

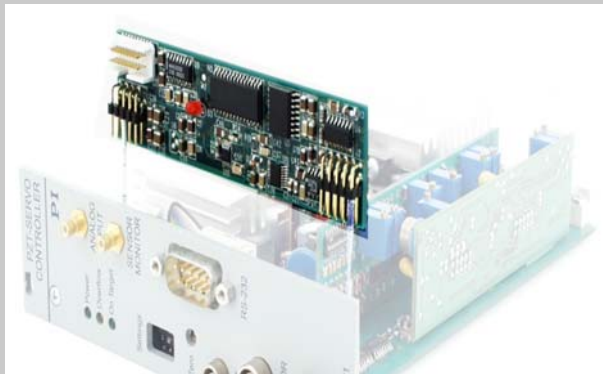
PZ116E User Manual

E-816 Computer Interface and Command Interpreter

Submodule for PZT Controller

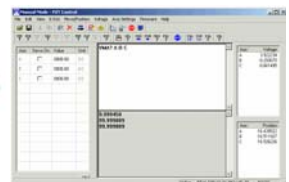
Release: 1.2.0

Date: 2007-06-22



This document describes the following product(s):

- E-816.00
Computer Interface and Command Interpreter Submodule (firmware version 2.11) for PZT Controllers



Physik Instrumente (PI) GmbH & Co. KG is the owner of the following company names and trademarks:

PI®

Hyperbit™

Hyperbit™ is protected by the following patents:

US Patent 6,950,050

The following designations are protected company names or registered trademarks of third parties:

Microsoft, Windows, LabView

Copyright by 1999–2007 Physik Instrumente (PI) GmbH & Co. KG, Karlsruhe, Germany

The text, photographs and drawings in this manual enjoy copyright protection. With regard thereto, Physik Instrumente (PI) GmbH & Co. KG reserves all rights. Use of said text, photographs and drawings is permitted only in part and only upon citation of the source.

First printing 2007-06-22

Document Number PZ116E, ECo, Release 1.2.0

E-816_User_PZ116E120.doc

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 This manual is superseded by any new release.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

www.pi.ws.

Related Documents

The software tools which are delivered with the E-816 Computer Interface and Command Interpreter Submodule for PZT Controller and the piezo controller in which the E-816 is integrated are described in their own manuals. 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.


E-816 DLL, PZ120E

E-816 LabVIEW Software Manual, PZ121E

PZTControl Software Manual, PZ146E

Declaration of Conformity

according to ISO / IEC Guide 22 and EN 45014

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

The manufacturer hereby declares that the product

Product Name: **Computer Interface and Command Interpreter
Submodule**

Model Numbers: **E-816**

Product Options: **all**

complies—if installed in a compatible system from PI—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

August 24, 2004
Karlsruhe, Germany



Dr. Karl Spanner
President

Table of Contents

1.	Introduction.....	5
1.1.	Key Features.....	5
1.2.	Analog Operation	5
1.3.	InterLinking Multiple E-816s.....	5
1.3.1.	System Overview	6
1.3.2.	Bus Terminology	6
2.	Starting Operation	7
3.	Software	8
3.1.	Overview	8
3.2.	Downloading Software Updates.....	8
3.2.1.	E-816 Software	8
3.2.2.	LabVIEW Analog Driver Software.....	9
4.	GCS Command Syntax.....	10
4.1.	General Format.....	10
4.2.	Commands with Floating Point Arguments	11
4.3.	Command Sequence Examples	11
5.	Command Survey	13
6.	Command Reference (Alphabetical)	14
7.	Pin Assignments.....	27
8.	Appendix	28
8.1.	Factory Defaults.....	28
8.2.	Voltages and Positions	28
8.3.	Calibration Register Structure	29
8.4.	Drift Compensation	31
8.5.	Master/Slave Command Processing.....	32
8.5.1.	Setting Channel Names	32
8.5.2.	Setting Correct Baud Rate of Host Software	33
8.5.3.	Checking Connection and Master Unit	33
8.5.4.	Enabling Computer-Controlled Operation.....	33
8.6.	Interfaces	33
8.6.1.	Target (Analog) Output	33
8.6.2.	Sensor (Analog) Input.....	33
8.6.3.	PZT Voltage (Analog) Input	34

8.6.4.	Wave Table Trigger (Digital) Input	34
8.6.5.	Other Digital I/O	34
8.7.	Wave Table Functionality.....	34
8.8.	Read/Write Speed.....	34

1. Introduction

The E-816 is a submodule that plugs onto a PZT controller main board. It is user installable, but typically comes preinstalled as part of a pre-calibrated, ready-to-use system. The E-816 provides an RS-232 ASCII interface over which the device on which the submodule is installed can be controlled by a host computer.

1.1. Key Features

- 20-bit A/D and D/A resolution
- RS-232 interface with up to 115,200 baud data rate
- I²C bus for interlinking up to 12 units in a network
- 60 to 300 read or write commands per second
- 0.02% control precision
- Individual calibration registers for absolute voltage and position commands
- Analog input channels for monitoring actual PZT voltage and displacement
- Compatible with PI General Command Set
- DLL Library, COM Server and LabVIEW drivers provided
- Graphical user interface *PZTControl* software provided

1.2. Analog Operation

Controlling the PZT controller with a computer-generated analog signal (e.g. from a DAQ board) does not involve the E-816 and is not covered in this manual. However, the E-816 CD does contain the documentation on the PI LabVIEW driver set supporting that operating mode, including the Hyperbit™ drivers which make possible position resolution higher than that of the DAQ board used. Installation of the LabVIEW Analog Driver set is also a Setup option on the E-816 CD. New releases of the LabVIEW Analog Driver set are available from the download area at www.pi.ws. See the included E500T0011 Technical Note for instructions. For the Hyperbit™ extension, contact your PI Sales Engineer.

1.3. InterLinking Multiple E-816s

If you want to use multiple E-816s in one system, interlinking them on the embedded I²C bus is the best solution. The benefits are:

- Only one host-computer serial port is necessary for accessing all the E-816s.
- Support for up to 12 E-816.00s on one I2C bus. Logical support for up to 24 E-816.00s, when proper back-panel connections are provided.

