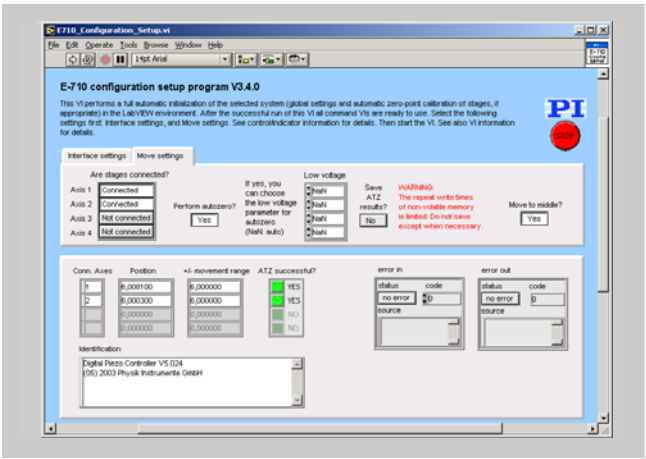
Piezo · Nano · Positioning

PZ 141E Software Manual

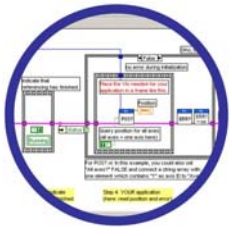
E-710 LabView Driver Library

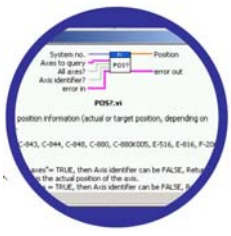
Release: 3.4.4b Date: 2007-01-10



This document describes software for use with the following product(s):

- E-710 Digital Piezo Controller;
  3- and 4-axis versions firmware
  rev. 5.025/6.025 or newer and 7.xxx,
  6-axis version firmware
  rev. 2.12 or newer





© Physik Instrumente (PI) GmbH & Co. KG
Auf der Römerstr. 1 · 76228 Karlsruhe, Germany
Tel. +49 721 4846-0 · Fax: +49 721 4846-299
info@pi.ws · www.pi.ws

Table of Contents

© Copyright 2007 by Physik Instrumente (PI) GmbH & Co. KG

Release: 3.4.4b
File:E710_GCSLabVIEW_PZ141E_344b.doc, 1952768 Bytes

0. Disclaimer

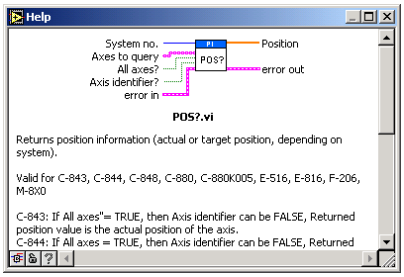
This software is provided “as is”. Physik Instrumente (PI) does not guarantee that this software is free of errors and will not be responsible for any damage arising from the use of this software. The user agrees to use this software on his own responsibility.

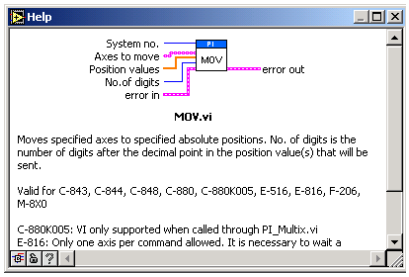
1. Introduction

The LabVIEW software consists of a collection of virtual instrument (VI) drivers. All functionality involves invoking one or more VIs with the appropriate parameter and global variable settings.

These VIs are provided to ease the task of programming your application. They, and the accompanying documentation, assume a prior knowledge of proper LabView programming techniques. The provided "Simple Test” and “Configuration Setup” VIs help to solve the essential initialization steps, but are not intended to provide an out-of-the-box, universal solution to a particular application.

To minimize the need for consulting the manual during programming, each VI comes with a detailed VI description that appears in the *Context Help* window when you move the cursor over the VI icon. Use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window.





1.1. PI General Command Set (GCS)

This VI driver set supports the *PI General Command Set*, which is based on ASCII communication with well-defined commands and replies. This makes it possible to control different PI systems, such as the *E-516 Display Module* or the *C-880 Multi-Axis Controller*, with only one driver set simply by “wiring” the correct command parameters to the associated VIs.

Translation Libraries

To control PI systems that are not originally compatible with the *PI General Command Set*, such as the *E-710 Digital Piezo Controller* or the *C-843 Motion Control Board*, libraries are used to translate *PI General Command Set* commands to other controller-specific languages. The universal library which adds this functionality is GCSTranslator.dll; it must be installed on the computer in the GCS_LabVIEW\Low Level folder, no matter whether the system being controlled is *PI General Command Set* compatible or not. To control certain systems (such as a non-GCS-compatible system or a PC board), one or more system-specific DLLs and data files (e.g. PIStages.dat) must also be installed. If you install this driver set from within the setup program of the PI software CD ROM, this installation is done automatically. If you want to install this driver set manually, please run “GCSLibrarySetup.exe” from the CD-ROM that came with your system. This setup

tool makes sure that all necessary libraries and their data files are correctly registered in the Windows™ environment and can be found by the GCS drivers.

If LabVIEW still cannot find PIStages.dat, it may be because it is marked read-only. To see, open Microsoft Explorer, right-click the file PIStages.dat and select *Properties*. Make sure that the *read-only* attribute is not checked.

Once the libraries and data files for the system to control are installed, this LabVIEW driver set can be used to control a non-GCS-compatible system just like any GCS-compatible system, and PCI/ISA-based controller boards by selecting “DLL” as communication interface (see Section "First Steps for GCS-Compatible PI Controllers" on p. 7 and the "PI Open Interface.vi" / “PI Open Interface of one system.vi“ command description on p. 20 / 20).

