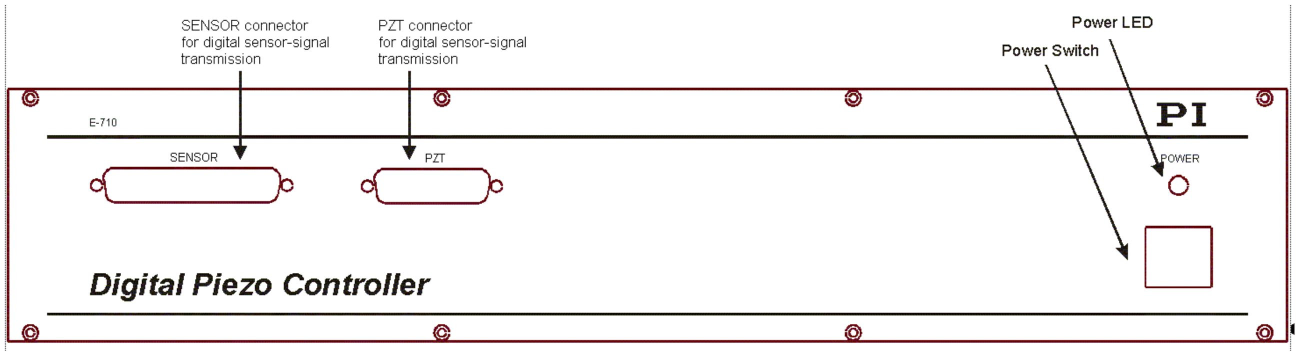

SENSOR connector for digital sensor-signal transmission
PZT connector for digital sensor-signal transmission
Power LED
Power Switch
E-710
PI
SENSOR
PZT
POWER
Digital Piezo Controller

Fig. 62: E-710.APS and E-710.APS0 controller front panel

Front panel connectors	For connection to	Details and pinouts, page
PZT socket and SENSOR socket	For the connection between controller and sensor interface box.	14
PIO Interface connector, only with E-710.APD models, not on E-710.3CD and E-710.A3D	PIO command source	158

For controller rear panel layout and connectors, see p. 155.

Sensor Interface Box:

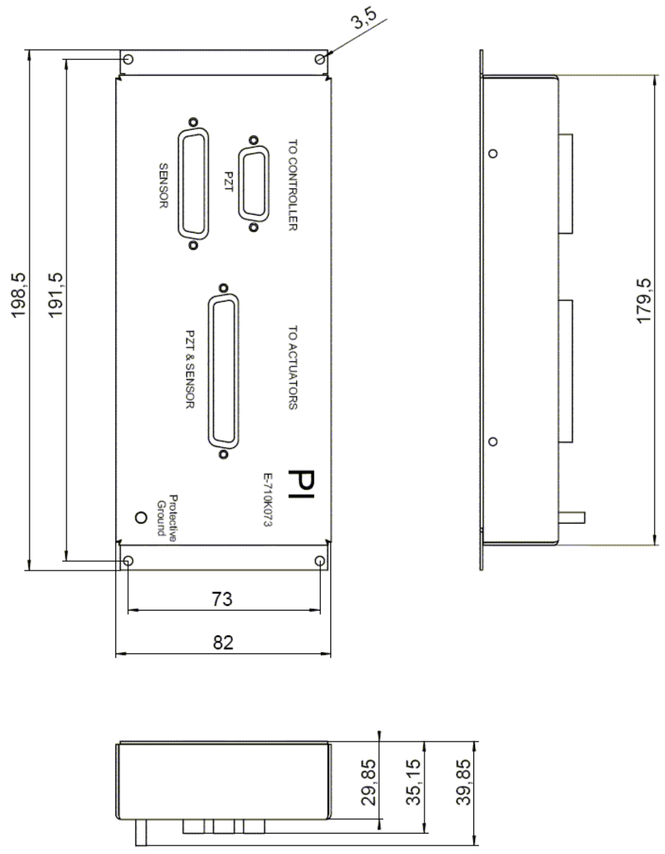


Fig. 63: Sensor interface box labeling and dimensions in mm; decimal places separated by commas

Front panel connectors	For connection to	Details and pinouts, page
PZT socket and SENSOR socket	For the connection between controller and sensor interface box.	14
"PZT & SENSOR" Sub-D special connector with 3 coax lines and 22 pins for: 3 x PZT drive voltage and ground 3 x TARGET (Sensor Target) and shield 3 x PROBE (Sensor Probe) and shield	PZT Stage (PZT drives and sensors)	156

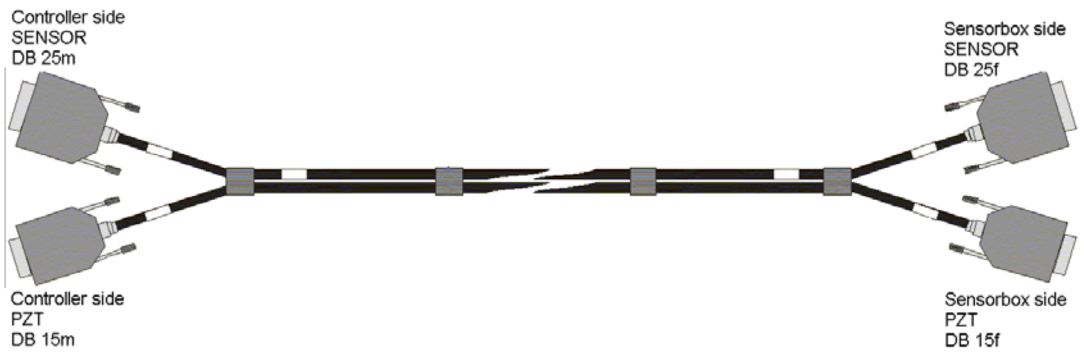


Fig. 64: Connecting cable for controller and sensor interface box; order# E-710.APSx

17.2.6 Rear Panel of 3- and 4-Axis Models without Analog Input

There is no difference between the rear panels of the various 3- and 4-axis versions without Analog Input.

