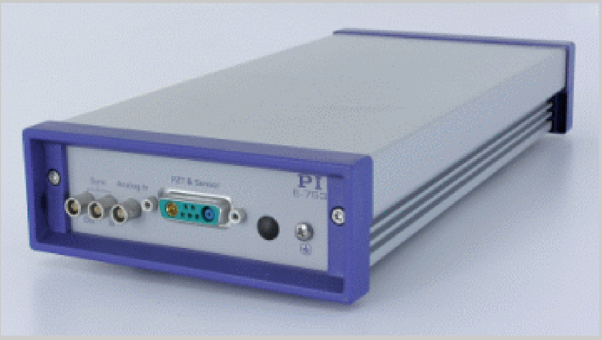
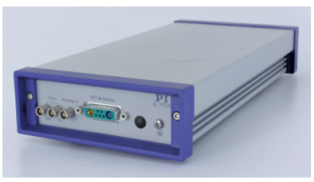
PZ193E User Manual
E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors

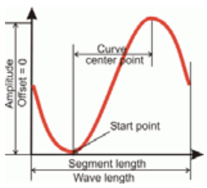
Version: 1.0.0 Date: 24 October 2007



This document describes the following product:

- E-753.1CD
  High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors





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

Piezo · Nano · Positioning

Declaration of Conformity

according to ISO / IEC Guide 22 and EN 45014

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

The manufacturer hereby declares that the product

Product Name: **High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors**

Model Numbers: **E-753**

Product Options: **all**

complies with the following European directives:

73/23/EEC, Low voltage directive

89/336/EEC, EMC-Directive

The applied standards certifying the conformity are listed below.

Electromagnetic Emission: EN 61000-6-3, EN 55011

Electromagnetic Immunity: EN 61000-6-1

Safety (Low Voltage Directive): EN 61010-1

October 24, 2007
Karlsruhe, Germany

Dr. Karl Spanner
President

First printing 24 October 2007
Document Number PZ193E, BRo, Release 1.0.0
E-753_User_PZ193E.doc

About this Document

Users of this Manual

This manual is designed to help the reader to operate the E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors. 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-753 as well as the software and hardware installation procedures and the commands which are required to put the associated motion system into operation.

Conventions

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

WARNING

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

DANGER

Indicates the presence of hazardous 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.

The software tools and the mechanics which might be mentioned within this documentation are described in their own manuals. All documents are available as PDF files. Updated releases are available for download at www.pi.ws (http://www.pi.ws) or via email: contact your Physik Instrumente Sales Engineer or write info@pi.ws (mailto:info@pi.ws).

Related Documents

PIMikroMove SM148E E-753 GCS LabVIEW PZ186E E-753 GCS DLL PZ194E	NanoCapture SM071E GCSData SM146E

All documents are available as PDF files on the distribution CD. Updated releases are available for download at www.pi.ws or via email: contact your Physik Instrumente sales engineer or write info@pi.ws.

Contents

1 Introduction

- For Nanopositioning Systems with Capacitive Sensors
- Powerful Digital Controller: 60 MHz, 32-bit, Floating Point DSP; 25 kHz Sampling Rate; 24-bit DAC
- Communication via Ethernet (TCP/IP), RS-232
- Automatic Calibration Data Coaching from Stage ID-Chip
- Analog Input for Control Signal or Sensor Signal
- Digital I/O Lines for Triggering Tasks
- Extensive Software Support

The E-753 single-channel digital piezo controller is ideal when it comes to meeting the most demanding accuracy and dynamic-performance requirements of nanopositioning systems with capacitive sensors.

Digital Linearization and Control Algorithms for Highest Accuracy

Linearization algorithms based on higher-order polynomials will not only improve the positioning accuracy to 0.001% of the travel range, but make possible simple replacement of the mechanics as well.

During fast periodic motion, as typical for scanning applications, the tracking accuracy can be further improved with Dynamic Digital Linearization (DDL). This optionally available control algorithm reduces the tracking error by a factor of up to 1000.

Higher Velocity and Bandwidth for Dynamic Applications

The controller is perfectly suited for high-dynamics operation thanks to its high-resolution DA-converter and high-performance voltage amplifier. The high-speed processor with a sensor sampling rate of 25 kHz assures settling times in the millisecond range.

Flexibility for a Variety of Applications

PI nanopositioning systems which are equipped with an ID-chip and calibrated with a digital controller have the mechanics-related calibration and servo-control parameters stored in the chip. The controller automatically adapts to the connected mechanics by the appropriate use of this data, so that recalibration is not necessary when system components are replaced.

The interfaces enable command of the mechanics either from a PC or with an analog control signal connected directly to the device. The digital I/O can be programmed for the most diverse types of triggering tasks.

The integrated wave generator can save and output periodic motion profiles. In addition to sine and triangle waves, arbitrary, user-defined profiles can be created.

The flexibly configurable data recorder enables simultaneous recording and read-out of up to 8 input and output signals, such as for sensor positions or control voltages. This makes the later association of events with saved data points possible.

Simple System Integration

Last but not least, all parameters can be checked and reset via software. System setup and configuration is done with the included NanoCapture™ and PIMikroMove® user-interface software. Interfacing to custom software is facilitated with included LabView drivers and DLLs. System programming is the same with all PI controllers, so controlling a system with a variety of different controllers is possible without difficulty.

In this document, the E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors is also referred to as "E-753" or "controller".

1.1 Prescribed Use

Based on their design and realization, E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensorss are intended to drive capacitive loads, in the present case, piezoceramic actuators. E-753s must not be used for applications other than stated in this manual, especially not for driving ohmic (resistive) or inductive loads.
Observe the safety precautions given in this User Manual.

E-753s can be operated in closed-loop mode using capacitive position sensors. Appropriate sensors provided by PI are integrated in the mechanics according to the mechanics product specifications.

The E-753 may only be used for applications suitable according to the device specifications. Operation other than instructed in this User Manual may affect the safeguards provided.

The verification of the technical specifications by the manufacturer does not imply the validation of complete applications. In fact the operator is responsible for the process validation and the appropriate releases.

The E-753 is a laboratory apparatus as defined by DIN EN 61010. It meets the following minimum specifications for safe operation (any more stringent specifications in the technical data table are, of course, also met):

- Indoor use only
- Altitude up to 2000 m
- Temperature range 5°C to 40°C
- Max. relative humidity 80% for temperatures up to 31°C, decreasing linearly to 50% relative humidity at 40°C
- Line voltage fluctuations not greater than ±10% of the line voltage
- Transient overvoltages as typical for public power supply
Note: The nominal level of the transient overvoltage is the standing surge voltage according to the overvoltage category II (IEC 60364-4-443).
- Degree of pollution: 2

1.2 Safety Precautions

WARNING—READ INSTRUCTION

Install and operate the E-753 Amplifier / 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 amplifier(s) used by the E-753 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:

- Do not touch the pins of the "PZT & Sensor" sub-D mix connector. The high voltage output may be active whenever the controller is turned on. The output value depends on the last control value and remains even after you have quit the terminal or the program from which the control value was commanded. Voltages between -30 V and +135 V can be present on the sub-D mix connector.

- The E-753 digital piezo controller does not contain any user-serviceable parts. Never disassemble the device. Hazardous voltage can be present on the internal components.

WARNING

Connect the AC power cord of the external power supply to the wall socket (100 to 240 VAC).

To disconnect the system from the supply voltage completely, remove the power plug from the wall socket.

Install the system near the AC outlet and such that the AC power plug can be reached easily.

CAUTION

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

CAUTION

Thermally stable systems have the best performance. For a thermally stable system, power on the E-753 at least one hour before you start working with it.

CAUTION

If the internal temperature goes out of range (□ 85 °C), the high voltage output of the E-753 will be deactivated. In that case communication with the E-753 is still possible, but any output-control values will be ignored, and the mechanics will not move.

CAUTION

!

Incorrect parameter values may lead to improper operation or damage to your hardware. Be careful when changing parameters.

It is strongly recommended to save the parameter values of the E-753 to a file on the host PC before you make any changes in non-volatile memory. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-753, use the *Device Parameter Configuration* window of NanoCapture™. See the NanoCapture™ manual for more information.

CAUTION

!

If the piezo stage starts oscillating (humming noise), switch off closed-loop operation immediately (set servo off). Otherwise the piezo stage could be irreparable damaged. Adjust the servo parameters (notch filter frequency, servo-loop P-term (loop gain), servo-loop I-term (time constant), servo-loop slew rate; see "Servo-Controller Dynamic Calibration" (p. 97) and the NanoCapture™ manual for more information).

CAUTION

!

Before you work with Dynamic Digital Linearization (DDL), use the NanoCapture™ software to eliminate any residual oscillations by adjusting the servo parameters (notch filter frequency, servo-loop P-term (loop gain), servo-loop I-term (time constant), servo-loop slew rate; see "Servo-Controller Dynamic Calibration" (p. 97) and the NanoCapture™ manual for more information). Using DDL could be critical if there are any residual oscillations in the system; DDL will then cause the oscillations to build up—the more wave generator cycles are used for DDL initialization, the stronger the effect.

1.3 Unpacking

Unpack the E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors with care. Compare the contents against the items covered by the contract and against the packing list.
The following components are included:

- E-753 High-Speed Single-Channel Digital Piezo Controller for Capacitive Sensors
- M-500.PS power supply with line cord
- C-815.34 RS-232 cable for connecting controller and host PC (null-modem cable)
- C-815.553 straight-through network cable
- C-815.563 special cross-over network cable which is required when the E-753 is connected to a PC with an Ethernet card
- E-753.CD Distribution CD, containing host software (see "Software Overview") and all manuals as PDF files
- E-753 User Manual in printed form (this document)

Inspect the contents for signs of damage. If parts are missing or you notice signs of damage, contact PI immediately.
Save all packing materials in case the product need be shipped again.

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

1.4 Additional Components

Contact your PI Sales Engineer or write info@pi.ws, if you need one of the following options / accessories:

E-710.SCN Dynamic Digital Linearization (DDL) option. The DDL makes it possible to achieve significantly better position accuracy for dynamic applications with periodic motion. It is used in conjunction with the wave generator output and in addition to the "normal" servo algorithm in closed-loop operation. You can activate the DDL after purchase and without opening the device. See "Dynamic Digital Linearization (DDL)" (p. 85) for more information.

E-500.ACD CD with LabVIEW™ analog driver set, available free of charge upon request. The E-753 can be controlled via a signal connected to its analog input line (see "Using the Analog Input" (p. 29) for more information). This signal can, for example, be generated using a D/A board in a PC. PI offers a LabVIEW™ driver set which can be used with certain D/A boards. It is compatible with the PI General Command Set (GCS) LabVIEW™ driver set available for all newer controllers from PI.
The PI LabVIEW™ drivers support all D/A converter boards from National Instruments that are compatible with DAQmx8.3. LabVIEW™ compatibility is given from version 7.1 upwards.

E-500.HCD Access to HyperBit™ Functionality for Enhanced System Resolution (Supports certain D/A boards). PI's patented Hyperbit™ technology for providing position resolution higher than that of the D/A board is in the E-500.ACD driver set. Activating Hyperbit™ requires purchase of the password, which can be obtained from PI under Order No. E-500.HCD.

K040B0107 Sync cable. If multiple E-753s are used, their sensor signals can be synchronized. To synchronize *n+1* E-753s you need *n* synchronization cables. See "Synchronization of Multiple Controllers" (p. 38) for more information.

1.5 Motion System Requirements

To start working with the E-753 digital piezo controller, your motion system must include the following components:

- A PC with Windows operating system (2000, XP, Vista). Software compatible with Linux is available on request
- Communication interface to the PC:
  A free COM port on the PC or
  An Ethernet card in the PC or
  A free access point on a network to which the PC is connected
- E-753 with power supply
- The mechanics (piezo stage) with which the controller was calibrated, or any suitable Physik Instrumente mechanics with ID-chip
- RS-232 or network cable to connect controller and host PC, or to connect the controller to the network
- E-753 CD with host software

1.6 Software Description

The following host software is included on the E-753 CD:

Software Tool	Short Description	Recommended for
GCS DLL (Windows DLL Library)	Allows program access to the E-753 from languages like C++. The functions in the DLL are based on the PI General Command Set (GCS).	Recommended for customers who want to use a DLL library for their applications. Needed by the LabVIEW driver set and by PIMikroMove®
LabVIEW drivers	LabVIEW is a software tool (available separately from National Instruments) for data acquisition and process control. The E-753 LabVIEW software consists of a collection of virtual instrument (VI) drivers for the E-753 controller. This driver set supports the PI General Command Set (GCS).	Users who want to use LabVIEW for programming their application
NanoCapture™	A powerful graphical user interface which gives easy access to step response measurement or waveform motion, but also provides access to 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, AutoZero adjustment, wave editor, command entry	Users who want to test the equipment before or instead of programming an application and who want to optimize the servo-loop behavior and other controller parameters.

PIMikroMove®	This GUI for Microsoft Windows platforms is the operating software for this and many other PI controllers. PIMikroMove® permits you to start your motion system—host PC, controller and stage(s)—immediately without the need to write customized software. It offers motion-control displays and features that in many cases make it unnecessary to deal with ASCII-format commands. It also has a complete command input facility, which represents an easy way to experiment with various commands. PIMikroMove® uses the GCS DLL described above to command the controller.	Users who want to test the equipment before or instead of programming an application and who want to learn how to use the commands. For motorcontrollers, PIMikroMove® offers an easy way to optimize the servo parameters.
PITerminal	PITerminal is a Windows GUI which can be used as a simple terminal with almost all PI controllers. See the PIMikroMove® manual for a description.	Users who want to send the commands of the PI General Command Set (GCS) directly.

2 Operation

2.1 Front and Rear Panel Elements

2.1.1 Front Panel Elements

Figure 1: E-753 front panel

Name	Function
Sync Out	Lemo socket for synchronization of multiple E-753s. Connects to next E-753 ("Synchronization of Multiple Controllers" (p. 38)). See "Sync Out Socket" (p. 224) for pinout.
Sync In	Lemo socket for synchronization of multiple E-753s. Connects to previous E-753 ("Synchronization of Multiple Controllers" (p. 38)). See "Sync In Socket" (p. 223) for pinout.
Analog In	Lemo socket for analog input line; connect an external sensor or control-signal source ("How to work with the Analog Input" (p. 29)). See "Analog In Socket" (p. 222) for pinout and specifications.
PZT & Sensor	Sub-D Mix connector for piezo stage; carries the voltage for the piezo actuator (-30 to 135 V) and the signal of the capacitive sensor in the mechanics. See "PZT & Sensor Socket" (p. 221) for pinout.
	Because grounding is not assured over the power connection, the ground screw marked with this symbol must be connected to a protective ground.

2.1.2 Rear Panel Elements

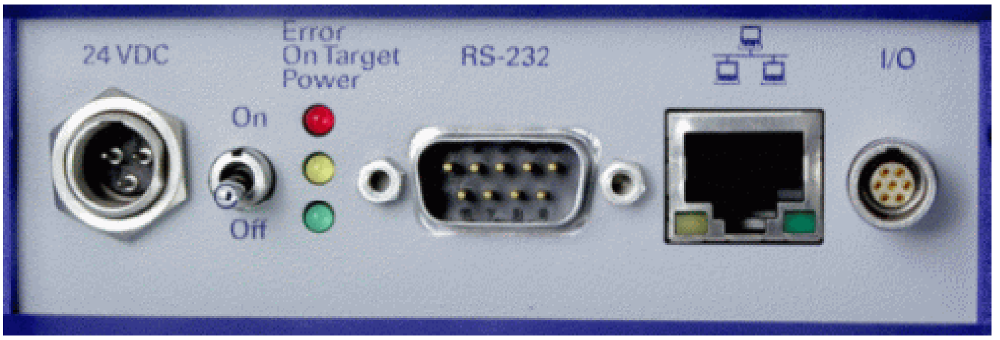
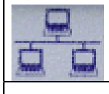


Figure 2: E-753 rear panel

Name	Function
24 VDC	Connector for power supply. See "24 VDC Socket" (p. 223) for pinout.
On / Off toggle switch	Switch power on / off
Error LED (red)	Error indicator; when LED lights up, error code is non-zero and can be queried and cleared using the ERR? command (p. 133).
On Target LED (yellow)	On-target indicator (only relevant in closed-loop operation); lights up if the axis is on target; the on-target state can also be queried with the ONT? command (p. 147).
Power LED (green)	Power on and ready indicator. After the boot sequence which takes about 20 seconds, the LED blinks in normal operation, representing the DSP activity. Note that depending on the communication interface used, it might take another period of about 30 seconds before communication can be established. Permanent glow of the Power LED indicates that a command is received.
RS-232	Serial connection to host PC. See "RS-232 Socket" (p. 223) for pinout.
	Network connection over TCP/IP. See "TCP/IP Connection" (p. 59) for more information.
I/O	Lemo socket for digital input and output lines; can be used to trigger external devices and to send start/stop signals to the wave generator. See "I/O" socket (p. 222) for the availability of the lines and for pinout.

2.2 First Steps

2.2.1 Installing the Software on the Host PC

- Insert the E-753 CD in your host PC.
- If the Setup Wizard does not open automatically, start it from the root directory of the CD with the  icon.
- Follow the on-screen instructions. You can choose between “typical” and “custom” installation. Typical components are LabView drivers, DLLs, and PIMikroMove® and NanoCapture(TM). “Typical” is recommended.

For an overview over the host software provided see "Software Description" (p. 11).

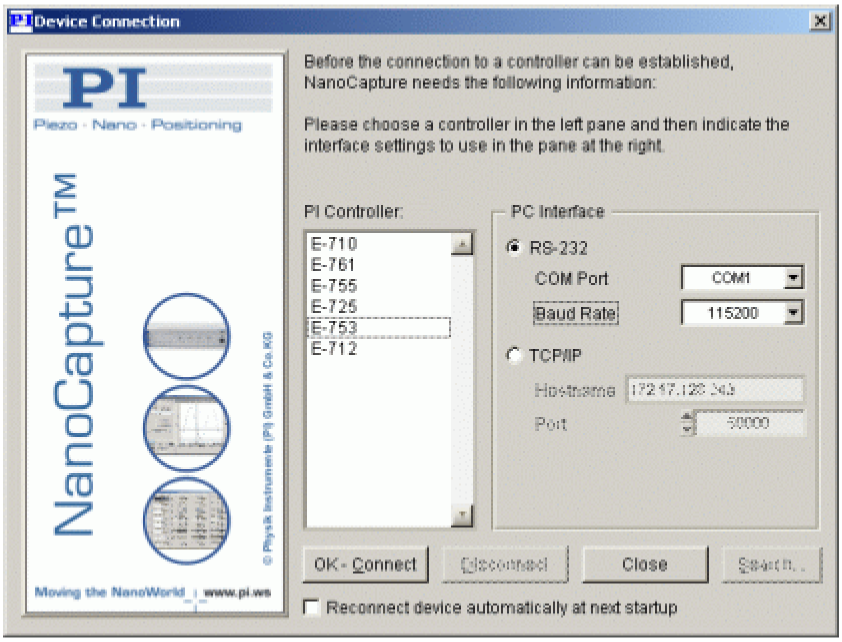
2.2.2 Quick Start

The following instructions illustrate the first steps with the E-753 using NanoCapture™ because that software offers a convenient user interface for operating the system. See the NanoCapture™ Manual for the complete, more-detailed software description and "Installing the Software on the Host PC" (p. 15) for how to install the program.

1 Interconnect your system. Find details in a dedicated section (p. 20). If you want to synchronize multiple E-753s, see also "Synchronization of Multiple Controllers" (p. 38).

2 Power on the system. Find details in a dedicated section (p. 22).

3 Start NanoCapture™ on the host PC. It opens with the *Device Connection* window.



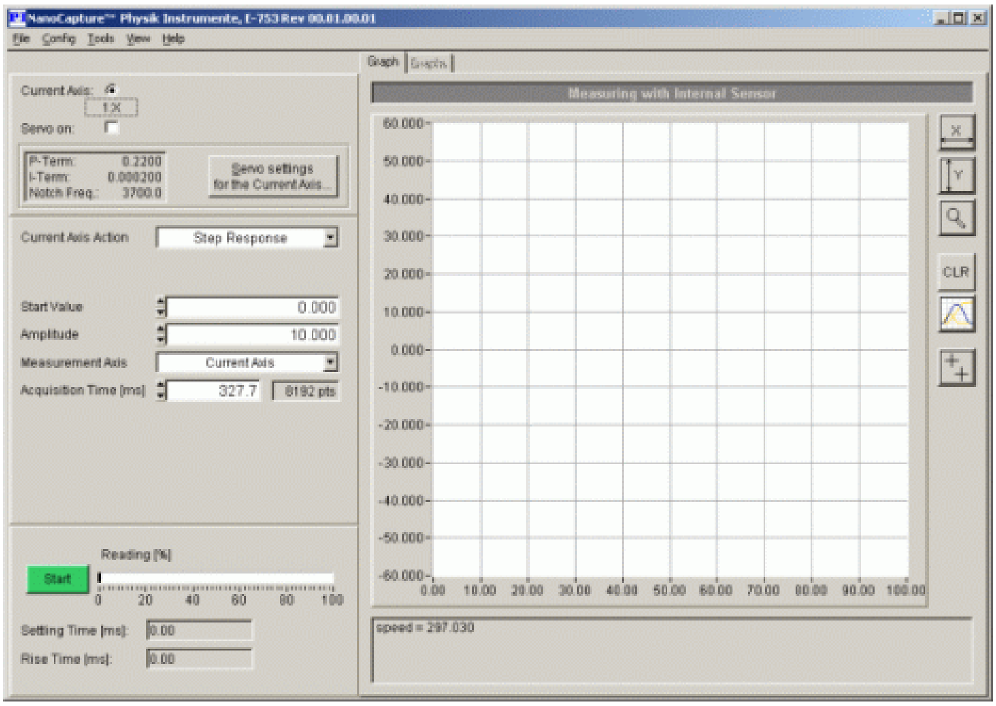
4 Establish a connection to the E-753 from NanoCapture™. This can be done over either the TCP/IP or the RS-232 interface, using the corresponding entry in the *Device Connection* window (the interface selected requires a corresponding hardware connection!). See "Communication" (p. 57) for details.

Notes:

Before communication between E-753 and host PC can be established, the power-on sequence of the controller must have finished successfully (takes approx. 20 s, after which the Power LED blinks continuously at regular intervals). TCP/IP communication: If no DHCP server is available on the network or if a point-to-point connection between host PC and controller is being used, it might take another period of about 30 seconds before communication is possible.

If the controller is already connected to your or another host PC via TCP/IP, a second TCP/IP session cannot be established.

When the connection is successfully established, the NanoCapture™ main window opens automatically (see figure below).



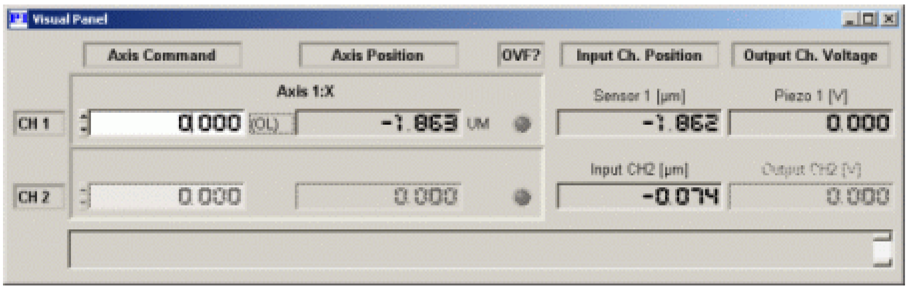
5 If necessary, perform the AutoZero procedure. To do this, open the *AutoZero* window using the *Tools* ⇒ *AutoZero* menu sequence. During the AutoZero procedure, the output voltage which is to be applied when the axis is at the zero position is set, and an appropriate calibration routine is started. See "AutoZero Procedure" (p. 23) for details and for the decision as to whether AutoZero is necessary.

6 Make some test moves:

In the NanoCapture™ main window, make sure that axis 1 is activated (*Current Axis* radio button must be selected) and in open-loop operation (*Servo on* must be unchecked).

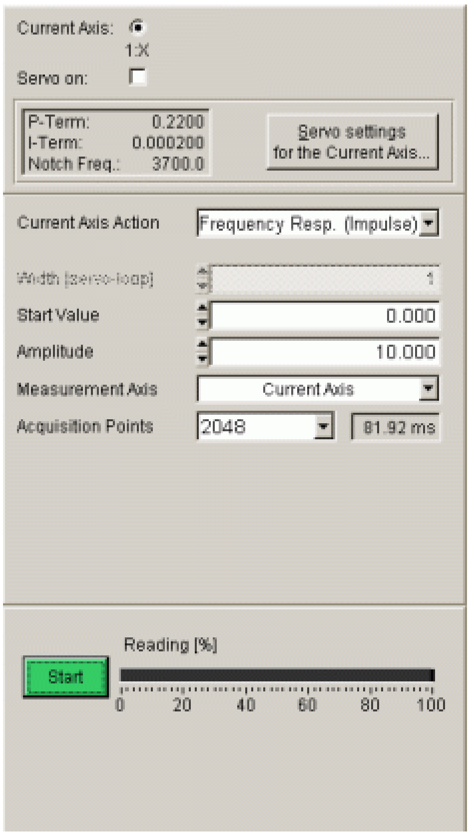
To make first test moves, open the *Visual Panel* window (see figure below) using the *View* ⇒ *Visual Panel* menu entry. In the *Axis Command* field for axis 1, enter the control value 0 and press Enter on your keyboard. Then enter the value 10 and press Enter. Increment the *Axis Command* control value this way by steps of 10, up to the upper travel range limit of your stage, and then reduce it in an analogous manner back to zero. In doing so, observe the position display for axis 1 (in the *Axis Position* field) and the current output voltage for the piezo actuator in the stage (in the *Piezo 1* field). The values should follow the commanded control values: The axis position should always correspond approximately to the commanded value and the output voltage should become noticeably different from 0 V and then go back to zero again during the procedure (with the E-753s, the output voltage range is -30 to +135 V; the output voltage corresponding to a given control value depends on the nominal travel range of the connected stage).

You can also increment or reduce the control value in the *Axis Command* field by steps of 1 using the arrow controls beside the field.

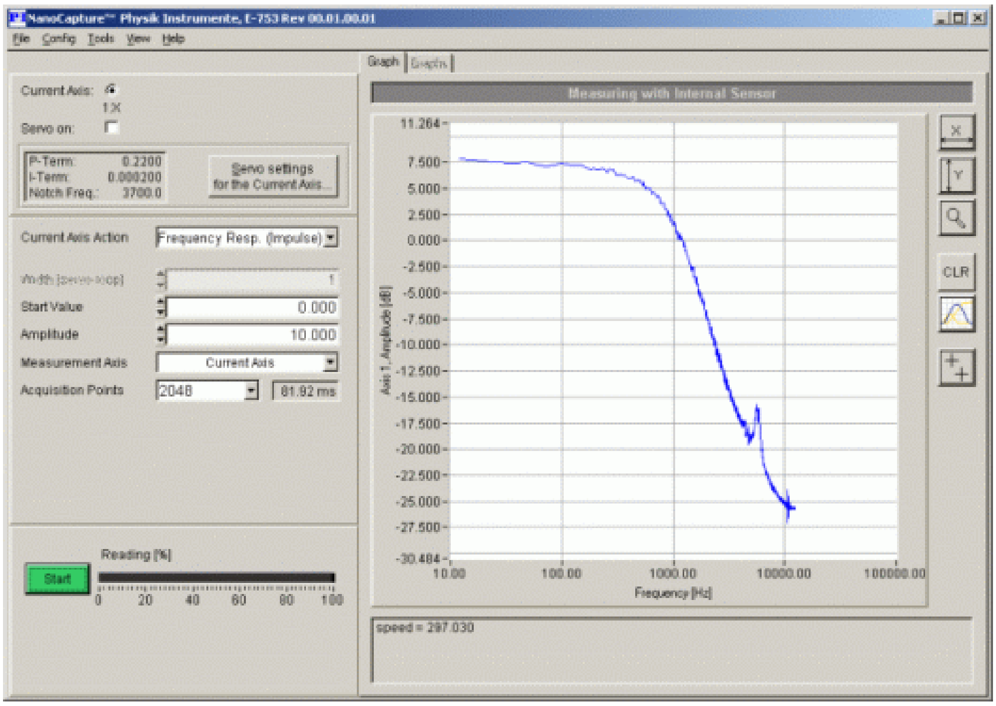


7 Make an open-loop frequency response measurement in the NanoCapture™ main window to determine the resonant frequency:

Preparation (see figure below :
Select *Frequency Resp. (Impulse)* for *Current Axis Action* and *Current Axis* for *Measurement Axis*. The default entries for *Start Value* and *Amplitude* can be used, but make sure that the values are inside the travel range of the stage. They correspond to axis positions in µm (as long as the preset system configuration for the transformation of axis control values to drive voltage values is correct).



Perform the impulse motion:
Press the green *Start* button to start performing the impulse, recording the response and analyzing and displaying it in the form of a Bode frequency response diagram (internally, NanoCapture™ sends the required DRC (p. 128), IMP (p. 143) and DRR? (p. 130) commands to the E-753). See figure below for a typical result.



In the Bode frequency response diagram, you can identify the resonance peak (in the figure above at about 5.7 kHz). If there are resonances which are intolerable in your application, adjust the notch filter settings of the E-753 before you switch to closed-loop operation for the first time (servo on). Furthermore, it might be necessary to readjust the preset servo parameters. See "Servo Controller Dynamic Calibration" (p. 97) for more information.

2.3 How to Interconnect the System

1 Connect the E-753 either to the same TCP/IP network as the host PC, or to the host PC directly:

• Use a normal network cable if connecting to a network access point. Note that many network administrators have set their networks to forbid unknown devices like the E-753 to log on.

• To connect directly to the host PC, use either a cross-over network cable or a null-modem RS-232 cable.

See "Communication" (p. 57) for more information.

2 If you want to use the digital I/O lines of the E-753 to trigger external devices and/or to send start/stop signals to the integrated wave generator, connect these devices/signals to the "I/O" socket (p. 222)

on the rear panel of the E-753. See "Using Trigger Input and Output" (p. 38) for more information.

3 If you want to use the analog input line of the E-753 for an external sensor or for control value generation, connect the appropriate input to the "Analog In" socket (p. 222) on the front panel of the E-753. See "How to work with the Analog Input" (p. 29) for more information.

4 Connect the piezo stage to the "PZT & Sensor" socket (p. 221) of the E-753:

A label on the case of the E-753 indicates which piezo stage (axis) was calibrated with which controller (axis). Be sure to respect this assignment when connecting the stage to the controller. When you are using a piezo stage with ID-chip together with the E-753, piezo stages can be easily exchanged because the calibration data is in the ID-chip; for details see "ID-Chip Support / Stage Replacement" (p. 95).

Note: When you connect a stage when the controller is powered on, the ID-chip of the stage is not read by the controller. To read the ID-chip data, the controller must be power cycled or rebooted using the RBT command (p. 148) or the corresponding host software functions.

5 Connect the ground screw on the frontpanel of the E-753 to a protective ground because grounding is not assured over the power connection.

6 Connect the "24 VDC" socket (p. 223) of the E-753 to the included wide-range power supply and connect the AC power cord of the power supply to the wall socket.

The E-753 comes with a 24 V wide-range-input power supply that can be used with line voltages from 100 VAC to 240 VAC at 50 or 60 Hz.

This section refers to a system with a stand-alone E-753. Synchronization of multiple E-753s is described in "Synchronization of Multiple Controllers" (p. 38).

2.4 Power On / Reboot Sequence

CAUTION

!

Thermally stable systems have the best performance. For a thermally stable system, power on the E-753 at least one hour before you start working with it.

To power on the E-753, first switch on the external power supply and then the controller itself using the On/Off toggle switch on the rear panel.

On power-on or reboot (with the RBT command (p. 148)), the E-753 performs firmware verification and copies information from non-volatile memory to volatile memory. The power-on/reboot sequence takes about 20 seconds. During this time, all LEDs light up with different intervals. Afterwards the controller goes into the normal operating state.

The Power LED blinks in normal operation, representing the DSP activity. Permanent glow of this LED indicates that currently a command is received. After approx. 1 second, the LED resumes blinking. Note that the host software, e.g. PIMikroMove®, may continuously send polling commands to the E-753 and the Power LED will not blink anymore.

When a different stage with ID-chip is connected to the controller, the stage parameters from the ID-chip will be written both to non-volatile and volatile memory upon the next controller power-on or reboot. On subsequent boots with the same stage, the complete parameters will be rewritten only if the "Power Up Read ID-Chip" option is enabled. By default, this option is disabled to maintain optimized parameter settings on the controller. See "ID-Chip Support / Stage Replacement" (p. 95) for more information.

NOTE

Before communication between E-753 and host PC can be established, the power-on sequence of the controller must have finished successfully (takes approx. 20 s, after which the Power LED blinks continuously at regular intervals). TCP/IP communication: If no DHCP server is available on the network or if a point-to-point connection between host PC and controller is being used, it might take another period of about 30 seconds before communication is possible.

2.5 AutoZero Procedure

For systems with linear piezo actuators, both the range of sensor position values and the range of the output drive voltages are limited. If mechanical drift of the piezo actuator causes too great a shift in the relation between these ranges, then the usable closed-loop travel range will be reduced. Such an offset can be compensated by the AutoZero function. The following example illustrates the AutoZero functionality.

Take a linear single-axis piezo stage and its sensor with a position range from 0 µm to 200 µm. The controller has an output voltage range from –30 V to +135 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 changes by 100 V. In this case the controller has a voltage reserve of about 30 V in negative and 35 V in positive direction. Ideally the position of the piezo stage should be 0 µm if the closed-loop control is switched off and the output voltage of the controller is 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 30% of the position range (60 µm) in negative direction / 35% (70 µm) in positive direction, this offset can be compensated by the controller when closed-loop control is activated, using the 30% / 35% voltage reserve. If the position offset is greater than 60 µm / 70 µm, it will no longer be possible to move the stage to 200 µm / 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.

NOTE

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

When to perform AutoZero

When the piezo stage is first integrated into the application environment, AutoZero must be run. Afterwards, AutoZero should only be executed in the following cases:

- The system is subjected to temperature changes.
- The load applied to the stage has changed (note that the effect of load changes depends on the stiffness of the stage).

- The integration of the stage has changed (e.g. the orientation).

Especially if absolute moves are needed, AutoZero should not be executed during normal operation because AutoZero changes the mechanical zero position of the piezo stage.

NOTE

AutoZero is not effective on non-linear axes.

AutoZero can be performed using the ATZ command (p. 117). See the command description for further information on the AutoZero procedure.

NOTES

The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished. See "Control Value Generation" (p. 51) for more information.

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

In NanoCapture™, use the *Tools* ⇒ *AutoZero* menu sequence to open the *AutoZero* window (see figure below) where you can start the AutoZero procedure.

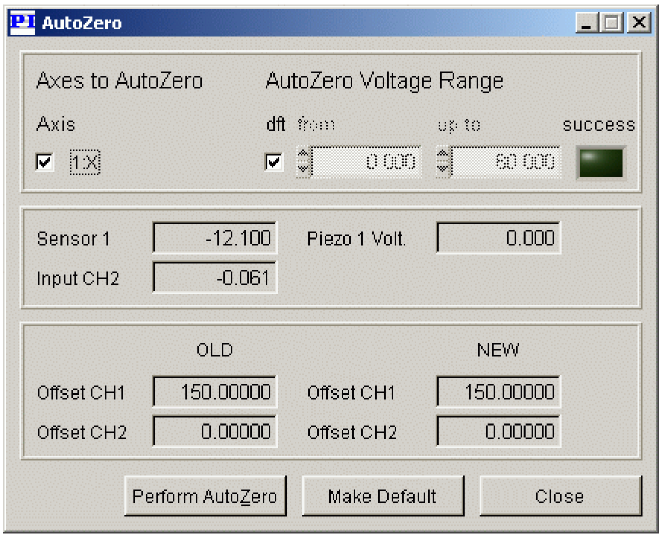


Figure 3: AutoZero window for E-753 in NanoCapture(TM)

Press Perform *AutoZero* to start the AutoZero procedure. The green *success* LED indicates the success of the procedure. The resulting offset constant value can be saved to the non-volatile memory of the E-753 by pressing the *Make Default* button. See the NanoCapture™ manual for more information.

2.6 How to Command Axis Motion

The E-753 can be operated in open-loop or closed-loop mode. In closed-loop operation, sensor feedback participates in the control value generation, while it is not used in open-loop operation. The operating mode can be selected with the SVO command (p. 163), and the axis motion can result from multiple control sources (see "Control Value Generation" (p. 51) for details):

- Move commands: MOV (p. 144) and MVR (p. 145) in closed-loop operation; SVA (p. 160) and SVR (p. 164) in open-loop operation; IMP (p. 143) and STE (p. 158) for both operating modes (see below for examples)

- Wave generator output for periodic motion in either operating mode (see "Wave Generator" (p. 70) for more information and examples)

- Analog control input for motion in either operating mode (see "Using the Analog Input" (p. 29) for more information and examples)

The sources are listed above according to their write priority, starting with the lowest priority. This means that, for example, the wave generator output will overwrite the control value given by a move command.

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

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

Command String to Send	Action Performed
SVO?	Check servo-control state. The response should be "1=0" which means that axis 1 is in open-loop operation, i.e. there is no correction of drift or other effects.
VOL?	Check the current output voltage. The response should be 0 V unless otherwise in the preset system configuration.
POS?	Check the current position of axis 1. The current position value should be approximately 0 µm, due to the calibration settings of the system.

SVR 1 10	Send this command five times. With each command, axis 1 moves relative by 10 µm (approximately). Note: This is true as long as the preset system configuration for the transformation of axis control values to driving voltage values was not changed.
VOL?	Check the current output voltage. It should be positive and noticeably different from 0 V. With the E-753s output voltage range of -30 to +135 V, the current output voltage depends on the nominal travel range of the connected stage.
POS?	The current position value should be approximately 50 µm.

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

Command String to Send	Action Performed
SVO 1 1	Set servo-control on (closed-loop operation) for axis 1; this also writes the current axis position to the target register, to avoid jumps of the mechanics.
POS?	Get current position of axis 1. The current axis position value should be approximately 50 µm.
MOV 1 10	Axis 1 moves to an absolute position of 10 µm.
POS?	The current position of axis 1 should be exactly 10 µm.
MVR 1 24	Axis 1 moves relative by 24 µm.
POS?	The new position should be exactly 34 µm

The E-753 has a real-time data recorder. It is able to record several input and output signals from different data sources during the axis motion. Data recording is triggered automatically with four commands:

- STE (p. 158) (step response measurement),
- IMP (p. 143) (impulse response measurement),
- WGO (p. 179) (wave generator start) and
- WGR (p. 183) (restarts recording when the wave generator is running).

Recording always takes place for all data recorder tables and ends when the data recorder tables are completely filled. See "Data Recording" (p. 66) for more information.