Units and GCS

The GCS system uses physical units of measure. Most controllers and GCS software have default conversion factors chosen to convert hardware-dependent units (e.g. encoder counts) into mm or degrees, as appropriate. These defaults are generally taken from a database of stages that can be connected. The direction of motion associated with positive and negative relative moves can also be controlled by parameter settings. In some cases an additional scale factor can be applied, making a second physical unit available without overwriting the conversion factor for the first. It is also sometimes possible to enter a conversion factor as numerator and denominator of a fraction, reducing the number of digits and outside calculations needed for high-precion entry of gearhead system values. See the DFF.vi and SPA.vi command descriptions, taking special note of the sections referring specifically to your controller.

1.2. Scope of This Manual

This manual covers only VIs which can be used with the product with which it came.

1.3. VI Structure

The folder structure of the LabVIEW drivers consists of the main folder “GCS_LabVIEW” with the sub-folder “Low Level”.

The main folder “GCS_LabVIEW" contains a terminal VI, a configuration VI (XXXX_Configuration_Setup.vi with XXXX being the PI product number of your system), a simple test VI, and, if available, several sample programs.

The sub-folder “Low Level” contains VIs for the following functions:

- Establishing communication with different PI systems which support the PI General Command Set via RS-232 or GPIB interfaces
- Defining the parameter IDs of the connected axes
- Sending and receiving ASCII characters to/from the specified system
- Sending system-specific commands (system-specific commands are separated into function-specific LLBs).

Additionally, the sub-folder “Low Level” contains GCSTranslator.dll.

Following the data flow concept of LabVIEW, all VIs have their wiring inputs on the left side and their wiring outputs on the right side of each connector pane. For quick integration, this **connector pane** in most cases has the following pattern:

1	7	9	11	13	15
2					16
3					17
4	8	10	12	14	18
5					19
6					20

The terminals are assigned as follows (if the mentioned, control/indicator is present in one of the supplied libraries):

1 System number
2 Optical board, Interface, or other main input control
3 Axes to query, Affected axes, Number of systems, or other main input control
4 All axes?, Invert order?, or other main input control
5 Axis identifier?, No. of digits, or other main input control
6 Error in
7 Parameter number, Without axis ID?, or other input control
8 Step size, or other input control
9 AA step size, or other input control
10 Input control
11 Input control or output indicator
12 Input control or output indicator
13 Input control or output indicator
14 Input control or output indicator
15 Hidden error, Connected axes, String read, or other main output indicator
16 Axes to query out, Bytes read, or other main output indicator
17 No. of rows, or other main output indicator
18 Output indicator
19 Output indicator
20 Error out

Also note that this driver set does not use the standard LabVIEW error numbers recommended by National Instruments, but rather those used by PI controllers. As a result, the error texts displayed by LabVIEW will not describe the error accurately. Use “GCSTranslateError.vi” to get the description of a PI GCS error number. Some VIs use an additional indicator Hidden error to indicate that the selected system has been queried for a controller error with „ERR?“ and reported an error number ≠ zero.

See also chapter 5 on p. 78 for a summary of error numbers produced by this driver set.

If you are using LabVIEW 7.1, uncheck *Enable automatic error handling dialogs* in *Tools→Options→New and Changed in 7.x* to prevent that LabVIEW suspends execution and displays an error dialog box for any error that occurs during the execution of the VIs.

Important:

Before running any VIs to control a connected system, "**XXXX_Configuration_Setup.vi**" (located in the main folder, with XXXX being the PI product number of your system) must be run. This initialization VI performs all necessary steps automatically:

1. It opens the communications port,
2. It defines the IDs for the connected axes,
3. It references the connected stages, depending on if the controller requires a referencing before axes can be moved and on your custom settings,
4. It defines the controller name.

After these steps all parameters are saved into global variables, so that other VIs invoked during the same LabView session can access this data at runtime.

As the initialization is a complex procedure which uses a large number of sub-VIs, **XXXX_Configuration_Setup.vi** is password-protected, meaning that you cannot see or modify the diagram. In this way, the full initialization is packed into one single and fully tested procedure which you simply insert into your own application program. For security reasons as well as your convenience, we recommend that you not modify this VI.

For testing, the easiest method is to call "PI Terminal.vi", which is located in the "GCS_LabVIEW" main folder. This is a "stand-alone" routine that calls "PI Ask for Communication Parameters.vi" first and then opens the specified ports. It does not, however, define the connected axes of the (motion) systems. A more system-specific sample VI is "XXXX_Simple_Test.vi" (with XXXX being the PI product number of your system), also located in the "GCS_LabVIEW" main folder.

1.4. Working with two PI products which understand PI's General Command Set (GCS) in LabVIEW

When installing the LabVIEW programming support for two different PI products, there are two "Low Level" folders installed, one in each product-specific LabVIEW driver set. This is because every product comes with only the VIs which are used with the product. Another product may have different libraries or different library contents due to the product supporting more or fewer functions. When working with two product-specific LabVIEW driver set installations on one computer, it is important to make sure that LabVIEW always uses the right libraries.

a) When working separately with two products, the "Low Level" folder of each product must be located in the same folder as the product-specific main VI which calls sub-VIs from the product-specific driver set. Otherwise LabVIEW will start searching for sub VIs whereever it finds them, which may result in version conflicts and "broken Run" arrows. Please make sure that no VIs are saved under LabVIEWs own "user.lib" sub-folder. If they are LabVIEW will always find them there first, which will cause errors in many cases.