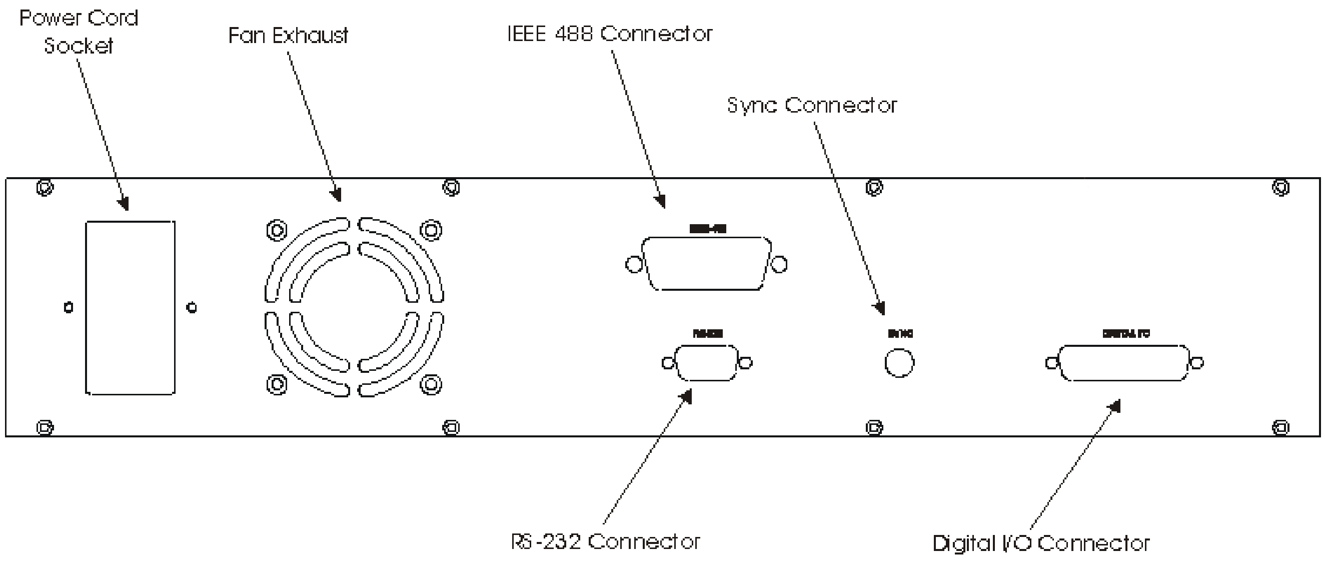


Fig. 65: E-710 Rear Panel of the 3- and 4-axis Model Types without Analog Input

Rear panel connectors	For connection to	Details and pinouts, page
Digital I/O Connector	External unit	157
IEEE 488 Connector	Host computer	30, 164
RS-232 Connector	Host computer	30, 33, 165
Sync Connector	Used for synchronizing multiple E-710s	32, 164

17.2.7 Rear Panel of 3- and 4-Axis Models with Analog Input

There is no difference between the rear panels of the various 3-axis versions with Analog Input.

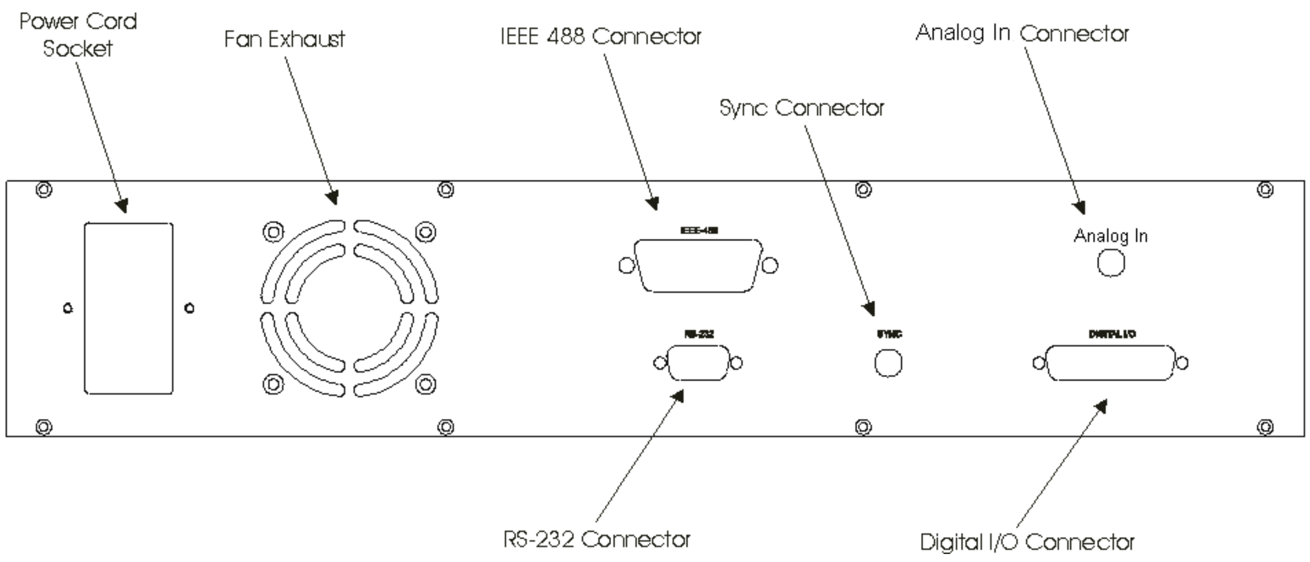


Fig. 66: E-710 Rear Panel of the 3-axis Model Types with Analog Input

Rear panel connectors	For connection to	Details and pinouts, page
Digital I/O Connector	External unit	157
IEEE 488 Connector	Host computer	30, 164
RS-232 Connector	Host computer	30, 33, 165
Sync Connector	Used for synchronizing multiple E-710s	32, 164
Analog In Lemo 4pin connector	Analog input (e.g. for an user defined external sensor)	36, 163

17.2.8 LEMO Connectors, 4-Axis Models

With the 4-axis models E-710.4CL and E-710.P4L LEMO connectors are used for both sensor and PZT drive connections.

The sensor connection is over two cables per sensor—one for the target and one for the probe, each with its own shield.

The drive connection has one cable per PZT. The drive cables/connectors carry voltages of up to 130 V. The positive voltage is on the inner conductor.

17.2.9 Sub-D Special Connectors, 3- and 4-axis models

The two different kinds of sub-D special connectors which are used to connect piezo stages to the controller are illustrated below:

Sub-D Mix Connector with 3 coax lines and 22 single pins:

Pin		Signal
Coax inner lines		
A1		Probe Y-1
A2		Probe Y-2
A3		Probe X
Standard pins		
1		Target Y-1 inner line
	12	Target Y-1 shield
2		Target Y-2 inner line
	13	Target Y-2 shield
3		GND
	15	Not used
4		ID CH1
	16	GND
5		GND
	17	ID CH3
6		ID CH2
7		Piezo Y-2
8		Piezo Y-1
	19	GND Piezo Y1 + Y2
9		Piezo X-2
10		Piezo X-1
	21	GND Piezo X1 + X2
11		Target X inner line
	22	Target X shield

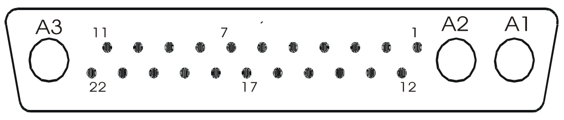


Fig. 67: Sub-D Mix Connector with 3 coax lines and 22 single pins

Note:

Probe and *Target* are capacitive sensor connections.

X, Y-1, Y-2 are channel designations.

Piezo lines carry PZT drive voltages of up to 130 V.