2.7 How to Customize the System

NOTES

Values stored in non-volatile memory are power-on defaults, so that the system can be used in the desired way immediately. Note that PI records the data files of every E-753 controller calibrated at the factory for easy restoration of original settings should that ever be necessary.

For stages with ID-chip, to make the optimized settings available in the future, the option "Power Up Read ID-Chip" must have "disabled" as its power-on default (value of parameter 0X0F000000 = 0 in non-volatile memory). See "ID-Chip Support / Stage Replacement" (p. 95) for more information.

2.7.1 Set System Parameters

!

CAUTION

Incorrect parameter values may lead to improper operation or damage to your hardware. Be careful when changing parameters.

It is strongly recommended to save the parameter values of the E-753 to a file on the host PC before you make any changes in non-volatile memory. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-753, use the *Device Parameter Configuration* window of NanoCapture™. See the NanoCapture™ manual for more information.

To adapt the E-753 to your application, you can modify parameter values. The parameters available depend on the controller firmware. With HPA? (p. 136) you can obtain a list of all available parameters with information about each (e.g. short descriptions). The volatile and non-volatile memory parameter values can be read with the SPA? (p. 157) or SEP? (p. 153) commands, respectively. Note that many parameters are "protected" by higher command levels, as indicated in the "Command Level" column in the "Parameter Overview" table (p. 205). By going to command level 1 using the CCL command (p. 119), it is possible to change level-1 parameters. Parameters with level 2 or higher are reserved for service personnel.

Using the "general" modification commands SPA, RPA, SEP and WPA, all parameters for which the currently active command level has write permission can be changed in volatile memory (SPA (p. 154), RPA (p. 149)) or in non-volatile memory (SEP (p. 152), WPA (p. 186)). It is recommended that any modifications be first made with SPA, and when the controller runs well, saved using WPA. Do not change the current interface settings with SPA—except of the baud rate—because it will not be possible to maintain communication afterwards.

In addition to the "general" modification commands, there are commands which change certain specific parameters. All the commands listed below, except of IFS, change the parameter value only in volatile memory, and WPA must be used to save the value to non-volatile memory. IFS changes and saves the interface parameters directly in non-volatile memory only.

AOS (p. 114) (analog input offset)
ATZ (p. 117) (Autozero Matched Offset as the autozero result)
DPO (p. 127) (internal DDL processing parameters)
IFC (p. 138) (baud rate for RS-232 serial connection)
RTR (p. 150) (record table rate)
VEL (p. 170) (Servo Loop Slew-Rate)
WOS (p. 184) (wave generator output offset)
WTR (p. 189) (wave table rate)

IFS (p. 140) (interface parameters in non-volatile memory)

The NanoCapture™ host software gives access to parameter values in a more convenient way. Use its *Device Parameter Configuration* window to check/edit the individual parameters. See the NanoCapture™ manual for more information.

See "Controller Parameters" (p. 203) for detailed information.

2.7.2 Adjustment for Load Changes

If the controller and the attached piezo stages are ordered together and if PI has sufficient knowledge of your application, then the parameters of the closed-loop control algorithm (servo parameters) will be set to suitable values at the factory, and, if present, saved in the stage's ID-chip (p. 95). Modification of those parameters will, however, be necessary if the load applied to the piezo stage is changed. In this case, for each servo-controlled axis the following parameters may need to be modified in the controller:

- Notch filter frequencies (first notch filter: parameter ID 0x08000100, second notch filter: parameter ID 0x08000101)
- Servo-loop loop gain (P-term, parameter ID 0x07000300)
- Servo-loop time constant (I-term, parameter ID 0x07000301)

- Servo-loop slew rate (parameter ID 0x07000200)

It is most convenient to use NanoCapture™ to change these parameters.

NOTE

You have to switch to command level 1 before you can change the servo parameters (see the CCL command (p. 119) or the NanoCapture™ manual).

See "Servo-Controller Dynamic Calibration" (p. 97) for detailed information.

2.8 Using the Analog Input

2.8.1 How to Work with the Analog Input - Overview

The E-753 provides an analog input line. See "Analog In Socket" (p. 222) for pinout and specifications. For highest resolution, it is recommended to use the full range of ±10 V.

You can use the analog input line as follows:

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

Irrespective of the intended usage, the analog input values must first be scaled to suitable position values (see "Scaling the Analog Input" (p. 31)). Then, to set the usage of the analog input, it is furthermore necessary to change certain controller parameters. See "Connect an External Sensor" (p. 34) or "Connect a Source for Control Value Generation" (p. 35) for details.

NOTES

Wherever changing parameter values is mentioned, you can do this using SPA (p. 154) (volatile memory) or SEP (p. 152) (non-volatile memory). Furthermore, you can use WPA (p. 186) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. To have write access to certain parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119). To read parameter values, query with the SPA? (p. 157) or SEP? (p. 153) commands.

The NanoCapture™ host software gives access to parameter values in a more convenient way. Use its *Device Parameter Configuration* window to check/edit the individual parameters. See the NanoCapture™ manual for more information.

The analog input line is represented in the controller firmware as input signal channel 2 (see "Accessible Items and Their Identifiers" (p. 43) for more information).

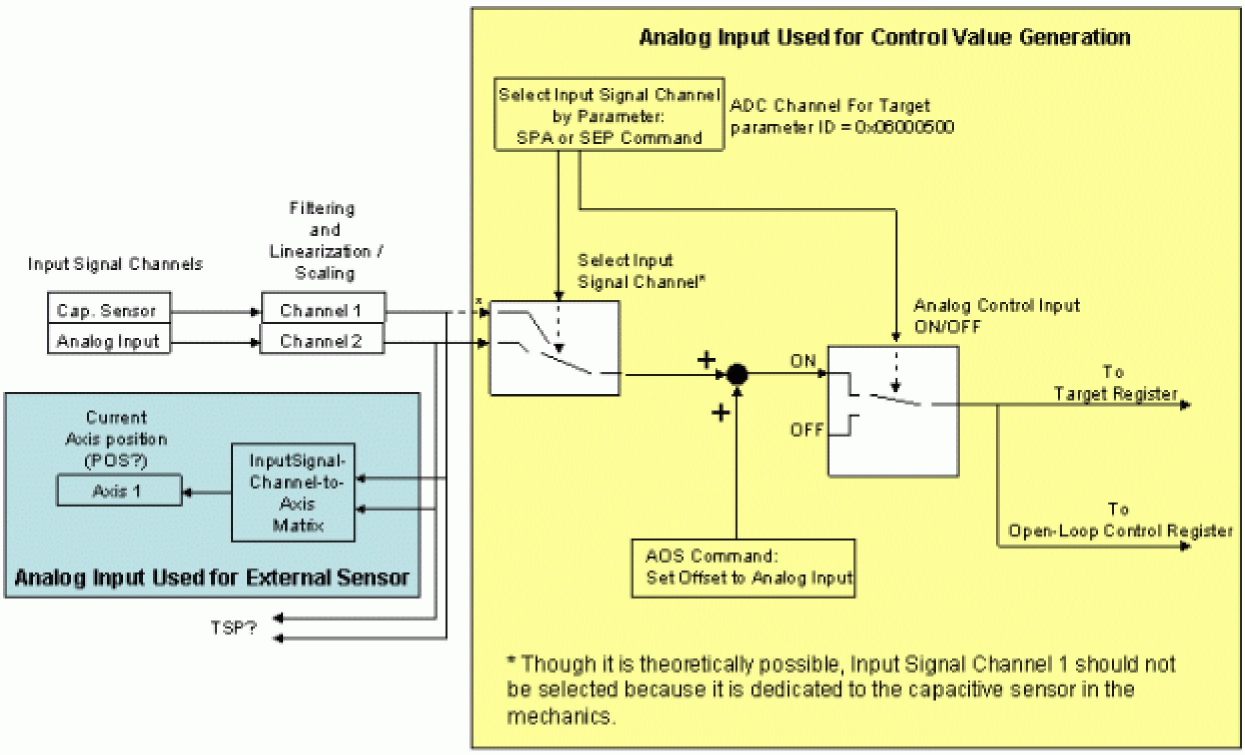


Figure 4: Overview over the usage of the analog input line

NOTE

The number of analog input lines available on the controller can be calculated by subtracting the Number Of Sensor Channels parameter, ID 0x0E000B03, from the Number Of Input Channels parameter, ID 0x0E000B00.

2.8.2 Scaling the Analog Input

Before the analog input line can be used with an external sensor or with a control-signal source, the input levels must be associated with suitable position values. To do this, adjust the OFFSET (parameter ID 0x02000200) and the GAIN (parameter ID 0x02000300) of the Mechanics linearization polynomial according to the travel range of the stage and the input signal range. See below for details. The TSP? command (p. 168) reports the analog input values after the scaling as position values in µm.
In addition, the digital filter parameters can be adjusted. See "Digital processing" (p. 48) for details.

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

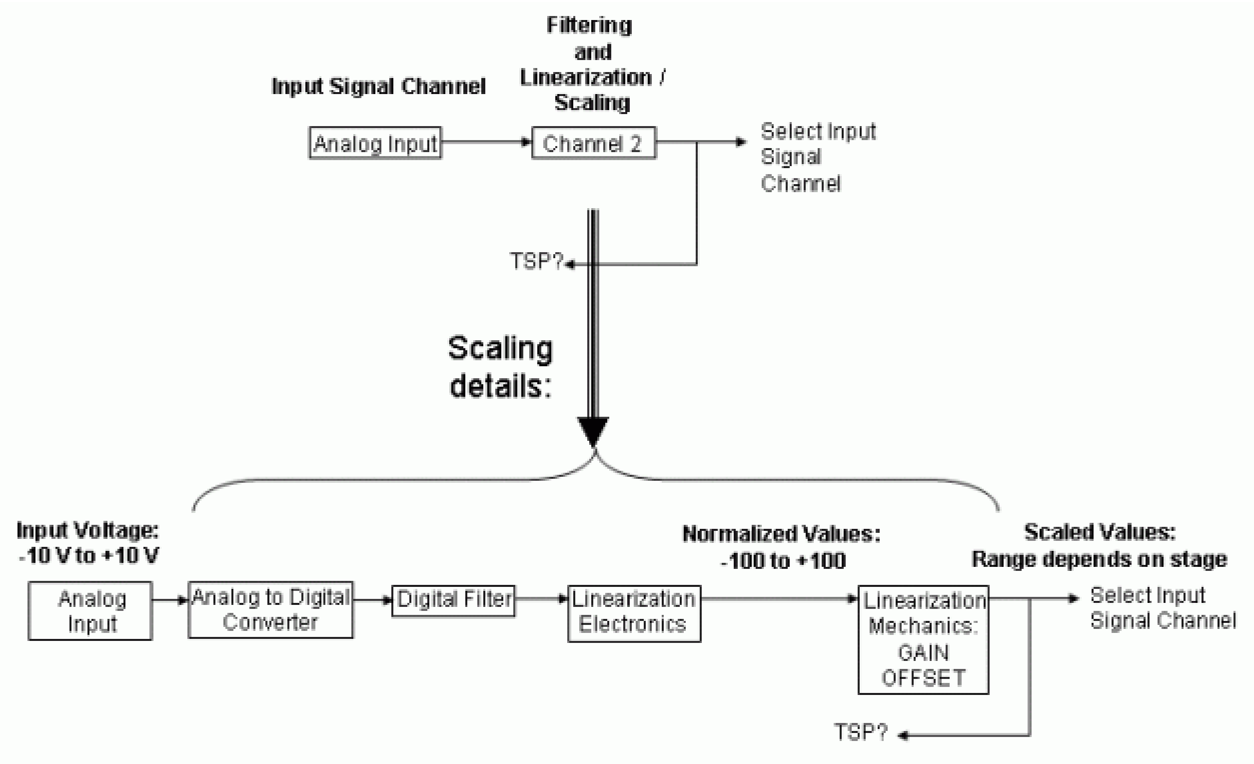


Figure 5: Processing of the analog input signal, detail from the overview figure above

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

Normalized Value:
The polynomial used for electronics linearization (see "Digital processing" (p. 48) for details) converts the analog input voltage to a number in the range of -100 to +100. An input voltage value of -10 V always corresponds to -100, and +10 V corresponds to +100 respectively.

Scaled Value:
The range depends on the stage and can be set by the coefficients of the polynomial used for Mechanics linearization (see "Digital processing" (p. 48) for details):

ScaledValue = OFFSET + GAIN * NormalizedValue

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

0x02000500, 0x02000600).

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

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

The values of "MinScaledValue" and "MaxScaledValue" depend on the travel range of the stage:
"MinScaledValue" is given by the TMN? (p. 166) answer (is defined by the Range Limit min parameter, ID = 0x07000000), and "MaxScaledValue" is given by the TMX? (p. 166) answer (is defined by the Range Limit max parameter, ID = 0x07000001).

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

Example 1:
The full range of -10 V to +10 V is to be used (this is recommended for highest resolution).
MinNormalizedValue = -100
MaxNormalizedValue = +100
GAIN = (120 - (-20)) / (100 - (-100)) = 0.7
OFFSET = 120 - 0.7 * 100 = 50
ScaledValue = 50 + 0.7 * NormalizedValue
So you have to send
SPA 2 0x02000200 50
SPA 2 0x02000300 0.7
to adjust the GAIN and OFFSET parameters for input signal channel 2 (= the analog input line) in the E-753.

Example 2:
Only positive input voltages are to be used, i.e. the range is 0 V to +10 V.
MinNormalizedValue = 0
MaxNormalizedValue = +100
GAIN = (120 - (-20)) / (100 - 0) = 1.4
OFFSET = 120 - 1.4 * 100 = -20
ScaledValue = -20 + 1.4 * NormalizedValue
Send:
SPA 2 0x02000200 -20
SPA 2 0x02000300 1.4

Example 3:
Positions with positive sign shall correspond to positive input voltages, and positions with negative sign shall correspond to negative input voltages. The positive input voltage ranges to +10 V.
Then, the following is valid provided that the absolute value of the negative positions will never be greater than the positive positions.
MinNormalizedValue = 0
MaxNormalizedValue = +100
GAIN = (120 - 0)) / (100 - 0) = 1.2
OFFSET = 120 - 1.2 * 100 = 0
ScaledValue = 1.2 * NormalizedValue
Send:
SPA 2 0x02000200 0
SPA 2 0x02000300 1.2
Note that these OFFSET and GAIN values would also be valid for a stage with a travel range of 0 to 120 µm if there were only positive input voltages to +10 V.

Example 4:
The same conditions as in example 3 are valid, but the positive input voltages range to +5 V.
MinNormalizedValue = 0
MaxNormalizedValue = +50
GAIN = (120 - 0)) / (50 - 0) = 2.4
OFFSET = 120 - 2.4 * 50 = 0
ScaledValue = 2.4 * NormalizedValue
Send:
SPA 2 0x02000200 0
SPA 2 0x02000300 2.4

2.8.3 Use as External Sensor Input

To let the sensor on the analog input line participate in the position monitoring of an axis, set the corresponding coefficient in the InputSignalChannel-to-Axis matrix to 1 for that axis. With the E-753, this is the Position From Sensor 2 (parameter (ID 0x07000501) for axis 1. Send:
SPA 1 0x07000501 1
to change the coefficient in volatile memory. In NanoCapture™, this parameter is available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

One possible application could be that only the external sensor on the analog input line is to be used for position control of axis 1. In this case, the signal of the internal capacitive sensor must be excluded from the position monitoring of axis 1. To do this, set the Position From Sensor 1 coefficient in the InputChannel-to-Axis matrix to zero by sending:
SPA 1 0x07000500 0

The position of axis 1 (i.e. the POS? response) will then be based on the external sensor only , but it is still possible to read the signals of both the capacitive and the external sensor using the TSP? command.

NOTES

The analog input values must be scaled to suitable position values. See "Scaling the Analog Input" (p. 31) for more information.

Make sure that internally, the analog input line is not connected to an axis for control value generation. With the E-753, this means that the value of the ADC Channel For Target parameter (ID 0x06000500) must not be 2. In NanoCapture™, this parameter is available in the *Target Manipulation* parameter group in the *Device Parameter Configuration* window.

2.8.4 Use as Control Value Generation Source

To enable the analog control input for an axis, the corresponding input signal channel must be connected to that axis. This is done with the ADC Channel For Target parameter (ID 0x06000500). If the connection of axis and input signal channel is saved as the power-on default, the axis can be commanded via analog input immediately after controller start-up, and no host PC is required. With the E-753, the parameter 0x06000500 must be set to 2 (input signal channel 2) for axis 1. Send:
SPA 1 0x06000500 2
to enable the connection in volatile memory. Note that in NanoCapture™, this parameter is available in the *Target Manipulation* parameter group in the *Device Parameter Configuration* window.

When the analog control input is enabled for an axis, then it overwrites the values of all other control sources for that axis except those from the AutoZero procedure. The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished. See "Control Value Generation" (p. 51) for more information.
An offset value can be added to the analog input scaled value for an axis using the AOS command (p. 114).

When no input signal channel is connected to an axis (i.e. the value of the ADC Channel For Target parameter is 0), the analog control input is disabled for that axis (including the offset set with AOS).

When the analog input is used as control source and the axis motion is stopped with STP (p. 160) or #24 (p. 113), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis. See the description above.

When the analog input is being used as control source and servo is switched off, the axis motion will continue in open-loop mode.

NOTES

The analog input values must be scaled suitable position values. See "Scaling the Analog Input" (p. 31) for more information.

Make sure that in the InputSignalChannel-to-Axis matrix, the coefficient of the analog input is set to zero. With the E-753, this is the value of the Position From Sensor 2 parameter (ID 0x07000501). In NanoCapture™, this parameter is available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window.

2.8.5 Analog-Input-Related Commands and Parameters

Command	Description	Notes
AOS (p. 114)	Set Analog Input Offset	Adds an offset value to the analog input scaled value (Analog Target Offset, ID 0x06000501). This offset is active as long as the analog input is enabled as control source for this axis.
AOS? (p. 116)	Get Analog Input Offset	Reads the current value of Analog Target Offset, parameter ID 0x06000501, from volatile memory
SEP (p. 152)	Set Nonvolatile Memory Parameters	Can be used to set the power-on default configuration for analog input usage.
SEP? (p. 153)	Get Nonvolatile Memory Parameters	Reads the current parameter values from non-volatile memory
SPA (p. 154)	Set Temporary Memory Parameters	Can be used to set a temporary configuration for analog input usage.
SPA? (p. 157)	Get Temporary Memory Parameters	Reads the current parameter values from volatile memory (RAM)
TAD? (p. 165)	Get ADC Value Of Input Signal	Reports the current ADC value of the analog input, dimensionless

TNS? (p. 167)	Get Normalized Input Signal Value	Reports the resulting value for the analog input after the electronics linearization, dimensionless
TSP? (p. 168)	Get Input Signal Position Value	Reports the resulting value for the analog input after the mechanics linearization (scaling), the unit is µm
WPA (p. 186)	Save Parameters To Nonvolatile Memory	Can be used to save the currently active configuration (including analog input usage) to non-volatile memory, where it becomes the power-on default.

See "How to work with the Analog Input" (p. 29) for more information. For detailed command descriptions see "Command Reference" (p. 112). For the identifiers of the items which can be addressed with the commands see "Accessible Items and Their Identifiers" (p. 43).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x02000200	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 1 (Offset)
0x02000300	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 2 (Gain)
0x05000000	1	Input Signal Channel	2	INT	Digital Filter Type
0x05000001	1	Input Signal Channel	2	FLOAT	Digital Filter Bandwidth
0x05000002	1	Input Signal Channel	2	INT	Digital Filter Order
0x06000500	1	Logical Axis	1	INT	ADC Channel for Target; if 0, then the analog control input is disabled for the axis
0x06000501	1	Logical Axis	1	FLOAT	Analog Target Offset
0x07000501	1	Logical Axis	1	FLOAT	Position from Sensor 2
0x0e000b00	3	System	1	INT	Number of input channels
0x0e000b03	3	System	1	INT	Number of sensor channels

See "Controller Parameters" (p. 203) for more information regarding the controller parameters and their handling.

2.9 Using Trigger Input and Output

It is possible to trigger external devices and to send start/stop signals to the wave generator with the digital I/O lines of the E-753. See "I/O" socket (p. 222) for the lines and pinout.

You can program the OUT2 output line (TTL, active high) to trigger other devices using CTO (p. 122). If CTO is used in combination with TWS (p. 169), the trigger output will be synchronized with the wave generator output. See "Trigger Output Synchronized with Wave Generator" (p. 81) for an example.
The OUT1 output line carries the servo cycle pulses and is not accessible by command.

The IN1 and IN2 input lines (TTL, active high) can be used in conjunction with the WGO command (p. 179) to trigger the wave generator output (IN1 and IN2) and to stop it (IN2). See "Wave Generator Started by Trigger Input" (p. 82) for an example.

2.10 Synchronization of Multiple Controllers

If multiple E-753s are used, their sensor signals can be synchronized.

To synchronize *n+1* E-753s, you need *n* special synchronization cables which can be obtained from PI (order# K040B0107, see "Additional Components" (p. 9)). Connect the Sync Out socket of the first device to the Sync In socket of the second device, the Sync Out socket of the second device to the Sync In socket of the third device and so on (see "Sync Out Socket" (p. 224) and "Sync In Socket" (p. 223) for pinout and specifications).

Each connector of a synchronization cable matches to only one of the sockets on the E-753, so do not apply force when connecting them.

2.11 Updates

2.11.1 Software 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 the software download. This password is provided on the E-753 CD in the E-753 Releasenews PDF file 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 are provided in the E-753 Releasenews xxxxx.pdf on the E-753 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-753*
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 (includes the manual versions that were with the release)

2.11.2 Firmware Updates

The current firmware revision of your E-753 can be identified in the answer of the *IDN? (p. 113) command or via the values of the following parameters:

Firmware Name, ID 0xFFFF0007
Firmware Version, ID 0xFFFF0008
Date Of Firmware, ID 0xFFFF000E

Query with the SPA? (p. 157) or SEP? (p. 153) command.

Firmware updates can be made by running the Firmware Update Wizard on the host computer, using the same communication interfaces as for normal operation. When updated firmware becomes available, it will be accompanied by the Firmware Update Wizard and update instructions.

The Firmware Update Wizard guides you through the update of the E-753 firmware.

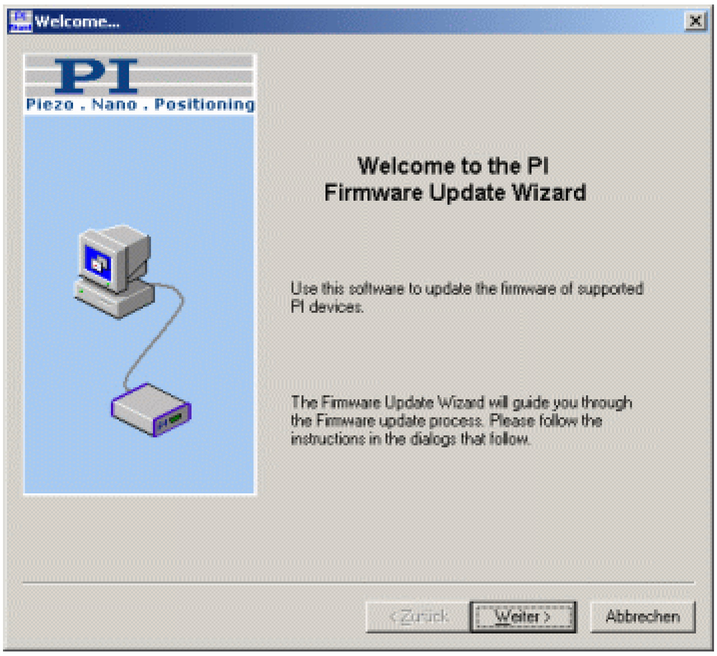


Figure 6: The Firmware Update Wizard Start Screen

3 System Description

3.1 Basic Elements

For successful operation of the E-753, you should familiarize yourself with the following features of the device.

Logical Axes:

Digital piezo controllers from PI control logical axes. Multiple piezo amplifiers ("output signal channels") and multiple sensors ("input signal channels") can be involved in the motion of one logical axis, just as one amplifier or one sensor can participate in the motion of more than one logical axis. See "Accessible Items and Their Identifiers" (p. 43) and "Processing Steps" (p. 45) for more information.

The E-753 controls one logical axis of a mechanics.

Input and Output Signals:

Most of the input and output signals provided by the E-753 are used as interface to the connected mechanics (sensor signals and amplifier output). In addition, some signals can be used as control input or for triggering purposes. See "Accessible Items and Their Identifiers" (p. 43) for details.

Communication Interfaces:

The E-753 can be controlled from a host computer (not included) with ASCII commands sent via:

- TCP/IP
- RS-232 serial connection

Both interfaces can be active simultaneously. The commands from the interfaces are queued in the order the completed command lines are received.
See "Communication" (p. 57) for more information.

Controller Firmware:

The firmware comprises the ASCII command set and the controller parameters and also includes some special features. For version information and updates see "Firmware Update" (p. 39).

- ASCII Commands:

  The E-753 understands the PI General Command Set (GCS; version 2.0).
  The PI General Command Set (GCS) is supported by a wide range of

PI systems. This command set is well-suited for positioning tasks with one or more axes. The command set itself is independent of the specific hardware (controller or attached stages).

Commands are used, for example, to set operating modes, to initiate motion of the mechanics and to query system and motion values. See "GCS Commands" (p. 106) for more information.

- Controller Parameters:

  The key features of the controller are mirrored in parameters. They represent the hardware basics and the calibration setup of the system. Some of the parameters are protected so that their factory settings can not be changed, other parameters can be modified by the user to adapt the system to the individual application. See "Controller Parameters" (p. 203) for more information.

  The piezo stage which is connected to the E-753 may contain an ID-chip (located in the stage connector). This ID-chip holds selected parameters which will be written to non-volatile and volatile memory when the stage is connected for the first time. See "ID-Chip Support / Stage Replacement" (p. 95) for more information.

  Note that PI records data files of every E-753 controller calibrated at the factory for easy restoration of original parameter settings after shipping.

- Command Levels:

  "Command levels" determine the availability of commands and the write access to the controller parameters. Changing the current active command level may require a password and can be done with the CCL command (p. 119).

- Special Features:

  Wave generator: Each axis can be controlled by a "wave generator" which outputs user-specified patterns, so-called "waveforms". This feature is especially important in dynamic applications which require periodic, synchronous motion of the axes. See "Wave Generator" (p. 70) for more information.

  Data recorder: The E-753 comprises a real-time data recorder. It is able to record several input and output signals (e.g. current position, sensor input, output voltage) from different data sources (e.g. logical axes or input and output signal channels). See "Data Recording" (p. 66) for more information.

- Control algorithm for Closed-Loop Operation:

For better position accuracy and performance, the E-753 can be operated in closed-loop mode. A proportional-integral (P-I) servo-control algorithm (with sensor feedback) will then apply corrections to the control value. See "Control Value Generation" (p. 51) and "Servo-Controller Dynamic Calibration" (p. 97) for more information.

In addition to the P-I controller, the Dynamic Digital Linearization (DDL) feature can be used for dynamic applications calling for periodic motion (wave generator operation). The Dynamic Digital Linearization (DDL) feature must be expressly ordered. See "Dynamic Digital Linearization (DDL)" (p. 85) and "Wave Generator" (p. 70) for more information.

Software on Host PC

Usually, a host computer is used to operate or at least configure the E-753. Therefore an ample array of software tools for installation on the host computer comes with the E-753. For a complete list of all software on the E-753 CD, see "Software Description" (p. 11).

3.2 Accessible Items and Their Identifiers

The identifiers listed below are used to address the appropriate items with the commands of the PI General Command Set (GCS) which is supported by the firmware of the E-753. They are factory defaults and can not be changed by the user.

- Logical axis: one axis, the identifier is 1.
  A logical axis is an axis of a linear, orthogonal coordinate system and represents a basic direction of motion in the E-753 firmware. All motion of the mechanics is commanded for logical axes.
  The number of axes is given by the Number Of System Axes parameter, ID 0x0E000B02.
  See "Processing Steps" (p. 45) for more information regarding the interrelation of logical axes and input / output signal channels.

- Input signal channels: two channels, the identifiers are 1 and 2.
  1 identifies the line for the capacitive sensor integrated in the mechanics (connects to the "PZT & Sensor" (p. 221) socket).
  2 identifies the analog input line which connects to the "Analog In" (p. 222) socket. This line can be used for an external sensor or as a source for control value generation (see "Using the Analog Input" (p. 29) for more information).
  The number of input signal channels is given by the Number Of Input Channels parameter, ID 0x0E000B00. The Number Of Sensor Channels parameter, ID 0x0E000B03, gives the number of

(capacitive) sensors which are directly integrated in the mechanics (i.e. line 1 with the E-753) and hence is always less than or equal to the number of input signal channels.

- Output signal channel: one channel, the identifiers is 1.
  This line carries the voltage for the piezo actuator in the mechanics (connects to the "PZT & Sensor" (p. 221) socket); the voltage is provided by the piezo amplifier of the E-753.
  The number of output signal channels is given by the Number Of Output Channels parameter, ID 0x0E000B01. The Number Of Piezo Channels parameter, ID 0x0E000B04, is always less than or equal to the number of output signal channels and gives the number of piezo voltage amplifiers dedicated to the actuators in the mechanics. For the E-753, the values of both parameters are identical because there are no output signal channels other than that for the piezo voltage of the mechanics.

- Digital output lines: two lines (OUT1 and OUT2, for pinout see "I/O Socket" (p. 222)), the identifier for OUT2 is 1; OUT1 is not accessible by commands
  The number of digital output lines that are accessible by commands is given by the Number Of Trigger Outputs parameter, ID 0x0E000B05.
  See "Using Trigger Input and Output" (p. 38) for more information.

- Digital input lines: two lines (IN1 and IN2, for pinout see "I/O Socket" (p. 222)). Though they are used to trigger the wave generator output (WGO (p. 179) command), they are not directly accessible by command and therefore have no identifiers.
  See "Using Trigger Input and Output" (p. 38) for more information.

- Wave generator: one generator, the identifier is 1
  The number of wave generators is the same as the number of logical axes, and each wave generator is dedicated to one axis.
  See "Wave Generator" (p. 70) for more information.

- Wave tables (memory tables for waveform data): 10 tables with a total of 65536 points, the identifiers are 1 to 10
  The number of wave tables is given by the Number of Waves parameter, ID 0x1300010A. See "Wave Generator" (p. 70) for more information.

- DDL table (memory table for the data of the Dynamic Digital Linearization (DDL) feature): one table with 65536 points, identifier is 1
  The number of DDL tables is the same as the number of logical axes, and each DDL table is dedicated to one axis.
  See "Dynamic Digital Linearization (DDL)" (p. 85) for more information.

- Data recorder tables (memory tables for recorded data): up to 8 tables with a total of 65536 points, the identifiers start with 1 and continue sequentially up to the number of tables.
  The number of data recorder tables can be set via the Data Recorder Chan Number parameter, ID 0x16000300. The maximum number of tables is limited by the Max Number Of Data Recorder Channels parameter, ID 0x16000100.
  See "Data Recording" (p. 66) for more information.

- Whole system: the E-753 as a whole, the identifier is 1

- Hardware modules: three internal modules of the E-753 (controller board, Ethernet board, sensor board), identifiers are 1 to 3

3.3 Processing Steps

The E-753 controls the motion of one logical axis of a mechanics. The control value can be given by multiple sources. Depending on the operating mode, a control loop with feedback of up to two sensors can be used to maintain the axis position. The control value is then transformed to the input voltage for the piezo amplifier, whose output drives the actuator in the mechanics.

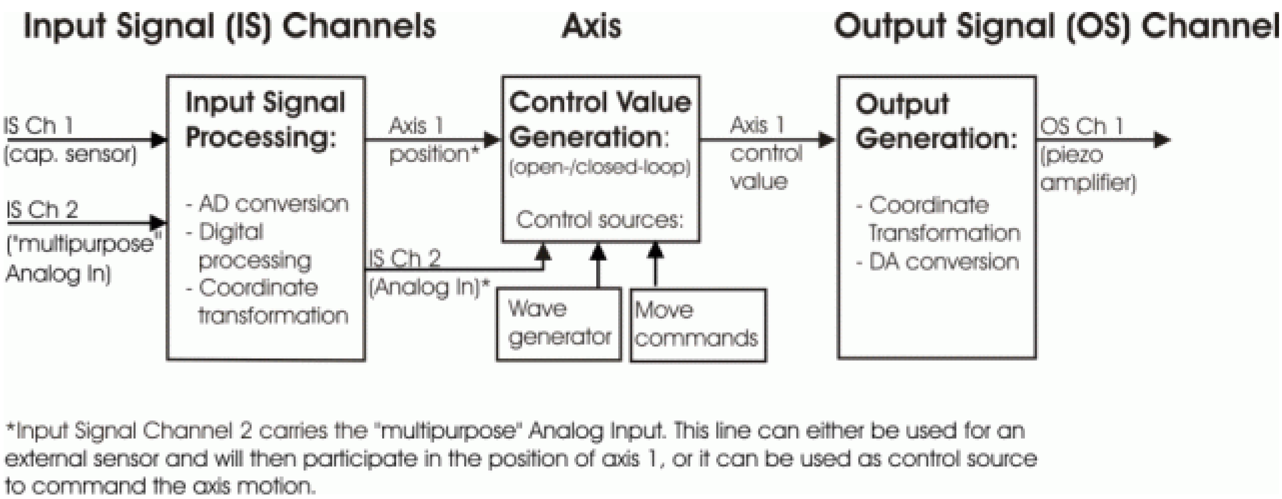


Figure 7: Processing Overview for E-753

The individual processing steps and the corresponding controller parameters are described in the subsections below.

NOTES

Wherever changing parameter values is mentioned, you can do this using SPA (p. 154) (volatile memory) or SEP (p. 152) (non-volatile memory). Furthermore, you can use WPA (p. 186) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. To have write access to certain parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119). To read parameter values, query with the SPA? (p. 157) or SEP? (p. 153) commands.

The NanoCapture™ host software gives access to parameter values in a more convenient way. Use its *Device Parameter Configuration* window to check/edit the individual parameters. See the NanoCapture™ manual for more information.

3.3.1 Input Signal Processing

The E-753 provides two channels for analog input signals—the first channel is for the signal from the capacitive position sensor in the mechanics, the second channels carries a "multipurpose" analog input. The following processing is applied to both input signal channels:

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

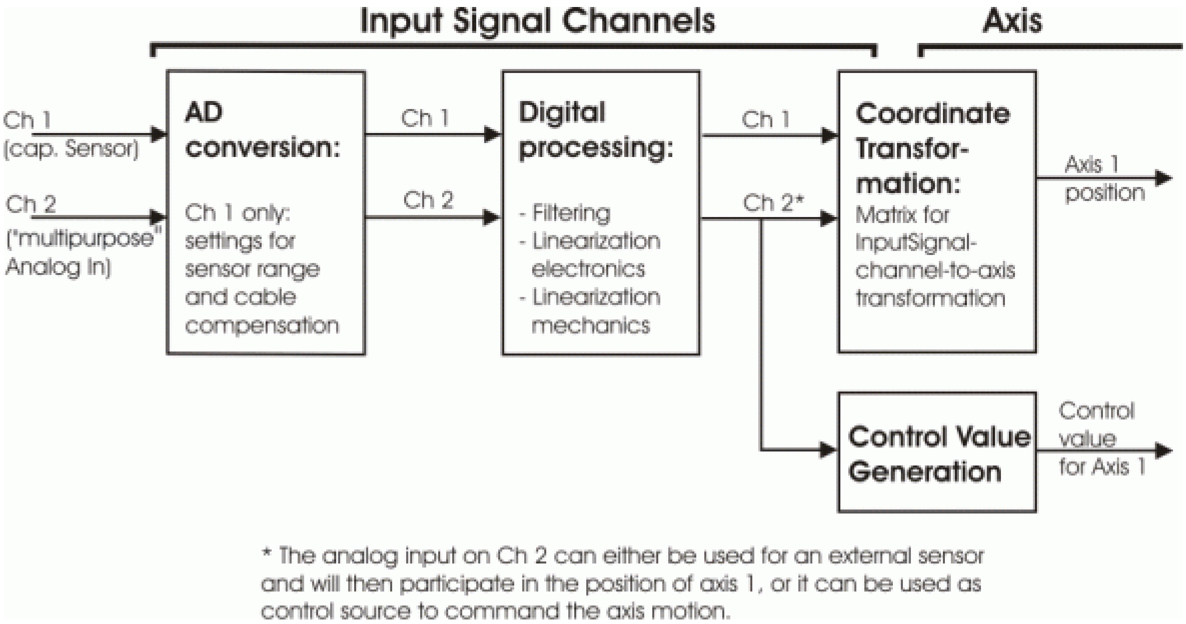


Figure 8: E-753 Input signal processing

Analog to digital conversion

The capacitive sensor in the mechanics (input signal channel 1) and the associated analog circuitry convert the mechanical distance change to an analog voltage change. The analog voltage is proportional to the distance change. The following settings influence the analog sensor signal and can be selected by parameters:

- Sensor Range
  parameter ID 0x02000100
  Scales the standard sensor range (by changing the capacitance defined as the mid-position value). Example: a 100 µm sensor can also be used over a measurement range of 125 µm with a factor of 1.25 or over a 300 µm range with a factor of 3. Possible values:
  1 = 3.00x
  2 = 2.13x
  3 = 1.25x
  4 = 1.00x
  This parameter also selects automatically the corresponding polynomial for electronics linearization (see "Digital processing" (p. 48)).

- Sensor Cable Compensation,
  parameter ID 0x02000103
  When set to zero, cable compensation is disabled. Otherwise the cable compensation is enabled and the electronics polynomial for cable compensation is selected. The value is determined at the factory and should only be changed when an additional extension cable or adapter is used, or when the length of the stage cable is changed subsequent to the calibration at the factory. In this case, contact your Physik Instrumente Sales Engineer or write info@pi.ws to obtain a suitable value.

For the multipurpose analog input (input signal channel 2), the sensor range and cable compensation parameters are irrelevant.

Note that in NanoCapture™, the parameters are available in the *Sensor Electronics* parameter groups in the *Device Parameter Configuration* window.

The result of the analog to digital conversion can be queried with the TAD? command (p. 165) for both channels.