b) When working with two products in parallel, the libraries should be combined. Under LabVIEW 7.1 (but also under older LabVIEW versions), this is quite easy using LabVIEWs "Library manager", which can be found on the "Tools" menu. First, make a backup copy of the older library you want to combine with the newer one. In Library manager, open the newer library (e.g. "Special.llb" from the C-865 release) in the left window, and the older library (e.g. "Special.llb" from the older C-843 release) in the right window. Then choose "Show dates", "Disable files with identical dates", and "Sort alphabetically". This will show you only the VIs which have different file dates or which are only present in one of these libraries, but will not show any identical VIs. Now select all VIs which are still shown on the left window and copy them to the right window. In this way, you get a library with all VIs used by both product driver sets, and the newest version of each. Do this for all libraries in the low level folder. Make sure to work thereafter with the combined libraries instead of the product-specific libraries.

1.5. First Steps for GCS-Compatible PI Controllers

1.5.1. C-843

To control one or more C-843 boards with this driver set, “C843_GCS_DLL.dll”, “MC.dll”, “PiStages.dat” and the C-843 device driver must be installed on your computer. See chapter 1.1 for information about methods for proper installation of the first three items. A description of how to install the C-843 device driver is given in the C-843 User Manual. The following steps must then be performed:

Step 1 (advanced users can skip this step): To check communication with the C-843 board in the host PC, run “C843_Simple_Test.vi”. This VI will return the ID string of the C-843 board and the axis IDs of the connected axes. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: C843_Configuration_Setup.vi May Cause Move

When you start “C843_Configuration_Setup.vi” with All axes? = TRUE, the VI will automatically determine which axes have a reference switch and which have limit switches and, if the referencing mode of these axes is ON, will move these stages to these sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move. If referencing is not possible (because the connected stage has no reference or limit switch) or not desired, referencing mode (the mode which tells the controller to reference the stages or not) can be switched off. See description of RON for details and warnings.

Open “C843_Configuration_Setup.vi”. Select your C-843 board (2- or 4-axis version, board number) and leave “Use dialog to define connected stages” = TRUE. Run the VI. In the following screen, specify which stages you have connected to which axes and press OK. This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close “C843_Configuration_Setup.vi”; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement “C843_Configuration_Setup.vi” as an initialization VI in your software. See chapter 3 for a detailed description of “C843_Configuration_Setup.vi” and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

If you do not find your stage in the drop-down list, press CANCEL. You can then either define a User Stage with the Stage Editor, or you can contact PI to see about getting a new stage list: the "PIStages.dat" file contains all relevant stage parameters.

Default axis names are 1 to 4, but can be changed using “SAI.vi”.

1.5.2. C-843.PM

With this driver set, “C843_PM_GCS_DLL.dll”, “MC.dll”, “PiStages.dat” and the C-843 device driver must be installed on your computer if you wish to control axes on PIs linear piezo motor stages. It can be used with axes on other motorized stages connected to the same or other C-843 boards in the same machine. See chapter 1.1 for information about methods for proper installation of the first three items. A description of how to install the C-843 device driver is given in the C-843 User Manual. The following steps must then be performed:

Step 1 (advanced users can skip this step): To check communication with the C-843 board in the host PC, run “C843_PM_Simple_Test.vi”. This VI will return the ID string of the C-843 board and the axis IDs of the connected axes. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: C843_PM_Configuration_Setup.vi May Cause Move

When you start “C843_PM_Configuration_Setup.vi” with All axes? = TRUE, the VI will automatically determine which axes have a reference switch and which have limit switches and, if the referencing mode of these axes is ON, will move these stages to these sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move. If referencing is not possible (because the connected stage has no reference or limit switch) or not desired, referencing mode (the mode which tells the controller to reference the stages or not) can be switched off. See description of RON for details and warnings.

Open “C843_PM_Configuration_Setup.vi”. Select your C-843 board (2- or 4-axis version, board number) and leave “Use dialog to define connected stages” = TRUE. Run the VI. In the following screen, specify which stages you have connected to which axes and press OK. This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close “C843_PM_Configuration_Setup.vi”; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement “C843_PM_Configuration_Setup.vi” as an initialization VI in your software. See chapter 3 for a detailed description of “C843_PM_ Configuration_Setup.vi” and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

If you do not find your stage in the drop-down list, press CANCEL. You can then either define a User Stage with the Stage Editor, or you can contact PI to see about getting a new stage list: the "PIStages.dat" file contains all relevant stage parameters.

Default axis names are 1 to 4, but can be changed using “SAI.vi”.

1.5.3. C-848

Step 1: The C-848 controller is delivered pre-configured. Before you start, please check that the current configuration matches your stage connections. See the C-848 User Manual for a detailed description of this step.

Step 2 (advanced users can skip this step): To check communication between the C-848 controller and the host PC, run "C848_Simple_Test.vi". This VI will return the ID string of the C-848 controller, the axis IDs of the connected axes, and their current positions. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions. Please make sure that you have all reported axes connected. If you want to work with only some of these axes, remember the IDs of those axes so that you can enter them in the Axes to move control in "C848_Configuration_ Setup.vi"; thereafter the other axes will not be moved when executing "C848_Configuration_Setup.vi".

Step 3:

WARNING: C848_Configuration_Setup.vi May Cause Move

When you start "C848_Configuration_Setup.vi" with All axes? = TRUE, the VI will automatically determine which axes have a reference switch and which have limit switches and, if the referencing mode of these axes is ON, will move these stages to these sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move. If referencing is not possible (because the connected stage has no reference or limit switch) or not desired, referencing mode (the mode which tells the controller to reference the stages or not) can be switched off (see chapter 3). See description of RON for details and warnings.