Sub-D Mix Connector with 2 coax lines and 5 single pins:

Pin		Signal
Coax inner lines:		
A1		PZTOUT (LV)
A2		Probe
Standard pins		
3		PZTGND
	1	DOW
4		+ 15 V Supply for external devices
	2	AGND Target and ID ground
5		Target

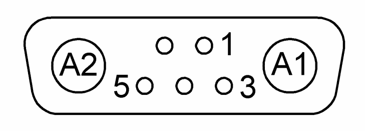


Fig. 68: Sub-D Mix Connector with 2 coax lines and 5 single pins

Note:

Probe and *Target* are capacitive sensor connections.

PZTOUT line carries PZT drive voltage of up to 130 V.

17.2.10 Digital I/O Connector, 3- and 4-axis models

The DB25 digital I/O connector (rear panel, see fig. 62 for location) carries the following signals:

2:	Wave generator start, trigger input
15:	Wave generator hold and start, trigger input (Firmware Rev. 5.030/6.030 and newer)
1, 3, 5, 7 , 9, 11, 13:	Ground
8:	Command cycle trigger
10:	Wave generator, trigger output 3
20:	Sensor cycle trigger
21:	Wave generator, trigger output 1
22:	Wave generator, trigger output 2
23:	Wave generator, trigger output 4

The voltage level of all lines is TTL (maximum input level: +5 V). The input line has an internal pullup resistor of 20 kΩ. That means that the open-circuit input level is +5 V.

Maximum output current (in both high and low states): 8 mA

17.2.11 PIO Connector Pinout, 3- and 4-Axis Models Only

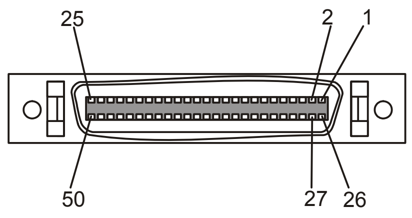


Fig. 69: PIO Interface Connector

Pin	Signal	Pin	Signal
1	ON_TARGET4 channel 4	26	GND
2	GND	27	D10 data bit 10
3	ON_TARGET3 channel 3	28	GND
4	Reserved	29	D9 data bit 9
5	ON_TARGET2 channel 2	30	GND
6	RW read/write	31	D8 data bit 8
7	ON_TARGET1 channel 1	32	GND
8	GND	33	D7 Data bit 7
9	A1 address (= channel -1) bit 1	34	GND
10	GND	35	D6 data bit 6
11	A0 address (= channel -1) bit 0	36	GND
12	GND	37	D5 data bit 5
13	STRB strobe	38	GND
14	GND	39	D4 data bit 4
15	LD latch new data (on all axes simultaneously)	40	GND
16	GND	41	D3 data bit 3
17	D15 data 15	42	GND
18	GND	43	D2 data bit 2
19	D14 data bit 14	44	GND
20	GND	45	D1 data bit 1
21	D13 data bit 13	46	GND
22	GND	47	D0 data bit 0
23	D12 data bit 12	48	GND
24	GND	49	A2 address (= channel -1) bit 2
25	D11 data bit 11	50	GND

PIO Electrical Specification	Limit	Min	Max
High input voltage		2.0 V	5 V
Low input voltage		0 V	0.8 V
High output voltage	Output current 4.0 mA	2.4 V	
Low output voltage	Output current 12.0 mA		0.4 V
Input current	Output voltage 5 V		0.25 mA
Input current	Output voltage 0 V		0.5 mA
Output current total			100 mA
Input rise time			250 ns
Output rise time (200 pF capacitor)			50 ns
Output rise time (no load connected)			20 ns
Isolation voltage		250 V	

17.3 6-Axis Model-Specific Hardware Details

17.3.1 E-710 6-Axis Models Front Panel

The only item of interest on the front panel of all 6-axis versions is the POWER LED (see Section 2.4 on page 15 for the LEDs behavior during power up).

17.3.2 E-710.6CD Rear Panel

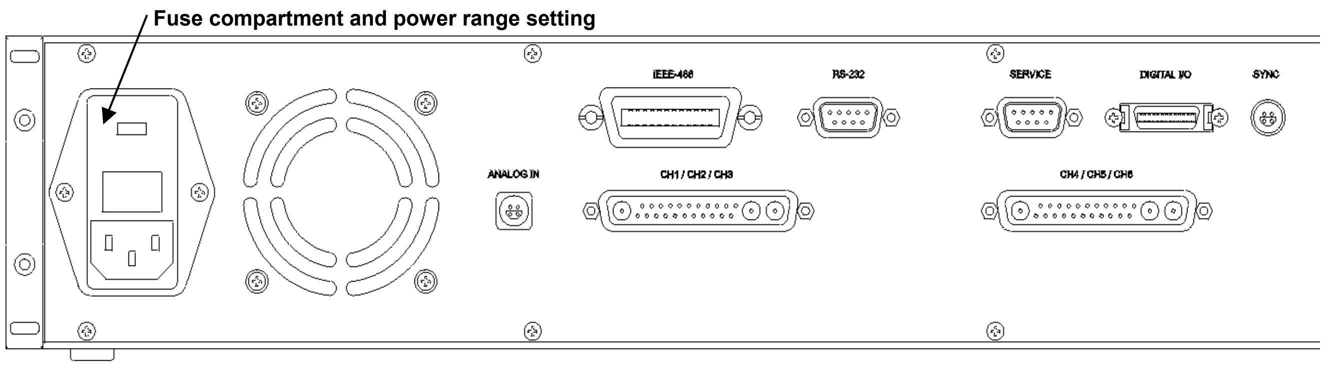


Fig. 70: E-710.6CD rear panel

Rear panel connectors	For connection to	Details and pinouts, page
2 Sub-D special connectors each with 3 coax lines and 22 pins for*: 4 x PZT drive voltage and ground 3 x TARGET (Sensor Target) and shield 3 x PROBE (Sensor Probe) and shield	PZT Stages (PZT drives and sensors)	162
ANALOG IN Lemo 4pin connector	Analog input (e.g. for an user defined external sensor)	163
Digital I/O Connector	External unit	163
IEEE 488 Connector	Host computer	30, 164
RS-232 Connector	Host computer	30, 33, 165
Sync Connector	Used for synchronizing multiple E-710s	32, 164
Service Connector**	Serial interface for service tasks	

*Connector CH1/CH2/CH3 is used for connecting mechanics (e.g. stage). The mechanics can have at most 4 PZTs and 3 capacitive sensors. Internally the PZTs are defined as PZT1 – PZT4; sensors are defined as Sensor1 – Sensor3. Connector CH4/CH5/CH6 is the same type as Connector CH1/CH2/CH3 but the PZTs are defined internally as PZT4 – PZT8; sensors as Sensor4 – Sensor6.

**For service tasks. The hardware structure of the 6-axis controller comprises two boards. Firmware updates for the first board are done over the normal RS-232 interface and for the second board over the Service interface.

17.3.3 E-710.6SD and E-710.6SD0 Components

E-710.6SD and E-710.6SD0 comprise a digital piezo controller and a separate interface box for digital sensor-signal transmission for applications where the distance between the mechanics and electronics is greater than 3 m. The connection between controller and sensor interface box is done via the connecting cable E-710.DSTx (x encodes the cable length; see Fig. 74 on p. 162).