Digital processing

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

- Digital filtering
- Electronics linearization
- Mechanics linearization

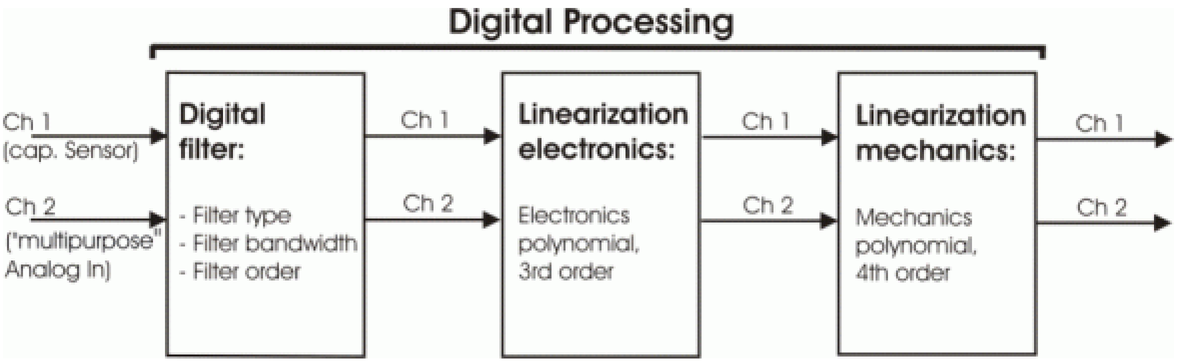


Figure 9: E-753 Digital processing of the input signals

The following parameters determine the digital filter settings:

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

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

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

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

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

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

x – *filtered sensor ADC value*

y – *linearized sensor value*

To make the system components easily replaceable, sensor (i.e. mechanics) and electronics use separate polynomials. The coefficients of the polynomials are determined at the factory. Some terms of the polynomials are provided for future application and presently set to zero. The following terms are currently in use:

- Electronics linearization: offset, gain, 2nd and 3rd order correction. The corresponding coefficients of the polynomial are given by the parameters 0x03000100 to 0x03000400. They are independent of the connected mechanics and may not be changed by the user. In NanoCapture™, these parameters are available in the *Sensor Electronics* parameter groups in the *Device Parameter Configuration* window.

- Mechanics linearization: offset, gain, 2nd, 3rd and 4th order correction. The corresponding coefficients of the polynomial are given by the parameters 0x02000200 to 0x02000600. They depend on the

connected mechanics. In NanoCapture™, these parameters are available in the *Sensor Mechanics* parameter groups in the *Device Parameter Configuration* window.
For the capacitive sensor in the mechanics (input signal channel 1), the parameters should not be changed by the user. For the analog input line (input signal channel 2), changing the offset and gain values is required to scale the analog input to suitable position values (see "Using the Analog Input" (p. 29) for more information and examples).
If the connected mechanics has an ID-chip, the coefficients will be read in from the ID-chip (see "ID-Chip Support / Stage Replacement" (p. 95) for more information).

The TNS? command (p. 167) reports the result after the linearization for the electronics (normalized value, dimensionless), while the TSP? command (p. 168) reports the result after the linearization for the mechanics (scaled value, the unit is µm).

Coordinate transformation

Up to 2 sensors can be used to monitor the position of axis 1: the capacitive sensor in the mechanics (input signal channel 1) is always active, while an additional, external sensor can optionally be connected to the Analog In line (input signal channel 2). The axis position is calculated from the position values of the two input signal channels using a coordinate transformation matrix which has 1 row and 2 columns:

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

In equation form:

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

The important matrix coefficients are given by the following parameters:

- a_{11} is given by Position From Sensor 1, parameter ID 0x07000500
  This coefficient is for the capacitive sensor in the mechanics. It is determined during calibration at the factory and has a non-zero value. The preset value of the coefficient should not be changed unless the capacitive sensor is to be excluded from the position feedback of the axis, e.g. if an external sensor is connected to the analog input line (see "Using the Analog Input" (p. 29) for more information).

- a_{12} is given by Position From Sensor 2, parameter ID 0x07000501
  This coefficient is for the Analog In line (input signal channel 2) and should therefore be set to zero as long as no external sensor is connected to the Analog In socket or when the analog input is used for

control value generation (see "Using the Analog Input" (p. 29) for more information).

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

The coefficients of the transformation matrix are given by a calibration procedure before shipment. If the connected mechanics has an ID-chip, the coefficients will be read in from the ID-chip (see "ID-Chip Support / Stage Replacement" (p. 95) for more information). If necessary, the parameter for the Analog In line can also be changed by the user (e.g. when an external sensor is to be connected to Analog In).

While TSP? (p. 168) reports the position values of the input signal channels, the POS? command (p. 148) reports the axis position after the matrix transformation (the unit is µm).

3.3.2 Control Value Generation

The control value for the axis motion can result from multiple sources (see below). Furthermore, the feedback from up to two sensors can be used to maintain the axis position, depending on the current operating mode. The interpretation of the control value depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). By default, the matrix is set up so that control values correspond numerically to axis position values.

The E-753 provides the following operating modes:

- Open-loop control (also referred to as "servo-off state" in this document): sensor feedback is not used
- Closed-loop control (also referred to "servo-on state" in this document): sensor feedback participates in the control value generation.
  A proportional-integral (P-I) servo-controller is used to generate corrections to the control value. In addition, the two notch filters integrated in the E-753 are used, and the velocity is limited by a special slew rate setting. The settings for the PI controller, the notch filters and the slew rate are accessible as parameters. See "Servo Controller Dynamic Calibration" and "Controller Parameters" (p. 203) for more information.

The operating mode can be selected with the SVO command (p. 163). By default, open-loop control is active after power-on. Using the Power Up Servo On Enable parameter (ID 0x07000800), you can set up the device to start with closed-loop control.

The control sources write their values to the open-loop control register or to the target register, depending on the current operating mode (see also the figures below). The different control sources have different priorities, i.e. some sources can overwrite the control values given by other sources. The E-753 supports the following control sources:

- SVO (p. 163) command: To prevent jumps of the mechanics, writing is done as follows when the operating mode is changed with the SVO command:
  Switching from open-loop to closed-loop control writes the current axis position to the target register.
  Switching from closed-loop to open-loop control writes the current (closed-loop) control value to the open-loop control register.
  This control source has the lowest priority and can be overwritten by all other sources.

- Move commands (MOV (p. 144) and MVR (p. 145) in closed-loop operation; SVA (p. 160) and SVR (p. 164) in open-loop operation): Write new values to the corresponding register when processed for the axis. The move commands can be overwritten by all other sources.

- IMP (p. 143) and STE (p. 158) commands: generate an impulse or step response. Only relative values are written to the appropriate register. Relative values are added to the appropriate register. Is overwritten by wave generator, analog input and the AutoZero procedure.

- Wave Generator: When the wave generator is enabled with WGO (p. 179), its output is written to the appropriate register. A wave generator overwrites all other control sources except the analog input and the AutoZero procedure.
  An offset value can be added to the wave generator output using the WOS command (p. 184).

- Analog Input: When the analog control input is enabled for the axis then it writes to the appropriate register. It overwrites the values of all other sources except those from the AutoZero procedure.
  An offset value can be added to the analog input scaled value using the AOS command (p. 114).

  Note: To enable the analog control input for an axis, an input signal channel must be connected to that axis. This is done via the ADC Channel For Target parameter (ID 0x06000500), which can be set by the SPA (p. 154) or SEP (p. 152) command. In addition, the InputSignalChannel-to-Axis matrix coefficient for the selected input signal channel must be set to zero. When no input signal channel is connected to the axis, the analog control input is disabled (including the offset set with AOS).
  When the analog input is used as control source and the axis motion is

stopped with STP (p. 160) or #24 (p. 113), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis.
See "How to work with the Analog Input" (p. 29) and "Input Signal Processing" (p. 46) for more information.

- AutoZero procedure: this calibration procedure is started by the ATZ command (p. 117) and performed in open-loop operation only (if servo is on, it will be switched off automatically during the AutoZero procedure and on again afterwards). The AutoZero procedure has the highest priority, i.e. it will overwrite the control values given by all other sources. When the analog control input is enabled, it will be disabled automatically at the start of the AutoZero procedure and reenabled again when AutoZero is finished.   See "AutoZero Procedure" (p. 23) for details.

Open-Loop Control

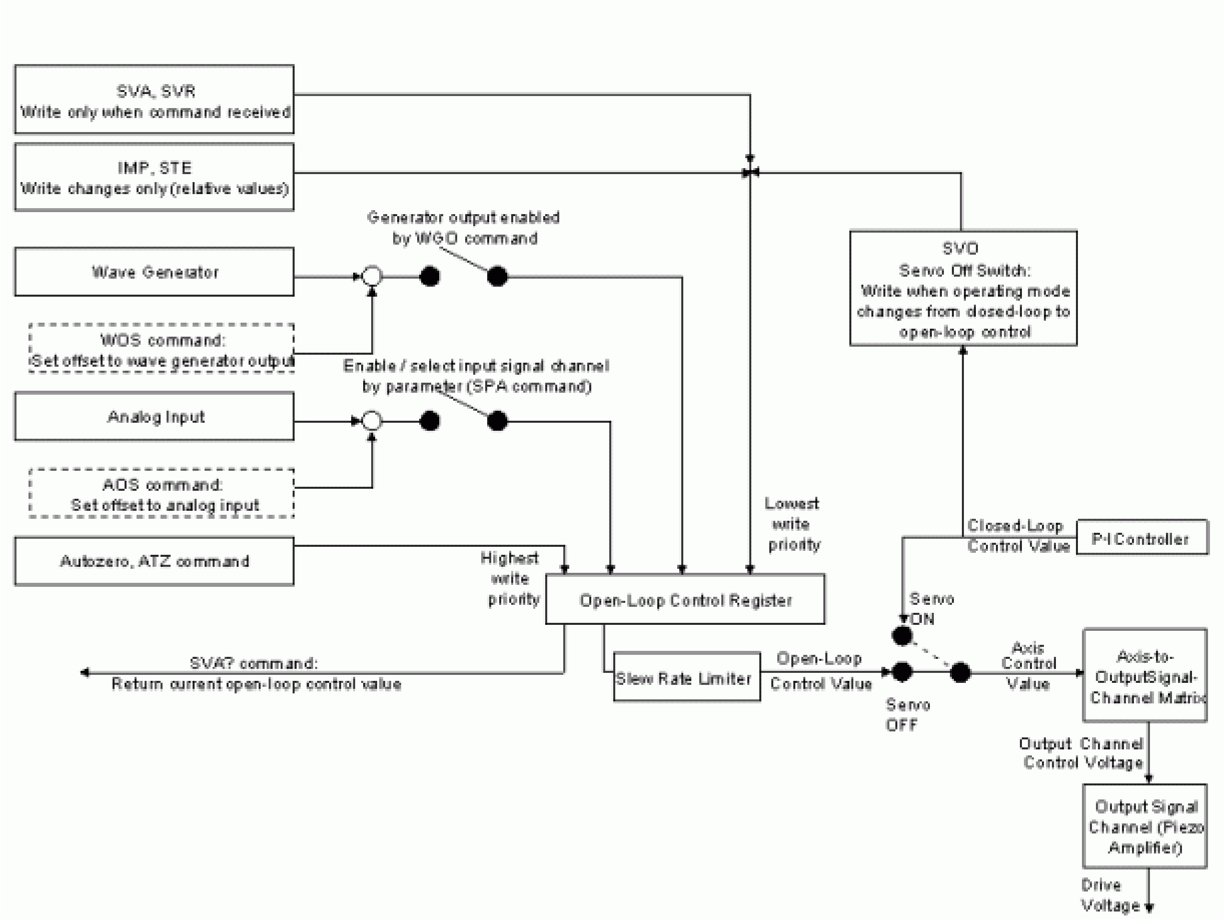


Figure 10: Control sources in open-loop operation

Closed-Loop Control

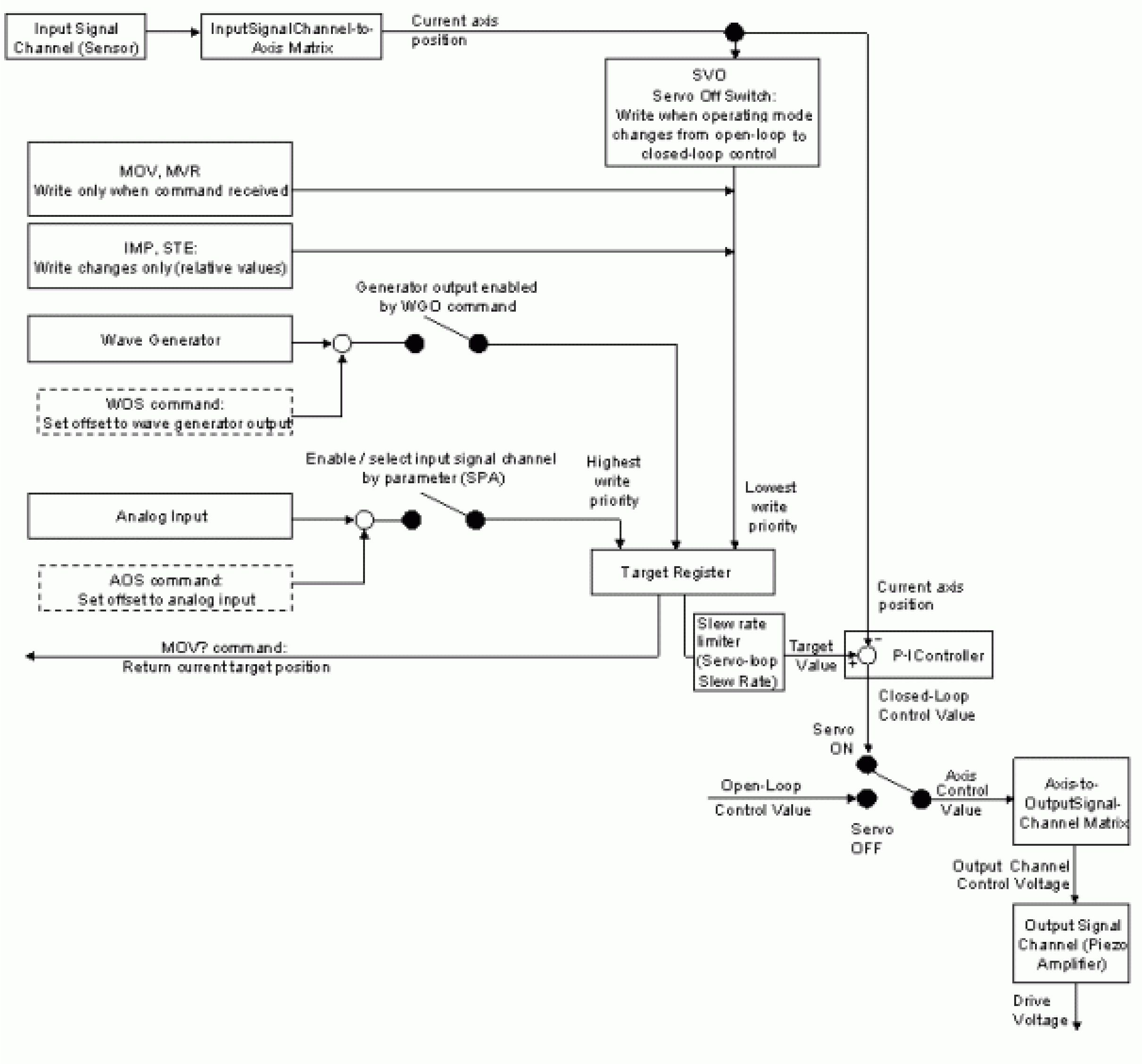


Figure 11: Control sources in closed-loop operation

3.3.3 Output Generation

The control value for the axis is transformed to a control voltage value for the output channel. After the digital-to-analog conversion, this value is sent to the piezo amplifier whose output drives the actuator in the mechanics.
The transformation of the axis control value to the control voltage value for the output channel is done via the Axis-to-OutputSignalChannel matrix.

Because the E-753 controls only one logical axis and is equipped with only one output signal channel (piezo amplifier), this matrix has 1 row and 1 column:

$OutputCh_1 = p_{11}Axis_1$

The p_{11} coefficient is given by Driving Factor Of Piezo, parameter ID 0x09000000. In NanoCapture™, this parameter is available in the *Axis Definition* parameter group in the *Device Parameter Configuration* window. During calibration at the factory, this coefficient is set numerically to the number of volts per axis unit for the attached piezo actuator. Thus both the closed-loop control value and the open-loop control value correspond numerically to axis position values. This means that all control sources always command with axis position values, irrespective of the current operating mode (p. 51).
If the connected mechanics has an ID-chip, the p_{11} coefficient will be read in from the ID-chip (see "ID-Chip Support / Stage Replacement" (p. 95) for more information). The parameter should not be changed by the user.

Note: With multi-axis systems which are equipped with multiple amplifiers, the elements of the Axis-to-OutputSignalChannel matrix would also be used to specify the contribution of each output signal channel to the motion of the various logical axes.

The VOL? command (p. 171) reports the drive voltage output of the output signal channel (in volts).

4 Communication

4.1 Interfaces Available

The E-753 can be controlled from a host computer (not included) with ASCII commands sent via:

- TCP/IP
- RS-232 serial connection

Both interfaces can be active simultaneously. The commands from the interfaces are queued in the order the completed command lines are received.

NOTES

Before communication between E-753 and host PC can be established, the power-on sequence of the controller must have finished successfully (takes approx. 20 s, after which the Power LED blinks continuously at regular intervals). TCP/IP communication: If no DHCP server is available on the network or if a point-to-point connection between host PC and controller is being used, it might take another period of about 30 seconds before communication is possible.

It is not possible to connect multiple TCP/IP command streams to the E-753 via TCP/IP—only one TCP/IP connection is possible at any one time.

Using multiple interfaces simultaneously may cause problems with the host software.

It is also possible to set up the E-753 so that it can be commanded directly by an analog input signal. See "Using the Analog Input" (p. 29) for more information.

With PIMikroMove®, it is possible to connect to multiple controllers in one program instance. With NanoCapture™ and PITerminal, you need a new program instance for each E-753 you want to connect to.

4.2 Default and Current Settings

The default communication parameters are stored on the E-753. You can read the default settings using the IFS? command (p. 142) and change them with IFS (p. 140). Changes become active with the next power-on, when the default values are loaded into the controllers RAM.

The currently active baud rate for RS-232 serial connections can be temporarily changed with IFC (p. 138). The new setting becomes active immediately and the host PC interface configuration may need to be changed to maintain communication. When the controller is powered down, the baud rate setting made with IFC is lost if it was not saved with WPA (p. 186). To read all current active communication parameters use the IFC? command (p. 138).

NOTE

The IFS, IFS?, IFC and IFC? commands affect the controller side only.

It is also possible to change the default settings with SEP (p. 152) and to read them with the SEP? command (p. 153), but do not activate them with RPA (p. 149) or change the current settings with SPA (p. 154)—except of the baud rate—because it will not be possible to maintain communication afterwards. The appropriate parameter IDs are given below.

The factory defaults of the communication settings stored in the controller are as follows (response to IFS?):

- RSBAUD: gives the baud rate to be used for RS-232 communication, default is 115200
  also accessible as Uart Baudrate parameter, ID 0x11000400

- IPADR: the first four portions specify the default IP address for TCP/IP communication, the last portion specifies the default port to be used, default is 192:168.0.1:50000
  also accessible as IP Address parameter, ID 0x11000600
  Note: While the IP address can be changed, the port must always be 50000!

- IPSTART: defines the startup behavior for configuration of the IP address for TCP/IP communication,
  default is 1 = use DHCP to obtain IP address, if this fails, use IPADR
  also accessible as IP Configuration parameter, ID 0x11000800

- IPMASK: gives the IP mask to be used for TCP/IP communication, default is 255.255.255.0
  also accessible as IP Mask parameter, ID 0x11000700

- MACADR: is the unique address of the network hardware in the E-753, read-only, example: 0:80:194:101:84:1
  also accessible as MAC Address parameter, ID 0x11000B00

NOTES

For communication via TCP/IP, the current used IP address and the startup behaviour partially depend on the network type. For that reason, the settings for IPADR and IPSTART may be ignored:

If a DHCP server is present in the network, the IPSTART setting is ignored and the IP address is always obtained from the DHCP server.

If the E-753 is directly connected to the Ethernet card in the PC (no DHCP server is present), the current IP address of the E-753 will be as follows:
for IPSTART = 0, the IPADR setting will be used
for IPSTART = 1, the default value 192.168.0.1 will be used.

4.3 TCP/IP Connection

The TCP/IP connection is available on the rear panel of the controller, via the RJ 45 socket with the network icon. The access differs depending on the network availability, which can be as follows:

- Network with DHCP server
- PC equipped with an Ethernet card or network without DHCP server

For the default IP address, IP mask and startup behaviour settings of the TCP/IP interface see "Default and Current Settings" (p. 58).

!

CAUTION

If the communication between host PC and E-753 is done via TCP/IP, do not use RPA (p. 149) after you have changed the parameters of the TCP/IP communication with IFS (p. 140) or SEP (p. 152) in non-volatile memory, because it will not be possible to maintain communication afterwards.

NOTES

Make sure that your network administrator has not set the network to forbid unknown devices like the E-753 from logging on.

Only one TCP/IP connection is allowed at any one time: Presently, only one port (50000) is available at the E-753 so that only one application at a time can use the TCP/IP connection. The default port setting can not be changed.

4.3.1 Network with DHCP Server

If a network with DHCP server is available, use the straight-through network cable which comes with the controller. Connect the controller to a network access point and power cycle it. Factory default: The controller will automatically obtain an IP address over DHCP. If this should not be successful within 30 seconds, the default IP address of the controller will automatically be used (see "Default and Current Settings" (p. 58)).

In the host software (e.g. NanoCapture™, PIMikroMove®, PITerminal or LabView drivers), you can use the "Search for controllers" functionality in the connection dialog to see all available E-753 controllers with their IP address and port settings. In NanoCapture™ you have, for example, to press the *Search...* button in the *Device Connection* window (see figures below). In the resulting list, click on the controller to which you want to connect. In the connection dialog, check that the IP address and port number were correctly transferred to the *Hostname* and *Port* fields. Then press the *OK-Connect* button to establish the connection.

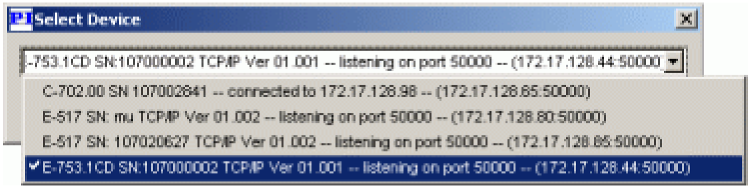


Figure 12: Search results in NanoCapture(TM)

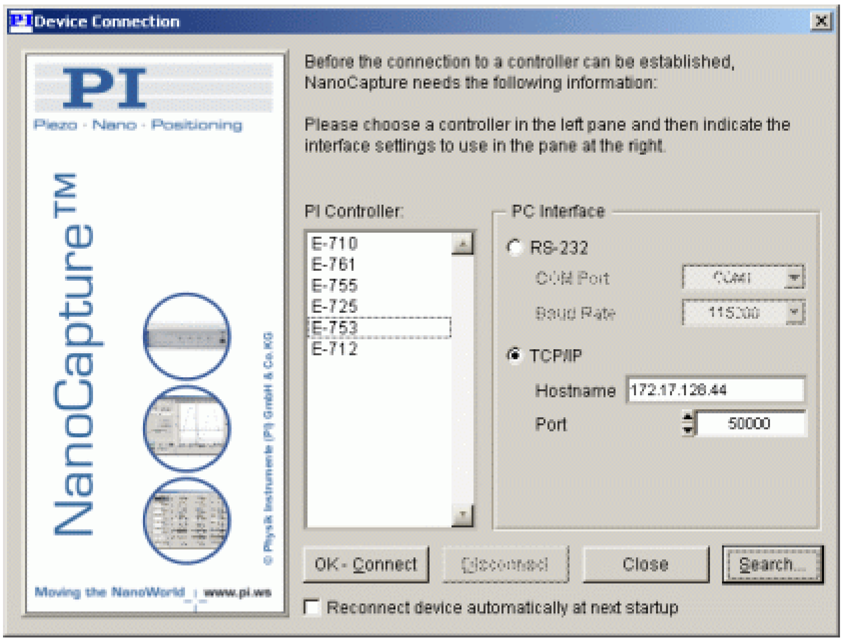


Figure 13: The connection dialog in NanoCapture(TM), press Search to search for controllers and OK-Connect to establish the connection to the selected controller

---

NOTE

In the controller selection list, you can also identify the controllers which already have a TCP/IP connection open. If you try to connect to such a controller, an error message will be generated as no multiple TCP/IP connections are possible.

4.3.2 PC with Ethernet Card

When your PC is equipped with a free Ethernet connection and no DHCP server is available, the IP address and IP mask settings of PC and controller must be compatible with each other. Otherwise no connection can be established. You can configure either the PC or the controller settings to be compatible. If you have a network with multiple E-753s, the settings of the individual controllers must be changed to have unique IP addresses for all devices in the network. See below for how to proceed.

NOTE

After the power-on / reboot sequence which takes about 20 seconds, you might have to wait for another period of about 30 seconds before communication is possible.

If you want to change the PC settings:

Configure the connection on the PC according to the IP address and IP mask settings of the controller (see "Default and Current Settings" (p. 58) for the default controller settings). Note that the following steps may vary in some details depending on the version of your Windows operating system:

1. Follow the menu sequence *Start* ⇒ *Settings* ⇒ *Control Panel* ⇒ *Network and Dial-up Connections* ⇒ *Local Area Connection*.

2. In the *Local Area Network (LAN)* status window, click *Properties*.

3. In the *Properties* window of the LAN connection, click on *Internet Protocol (TCP/IP)*.

4. In the *Internet Protocol (TCP/IP) Properties* window, activate *Use the following IP address.* Make a note of the current *IP address* and *Subnet mask* settings, if any, in case they need to be restored later. Then adapt the *IP address* and *Subnet mask* settings to make them compatible with the settings of your E-753:

   Set the first three portions of *IP address* identical to those of the IP address of the E-753, while the last portion must be different. One possible *IP address* setting would be, for example, 192.168.0.2 (the default IP address of the E-753 is 192.168.0.1). Do not use "255" for the last portion.

   Set *Subnet mask* to 255.255.255.0 (if the IP mask of the E-753 is 255.255.255.0).

   Confirm with *OK*. An example is shown in the figure below.

5. Connect the E-753 to the Ethernet card in the PC using the included, special, cross-over cable ("point-to-point" connection). If a hub is used to allow for connection of several devices to the PC, connect the controller to the hub using the straight-through network cable.

6. Power on the E-753.

7. Use the "Search for controllers" functionality, described in "Network with DHCP Server" (p. 60), to establish the connection between PC and the E-753.

NOTE

If the connection fails, change the last portion of the *IP address* setting on the PC and try again to connect via "Search for controllers".

Adapt *IP address* and *Subnet mask*; do not change the *Default gateway* setting

Press *OK*

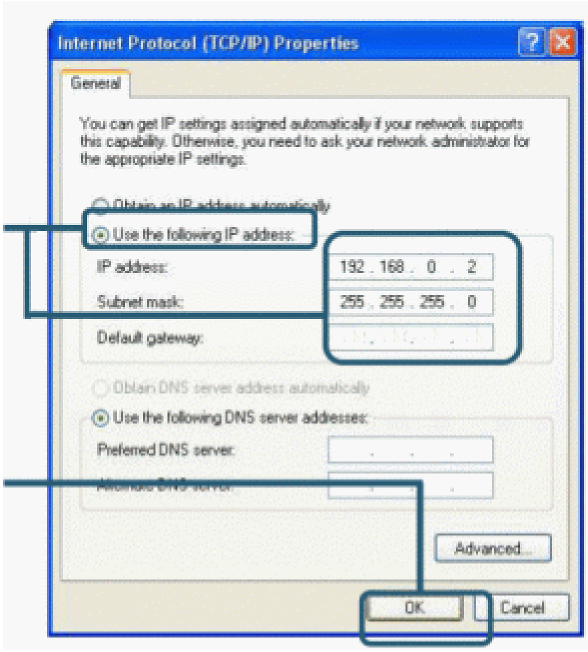


Figure 14: Internet Protocol (TCP/IP) Properties window, the settings shown are only examples, maybe they does not match that of your controller

If you want to change the controller settings:

1 Establish a serial connection between PC and E-753 as described in "RS-232 Serial Connection" (p. 64).

2 Use the IFS command (p. 140) in the command entry facility of the program to adapt the IP address and IP mask settings of the E-753 to those of the PC (to check the PC settings, you can open the *Internet Protocol (TCP/IP) Properties* window as described above):

To change the IP mask (subnet mask), send
IFS 100 IPMASK *mask*
mask must be identical to the Subnet mask setting of the PC.

To change the IP address, send
IFS 100 IPADR *address*

At least the last portion of the IP address must be different from that of the PC and any other device in the same network (the applicable address settings depend on the IP mask setting). If, for example, the PC has the IP address 172.21.0.1, send
IFS 100 IPADR 172.21.0.2:50000
Do not use "255", and do not change the port setting (must always be 50000).

3 Make sure that the correct startup behaviour for the IP address configuration is selected:

   Ask with the IFS? command. In the answer, IPSTART must be 0 (= use IP address defined with IPADR). If the IPSTART value differs from 0, send
   IFS 100 IPSTART 0

4 Close the connection.

5 Connect the E-753 to the Ethernet card in the PC using the included, special, cross-over cable ("point-to-point" connection). If you connect the E-753 to a free access point (e.g. to a hub) on a network to which the PC is connected, use the straight-through network cable.

6 Power-cycle the E-753.

7 Use the "Search for controllers" functionality, described in "Network with DHCP Server" (p. 60), to establish the connection between PC and the E-753.

4.4 RS-232 Serial Connection

The serial communications port is accessed via a sub-D 9m "RS-232" (p. 223) connector on the rear panel of the controller. Use the included null-modem cable to connect the controller to the host PC; if the PC has only one COM port, it is probably COM 1.

The serial port on the E-753 is preset to the following parameters:
115,200 baud, 8 bits, no parity, RTS/CTS.

In the connection dialog of the host software (e.g. NanoCapture™, PIMikroMove®, PITerminal or LabView drivers), you make the settings on the host PC side. Select the correct PC COM port and make sure that the baud rate (and other settings) in the dialog match those of the E-753. Otherwise no communication can be established.

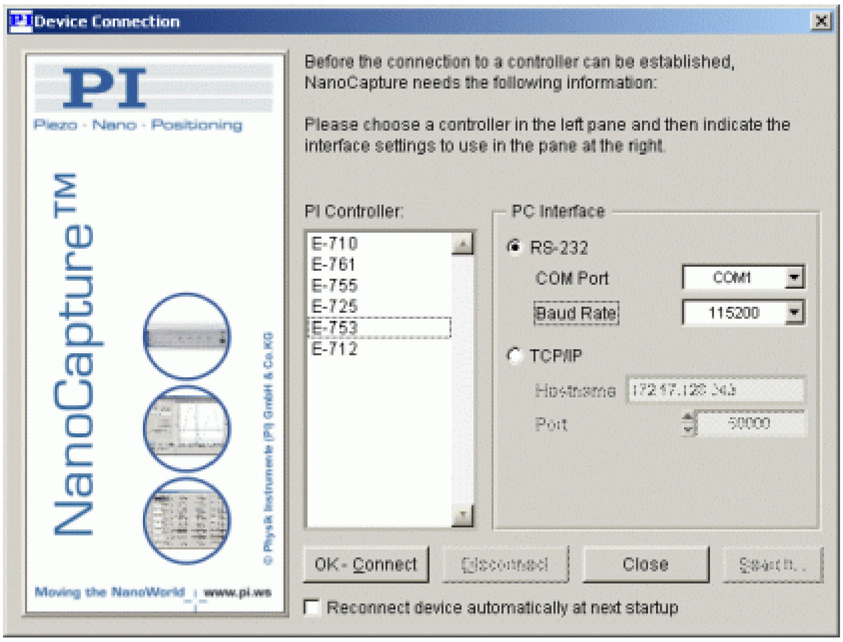


Figure 15: RS-232 configuration of the host PC side in NanoCapture(TM)

If you have established a connection and want to change the currently active baud rate, proceed as follows:

1. Use the IFC command (p. 138) in the command entry facility of the program, e.g. by sending IFC RSBAUD 57600.
2. Close the connection.
3. Open the connection again with the baud rate you just set with IFC (in the example 57600).

NOTE

It is recommended that the host PC have a "genuine" hardware RS-232 interface. If the host PC uses a USB-to-serial adapter instead, data loss could occur during communication, especially when transferring large amounts of data.

5 Data Recording

5.1 How to Use the Data Recorder

The E-753 includes a real-time data recorder. It is able to record several input and output signals (e.g. current position, sensor input, output voltage) from different data sources (e.g. controller axes or input and output channels). The gathered data is stored (temporarily) in "data recorder tables"—each table contains the signal from one data source. You can configure the data recorder flexibly, e.g. select the type of data and the data source. Furthermore, you can choose the number of record tables and hence influence their size.

Trigger Recording

Data recording is triggered automatically with four commands:

- STE (p. 158) (step response measurement),
- IMP (p. 143) (impulse response measurement),
- WGO (p. 179) (wave generator start) and
- WGR (p. 183) (restarts recording when the wave generator is running).

Recording always takes place for all data recorder tables and ends when the data recorder tables are completely filled.

Read Data

The last recorded data can be read with the DRR? command (p. 130). The data is reported in GCS array format. For details regarding GCS array see the separate manual (SM146E), which is provided on the E-753 CD. Reading out recorded data can take some time, depending on the number of points to be read! It is possible to read the data while recording is still in progress.

Configure Recording

For general information regarding the data recording you can send HDR? (p. 134), which lists available record options, and gives information about additional parameters and commands concerned with data recording.

The data recorder configuration, i.e. the assignment of data sources and record options to the recorder tables, can be changed with DRC (p. 128), and the current configuration can be read with DRC? (p. 130). The default setting is that the current position of axis 1 is recorded for all data recorder tables.

The number of available data recorder tables can be read with the TNR? (p. 166) command. The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. You can change the parameter value to increase or decrease the number of data recorder tables. For E-753, the number of tables must be in the range of 1 to 8.

The total number of points available for data recording is given by the Data Recorder Max Points parameter, ID 0x16000200. The controller allocates these points in equal shares to the available tables (i.e. to the number of tables given in the TNR? answer). For E-753, the total number of points is 65536. If, for example, TNR? replies 8, each table is comprised of 8192 points.

The data recorder sampling period can be read with the RTR? command (p. 151). The answer gives the value of the Data Recorder Table Rate parameter (ID 0x16000000) whose default value is one servo cycle. You can cover longer periods by increasing this value. Use the RTR command (p. 150) or change the parameter value directly.

Wherever changing parameter values is mentioned, you can do this using SPA (p. 154) (volatile memory) or SEP (p. 152) (non-volatile memory). Furthermore, you can use WPA (p. 186) to copy the current values from volatile memory to non-volatile memory, where they become the power-on defaults. To have write access to certain parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119). To read parameter values, query with the SPA? (p. 157) or SEP? (p. 153) commands.

When the controller is powered down, the contents of the data recorder tables, the configuration settings made with DRC (p. 128) and all settings which were only made in volatile memory are lost. On power on, all settings are reset to their power-on defaults.

5.2 Data-Recorder Related Commands and Parameters

Command	Description	Notes
DRC (p. 128)	Set Data Recorder Configuration	Assigns data sources and record options to data recorder tables. Settings will be lost on controller power down or reboot.
DRC? (p. 130)	Get Data Recorder Configuration	Reads current data recorder settings
DRR? (p. 130)	Get Recorded Data Values	Reading can take some time, depending on the number of points.
HDR? (p. 134)	Get All Data Recorder Options	Lists available record options, gives information about additional parameters and commands concerned with data recording
IMP (p. 143)	Start Impulse and Response Measurement	Triggers recording
RTR (p. 150)	Set Record Table Rate	Changes the data recorder table rate in volatile memory (Data Recorder Table Rate parameter, ID 0x16000000)
RTR? (p. 151)	Get Record Table Rate	Reads the current setting of the data recorder table rate (Data Recorder Table Rate parameter, ID 0x16000000)
STE (p. 158)	Start Step and Response Measurement	Triggers recording
TNR? (p. 166)	Get Number of Record Tables	Reads the number of available data recorder tables (Data Recorder Chan Number parameter, ID 0x16000300)
WGO (p. 179)	Set Wave Generator Start/Stop Mode	Triggers recording
WGR (p. 183)	Start Recording Synchronous to Wave Generator	Triggers recording

See "How to use the Data Recorder" (p. 66) for more information. For detailed command descriptions see "Command Reference" (p. 112). For the identifiers of the items which can be addressed with the commands see "Accessible Items and Their Identifiers" (p. 43).

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

See "Controller Parameters" (p. 203) for more information regarding the controller parameters and their handling.

6 Wave Generator

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

In "How to Work with the Wave Generator" (p. 70) and "Wave Generator Examples" (p. 75) you will learn how to use the wave generator, and "Wave-Generator-Related Commands and Parameters" (p. 83) gives an overview.

The Dynamic Digital Linearization (DDL) option can be used in conjunction with the wave generator output in addition to the "normal" servo algorithm. See "Dynamic Digital Linearization (DDL)" (p. 85) for more information.

During the wave generator output, data is recorded in "record tables" on the controller. See "Data Recording" (p. 66) for more information.

6.1 How to Work with the Wave Generator

The following subsections describe the wave generator handling in detail. See also "Wave Generator Examples" (p. 75).

6.1.1 Basic Data

The number of wave tables can be queried using the SPA? command (p. 157), parameter ID 0x1300010A. The E-753 has 10 wave tables for creating and (temporarily) storing arbitrary waveforms (identifiers are 1 to 10).
To ask for the number of wave generators, use the TWG? command (p. 169). As a single-axis controller, the E-753 has only one wave generator (identifier is 1).
A certain amount of the controllers memory space is reserved for the waveform data (ask with the SPA? command (p. 157), parameter ID 0x13000004). E-753 provides 65536 data points for waveform definition. This memory space is (temporarily) allocated to the individual wave tables during the waveform definition.

6.1.2 Basic Operation

1. Define the waveform segment-by-segment using the WAV command (p. 172). The waveform will be written to the selected wave table.

2. Connect the wave generator to the wave table using the WSL command (p. 187).

3. Start the wave generator output and hence the motion of the axis using the WGO command (p. 179). You can choose several start options (e.g. start/stop by external trigger, initialization/usage of the Dynamic Digital Linearization (DDL) feature; see the description of the WGO command and "Dynamic Digital Linearization (DDL)" (p. 85) for more information).
When starting the wave generator, data recording is started automatically.

4. Stop the wave generator output with WGO or #24 (p. 113) or STP (p. 160).

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

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

6.1.3 Additional Steps and Settings

You can calculate the memory space remaining if you ask with WAV? (p. 177) for the current wave table lengths. To release memory space, delete the content of selected wave tables with the WCL command (p. 177).

After you send the waveform definition to the wave table (with WAV (p. 172)), it is always a good idea to check it by reading back the waveform sequence from the controller before actually outputting it. This can be done by the GWD? command (p. 133). Note that the response to GWD? does not contain any offset to the wave generator output set with WOS (p. 184).