To control one or more C-848 controllers with this driver set, run "C848_Configuration_Setup.vi". This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close "C848_Configuration_Setup.vi"; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement "C848_Configuration_Setup.vi" as an initialization VI in your software. See chapter 3 for a detailed description of "C848_Configuration_Setup.vi" and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Before using a joystick connected to the C-848 controller, calibrate the joystick by running "PI Terminal.vi". Type "JEN CALIB" and follow the instructions on the screen.

Before using a joystick connected to the host PC, install joystick driver and calibrate joystick in the Windows control panel.

1.5.4. C-865

To control one or more C-865s with this driver set, “C865_GCS_DLL.dll”, “ MC_C865.dll ”, and “PiStages.dat” must be installed on your computer. See Section 1.1 for information about methods for proper installation of the first three items. The following steps must then be performed:

Step 1 (advanced users can skip this step): To check communication with the C-865 controller, run “C865_Simple_Test.vi”. This VI will return the ID string of the C-865 controller. See Section 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: C865_Configuration_Setup.vi May Cause Move

When you start “C865_Configuration_Setup.vi” with Is axis connected and can be moved? = TRUE, the VI will automatically determine if the connected axis has a reference switch or limit switch and, if the referencing mode of this axis is ON, will move the stage to one of the switches. It is therefore important to make sure that items connected to or mounted on the connected stage will not be damaged by such a move. If referencing is not possible (because the connected stage has no reference or limit switch) or not desired, referencing mode (the mode which tells the controller to reference the stage or not) can be switched off. See description of RON for details and warnings.

Open “C865_Configuration_Setup.vi”. Select the RS-232 settings (port number and appropriate baudrate) and leave Use dialog to define connected stage = TRUE. Run the VI. In the following screen, specify which stage you have connected and press OK. This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of the axis ID, the initialization of the connected stage, including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close “C865_Configuration_Setup.vi”; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement “C865_Configuration_Setup.vi” as an initialization VI in your software. See Section 3 for a detailed description of “C865_Configuration_Setup.vi” and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

If you do not find your stage in the drop-down list, press CANCEL. You can then either define a User Stage with the Stage Editor, or you can contact PI to see about getting a new stage list: the "PIStages.dat" file contains all relevant stage parameters.

Default axis name is “1”, but can be changed using “SAI.vi”.

If the controller does not respond, please reset it using the reset button (unlabeled) on the rear panel of the C-865 controller.

See also “C865_Sample_Application_1.vi” and “C865_Sample_Application_1a.vi” as sample VIs showing how to implement “C865_Configuration_Setup.vi” as the initialization VI for the C-865 in your application.

Before using a joystick connected to the host PC, install joystick driver and calibrate joystick in the Windows control panel.

1.5.5. C-880

Step 1: The C-880 controller is delivered pre-configured. Before you start, please check that the current configuration matches your stage connections. See the C-880 User Manual for a detailed description of this step.

Step 2 (advanced users can skip this step): To check communication between the C-880 controller and the host PC, run "C880_Simple_Test.vi". This VI will return the ID string of the C-880 controller, the axis IDs of the connected axes, and their current positions. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions. Please make sure that you have all reported axes connected. If you want to work with only some of these axes, remember the IDs of those axes so that you can enter them in the Axes to move control in "C880_Configuration_ Setup.vi"; thereafter the other axes will not be moved when executing "C880_Configuration_Setup.vi".

Step 3:

WARNING: C880_Configuration_Setup.vi May Cause Move

When you start "C880_Configuration_Setup.vi" with All axes? = TRUE, the VI will automatically determine which axes have a reference switch and which have limit switches and, if the referencing mode of these axes is ON, will move these stages to these sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move. If referencing is not possible (because the connected stage has no reference or limit switch) or not desired, referencing mode (the mode which tells the controller to reference the stages or not) can be switched off (see chapter 3). See description of RON for details and warnings.

To control one or more C-880 controllers with this driver set, run "C880_Configuration_Setup.vi". This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close "C880_Configuration_Setup.vi"; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement "C880_Configuration_Setup.vi" as an initialization VI in your software. See chapter 3 for a detailed description of "C880_Configuration_Setup.vi" and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Before using a joystick connected to the C-880 controller, calibrate the joystick by running "PI Terminal.vi". Type "JEN CALIB" and follow the instructions on the screen.

Before using a joystick connected to the host PC, install joystick driver and calibrate joystick in the Windows control panel.

1.5.6. C-880K005

To control a C-880K005 controller with this driver set, you must perform the following steps:

1. Run “PI Open Interface.vi”
2. Run “Multix axis assignment.vi”.
3. Reference all connected stages using “PI_Multix.vi” with Command = REF and All axes? = TRUE. If no referencing is desired, refencing mode can be switched off using Command = RON. With referencing mode off, only relative moves can be commanded (using Command = MVR), unless the actual position is set with Command = POS.vi. Thereafter both relative and absolute moves can be commanded.

OR

1. Run C880K005_Simple_Test.vi
2. Proceed as in step 3 above.

After these steps, a number of selected commands can be used by calling “PI_Multix.vi”.

The control concept of the C-880K005 is based on the assumption that several C-880 multi-axis motion controllers are connected to the C-880K005, which is commanded over a single interface from the host PC. In this way, the number of axes to command is not limited by the number of connectors available on a single C-880. To ease handling so many axes, the user does not have to worry about the individual C-880 controllers, but only the C-880K005 controller with axes 1 to N, N being the sum of all axes connected to all interconnected C-880s.