Controller:

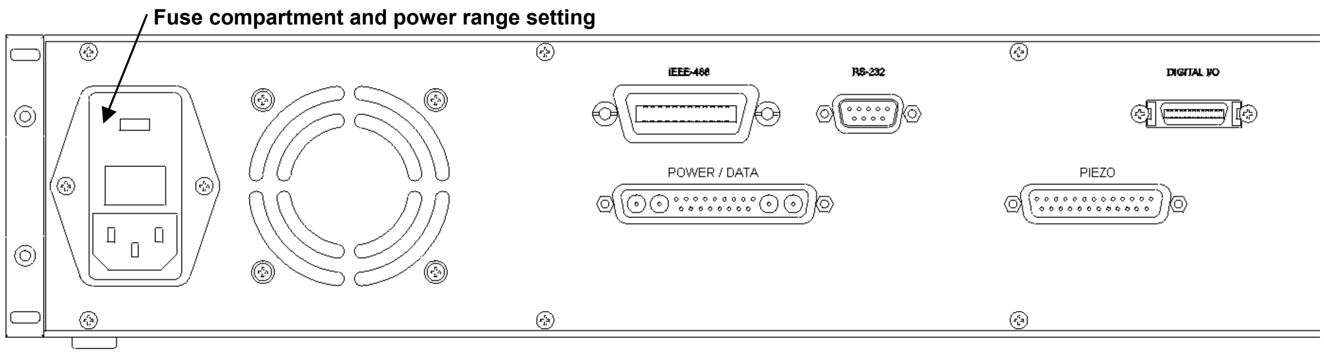


Fig. 71: E-710.6SD and E-710.6SD0 controller rear panel

Rear panel connectors	For connection to	Details and pinouts, page
IEEE 488 Connector	Host computer	30, 164
RS-232 Connector	Host computer	30, 33, 165
Digital I/O Connector	External unit	163
PIEZO socket and POWER / DATA socket	For the connection between controller and sensor interface box.	14

Sensor Interface Box:

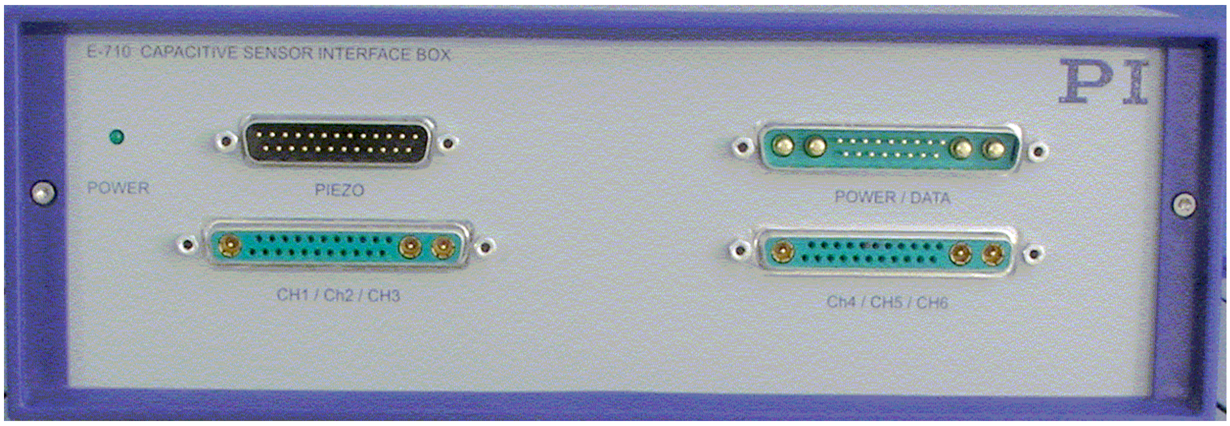


Fig. 72: Sensor interface box front panel

Front panel connectors	For connection to	Details and pinouts, page
"Ch1/Ch2/CH3" and "CH4/CH5/CH6" Sub-D special connectors each with 3 coax lines and 22 pins for*: 4 x PZT drive voltage and ground 3 x TARGET (Sensor Target) and shield 3 x PROBE (Sensor Probe) and shield	PZT Stages (PZT drives and sensors)	162
PIEZO socket and POWER / DATA socket	For the connection between controller and sensor interface box.	14

*Connector CH1/CH2/CH3 is used for connecting mechanics (e.g. stage). The mechanics can have at most 4 PZTs and 3 capacitive sensors. Internally the PZTs are defined as PZT1 – PZT4; sensors are defined as Sensor1 – Sensor3. Connector CH4/CH5/CH6 is the same type as Connector CH1/CH2/CH3 but the PZTs are defined internally as PZT4 – PZT8; sensors as Sensor4 – Sensor6.

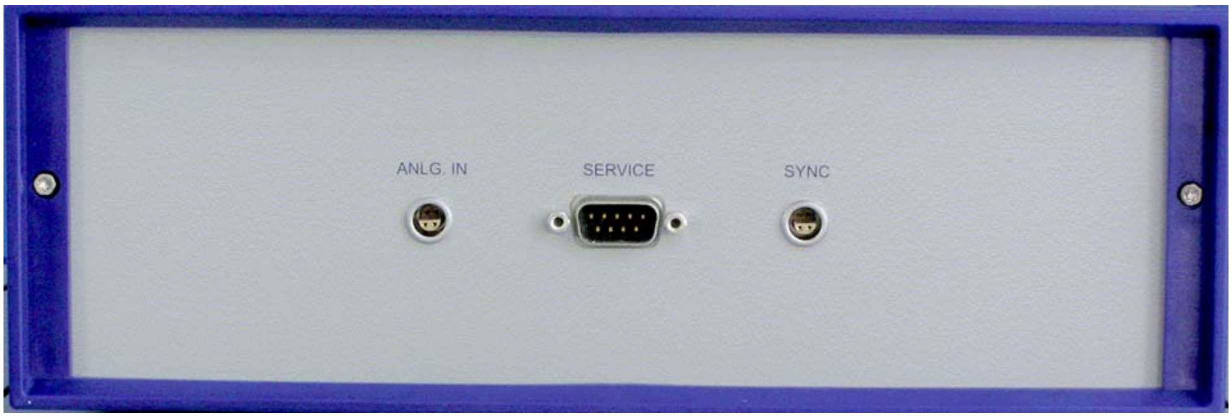


Fig. 73: Sensor interface box rear panel

Rear panel connectors	For connection to	Details and pinouts, page
ANLG. IN Lemo 4pin connector	Analog input (e.g. for an user defined external sensor)	36, 163
Service Connector*	Serial interface for service tasks	
Sync Connector	Used for synchronizing multiple E-710s	32, 164

*For service tasks. The hardware structure of the 6-axis controller comprises two boards with firmware updates for the first board being done over the normal RS-232 interface and for the second board over the Service interface.