You can add an offset to the output of a wave generator using the WOS command (p. 184). Thereafter, the output of the specified wave generator is then the sum of the offset value and the wave value:
Generator Output = Offset + Current Wave Value
If the wave generator is started with the option "start at the endpoint of the last cycle", the E-753 at the end of each output cycle equates the WOS offset value with the current generator output.
WOS sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory. You can also change this parameter with SPA (p. 154) or SEP (p. 152) and save the value to non-volatile memory with WPA (p. 186).
Deleting wave table content with WCL (p. 177) has no effect on the WOS settings.

For triggering purposes, the wave generator output can be coupled with the digital output line OUT2 of the controller (see "I/O" socket (p. 222)). You should first use TWC (p. 169) to set the signal state of the output line to "low" for all waveform points ("low" is also the power-on default). Then use the TWS command (p. 169) to define the trigger line actions by setting the desired signal states of the output line (high or low) for selected waveform points. At last, use the CTO command (p. 122) to activate the Generator Trigger mode for the output line.

The #9 (p. 112) single-character command can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode (e.g. with DDL). With WGO? you can ask for the last-commanded wave generator start options (WGO settings (p. 179)).

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

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

WTR sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory. You can change this parameter also with SPA (p. 154) or SEP (p. 152) and save the value to non-volatile memory with WPA (p. 186). The value is always valid for the whole system and can not be set separately for individual wave generators. The value of the parameter in volatile memory can be read with the WTR? command (p. 189).

WTR also sets the type of interpolation to use for the wave generator output. If the Wave Generator Table Rate is larger than 1, interpolation helps to avoid sudden position jumps of an axis controlled by the wave generator.

With WGR (p. 183) you can restart data recording while the wave generator is running. The recorded data can be read with the DRR? command (p. 130). See "Data Recording" (p. 66) for more information.

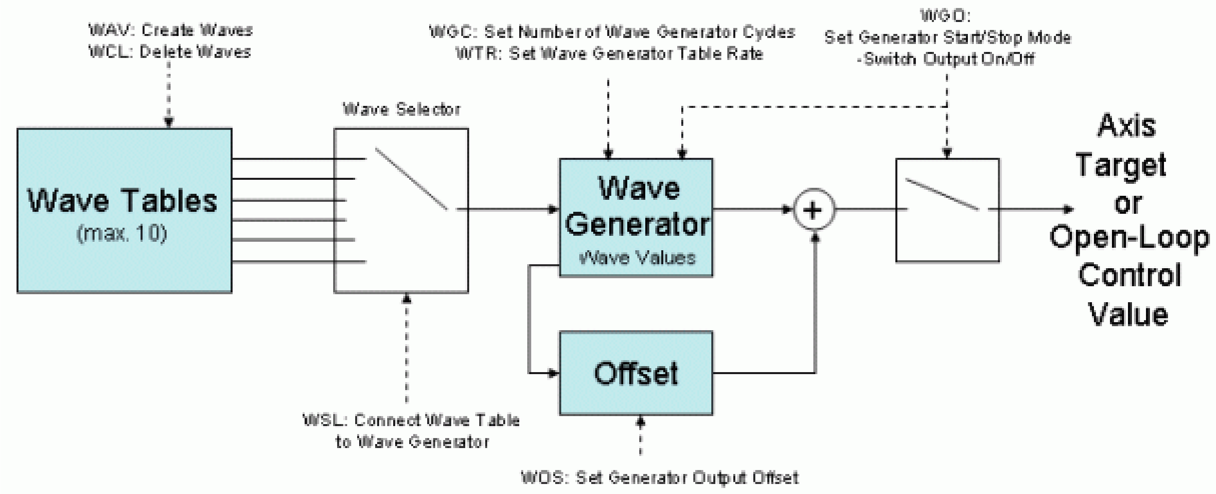


Figure 16: Wave generator block diagram

6.1.4 Application Notes

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

The frequency of the wave generator output depends, among other factors, on the wave table length. When you create waveforms, keep in mind that the usable frequency is limited by the available amplifier power. If the frequency is too high, overheating of the amplifier(s) can occur, and the piezo voltage output will be deactivated automatically. If that occurs, the piezo stage will not move although communication with the controller is still possible (see also "Operating Limits" (p. 224)).

Wave generator output and analog control input:
It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In that case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be commanded by analog control input, you can stop the wave generator output, but not restart it.
Wave generator output and move commands:
When the wave generator output is active, move commands like MOV (p. 144) or SVA (p. 160) are not allowed for the associated axis.

See "Control Value Generation" (p. 51) for details.

A wave generator outputs absolute values. In closed-loop operation (servo ON), the output is interpreted as target positions in either case. In open-loop operation (servo OFF), the interpretation of the wave generator output depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). By default, the matrix is set up so that commanded open-loop control values numerically correspond to axis position values.
Servo can not be switched off (SVO (p. 163)) while a wave generator is running for the axis.

As long as a wave generator is running, it is not possible to change (WSL (p. 187)) or to delete (WCL (p. 177)) the connected wave table (i.e. the waveform). The wave generator table rate (WTR (p. 189)), the number of output cycles (WGC (p. 178)), the wave offset (WOS (p. 184)) and the output trigger settings (TWS (p. 169)) can be modified while a wave generator is running.

When a wave generator finishes by running through a specified number of cycles completely, the final position will be the first point of the waveform, unless the option "start at the endpoint of the last cycle" was selected. In that case, the final position is the sum of the endpoint of the last output cycle and any offset defined with WAV (p. 172) for the waveform.

When the wave generator is stopped within an output cycle by command, the axis will remain at the last output position until a new position is commanded. If the wave generator is then restarted, it will normally continue with the first point of the waveform, unless started with the option "start wave generator output triggered by external signal", and the digital input line IN2 (see "I/O" socket (p. 222)) is used for triggering. In that case, the generator starts with the first rising edge which is detected on this input line, and it will be stopped when a falling egde is detected on this line. With the next rising edge, the generator output will continue at the waveform point where it was stopped. Starting and stopping the wave generator this way can be repeated indefinitely.
Wave generator output will continue even if the terminal or the program from which it was started is quit or if the high voltage output is deactivated.
See the WGO command (p. 179) for more information.

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

- Wave table content (WAV (p. 172))
- Assignment of wave tables to wave generators (WSL (p. 187))
- Output trigger settings (TWS (p. 169))
- Number of cycles for wave generator output (WGC (p. 178))

The following settings can be saved with WPA (p. 186) to non-volatile memory, where they become the power-on defaults:

- Wave offset (WOS (p. 184))
- Wave generator table rate (WTR (p. 189))

The different software interfaces provided for the controller also support use of the wave generator. Waveforms can be defined, stored and displayed in and by the software in a more user-friendly way than in a terminal using WAV (p. 172) and WGO (p. 179). If using the wave generator with the GCS DLL, PIMikroMove®, NanoCapture™ or LabView, read the descriptions in the associated software manual first.

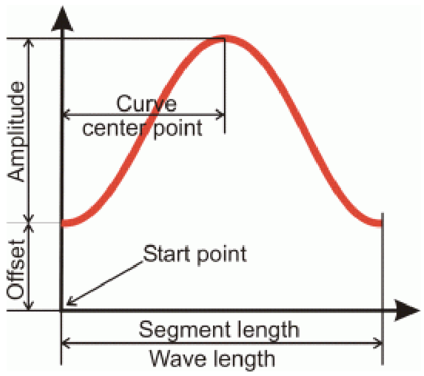
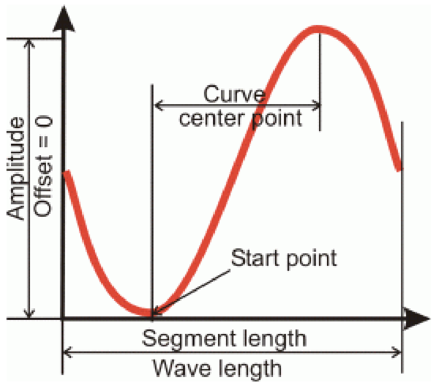
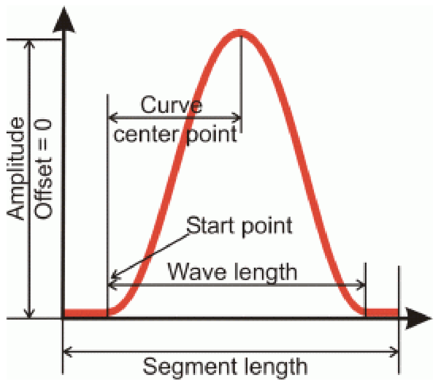
6.2 Wave Generator Examples

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

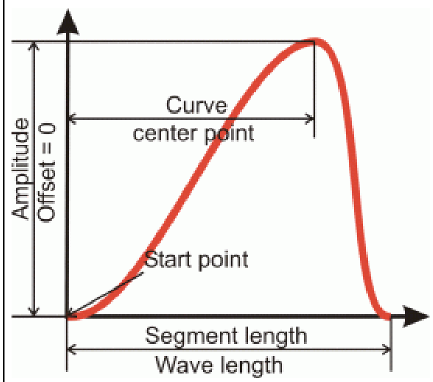
6.2.1 Defining Waveforms

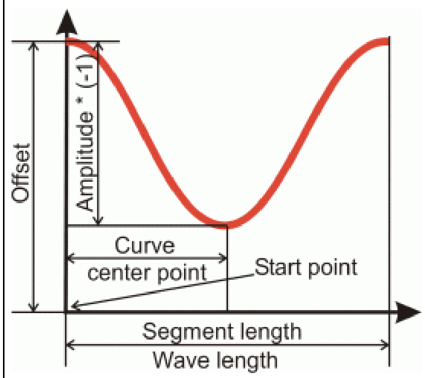
Examples for how to define waveform segments for the wave tables, based on predefined curve shapes (each WAV command defines a waveform segment which either replaces or is appended to the waveform in the specified wave table):

Sine Curves

WAV command	Comments	Waveform Segment
WAV 2 X SIN_P 2000 20 10 2000 0 1000 <WaveTableID> = 2 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 20 <Offset> = 10 <WaveLength> = 2000 <StartPoint> = 0 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, waveform offset = 10 (Do not confuse with the wave generator output offset set with WOS!), symmetric curve	
WAV 2 X SIN_P 2000 30 0 2000 499 1000 <WaveTableID> = 2 <AppendWave> = X <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 30 <Offset> = 0 <WaveLength> = 2000 <StartPoint> = 499 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, symmetric curve	
WAV 2 & SIN_P 2000 25 0 1800 100 900 <WaveTableID> = 2 <AppendWave> = & <WaveType> = SIN_P <SegLength> = 2000 <Amp> = 25 <Offset> = 0 <WaveLength> = 1800 <StartPoint> = 110 <CurveCenterPoint> = 900	The defined segment will be appended to the existing wave table contents, symmetric curve	

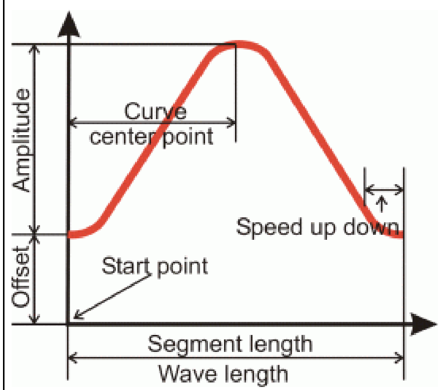
<table>
<tr><th>WAV command</th><th>Comments</th><th>Waveform Segment</th></tr>
<tr><td>WAV 3 X SIN_P 4000 20 0 4000 0 3100

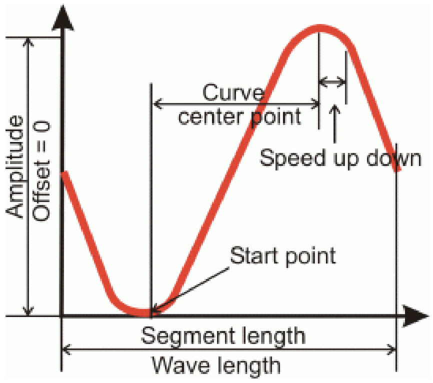
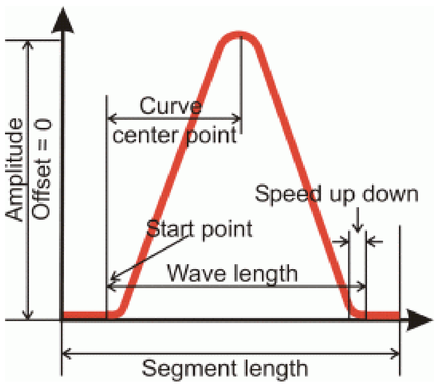
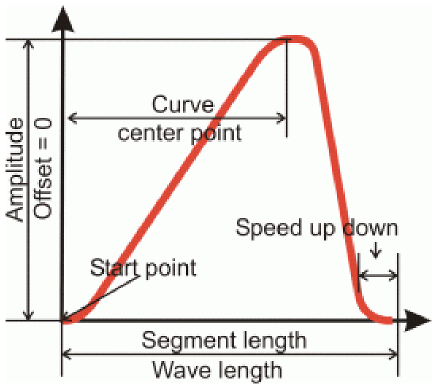
<WaveTableID> = 2
<AppendWave> = X
<WaveType> = SIN_P
<SegLength> = 4000
<Amp> = 20
<Offset> = 0
<WaveLength> = 4000
<StartPoint> = 0
<CurveCenterPoint> = 3100</td><td>The previous contents of the wave table are overwritten by the new segment, asymmetric curve</td><td>
</td></tr>
<tr><td>WAV 1 X SIN_P 1000 -30 45 1000 0 500

<WaveTableID> = 2
<AppendWave> = X
<WaveType> = SIN_P
<SegLength> = 1000
<Amp> = -30
<Offset> = 45
<WaveLength> = 1000
<StartPoint> = 0
<CurveCenterPoint> = 500</td><td>The previous contents of the wave table are overwritten by the new segment, negative-amplitude curve, symmetric curve</td><td>
</td></tr>
</table>

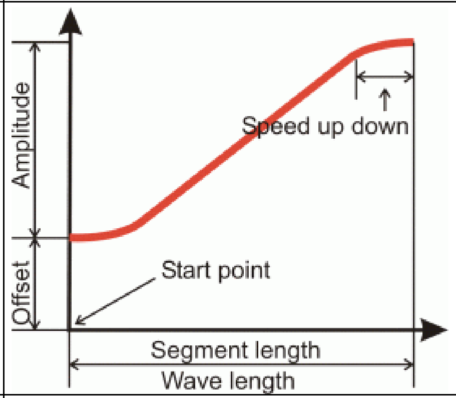
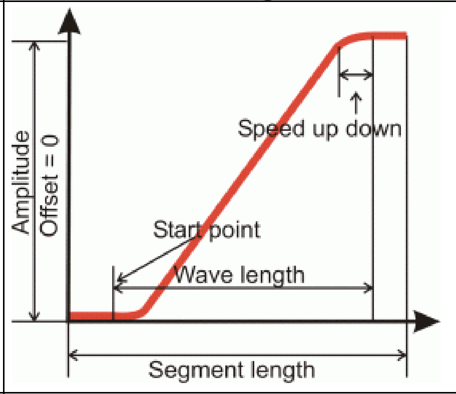
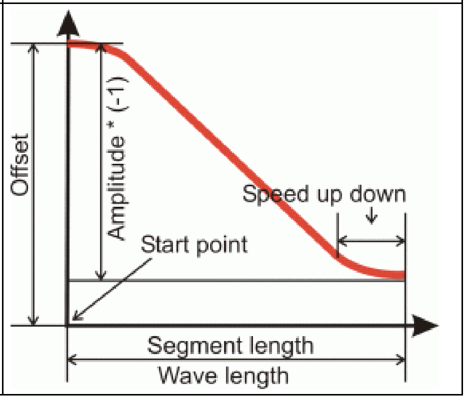
Ramp Curves

<table>
<tr><th>WAV command</th><th>Comments</th><th>Waveform Segment</th></tr>
<tr><td>WAV 4 X RAMP 2000 20 10 2000 0 100 1000

<WaveTableID> = 4
<AppendWave> = X
<WaveType> = RAMP
<SegLength> = 2000
<Amp> = 20
<Offset> = 10
<WaveLength> = 2000
<StartPoint> = 0
<SpeedUpDown> = 300
<CurveCenterPoint> = 1000</td><td>The previous contents of the wave table are overwritten by the new segment, waveform offset = 10 (Do not confuse with the wave generator output offset set with WOS!) symmetric curve</td><td>
</td></tr>
</table>

WAV command	Comments	Waveform Segment
WAV 4 X RAMP 2000 35 0 2000 499 100 1000 <WaveTableID> = 4 <AppendWave> = X <WaveType> = RAMP <SegLength> = 2000 <Amp> = 35 <Offset> = 0 <WaveLength> = 2000 <StartPoint> = 499 <SpeedUpDown> = 300 <CurveCenterPoint> = 1000	The previous contents of the wave table are overwritten by the new segment, symmetric curve	
WAV 5 X RAMP 2000 15 0 1800 120 50 900 <WaveTableID> = 5 <AppendWave> = X <WaveType> = RAMP <SegLength> = 2000 <Amp> = 15 <Offset> = 0 <WaveLength> = 1800 <StartPoint> = 120 <SpeedUpDown> = 150 <CurveCenterPoint> = 900	The previous contents of the wave table are overwritten by the new segment, symmetric curve	
WAV 5 & RAMP 3000 35 0 3000 0 200 2250 <WaveTableID> = 5 <AppendWave> = & <WaveType> = RAMP <SegLength> = 3000 <Amp> = 35 <Offset> = 0 <WaveLength> = 3000 <StartPoint> = 0 <SpeedUpDown> = 200 <CurveCenterPoint> = 2250	The defined segment will be appended to the existing wave table contents, asymmetric curve	

Single Scan Line Curves

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

6.2.2 Modifying the Wave Generator Table Rate

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

Command String to Send	Action Performed
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
SPA? 1 0x0E000200	Ask for the servo update time of the controller (reading the wave table for wave generator output is clocked by servo cycles). E-753, for example, has a servo update time of 40 µs.
WTR?	Ask for the current wave table rate and interpolation settings, default is wave table rate = 1 (i.e. each wave table point will be output for a duration of one servo cycle). The duration of one wave generator output cycle will be: Servo Update Time (in s) * WTR value * Number of Points = Output Duration (in s) 0.000040 s * 1 * 2000 = 0.08 s
WTR 0 3 1	Set the wave table rate to 3, tripling the duration of one wave generator output cycle, with linear interpolation (each wave table point will now "occupy" 3 servo cycles, but with linear interpolation applied to smooth the output). Duration of one output cycle will now be: 0.000040 s * 3 * 2000 = 0.24 s Note that the WTR command must always specify all wave generators in the system (<WaveGenID> must be 0).

6.2.3 Trigger Output Synchronized with Wave Generator

Using the digital output line of the E-753, it is possible to trigger external devices. See "I/O" socket (p. 222) for the availability of the lines (pinout) and "Using Trigger Input and Output" (p. 38) for an overview.

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

Command String to Send	Action Performed
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
TWC	Clears all output trigger settings related to the wave generator by switching the signal state for all points to "low" (the power-on default state is also "low"). It is recommended to use TWC before new trigger actions are defined.
TWS 1 500 1 1 1500 1 1 1900 1 1 2000 1	Set trigger actions for the digital output line OUT2 (identifier is 1): at the waveform points 500, 1500, 1900 and 2000 it is set high; at all other points the state of the line is low (due to the TWC usage).
CTO 1 3 4	The digital output line OUT2 is set to "Generator Trigger" mode.
WSL 1 2	Connect Wave Generator 1 (Axis 1) to Wave Table 2
WGO 1 1	Start output of Wave Generator 1 immediately (synchronized by servo cycle). Now the trigger output action will take place as specified.
WGO 1 0	Stop output of Wave Generator 1 and hence also the trigger output.

6.2.4 Wave Generator Started by Trigger Input

Using the digital input lines of the E-753, it is possible to apply start/stop signals for the wave generator output. See "I/O" socket (p. 222) for the availability of the lines (pinout) and "Using Trigger Input and Output" (p. 38) for an overview.

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

Command String to Send	Action Performed
WAV 2 X SIN_P 2000 20 10 2000 0 1000	Define a sine waveform for Wave Table 2, the segment length and hence the number of points in the wave table is 2000
WSL 1 2	Connect Wave Generator 1 (Axis 1) to Wave Table 2
WGO 1 2	Start output of Wave Generator 1 triggered by external signal. To provide the external signal, the digital input lines IN1 or IN2 can be used. If IN1 is used: The wave generator output starts with the first rising edge which is detected on this input line. If IN2 is used: The generator output starts with the first rising edge which is detected on this input line, and it will be stopped when a falling edge is detected on this line. With the next rising edge, the generator output will continue at the waveform point where it was stopped. Starting and stopping the wave generator this way can be repeated indefinitely. If output cycle limitations were made with WGC: with each generator restart the counting of the output cycles continues, and the generator will be stopped when the given number of cycles are completed, irrespective of any further trigger pulses. It is possible to mix the usage of both digital input lines.
WGO 1 0	Stop output of Wave Generator 1 (any further trigger pulses will be ignored). You can also use #24 or STP.

6.3 Wave-Generator-Related Commands and Parameters

Command	Description	Notes
CTO (p. 122)	Set Configuration Of Trigger Output	Activates the Generator Trigger output mode which is required for the triggerline actions set with TWS.
DDL? (p. 126)	Get DDL Table Values	Reads the current content of the DDL table(s). DDL initialization and usage are started by the wave generator start command (WGO).
DRR? (p. 130)	Get Recorded Data Values	Reads the last recorded data. Data recording is triggered by the WGO and WGR commands (among others).
GWD? (p. 133)	Get Wave Table Data	Should be used to check the waveform before the wave generator output is started.
TWC (p. 169)	Clear All Wave Related Triggers	Clears only the TWS settings, but not the CTO settings.
TWG? (p. 169)	Get Number Of Wave Generators	Number of wave generators = number of axes
TWS (p. 169)	Set TriggerLine Action To Waveform Point	In addition, the CTO command must be used to activate the Generator Trigger mode for the desired digital output line.
WAV (p. 172)	Set Waveform Definition	A waveform must be defined before the wave generator output can be started.
WAV? (p. 177)	Get Waveform Definition	Reads the current wave table length.
WCL (p. 177)	Clear Wave Table Data	Clears the wave table content, but not the WSL and WOS settings.
WGC (p. 178)	Set Number Of Wave Generator Cycles	If WGC is not used, the wave generator must be stopped with WGO, #24 (p. 113) or STP (p. 160).
WGC? (p. 178)	Get Number Of Wave Generator Cycles	
WGO (p. 179)	Set Wave Generator Start/Stop Mode	The WGO command starts the wave generator output. It provides several start options, e.g. "Start wave generator output triggered by external signal", "Use and reinitialize DDL" or "Use DDL".
WGO? (p. 183)	Get Wave Generator Start/Stop Mode	Gets the last commanded start options, but not the activation status (use #9 instead)

WGR (p. 183)	Starts Recording in Sync with Wave Generator	Restarts data recording as long as a wave generator is running.
WOS (p. 184)	Set Wave Generator Output Offset	Sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory.
WOS? (p. 185)	Get Wave Generator Output Offset	Gets the value of the Wave Offset parameter, ID 0x1300010b, from volatile memory.
WSL (p. 187)	Set Connection Of Wave Table To Wave Generator	Must be set before the wave generator can be started.
WSL? (p. 188)	Get Connection Of Wave Table To Wave Generator	
WTR (p. 189)	Set Wave Generator Table Rate	Sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory; determines the interpolation type for wave table rate values > 1.
WTR? (p. 190)	Get Wave Generator Table Rate	Gets the value of the Wave Generator Table Rate parameter (ID 0x13000109) from volatile memory and the current interpolation type.
#9 (p. 112)	Get Wave Generator Status	Gets the current activation status of the wave generator, but not the start options (use WGO? instead)

See "How to Work with the Wave Generator" (p. 70) for more information. For detailed command descriptions see "Command Reference" (p. 112). For the identifiers of the items which can be addressed with the commands see "Accessible Items and Their Identifiers" (p. 43).

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

See "Controller Parameters" (p. 203) for more information regarding the controller parameters and their handling.

7 Dynamic Digital Linearization (DDL)

Dynamic Digital Linearization (DDL) makes it possible to achieve significantly better position accuracy in dynamic applications with periodic motion. It is used in conjunction with the wave generator output and in addition to the "normal" servo algorithm in closed-loop operation. DDL "observes" axis motion over one or more wave generator output cycles (DDL initialization). The information gathered is written to "DDL tables" and can then be used to refine the control output signals.

"Working Principle" (p. 85) describes the DDL basics, "How to Activate the DDL Licence" (p. 89) gives information on how to get the DDL started the first time, in "How to work with the DDL" (p. 91) you will learn how to use the DDL feature, and "DDL-Related Commands and Parameters" (p. 93) gives a summary.

See also "Wave Generator" (p. 70).

7.1 Working Principle

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

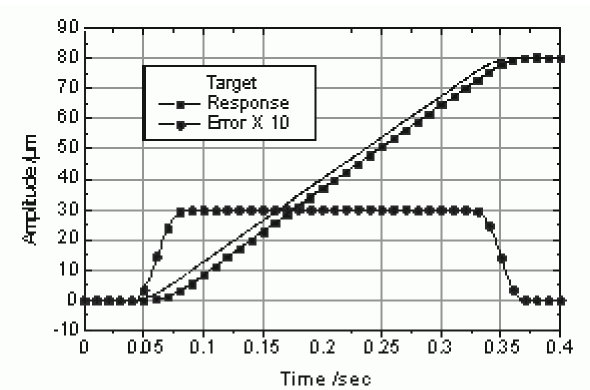


Figure 17: Tracking error for typical linear PID servo system

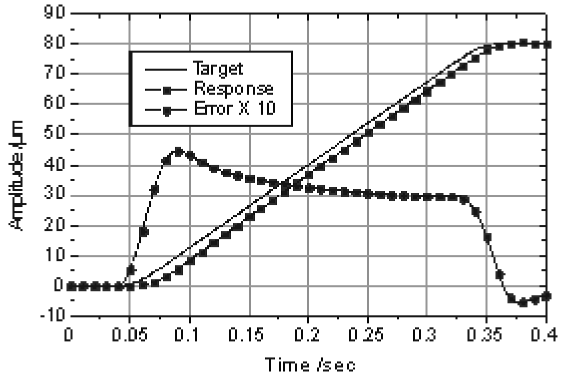


Figure 18: Tracking error for linear bi-directional scanning application

In addition, for linear bi-directional scanning applications, the tracking error is not a constant, as with other types of linear PID servo systems, but changes with the scan position. This is caused by the piezo actuators non-linearity and the servo-controller'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 motion 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.

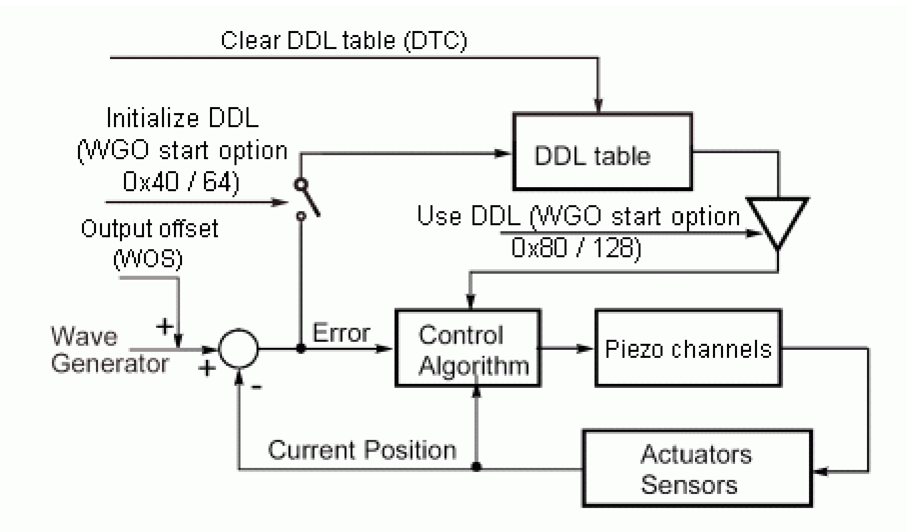


Figure 19: DDL block diagram

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.

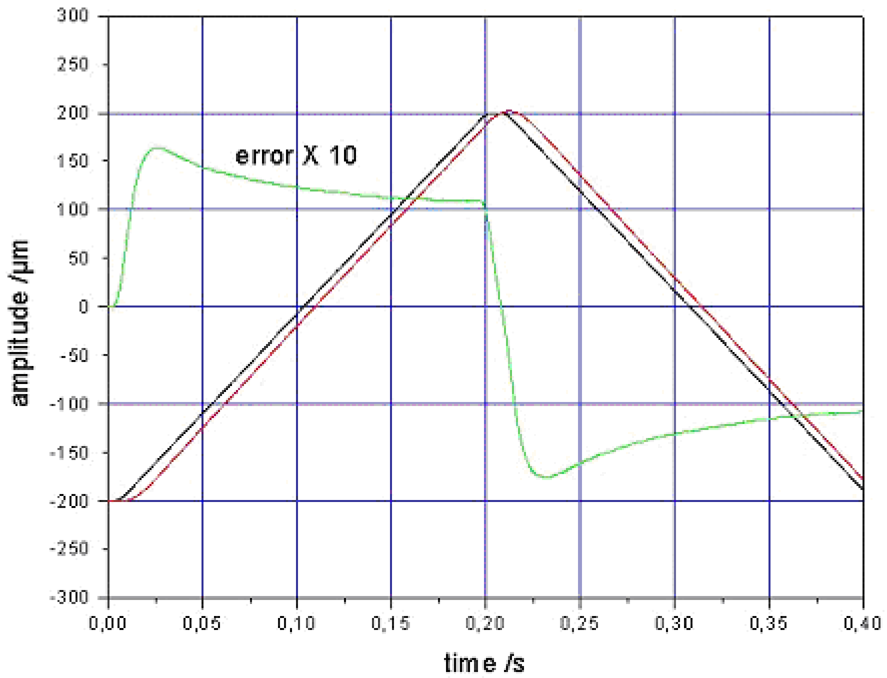


Figure 20: Without DDL

With DDL, after one initialization 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.

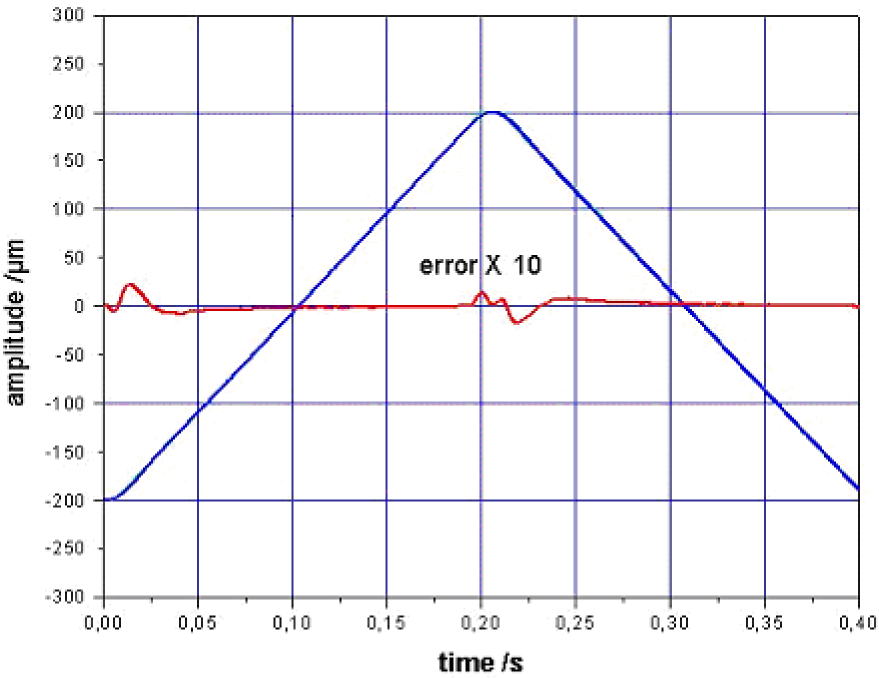


Figure 21: With DDL, one initialization cycle

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

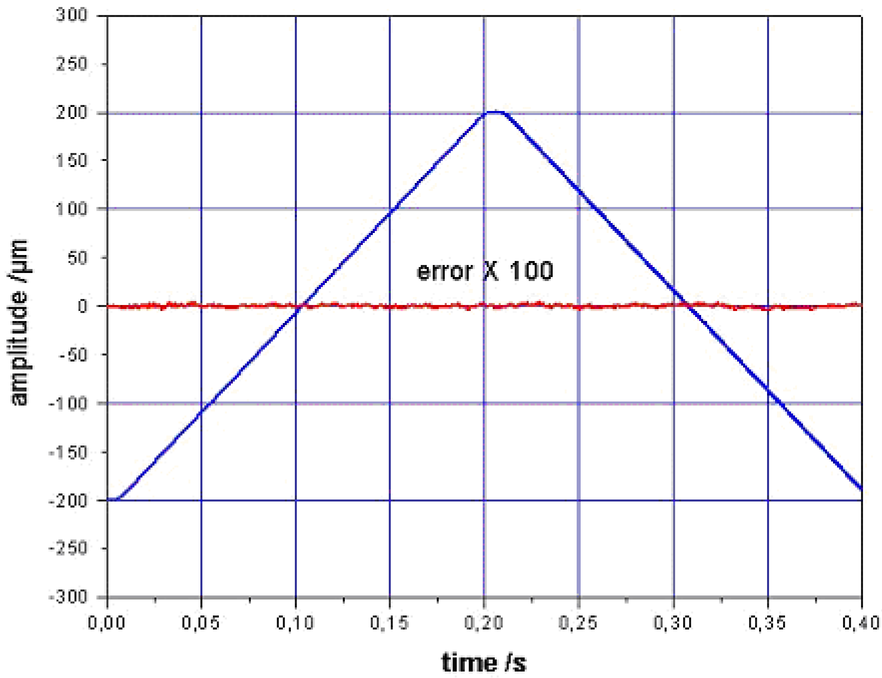


Figure 22: With DDL, after several DDL initialization cycles

7.2 How to Activate the DDL License

The Dynamic Digital Linearization (DDL) feature must be expressly ordered (order# E-710.SCN). You can activate DDL after purchase and without opening the device.
To activate the DDL license, proceed as follows:

1 Order DDL for your controller and inform PI of the controller serial number—PI generates your DDL license number based on this information.
The serial number is saved in the controller as the Device S/N parameter, ID 0x0D000000.
You can read it in NanoCapture™ or using the SPA? (p. 157) or SEP? (p. 153) command:
Send: SPA? 1 0x0D000000 (reads the value from volatile memory)
or
Send: SEP? 1 0x0D000000 (reads the value from non-volatile memory)

2 PI will provide you with the DDL license number.

3 Switch to command level 1 using the CCL command (p. 119):
Send: CCL 1 advanced

4 Write the DDL license number to the non-volatile memory of your controller as the DDL License parameter, ID 0x0E000400.
You can do this in NanoCapture™ or using the SEP command (p. 152):
Send: SEP 100 1 0x0E000400 *licensenumber*

5 Power cycle the controller.

6 Check if the DDL activation was successful. To do this, read the value of the DDL License Valid parameter, ID 0x0E000401 in NanoCapture™ or using SPA? or SEP?.
Send: SPA? 1 0x0E000401
or
Send: SEP? 1 0x0E000401
DDL activation was successful if the parameter value is 1.
Otherwise the DDL feature is still deactivated.

If the DDL activation fails:

- Check if the correct serial number was used for license number generation.
- Check whether the correct license number was entered into controller memory:
Send: SEP? 1 0x0E000400
- Make sure that command level 1 is active during license activation.

7.3 How to work with the DDL

NOTE

The Dynamic Digital Linearization (DDL) feature must be expressly ordered (order# E-710.SCN). You can activate it after purchase and without opening the device. See "How to activate the DDL License" (p. 89) for more information.

If you are not sure if the DDL feature is activated on your controller, use the SPA? command (p. 157) to check the value of parameter 0x0E000401 (DDL License Valid). If the parameter value is 1, the DDL feature is activated, otherwise it is deactivated.

DDL related commands will not provoke an error when the DDL feature is deactivated.

!

CAUTION

Before you work with Dynamic Digital Linearization (DDL), use the NanoCapture™ software to eliminate any residual oscillations by adjusting the servo parameters (notch filter frequency, servo-loop P-term (loop gain), servo-loop I-term (time constant), servo-loop slew rate; see "Servo-Controller Dynamic Calibration" (p. 97) and the NanoCapture™ manual for more information). Using DDL could be critical if there are any residual oscillations in the system; DDL will then cause the oscillations to build up—the more wave generator cycles are used for DDL initialization, the stronger the effect.

The number of DDL tables can be queried using the TLT? command (p. 165). It corresponds to the number of axes available on the controller.
As a single-axis controller, the E-753 has only one DDL table (and only one wave generator).
A certain amount of memory space is reserved for the DDL data (ask with the SPA? command (p. 157), parameter ID 0x1400000B).
For the E-753, 65536 points are available. The DDL table does not always use the complete memory space—the number of points in a DDL table corresponds to the length of the waveform which is output during the DDL initialization.
You can ask for the current DDL table length using the DTL? command (p. 132).

DDL tables are automatically filled with data when a wave generator is started with the "Use and reinitialize DDL" option activated. It is also possible to write data "manually" with the DDL command (p. 125)—in this case, make sure that the DDL table length is correct (see DDL command description for more information).

NOTES

The DDL initialization must be repeated when a new stage is connected, the servo parameters are changed (e.g. due to load changes) or the waveform is changed.

It is recommended to start the DDL initialization for all axes at the same time. Each new initialization will stop all running initialization processes. The initialization process is also stopped by the DDL command.

How to initialize DDL for an axis:

1 Define a waveform using the WAV command (p. 172).

2 Optionally: Set the number of wave generator cycles to use for DDL initialization. This can be done using the SPA (p. 154) or SEP (p. 152) command with the DDL Repeat Number parameter (ID 0x14000001). The factory default is 35. (To have write access to the parameter, it might be necessary to switch to the proper command level using CCL (p. 119)).

3 Not essential but recommended: Delete DDL table content which is no longer used. This can be done using the DTC command (p. 132).

4 When the servo parameters (notch filter frequency, servo-loop P-term, servo-loop I-term and servo-loop slew rate) have changed: Recalculate the internal DDL processing parameters using the DPO command (p. 127).

5 Assign the waveform to the axis using the WSL command (p. 187).

6 Switch the servo on for the axis using the SVO command (p. 163) (closed-loop operation).

7 Start the wave generator with the WGO command (p. 179), with the "Use and reinitialize DDL" option activated.

8 Optionally: Check the content of the DDL table using the DDL? command (p. 126).

How to use DDL for an axis:

As long as your application does not change, you can use the current DDL table content without new initialization. In this case, start the wave generator with WGO and the "Use DDL" option activated.