The C-880K005 is called the “controller” and handles the communication to all connected C-880 controllers, which are called VControllers (virtual controllers). “PI_Multix.vi” must be used to send commands. For ease of operation, when running “Multix axis assignment.vi”, all connected axes of all connected C-880 controllers are queried and the axis IDs 1 to N are assigned to these axes automatically.

A C-880 connected to the C-880K005 can be directly commanded by setting it active (“ACT.vi”). The C-880K005 communications controller then passes subsequent commands to the active C-880, except for commands which, by their nature, must be directly handled by the C-880K005 (e.g. WAA).

Example: Three C-880 controllers are connected to the C-880K005, and each C-880 controller has 12 axes designated A to L on each of the C-880’s. These ID’s are taken as VAxis IDs and the axis IDs 1 to 36 are assigned for the C-880K005 controller. To command axis A of C-880 1 (VController 1) and axis B of C-880 3 (VController 3), the user commands axes 1 and 26 in “PI_Multix.vi”; the axis and value parsing is done internally.

See “C880K005_Simple_Test.vi” and “C880K005_Configuration_Setup.vi” for sample programs for the driver configuration of the C-880K005.

1.5.7. E-516

Step 1 (advanced users can skip this step): To check communication with the E-516 controller, run "E516_Simple_Test.vi". This VI will return the ID string and the help string of the E-516 controller, the available axis IDs and positions of all axes. See Section 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: E516_Configuration_Setup.vi May Cause Move

When you start "E516_Configuration_Setup.vi" with Move all axes to middle? = TRUE, or Move all axes to middle? = FALSE and Axes to move = TRUE for some axes, the VI will move all axes or the selected axes to their middle positions. It is therefore important to make sure that items connected to or mounted on the connected stages will not be damaged by such a move.

Open "E516_Configuration_Setup.vi". First select the interface settings (Interface = "RS232" or "GPIB", RS232 settings = Portnumber and appropriate Baudrate, or GPIB settings = Bus and Address). Select whether a wave generator output is to be stopped (if you are not sure if there is any wave generator output running, leave this control TRUE) and whether the axes are to be moved to the midpoints of their travel ranges.

Then run the VI. It performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: definition of the axis IDs, the online setting of the controller, the servo setting of the axes and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close "E516_Configuration_Setup.vi"; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can include "E516_Configuration_Setup.vi" as an initialization VI in your software. See Section 3 for a detailed description of "E516_Configuration_Setup.vi" and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Default position unit is µm, default velocity unit is µm/ms.

To use wave-generator-specific VI's, whose names start with "WGWaveEditor_*.vi", "WGWaveEditor.dll" must be installed on your computer. During the installation of "WGWaveEditor.dll" "NTGraph.ocx" will be installed also. If "NTGraph.ocx" is not registered correctly in the Windows environment, the editor of "WGWaveEditor.dll" will not function.

See "E516_WaveGenerator_Sample_Program.vi" for a sample program using these VIs.

1.5.8. E-710

To control one or more E-710s with this driver set, “E710_GCS_DLL.dll” must be installed on your computer. See Section 1.1 for information about methods for a proper installation. The following steps must then be performed:

Step 1 (advanced users can skip this step): To check communication with the E-710 controller, run “E710_Simple_Test.vi”. This VI will return the ID string of the E-710 controller and the available axis IDs. See Section 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: E710_Configuration_Setup.vi May Cause Move

When you start “E710_Configuration_Setup.vi” with Perform autozero? = TRUE and/or Move to middle? = TRUE, the VI will perform an automated zero-point calibration of the connected linear axes and/or move these axes to their middle positions. It is therefore important to make sure that items connected to or mounted on the connected stages will not be damaged by such a move.

Open “E710_Configuration_Setup.vi”. First select the interface settings (Interface = “RS232” or “GPIB”, RS232 settings = Portnumber and appropriate Baudrate, or GPIB settings = Bus and Address), and specify if stages are connected. Select whether the automated zero-point calibration is to be performed (linear axes only), whether the Low voltage parameter is to be defined automatically or manually, and whether the axes are to be moved to the midpoints of their travel ranges.

WARNING:

The repeat write times of the internal non-volatile memory of the E-710 controller are limited. Do not use commands which write to the EEPROM (e.g. WPA, SEP) except when necessary. See User Manual for details.

Then run the VI. It performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of the axis IDs, the initialization of the axes (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close “E710_Configuration_Setup.vi”; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement “E710_Configuration_Setup.vi” as an initialization VI in your software. See Section 3 for a detailed description of “E710_Configuration_Setup.vi” and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Default axis IDs are “1”,”2”,”3”,”4”.

Default position unit is µm, default velocity unit is µm/ms.

Due to the emulation of the native E-710 command set, the execution of “MOV.vi” and “MVR.vi” is noticeably slower than that of the native firmware commands. Therefore this driver set provides the special non-GCS motion functions “NMOV.vi” and “NMVR.vi” for the case that your application requires quickest possible response to motion commands. See the VI reference of these two VIs for details.

Before using a joystick connected to the host PC, install joystick driver and calibrate joystick in the Windows Control Panel.

1.5.9. E-816

When controlling the E-816, timing problems can occur if several command VIs are run in rapid sequence, resulting in lost commands. To prevent such communication errors, it is recommended that you include a certain wait time between the different programming steps, depending on the command to be executed. This is especially true for commands that need a certain execution time inside the E-816 module, like MOV, MVR, SPA, SVA, SVR, RST, WPA, SWT and WTO. Only one axis per command can be controlled. “Split num query command.vi” can be used to query POS?, MOV?, VOL?, SVA? for multiple axes at a time.