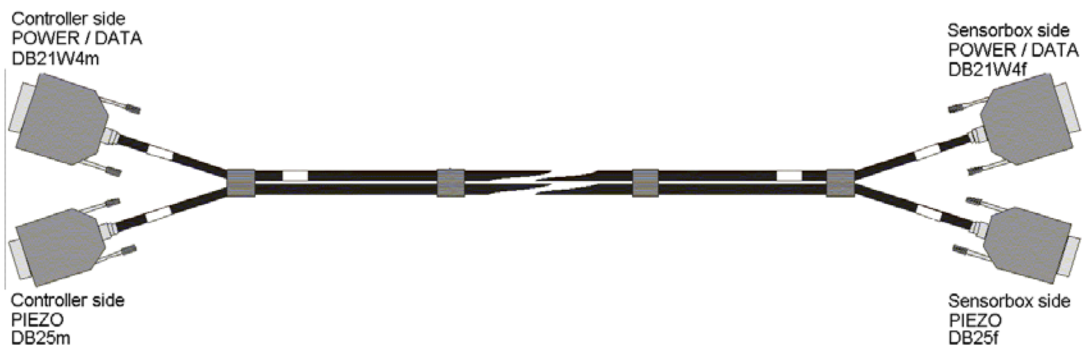


Fig. 74: Connecting cable for controller and sensor interface box; order# E-710.DST4x

17.3.4 Sub-D Special Connector, 6-axis Models

Sub-D Mix Connector with 3 coax lines and 22 single pins:

The following table shows the pin assignment for both the CH1/CH2/CH3 connector and the CH4/CH5/CH6 connector.

Pin Number	Function
A1	Sensor Probe CH2 resp. CH5
A2	Sensor Probe CH3 resp. CH6
A3	Sensor Probe CH1 resp. CH4
1	Sensor Target CH2 resp. CH5
2	Sensor Target CH3 resp. CH6
3	ID-CHIP GND CH1 resp. CH4
4	ID-CHIP CH1 resp. CH4
5	ID-CHIP GND CH2 resp. CH5
6	ID-CHIP CH2 resp. CH5
7	PZT4 resp. PZT8
8	PZT3 resp. PZT7
9	PZT2 resp. PZT6
10	PZT1 resp. PZT5
11	Sensor Target CH1 resp. CH4
12	Sensor Target Shield CH2 resp. CH5
13	Sensor Target Shield CH3 resp. CH6
14	NC
15	NC
16	ID-CHIP GND CH3 resp. CH6
17	ID-CHIP CH3 resp. CH6
18	PZT GND PZT4 resp. PZT8
19	PZT GND PZT3 resp. PZT7
20	PZT GND PZT2 resp. PZT6
21	PZT GND PZT1 resp. PZT5
22	Sensor Target Shield CH1 resp. CH4

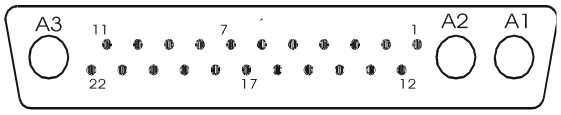


Fig. 75: Sub-D Mix Connector with 3 coax lines and 22 single pins

Note:

Probe and *Target* are capacitive sensor connections.

PZT lines carry PZT drive voltages of up to 130 V.

17.3.5 Digital I/O Connector, 6-Axis Models

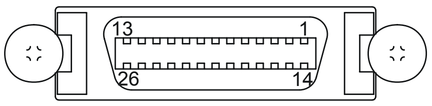
Standard: TTL

Maximum output current:

High level (>2.4V): max. 3.4 mA

Low Level (<0.5V): max 16 mA

The MDR 26-conductor digital I/O connector (see fig. 62 for its location on the rear panel) carries the following signals:

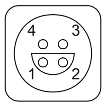


Pin Number	Function
1	GND
2	D_IN0
3	D_IN1, Wave Generator Trigger Input, Generator Start
4	D_IN2, Wave Generator Trigger Input, Generator Hold and Start (Firmware Rev. 2.11 and higher)
5	GND
6	D_IN3
7	D_IN4
8	D_IN5
9	GND
10	D_IN6
11	D_IN7
12	NC
13	GND

Pin Number	Function
14	D_OUT0, Default usage is Servo Trigger
15	D_OUT1, Default usage is Command Trigger
16	D_OUT2, Default usage is Trigger 1
17	GND
18	D_OUT3, Default usage is Trigger 2
19	D_OUT4, Default usage is Trigger 3
20	D_OUT5, Default usage is Trigger 4
21	GND
22	D_OUT6
23	D_OUT7
24	NC
25	GND
26	NC

17.4 ANALOG IN Connector

Using the ANALOG IN connector (LEMO FFA.OS) analog input, such as a target offset or a preprocessed signal from an external position sensor or a focus detector, can be transferred to the controller. See p. 32 for details. Be sure to use an appropriate GND for single-line signals.



Pin Number	**Function**
1	Analog Input, positive (keep within 10 V of GND, ~1 M-ohm input impedance vis-a-vis GND)
2	NC
3	NC
4	Analog Input, negative (keep within 10 V of GND, ~5 k-ohm input impedance vis-a-vis GND)

17.5 SYNC Connector and Synchronization Cable

Using their SYNC connectors (LEMO FFA.OS), you can synchronize two E-710s via a special cable which can be obtained from PI, order# E71000207. See p. 32 for synchronization details.

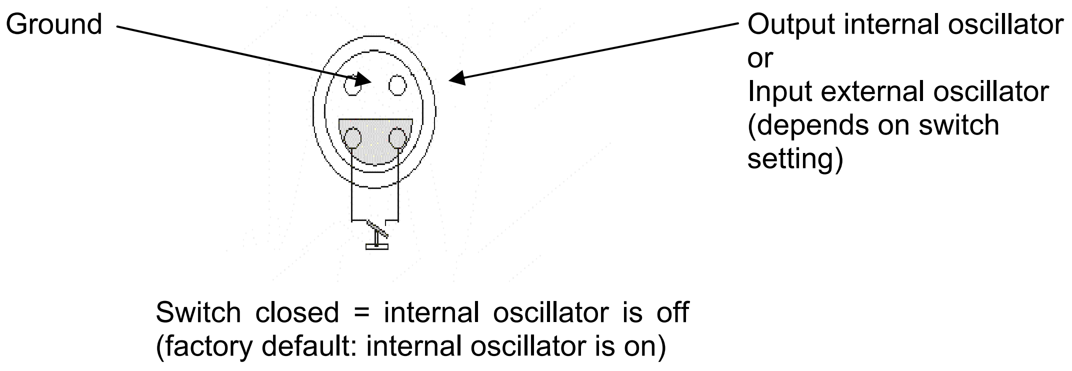

Ground
Output internal oscillator or Input external oscillator (depends on switch setting)
Switch closed = internal oscillator is off (factory default: internal oscillator is on)