1.3.1. System Overview

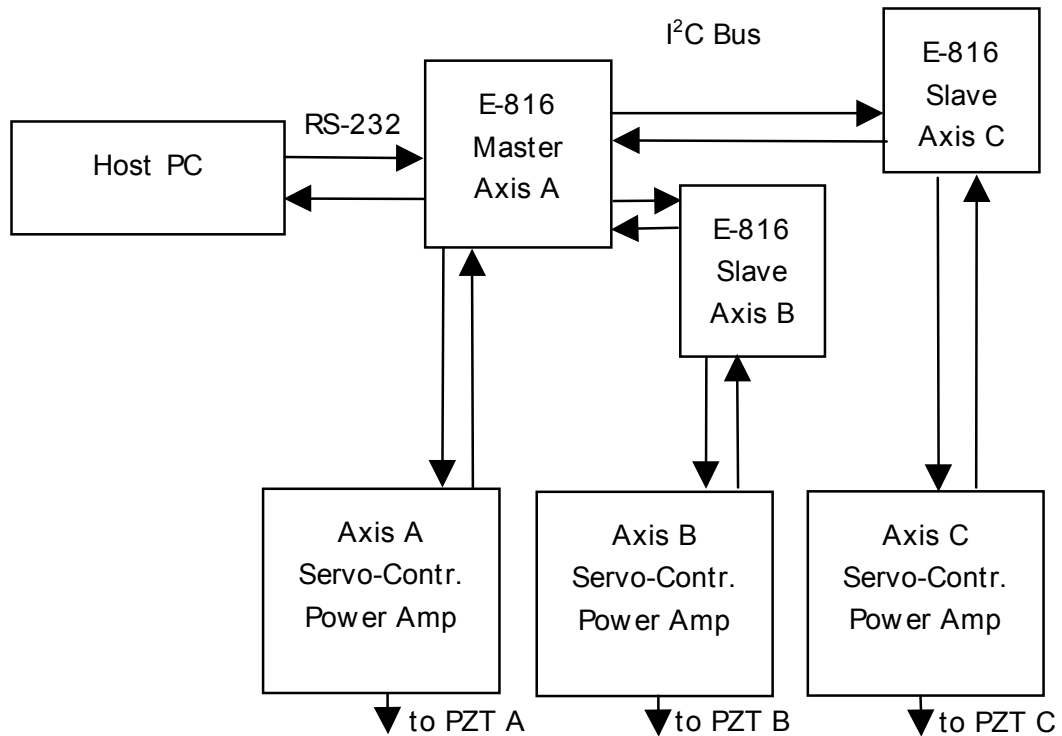


Fig. 1: System overview

1.3.2. Bus Terminology

Master Unit: the unit which is connected with computer via the serial communications cable. The master unit is always addressable with the axis designator “A”.


Slave Unit: a unit which is connected with the master unit via the I²C bus but with no direct connection to the computer. A few commands cannot be addressed to a slave unit, so it may occasionally be necessary to move the RS-232 cable from one unit to another to execute them.

Channel or Axis Designator or Name: When multiple units are connected via the I²C bus, arriving commands are forwarded to the proper unit according to its channel designator.

2. Starting Operation

1. Set up the system as described in the "Starting Operation" or "Quick Start" section of the User Manual for the unit in which the E-816 is installed (e.g. E-621).
2. Be sure the unit in which the E-816 is installed is configured to allow computer-controlled operation (see Section 8.5.4, p. 33 for details). Note that controlling the unit with a computer-generated analog signal is considered Analog Operation and is not fully covered in this manual (see "Introduction" above).

3. Install the host software on the host PC

- a) Insert the E-816 CD in your host PC
- b) If the Setup Wizard does not open automatically, start it from the root directory of the CD with the  icon.
- c) Follow the on-screen instructions. You can choose between *Complete* and *Custom* installation. With *Custom*, you can, for example, install the LabVIEW drivers for digital or analog operation, or both.

Select the features you want to install, and d

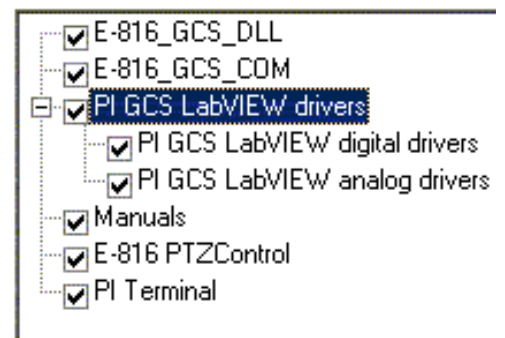


Fig. 2: E-816 Custom Setup

4. Start a terminal program or *PZTControl* and establish a connection to the first unit over RS-232 at 115,200 baud
5. If you are planning to run networked E-816s, proceed as follows
 - a) Check with the SAI? command to see if multiple axis names are configured. E-816s delivered together installed in the same chassis will be preconfigured with unique channel names.
 - b) If not, decide on the desired single-character names
 - c) Addressing the unit directly connected to the host PC over RS-232 as "A", set its axis name to the value desired using the SCH command
 - d) Store the value in non-volatile memory with WPA 100 (see "Setting Channel Names", p. 32 for details). It will become effective after the next reset of the device (RST command or power cycle)
 - e) Connect a different unit directly to the host PC via RS-232
 - f) Repeat the previous two steps for each unit to be networked
 - g) If you have not already reset the devices, power cycle them now
 - h) Connect the unit which is to serve as master to the host PC, and check the available axes with SAI?
6. Send a few test moves to different axes to check the system

3. Software

3.1. Overview

Software Tool	Purpose	Recommended for
<i>PZTControl</i>	This GUI for Microsoft Windows platforms is the operating software for the E-816. <i>PZTControl</i> 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 command formats. It also has a complete command input facility, which represents an easy way to experiment with various commands. <i>PZTControl</i> uses the GCS DLL described below to command the controller.	Users who want to test the equipment before or instead of programming an application and who want to optimize the servo-loop parameters
LabVIEW Digital Drivers	The E-816 LabVIEW digital drivers support communication via RS-232 with one or more controllers containing E-816 interface submodules. This driver set supports the <i>PI General Command Set (GCS)</i> .	Users who want to use LabVIEW (a software tool for data acquisition and process control, available separately from National Instruments) for programming their application
LabVIEW Analog Drivers	The LabVIEW analog drivers support using a DAQ board to generate an analog signal to operate a controller with an analog input. This driver set supports the <i>PI General Command Set (GCS)</i> .	Users who want to use LabVIEW (a software tool for data acquisition and process control, available separately from National Instruments) with a DAQ board to control their application using an analog signal
GCS DLL (Windows Dynamic Link Library)	Allows program access to the E-816 from languages like C++. The functions in the DLL are based on the <i>PI General Command Set (GCS)</i> .	Recommended for users who want to use a DLL library for their applications. Needed by <i>PZTControl</i>
GCS COM (Component Object Model) Server	Virtually the same as the DLL, but supporting the Microsoft COM Object interface for languages like Visual Basic	For users who need a COM Server instead of a DLL
PI Terminal	A terminal program for sending and receiving commands over ASCII interfaces supported by PI controllers, like the RS-232 interface of the E-816	For users who just want to experiment with the various commands

3.2. Downloading 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.

3.2.1. E-816 Software

The password for E-816 software is provided on the E-816 CD in the E-816 Update 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-816 Update_xxxxxx.pdf on the E-816 CD and click on *Login*
- 4 Click on *Download* in the navigation bar across the top
- 5 Click on the *Piezo Drivers & Nanopositioning Controllers* category
- 6 Click on *E-816*
- 7 Click on *Release* (if you click on *Documents* you will get the latest manuals, which may be newer than those in the release)
- 8 Click the *download* button below the latest CD-Mirror (includes the manuals that were with the release)

3.2.2. LabVIEW Analog Driver Software

The password for the LabVIEW Analog Driver set is in the E500T0011 Technical Note, which also contains detailed downloading instructions.