1.5.10. F-206

This driver set (PI General LabVIEW Driver Set) and the F-206 LabVIEW driver set that also comes with the F-206 system are fully compatible and can be used in parallel. The F-206 can be fully controlled with the PI General LabVIEW Driver Set.The axis identifiers of the F-206 (X,Y,Z,U,V,W), NanoCube (K,L,M, if present) and additional axes (A,B, if any) cannot be changed.

Step 1 (advanced users can skip this step): To check communication between the F-206 controller and the host PC, run "F206_Simple_Test.vi". This VI will return the ID and help strings of the F-206 controller and the axis IDs and stage names of the connected axes (according to your selection of Is a NanoCube present? and How many additional axes are present?). If you have ordered the AC8 option, you can drive up to two additional separate, motor-driven axes (PWM-compatible motors with position control) with the F-206 controller (see also the F-206 User Manual). If you have ordered the NCU option, you can drive a 3-axis piezo stage ("NanoCube") with the F-206 controller. Before you proceed with step 2, please check that the current configuration matches your stage connections. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: F206_Configuration_Setup.vi May Cause Move

When you start "F206_Configuration_Setup.vi" with Initialize hexapod? = TRUE and/or Initialize additional axes? = TRUE, the VI will automatically move the Hexapod (and NanoCube, if present) and/or the additional axes to their sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move.

To control one or more F-206 controllers with this driver set, run "F206_Configuration_Setup.vi". This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close "F206_Configuration_Setup.vi"; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement "F206_Configuration_Setup.vi" as an initialization VI in your software. See chapter 3 for a detailed description of "F206_Configuration_Setup.vi" and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

1.5.11. M-840 / M-850

This driver set (PI General LabVIEW Driver Set) and the M-840 / M-850 LabVIEW driver set that comes with the M-840 / M-850 system are fully compatible and can be used in parallel. The M-840 / M-850 can be fully controlled with the PI General LabVIEW Driver Set and is called “M-8X0” from here on. The axis identifiers of the M-840 / M-850 and additional axes (if any) cannot be changed.

Step 1 (advanced users can skip this step): To check communication between the M-8X0 controller and the host PC, run “M8X0_Simple_Test.vi”. This VI will return the ID and help strings of the M-8X0 controller and the axis IDs and stage names of the connected axes (according to your selection of How many additional axes? are connected to the M-8X0 controller). If you have ordered the AC8 option, you can drive up to two additional separate, motor-driven axes (PWM-compatible motors with position control) with the M-8X0 controller (see also the M-8X0 User Manual). Before you proceed with step 2, please check that the current configuration matches your stage connections. See chapter 3 for a description of this VI and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Step 2:

WARNING: M8X0_Configuration_Setup.vi May Cause Move

When you start “M_8X0_Configuration_Setup.vi” with Initialize hexapod? = TRUE and/or Initialize additional axes? = TRUE, the VI will automatically move the Hexapod and/or the additional axes to their sensor switches. It is therefore important to make sure that items connected to or mounted on connected stages cannot be damaged by such a move.

To control one or more M-8X0 controllers with this driver set, run “M8X0_Configuration_Setup.vi”. This VI performs all steps necessary for a full configuration of the driver VIs in the LabVIEW environment: the definition of axis IDs, the initialization of the connected stages including referencing (if appropriate) and the definition of the controller name. During your testing phase (when you simply run the VIs without wiring them together into a program), do not close “M8X0_Configuration_Setup.vi”; otherwise all global settings will be lost and the driver VIs will not work. When programming your application, you can implement “M8X0_Configuration_Setup.vi” as an initialization VI in your software. See chapter 3 for a detailed description of “M8X0_Configuration_Setup.vi” and use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

2. Low Level VIs

The following low-level VIs can be found in the “Low Level” folder:

2.1. Communication VIs (“Communication.llb”):

2.1.1. Close connection if open.vi

Valid for	C-843, C-843.PM, C-844, C-865, E-710 (but must be present in Communication.llb for all other systems also)
Input	System number (1), Error in (no error)
Output	Was connected? (T/F), Error out
Remarks	This VI checks if the connection to the selected system is already open and, if it is, it closes this connection.

2.1.2. Find baudrate.vi

Valid for	C848, C-880, C-880K005, E-516, E-816, F-206, M-8X0
Input	System number (1), RS-232 Port number (0: COM1), Timeout (2000), Valid baudrates (array of 5 values), Flow control (All FALSE, x13, x11, x0), Error in (no error) C-848: Input and output HW handshake must be TRUE C-880: Input and output HW handshake must be TRUE C-880K005: Input and output HW handshake must be FALSE E-516: Input and output HW handshake must be TRUE E-816: Input and output HW handshake must be TRUE F-206: Input and output handshake settings must be FALSE M-8X0: Input and output handshake settings must be FALSE
Output	Baudrate out, Error out
Remarks	Opens COM port of given system with valid baudrates until status of Error out is false.