Fig. 76: Sync connector on E-710, front view

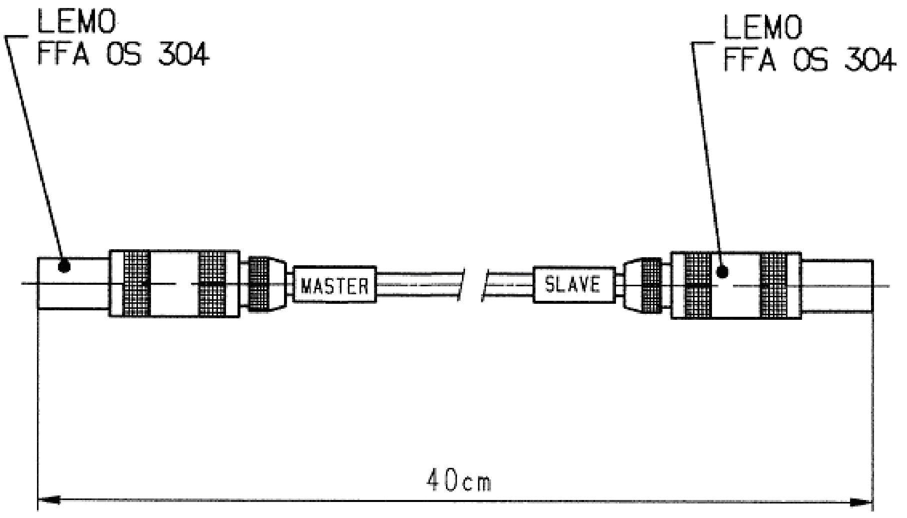


Fig. 77: Synchronization cable E71000207

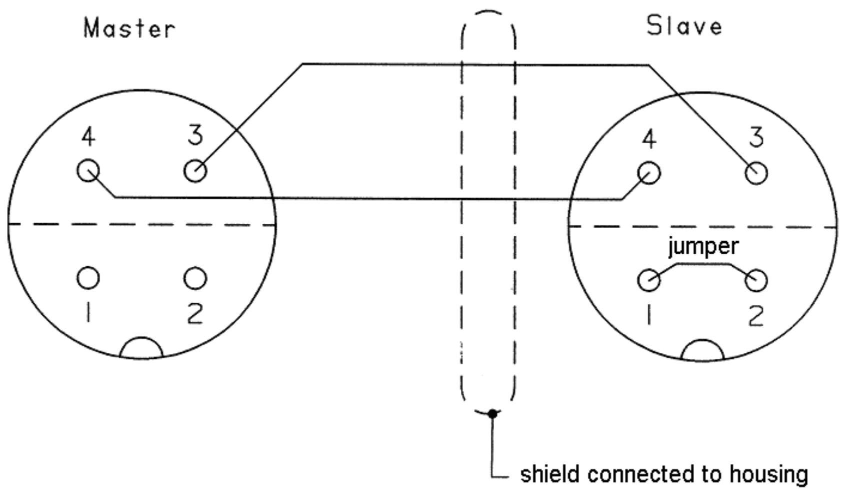


Fig. 78: Internal connections in E71000207 synchronization cable, connectors shown from soldering side

17.6 IEEE (GPIB) Interface

The E-710 IEEE interface (see Fig. 62 for its location on the rear panel of 3- and 4-axis versions, Fig. 66 for 6-axis version) is a slave and conformant with the industry standard (use shielded cable). The factory default address setting is 4. See Section 5.2.1 on p. 30 for how to change the power-on default. If more than one E-710 is to be connected to the same IEEE bus, the addresses of at least all but one must be changed.

Currently only bit 4 (hex: 0x10) of the *Serial Poll Response Byte* is used. According to IEEE 488.2, bit number 4 is the *Message Available Bit* (MAV). It is set to 1 by the controller when it has data to send.

17.7 RS-232 Interface

The E-710 RS-232 interface (see Fig. 62 for its location on the rear panel of 3- and 4-axis versions, Fig. 66 for 6-axis version) conforms to the 9-pin, sub-D industry standard implementation (use the C-815.34 cable that comes with the controller). RTS/CTS hardware handshaking is used, as the settings are 8 bits, 1 stop bit and no parity. The factory default baud rate is 9600 baud. See Section 5.1 on p. 30 for how to change the power-on default. The 9600 baud setting can be obtained at any time by a special power-up sequence, as described on p. 33.

18 PIO Interface

The variety of commands that can be issued over the PIO is limited (see p. 84 for command descriptions), but they offer a considerable speed advantage. It may seem that the PIO is much too fast for the E-710 controller. This would be true if the transmission link were to be used continuously. In nearly every application, however, the host computer has lot of other tasks in addition to positioning the piezo stage. It is therefore important to reduce data transmission time to a minimum, and this is what the PIO can do.

PIO interface commands and responses are sent to the E-710 as digital voltage levels on the various interface lines. The sections below are for those wishing to design of custom hardware. The PIO connector pinout is described in the previous section.

18.1 Guidelines for PC Cards Communicating with the E-710 PIO

If the application requires that position data be both read and written, it is important to realize that the levels on the address and control lines must not change when the direction of the position data lines is reversed. This fact must be taken into account when choosing the connecting hardware (usually a digital-interface PC card). The card must allow configuration of the address and control lines independent of the data lines.

Some cards, such as the PC DIO-24 from National Instruments with 24 lines, would seem to be sufficient (if one does without the on-target signals). However, the lines cannot be reconfigured independently: changing a 8-bit group between input and output produces pulses that can lead to errors in the E-710. The reason is that all the lines are controlled by a single 82C55A chip. The National Instruments PC-DIO-24 is therefore not suitable for bidirectional operation of the E-710 PIO.

The E-710 PIO has been tested with the National Instruments PCI-DIO-96 card, which has four 82C55A chips (actually, two would be enough). The demo program was written for this card. The 16 position data lines must be on one chip and the other lines (control, address and on-target) on one or more other chips.

If, however, the application involves sending position data in one direction only (i.e. read only or write only—unidirectional operation) so that the data direction of the lines never needs to be changed during operation, then a PC-DIO-24 card could be used with the restrictions implied.

18.1.1 Number of Data and Control Lines:

Signal levels: TTL, max. 5V

Outputs (from E-710)	4 data lines (on-target signals)
Inputs (to E-710)	3 address lines + 3 control lines
Bidirectional lines	16 position data lines

18.1.2 Setup and Programming Notes

The signal levels on the control lines must remain constant and stable during the period when the data direction of the bidirectional lines is reversed. This condition may not be met with PC cards using chips like the 82C55A if the control and data lines are on the same chip. Even though the three 8-bit blocks are separately configurable during operation, reversing the data direction of one block causes momentary disturbance on the other lines.

18.2 PIO Line-Timing Considerations

The information in this section is primarily for users wishing to build custom hardware to drive the PIO interface, or who need to troubleshoot hardware problems.