4. GCS Command Syntax

Commands are used to set operating modes, transfer motion parameters and query system and motion values. Because of the variety of functions and parameters, a sequence of commands is usually needed to achieve a specific system action.

4.1. General Format

As seen in the examples, commands are transmitted as ASCII characters in the following format:

CMD \square SP \square XsV.V \square LF

where

CMD command mnemonic (token)

\square SP space (char #32).

X axis designation (A , B , C, ...),

s sign (positive values can be transmitted without sign)

V.V parameter, values are floats or integers, depending on the command.

\square LF LineFeed (Char #10).

The space and LineFeed are not marked explicitly in all the examples in this manual. Current firmware versions also accept CR (carriage return, ASCII 13) as command termination character.

Example:

Send: MOV A10.0
 Move axis A to position 10.0.

Response Format:

Some commands deliver a report (response); all reports are made up of ASCII characters and have one of the following formats:

sV.V \square LF

sV \square LF

where:

s sign (positive values are transmitted without sign)

V.V or **V** result, representation of double-precision or integer values, depending the command

\square LF LineFeed (Char #10).

Example:

Send: POS? A
Report: 1.0006

4.2. Commands with Floating Point Arguments

Some commands accept parameters in floating point format. For these parameters the following syntax is possible:

```
svLF  
sv.vLF  
sv.vEsxxLF
```

where:

s sign(positive values can be without sign)
v ASCII digit or digits(will be converted into floating point internally by firmware)
v.v the decimal separator must be a ".", not ",",
E exponent separator
sxx 2-digit exponent with optional sign
`LF` LineFeed (Char #10).

In reports, floating point values are always represented as follows:

```
sv.vvvvLF
```

where:

s sign (positive values transmitted without sign)
v.vvvv always transmitted with 4 digits after decimal point (internal calculations use double precision)
`LF` LineFeed (Char #10).

4.3. Command Sequence Examples

In the first example, a PZT is moved in closed-loop operation to three different positions, one after the other. After each move, the current position is queried.

Hardware:

E-621 connected to a LVPZT with sensor, range 50µm

Example #1

COMMAND	Reading	Function
SVO A1		Set channel to servo-on mode. This enables the POS commands and disables all VOL commands for the channel. (Make sure that jumpers/switches on the E-621 are set so as to enable servo-control switching).
MOV A30.5		Command Axis A to 30.5 μm
POS? A	30.4902	Query of the current position of Axis A
MOV A20		Command Axis A to 20 μm
POS? A	19.8516	Query of the current position of Axis A
MOV A35		Command Axis A to 35 μm
POS? A	35.0243	Query of the current position of Axis A

Example #2

This example runs the PZT in open-loop mode.

Hardware:

E-621 connected to a LVPZT with sensor, range 50 μm

COMMAND	Reading	Function
ERR?		Clear (master unit) error flag
SVA C80		Set the Axis C PZT voltage to +80.0 volts
VOL? A	79.9947	Query the current axis C output voltage. This command reads the A/D converter and delivers the current output. Note the difference to the SVA? command, which reports the last <i>programmed (commanded)</i> value.
SVA A150		Command output of 150.00 volts to the PZT connected to channel A (this is over the limit)
ERR?	0	No error is set (master unit)
OVF?	0	Overflow is off
SVA? A	150	Query axis A last commanded voltage
VOL? A	110.34	Query the voltage actually on the PZT

5. Command Survey

Command	Parameter	Description
<i>The commands in this section are executed by the master unit only (unit with RS-232 cable to host computer) and cannot be addressed to a slave.</i>		
ERR?		Get master unit error code
I2C?		Get status of I ² C bus
*IDN?		Get master unit version information
BDR	<i>N</i>	Set master unit baudrate
BDR?		Get master unit baudrate
AVG	<i>N</i>	Set number of samples to use for averages on master unit
AVG?		Get number of samples being used for averages on master unit
SCH	<i>c</i>	Set second axis name of master unit
SCH?		Get second axis name of master unit (if none, responds A)
SAI?		Get names assigned to all connected (networked) axes
SPA	<i>An x.x</i>	Set specified parameter of the master unit axis*
SPA?	<i>An</i>	Get specified parameter of the master unit axis*
WPA	<i>password</i>	Write all master unit parameters to master unit flash ROM **
RST		Reset the master unit

<i>The following commands are forwarded by the master unit to the unit assigned to control the specified axis for execution.</i>		
DCO	<i>An</i>	Set D/A converter drift compensation on or off
DCO?	<i>A</i>	Get D/A converter drift compensation setting
MOV	<i>Ax.x</i>	Move the given axis to absolute position
MVR	<i>Ax.x</i>	Move the given axis relative to current position
MOV?	<i>A</i>	Read the last commanded position of the given axis
SVA	<i>Ax.x</i>	Set the given axis to absolute PZT voltage
SVR	<i>Ax.x</i>	Change the given axis PZT voltage relative to current value
SVA?	<i>A</i>	Read the last commanded PZT voltage of the given axis
POS?	<i>A</i>	Read the actual position of the given axis
VOL?	<i>A</i>	Read the actual PZT voltage of the given axis
OVF?	<i>A</i>	Get overflow status of the given axis
ONT?	<i>A</i>	Get on-target status of the given axis
SVO	<i>An</i>	Set servo-ON/OFF status of the given axis
SVO?	<i>A</i>	Get servo-ON/OFF status of the given axis
SWT	<i>An x.x</i>	Set wave table data
SWT?	<i>An</i>	Get wave table data
WTO	<i>An x.x</i>	set wave table output
SSN?	<i>A</i>	Get serial number

* Parameters 1-6 are for PI use only, no write access for customer

** Password protected, must use "WPA 100" to write the parameters to flash ROM.

6. Command Reference (Alphabetical)

*IDN? (Get Identity Number)	*IDN?
------------------------------------	--------------

Command Type: Report Command
Description: Reports master unit device identity number
Format: *IDN?
Arguments: none
Response: One-line string, terminated by LF.

AVG (Set Average Times)	AVG
--------------------------------	------------

Command Type: Configuration Command
Description: Sets the number of samples to be used when calculating averages on the master unit. Larger values mean more stable output, but slower measurement speed. The default is 32.
Format: **AVG** *n*
Arguments: *n*, the number of samples. It must be one of following values: 1, 2, 4, 8, 16, 32 or 64; sending other values can have unpredictable results and may not set the error flag.
Response: None

AVG? (Read Average Times)	AVG?
----------------------------------	-------------

Command Type: Report Command
Description: Reports the current average setting.
Format: **AVG?**
Response: The number of samples used to calculate averages on the master unit. Will be 1, 2, 4, 8, 16, 32 or 64.