NOTE

If the DDL performance does not prove satisfactory or if errors occur during DDL use:

- Recalculate the internal DDL processing parameters using DPO (p. 127).
- Reduce the frequency of the wave generator output to make sure that the axis can follow the target values. To do this, define a new, optimized waveform with the WAV command (p. 172). You can also use the WTR command (p. 189), which sets the wave generator table rate, to optimize the output frequency.

After such changes DDL initialization must be repeated.

NOTE

The DDL table content and the calculated processing parameters will be lost when the controller is powered down or rebooted.

7.4 DDL-Related Commands and Parameters

Command	Description	Notes
DDL (p. 125)	Set DDL Table Values	Can be used to fill the DDL table(s) "manually" with data
DDL? (p. 126)	Get DDL Table Values	Reads the current content of the DDL tables(s)
DPO (p. 127)	DDL Parameter Optimization	Recalculates the DDL processing parameters (Time Delay Max, ID 0x14000006, and Time Delay Min, ID 0x14000007), required if the servo parameters have changed
DTC (p. 132)	Clear DDL Table Data	It is recommended that the content of DDL tables which are not used be deleted. This avoids errors due to unsufficient memory space during the DDL initialization. On controller power down or reboot, the content of the DDL tables will be erased automatically.

DTL? (p. 132)	Get DDL Table Length	Returns the value of the Max DDL Points parameter, ID 0x1400000B
TLT? (p. 165)	Get Number of DDL Tables	The reported value corresponds to the number of axes available on the controller
WAV (p. 172)	Set Waveform Definition	Since DDL works only in conjunction with the wave generator output, the waveform must first be defined
WGO (p. 179)	Set Wave Generator Start/Stop Mode	The WGO command starts the wave generator output and provides two modes for the DDL: "Use and reinitialize DDL" and "Use DDL"

See "How to Work with the DDL" (p. 91) for more information. For detailed command descriptions see "Command Reference" (p. 112). For the identifiers of the items which can be addressed with the commands see "Accessible Items and Their Identifiers" (p. 43).

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description
0x0e000400	1	System	1	INT	DDL license
0x0e000401	3	System	1	INT	DDL license valid
0x14000001	1	Logical Axis	1	INT	DDL repeat number
0x14000006	1	Logical Axis	1	FLOAT	Time Delay Max
0x14000007	1	Logical Axis	1	FLOAT	Time Delay Min
0x14000008	1	Logical Axis	1	INT	Time Delay Change Rule
0x1400000a	1	Logical Axis	1	INT	DDL Zero Gain Number
0x1400000b	3	System	1	INT	Max DDL Points
0x14000100	2	Logical Axis	1	FLOAT	Autocal Time Delay Factor
0x14000101	2	Logical Axis	1	FLOAT	Autocal Min/Max Time Delay Factor

See "Controller Parameters" (p. 203) for more information regarding the controller parameters and their handling.

8 Calibration Procedures

8.1 ID-Chip Support / Stage Replacement

The piezo stage which is connected to the E-753 may contain an ID-chip (located in the stage connector). The following data is stored in the ID-chip (and can not be modified there by the customer):

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

The parameters which are usually stored in ID-chips are marked in the table in "Parameter Overview" (p. 205), but the list can differ slightly among the different mechanics provided by PI.

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

NOTE

When you connect a stage when the controller is powered on, the ID-chip of the stage is not read by the controller. To read the ID-chip data, the controller must be power cycled or rebooted using the RBT command (p. 148) or the corresponding host software functions.

A piezo stage can be easily exchanged due to the functionality of the ID-chip. Consider the following when replacing stages with ID-chips.

"Simple" Replacement

Normally, when you replace a piezo stage with a new unit and you are using standard factory settings for all parameters, you do not have to adjust anything. The ID-chip holds all information needed. At power-on of the system, the firmware reads the stage type and serial number stored in the ID-chip and compares this data to the data from the last connected stage, stored in the controller:

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

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

Upgrade or Repair of Stages

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

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

1. In NanoCapture™, open the *Device Parameter Configuration* window (*Config ⇒ Device Parameter Configuration*) and select the *System Mechanics* groups where you can enable the option. When this is done for all input signal channels associated with the stage, press the "Write selected edit values to default settings" button in the icon bar of the *Device Parameter Configuration* window.

   Alternatively you can use the following commands in a terminal to enable the option:
   SEP 100 1 0X0F000000 1 for input signal channel 1
   SEP 100 2 0X0F000000 1 for input signal channel 2 , etc.

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

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

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

Alternatively you can use the following commands in a terminal to disable the option:
SEP 100 1 0X0F000000 0 for input signal channel 1,
SEP 100 2 0X0F000000 0 for input signal channel 2, etc.

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

8.2 Servo-Controller Dynamic Calibration

If the controller and the attached piezo stages are ordered together and if PI has sufficient knowledge of your application, then the parameters of the closed-loop control algorithm (servo parameters) will be set to suitable values at the factory, and, if present, saved in the stage's ID-chip (p. 95). Modification of those parameters will, however, be necessary if the load applied to the piezo stage is changed. In this case, for each servo-controlled axis the following parameters may need to be modified in the controller:

- Notch filter frequencies (first notch filter: parameter ID 0x08000100, second notch filter: parameter ID 0x08000101)
- Servo-loop loop gain (P-term, parameter ID 0x07000300)
- Servo-loop time constant (I-term, parameter ID 0x07000301)
- Servo-loop slew rate (parameter ID 0x07000200)

It is most convenient to use NanoCapture™ to change these parameters.

NOTE

You have to switch to command level 1 before you can change the servo parameters (see the CCL command (p. 119) or the NanoCapture™ manual).

8.2.1 Overview

8.2.1.1 Suppressing Mechanical Resonances with Notch Filters

Mechanical resonances of the system exaggerate the response to certain frequencies in the control signal. The E-753 has two notch filters on the output of the servo-control loop to compensate for resonances in the mechanics by reducing the corresponding frequency components in the control signal. To determine the resonant frequencies and set the notch filters properly, observe the system response to an impulse in open-loop operation.

8.2.1.2 Control-Loop Parameters

The control-loop settings can be used to optimize settling time and control overshoot. The parameters which can be set for the axis include the loop gain (P-term) and the time constant (I-term). To set these parameters, the system response to a closed-loop step move is observed. The maximum closed-loop slew rate (velocity) can also be set.

Loop Gain (P-Term)

This parameter also affects the velocity 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 of the control value (step response). NanoCapture™ can display the step response of the stage as a graph.

Time Constant (I-Term)

The value of the time constant depends on the frequency of the first notch filter. It is therefore adjusted automatically whenever the frequency of the first notch filter 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 also be modified in the *Dynamic Tuner* window of NanoCapture™ explicitly.

Servo-Loop Slew Rate

The maximum servo-loop slew rate depends on the size of the piezo actuators and the robustness of the attached mechanics. This parameter limits the maximum velocity at which the axis will be commanded to move from the current position to a new target (only in closed-loop operation).

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

8.2.2 Adjustment Procedures

NOTES

For stages with ID-chip, to make the optimized settings available in the future, the option "Power Up Read ID-Chip" must have "disabled" as its power-on default (value of parameter 0X0F000000 = 0 in non-volatile memory). See "ID-Chip Support / Stage Replacement" (p. 95) for more information.

When you have finished the adjustment, i.e. when the stage performance is satisfactory, you can press the *Save as Default* buttons in the *Dynamic Tuner* window of NanoCapture™ to save the settings to the non-volatile memory of the E-753 as power-on defaults.

As long as the new settings have not yet been saved, pressing the *Reset all to defaults* buttons in the *Dynamic Tuner* window of NanoCapture™ permits resetting the P, I and Notch Filter terms to the values they had at power-on. This can be very useful to restore stable values if changing the servo parameters produces disturbing noises.

The servo parameters are also listed in the *Servo n* parameter groups in the *Device Parameter Configuration* window of NanoCapture™.

8.2.2.1 Setting Notch Filters

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 with NanoCapture™, the *Time Constant* servo parameter in the *P-I Controller* pane is adjusted automatically in accordance.

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

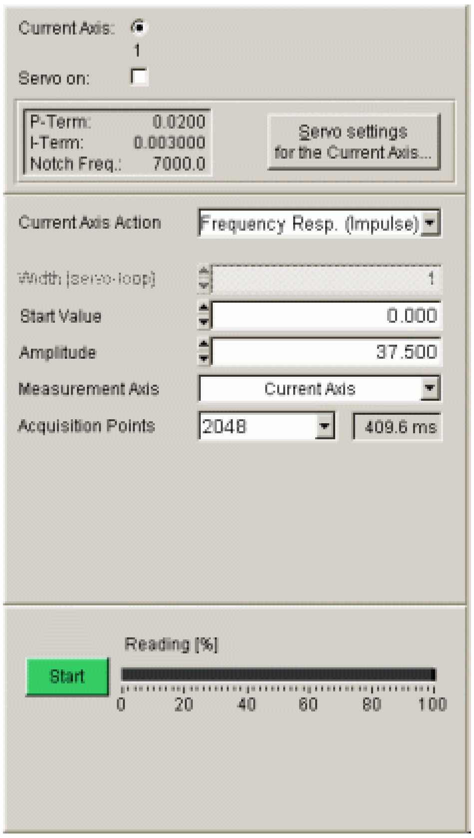
1 Make sure the mechanics is mounted and connected to the controller in exactly the same way as in the application. The load on the mechanics is especially important.

2 Start NanoCapture™ on a host PC connected to the E-753 (see the NanoCapture™ software manual on the included CD for details).

3 Use the *Current Axis* radio buttons to select the axis to set.

4 The measurement must be done in open-loop mode. Uncheck the *Servo ON* box for the axis.

5 In the *Current Axis Action* field, select *Frequency Response* (see figure below).



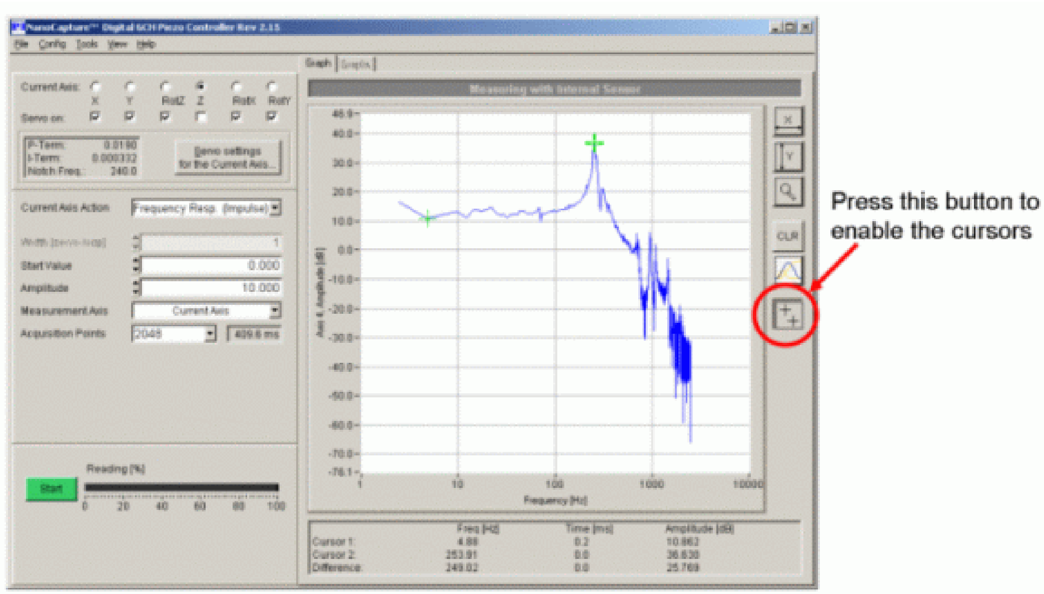
6 Set *Start Value* to 0 and *Amplitude* to about 15% of the axis travel range.

7 Set *Measurement Axis* to *Current Axis* (i.e. measure and display the axis which receives the impulse)

8 Start the measurement by clicking on the *Start* button. An impulse move is commanded (with the IMP (p. 143) command), the recorded

position data analyzed and displayed as a Bode frequency response diagram (see figure below).

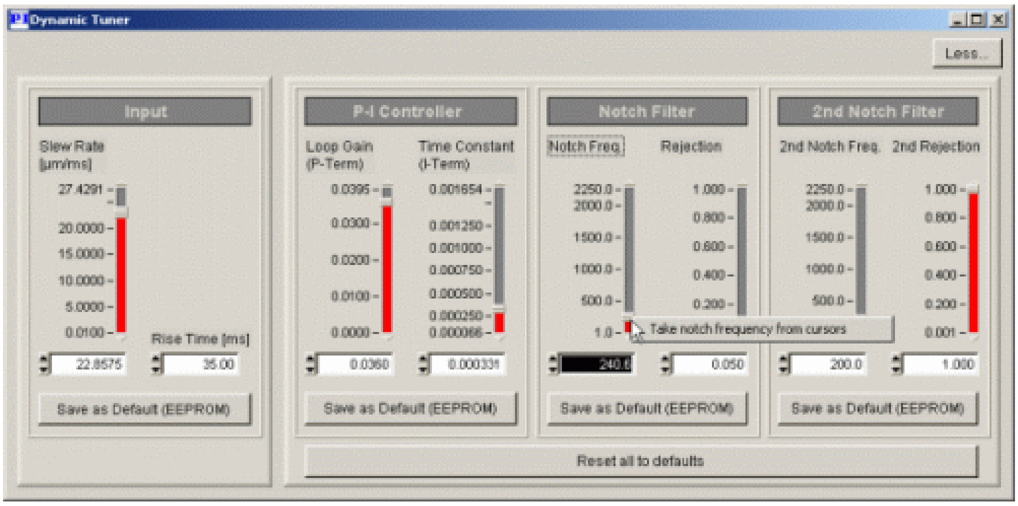
9 On the Bode plot, identify the resonance peak. To do so, you can place a cursor on the peak and read out the cursor value which is displayed below the graph (see figure below for a general example).



It is possible to adapt the notch filter setting automatically to the measured resonance peak (see step 11 below)—if you want to do so, 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.

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 (see figure below).



11 Move the *Notch Freq.* slider to adjust the notch frequency or right-click on the slider and click on the *Take notch frequency from cursors* item on the context menu that appears (see figure above). The notch filter frequency will then be taken from the cursor which has the higher frequency value.

12 If you have finished the settings, you can press the *Save as Default* buttons to save the settings in the controller as power-on defaults (the other settings in this window will be discussed later).

13 If desired, repeat this procedure for the second notch filter, using the next higher resonance and the *2nd Notch Freq.* slider.

8.2.2.2 Setting Control Loop Parameters

The servo parameters described above (p. 98) are optimized empirically by observing the effect of different values on a closed-loop Step Response. Proceed as follows with each axis you want to optimize:

1 Use the *Current Axis* radio button(s) to select the axis to set

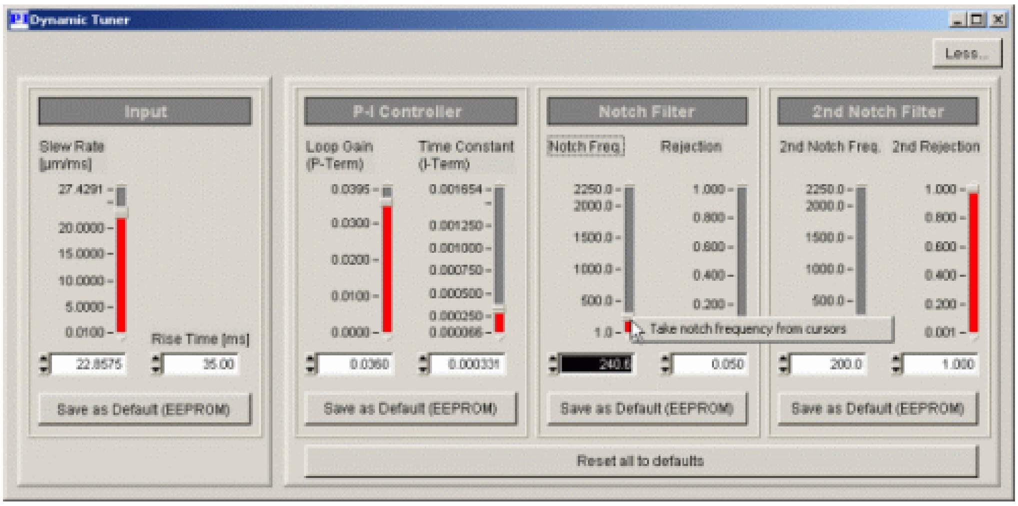
2 Set servo-control ON (check the *Servo ON* box) for that axis

3 Select *Step Response* in the *Current Axis Action* field

4 Set *Start Position* to 0 and *Amplitude* to about 15% of the axis travel range (if you activate the *Relative step* checkbox the step will start from the current position).

5 Set *Measurement Axis* to *Current Axis* (so that moved axis and measured axis are the same).

6 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 (see figure below).

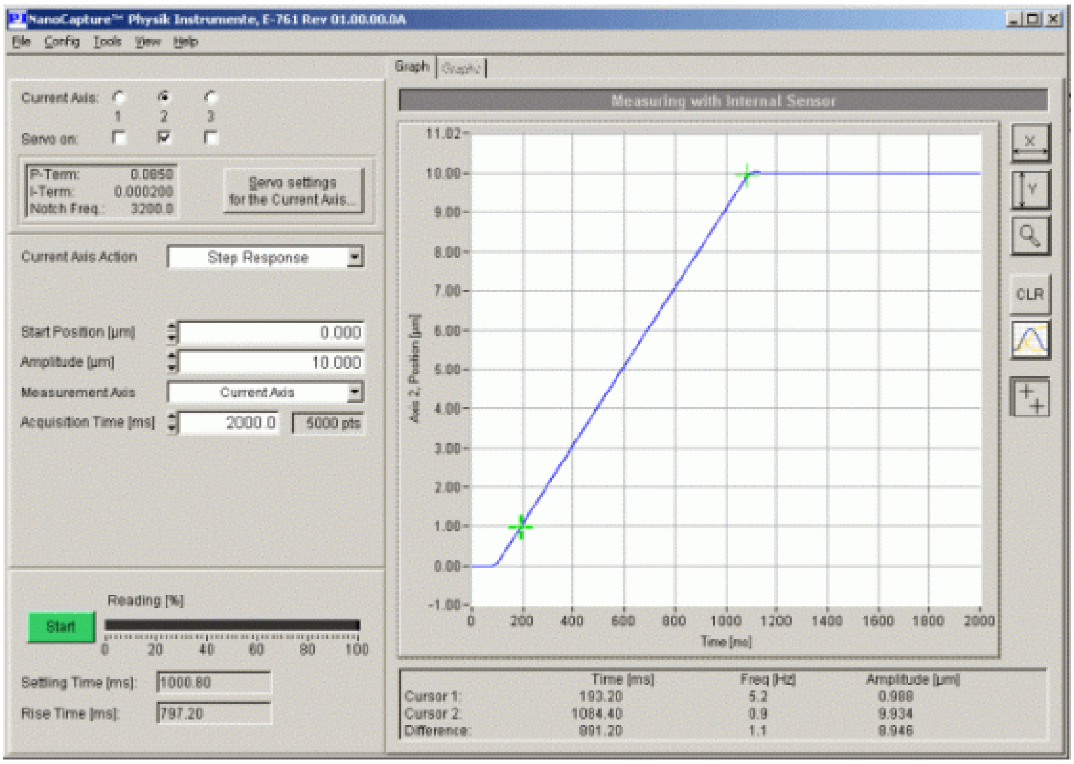


7 Change *Loop Gain (P-Term)*, *Slew Rate* (this is the servo-loop slew rate) and/or *Time Constant (I-Term)* slightly as desired.

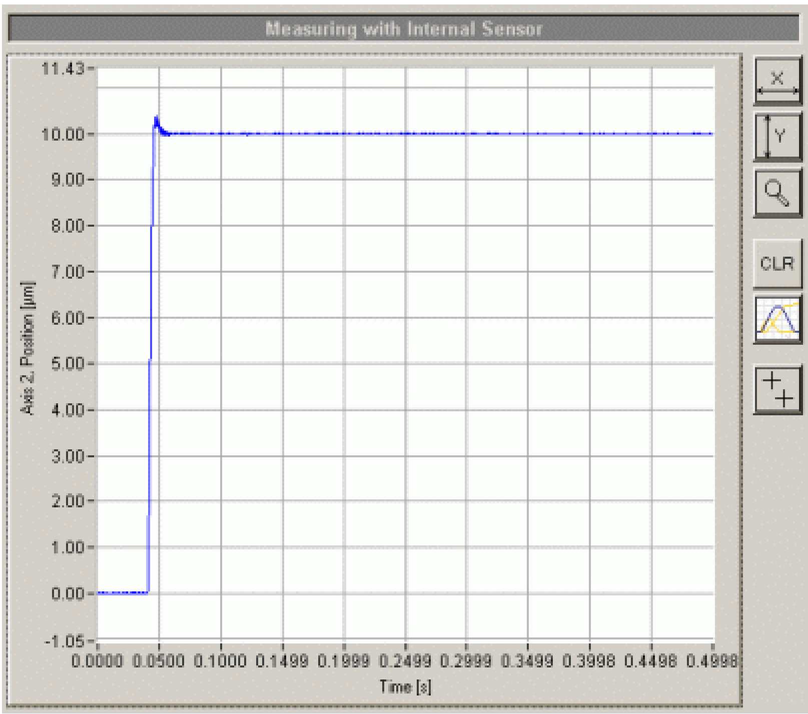
8 Click *Start* in the main window to perform a step.

9 Observe the results, comparing with the examples shown in the figures below.

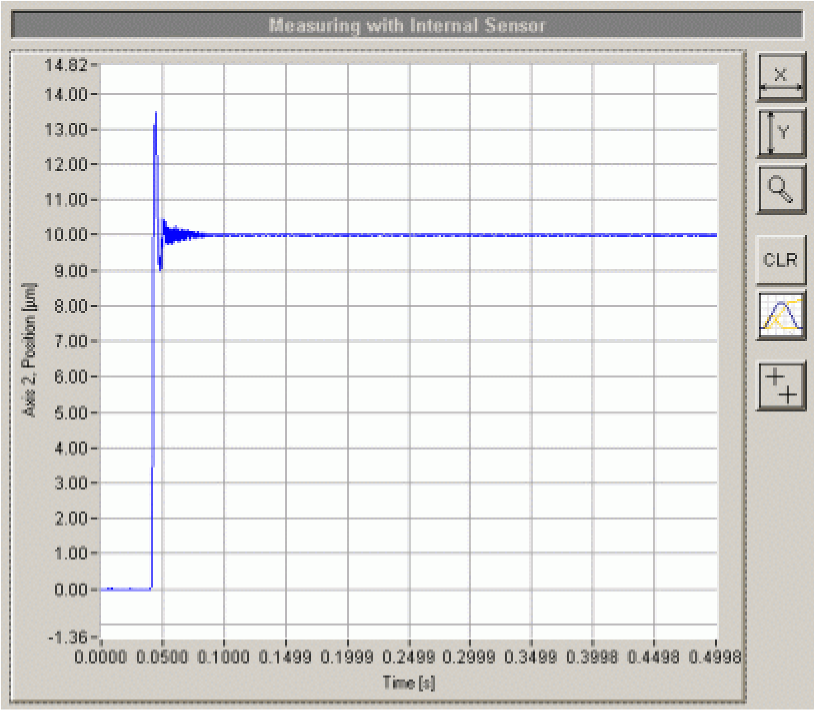
If the loop gain value is very low then the rise rate of the stage response curve will also very low:



If the loop gain value is increased then the rise rate also rises, but the overshoot will also increase. The figure below shows a step response with a small overshoot:



If the loop gain is further increased 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 stage will oscillate.

10 Return to the *Dynamic Tuner* window, refine the settings and repeat until satisfied.

11 Press *Save as Default* for each of the changed parameter groups.

9 GCS Commands

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

9.1 Format

9.1.1 Notation

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

<...>	Angle brackets indicate an argument of a command, can be an item identifier (p. 43) or a command-specific parameter
[…]	Square brackets indicate an optional entry
{…}	Braces indicate a repetition of entries, i.e. that it is possible to access more than one item (e.g. several axes) in one command line.
LF	LineFeed (ASCII char #10), is the default termination character
SP	Space (ASCII char #32)

9.1.2 GCS Syntax

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

CMD ::= character1 character2 character3 [?]

Exceptions:

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

The command mnemonic is not case-sensitive.

General:

CMD[{SP<argument>}]LF

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

Example:
Send: MOVSP1SP10.0LF
to move Axis 1 to position 10.0 µm

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

If part of a command line can not be executed, the line is not executed at all.

When all arguments are optional and are omitted, the command is executed for all possible argument values. For example,
RPALF
resets all parameters in volatile memory.

The <AxisID> argument is used for the logical axes of the controller. Depending on the controller, an axis could be identified with up to 16 characters—all alphanumeric characters and the underscore are allowed. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers supported by the E-753.

Definitions for query commands (report commands):

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

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

Reply syntax:

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

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

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

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

9.2 Command Survey

9.3 Command Reference (alphabetical)

#5 (Request Motion Status)

Description: Requests motion status of the axes. Only effective in closed-loop operation (servo ON).

Format: #5 (single ASCII character number 5)

Arguments: none

Response: The answer <uint> is bit-mapped and returned as the decimal sum of the following codes:
1=first axis is moving
2=second axis is moving
4=third axis is moving
...

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

#9 (Get Wave Generator Status)

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

The #9 (p. 112) single-character command can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode (e.g. with DDL). With WGO? you can ask for the last-commanded wave generator start options (WGO settings (p. 179)).

Format: #9 (single ASCII character number 9)

Arguments: none

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

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

#24 (Stop All Motion)

Description: Stops all motion abruptly.

This includes motion of all axes caused by move commands (MOV (p. 144), MVR (p. 145), SVA (p. 160), SVR (p. 164)), by the wave generator (WGO (p. 179)), by analog control input and AutoZero motion (ATZ (p. 117)).

Sets error code to 10.

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

This command is identical in function to STP (p. 160), but only one character must be send via the interface. Therefore #24 can also be used while the controller is performing time-consuming tasks.

Format: #24 (ASCII character 24)

Arguments: none

Response: none

Notes: When the analog input is used as control source and the axis motion is stopped with STP (p. 160) or #24 (p. 113), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis. See "How to work with the Analog Input" (p. 29) for more information.

*IDN? (Get Device Identification)

Description: Reports the device identity number.

Format: *IDN?

or

IDN?

Arguments: none

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

Notes: For E-753, *IDN? replies something like:

Physik Instrumente, E-753.1CD, 107000001, 00.01.00.01

AOS (Set Analog Input Offset)

Description: Set an offset to be added to the analog input scaled value for the given axis (corresponding parameter is Analog Target Offset, ID 0x06000501).

This offset is only effective when an input signal channel of the controller is connected to the axis for control-value generation. The connection can be made via the "ADC Channel for Target" parameter (parameter ID 0x06000500) using SPA (p. 154) or SEP (p. 152).

The control value for an axis which is connected to an input signal channel consists of:
Control Value = Analog Input Scaled Value of the Input Signal Channel + Offset

CAUTION: There is no range check for the given <Offset> value. Make sure that the resulting control value does not exceed the travel range limits of the axis (Range Limit Min, parameter ID 0x70000000 and Range Limit Max, parameter ID 0x70000001).

The AOS command changes the offset setting in volatile memory (RAM) only. On controller power-on or reboot, the offset value is loaded from the controllers non-volatile memory, and any changes made with AOS will be lost unless they have been saved.

To save the currently valid AOS setting to non-volatile memory, where it becomes the power-on default, use WPA (p. 186).

To have write access to the parameter(s), it might be necessary to switch to the proper command level using CCL (p. 119).

Format: AOS {<AxisID> <Offset>}

Arguments	<AxisID> is one axis of the controller <Offset> is the offset value, any floating point number. In closed-loop operation (servo ON), the offset is interpreted as position value in either case. In open-loop operation (servo OFF), with the default settings of the Axis-to-OutputSignalChannel matrix, the offset also corresponds numerically to axis position (see "Output Generation" (p. 55) for more information).
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	See also "Control Value Generation" (p. 51) and "How to work with the Analog Input" (p. 29).
Example:	The E-753 is in closed-loop operation (servo on) in this example, i.e. the control sources write to the target register whose current value can be read with the MOV? command. In open-loop operation, you would use SVA? instead to ask for the current value of the open-loop control register.

Send:	CCL 1 advanced
Note:	Switch to command level 1 before you change parameter values with SPA or SEP.
Send:	SPA 1 0x06000500 2
Note:	Select input signal channel 2 as control source for axis 1. Now the control value of axis 1 will result from the scaled input value of channel 2 plus the offset.
Send	AOS 1 0.0
Note:	Set digital offset of axis 1 zero.
Send	TSP? 2
Receive	2=3.22
Note:	Request the filtered and scaled value of input signal channel 2. The current value is 3.22. This value plus the offset is the current target value of axis 1.
Send	MOV? 1
Receive	1=3.22
Note:	Request the current target position of axis 1. The target position and the scaled value of input signal channel 2 are the same

	because the offset is zero.
Send	AOS 1 1.50
Note:	Set offset of axis 1 to 1.5.
Send	TSP? 2
Receive	2=3.22
Send	MOV? 1
Receive	1=4.72
Note:	Now the target value of axis 1 is the scaled value of input signal channel 2 plus the offset of axis 1.
Send	MOV 1 6.0
Send	ERR?
Receive	72
Note:	As long as the control value of axis 1 is given by an analog input, it is not possible to set the target using the MOV command.
Send:	SPA 1 0x06000500 0
Note:	Disconnect any analog input from axis 1. Now its target position can be set by the MOV command. The AOS setting is no longer effective for the control value generation of axis 1.

AOS? (Get Analog Input Offset)

Description: Get currently valid offset to the analog input scaled value for the given axis (Analog Target Offset parameter value in volatile memory (ID 0x06000501)).

Get all axes when <AxisID>=""

Format: AOS? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<Offset> is the offset value, see AOS (p. 114) for details

Troubleshooting: Illegal axis identifier

ATZ (Set Automatic Zero Point Calibration)

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

This command can be interrupted by #24 (p. 113) or STP (p. 160).

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

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

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

The success of the automatic zero-point calibration can be queried with the ATZ? command (p. 119).

The automatic zero-point calibration can take several seconds. During this time, the controller is busy and only very limited able to execute or answer commands.

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

Arguments: <AxisID> is one axis of the controller

<LowVoltage> gives the voltage value to be applied at the zero position of the axis; in volts; float. Can also be NaN ("not a number")—in this case the value of the Autozero Low Voltage parameter saved in the controller (ID 0x07000A00) will be used.

If all arguments are omitted, ATZ will be carried out for all linear axes using their AutoZero Low Voltage parameter values.

Response: none

Troubleshooting: ATZ will be not successful when an invalid axis identifier is used, e.g.

ATZ 9 NAN

or when NaN was omitted and no voltage value was given

Notes:

CAUTION: The ATZ procedure will move the axis, and the motion may cover the whole travel range. Make sure that it is safe for the stage to move.

Procedure details:

To match voltage and position as required, the axis is moved—the motion range is specified by the <LowVoltage> value given in the ATZ command (lower limit) and by the Autozero High Voltage parameter value saved in the controller (parameter ID 0x07000A01; upper limit). The final position is the zero position, with the given <LowVoltage> value applied.

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

If NaN is entered for the <LowVoltage> value, the AutoZero Low Voltage parameter value saved in the controller will be used (parameter ID 0x07000A00). You can modify this parameter with SPA (p. 154) or SEP (p. 152).

The result of the AutoZero procedure is accessible as the value of the Autozero Matched Offset parameter (parameter ID 0x02000104).

To save the current valid values of the above-mentioned parameters to non-volatile memory, where they become the power-on defaults, use WPA (p. 186). To have write access to the parameters, it might be necessary to switch to a higher command level using CCL (p. 119).

See also "Perform AutoZero" (p. 23).

Example 1:	Send:	SEP? 1 0x07000A00
	Receive:	1 0x7000a00=0.000000e+00
	Note:	The value of the AutoZero Low Voltage parameter saved in the controller is 0 V.
	Send:	ATZ 1 NaN
	Note:	Starts autozero for axis 1 with the value of the AutoZero Low Voltage parameter. Do not omit "NaN"!
	Send:	ATZ? 1
	Receive:	1
	Note:	Autozero for axis 1 was successful
Example 2:	Send:	ATZ 1 15.0
	Note:	Starts autozero for axis 1 with a voltage value of 15 V
	Send:	ATZ? 1
	Receive:	0
	Note:	Autozero for axis 1 was not successful

ATZ? (Get Automatic Zero Point Calibration)

Description: Query success or failure of the automatic zero-point calibration (see ATZ (p. 117) for details).

Format: ATZ? [{<AxisID>}]

Arguments <AxisID> is one axis of the controller

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

where

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

Troubleshooting: Illegal axis identifier

CCL (Set Command Level)

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

Format: CCL <Level> [<PSWD>]

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

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

The following command levels and passwords are valid:

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

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

Level > 1 is provided for PI service personnel only. Users can not change to a level > 1. Contact your Physik Instrumente Sales Engineer or write info@pi.ws if there seem to be problems with level 2 or higher parameters.

Response: none

Troubleshooting: Invalid password

Notes: HLP? (p. 135) lists all commands available in the current command level.

HPA? (p. 136) lists the parameters including the information about which command level allows write access to them. For more information about parameter handling see "Controller Parameters" (p. 203).

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

CCL? (Get Command Level)

Description: Get the active "command level".

Format: CCL?

Arguments: none

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

Notes: <Level> should be 0 or 1.

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

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

CST? (Get Stage Type Of Selected Axis)

Description: Returns the name of the connected stage for the queried axis.

Format: CST? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

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

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

You can change the parameter value using SPA (p. 154) or SEP (p. 152).

If the parameter should be empty, "Default_Stage_Z" will be returned.

CSV? (Get Current Syntax Version)

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

Format: CSV?

Arguments: none

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

CTO (Set Configuration of Trigger Output)

Description: Configures the trigger output conditions for the given digital output line.

The trigger output conditions will become active immediately.

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

Arguments: <TrigOutID> is one digital output line of the controller, see below for details

<CTOPam> is the CTO parameter ID in decimal format, see below for the available IDs

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

Response: None

Available output lines and trigger conditions:

<TrigOutID> corresponds to the digital output line OUT2 whose ID is 1; see "I/O" socket (p. 222).

<CTOPam> parameter IDs available for E-753:
1 = TriggerStep
2 = Axis
3 = TriggerMode
5 = MinThreshold
6 = MaxThreshold

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

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

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

for TriggerMode (default value is 2):
0 = position distance; with this TriggerMode, a trigger pulse is written whenever the axis has covered the TriggerStep distance (<CTOPam> ID 1)
2 = OnTarget; with this TriggerMode, the on-target status of the selected axis is written to the selected trigger output line (this status can also be read with the ONT? command)
3 = MinMaxThreshold; with this TriggerMode, values for MinThreshold and MaxThreshold (<CTOPam> IDs 5 and 6) must be defined. When the axis position of the selected axis is inside the band specified by the MinThreshold and MaxThreshold values, the selected trigger output line is set high, otherwise it is set low.
4 = Generator Trigger; with this TriggerMode, the trigger line action must be defined with TWS (p. 169)

for MinThreshold/MaxThreshold: position value in physical units; used for the MinMaxThreshold TriggerMode; both values must be set to form a band (no default values)

Example 1: A pulse on the digital output line OUT2 (ID 1) is to be generated whenever the stage (i.e. axis 1) has covered a distance of 0.05 µm. The following parameters must be set:

TrigOutID = 1
TriggerMode = 0
TriggerStep = 0.05

Send: CTO 1 3 0 1 1 0.05

Example 2: On the digital output line OUT2 (ID 1), pulses are to be generated at certain waveform points during the wave generator output, i.e. the trigger outputs are to be controlled by the wave generator. To do this, the trigger line must be programmed using the TWS and TWC commands, and the corresponding trigger mode is to be set by CTO.

Send: TWC

Note: Clears all trigger settings for the wave generator by switching the signal state for all points to "low". It is recommended to do this before new trigger actions are defined.

Send: TWS 1 1 1 1 2 0 1 3 0

Note: Sets trigger action for output line OUT2 (identifier is 1), at waveform point 1 it is set high, points 2 and 3 are set low.

Send: CTO 1 3 4

Note: The TriggerMode for output line OUT2 (ID 1) is set to "Generator Trigger". Now the wave generator can be started with WGO, and the trigger action will take place as specified. See also "Trigger Output Synchronized With Wave Generator" (p. 81).

CTO? (Get Configuration of Trigger Output)

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

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

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

<CTOPam>: parameter ID; see CTO

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

Response: One or more lines of the following format:

<TrigOutID> <CTOPam>"="<Value>

For <Value> see CTO.

DDL (Set DDL Table Value(s))

Description: Dynamic Digital Linearization (DDL) data load: writes data to the given DDL table.

CAUTION: Write the correct number of points to the DDL table. It must be equal to the length of the waveform which is output with the "Use DDL" option (see WGO (p. 179)) for the corresponding axis.

The DDL command will stop a running DDL initialization process.

The DDL table content will be lost when the controller is powered down or rebooted.

Format: DDL <DDLtableID> <StartPoint> {<ValueN>}

Arguments: <DDLtableID> is one DDL table of the controller

<StartPoint> is the start point in the DDL table, starts with index 1

<ValueN> is the value of point n

Response: none

Troubleshooting: Not enough memory space available: delete the content of DDL tables which are not used. See the DTC command (p. 132).

Notes: As a single-axis controller, the E-753 has only one DDL table (and only one wave generator). 65536 points are available.

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Example:

Send: DDL 1 10 2 4 6 8 10 12 14 16

Note: The values 2, 4, 6, 8, ... are written to DDL table 1, starting with the 10th point in the table

DDL? (Get DDL Table Value(s))

Description: Dynamic Digital Linearization (DDL) data query: Gets the DDL data from the specified DDL table.

Only tables with the same length can be read in the same command line. Because DDL tables do not have a common length, use the DTL? (p. 132) command to read the table length before reading the table data.

Format: DDL? [<StartPoint> [<NumberOfPoints> [{<DDLtableID>}]]]

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

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

<DDLtableID> is one DDL table of the controller

Response: The DDL data as GCS array, see the separate manual for the GCS array, SM 146E, and the example below

Troubleshooting: The DDL tables to be read with the same DDL command line have different lengths

Note: As a single-axis controller, the E-753 has only one DDL table (and only one wave generator). 65536 points are available.

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Example:

```
DDL? 1 100 1
# TYPE = 1
# SEPARATOR = 9
# DIM = 1
# SAMPLE_TIME = 0.000040
# NDATA = 100
# NAME0 = DDL of axis1
# END_HEADER
0
0
0
1
1
...
```

DPO (DDL Parameter Optimization)

Description: Dynamic Digital Linearization (DDL) Parameter Optimization. Recalculates the internal DDL processing parameters (Time Delay Max, ID 0x14000006, Time Delay Min, ID 0x14000007).