2.1.3. GCSTranslator DLL Functions.vi

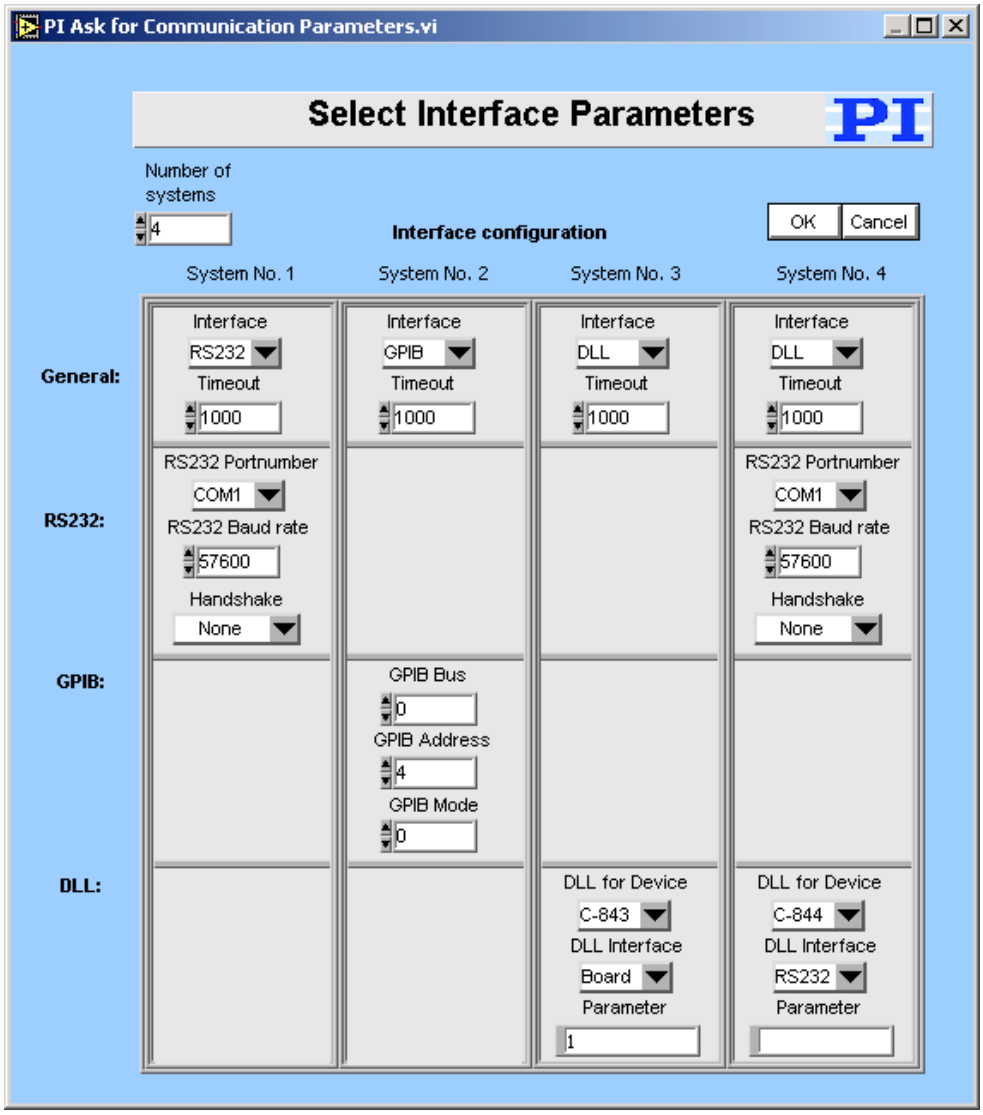
Valid for	C-843, C-843.PM, C-844, C-865, E-710 (but must be present in Communication.llb for all other systems also)
Input	System number (1), Function (C844_IsDLLAvailable), String buffer (empty string), String input (empty string), Error in (no error)
Output	DLL I32 Return value, Numerical output, Boolean output (T/F), String output, Error out
Remarks	This VI calls a given function from GCSTranslator.dll. GCSTranslator.dll must be installed. To call a system-specific function, the system-specific GCS DLL must be installed also. Warning: For XXX_GcsGetANswer , String buffer must be large enough, otherwise the application may crash. Call XXX_GcsGetANswerSize first to determine necessary string length.

2.1.4. Global1.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	None
Output	None
Remarks	A global variable which contains communication setup information.

2.1.5. PI Ask for Communication Parameters.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	None
Output	Number of systems, Cancel (T/F), Interface configuration, DLL interface configuration, Flow control
Remarks	A user-interface VI for setting up communications parameters (RS-232 or GPIB, number of systems, baudrate, timeout etc.).



2.1.6. PI Open Interface.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Number of systems (1), Interface configuration (RS232, 5000, COM1, 57600), DLL Interface configuration (C-843, Board, 1), Flow control (All FALSE, x13, x11, x0)
Output	Error out
Remarks	Establishes communication with up to four connected systems. The interface and error statuses of all connected systems are cleared by this VI, which sends XXX (no command), *IDN? and ERR?.

C-843: Interface = DLL, DLL for Device = C-843, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).

C-843.PM: Interface = DLL, DLL for Device = C-843.PM, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).

C-844: Interface = DLL, DLL for Device = C-844, DLL Interface = RS232 or GPIB, Parameter = empty string, RS232 baud rate = 9600

C-865: Interface = DLL, DLL for Device = C-865, DLL Interface = RS232, Parameter = empty string, RS232 baud rate = set as appropriate

C-880: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

C-848: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

C-880K005: Interface = RS232, Input and output HW handshake must be FALSE.

E-516: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

E-710: Interface = DLL, DLL for Device = E-710, DLL Interface = RS232 or GPIB, Parameter = empty string

E-816: Interface = RS232 (supports only RS-232 communication), Input and output HW handshake must be TRUE.

F-206: Interface = RS232 or GPIB, The error status will not be cleared by this VI. The first ERR? query will report a hidden error with error code 1, which will be cleared during system initialization (INI). RS232: Input and output handshake settings must be FALSE.

M-8X0: Interface = RS232 or GPIB. RS232: Input and output handshake settings must be FALSE.