BDR x.x (Set Baud Rate)	BDR
Command Type:	System Configuration Command
Description:	Set master unit RS-232 communications baud rate.
Format:	BDR x.x
Arguments:	x.x, the desired baud rate in thousands (must be 9.6, 19.2, 38.4, 57.6, 115.2).
Response:	None
Note:	<p>(1) This command only changes the setting in RAM; the new setting will be lost when the unit is powered down or reset (RST command) unless the RAM settings are written to ROM with WPA xxx</p> <p>(2) The new baud rate does not take effect before the next power on or RST. (So to take effect at all, it must be saved to ROM!)</p> <p>(3) Incorrect entries (such as 56) have unpredictable results and may not set an error status. Check RAM setting with BDR? before attempting to use it.</p>
<i>Example:</i>	<p>The current power-on baud rate is 9.6 kbps, the desired baud rate is 115.2 kbps. Start the terminal program at 9.6 kbps, type in the following command:</p> <pre>BDR 115.2</pre> <pre>BDR?</pre> <p>Response: 115.2</p> <pre>WPA 100</pre> <pre>RST</pre> <p>Now the device runs at a baud rate of 115.2 kbps. You must now change baud rate setting of host-software from 9.6 to 115.2 kbps.</p>
IMPORTANT:	<p>If you forget the current ROM baud rate and cannot communicate with the device, you can determine the baud rate in one of three ways:</p> <ul style="list-style-type: none"> • Use the baud rate search function of the terminal software to let it figure out what baud rate the unit is set to. • Set host software, such as <i>PITerminal</i>, to any baud rate, reset E-816 with power-off/power-on, type the *IDN? command and check the reply from E-816. If there is no reply, try another baud rate. • Count the number of short blinks of the LED on the E-816 plug-in card after power on (with an E-816 on an E-621, light from this LED can be seen viewing upwards through the front-panel dip-switch orifice marked "settings"). 1, 2, 3, 4 or 5 short blinks indicate a baud rate of 115.2, 57.6, 38.4, 19.2 or 9.6 respectively.

BDR? (Get Baud Rate)**BDR?**

Command Type: Report Command

Description: Get the master unit RAM baud rate setting for RS-232 communication.

Format: BDR?

Arguments: None.

Response: Can be 9.6, 19.2, 38.4, 57.6, 115.2

Note: This is the setting that will be written to flash ROM if a WPA command is executed. The baud rate in use is the one that was in ROM at the time of the last power-on.

DCO (Set Drift Compensation Mode)**DCO**

Command Type: Configuration Command

Description: Sets the drift compensation for the digital-analog converter ON or OFF. Drift compensation avoids unwanted changes in displacement over time, but is recommended for static operation only. For a detailed description see p. 31.

Format: DCO Xn

Parameters: X: Axis identifier.
n: 1 or 0. 1 for ON, 0 for OFF.

Response: none

Note that drift compensation is not performed during wave table output, even if DCO is set to 1.

DCO? (Get Drift Compensation Mode)**DCO?**

Command Type: Report Command

Description: Reports the Drift Compensation Mode setting

Format: DCO? X

Parameters: X: Axis identifier.

Response: "1" for ON, "0" for OFF

ERR? (Get Error Message)**ERR?**

Command Type:	Report Command																
Description:	Reports master unit error code. This command also clears the error code. It is recommended that the system status be queried whenever a command fails.																
Format:	ERR?																
Arguments:	none																
Response:	Error Codes																
	<table> <tr> <th><u>Code</u></th><th><u>Meaning</u></th></tr> <tr> <td>0</td><td>No error</td></tr> <tr> <td>1</td><td>Parameter syntax error</td></tr> <tr> <td>5</td><td>Cannot set position before INI or when servo is off</td></tr> <tr> <td>303</td><td>Cannot set voltage when servo on</td></tr> <tr> <td>304</td><td>Received command is too long</td></tr> <tr> <td>305</td><td>Error in reading/writing EEPROM.</td></tr> <tr> <td>306</td><td>Error in I2C bus.</td></tr> </table>	<u>Code</u>	<u>Meaning</u>	0	No error	1	Parameter syntax error	5	Cannot set position before INI or when servo is off	303	Cannot set voltage when servo on	304	Received command is too long	305	Error in reading/writing EEPROM.	306	Error in I2C bus.
<u>Code</u>	<u>Meaning</u>																
0	No error																
1	Parameter syntax error																
5	Cannot set position before INI or when servo is off																
303	Cannot set voltage when servo on																
304	Received command is too long																
305	Error in reading/writing EEPROM.																
306	Error in I2C bus.																

I2C? (Get I2C Status)**I2C?**

Command Type:	Report Command																
Description:	Reports the status of the I ² C bus that connects networked controllers. Status codes are cleared after they are reported.																
Format:	I2C?																
Arguments:	None																
Response:	<p>Status codes of I²C-Bus are reported as bit-mapped hex values. I²C bus status code bits:</p> <table> <tr> <td>bit 0 (LSB):</td><td>CHK_SEN0 timeout</td></tr> <tr> <td>bit 1:</td><td>CHK_PEN0 timeout</td></tr> <tr> <td>bit 2:</td><td>CHK_RSEN0 timeout</td></tr> <tr> <td>bit 3:</td><td>CHK_RW0 timeout</td></tr> <tr> <td>bit 4:</td><td>CHK_BF0 timeout</td></tr> <tr> <td>bit 5:</td><td>CHK_BF1 timeout</td></tr> <tr> <td>bit 6:</td><td>CHK_ACK0 timeout</td></tr> <tr> <td>bit 7: (MSB)</td><td>SLAVE_BUSY timeout</td></tr> </table>	bit 0 (LSB):	CHK_SEN0 timeout	bit 1:	CHK_PEN0 timeout	bit 2:	CHK_RSEN0 timeout	bit 3:	CHK_RW0 timeout	bit 4:	CHK_BF0 timeout	bit 5:	CHK_BF1 timeout	bit 6:	CHK_ACK0 timeout	bit 7: (MSB)	SLAVE_BUSY timeout
bit 0 (LSB):	CHK_SEN0 timeout																
bit 1:	CHK_PEN0 timeout																
bit 2:	CHK_RSEN0 timeout																
bit 3:	CHK_RW0 timeout																
bit 4:	CHK_BF0 timeout																
bit 5:	CHK_BF1 timeout																
bit 6:	CHK_ACK0 timeout																
bit 7: (MSB)	SLAVE_BUSY timeout																

MOV (Move Axis Absolute)		MOV
Command Type:	Move Command	
Description:	<p>Move the specified axis to the commanded absolute position.</p> <p>Before using this command, please make sure the E-816 has servo-control mode set ON. (The E-816 simply provides an on/off signal for the associated servo-controller, which must be activated and properly configured. See the corresponding manuals for details)</p>	
Format:	MOV Ax.x	
Arguments:	<p>A : axis identifier;</p> <p>x.x: the commanded value, in μm.</p> <p>Both parameters are required.</p>	

MOV? (Read Commanded Position)		MOV?
Command Type:	Report Command	
Description:	Reports the commanded position of the specified axis	
Format:	MOV? A	
Arguments:	A : axis identifier. The parameter is required.	
Response:	<p>When used after a MOV or MVR command, reports the commanded position of the specified axis. The value returned is floating point in μm.</p>	

MVR (Move Axis Relative)		MVR
Command Type:	Move Command	
	<p>Description: increment/decrement the position of the specified axis by the commanded value.</p> <p>Before using this command, please make sure the E-816 has servo-control mode set ON. (The E-816 simply provides an on/off signal for the associated servo-controller, which must be activated and properly configured. See the corresponding manuals for details)</p>	
Format:	MVR Ax.x	
Arguments:	<p>A : axis identifier,</p> <p>x.x: the commanded value, in μm.</p> <p>Both parameters are required.</p>	
Response:	none	

ONT? (Get On Target Status)**ONT?**

Command Type:	Report Command
Description:	Report the on-target status of the given axis as reported by the servo-controller (e.g. the E-802).
Format:	ONT? X
Parameter:	X: Axis identifier.
Response:	Replies 1 when the given axis is on target, 0 otherwise.
Note:	For servo-controllers that do not provide an on-target signal, the return value is meaningless. See the servo-controller (submodule) manual for more information.