DPO usage is required when the servo parameters (notch filter frequency, servo-loop P-term, servo-loop I-term and servo-loop slew rate) have changed for an axis.

The DPO command changes the processing parameters in volatile memory (RAM) only. On controller power-on or reboot, the parameter values are loaded from the controllers non-volatile memory, and any changes made with DPO will be lost unless they have been saved.

To save the currently valid processing parameters to non-volatile memory, where they become the power-on default, use WPA (p. 186).

To have write access to the parameters, it might be necessary to switch to a higher command level using CCL (p. 119).

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Format: DPO [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response: none

DRC (Set Data Recorder Configuration)

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

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

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

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

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

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

Response: none

Notes: The number of available data recorder tables can be read with TNR? (p. 166). The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. Using SPA (p. 154) or SEP (p. 152) you can change the parameter value in the range of 1 to 8 to increase or decrease the number of data recorder tables.

The total number of points available for data recording is 65536 (Data Recorder Max Points, ID 0x16000200). These points are allocated in equal shares to the available tables (i.e. that number is given in the answer to TNR?).

With HDR? (p. 134) you will obtain a list of available record options and information about additional parameters and commands concerned with data recording.

For detailed information see "Data Recording" (p. 66).

Available record options with the appropriate data sources:

1=Target Position of axis (i.e. target value in closed-loop operation), corresponds to the MOV? response

2=Current Position of axis, corresponds to the POS? response

3=Position Error of axis

7=Control Voltage of output signal channel (after the Axis-to-OutputSignalChannel transformation but before the output type definition as axis position or piezo driving voltage), corresponds to the VOL? response

13=DDL Output of axis (DDL table values), corresponds to the DDL? response

14=Open Loop Control of axis (i.e. open-loop control value), corresponds to the SVA? response

15=Control Output of axis (before the Axis-to-OutputSignalChannel transformation)

16=Voltage of output signal channel (after the Axis-to-OutputSignalChannel transformation and the output type definition, can be axis position or piezo drive voltage)

17=Sensor Normalized of input signal channel, corresponds to the TNS? response

18= Input signal channel, after sensor filtering

19=Input signal channel, after sensor electronics linearization

20=Input signal channel, after sensor mechanics linearization, corresponds to the TSP? response

22=Slowed Target of axis (in closed-loop operation), target position after slew rate limitation

See "Control Value Generation" (p. 51) and "Output Generation" (p. 55) for more information.

Note: Output type definition as axis position or piezo drive voltage is not supported by the E-753. It has only one output signal channel which is required to drive the mechanics. Other controllers may be equipped with an additional analog output channel for which output type definition makes sense.

DRC? (get Data Recorder Configuration)

Description: Returns settings made with DRC (p. 128).

Format: DRC? [{<RecTableID>}]

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

Response: The current DRC settings:

<RecTableID>"="<Source> <RecOption>

where

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

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

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

DRR? (Get Recorded Data Values)

Description: Reading of the last recorded Data Set.

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

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

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

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

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

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

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

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

With HDR? (p. 134) you will obtain a list of available record options and trigger options and information about additional parameters and commands concerned with data recording.

For detailed information see "Data Recording" (p. 66).

Example:

DRR? 1 10 1 2 4
TYPE = 1
SEPARATOR = 9
DIM = 3
SAMPLE_TIME = 0.000040
NDATA = 10
NAME0 = Open Loop Control of axis1
NAME1 = Current Position of axis1
NAME2 = Voltage of output chan1
END_HEADER

203.953506	197.838318	41.349766
183.953506	197.838379	40.469769
183.953506	197.838394	40.469769
183.953506	197.838486	40.469769
183.953506	197.838562	40.469769
183.953506	197.838593	40.469769
183.953506	197.838684	40.469769
183.953506	197.838776	40.469769
183.953506	197.838913	40.469769
183.953506	197.839157	40.469769

DTC (Clears DDL Table Data)

Description: Clears the given DDL table.

DDL table content is also deleted when new DDL data is written to the table during an initialization process (WGO (p. 179) with "Use and reinitialize DDL" start option) or with the DDL command (p. 125). But only DTC marks DDL tables as "free" so that their memory space can be reallocated. Before new DDL data are written, it is therefore recommended to apply DTC to tables whose content is no longer used. This will avoid error messages during the next write operation.

The DTC command also stops a running DDL initialization process.

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Format: DTC {<DDLtableID>}

Arguments: <DDLtableID> is one DDL table of the controller

Response: none

DTL? (Get DDL Table Length)

Description: Get Dynamic Digital Linearization (DDL) table length.

The table length should be read before reading data with the DDL? (p. 126) command.

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Format: DTL? [{<DDLtableID>}]

Arguments: <DDLtableID> is one DDL table of the controller

Response: {<DDLtableID>"="<DDLTableLength> LF}

where

<DDLTableLength> is the length of the table in number of points

ERR? (Get Error Number)

Description: Get error code <int> of the last occurred error and reset the error to 0.

Only the last error is buffered. Therefore you should call ERR? after each command.

The error codes and their descriptions are fully listed in "Error Codes" (p. 191).

Format: ERR?

Arguments: none

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

Troubleshooting: Communication breakdown

GWD? (Get Wave Table Data)

Description: Query waveform shape for given wave table.

Depending on the waveform definition with WAV (p. 172), the wave tables may have different lengths. Due to the GCS array response format definition, it is not possible to read from tables of different lengths with one command line.

The response to GWD? does not contain any offset to the wave generator output set with WOS (p. 184).

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

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

If *all* wave tables have the same length, arguments are optional as follows:

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

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

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

<WaveTableID> is one wave table of the controller; all specified wave tables must have the same length

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

Example:

Send: GWD? 1 100 1 2 3

Receive:
```
# TYPE = 1
# SEPARATOR = 9
# DIM = 3
# SAMPLE_TIME = 0.000040
# NDATA = 100
# NAME0 = Wave Table1
# NAME1 = Wave Table2
# NAME2 = Wave Table3
# END_HEADER
0 0 0
0 0 0
0 1 0
0 1 1
0 2 1
...
```

HDR? (Get All Data Recorder Options)

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

Format: HDR?

Arguments: none

Response

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

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

#Parameters to be set with SPA

[{<ParameterID>“=“<DescriptionString>}]

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

<end of help

Example:

```
hdr?
#RecordOptions
1=Target Position of axis
2=Current Position of axis
3=Position Error of axis
7=Control Voltage of output chan
13=DDL Output of axis
14=Open Loop Control of axis
15=Control Output of axis
16=Voltage of output chan
17=Sensor Normalized of input chan
18=Sensor Filtered of input chan
19=Sensor ElecLinear of input chan
20=Sensor MechLinear of input chan
22=Slowed Target of axis
#TriggerOptions
0=Default
#Parameters to be set with SPA
0x16000000=Data Recorder Table Rate
0x16000300=Data Recorder Chan Number
end of help
```

Note: TriggerOptions = 0 (default) means that recording is triggered by the IMP (p. 143), STE (p. 158), WGO (p. 179) and WGR (p. 183) commands

HLP? (Get List Of Available Commands)

Description: List a help string which contains all commands available given the current command level.

Format: HLP?

Arguments: none

Response: List of commands available

Troubleshooting: Communication breakdown

HPA? (Get List Of Available Parameters)

Description: Responds with a help string which contains all available parameters of all levels with short descriptions. See "Controller Parameters" (p. 203) for further details.

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

SPA (p. 154) affects the parameter settings in volatile memory (RAM).

WPA (p. 186) copies parameter settings from RAM to non-volatile memory.

SEP (p. 152) writes parameter settings directly into non-volatile memory (without changing RAM settings).

RPA (p. 149) resets RAM to the values from non-volatile memory.

Format: HPA?

Arguments: none

Response {<PamID>"="<string> LF}

where

<PamID> is the ID of one parameter, hexadecimal format

<string> is a string which describes the corresponding parameter. The string has following format:

<CmdLevel>TAB<MaxItem>TAB<DataType>TAB<FunctionGroupDescription>TAB<ParameterDescription>[{TAB<PossibleValue>"="<ValueDescription>}]

where

<CmdLevel> is the command level which allows write access to the parameter value

<MaxItem> is the maximum number of items of the same type which are affected by the parameter (the meaning of "item" depends on the parameter, can be axis, output signal channel, input signal channel, the whole system or internal hardware modules)

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

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

<ParameterDescription> is the parameter name

<PossibleValue> is one value from the allowed data range

<ValueDescription> is the meaning of the corresponding value

IFC (Set Interface Parameters Temporarily)

Description: Interface configuration.

The baud rate setting for the RS-232 serial interface is specified. After IFC is sent, the new setting becomes active and the host PC interface configuration may need to be changed to maintain communication (close the current connection and re-open it with the new configuration, see "Communication" (p. 57)).

Baud rate settings made with IFC are lost when the controller is powered down. To save settings to non-volatile memory and thus make them the power-on defaults, use IFS (p. 140) instead. Alternatively, you can change the baud rate setting with SPA (p. 154) or SEP (p. 152) and save the current value with WPA (p. 186) to non-volatile memory. For the appropriate parameter ID, see below.

Format: IFC {<InterfacePam> <PamValue>}

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

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

The following interface parameters can be set:

RSBAUD
<PamValue> gives the baud rate to be used for RS-232 communication, default is 115200;
is also accessible as parameter ID 0x11000400, Uart Baudrate

Response: None

IFC? (Get Current Interface Parameters)

Description: Get the interface configuration parameter values from volatile memory.

The values from volatile memory can also be queried with SPA? (p. 157), for the corresponding parameter IDs see below.

Format: IFC? [{<InterfacePam>}]

Arguments: <InterfacePam> is the interface parameter to be queried, can be RSBAUD, IPADR, IPSTART, IPMASK and MACADR

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

where

<PamValue> gives the value of the interface parameter from volatile memory

For <InterfacePam> = RSBAUD, <PamValue> gives the current baud rate of the RS-232 communication;
is also accessible as parameter ID 0x11000400, Uart Baudrate

For <InterfacePam> = IPADR, the first four portions of <PamValue> gives the IP address used for TCP/IP communication, the last portion gives the port;
is also accessible as parameter ID 0x11000600, IP Address

For <InterfacePam> = IPSTART, <PamValue> gives the current startup behavior setting for configuration of the IP address for TCP/IP communication,
0 = use IP address defined with IPADR
1 = use DHCP to obtain IP address, if this fails, use IPADR;
is also accessible as parameter ID 0x11000800, IP Configuration

For <InterfacePam> = IPMASK, <PamValue> gives the current IP mask setting to be used for TCP/IP communication, in the form uint.uint.uint.uint;
is also accessible as parameter ID 0x11000700, IP Mask

For <InterfacePam> = MACADR, <PamValue> gives the fixed, unique address of the network hardware in the E-753;
is also accessible as parameter ID 0x11000B00, MAC Address

Notes: The IFC? answer may not show the current valid values for IP address and startup behaviour. For communication via TCP/IP, the current used IP address and the startup behaviour partially depend on the network type. For that reason, the settings for IPADR and IPSTART may be ignored:

If a DHCP server is present in the network, the IPSTART setting is ignored and the IP address is always obtained from the DHCP server.

If the E-753 is directly connected to the Ethernet card in the PC (no DHCP server is present), the current IP address of the E-753 will be as follows:
for IPSTART = 0, the IPADR setting will be used
for IPSTART = 1, the default value 192.168.0.1 will be used.

IFS (Set Interface Parameters As Default Values)

Description: Interface parameter store.

The power-on default parameters for the interface are changed in non-volatile memory, but the current active parameters are not. Settings made with IFS become active with the next power-on.

To change the baud rate setting for the RS-232 serial connection immediately (but temporarily) use IFC (p. 138).

It is also possible to change the default settings in non-volatile memory with SEP (p. 152) and to read them with the SEP? (p. 153) command. Do not use RPA (p. 149) to activate them—except of baud rate changes after which the host PC interface configuration may need to be changed— because it will not be possible to maintain communication afterwards. For the appropriate parameter IDs see below.

Warning: The number of write cycles of non-volatile memory is limited. Do not write default values except when necessary.

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

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

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

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

The following interface parameters can be set:

RSBAUD
<PamValue> gives the baud rate to be used for RS-232 communication, default is 115200;
is also accessible as parameter ID 0x11000400, Uart Baudrate

IPADR
The first four portions of <PamValue> specify the default IP address for TCP/IP communication, the last portion specifies the default port to be used, default is 192:168.0.1:50000;
is also accessible as parameter ID 0x11000600, IP Address
Note: While the IP address can be changed, the port must always be 50000!

IPSTART
<PamValue> defines the startup behavior for configuration of the IP address for TCP/IP communication,
0 = use IP address defined with IPADR
1 = use DHCP to obtain IP address, if this fails, use IPADR (default);
is also accessible as parameter ID 0x11000800, IP Configuration

IPMASK
<PamValue> gives the IP mask to be used for TCP/IP communication, in the form uint.uint.uint.uint, default is 255.255.255.0;
is also accessible as parameter ID 0x11000700, IP Mask

Response: None

Notes: For communication via TCP/IP, the current used IP address and the startup behaviour partially depend on the network type. For that reason, the settings for IPADR and IPSTART may be ignored:

If a DHCP server is present in the network, the IPSTART setting is ignored and the IP address is always obtained from the DHCP server.

If the E-753 is directly connected to the Ethernet card in the PC (no DHCP server is present), the current IP address of the E-753 will be as follows:
for IPSTART = 0, the IPADR setting will be used
for IPSTART = 1, the default value 192.168.0.1 will be used.

IFS? (Get Interface Parameters As Default Values)

Description: Get the interface configuration parameter values stored in non-volatile memory (i.e. the current power-on default)

Format: IFS? [{<InterfacePam>}]

Arguments: <InterfacePam> is the interface parameter to be queried, can be RSBAUD, IPADR, IPSTART, IPMASK or MACADR, see IFS (p. 140) for details

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

where

<PamValue> is the value of the interface parameter in non-volatile memory, see IFS for possible values

For <InterfacePam> = MACADR, <PamValue> gives the unique address of the network hardware in the E-753 (is also accessible as parameter ID 0x11000B00, MAC Address)

IMP (Start Impulse And Response Measurement)

Description: Starts performing an impulse and recording the impulse response for the given axis.

An "impulse" consists of a relative move of the specified amplitude followed by an equal relative move in the opposite direction. Irrespective of the current operating mode (servo on or off), the impulse is performed relative to the current position.

The data recorder configuration, i.e. the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 128).

The recorded data can be read with the DRR? command (p. 130).

Format: IMP <AxisID> <Amplitude>

Arguments <AxisID> is one axis of the controller

<Amplitude> is the height of the impulse

In closed-loop operation (servo ON), the given amplitude is interpreted as relative position value in either case. In open-loop operation (servo OFF), with the default settings of the Axis-to-OutputSignalChannel matrix, the amplitude also corresponds numerically to a relative axis position (see "Output Generation" (p. 55) for more information).

Response: none

Troubleshooting: The control value resulting from the specified impulse height is out of limits:

Open-loop operation: the amplitude limitation depends on the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001)
Closed-loop operation: use TMN? (p. 166) and TMX? (p. 166) to ask for the current valid travel range limits.

Motion commands like IMP are not allowed when analog control input or wave generator output are active. See "Control Value Generation" (p. 51) for details.

MOV (Set Target Position)

Description:	Set new absolute target position for given axis. During a move, a new move command resets the target to a new value and the old one may never be reached. Servo must be enabled for the commanded axis prior to using this command (closed-loop operation). This command can be interrupted by #24 (p. 113) and STP (p. 160).
Format:	MOV {<AxisID> <Position>}
Arguments	<AxisID> is one axis of the controller <Position> is the new absolute target position in physical units.
Response:	none
Troubleshooting:	Target position out of limits. Use TMN? (p. 166) and TMX? (p. 166) to ask for the current valid travel range limits. Illegal axis identifier Servo is Off for one of the axes specified. Motion commands like MOV are not allowed when analog control input or wave generator output are active on the axis. See "Control Value Generation (p. 51)" for details.

Example 1:

Send:	MOV 1 10
Note:	Axis 1 moves to 10 (target position in µm)

Example 2:

Send:	MOV 1 243
Send:	ERR?
Receive:	7
Note:	The axis does not move. The error code "7" in the reply to the ERR? command (p. 133) indicates that the target position given in the move command is out of limits.

MOV? (Get Target Position)

Description: Returns last valid commanded target position. The target position can be changed by various sources, e.g. by commands that cause motion (MOV (p. 144), MVR (p. 145), IMP (p. 143), STE (p. 158)), by the wave generator and by an analog input signal. See "Control Value Generation" (p. 51) for details.

Note that MOV? gets the commanded positions. Use POS? (p. 148) to get the current positions.

Format: MOV? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the last commanded target position in physical units (usually µm)

Troubleshooting: Illegal axis identifier

MVR (Set Target Relative To Current Position)

Description: Move given axes relative to the last commanded target position.

The new target position is calculated by adding the given value <Distance> to the last commanded target value.

Servo must be enabled for the commanded axis prior to using this command (closed-loop operation).

This command can be interrupted by #24 (p. 113) and STP (p. 160).

Format: MVR {<AxisID> <Distance>}

Arguments: <AxisID> is one axis of the controller.

<Distance> gives the distance to move; the sum of the distance and the last commanded target position is set as new target position (in physical units).

Response:	none	
Troubleshooting:	Target position out of limits. Use TMN? (p. 166) and TMX? (p. 166) to ask for the current valid travel range limits, and MOV? (p. 144) for the current target. Illegal axis identifier Servo is Off for one of the specified axes Motion commands like MVR are not allowed when analog control input or wave generator output are active for any specified axis. See "Control Value Generation" (p. 51) for details.	
Example:	Send:	MOV 1 0.5
	Note:	This is an absolute move.
	Send:	POS? 1
	Receive:	1=0.500000
	Send:	MOV? 1
	Receive:	1=0.500000
	Send:	MVR 1 2
	Note:	This is a relative move.
	Send:	POS? 1
	Receive:	1=2.500000
	Send:	MVR 1 2000
	Note:	New target position of axis 1 would exceed motion range. Command is ignored, i.e. the target position remains unchanged, and the axis does not move.
	Send:	MOV? 1
	Receive:	1=2.500000
	Send:	POS? 1
	Receive:	1=2.500000

ONT? (Get On Target State)

Description: Get on-target status of given axis.

If all arguments are omitted, gets status of all axes.

Is influenced by two parameters: settling window (On Target Tolerance, ID 0x07000900) and settling time (Settling Time, ID 0x07000901).

The on-target status is true when the current position is inside the settling window and stays there for at least the settling time. The settling window is centered around the target position.

Format: ONT? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

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

where

<uint> = "1" when the specified axis is on-target, "0" otherwise.

Troubleshooting: Illegal axis identifier

In open-loop operation (servo off), the axis will never be on target because sensor feedback is not used and hence the current on-target status can not be determined.

OVF? (Get Overflow State)

Description: Get overflow status of given axis.

If all arguments are omitted, gets status of all axes.

Overflow means that the control variables are out of range (can only happen if controller is in closed-loop operation).

Format: OVF? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

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

where

<uint> = "0" (axis is not in overflow) or "1" (axis is in overflow)

Troubleshooting: Illegal axis identifier

POS? (Get Real Position)

Description: Returns the current axis position.

If all arguments are omitted, gets current position of all axes.

To request the current position of input signal channels (sensors) in physical units, use the TSP? (p. 168) command instead.

Format: POS? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

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

where

<float> is the current axis position in physical units (e.g. µm)

Troubleshooting: Illegal axis identifier

RBT (Reboot System)

Description: Reboot system. Controller behaves just like after power-on.

Format: RBT

Arguments: none

Response: none

RPA (Reset Volatile Memory Parameters)

Description: Resets the given parameter of the given item. The value from non-volatile memory is written into volatile memory.

This procedure can take a few seconds.

Related commands:

With HPA? (p. 136) you can obtain a list of the available parameters. SPA (p. 154) affects the parameter settings in volatile memory, WPA (p. 186) writes parameter settings from volatile to non-volatile memory, and SEP (p. 152) writes parameter settings directly into non-volatile memory (without changing the settings in volatile memory).

See SPA for an example.

CAUTION: If the communication between host PC and E-753 is done via TCP/IP, do not use RPA (p. 149) after you have changed the parameters of the TCP/IP communication with IFS (p. 140) or SEP (p. 152) in non-volatile memory, because it will not be possible to maintain communication afterwards.

Format: RPA [{<ItemID> <PamID>}]

Arguments <ItemID> can be an axis identifier, an input signal or output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205).

Response: none

Troubleshooting: Illegal item identifier, wrong parameter ID

RTR (Set Record Table Rate)

Description: Sets the record table rate, i.e. the number of servo-loop cycles to be used in data recording operations (Data Recorder Table Rate parameter, ID 0x16000000). Settings larger than 1 make it possible to cover longer time periods.

The duration of the recording can be calculated as follows:

Rec. Duration (in s) = Servo Update Time (in s) * RTR value * Number of Points

where

Servo Update Time is given by parameter 0x0E000200

Number of Points is the length of the data recorder table

For more information see "Data Recording" (p. 66).

The record table rate set with RTR is saved in volatile memory (RAM) only. To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 186). Changes not saved with WPA will be lost when the controller is powered down. To have write access to the parameter, it might be necessary to switch to a higher command level using CCL (p. 119).

Format: RTR <RecordTableRate>

Arguments: <RecordTableRate> is the table rate to be used for recording operations (unit: number of servo-loop cycles), must be an integer value larger than zero

Response: None

RTR? (Get Record Table Rate)

Description: Gets the current record table rate, i.e. the number of servo-loop cycles used in data recording operations (Data Recorder Table Rate parameter value in volatile memory (ID 0x16000000)).

For more information see "Data Recording" (p. 66).

Format: RTR?

Arguments: None

Response: <RecordTableRate> is the table rate used for recording operations (unit: number of servo-loop cycles)

SAI? (Get List Of Current Axis Identifiers)

Description: Gets the axis identifiers.

See also "Accessible Items and Their Identifiers" (p. 43).

Format: SAI? [ALL]

Arguments: [ALL] is optional and provided for compatibility with controllers which allow for axis deactivation. [ALL] then ensures that the answer also includes the axes which are "deactivated" (i.e. not connected to stages).

Response: {<AxisID> LF}

<AxisID> is one axis of the controller.

SEP (Set Non-Volatile Memory Parameters)

Description: Set a parameter of a given item to a different value in non-volatile memory, where it becomes the new power-on default.

After parameters were set with SEP, you can use RPA (p. 149) to activate them (write them to volatile memory) without controller reboot.

Caution: This command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Related commands:

HPA? (p. 136) returns a list of the available parameters.

SPA (p. 154) writes parameter settings into volatile memory (without changing the settings in non-volatile memory).

WPA (p. 186) writes parameter settings from volatile to non-volatile memory.

To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119).

See SPA for an example.

Warning: The number of write cycles of non-volatile memory is limited. Do not write default values except when necessary.

Format: SEP <Pswd> {<ItemID> <PamID> <PamValue>}

Arguments <Pswd> is the password for writing to non-volatile memory, default is "100"

<ItemID> can be an axis identifier, an input signal or output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205).

<PamValue> is the value to which the given parameter of the given item is set

Response: none

Troubleshooting: Illegal item identifier, wrong parameter ID, invalid password, command level too low for write access

SEP? (Get Non-Volatile Memory Parameters)

Description: Get the value of a parameter of a given item from non-volatile memory.

With HPA? (p. 136) you can obtain a list of the available parameters and their IDs.

Format: SEP? [{<ItemID> <PamID>}]

Arguments: <ItemID> can be an axis identifier, an input signal or output signal channel, the whole system or an internal hardware module; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205).

Response: {<ItemID> <PamID>"="<PamValue> LF}

where

<PamValue> is the value of the given parameter for the given item

Troubleshooting: Illegal item identifier, wrong parameter ID

SPA (Set Temporary Memory Parameters)

Description: Set a parameter of a given item to a value in volatile memory (RAM). Parameter changes will be lost when the controller is powered down or rebooted or when the parameters are restored with RPA (p. 149).

Caution: This command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!

Do not change the current settings of the communication interface—except of the baud rate—because it will not be possible to maintain communication afterwards.

Related commands:

HPA? (p. 136) returns a list of the available parameters.

SEP (p. 152) writes parameter settings directly into non-volatile memory (without changing the settings in volatile memory).

WPA (p. 186) writes parameter settings from volatile to non-volatile memory.

RPA resets volatile memory to the value in non-volatile memory.

To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119).

Format: SPA {<ItemID> <PamID> <PamValue>}

Arguments	<ItemID> can be an axis identifier, an input signal or output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items. <PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205). <PamValue> is the value to which the given parameter of the given item is set
Response:	none
Troubleshooting:	Illegal item identifier, wrong parameter ID,value out of range, command level too low for write access

Example 1:

Send:	SPA 1 0x16000000 8
Note:	Set the Data Recorder Table Rate for the controller to 8, parameter ID written in hexadecimal format
Send:	SPA 1 369098752 2
Note:	Sets the Data Recorder Table Rate for the controller to 2, parameter ID written in decimal format

Example 2: The analog input line (input signal channel 2 in this case), is to be used as control source for axis 1. For that purpose, the corresponding coefficient in the InputSignalChannel-to-Axis matrix (Position From Sensor 2, parameter ID 0x07000501) must be set to 0 for axis 1 so that the analog input line does not participate as sensor in the axis position calculation.

Send:	CCL 1 advanced
Note:	Switch to command level 1 because this level is required for write access to the Position From Sensor 2 parameter.
Send:	SPA 1 0x07000501 0
Note:	The analog input line no longer participates in the position calculation of axis 1. The setting is made in volatile memory only.

Now make further configuration settings in volatile memory using SPA and then test the functioning of the system. See "Using the Analog Input" (p. 29) for more information. If everything is okay and you want to use this system configuration after the next power-on, save the parameter settings from volatile to non-volatile memory.

Send:	WPA 100
Note:	When WPA is used without specifying any parameters, all currently valid parameter values from volatile memory are saved.
Send:	SEP? 1 0x07000501
Receive:	1 0x7000501=0.000000e+00
Note:	Check the parameter settings in non-volatile memory.

Example 3: The task performed in example 2 can also be done in the following way, provided you are sure that the new system configuration will work:

Send: CCL 1 advanced

Note: Switch to command level 1 because this level is required for write access to the Position From Sensor 2 parameter.

Send: SEP 100 1 0x07000501 0

Note: The analog input line no longer participates in the position calculation of axis 1. The setting is made in non-volatile memory and hence is the new power-on default, but is not yet active.

Make further configuration settings in non-volatile memory using SEP. See "Using the Analog Input" (p. 29) for more information. To use the new settings immediately, you now have to load them to volatile memory (otherwise they would become active after the next power-on or reboot of the controller).

Send: RPA

Note: The new configuration is now active.

Send: SPA? 1 0x07000501

Receive: 1 0x7000501=0.000000e+00

Note: Check the parameter settings in volatile memory.

SPA? (Get Temporary Memory Parameters)

Description: Get the value of a parameter of a given item from volatile memory (RAM).

With HPA? (p. 136) you can obtain a list of the available parameters and their IDs.

Format: SPA? [{<ItemID> <PamID>}]

Arguments: <ItemID> can be an axis identifier, an input signal or output signal channel, the whole system or an internal hardware module; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205).

Response: {<ItemID> <PamID>"="<PamValue> LF}

where

<PamValue> is the value of the given parameter for the given item

Troubleshooting: Illegal item identifier, wrong parameter ID

STE (Start Step And Response Measurement)

Description: Starts performing a step and recording the step response for the given axis.

A "step" consists of a relative move of the specified amplitude. Irrespective of the current operating mode (servo on or off), the step is performed relative to the current position.

The data recorder configuration, i.e. the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 128).

The recorded data can be read with the DRR? (p. 130) command.

Format: STE <AxisID> <Amplitude>

Arguments <AxisID> is one axis of the controller

<Amplitude> is the height of the step

In closed-loop operation (servo ON), the given amplitude is interpreted as relative position value in either case. In open-loop operation (servo OFF), with the default settings of the Axis-to-OutputSignalChannel matrix, the amplitude also corresponds numerically to a relative axis position (see "Output Generation" (p. 55) for more information).

Response: none

Troubleshooting: The control value resulting from the specified step height is out of limits:

open-loop operation: the amplitude limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001)
closed-loop operation: use TMN? (p. 166) and TMX? (p. 166) to ask for the current valid travel range limits.

Motion commands like STE are not allowed when analog control input or wave generator output are active. See "Control Value Generation" (p. 51) for details.

STP (Stop All Motion)

Description: Stops all motion abruptly.

This includes motion of all axes caused by move commands (MOV (p. 144), MVR (p. 145), SVA (p. 160), SVR (p. 164)), by the wave generator (WGO (p. 179)), by analog control input and autozero motion (ATZ (p. 117)).

Sets error code to 10.

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

This command is identical in function to #24 (p. 113) which should be preferred when the controller is performing time-consuming tasks.

Format: STP

Arguments: none

Response: none

Troubleshooting: Communication breakdown

Notes: When the analog input is used as control source and the axis motion is stopped with STP (p. 160) or #24 (p. 113), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis. See "How to work with the Analog Input" (p. 29) for more information.

SVA (Set Open-Loop Axis Value)

Description: Set absolute open-loop control value to move the axis.

Servo must be switched off (open-loop operation) when using this command.

This command can be interrupted by #24 (p. 113) and STP (p. 160).

Format: SVA {<AxisID> <Amplitude>}

Arguments	<AxisID> is one axis of the controller <Amplitude> is the new absolute open-loop control value (dimensionless) The interpretation of the amplitude value depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). With the default matrix coefficients, open-loop control values numerically correspond to axis position values.
Response:	none
Troubleshooting:	The control value specified by the given amplitude is out of limits. The limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001) of the output signal channels which would be involved in the axis motion. Illegal axis identifier Servo is On for one of the specified axes Motion commands like SVA are not allowed when analog control input or wave generator output are active. See "Control Value Generation" (p. 51) for details.

Example 1:

Send:	SVA 1 10
Note:	Assumed that the default Axis-to-Output SignalChannel matrix settings are valid so that the amplitude corresponds to a position value, axis 1 moves to 10 µm, but with no position control (in open-loop operation there will be no correction of drift or other effects).

Example 2:

Send:	SVA 1 300
Send:	ERR?
Receive:	17
Note:	The axis does not move. The error code "17" reported by the ERR? command (p. 133) indicates that the open-loop value set by SVA is out of limits.

SVA? (Get Open-Loop Axis Value)

Description: Returns last valid open-loop control value of given axis. The open-loop control value is changed by multiple sources, e.g. by commands that cause motion (SVA (p. 160), SVR (p. 164), IMP (p. 143), STE (p. 158)), by the wave generator and by an analog input signal. See "Control Value Generation" (p. 51) for details.

Format: SVA? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the last commanded open-loop control value (dimensionless). The interpretation of the open-loop control value depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). By default, the matrix is set up so that control values correspond to axis position values (in µm).

Troubleshooting: Illegal axis identifier

SVO (Set Servo State)

Description: Sets servo-control state for given axes (open-loop or closed-loop operation). Whenever the servo state is changed, SVO writes a control value to the target register or the open-loop control register. See "Control Value Generation" (p. 51) for more information.

The current servo state affects the applicable move commands:
servo-control off: use SVA (p. 160) and SVR (p. 164)
servo-control on: use MOV (p. 144) and MVR (p. 145)

When servo is switched off while the axis is moving, the axis stops. Exception: When the analog input is being used as control source and servo is switched off, the axis motion will continue in open-loop mode.

Servo-control can not be switched off while the wave generator is running for the axis.

Using the Power Up Servo On Enable parameter (ID 0x07000800), you can configure the controller so that servo is automatically switched on upon power-on or reboot. To do this, set the value of the parameter to 1 in non-volatile memory (using SEP (p. 152) or SPA (p. 154)+ WPA (p. 186)).

Format: SVO {<AxisID> <ServoState>}

Arguments: <AxisID> is one axis of the controller

<ServoState> can have the following values:
0 = servo off (open-loop operation)
1 = servo on (closed-loop operation)

Response: none

Troubleshooting: Illegal axis identifier

SVO? (Get Servo State)

Description: Gets servo-control state for given axes.

If all arguments are omitted, gets status of all axes.

Format: SVO? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<ServoState> is the current servo state of the axis:
0 = servo off (open-loop operation)
1 = servo on (closed-loop operation)

Troubleshooting: Illegal axis identifier

SVR (Set Relative Open-Loop Axis Value)

Description: Set open-loop control value relative to the current open-loop control value to move the axis.

The new open-loop control value is calculated by adding the given value <Difference> to the last commanded open-loop control value.

Servo must be off when using this command (open-loop operation).

This command can be interrupted by #24 (p. 113) and STP (p. 160).

Format: SVR {<AxisID> <Difference>}

Arguments <AxisID> is one axis of the controller

<Difference> is the value which is added to the current open-loop control value (dimensionless)
The interpretation of the difference value depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). With the default matrix coefficients, open-loop control values numerically correspond to axis position values.

Response: none

Troubleshooting: The specified control value is out of limits. The limitation results from the voltage limit parameters (IDs 0x0B000007, 0x0B000008, 0x0C000000 and 0x0C000001) of the output signal channels which would be involved in the axis motion.

Illegal axis identifier

Servo is On for one of the specified axes

Motion commands like SVR are not allowed when analog control input or wave generator output are active. See "Control Value Generation" (p. 51) for details.

TAD? (Get ADC Value Of Input Signal)

Description: Get the current value from the specified input signal channel's A/D converter. This value represents the digitized signal value without filtering and linearization. Using this command it is possible to check for sensor overflow.

Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 45)). TAD? reads the values for the individual input signal channels, not for a logical axis.

Format: TAD? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<float> is the current A/D value, dimensionless

TLT? (Get Number of DDL Tables)

Description: Tell number of Dynamic Digital Linearization (DDL) tables available on the controller.

Format: TLT?

Arguments: none

Response <uint> is the number of DDL tables which are available

TMN? (Get Minimum Commandable Position)

Description: Get the minimum commandable position in physical units.

The minimum commandable position is defined by the Range Limit min parameter, ID 0x07000000.

Format: TMN? [{ <AxisID>}]

Arguments: <AxisID>: is one axis of the controller

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

where

<float> is the minimum commandable position in physical units

TMX? (Get Maximum Commandable Position)

Description: Get the maximum commandable position in physical units.

The maximum commandable position is defined by the Range Limit max parameter, ID 0x07000001.

Format: TMX? [{ <AxisID>}]

Arguments: <AxisID>: is one axis of the controller

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

where

<float> is the maximum commandable position in physical units

TNR? (Get Number of Record Tables)

Description: Get the number of data recorder tables currently available on the controller.

Format: TNR?

Arguments: none

Response <uint> is the number of data recorder tables which are currently available

Notes: The answer gives the value of the Data Recorder Chan Number parameter, ID 0x16000300. You can change the parameter value to increase or decrease the number of data recorder tables.

For more information see "Data Recording" (p. 66).

TNS? (Get Normalized Input Signal Value)

Description: Get the normalized value for the given input signal channel. This value is internally the input for the mechanics linearization.

Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 45)). TNS? reads the values for the individual input signal channels, not for a logical axis.

Format: TNS? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<float> is the normalized value ranging from controller specific minimum to maximum (e.g. -100 to 100), dimensionless

TPC? (Get Number of Output Signal Channels)

Description: Get the number of output signal channels available on the controller.

The piezo channels are a subset of the number of output signal channels. They are used to drive the piezo actuators in the mechanics and can be queried with the Number Of Piezo Channels parameter, ID 0x0E000B04. See "Accessible Items and Their Identifiers" (p. 43) for more information.

Format: TPC?

Arguments: none

Response <uint> is the number of output signal channels which are available; the answer gives the value of the Number Of Output Channels parameter, ID 0x0E000B01

TSC? (Get Number of Input Signal Channels)

Description: Get the number of input signal channels available on the controller.

The sensor channels are a subset of the number of input signal channels. They are used for the sensors which are integrated in the mechanics and can be queried with the Number Of Sensor Channels parameter, ID 0x0E000B03. See "Accessible Items and Their Identifiers" (p. 43) for more information.

Format: TSC?

Arguments: none

Response <uint> is the number of input signal channels which are available; the answer gives the value of the Number Of Input Channels parameter, ID 0x0E000B00

TSP? (Get Input Signal Position Value)

Description: Requests the current position of the selected input signal channel in physical units (e.g. µm).

Multiple input signal channels (sensors) could be involved in the control of one logical axis (see "Processing Steps" (p. 45)). TSP? reads the position values for the individual input signal channels, not for a logical axis. To get the current position of an axis, use POS? (p. 148) instead.

Format: TSP? [{<InputSignalID>}]

Arguments: <InputSignalID> is one input signal channel of the controller

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

where

<float> is the current position of the input signal channel, in physical units

TWC (Clear All Wave Related Triggers)

Description: Clears all output trigger settings for the wave generators (the settings made with TWS (p. 169)) by switching the signal state for all points to "low".

For a detailed description see "Wave Generator" (p. 70) and "Using Trigger Input and Output" (p. 38).

Format: TWC

Arguments: none

Response: none

TWG? (Get Number of Wave Generators)

Description: Get the number of wave generators available on the controller.

Format: TWG?

Arguments: none

Response <uint> is the number of wave generators which are available

TWS (Set Trigger Line Action To Waveform Point)

Description: Associates output trigger line and trigger line action (signal state high or low) with waveform point.

The power-on default state of all points is low. Afterwards, the signal state of the trigger output line can be switched to "low" for all points using the TWC (p. 169) command. It is recommended to use TWC before trigger actions are set with TWS.

Generator trigger mode must be activated for the selected trigger output line with the CTO command (p. 122).

See also "Wave Generator" (p. 70) and "Using Trigger Input and Output" (p. 38).

Format: TWS {<TrigOutID> <PointNumber> <Switch>}

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

<PointNumber> is one point in the waveform, starts with index 1
You can calculate the time for the point as follows:
time = generator cycle time * PointNumber
with
generator cycle time = Servo Update Time * WTR value
where
Servo Update Time in seconds is given by parameter 0x0E000200
WTR (wave table rate) value is the number of servo cycles the output of a waveform point takes, default is 1

<Switch> is the signal state of the trigger output line: 0 = low, 1 = high

Response: None

Example:

Send: TWS 1 1 1 1 2 0 1 3 0

Note: Sets trigger actions for the output line OUT2 (identifier 1), at waveform point 1 it is set high, points 2 and 3 are set low.

VEL (Set Velocity)

Description: Set velocity of given axes.

The VEL setting only takes effect when the given axis is in closed-loop operation (servo on).

VEL can be changed while the axis is moving.

Format: VEL {<AxisID> <Velocity>}

Arguments: <AxisID> is one axis of the controller

<Velocity> is the velocity value in physical units/s and must be > 0.

Response: none

Troubleshooting: Illegal axis identifiers, axis is under joystick control (via host PC)

Notes: VEL concerns the value of the Servo Loop Slew-Rate parameter, ID 0x07000200.