18.2.1 PIO Write Operation Timing

All timing sequence values are given for a cable length of about 0.5 m. For longer cables, some headroom over specified minimum times is recommended.

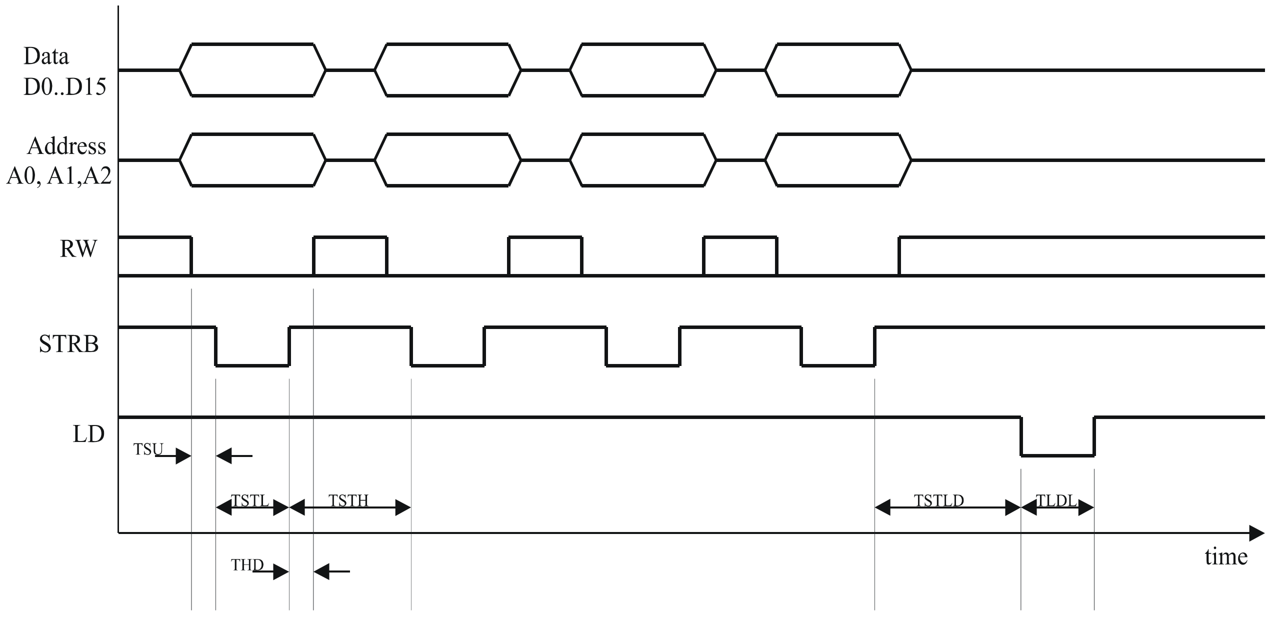


Fig. 79: Data Write Timing

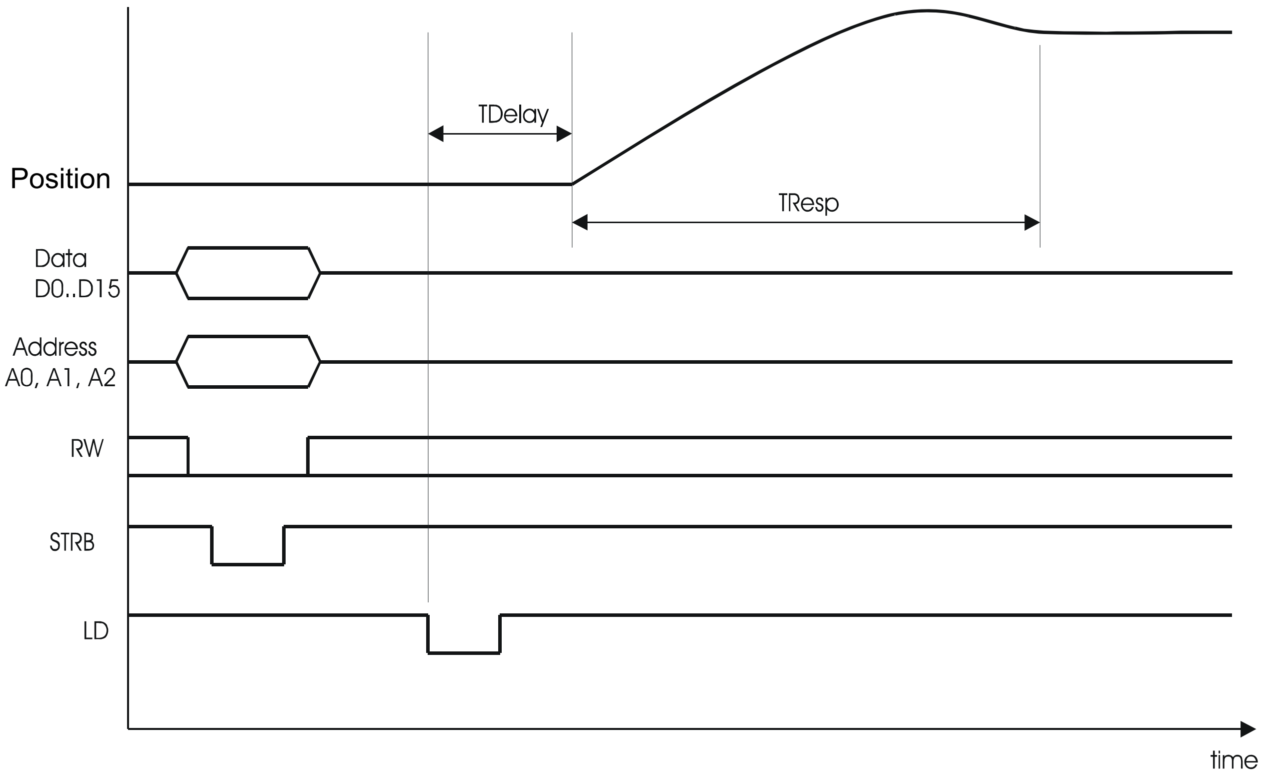


Fig. 80: PIO Data Write and Settling of the Piezo Stage Position

PIO Timing Definitions:

Parameter	Description	Min	Max
TSTL	STRB active (low)	400 ns	
TSU	Setup time of data and control lines	100 ns	
THD	Hold time of data and control lines	100 ns	
TSTH	STRB not active (high)	200 ns	
TSTLD	STRB not active (high) before LD is set active (low)	200 ns	
TLDST	LD not active, See Section 18.2.1 “PIO Write Operation Timing” p. 167	100 µs	
TLDL	LD active (low)	600 ns	
Tdelay			200 µs
Tresp			approx. 2–20 ms
TITR	Rise time of any input signal		250 ns

18.2.2 PIO Read Operation Timing

All timing sequence values are given for a cable length of about 0.5 m. For longer cables some headroom over specified minimum times is recommended.