OVF? (Get Overflow Status)**OVF?**

Command Type:	Report Command
Description:	Reports the status of the overflow signal of the given axis.
Format:	OVF? X
Arguments:	X:Axis identifier.
Response:	Reply 1 if overflow occurred for the given axis, 0 otherwise. See the hardware manual for the board on which the E-816 is installed for information on when the overflow signal is on.

POS? (Read Real Position)**POS?**

Command Type:	Report Command
Description:	<p>Reports the current, actual position of the specified axis, as measured by the sensor and digitized by the ADC. The difference between the POS? and MOV? commands is that POS? gives the real position measured by sensor, while MOV? gives the last commanded position (see MOV).</p> <p>The reported value is an averaged result. The number of values used for the average can be changed with the AVG <i>n</i> command.</p>
Format:	POS? A
Arguments:	A : axis identifier. This parameter is required.
Response:	The current position of the specified axis. The value is floating point, in μm .

RST (Reset the Master Unit)	RST
------------------------------------	------------

Command Type: Configuration Command

Description: Reset the master unit.

Format: RST

Arguments: *none*

Response: *none*

NOTE **Wait about 10s to let the unit get ready for next command**

SAI? (Get Axis Index)	SAI?
------------------------------	-------------

Command Type: Report Command

Description: Get channel (axis) names of all connected devices.

Format: **SAI?**

Arguments: None

Response: String with all channel designators.

Example: You send:

SAI?

Response: ABCF

means 4 devices with channel name 'A', 'B', 'C' and 'F' are connected on the I²C bus.

SCH (Set Channel Name)	SCH
-------------------------------	------------

Command Type: System Configuration Command

Description: Set the channel name (axis designator) of the master unit. The axis name set is that to be used to address this unit when it is a slave. If the name is set to A with this command, the unit cannot be addressed as slave.

Format: **SCH c**

Arguments: *c*, the channel name of the device, can be any letter from A to X.

Note: The master unit can always be addressed using A; commands with A as channel designator will always be executed by the master unit. An "SCH A" command thus serves only to delete an unknown or unwanted setting that may have been made earlier.

Response: None

Example: You received three E-816s from PI and want to interlink them via I2C bus. The following steps should be performed:

Step 1: plug the RS-232 cable from the host into the first unit, set host software to correct baud rate, then set desired channel name by command:

SCH A

WPA 100

RST

Note that this unit will never be accessible as a slave.

Step 2: plug the RS-232 cable into the second unit, set correct baud rate, then set desired channel name by command:

SCH B

WPA 100

RST

Step 3: plug the RS-232 cable into the third unit, set correct baud rate, then set desired channel name by command:

SCH C

WPA 100

RST

Step 4: make sure all three units are connected to the I2C bus (e.g. plugged into the same chassis), and plug the RS-232 cable into the *first* unit. Type in the following command:

SAI?

Response: ABC

The device should reply ABC indicating that all 3 units are accessible via the one RS-232 port with the respective axis names.

The above procedure also saves these settings to flash ROM in each unit, so they will be available after subsequent power-ons.

SCH? (Get Channel Name)		SCH?
Command Type:	System Setting Command	
Description:	Report the channel name of the master unit.	
Format:	SCH?	
Arguments:	None	
Response:	Channel name of the master unit, will be an upper-case letter from A to X	

SPA (Set Parameter)	SPA
Command Type:	Configuration Command
Description:	Set specified parameters for the master unit. These are for the most part stage-related parameters obtained from calibration operations. This command saves the parameters in RAM only. To save these and other currently valid parameters to flash ROM, where they become the power-on defaults, you must use the WPA command. Parameter changes not saved with WPA will be lost when the E-816 is powered off. In this firmware version, each axis has 10 calibration parameters. See p. 30 in Section 8.3 "Calibration Register Structure" for the definitions of the different parameters. NOTE: parameters with IDs 1-6 are reserved for PI use only; customers have no write access to these parameters. Normally, the system is fully calibrated and tested before it leaves the factory.
Format:	SPA <i>An x.x</i>
Arguments:	<i>A</i> : axis identifier, must be either 'A' or the name of the master axis, <i>n</i> : the ID of the parameter, should be in range of 7 to 10.

SPA? (Get Parameter)	SPA?
Command Type:	Report Command
Description:	Read specified parameter setting for the master axis. Addressing this command to a slave axis gives a parameter error.
Format:	SPA? <i>An</i>
Arguments:	<i>A</i> : axis identifier, must be either 'A' or the name of the master axis <i>n</i> : the ID of the parameter, should be in range of 1 to 10.
Response:	Value of the specified parameter in floating point.

SSN? (Get Serial Number)	*IDN?
Command Type:	Report Command
Description:	Reports serial number of device on addressed axis
Format:	SSN? <i>A</i>
Arguments:	<i>A</i> : axis identifier
Response:	One-line string with device serial number, terminated by LF.

SVA (Set PZT Voltage Absolute)	SVA
---------------------------------------	------------

Command Type: Voltage Command

Description: Set the PZT voltage of the specified axis to the commanded value.
Before using this command, please make sure the E-816 has servo-control mode set OFF. (The E-816 simply provides an on/off signal for the associated servo-controller, which must be activated and properly configured. See the corresponding manuals for details)

Format: SVA Ax.x

Arguments: A : axis identifier

x.x: the commanded value in volts.
Both parameters are required.

Response: none

SVA? (Get the Commanded PZT Voltage)	SVA?
---	-------------

Command Type: Report Command

Description: Reports the commanded PZT voltage of the specified axis

Format: SVA? A

Arguments: A : axis identifier

Response: When used after an SVA or SVR command, reports the last commanded PZT voltage of the specified axis.

SVO (Set Servo-Control Mode)	SVO
-------------------------------------	------------

Command Type: Configuration Command

Description: Set the servo-control mode of the given axis. When ON, the Servo-on signal is sent to the PZT controller. See the manuals for the controller in question to ensure that no contradictory settings are in effect.

Format: SVO Xn

Arguments: X: Axis identifier.
n: 1 Servo-control mode, 1 for set to ON, 0 for set to OFF.