The velocity set with VEL is saved in volatile memory (RAM) only. To save the currently valid value to non-volatile memory, where it becomes the power-on default, you must use WPA (p. 186). Changes not saved with WPA will be lost when the controller is powered down. To have write access to the parameter, it might be necessary to switch to a higher command level using CCL (p. 119).

VEL? (Get Velocity)

Description: Get the current value of the Servo Loop Slew-Rate parameter, ID 0x07000200, from volatile memory.

If all arguments are omitted, gets current value of all axes.

Format: VEL? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

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

where

<float> is the current active velocity value in physical units / s.

VOL? (Get Voltage Of Output Signal Channel)

Description: Read the current voltage value of the given output signal channel (with piezo amplifier channels, this is the drive voltage output for the piezo actuators).

Multiple output signal channels (piezo amplifiers) can be involved in the motion of one logical axis (see "Processing Steps" (p. 45)). Note that VOL? reads the current voltage values for the individual output signal channels, not for a logical axis.

Format: VOL? [{<OutputSignalID>}]

Arguments: <OutputSignalID> is one output signal channel of the controller

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

where

<float> is the current voltage value in V

WAV (Set Waveform Definition)

Description:

Define waveform of given type for given wave table.

To allow for flexible definition, a waveform (wave table contents) can be built up by adding "segments". Each segment is defined with a separate WAV command. To add a segment, the <AppendWave> argument (see below) is used to concatenate the new segment to the existing wave table contents. (To change individual segments later, or to modify their order, the complete waveform must be recreated segment-by-segment.)

A segment can be based on predefined "curve" shapes (see the <WaveType> argument below).

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

The waveform values are absolute values.

The frequency of the wave generator output depends, among other factors, on the wave table length. When you create waveforms, keep in mind that the usable frequency is limited by the available amplifier power. If the frequency is too high, overheating of the amplifier(s) can occur, and the piezo voltage output will be deactivated automatically. If that occurs, the piezo stage will not move although communication with the controller is still possible (see also "Operating Limits" (p. 224)).

The duration of one output cycle for the waveform can be calculated as follows:

Output Duration = Servo Update Time * WTR value * Number of Points

where

Servo Update Time in seconds is given by parameter 0x0E000200

WTR (wave table rate) value gives the number of servo cycles the output of a waveform point lasts, default is 1

Number of Points is the length of the wave table (which is the sum of the lengths of all segments in this table)

See "How to work with the Wave Generator" (p. 70) for more information.

Format:

WAV <WaveTableID> <AppendWave> <WaveType> <WaveTypeParameters>

Arguments: <WaveTableID> is the wave table identifier.

<AppendWave> This can be "X" or "&":
"X" clears the wave table and starts writing with the first point in the table.
"&" appends the defined segment to the already existing wave table contents (i.e. concatenates a segment to lengthen the waveform).

<WaveType> The type of curve used to define the segment. This can be one of
"PNT" (user-defined curve)
"SIN_P"(inverted cosine curve)
"RAMP" (ramp curve)
"LIN" (single scan line curve)

<WaveTypeParameters> stands for the parameters of the curve and can be as follows:

For "PNT":

<WaveStartPoint> <WaveLength> {<WavePoint>}

<WaveStartPoint> The index of the starting point. Must be 1.

<WaveLength> The length of the user-defined curve in points. The segment length, i.e. the number of points written to the wave table, is identical to the <WaveLength> value.

<WavePoint> The value of one single point.

For "SIN_P":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint> <CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table.

<Amp>: The amplitude of the sine curve.

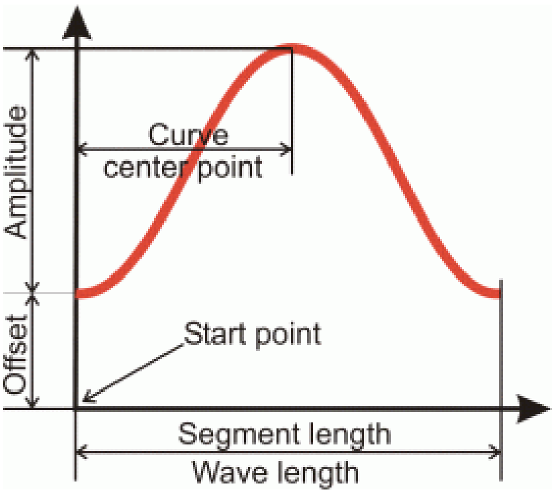
<Offset>: The offset of the sine curve.

<WaveLength>: The length of the sine curve in points.

<StartPoint>: The index of the starting point of the sine curve in the segment. Gives the phase shift. Lowest possible value is 0.

<CurveCenterPoint>: The index of the center point of the sine curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for more examples see "Defining Waveforms" (p. 75)):



For "RAMP":

<SegLength> <Amp> <Offset> <WaveLength>
<StartPoint> <SpeedUpDown> <CurveCenterPoint>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table.

<Amp>: The amplitude of the ramp curve.

<Offset>: The offset of the ramp curve.

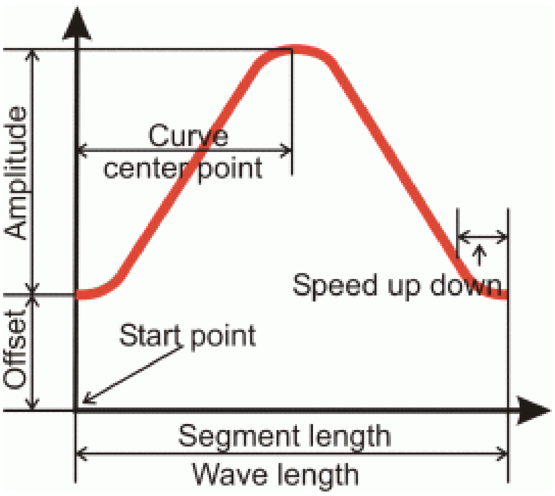
<WaveLength>: The length of the ramp curve in points.

<StartPoint>: The index of the starting point of the ramp curve in the segment. Gives the phase shift. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

<CurveCenterPoint>: The index of the center point of the ramp curve. Determines if the curve is symmetrical or not. Lowest possible value is 0.

Example (for more examples see "Defining Waveforms" (p. 75)):



For "LIN":

<SegLength> <Amp> <Offset> <WaveLength> <StartPoint> <SpeedUpDown>

<SegLength>: The length of the wave table segment in points. Only the number of points given by <SegLength> will be written to the wave table.

<Amp>: The amplitude of the scan line.

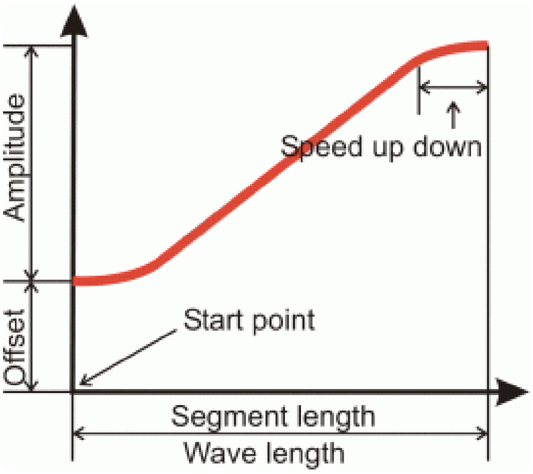
<Offset>: The offset of the scan line.

<WaveLength>: The length of the single scan line curve in points.

<StartPoint>: The index of the starting point of the scan line in the segment. Lowest possible value is 0.

<SpeedUpDown>: The number of points for speed-up and slow-down.

Example (for more examples see "Waveform Definition" (p. 75)):



Note for the Sin_P, RAMP and LIN wave types:
If the <SegLength> value is larger than the <WaveLength> value, the missing points in the segment are filled with the endpoint value of the curve.

Response: None

Troubleshooting: Invalid wave table identifier

The total number of points for the waveform (which may consist of several segments) exceeds the available number of points.

WAV? (Get Waveform Definition)

Description: Get the value of a wave parameter for a given wave table.

See "How to work with the Wave Generator (p. 70)" for more information.

Format: WAV? [{<WaveTableID> <WaveParameterID>}]

Arguments: <WaveTableID> is the wave table identifier.

<WaveParameterID> is the wave parameter ID, 1 = current wave table length in number of points; more parameters may be defined in the future

Response: {<WaveTableID> <WaveParameterID>"="<float> LF}

where

<float> depends on the <WaveParameterID>; gives the current number of waveform points in the wave table for <WaveParameterID> = 1

Troubleshooting: Invalid wave table identifier

WCL (Clear Wave Table Data)

Description: Clears the content of the given wave table.

As long as a wave generator is running, it is not possible to clear the connected wave table.

For a detailed description see "Wave Generator" (p. 70).

Format: WCL {<WaveTableID>}

Arguments: <WaveTableID> is the wave table identifier.

Response: none

WGC (Set Number Of Wave Generator Cycles)

Description: Sets the number of output cycles for the given wave generator (the output itself is started with WGO (p. 179)).

If the digital input line 2 is used to trigger the wave generator output (see WGO for details), the count of output cycles continues with each generator restart. The generator will be stopped when the number of cycles given by WGC are completed, irrespective of any further trigger pulses.

For a detailed description see "Wave Generator" (p. 70).

Format: WGC {<WaveGenID> <Cycles>}

Arguments: <WaveGenID> is the wave generator identifier

<Cycles> is the number of wave generator output cycles. If cycles = 0 then the waveform is output without period limitation until it is stopped by WGO or #24 (p. 113) or STP (p. 160).

Response: None

WGC? (Get Number Of Wave Generator Cycles)

Description: Gets the number of output cycles set for the given wave generator.

For a detailed description see "Wave Generator" (p. 70).

Format: WGC? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<Cycles> LF}

where

<Cycles> is the number of wave generator output cycles set with WGC (p. 178).

WGO (Set Wave Generator Start/Stop Mode)

Description: Start and stop the specified wave generator in the given mode. In addition, one data recording cycle is started.

The number of output cycles can be limited by WGC (p. 178).

Using the WTR command (p. 189), you can lengthen the individual output cycles of the waveform.

The data recorder configuration can be made with DRC (p. 128). Recording can be restarted with WGR (p. 183).

Keep in mind that wave generator output will continue even if the terminal or the program from which it was started is quit. The wave generator output will also continue if the high voltage output is automatically deactivated due to amplifier overheating. I.e. if a certain number of output cycles was set, the output may be already finished when the high voltage output is reactivated.

The #9 (p. 112) single-character command can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode (e.g. with DDL). With WGO? you can ask for the last-commanded wave generator start options (WGO settings (p. 179)).

For more information see "Wave Generator" (p. 70).

For a detailed description of the DDL feature see "Dynamic Digital Linearization (DDL)" (p. 85).

Format: WGO {<WaveGenID> <StartMode>}

Arguments: <WaveGenID> is the wave generator identifier

<StartMode> is the start mode for the specified wave generator.
In the WGO command, you supply the start mode in hex or decimal format. When no bits are set (<StartMode> = 0), there is no wave generator output for the associated axis.
Note that bit 6 (0x40 or 64), bit 7 (0x80 or 128) and bit 8 (0x100 or 256) cannot start the wave generator output by themselves. They simply specify certain start options and must always be combined with one of the start modes specified in bit 0 (0x1 or 1) or bit 1 (0x2 or 2). See the examples below.

The start mode values in detail:

0: wave generator output is stopped. You can also use #24 (p. 113) or STP (p. 160) to stop the wave generator output, but WGO? (p. 183) will then still report the last commanded start mode.

bit 0 = 0x1 (hex format) or 1 (decimal format):
start wave generator output immediately, synchronized by servo cycle

bit 1 = 0x2 (hex format) or 2 (decimal format):
start wave generator output triggered by external signal, synchronized by servo cycle.
To provide the external signal, the digital input lines IN1 or IN2 can be used (see "I/O" socket (p. 222)).
If IN1 is used: The wave generator output starts with the first rising edge which is detected on this input line.
If IN2 is used: The generator output starts with the first rising edge which is detected on this input line, and it will be stopped when a falling edge is detected on this line. With the next rising edge, the generator output will continue at the waveform point where it was stopped. Starting and stopping the wave generator this way can be repeated indefinitely.
If output cycle limitations were made with WGC (p. 178): with each generator restart the count of output cycles continues, and the generator will be stopped when the given number of cycles are completed, irrespective of any further trigger pulses.

It is possible to mix the usage of both digital input lines.

bit 6 = 0x40 (hex format) or 64 (decimal format):
use and reinitialize DDL; start option.
The dynamic digital linearization (DDL) feature is used and reinitialized. It is recommended to start the DDL initialization for all axes at the same time. Each new initialization will stop all running initialization processes. The initialization process is also stopped by the DDL command (p. 125).

bit 7 = 0x80 (hex format) or 128 (decimal format):
use DDL; start option.
The dynamic digital linearization feature is used.

bit 8 = 0x100 (hex format) or 256 (decimal format):
wave generator started at the endpoint of the last cycle; start option.
The second and all subsequent output cycles each start at the endpoint of the preceding cycle. The final position is the sum of the endpoint of the last output cycle and any offset defined with WAV (p. 172) for the waveform.

Response: None

Troubleshooting: Invalid wave generator identifier

There is no wave table connected to the wave generator. Use WSL (p. 187) to connect a wave table.

Wave generator output and analog control input:
It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In that case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be commanded by analog control input, you can stop the wave generator output, but not restart it.
Wave generator output and move commands:
When the wave generator output is active, move commands like MOV (p. 144) or SVA (p. 160) are not allowed for the associated axis.
See "Control Value Generation" (p. 51) for details.

Example 1: Wave generator 1 is to be used with the DDL feature, i.e. bit 7 on, contributing a value of 0x80 (dec.: 128) to <StartMode>. Because bit 7 is only a "start option" and does not actually start the wave generator output, a "start mode" ("immediately" or "triggered by external signal") must be chosen in addition. In this example, the wave generator is to be started by an external trigger signal, so bit 2 must be turned on, contributing 0x2 (dec.: 2), obtaining a <StartMode> value of 0x82 (dec.: 130).

Send the following WGO command, with the <StartMode> given in hex format:

WGO 1 0x82

The same command with <StartMode> given in decimal format:

WGO 1 130

Example 2: Wave generator 1 is to be started with the "Use and reinitialize DDL" option (bit 6, value 0x40; dec.: 64). Furthermore the option "start at the endpoint of the last cycle" is to be active (bit 8, value 0x100; dec.: 256). The start mode is to be "immediately" (bit 0, value 0x1; dec.: 1). Hence the resulting <StartMode> value is in hex format

0x40 + 0x100 + 0x1 = 0x141

or in dec format

64 + 256 + 1 = 321

Send

WGO 1 0x141

or

WGO 1 321

WGO? (Get Wave Generator Start/Stop Mode)

Description: Get the start/stop mode of the given wave generator.

The #9 (p. 112) single-character command can be used to query the current activation state of the wave generators. The reply shows if a wave generator is running or not, but does not contain any information about the wave generator start mode (e.g. with DDL). With WGO? you can ask for the last-commanded wave generator start options (WGO settings (p. 179)).

Note that #24 (p. 113) or STP (p. 160) stop the wave generator output, but do not reset the start/stop mode settings so that WGO? will still report the start mode which was set by the last WGO (p. 179) command.

For more information see "Wave Generator" (p. 70).

Format: WGO? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<StartMode> LF}

where

<StartMode> is the last commanded start mode of the wave generator, in decimal format. The value may be the sum of several start options and one start mode. See the WGO command description for details.

WGR (Starts Recording In Sync With Wave Generator)

Description: Restarts recording when the wave generator is running (a first data recording cycle is started with the WGO command (p. 179) which starts the wave generator output).

The data recorder configuration can be made with DRC (p. 128). The recorded data can be read with the DRR? command (p. 130).

For more information see "Wave Generator" (p. 70) and "Data Recording" (p. 66).

Format: WGR

Arguments: None

Response: None

WOS (Set Wave Generator Output Offset)

Description: Sets an offset to the output of a wave generator. The current wave generator output is then created by adding the offset value to the current wave value:

Generator Output = Offset + Current Wave Value

Do not confuse the output-offset value set with WOS with the offset settings specified during waveform creation with WAV (p. 172). While the WAV offset affects only one segment (i.e. only one waveform), the WOS offset is added to all waveforms which are output by the given wave generator.

WOS sets the value of the Wave Offset parameter, ID 0x1300010b, in volatile memory. You can change this parameter also with SPA (p. 154) or SEP (p. 152) and save the value with WPA (p. 186) to non-volatile memory, where it becomes the power-on default. To have write access to the parameter, it might be necessary to switch to a higher command level using CCL (p. 119).

If the wave generator is started with the option "start at the endpoint of the last cycle", the E-753 at the end of each output cycle equates the volatile value of the Wave Offset parameter with the current generator output.

Deleting wave table content with WCL (p. 177) has no effect on the settings for the wave generator output offset.

For more information see "Wave Generator" (p. 70).

Format: WOS {<WaveGenID> <Offset>}

Arguments: <WaveGenID> is the wave generator identifier

<Offset> is the wave generator output offset, any float number.
In closed-loop operation (servo ON), the offset is interpreted as position value in either case. In open-loop operation (servo OFF), the interpretation of the offset depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). With the default matrix coefficients, open-loop control values numerically correspond to axis position values.

Response: None

WOS? (Get Wave Generator Output Offset)

Description: Reads the current value of the offset which is added to the wave generator output (Wave Offset parameter value in volatile memory (ID 0x1300010b) . This value results either from WOS (p. 184) / SPA (p. 154) / SEP (p. 152) settings, or from internal calculation during the wave generator output; see WOS for details.

For more information see also "Wave Generator" (p. 70).

Format: WOS? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<Offset> LF}

where

<Offset> is the current wave generator output offset.
In closed-loop operation (servo ON), the offset is interpreted as position value in either case. In open-loop operation (servo OFF), the interpretation of the offset depends on the settings of the Axis-to-OutputSignalChannel matrix (see "Output Generation" (p. 55) for more information). With the default matrix coefficients, open-loop control values numerically correspond to axis position values.

WPA (Save Parameters To Non-Volatile Memory)

Description: Write the currently valid value of a parameter of a given item from volatile memory (RAM) to non-volatile memory. The values saved this way become the power-on defaults.

Caution: If current parameter values are incorrect, the system may malfunction. Be sure that you have the correct parameter settings before using the WPA command.

RAM settings not saved with WPA will be lost when the controller is powered down or rebooted or when RPA (p. 149) is used to restore the parameters.

With HPA? (p. 136) you can obtain a list of all available parameters.

Use SPA? (p. 154) to check the current parameter settings in volatile memory.

Parameters can be changed in volatile memory with SPA (p. 154), AOS (p. 114), ATZ (p. 117), DPO (p. 127), IFC (p. 138), RTR (p. 150), VEL (p. 170), WOS (p. 184) and WTR (p. 189).

When WPA is used without specifying any arguments except of the password, all currently valid parameter values are saved.

To have write access to the parameter(s), it might be necessary to switch to a higher command level using CCL (p. 119).

See SPA for an example.

Warning: The number of write cycles of non-volatile memory is limited. Do not write default values except when necessary.

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

Arguments	<Pswd> is the password for writing to non-volatile memory, default is "100" <ItemID> can be an axis identifier, an input signal or output signal channel or the whole system; the item type depends on the parameter, see "Parameter Overview" (p. 205) for the item type concerned. See "Accessible Items and Their Identifiers" (p. 43) for the identifiers of the items. <PamID> is the parameter ID, can be written in hexadecimal or decimal format. Valid parameter IDs are given in "Parameter Overview" (p. 205).
Response:	none
Troubleshooting:	Illegal item identifier, wrong parameter ID, invalid password, command level too low for write access

WSL (Set Connection Of Wave Table To Wave Generator)

Description:	Wave table selection: connects a wave table to a wave generator or disconnects the selected generator from any wave table. Two or more generators can be connected to the same wave table, but a generator can not be connected to more than one wave table. Deleting wave table content with WCL (p. 177) has no effect on the WSL settings. As long as a wave generator is running, it is not possible to change its wave table connection. For more information see "Wave Generator" (p. 70).
Format:	WSL {<WaveGenID> <Wave TableID>}
Arguments:	<WaveGenID> is the wave generator identifier <WaveTableID> is the wave table identifier. If <WaveTableID> = 0 the selected generator is disconnected from any wave table.
Response:	None

WSL? (Get Connection Of Wave Table To Wave Generator)

Description: Get current wave table connection settings for the specified wave generator.

For more information see "Wave Generator" (p. 70).

Format: WSL? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier

Response: {<WaveGenID>"="<WaveTableID> LF}

where

<WaveTableID> is the wave table identifier. If <WaveTableID> = 0, no wave table is connected to the wave generator.

WTR (Set Wave Generator Table Rate)

Description: Set wave generator table rate and interpolation type:

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

WTR sets the value of the Wave Generator Table Rate parameter, ID 0x13000109, in volatile memory. You can change this parameter also with SPA (p. 154) or SEP (p. 152) and save the value to non-volatile memory with WPA (p. 186). The value is always valid for the whole system and can not be set separately for individual wave generators. The value of the parameter in volatile memory can be read with the WTR? command (p. 189).

WTR also sets the type of interpolation to use for the wave generator output. If the Wave Generator Table Rate is larger than 1, interpolation helps to avoid sudden position jumps of an axis controlled by the wave generator.

For more information see "Wave Generator" (p. 70). An application example can be found in "Modifying the Wave Generator Table Rate" (p. 80).

Format: WTR {<WaveGenID> <WaveTableRate> <InterpolationType>}

Arguments: <WaveGenID> is the wave generator identifier, must be zero which means that all wave generators are selected.

<WaveTableRate> is the table rate to be used for wave generator output (unit: number of servo-loop cycles), must be an integer value larger than zero

<InterpolationType> When a wave generator table rate higher than 1 is set, this option can be used to apply interpolation to the wave generator output between wave table points. The following interpolation types can be selected:
0 = no interpolation
1 = straight line (default)

Response: None

WTR? (Get Wave Generator Table Rate)

Description: Gets the current wave generator table rate, i.e. the number of servo-loop cycles used for wave generator output (Wave Generator Table Rate parameter value in volatile memory (ID 0x13000109)). Gets also the interpolation type used with table rate values > 1.

For more information see "Wave Generator" (p. 70). An application example can be found in "Modifying the Wave Generator Table Rate" (p. 80).

Format: WTR? [{<WaveGenID>}]

Arguments: <WaveGenID> is the wave generator identifier; if zero, all wave generators are queried

Response: {<WaveGenID>"="<WaveTableRate> <InterpolationType> LF}

where

<WaveTableRate> is the table rate used for wave generator output (unit: number of servo-loop cycles)

<InterpolationType> interpolation type applied to outputs between wave table points when a wave generator table rate higher than 1 is set:
0 = no interpolation
1 = straight line

9.4 Error Codes

The error codes listed here are those of the PI General Command Set. As such, some may be not relevant to your controller and will simply never occur.

Controller Errors

0	PI_CNTR_NO_ERROR	No error
1	PI_CNTR_PARAM_SYNTAX	Parameter syntax error
2	PI_CNTR_UNKNOWN_COMMAND	Unknown command
3	PI_CNTR_COMMAND_TOO_LONG	Command length out of limits or command buffer overrun
4	PI_CNTR_SCAN_ERROR	Error while scanning
5	PI_CNTR_MOVE_WITHOUT_REF_OR_NO_SERVO	Unallowable move attempted on unreferenced axis, or move attempted with servo off
6	PI_CNTR_INVALID_SGA_PARAM	Parameter for SGA not valid
7	PI_CNTR_POS_OUT_OF_LIMITS	Position out of limits
8	PI_CNTR_VEL_OUT_OF_LIMITS	Velocity out of limits
9	PI_CNTR_SET_PIVOT_NOT_POSSIBLE	Attempt to set pivot point while U,V and W not all 0
10	PI_CNTR_STOP	Controller was stopped by command
11	PI_CNTR_SST_OR_SCAN_RANGE	Parameter for SST or for one of the embedded scan algorithms out of range
12	PI_CNTR_INVALID_SCAN_AXES	Invalid axis combination for fast scan

13	PI_CNTR_INVALID_NAV_PARAM	Parameter for NAV out of range
14	PI_CNTR_INVALID_ANALOG_INPUT	Invalid analog channel
15	PI_CNTR_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
16	PI_CNTR_INVALID_STAGE_NAME	Unknown stage name
17	PI_CNTR_PARAM_OUT_OF_RANGE	Parameter out of range
18	PI_CNTR_INVALID_MACRO_NAME	Invalid macro name
19	PI_CNTR_MACRO_RECORD	Error while recording macro
20	PI_CNTR_MACRO_NOT_FOUND	Macro not found
21	PI_CNTR_AXIS_HAS_NO_BRAKE	Axis has no brake
22	PI_CNTR_DOUBLE_AXIS	Axis identifier specified more than once
23	PI_CNTR_ILLEGAL_AXIS	Illegal axis
24	PI_CNTR_PARAM_NR	Incorrect number of parameters
25	PI_CNTR_INVALID_REAL_NR	Invalid floating point number
26	PI_CNTR_MISSING_PARAM	Parameter missing
27	PI_CNTR_SOFT_LIMIT_OUT_OF_RANGE	Soft limit out of range
28	PI_CNTR_NO_MANUAL_PAD	No manual pad found
29	PI_CNTR_NO_JUMP	No more step-response values
30	PI_CNTR_INVALID_JUMP	No step-response values recorded
31	PI_CNTR_AXIS_HAS_NO_REFERENCE	Axis has no reference sensor
32	PI_CNTR_STAGE_HAS_NO_LIM_SWITCH	Axis has no limit switch
33	PI_CNTR_NO_RELAY_CARD	No relay card installed
34	PI_CNTR_CMD_NOT_ALLOWED_FOR_STAGE	Command not allowed for selected stage(s)
35	PI_CNTR_NO_DIGITAL_INPUT	No digital input installed
36	PI_CNTR_NO_DIGITAL_OUTPUT	No digital output configured
37	PI_CNTR_NO_MCM	No more MCM responses
38	PI_CNTR_INVALID_MCM	No MCM values recorded
39	PI_CNTR_INVALID_CNTR_NUMBER	Controller number invalid
40	PI_CNTR_NO_JOYSTICK_CONNECTED	No joystick configured
41	PI_CNTR_INVALID_EGE_AXIS	Invalid axis for electronic gearing, axis can not be slave
42	PI_CNTR_SLAVE_POSITION_OUT_OF_RANGE	Position of slave axis is out of range
43	PI_CNTR_COMMAND_EGE_SLAVE	Slave axis cannot be commanded directly when electronic gearing is enabled
44	PI_CNTR_JOYSTICK_CALIBRATION_FAILED	Calibration of joystick failed

45	PI_CNTR_REFERENCING_FAILED	Referencing failed
46	PI_CNTR_OPM_MISSING	OPM (Optical Power Meter) missing
47	PI_CNTR_OPM_NOT_INITIALIZED	OPM (Optical Power Meter) not initialized or cannot be initialized
48	PI_CNTR_OPM_COM_ERROR	OPM (Optical Power Meter) Communication Error
49	PI_CNTR_MOVE_TO_LIMIT_SWITCH_FAILED	Move to limit switch failed
50	PI_CNTR_REF_WITH_REF_DISABLED	Attempt to reference axis with referencing disabled
51	PI_CNTR_AXIS_UNDER_JOYSTICK_CONTROL	Selected axis is controlled by joystick
52	PI_CNTR_COMMUNICATION_ERROR	Controller detected communication error
53	PI_CNTR_DYNAMIC_MOVE_IN_PROCESS	MOV! motion still in progress
54	PI_CNTR_UNKNOWN_PARAMETER	Unknown parameter
55	PI_CNTR_NO_REP_RECORDED	No commands were recorded with REP
56	PI_CNTR_INVALID_PASSWORD	Password invalid
57	PI_CNTR_INVALID_RECORDER_CHAN	Data Record Table does not exist
58	PI_CNTR_INVALID_RECORDER_SRC_OPT	Source does not exist; number too low or too high
59	PI_CNTR_INVALID_RECORDER_SRC_CHAN	Source Record Table number too low or too high
60	PI_CNTR_PARAM_PROTECTION	Protected Param: current Command Level (CCL) too low
61	PI_CNTR_AUTOZERO_RUNNING	Command execution not possible while Autozero is running
62	PI_CNTR_NO_LINEAR_AXIS	Autozero requires at least one linear axis
63	PI_CNTR_INIT_RUNNING	Initialization still in progress
64	PI_CNTR_READ_ONLY_PARAMETER	Parameter is read-only
65	PI_CNTR_PAM_NOT_FOUND	Parameter not found in non-volatile memory
66	PI_CNTR_VOL_OUT_OF_LIMITS	Voltage out of limits
67	PI_CNTR_WAVE_TOO_LARGE	Not enough memory available for requested wave curve
68	PI_CNTR_NOT_ENOUGH_DDL_MEMORY	Not enough memory available for DDL table; DDL can not be started

69	PI_CNTR_DDL_TIME_DELAY_TOO_LARGE	Time delay larger than DDL table; DDL can not be started
70	PI_CNTR_DIFFERENT_ARRAY_LENGTH	The requested arrays have different lengths; query them separately
71	PI_CNTR_GEN_SINGLE_MODE_RESTART	Attempt to restart the generator while it is running in single step mode
72	PI_CNTR_ANALOG_TARGET_ACTIVE	Motion commands and wave generator activation are not allowed when analog target is active
73	PI_CNTR_WAVE_GENERATOR_ACTIVE	Motion commands are not allowed when wave generator is active
74	PI_CNTR_AUTOZERO_DISABLED	No sensor channel or no piezo channel connected to selected axis (sensor and piezo matrix)
75	PI_CNTR_NO_WAVE_SELECTED	Generator started (WGO) without having selected a wave table (WSL).
76	PI_CNTR_IF_BUFFER_OVERRUN	Interface buffer did overrun and command couldn't be received correctly
77	PI_CNTR_NOT_ENOUGH_RECORDED_DATA	Data Record Table does not hold enough recorded data
78	PI_CNTR_TABLE_DEACTIVATED	Data Record Table is not configured for recording
79	PI_CNTR_OPENLOOP_VALUE_SET_WHEN_SERVO_ON	Open-loop commands (SVA, SVR) are not allowed when servo is on
100	PI_LABVIEW_ERROR	PI LabVIEW driver reports error. See source control for details.
200	PI_CNTR_NO_AXIS	No stage connected to axis
201	PI_CNTR_NO_AXIS_PARAM_FILE	File with axis parameters not found
202	PI_CNTR_INVALID_AXIS_PARAM_FILE	Invalid axis parameter file
203	PI_CNTR_NO_AXIS_PARAM_BACKUP	Backup file with axis parameters not found
204	PI_CNTR_RESERVED_204	PI internal error code 204
205	PI_CNTR_SMO_WITH_SERVO_ON	SMO with servo on
206	PI_CNTR_UUDECODE_INCOMPLETE_HEADER	uudecode: incomplete header
207	PI_CNTR_UUDECODE_NOTHING_TO_DECODE	uudecode: nothing to decode
208	PI_CNTR_UUDECODE_ILLEGAL_FORMAT	uudecode: illegal UUE format
209	PI_CNTR_CRC32_ERROR	CRC32 error

210	PI_CNTR_ILLEGAL_FILENAME	Illegal file name (must be 8-0 format)
211	PI_CNTR_FILE_NOT_FOUND	File not found on controller
212	PI_CNTR_FILE_WRITE_ERROR	Error writing file on controller
213	PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE	VEL command not allowed in DTR Command Mode
214	PI_CNTR_POSITION_UNKNOWN	Position calculations failed
215	PI_CNTR_CONN_POSSIBLY_BROKEN	The connection between controller and stage may be broken
216	PI_CNTR_ON_LIMIT_SWITCH	The connected stage has driven into a limit switch, call CLR to resume operation
217	PI_CNTR_UNEXPECTED_STRUT_STOP	Strut test command failed because of an unexpected strut stop
218	PI_CNTR_POSITION_BASED_ON_ESTIMATION	While MOV! is running position can only be estimated!
219	PI_CNTR_POSITION_BASED_ON_INTERPOLATION	Position was calculated during MOV motion
301	PI_CNTR_SEND_BUFFER_OVERFLOW	Send buffer overflow
302	PI_CNTR_VOLTAGE_OUT_OF_LIMITS	Voltage out of limits
303	PI_CNTR_VOLTAGE_SET_WHEN_SERVO_ON	Attempt to set voltage when servo on
304	PI_CNTR_RECEIVING_BUFFER_OVERFLOW	Received command is too long
305	PI_CNTR_EEPROM_ERROR	Error while reading/writing EEPROM
306	PI_CNTR_I2C_ERROR	Error on I2C bus
307	PI_CNTR_RECEIVING_TIMEOUT	Timeout while receiving command
308	PI_CNTR_TIMEOUT	A lengthy operation has not finished in the expected time
309	PI_CNTR_MACRO_OUT_OF_SPACE	Insufficient space to store macro
310	PI_CNTR_EUI_OLDVERSION_CFGDATA	Configuration data has old version number
311	PI_CNTR_EUI_INVALID_CFGDATA	Invalid configuration data
333	PI_CNTR_HARDWARE_ERROR	Internal hardware error
555	PI_CNTR_UNKNOWN_ERROR	BasMac: unknown controller error
601	PI_CNTR_NOT_ENOUGH_MEMORY	not enough memory
602	PI_CNTR_HW_VOLTAGE_ERROR	hardware voltage error

603	PI_CNTR_HW_TEMPERATURE_ERROR	hardware temperature out of range
1000	PI_CNTR_TOO_MANY_NESTED_MACROS	Too many nested macros
1001	PI_CNTR_MACRO_ALREADY_DEFINED	Macro already defined
1002	PI_CNTR_NO_MACRO_RECORDING	Macro recording not activated
1003	PI_CNTR_INVALID_MAC_PARAM	Invalid parameter for MAC
1004	PI_CNTR_RESERVED_1004	PI internal error code 1004
2000	PI_CNTR_ALREADY_HAS_SERIAL_NUMBER	Controller already has a serial number
4000	PI_CNTR_SECTOR_ERASE_FAILED	Sector erase failed
4001	PI_CNTR_FLASH_PROGRAM_FAILED	Flash program failed
4002	PI_CNTR_FLASH_READ_FAILED	Flash read failed
4003	PI_CNTR_HW_MATCHCODE_ERROR	HW match code missing/invalid
4004	PI_CNTR_FW_MATCHCODE_ERROR	FW match code missing/invalid
4005	PI_CNTR_HW_VERSION_ERROR	HW version missing/invalid
4006	PI_CNTR_FW_VERSION_ERROR	FW version missing/invalid
4007	PI_CNTR_FW_UPDATE_ERROR	FW update failed

Interface Errors

0	COM_NO_ERROR	No error occurred during function call
-1	COM_ERROR	Error during com operation (could not be specified)
-2	SEND_ERROR	Error while sending data
-3	REC_ERROR	Error while receiving data
-4	NOT_CONNECTED_ERROR	Not connected (no port with given ID open)
-5	COM_BUFFER_OVERFLOW	Buffer overflow
-6	CONNECTION_FAILED	Error while opening port
-7	COM_TIMEOUT	Timeout error
-8	COM_MULTILINE_RESPONSE	There are more lines waiting in buffer
-9	COM_INVALID_ID	There is no interface or DLL handle with the given ID
-10	COM_NOTIFY_EVENT_ERROR	Event/message for notification could not be opened
-11	COM_NOT_IMPLEMENTED	Function not supported by this interface type
-12	COM_ECHO_ERROR	Error while sending "echoed" data

-13	COM_GPIB_EDVR	IEEE488: System error
-14	COM_GPIB_ECIC	IEEE488: Function requires GPIB board to be CIC
-15	COM_GPIB_ENOL	IEEE488: Write function detected no listeners
-16	COM_GPIB_EADR	IEEE488: Interface board not addressed correctly
-17	COM_GPIB_EARG	IEEE488: Invalid argument to function call
-18	COM_GPIB_ESAC	IEEE488: Function requires GPIB board to be SAC
-19	COM_GPIB_EABO	IEEE488: I/O operation aborted
-20	COM_GPIB_ENEB	IEEE488: Interface board not found
-21	COM_GPIB_EDMA	IEEE488: Error performing DMA
-22	COM_GPIB_EOIP	IEEE488: I/O operation started before previous operation completed
-23	COM_GPIB_ECAP	IEEE488: No capability for intended operation
-24	COM_GPIB_EFSO	IEEE488: File system operation error
-25	COM_GPIB_EBUS	IEEE488: Command error during device call
-26	COM_GPIB_ESTB	IEEE488: Serial poll-status byte lost
-27	COM_GPIB_ESRQ	IEEE488: SRQ remains asserted
-28	COM_GPIB_ETAB	IEEE488: Return buffer full
-29	COM_GPIB_ELCK	IEEE488: Address or board locked
-30	COM_RS_INVALID_DATA_BITS	RS-232: 5 data bits with 2 stop bits is an invalid combination, as is 6, 7, or 8 data bits with 1.5 stop bits
-31	COM_ERROR_RS_SETTINGS	RS-232: Error configuring the COM port
-32	COM_INTERNAL_RESOURCES_ERROR	Error dealing with internal system resources (events, threads, ...)
-33	COM_DLL_FUNC_ERROR	A DLL or one of the required functions could not be loaded
-34	COM_FTDIUSB_INVALID_HANDLE	FTDIUSB: invalid handle
-35	COM_FTDIUSB_DEVICE_NOT_FOUND	FTDIUSB: device not found

-36	COM_FTDIUSB_DEVICE_NOT_OPENED	FTDIUSB: device not opened
-37	COM_FTDIUSB_IO_ERROR	FTDIUSB: IO error
-38	COM_FTDIUSB_INSUFFICIENT_RESOURCES	FTDIUSB: insufficient resources
-39	COM_FTDIUSB_INVALID_PARAMETER	FTDIUSB: invalid parameter
-40	COM_FTDIUSB_INVALID_BAUD_RATE	FTDIUSB: invalid baud rate
-41	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_ERASE	FTDIUSB: device not opened for erase
-42	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_WRITE	FTDIUSB: device not opened for write
-43	COM_FTDIUSB_FAILED_TO_WRITE_DEVICE	FTDIUSB: failed to write device
-44	COM_FTDIUSB_EEPROM_READ_FAILED	FTDIUSB: EEPROM read failed
-45	COM_FTDIUSB_EEPROM_WRITE_FAILED	FTDIUSB: EEPROM write failed
-46	COM_FTDIUSB_EEPROM_ERASE_FAILED	FTDIUSB: EEPROM erase failed
-47	COM_FTDIUSB_EEPROM_NOT_PRESENT	FTDIUSB: EEPROM not present
-48	COM_FTDIUSB_EEPROM_NOT_PROGRAMMED	FTDIUSB: EEPROM not programmed
-49	COM_FTDIUSB_INVALID_ARGS	FTDIUSB: invalid arguments
-50	COM_FTDIUSB_NOT_SUPPORTED	FTDIUSB: not supported
-51	COM_FTDIUSB_OTHER_ERROR	FTDIUSB: other error
-52	COM_PORT_ALREADY_OPEN	Error while opening the COM port: was already open
-53	COM_PORT_CHECKSUM_ERROR	Checksum error in received data from COM port
-54	COM_SOCKET_NOT_READY	Socket not ready, you should call the function again
-55	COM_SOCKET_PORT_IN_USE	Port is used by another socket
-56	COM_SOCKET_NOT_CONNECTED	Socket not connected (or not valid)
-57	COM_SOCKET_TERMINATED	Connection terminated (by peer)
-58	COM_SOCKET_NO_RESPONSE	Can't connect to peer
-59	COM_SOCKET_INTERRUPTED	Operation was interrupted by a nonblocked signal
-60	COM_PCI_INVALID_ID	No device with this ID is present
-61	COM_PCI_ACCESS_DENIED	Driver could not be opened (on Vista: run as administrator!)