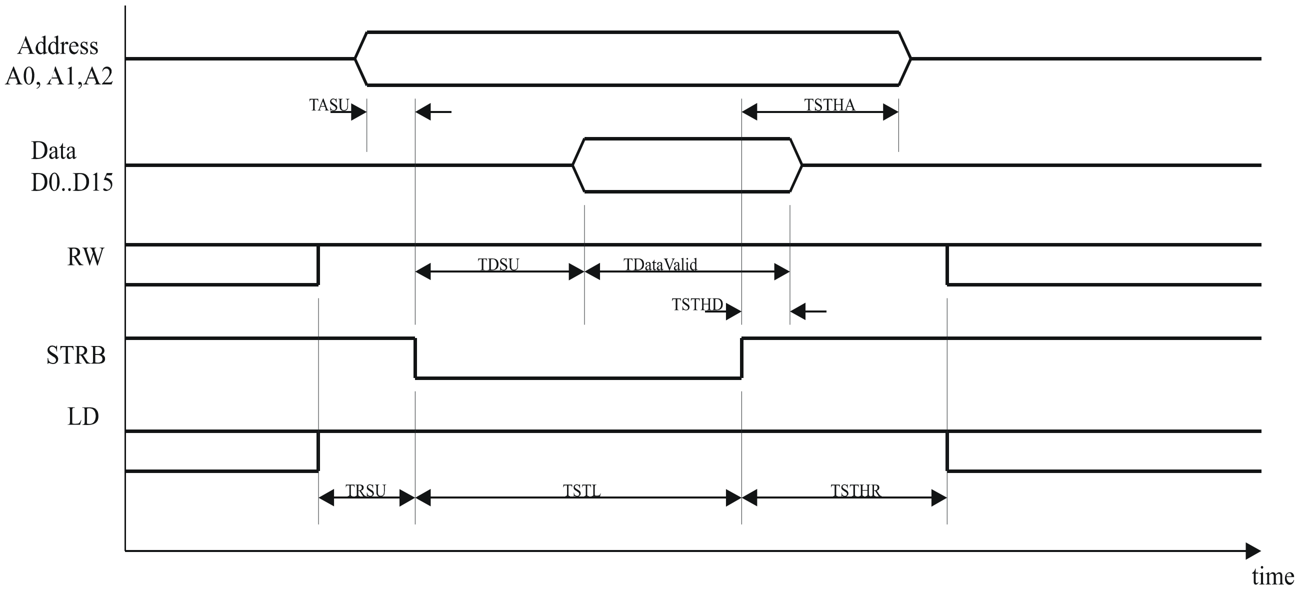


Fig. 81: Data read timing

Timing Definitions:

Parameter	Meaning	Min	Max
TASU	Address setup	0 ns	
TDSU	Data setup		900 ns
TRSU	RW line setup	0 ns	
TdataValid			TSTL – $TDSU_{max}$
TSTL	STRB active (Low)	900 ns	
TSTHD	STRB hold time		0 ns
TSTHA	Address hold time	0 ns	
TSTHR	RW line hold time	0 ns	

In firmware versions not older than 5.024 and 6.022, the On-Target4 line of the PIO connector can be operated in two different modes. It can either be the On-Target signal for axis 4, or it can be configured as an output data-valid, indicating that the PIO output data has been refreshed, and used as a trigger. See the D... Set/Read PIO ON-Target4 Line Function command on p. 104 for configuring this line.

When configured as data-valid signal, the timing is as follows:

- Frequency: same as servo-loop (5kHz)
- The position output lines are valid shortly before the rising edge and are guaranteed as long as the signal is in the TTL-high state (approx. 100µs high, 100µs low)

19 Terminology Table

The table below contains definitions and recent usage changes that were deemed necessary to ensure understanding of this product.

Current Term	Definition	Previous Usage(s)
activate DDL	make DDL available by entering the device-specific DDL license number	enable/activate DDL
average	sum of values divided by their number	mean
actual position	position, not relative to offset or baseline components	current position, real position
axis	one of the set of orthogonal logical axes for which servo-control can be enabled; axes are user definable	axis, channel
baseline position	axis-related position value that is set and used by various commands	default position
corrected axis	axis to benefit from a correction	error axis
enable DDL	turn on DDL compensation for a specific axis	enable/activate DDL
flatness compensation (non-linear)	feature allowing externally measured, repeatable, out-of-servo-loop, non-linear crosstalk from 2 axes to be corrected in a 3rd axis using a table of correction delta values (typically used to correct XY bumpiness in Z)	compensation
correction delta	additive correction factor	delta-Z
crosstalk compensation (linear)	feature allowing externally measured, linear crosstalk from one or more axes onto one or more other axes to be corrected	PZT gain optimization
current position	position at this very moment	actual position, real position
curve	a series of data points within a waveform or segment forming one ramp (triangle wave), single scan line or sine wave	curve segment, segment, axis
data point	one of an ordered collection of amplitude (position) values in data point memory	curve point
data point memory	ordered collection of data points; contents can be specified point by point or in groups (as curves); contents can be divided into segments (for filling) or waveforms (for output as targets)	position memory, scan-curve memory
internal number	value of an internal counter	internal counter value
move all together	move all axes to the target specified in previous MS commands	moving in one step
output waveform continuously	outputs unlimited number of cycles of a waveform from a wave generator	continuous move scan-curve

proportional gain	factor to be applied to the position specified in a proportional move command (SM) to calculate the target for the corresponding axis. When zero for an axis, that axis does not participate in proportional moves.	synchronous gain
proportional move	move of all axes having non-zero proportional gain settings; this feature enables straight-line moves in any direction in space with a single command	synchronous move
PZT channel	output channel associated with a piezo actuator, not necessarily parallel to the logical axes or the sensor channels	axis, channel
reset baseline	make the current position the new baseline position	make default position
segment	a group of contiguous data points in data point memory as defined with the PT command	axis, curve, scan curve, waveform
segment offset	an amplitude offset relative to the baseline position to be applied all points of a segment or waveform	curve function offset
sensor channel	input channel associated with one position-control sensor, before coordinate transformation to logical axes	axis, channel
single cycle output	output one waveform cycle from wave generator	single curve moving
step-size (autofocus)	size of the first step to use when attempting to reduce the out-of-focus signal to zero	focus amplitude
trigger line	electronic inputs and outputs that can be programmed in conjunction with the wave generator	trigger channel
wave generation	all functions involving definition of data points, segments, curves, waveforms as well as wave generator parameterization and operation	wave generator, scan generator, function generator
wave generator	one of two internal function blocks which can, on command, send a series of target positions to the servo-controllers of the axes	scan generator, function generator
wave generator move, waveform move	move in which a series of targets are sent directly from a wave generator to the servo-controller	scan-curve run, running wave generator, generator run, scan-curve move
waveform	A series of data points in data point memory which can be sent sequentially as targets by a wave generator	scan curve
waveform move enabled	an axis curve-control property: allow/ disallow wave generator control of the axis	curve moveable
target, MS target	target in an MS command for an axis which will only be moved to that target when a subsequent MS (Move All Together) command is executed	amplitude