Response: None

SVO? (Get Servo-Control Mode)	SVO?
--------------------------------------	-------------

Command Type: Report Command

Description: Reports the current servo-control mode of the given axis.

Format: SVO? X

Arguments: X: Axis identifier.

Response: "1" for servo-control mode ON, "0" for OFF

SVR (Set PZT Voltage Relative)	SVR
---------------------------------------	------------

Command Type: Voltage Command

Description: Increase/Decrease PZT voltage of the specified axis by the commanded value.

Before using this command, please make sure the E-816 is has servo-control mode set OFF. (The E-816 simply provides an on/off signal for the associated servo-controller, which must be activated and properly configured. See the corresponding manuals for details)

Format: SVR Ax.x

Arguments: A: axis identifier
x.x: the commanded value in volts.
Both parameters are required.

Response: none

SWT (Set Wave Table Data)	SWT
----------------------------------	------------

Command Type: Configuration Command

Description: Sets a wave-table data point. The data is stored in EEPROM and can be reused after next power-up.

Format: **SWT Xn x.x**

Arguments: X: Axis identifier.
n: the index of the wave table data point, must be in 0-63.
x.x is the wave-table data point value, it is in units of μm when servo is ON otherwise in units of V.

Response: 0 if successful, 1 otherwise.

Example: see **WTO** command

SWT? (Get Wave Table Data)	SWT
-----------------------------------	------------

Command Type: Report Command

Description: Gets a wave-table data point.

Format: **SWT Xn**

Arguments: X: Axis identifier.
n: the index of the wave-table data point, must be in 0 to 63.

Response: Value of the wave-table data point

Example: see **WTO** command

VOL? (Read Real PZT Voltage)		VOL?
Command Type:	Report Command	
Description:	Reports the current, actual PZT voltage of the specified axis, as measured by the A/D converter. The difference between the VOL? and SVA? commands is that VOL? gives the current PZT voltage measured by the ADC, while SVA? gives the commanded voltage from the last SVA command.	
Format:	VOL? A	
Arguments:	A : axis identifier. This parameter is required.	
Response:	The current PZT voltage of the specified axis. The value returned is in floating point, in volts.	

WPA (Write Parameters)		WPA
Command Type:	Configuration Command	
Description:	Master unit only: the current values of parameters settable by SPA, AVG, BDR and SCH are written to flash ROM, where they become the new power-on defaults. Note that the RAM values of BDR and SCH do not go into effect until after they are written to ROM and the system reset, so the RAM values may differ from the current operating values.	
Format:	WPA xxx	
Arguments:	xxx: password (the password is 100)	
Response:	none	
Problem Solver:	Incorrect password	
NOTE	If the current RAM values are incompatible, the system may malfunction. Be sure that you have entered the correct parameter settings before using this command.	

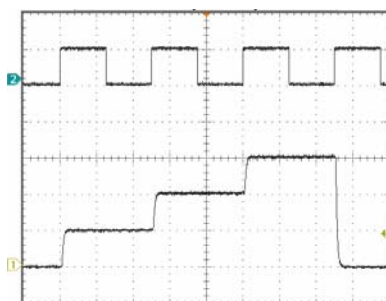
WTO (Enable/Disable Wave Table Output)**WTO**

Command Type:	Move Command
Description:	Sets the wave-table output mode for the given axis. During wave-table output drift compensation (see DCO) is not carried out even if set to 1.
Format:	WTO $Xn\ x.x$
Arguments:	<p>X: Axis identifier.</p> <p>n: the number of wave-table points to use for output. If $n=0$, wave-table output is disabled, otherwise wave-table data points from index 0 to $n-1$ are specified for output.</p> <p>$x.x$: if nonzero, output of the points specified by n will be started immediately and each point will be output for the amount of time specified by $x.x$ in milliseconds. Output will roll over from point $n-1$ to 0 and continue until stopped by a WTO $X0$ command.</p> <p>If omitted or 0, one wave-table point is output each time an external trigger signal is received. See User Manual for module on which E-816 submodule is installed for information on wave-table trigger signal routing.</p>
Response:	None
Remarks	Commands following WTO may be executed while wave-table output is in progress. It is even possible to change values in the wave table, with changes taking effect when the next point is output.

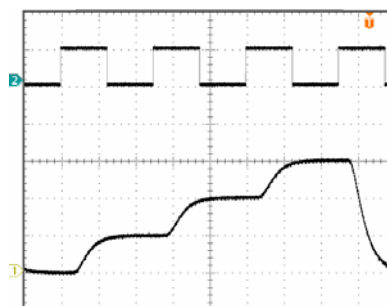
Example: **Servo Off, C-PZT = 5 μ**

Trigger signal

Piezo motion



10Hz



100Hz

Note: if the WTO command is entered with servo mode commanded OFF, the amplitudes are interpreted as voltages, otherwise the amplitudes are interpreted as positions (e.g. μm).

Command	Function
SVO A0	Servo OFF
SWT A00	Store 0 V as point 0
SWT A1 20	Store 20 V as point 1
SWT A2 40	Store 40 V as point 2
SWT A3 60	Store 60 V as point 3
WTO A4	Start wave output of 4 points per trigger signal

Trigger signal width: $\geq 100\ \mu\text{sec}$; Trigger interval $\geq 1.4\ \text{msec}$, i.e. max trigger frequency 700 Hz

7. Pin Assignments

The E-816 submodule plugs into two accurately placed sockets on a PCB, *j1* and *j2*, having the following pinout:

J1

PIN	
1	Analog Power Supply, +15V (5 mA)
2	Analog Power Supply, -15V (5 mA)
3	Analog GND
4	Digital Power Supply, +5V (60 mA)
5	Digital GND
6	RS232-RX
7	RS232-TX
8	RS232-RTS
9	RS232-CTS
10	Reserved

J2

PIN	
1	IN: Sensor Voltage
2	Analog GND
3	OUT: Target Voltage
4	IN: On-target signal
5	IN: Overflow signal
6	OUT: Servo ON/OFF control signal
7	IN: PZT voltage / 100 (i.e. approx. 0 to 1 V)
8	I2C-SCL
9	I2C-SDA
10	Wave table trigger signal input, trigger level 2 to 10 V

J3

Service connector, leave unconnected.

8. Appendix

8.1. Factory Defaults

The following E-816 factory defaults are valid for the first start-up, unless agreed otherwise before delivery.

- Number of readings to use for an average: 32*
- Channel name (axis designator): "A" and at most one other*, as specified when ordering; default is "A" only.
- Data rate: 9600 to 115,200 baud (default 115,200)*
- Data word: 8 bits
- Stop bits: 1
- Parity: none
- Flow control: RTS/CTS
- Termination character for commands: LF or CR, decimal 10 or 13
- Termination character for responses (always consist of a single line, unlike those from many other PI controllers): LF, decimal 10
- Cable connection (RS-232 null-modem): TxD - RxD
RxD - TxD
RTS - CTS
CTS - RTS
GND-.GND

Note: The host computer MUST use a flow-control protocol when communicating with the E-816.