DLL Errors

-1001	PI_UNKNOWN_AXIS_IDENTIFIER	Unknown axis identifier
-1002	PI_NR_NAV_OUT_OF_RANGE	Number for NAV out of range--must be in [1,10000]
-1003	PI_INVALID_SGA	Invalid value for SGA--must be one of 1, 10, 100, 1000
-1004	PI_UNEXPECTED_RESPONSE	Controller sent unexpected response
-1005	PI_NO_MANUAL_PAD	No manual control pad installed, calls to SMA and related commands are not allowed
-1006	PI_INVALID_MANUAL_PAD_KNOB	Invalid number for manual control pad knob
-1007	PI_INVALID_MANUAL_PAD_AXIS	Axis not currently controlled by a manual control pad
-1008	PI_CONTROLLER_BUSY	Controller is busy with some lengthy operation (e.g. reference move, fast scan algorithm)
-1009	PI_THREAD_ERROR	Internal error--could not start thread
-1010	PI_IN_MACRO_MODE	Controller is (already) in macro mode--command not valid in macro mode
-1011	PI_NOT_IN_MACRO_MODE	Controller not in macro mode--command not valid unless macro mode active
-1012	PI_MACRO_FILE_ERROR	Could not open file to write or read macro
-1013	PI_NO_MACRO_OR_EMPTY	No macro with given name on controller, or macro is empty
-1014	PI_MACRO_EDITOR_ERROR	Internal error in macro editor
-1015	PI_INVALID_ARGUMENT	One or more arguments given to function is invalid (empty string, index out of range, ...)
-1016	PI_AXIS_ALREADY_EXISTS	Axis identifier is already in use by a connected stage
-1017	PI_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
-1018	PI_COM_ARRAY_ERROR	Could not access array data in COM server
-1019	PI_COM_ARRAY_RANGE_ERROR	Range of array does not fit the number of parameters
-1020	PI_INVALID_SPA_CMD_ID	Invalid parameter ID given to SPA or SPA?

-1021	PI_NR_AVG_OUT_OF_RANGE	Number for AVG out of range--must be >0
-1022	PI_WAV_SAMPLES_OUT_OF_RANGE	Incorrect number of samples given to WAV
-1023	PI_WAV_FAILED	Generation of wave failed
-1024	PI_MOTION_ERROR	Motion error while axis in motion, call CLR to resume operation
-1025	PI_RUNNING_MACRO	Controller is (already) running a macro
-1026	PI_PZT_CONFIG_FAILED	Configuration of PZT stage or amplifier failed
-1027	PI_PZT_CONFIG_INVALID_PARAMS	Current settings are not valid for desired configuration
-1028	PI_UNKNOWN_CHANNEL_IDENTIFIER	Unknown channel identifier
-1029	PI_WAVE_PARAM_FILE_ERROR	Error while reading/writing wave generator parameter file
-1030	PI_UNKNOWN_WAVE_SET	Could not find description of wave form. Maybe WG.INI is missing?
-1031	PI_WAVE_EDITOR_FUNC_NOT_LOADED	The WGWaveEditor DLL function was not found at startup
-1032	PI_USER_CANCELLED	The user cancelled a dialog
-1033	PI_C844_ERROR	Error from C-844 Controller
-1034	PI_DLL_NOT_LOADED	DLL necessary to call function not loaded, or function not found in DLL
-1035	PI_PARAMETER_FILE_PROTECTED	The open parameter file is protected and cannot be edited
-1036	PI_NO_PARAMETER_FILE_OPENED	There is no parameter file open
-1037	PI_STAGE_DOES_NOT_EXIST	Selected stage does not exist
-1038	PI_PARAMETER_FILE_ALREADY_OPENED	There is already a parameter file open. Close it before opening a new file
-1039	PI_PARAMETER_FILE_OPEN_ERROR	Could not open parameter file
-1040	PI_INVALID_CONTROLLER_VERSION	The version of the connected controller is invalid
-1041	PI_PARAM_SET_ERROR	Parameter could not be set with SPA--parameter not defined for this controller!
-1042	PI_NUMBER_OF_POSSIBLE_WAVES_EXCEEDED	The maximum number of wave definitions has been exceeded

-1043	PI_NUMBER_OF_POSSIBLE_GENERATORS_EXCEEDED	The maximum number of wave generators has been exceeded
-1044	PI_NO_WAVE_FOR_AXIS_DEFINED	No wave defined for specified axis
-1045	PI_CANT_STOP_OR_START_WAV	Wave output to axis already stopped/started
-1046	PI_REFERENCE_ERROR	Not all axes could be referenced
-1047	PI_REQUIRED_WAVE_NOT_FOUND	Could not find parameter set required by frequency relation
-1048	PI_INVALID_SPP_CMD_ID	Command ID given to SPP or SPP? is not valid
-1049	PI_STAGE_NAME_ISNT_UNIQUE	A stage name given to CST is not unique
-1050	PI_FILE_TRANSFER_BEGIN_MISSING	A uuencoded file transferred did not start with "begin" followed by the proper filename
-1051	PI_FILE_TRANSFER_ERROR_TEMP_FILE	Could not create/read file on host PC
-1052	PI_FILE_TRANSFER_CRC_ERROR	Checksum error when transferring a file to/from the controller
-1053	PI_COULDNT_FIND_PISTAGES_DAT	The PiStages.dat database could not be found. This file is required to connect a stage with the CST command
-1054	PI_NO_WAVE_RUNNING	No wave being output to specified axis
-1055	PI_INVALID_PASSWORD	Invalid password
-1056	PI_OPM_COM_ERROR	Error during communication with OPM (Optical Power Meter), maybe no OPM connected
-1057	PI_WAVE_EDITOR_WRONG_PARAMNUM	WaveEditor: Error during wave creation, incorrect number of parameters
-1058	PI_WAVE_EDITOR_FREQUENCY_OUT_OF_RANGE	WaveEditor: Frequency out of range
-1059	PI_WAVE_EDITOR_WRONG_IP_VALUE	WaveEditor: Error during wave creation, incorrect index for integer parameter
-1060	PI_WAVE_EDITOR_WRONG_DP_VALUE	WaveEditor: Error during wave creation, incorrect index for floating point parameter

-1061	PI_WAVE_EDITOR_WRONG_ITEM_VALUE	WaveEditor: Error during wave creation, could not calculate value
-1062	PI_WAVE_EDITOR_MISSING_GRAPH_COMPONENT	WaveEditor: Graph display component not installed
-1063	PI_EXT_PROFILE_UNALLOWED_CMD	User Profile Mode: Command is not allowed, check for required preparatory commands
-1064	PI_EXT_PROFILE_EXPECTING_MOTION_ERROR	User Profile Mode: First target position in User Profile is too far from current position
-1065	PI_EXT_PROFILE_ACTIVE	Controller is (already) in User Profile Mode
-1066	PI_EXT_PROFILE_INDEX_OUT_OF_RANGE	User Profile Mode: Block or Data Set index out of allowed range
-1067	PI_PROFILE_GENERATOR_NO_PROFILE	ProfileGenerator: No profile has been created yet
-1068	PI_PROFILE_GENERATOR_OUT_OF_LIMITS	ProfileGenerator: Generated profile exceeds limits of one or both axes
-1069	PI_PROFILE_GENERATOR_UNKNOWN_PARAMETER	ProfileGenerator: Unknown parameter ID in Set/Get Parameter command
-1070	PI_PROFILE_GENERATOR_PAR_OUT_OF_RANGE	ProfileGenerator: Parameter out of allowed range
-1071	PI_EXT_PROFILE_OUT_OF_MEMORY	User Profile Mode: Out of memory
-1072	PI_EXT_PROFILE_WRONG_CLUSTER	User Profile Mode: Cluster is not assigned to this axis
-1073	PI_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
-1074	PI_INVALID_DEVICE_DRIVER_VERSION	The installed device driver doesn't match the required version. Please see the documentation to determine the required device driver version.
-1075	PI_INVALID_LIBRARY_VERSION	The library used doesn't match the required version. Please see the documentation to determine the required library version.
-1076	PI_INTERFACE_LOCKED	The interface is currently locked by another function. Please try again later.

10 Controller Parameters

10.1 Parameter Handling

To adapt the E-753 to your application, you can modify parameter values. The parameters available depend on the controller firmware. With HPA? (p. 136) you can obtain a list of all available parameters with information about each (e.g. short descriptions). The volatile and non-volatile memory parameter values can be read with the SPA? (p. 157) or SEP? (p. 153) commands, respectively. Note that many parameters are "protected" by higher command levels, as indicated in the "Command Level" column in the "Parameter Overview" table (p. 205). By going to command level 1 using the CCL command (p. 119), it is possible to change level-1 parameters. Parameters with level 2 or higher are reserved for service personnel.

Using the "general" modification commands SPA, RPA, SEP and WPA, all parameters for which the currently active command level has write permission can be changed in volatile memory (SPA (p. 154), RPA (p. 149)) or in non-volatile memory (SEP (p. 152), WPA (p. 186)). It is recommended that any modifications be first made with SPA, and when the controller runs well, saved using WPA. Do not change the current interface settings with SPA—except of the baud rate—because it will not be possible to maintain communication afterwards.

In addition to the "general" modification commands, there are commands which change certain specific parameters. All the commands listed below, except of IFS, change the parameter value only in volatile memory, and WPA must be used to save the value to non-volatile memory. IFS changes and saves the interface parameters directly in non-volatile memory only.

AOS (p. 114) (analog input offset)
ATZ (p. 117) (Autozero Matched Offset as the autozero result)
DPO (p. 127) (internal DDL processing parameters)
IFC (p. 138) (baud rate for RS-232 serial connection)
RTR (p. 150) (record table rate)
VEL (p. 170) (Servo Loop Slew-Rate)
WOS (p. 184) (wave generator output offset)
WTR (p. 189) (wave table rate)

IFS (p. 140) (interface parameters in non-volatile memory)

!

CAUTION

Incorrect parameter values may lead to improper operation or damage to your hardware. Be careful when changing parameters.

It is strongly recommended to save the parameter values of the E-753 to a file on the host PC before you make any changes in non-volatile memory. This way the original settings can be restored if the new parameter settings will not prove satisfactory. To save the parameter values and to load them back to the E-753, use the *Device Parameter Configuration* window of NanoCapture™. See the NanoCapture™ manual for more information.

NOTE

The NanoCapture™ host software gives access to parameter values in a more convenient way. Use its *Device Parameter Configuration* window to check/edit the individual parameters. See the NanoCapture™ manual for more information.

Each parameter refers to one of the following items (see the "Item Type Concerned" column in the table below):

- Whole system (controller)
- Hardware modules of the controller
- Logical axes
- Input or output signal channels

The "Max. No. of Items" column shows the maximum number of items for which the parameter is used. Example: "2" for parameter 0x02000200 means that this parameter has different values for each of the 2 input signal channels. For parameters which refer to the whole system the maximum number of items is always 1. See "Accessible Items and Their Identifiers" (p. 43) for the item identifiers to use with SPA, SEP or WPA when changing/saving parameter values or when asking for parameter values with the SPA? or SEP? commands.

Values stored in non-volatile memory are power-on defaults, so that the system can be used in the desired way immediately. Note that PI records the data files of every E-753 controller calibrated at the factory for easy restoration of original settings should that ever be necessary.

When the piezo stage is equipped with an ID-chip (located in the stage connector) and connected to the controller for the first time, the values for stage-related parameters will be written from the ID-chip to the volatile and non-volatile memory of the E-753 upon controller power-on. You can not overwrite the parameters in the ID-chip (this can only be done by PI). See "ID-Chip Support / Stage Replacement" (p. 95) for more information. The parameters stored in the ID-chip are marked in the "Notes" column in the table below.

10.2 Parameter Overview

See "Parameter Handling" (p. 203) for the meaning of the individual columns.

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x02000100	1	Input Signal Channel	2	INT	Sensor Range factor	ID-chip
0x02000101	1	Input Signal Channel	2	INT	Sensor Board Gain	ID-chip
0x02000102	1	Input Signal Channel	2	INT	Sensor Offset factor	ID-chip
0x02000103	1	Input Signal Channel	2	INT	Sensor Cable Compensation	ID-chip
0x02000104	1	Input Signal Channel	2	FLOAT	Autozero Matched Offset	
0x02000200	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 1	ID-chip
0x02000300	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 2	ID-chip
0x02000400	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 3	ID-chip
0x02000500	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 4	ID-chip
0x02000600	1	Input Signal Channel	2	FLOAT	Sensor Mech. Correction 5	ID-chip
0x03000100	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 1	
0x03000101	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 1	
0x03000102	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 1	
0x03000103	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 1	
0x03000104	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 5 1	
0x03000105	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 6 1	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x03000106	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 7 1	
0x03000107	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 8 1	
0x03000108	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 9 1	
0x03000200	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 2	
0x03000201	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 2	
0x03000202	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 2	
0x03000203	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 2	
0x03000204	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 5 2	
0x03000205	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 6 2	
0x03000206	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 7 2	
0x03000207	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 8 2	
0x03000208	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 9 2	
0x03000300	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 3	
0x03000301	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 3	
0x03000302	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 3	
0x03000303	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 3	
0x03000304	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 5 3	
0x03000305	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 6 3	
0x03000306	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 7 3	
0x03000307	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 8 3	
0x03000308	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 9 3	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x03000400	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 4	
0x03000401	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 4	
0x03000402	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 4	
0x03000403	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 4	
0x03000404	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 5 4	
0x03000405	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 6 4	
0x03000406	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 7 4	
0x03000407	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 8 4	
0x03000408	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 9 4	
0x03000500	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 1 5	
0x03000501	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 2 5	
0x03000502	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 3 5	
0x03000503	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 4 5	
0x03000504	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 5 5	
0x03000505	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 6 5	
0x03000506	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 7 5	
0x03000507	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 8 5	
0x03000508	2	Input Signal Channel	2	FLOAT	Sensor Elec. Correction 9 5	
0x03001000	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 1	
0x03001100	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 2	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x03001200	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 3	
0x03001300	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 4	
0x03001400	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 5	
0x03001500	2	Input Signal Channel	2	FLOAT	Sensor Offset Correction 6	
0x04000000	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 1	
0x04000001	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 2	
0x04000002	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 3	
0x04000003	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 4	
0x04000004	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 5	
0x04000005	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 6	
0x04000006	2	Input Signal Channel	2	FLOAT	PGA Correction of gain 7	
0x05000000	1	Input Signal Channel	2	INT	Digital Filter Type	ID-chip
0x05000001	1	Input Signal Channel	2	FLOAT	Digital Filter Bandwidth	ID-chip
0x05000002	1	Input Signal Channel	2	INT	Digital Filter Order	ID-chip
0x05000101	1	Input Signal Channel	2	FLOAT	User Filter Param. 1	
0x05000102	1	Input Signal Channel	2	FLOAT	User Filter Param. 2	
0x05000103	1	Input Signal Channel	2	FLOAT	User Filter Param. 3	
0x05000104	1	Input Signal Channel	2	FLOAT	User Filter Param. 4	
0x05000105	1	Input Signal Channel	2	FLOAT	User Filter Param. 5	
0x06000500	1	Logical Axis	1	INT	ADC Channel for Target	
0x06000501	1	Logical Axis	1	FLOAT	Analog Target Offset	

Parameter ID	Command Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x07000000	1	Logical Axis	1	FLOAT	Range Limit min	ID-chip
0x07000001	1	Logical Axis	1	FLOAT	Range Limit max	ID-chip
0x07000200	1	Logical Axis	1	FLOAT	Servo Loop Slew-Rate	ID-chip
0x07000201	1	Logical Axis	1	FLOAT	Open Loop Slew-Rate	
0x07000300	1	Logical Axis	1	FLOAT	Servo-loop P-Term	ID-chip
0x07000301	1	Logical Axis	1	FLOAT	Servo-loop I-Term	ID-chip
0x07000302	1	Logical Axis	1	FLOAT	Servo-loop D-Term	ID-chip
0x07000500	1	Logical Axis	1	FLOAT	Position from Sensor 1	ID-chip
0x07000501	1	Logical Axis	1	FLOAT	Position from Sensor 2	ID-chip
0x07000600	1	Logical Axis	1	CHAR	Axis Name	ID-chip
0x07000601	2	Logical Axis	1	CHAR	Axis Unit	ID-chip
0x07000800	1	Logical Axis	1	INT	Power Up Servo ON Enable	
0x07000802	1	Logical Axis	1	INT	Power Up AutoZero Enable	
0x07000900	1	Logical Axis	1	FLOAT	ON Target Tolerance	ID-chip
0x07000901	1	Logical Axis	1	FLOAT	Settling Time	
0x07000a00	1	Logical Axis	1	FLOAT	AutoZero Low Voltage	ID-chip
0x07000a01	1	Logical Axis	1	FLOAT	AutoZero High Voltage	ID-chip
0x07000c01	1	Logical Axis	1	FLOAT	Default voltage	ID-chip
0x07001005	1	Logical Axis	1	FLOAT	Position Report Scaling	
0x08000100	1	Logical Axis	1	FLOAT	Notch frequency 1	ID-chip
0x08000101	1	Logical Axis	1	FLOAT	Notch frequency 2	ID-chip
0x08000200	1	Logical Axis	1	FLOAT	Notch Rejection 1	ID-chip
0x08000201	1	Logical Axis	1	FLOAT	Notch Rejection 2	ID-chip
0x08000300	1	Logical Axis	1	FLOAT	Notch Bandwidth 1	ID-chip
0x08000301	1	Logical Axis	1	FLOAT	Notch Bandwidth 2	ID-chip
0x08000400	1	Logical Axis	1	FLOAT	Creep factor T1/sec	ID-chip
0x08000401	1	Logical Axis	1	FLOAT	Creep factor T2/sec	ID-chip
0x09000000	1	Logical Axis	1	FLOAT	Driving Factor of Piezo	ID-chip
0x0a000003	1	Output Signal Channel	1	INT	Select Output type	
0x0a000004	1	Output Signal Channel	1	INT	Select Output index	
0x0b000007	2	Output Signal Channel	1	FLOAT	Min Output Voltage of Amplifier	

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x0b000008	2	Output Signal Channel	1	FLOAT	Max Output Voltage of Amplifier	
0x0b000009	2	Output Signal Channel	1	FLOAT	Voltage of Amplifier with zero to Dac	
0x0c000000	1	Output Signal Channel	1	FLOAT	Soft Voltage Low Limit	ID-chip
0x0c000001	1	Output Signal Channel	1	FLOAT	Soft Voltage High Limit	ID-chip
0x0d000000	2	System	1	CHAR	Device S/N	
0x0d000100	2	Hardware Module	3	CHAR	Hardware S/N	
0x0d000200	2	Hardware Module	3	CHAR	Hardware Name	
0x0e000100	2	System	1	FLOAT	Sensor Sampling Time	
0x0e000200	3	System	1	FLOAT	Servo Update Time	
0x0e000400	1	System	1	INT	DDL license	
0x0e000401	3	System	1	INT	DDL license valid	
0x0e000b00	3	System	1	INT	Number of input channels	
0x0e000b01	3	System	1	INT	Number of output channels	
0x0e000b02	3	System	1	INT	Number of system axes	
0x0e000b03	3	System	1	INT	Number of sensor channels	
0x0e000b04	3	System	1	INT	Number of piezo channels	
0x0e000b05	3	System	1	INT	Number of trigger outputs	
0x0e000c00	2	System	1	FLOAT	Sensor OverSampling Time	
0x0e000c01	2	System	1	INT	Servo OverSampling Order	
0x0f000000	1	Input Signal Channel	2	INT	Power Up Read ID-Chip	
0x0f000100	1	Input Signal Channel	2	CHAR	Stage Type	ID-chip
0x0f000200	1	Input Signal Channel	2	CHAR	Stage Serial Number	ID-chip
0x0f000300	1	Input Signal Channel	2	CHAR	Stage assembly Date	ID-chip

Parameter ID	Com-mand Level	Item Type Concerned	Max. No. of Items	Data Type	Parameter Description	Notes
0x11000400	1	System	1	INT	Uart Baudrate	
0x11000600	1	System	1	CHAR	IP-Address	
0x11000700	1	System	1	CHAR	IP-Mask	
0x11000800	1	System	1	INT	IP-Configuration	
0x11000b00	2	System	1	CHAR	MAC-Address	
0x13000004	3	System	1	INT	Max Wave Points	
0x13000109	1	System	1	INT	Wave Generator Table Rate	
0x1300010a	3	System	1	INT	Number of Waves	
0x1300010b	1	Logical Axis	1	FLOAT	Wave Offset	
0x14000001	1	Logical Axis	1	INT	DDL repeat number	
0x14000006	1	Logical Axis	1	FLOAT	Time Delay Max	
0x14000007	1	Logical Axis	1	FLOAT	Time Delay Min	
0x14000008	1	Logical Axis	1	INT	Time Delay Change Rule	
0x1400000a	1	Logical Axis	1	INT	DDL Zero Gain Number	
0x1400000b	3	System	1	INT	Max DDL Points	
0x14000100	2	Logical Axis	1	FLOAT	Autocal Time Delay Factor	
0x14000101	2	Logical Axis	1	FLOAT	Autocal Min/Max Time Delay Factor	
0x16000000	1	System	1	INT	Data Recorder Table Rate	
0x16000100	3	System	1	INT	Max Number of Data Recorder Channels	
0x16000200	3	System	1	INT	Data Recorder Max Points	
0x16000300	1	System	1	INT	Data Recorder Chan Number	
0xffff0007	2	System	1	CHAR	firmware name	
0xffff0008	2	System	1	CHAR	firmware version	
0xffff000d	2	System	1	CHAR	short description of firmware	
0xffff000e	2	System	1	CHAR	date of firmware	
0xffff000f	2	System	1	CHAR	firmware developer	

11 Troubleshooting

Communication with controller does not work

Communication cable is wrong or defective

⇒ Check cable. Does it work properly with another device? For RS-232, a null-modem cable must be used. For TCP/IP connections: When connecting to a network hub or router (with DHCP-server) use the included straight-through network cable. When connecting directly to the host PC (point-to-point connection), connect the controller to an Ethernet connector in the PC using the included cross-over network cable.

The power-on/reboot sequence had not finished

⇒ Wait at least 20 seconds after power-on or reboot before you try to establish communications or to send commands. Note that even if the Power LED blinks permanently at regular intervals, it might take another period of about 30 seconds before communication between controller and host PC can be established, if you have a network without DHCP server or a point-to-point connection via TCP/IP.

The interface is not configured correctly

⇒ With the RS-232 interface, check port and baud rate. It is recommended that the host PC have a "genuine" RS-232 interface on board. If the host PC uses a USB-to-serial adapter instead, data loss could occur during communication, especially when transfering large amounts of data.

⇒ With the TCP/IP connection, connect the controller to a network access point before you power it on. Check IP address and IP mask (the settings of the devices in the network must be compatible, e.g. the IP address of each device must be unique). Make sure that your network administrator has not set the network to forbid unknown devices like the E-753 to log on. Note that if the controller is already connected to your or another host PC via TCP/IP, a second TCP/IP session cannot be established. Presently, only one port (50000) is available on the E-753 so that only one application at a time can use the TCP/IP connection.

Another program is using the interface

⇒ Close the other program.

Specific software has problems

⇒ See if the system works with some other software, e.g. a terminal or development environment. You can, for example, test the communication by simply starting a terminal program, e.g. PITerminal, and entering commands like *IDN? or HLP?. Note that multi-character commands are transferred as terminated by a LF (line feed) character and are executed only after the LF is received.

Stage does not move

Cable not connected properly

⇒ Check the connecting cable

Stage or stage cable is defective

⇒ Exchange stage with a working stage to test a new combination of controller and stage (only possible with stages which are equipped with ID-chips)

Wrong command or wrong syntax

⇒ Check the error code with the ERR? command (p. 133). "Error Codes" (p. 191) gives the complete error reference.

Wrong axis commanded

⇒ Check if the correct axis identifier is used and if the commanded axis is that of the desired stage (axis identifier akso required with single-axis systems!)

Move commands or wave generator output provoke errors and are ignored

⇒ The axis motion can result from multiple control sources (see "Control Value Generation" (p. 51) for details). The sources have different write priorities:

Motion commands like MOV, MVR, SVA, SVR, IMP and STE are not allowed (will cause an error) when analog control input or wave generator output are active.

It is possible to configure an axis for control by an analog input line while the wave generator output is active for that axis. In this case, the wave generator will continue running, but its output will no longer be used for control value generation. As long as the corresponding axis is set up to be commanded by analog control input, you can stop the wave generator output, but not restart it.

The analog control input is ignored

⇒ When the analog input is used as control source and the axis motion is stopped with STP (p. 160) or #24 (p. 113), the analog input channel is disconnected from the axis. To recommence commanding the axis via the analog input, the corresponding input signal channel must be reconnected to the axis. See "How to work with the Analog Input" (p. 29) for more information.

Offset between axis position range and voltage range is too large so that the commanded control value can not be realized

⇒ For systems with linear piezo actuators, both the range of sensor position values and the range of the output drive voltages are limited. If mechanical drift of the piezo actuator causes too great a shift in the relation between these ranges, then the usable closed-loop travel range will be reduced. Such an offset can be compensated by the AutoZero function. If AutoZero is compatible with your application, use the ATZ command (p. 117) or the AutoZero functionality of NanoCapture™ or PIMikroMove®.

Incorrect configuration

⇒ Check the parameter settings on the controller with the SPA? (p. 157) and SEP? (p. 153) commands.

The high voltage output of the controller is deactivated, while the communication with the controller is still possible

⇒ Check the error state with the ERR? command. If error code 603 is returned (PI_CNTR_HW_TEMPERATURE_ERROR), the internal temperature of the E-753 is out of the allowed range. The high voltage output is deactivated automatically when the internal temperature sensor detects 85 °C. The output will be reactivated automatically when the internal temperature has fallen to 70 °C.
Wait a few minutes to let the E-753 cool down. Note that the wave generator output will continue even if the high voltage output is deactivated, i.e. if a certain number of output cycles was set, they may have already finished when the high voltage output is reactivated.

How to avoid overheating:
Keep the ambient temperature at a noncritical value: Note that the difference between ambient temperature and internal temperature of the E-753 normally is about 20 Centigrade (36 Fahrenheit) degrees.
Place the system in a location with adequate ventilation. Allow at least 10 cm (4 inches) clearance from the top and the rear of the unit and 5 cm (2 inches) from each side. If this is not possible, keep the ambient temperature low.
When using the wave generator, it is recommended to reduce the frequency and/or the amplitude and/or the output duration to avoid overheating. See "Wave Generator" (p. 70) for more information.

Unsatisfactory system performance

The sensor values are not reliable, and the whole system is instable.

⇒ Only thermally stable systems can have the best performance. For a thermally stable system, power on the E-753 at least one hour before you start working with it.

Custom software accessing PI drivers does not run.

Wrong combination of driver routines/VIs

⇒ Check if system runs with Terminal program. If yes read the software manual and compare sample code from the E-753 CD to check the necessary driver routines.

12 Customer Service

Still having problems? Call your PI representative or write to info@pi.ws; please have the following information about your system ready:

- Product codes and serial numbers of all products in the system
- Current firmware version of the controller (if present)
- Version of drivers and / or host software (if present)
- Operating system on host PC (if present)

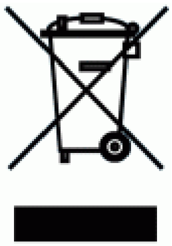
13 Old Equipment Disposal

In accordance with EU directive 2002 / 96 / EC (WEEE), as of 13 August 2005, electrical and electronic equipment may not be disposed of in the member states of the EU mixed with other wastes.

To meet the manufacturer's product responsibility with regard to this product, Physik Instrumente (PI) GmbH & Co. KG will ensure environmentally correct disposal of old PI equipment that was first put into circulation after 13 August 2005, free of charge.

If you have such old equipment from PI, you can send it to the following address postage-free:

Physik Instrumente (PI) GmbH & Co. KG
Auf der Römerstr. 1
76228 Karlsruhe, Germany



14 Technical Data

14.1 Specifications

	E-753.1CD	Tolerance
Function	Digital Controller for single-axis piezo nanopositioning systems with capacitive sensors	
Axes	1	Max.
Processor	DSP 32-bit floating point, 60 MHz	
Sampling rate, servo-control	25 kHz	
Sampling rate, sensor	100 kHz, with oversampling factor 4	
Sensor		
Servo characteristics	PI	
Sensor type	Capacitive	
Sensor channels	1	Max.
Sensor bandwidth	5.6 kHz	
Sensor resolution	17 bit	
Ext. synchronization	Yes	
Amplifier		
Max. output voltage	-30 V	±3
Min. output voltage	135 V	±3
Amplifier channels	1	Max.
Peak output power per channel	15 W	Max.
Average output power per channel	5 W	Max.

	E-753.1CD	Tolerance
Current limitation	Short-circuit proof	
Resolution DAC	24 bit	
Temperature sensor	Max. 85 °C, deactivation of the piezo voltage output	
Interfaces and operation		
Communication interfaces	RS-232, Ethernet	
Piezo connector	Sub-D special connector	
Sensor connector	Sub-D special connector	
Analog input	1 x ±10 V differential, ±70 V common mode voltage; 5 kHz bandwidth; 16 bit; for control input / sensor; Lemo	
Digital input	2 x Lemo, TTL	
Digital output	2 x Lemo; TTL	
Command set	PI General Command Set (GCS) 2.0	
User software	NanoCapture™, PIMikroMove®	
Software drivers	LabVIEW™ driver, DLLs	
Supported functionality	Wave-Gen, Trigger IO	
Display	LED for ONT, ERR, POWER	
Linearization	4th order polynomial, DDL	
Separate protective ground connector	Yes	

	E-753.1CD	Tolerance
Miscellaneous		
Operating temperature range	5 to 50 °C	
Mass	0.9 kg	
Dimensions	264 x 125 x 48 mm (with rubber feet)	
Power consumption	10 W	Max.
Operating voltage	24 VDC from external power supply (included)	

14.2 Pin Assignments

14.2.1 PZT & Sensor Socket

Connector type: Sub-D Mix connector for 2 coax lines and 5 single pins

Pin	Signal	Function
Coax inner lines:		
A1	output	PZTOUT
A2	input	Sensor Probe
Standard pins:		
1	bidirectional	DOW*
2	GND	AGND
3	GND	AGND
4	GND	CGND
5	output	Sensor Target

*DOW bus for ID memory

Note:
Probe and Target are the connections of the capacitive sensor in the mechanics.
The PZTOUT line carries the piezo voltage for the actuator in the mechanics, up to 135 V.

14.2.2 Analog In Socket

Connector type: LEMO EPG.00.302.NLN

Pin	Signal	Function
1	input	A_{IN+} Analog differential input
2	input	A_{IN-} Analog differential input

The socket case is connected to CGND.

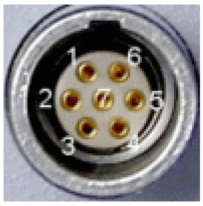
$|A_{IN+} - A_{IN-}| \leq 10$ V

Allowable common mode voltage: ±70 V
Bandwidth: 5 kHz

14.2.3 I/O Socket

Connector type: LEMO EPG.0B.307.HLN

Pin	Signal	Function
1	GND	Ground
2	nc	-
3	output	OUT2 (TTL, this digital output line can be configured with CTO (p. 122) and TWS (p. 169) for triggering tasks, identifier is 1)
4	output	OUT1 (TTL, digital output of the servo cycle; not accessible by command)
5	input	IN2 (TTL, digital input line 2 for wave generator start/stop, see WGO)
6	input	IN1 (TTL, digital input line 1 for wave generator start, see WGO (p. 179))
7	output	VDD (+5V)



14.2.4 24 VDC Socket

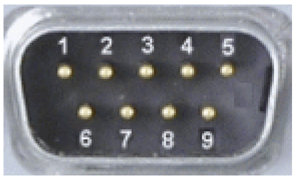
Connector type: TRA3M (m), Switchcraft

Pin	Signal	Function
1	input	0 V
2	input	+24 VDC
3	GND	CGND

14.2.5 RS-232 Socket

Connector type: Sub-D 9 pin (m)

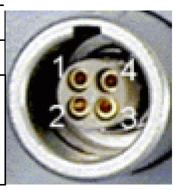
Pin	Function
1	nc
2	RXD receive data
3	TXD send data
4	nc
5	DGND ground
6	nc
7	RTS Hardware handshake, output
8	CTS Hardware handshake, input
9	nc



14.2.6 Sync In Socket

Connector type: LEMO EPG.00.304.NLN

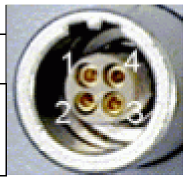
Pin	Function
1	SYNCBASE_I_P (LVDS, 4.8 MHz)
2	SYNCBASE_I_N (LVDS, 4.8 MHz)
3	SYNC100_I_P (LVDS, 100 kHz)
4	SYNC100_I_N (LVDS, 100 kHz)



14.2.7 Sync Out Socket

Connector type: LEMO EPA.00.304.NLN

Pin	Function
1	SYNCBASE_O_P (LVDS, 4.8 MHz)
2	SYNCBASE_O_N (LVDS, 4.8 MHz)
3	SYNC100_O_P (LVDS, 100 kHz)
4	SYNC100_O_N (LVDS, 100 kHz)



14.3 Operating Limits

In order to achieve minimum distortion of the output waveform, it is important to ensure that the amplitude of higher-frequency control input is reduced in proportion to the fall-off of the output voltage at these frequencies. For exact information on maximum operating frequency with a given piezo load (capacitance), refer to the individual operating limit graphs in the figure below.

Note that the operating limits of a given piezo amplifier depends on the amplifier power, the amplifier design, and, of course the capacitance of the piezo actuator. The capacitance of piezo ceramics changes significantly with amplitude, temperature, and load-up to approximately 200% of the unloaded, small-signal capacitance at room temperature.
The following equations describe the relationship between (reactive) drive power, actuator capacitance, operating frequency and drive voltage.
The average power that a piezo amplifier has to be able to provide for sinusoidal operation is given by:
$Pa \approx C \cdot Umax \cdot Up\text{-}p \cdot f$
Peak power for sinusoidal operation is:
$Pmax \approx \pi \cdot C \cdot Umax \cdot Up\text{-}p \cdot f$
Where:
Pa = average power [W]
Pmax = peak power [W]
C = PZT actuator capacitance [farad (As/v)]
f = operating frequency [Hz]
Umax = nominal voltage of the amplifier [V]
Up-p = peak-peak drive voltage [V]

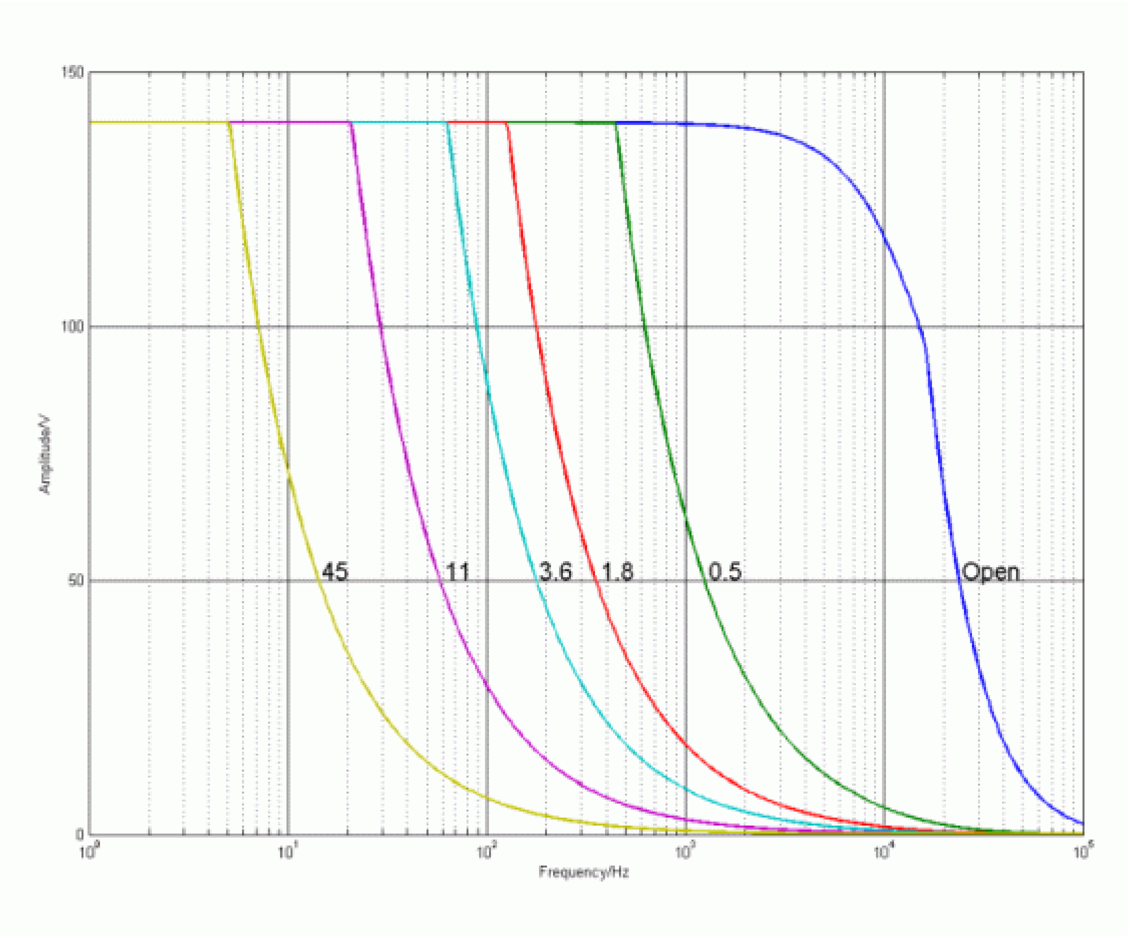


Figure 23: E-753 operating limits with various piezo loads. Capacitance values in µF.

15 Index

2

A

B

C

D

E

F

G

H

I

M