8.2. Voltages and Positions

Commanding the PZTs can be done either by voltage-related or by displacement-related commands. When using voltage-related commands, servo-control must be disabled, when using position-related commands, it must be enabled (SVO command).

Note: With the E-816 there are no software limits on the commanded voltages or positions (unlike the E-516). If a voltage or position beyond the travel range of the connected axis is sent, it will be accepted, but the axis will move only to the limit position. No error is set, and the overflow condition may not go on.

Example: Commanding output voltages:

SVO A0	disable servo-control on channel A
SVA A58.20	output 58.2 volts to the PZT connected to channel A
SVA A20	output 20.00 volts to the PZT

Example: Commanding PZT displacements:

SVO A1	enable servo-control on channel A
MOV A12.995	move channel-A PZT to 12.995 microns

* This value may be changed and then saved to ROM, where they become the new power-up defaults.

MOV A14	move PZT to 14 microns
MVR A-1	move PZT to 13 microns (-1 relative to last target position)

The **VOL?** and **POS?** commands are used for reading current voltages and positions.

The values reported by VOL? and POS? are the actual output voltages and displacements, respectively. If no sensor is connected, POS? reports zero.

8.3. Calibration Register Structure

One of the key benefits of the E-816 is the calibration register structure. After the device has completed the calibration runs in our lab, absolute positions and voltages can be commanded. The calibration data is stored in non-volatile ROM. Write access is protected by password.

As long as the system configuration has not changed or readjusted, calibration data is valid and does not need to be modified. Note that the values in the E-816 calibration registers operate “on top of” the hardware calibration settings of the potentiometers on the servo-control and amplifier boards. If the latter are adjusted, the corresponding E-816 parameters must be reset.

The E-816 has 10 calibration registers, as shown in the E-816 block diagram below. Six registers are for internal use and will be set by PI only (shown with dotted lines). For safety sake, we grant customers read access only. Four calibration registers—Ksen, Osen, Kpzt, Opzt—can be changed by the customer, based on the servo-controller, stage and amplifier used.

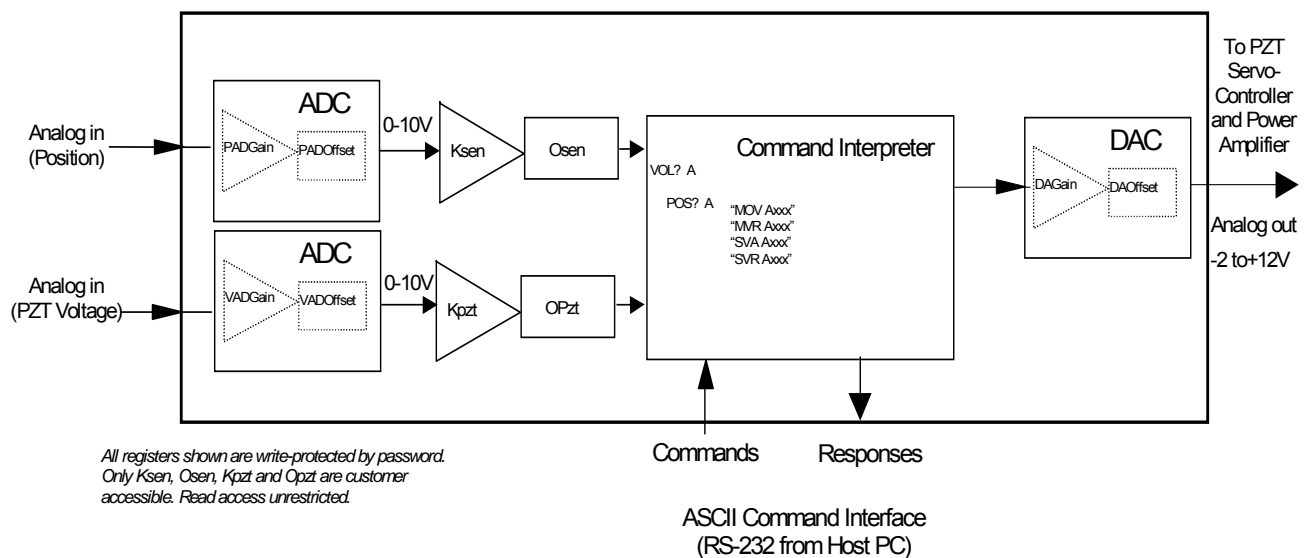


Fig. 3: E-816 block diagram and calibration registers

CAUTION: E-816s are fully calibrated together with the other components of the system before being shipped from PI. It is not necessary for customers to recalibrate them.

Calibration should only be done after consultation with PI, otherwise the internal configuration data may be destroyed by erroneous operation.

All stage-specific data are stored in flash ROM and a copy is made in RAM at power-up. The SPA and SPA? commands can be used to access the RAM copy. These commands have the following format:

SPA Xn v.v

SPA? Xn

where:

SPA[?] command mnemonic
X axis indicator
n parameter ID
v.v parameter, floating point
LF LineFeed (Char #10).

Example:

Command: SPA? A1

Report: -3.6427

ID	Name	Type	Parameter Description
1*	VADGain	Float	Gain of the A/D converter which measures the PZT voltage*
2*	VADOffset	Float	Offset of the A/D converter which measures the PZT voltage *
3*	PADGain	Float	Gain of the A/D converter which measures the sensor voltage. *
4*	PADOffset	Float	Offset of the A/D converter which measures the sensor voltage. *
5*	DAGain	Float	Gain of the D/A converter *
6*	DAOffset	Float	Offset of the D/A converter *
7	Ksen	Float	Sensor coefficient K_s in $\mu\text{m/V}$ (when sensor output changes by 1V, the position change of stage is $K_s \mu\text{m}$)
8	Osen	Float	Sensor offset O_s in μm (when sensor output is 0V, the actual position of stage is $O_s \mu\text{m}$)
9	Kpzt	Float	PZT voltage amplifier coefficient K_{pzt} in V/V (when D/A converter output changes by 1V, the PZT voltage at amplifier output changes by $K_{\text{pzt}} \text{ V}$)
10	Opzt	Float	PZT voltage amplifier offset O_{pzt} in V (when D/A converter output is 0V, the PZT voltage at amplifier output is O_{pzt})

* Parameters 1-6 are for PI use only, normally no write access for customer

Since the E-816 is designed as a plug-in module for PZT controllers and can be used with different combinations of high-voltage and low-voltage PZT amplifiers and position servo-controllers, the last 4 parameters (K_s , O_s , K_{pzt} , O_{pzt}) must be set to values appropriate for the PZT stage(s) and amplifier(s) installed. If the E-816 was ordered together with all amplifier plug-in modules, servo-

controllers and PZTs, these 4 parameters will have been set at the factory before shipment.

If, however, the E-816 is being used with separately purchased PI stages, the above 4 parameters must be determined and set. These parameters must also be reset if the hardware calibration elements on the servo-controller (e.g. E-802 submodule), sensor readout (e.g. E-801 submodule) or power amplifier (e.g. E-621) are moved. The formulas for calculating these parameters are:

$$K_{sen} = (P_{10} - P_0) / 10.0$$

$$K_{pzt} = (V_{10} - V_0) / 10.0$$

$$O_{sen} = P_0$$

$$O_{pzt} = V_0$$

Where:

P_{10} is the actual stage position when sensor monitor outputs 10 V

P_0 is the actual stage position when sensor monitor outputs 0V

V_{10} is the actual PZT voltage when the input to the PZT amplifier is 10 V in open-loop mode (servo OFF)

V_0 is the actual PZT voltage when the input to the PZT amplifier is 0 V in open-loop mode (servo OFF).

Example 1:

A stage has sensor monitor output of 0-10 V, the stage travel is to be 0 μ m to approx. 50 μ m, the PZT voltage is to be 0 and 100V when analog input is 0V and 10V, respectively, in open-loop mode. The settings must then be:

$$K_{sen} = (50.0 - 0.0) / 10.0 = 5.0$$

$$O_{sen} = 0.0$$

$$K_{pzt} = (100.0 - 0.0) / 10.0 = 10.0$$

$$O_{pzt} = 0.0$$

Example 2:

The PZT voltage in open-loop mode is to be -0.5 and 100.5 V when the control input is 0 V and 10 V respectively. Then:

$$K_{sen} = (25 - (-25)) / 10.0 = 5.0$$

$$O_{sen} = -25$$

$$K_{pzt} = (100.5 - (-0.5)) / 10.0 = 10.1$$

$$O_{pzt} = -0.5$$

Example 3:

A stage has sensor monitor output of 0-10 V, the nominal stage extension is 0 μ m-15 μ m, when the PZT voltage is 0 - 100V. However, after being calibrated with an external standard, the real PZT extension is found to be 0 μ m-14.5 μ m and the PZT voltage to be 0 - 98 V. Then

$$K_{sen} = (14.5 - 0) / 10.0 = 1.45$$

$$O_{sen} = 0$$

$$K_{pzt} = (98 - 0) / 10.0 = 9.8$$

$$O_{pzt} = 0$$

8.4. Drift Compensation

Drift compensation is provided to eliminate drift in the digital-analog converters on the E-816. Activating it with the DCO command (p. 16) will avoid unwanted change in displacement over time in static operation. It should be deactivated during dynamic operation. Drift compensation works by eliminating unjustified, slow changes in the E-816 analog-side target output based on the position, piezo voltage and digital target.

Drift compensation mode is disregarded during wave-table output. If DCO was on, after the wave table output has finished, it again becomes active. This can cause

slight motion as the digital and analog positions adjust. Therefore, if it is important to hold the position at the end of wave-table motion, deactivate DCO before starting.

8.5. Master/Slave Command Processing

If, for example, the computer sends the following command to the master unit:

MOV C5.0

The master unit will process it as follows:

- (1) If the master unit has been named channel "C", it will process the command and issue the response, if any.
- (2) Otherwise, the master unit looks for the unit with channel name "C". If it finds one, it forwards the command to that unit. If it does not find one, it sets an error status bit
- (3) The slave unit with channel name "C" receives the command from the master, processes it, and if necessary, returns a response.
- (4) The master unit receives the response (if any) from the slave unit and forwards it to the host computer.

The terms "channel" and "axis" are used interchangeably, as are "designator" and "name".

Note: The master unit is always reachable with channel designator "A". That means that if the computer sends a command like MOV A5.0, the command will always be processed by the master unit, even if there is a slave unit set up with channel name "A".

8.5.1. Setting Channel Names

To set the channel name of an E-816, you must:

- Connect that E-816 to the computer with the serial communications cable (make it the master unit). Now this E-816 is reachable with the special channel name "A".
- Set the terminal software to the same baud rate as the E-816.
- Use the **SCH** x command to assign the desired channel name, x, to the E-816. "A" must not be assigned to a unit which will later be accessed as a slave.
- Use the "WPA 100" command to write the setting to the E-816 EEPROM.
- Reset the device with the RST command or with a power off/on reset.

Example:

You receive an E-816 from PI and want to set its channel name to "E". The default baud rate of the E-816 is 115.2 kbps. Connect the device to the computer, start *PI Terminal* or *PZTControl* at a baud rate of 115.2 kbps, and send the following commands:

SCH?

Response: A

SCH E

SCH?

Response: E

WPA 100

RST

8.5.2. Setting Correct Baud Rate of Host Software

If you are networking multiple E-816s with different baud rates, you must set the baud rate of the host software to be the same as the baud rate of the current master unit. Otherwise all communication will fail.

8.5.3. Checking Connection and Master Unit

If you are networking multiple E-816s, you can get the axis names of all connected units with the SAI? command and determine which is the master unit by asking its axis name with the SCH? command. If these commands time out, there is probably a problem with the connection to the master unit. Check the baud rate settings as explained under the BDR command.

Example:

SAI?

Response: BCF

SCH?

Response: C

The responses indicate that 3 devices with channel names B, C and F are connected on the I²C bus. The master unit is C, meaning that it is addressable either as C or A.

8.5.4. Enabling Computer-Controlled Operation

The circuit board on which the E-816 is installed must be properly configured to allow computer control and to allow the E-816 to switch the servo-controller on and off. Improperly set main-board jumpers may not make themselves immediately apparent. With the E-621, for example, sw1 and sw3 must be OFF, sw2 ON and jumper J1 in position 2-3. See the corresponding User Manual for details.

8.6. Interfaces

RS-232, baud rate 9.6 – 115.2 kbps, RTS/CTS, 8 data bits, 1 stop bit, no parity

8.6.1. Target (Analog) Output

Resolution	20 bit
Output Range	0 to 10V
Update Rate	500 Hz
Precision	±1 mV w/o drift compensation

8.6.2. Sensor (Analog) Input

Resolution	20 bit
Input Range	0 to 10 V
Update Rate	6.4 kHz
Noise	2.5 LSB rms

Precision ± 1 mV

8.6.3. PZT Voltage (Analog) Input

Resolution piezo voltage 10 bit
 Input Range -0.25 V to 1.25 V
 Noise 1 LSB rms

8.6.4. Wave Table Trigger (Digital) Input

voltage range 2 to 10 V
 max. freq. 700 Hz.
 min. width: 100 μ s

8.6.5. Other Digital I/O

TTL

8.7. Wave Table Functionality

(See User Manual for module on which E-816 submodule is installed for information on wave table trigger signal routing)

Storage Depth 64 data points
 Output Rate up to 700 Hz (external trigger signal)

8.8. Read/Write Speed

Command Destination and Type	Time	Oper./s
Command for master unit, no response	3.3 ms	300
Command for master with 1* analog reading and response	4.2 ms	238
Command for master with 32* analog readings and response	9 ms	111
Command for any slave unit, no response	8 ms	125
Command for any slave with 1* analog reading and response	11.8 ms	85
Command for any slave with 32* analog readings and response	17 ms	60

*The number of analog readings averaged to determine a value is settable on the master unit with the AVG command; the factory default is 32.