2.1.7. PI Open Interface of one system.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System Number (1), Interface configuration (RS232, 5000, COM1, 57600), DLL Interface configuration (C-843, Board, 1), Flow control (All FALSE, x13, x11, x0)
Output	Error out

Remarks

Establishes communication with one connected system. **This VI is called automatically by "XXXX_Configuration_Setup.vi" (with XXXX being the PI product number of your system) and must be completed successfully before any other VI can use the interface.** The interface and error status of the chosen system are cleared by this VI, which sends XXX (no command), *IDN? and ERR?.

C-843: Interface = DLL, DLL for Device = C-843, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).

C-843.PM: Interface = DLL, DLL for Device = C-843.PM, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).

C-844: Interface = DLL, DLL for Device = C-844, DLL Interface = RS232 or GPIB, Parameter = empty string, RS232 baud rate = 9600

C-865: Interface = DLL, DLL for Device = C-865, DLL Interface = RS232, Parameter = empty string, RS232 baud rate = set as appropriate

C-880: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

C-848: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

C-880K005: Interface = RS232, Input and output HW handshake must be FALSE.

E-516: Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.

E-710: Interface = DLL, DLL for Device = E-710, DLL Interface = RS232 or GPIB, Parameter = empty string

E-816: Interface = RS232 (supports only RS-232 communication), Input and output HW handshake must be TRUE.

F-206: Interface = RS232 or GPIB, The error status will not be cleared by this VI. The first ERR? query will report a hidden error with error code 1, which will be cleared during system initialization (INI). RS232: Input and output handshake settings must be FALSE.

M-8X0: Interface = RS232 or GPIB. RS232: Input and output handshake settings must be FALSE.

2.1.8. PI Receive String.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Error in (no error)
Output	String read, Bytes read, Error out
Remarks	Read string from selected system.

2.1.9. PI ReceiveNCharacters RS232.vi

Valid for	C-848, C-880, C-880K005, E-516, E-816, F-206, M-8X0 (but must be present in Communication.llb for all other systems also)
Input	System number (1), Bytes to read (1), Error in (no error)
Output	String read, Bytes read, Error out
Remarks	Sub-vi for "PI Receive String" for RS 232 communication

2.1.10. PI ReceiveString GPIB.vi

Valid for	C-848, C-880, E-516, F-206, M-8X0 (but must be present in Communication.llb for all other systems also)
Input	System number (1), Error in (no error)
Output	String read, Bytes read, Error out
Remarks	Sub-vi for “PI Receive String” for GPIB communication.

2.1.11. PI Send String.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), String to send (empty string), Attach linefeed? (T), Error in (no error)
Output	Error out
Remarks	Sends command with or without trailing linefeed to selected system.

2.2. General Command VIs ("General command.llb"):

2.2.1. *IDN?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Error in (no error)
Output	Identification, Error out
Remarks	Returns system identification string.

2.2.2. Controller names.ctl

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	None
Output	None
Remarks	Type definition for control Controller names.

2.2.3. Define connected axes.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Read from controller?(F), Invert order?(F), Connected axes (empty string array), Error in (no error) C-843, C-843.PM: Read from controller = TRUE, Invert order = FALSE C-844: Read from controller = TRUE, Invert order = FALSE C-848: Read from controller = TRUE, Invert order = TRUE C-865: Read from controller = TRUE, Invert order = FALSE C-880: Read from controller = TRUE, Invert order = TRUE E-516: Read from controller = TRUE, Invert order = FALSE E-710: Read from controller = TRUE, Invert order = FALSE E-816: Read from controller = TRUE, Invert order = FALSE F-206: Read from controller = FALSE, Invert order = FALSE, Connected axes = X,Y,Z,U,V,W, (A,B,K,L,M optional) M-8X0: Read from controller = FALSE, Invert order = FALSE, Connected axes = X,Y,Z,U,V,W, (A,B optional)
Output	Connected axes out, Error out
Remarks	Writes connected axes into Global2.vi. **This VI is called automatically by "XXXX_Configuration_Setup.vi" (with XXXX being the PI product number of your system) and must be completed successfully before any other axis-specific command VI is called.** Requires "SAI?.vi" to be present.

2.2.4. Define connected systems.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	Controller names (cluster of 4 Enum controls, none), Change only one system? (F), System number (1), Error in (no error)

Output	Controller names out, Error out
Remarks	Defines connected systems and writes controller names into Global2.vi. **This VI is called automatically by "XXXX_Configuration_Setup.vi" (with XXXX being the PI product number of your system) and must be completed successfully before"General wait for movement to stop.vi" is called.** If Change only one system? is FALSE, all four entries from Controller names are written into "Global2.vi". If Change only one system? is TRUE, only the entry for the given system number is overwritten in "Global2.vi".

2.2.5. ERR?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Error in (no error)
Output	Hidden error (T/F), Error out
Remarks	Returns error information. Hidden error is TRUE if selected system reports error code ≠ 0. See appendix A for a list of PI error codes.

2.2.6. Global2.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Connected axes (empty string array), Connected systems (Cluster of 4 enum controls, none)
Output	None
Remarks	A global variable which contains identifiers for all connected axes of all connected systems and the names of all connected systems.

2.2.7. HLP?.vi

Valid for	C-843, C-843.PM, C-848, C-865, C-880, E-516, E-710
Input	System number (1), Error in (no error)
Output	Help string, Error out
Remarks	Returns help string.

2.2.8. HLT.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Affected axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? must be TRUE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Error out
Remarks	Stops motion of specified axes.

2.2.9. HPA?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Parameter help string, Error out
Remarks	Returns a help string containing information about valid parameter IDs.

2.2.10. MOV.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Axes to move (empty string array), Position values (empty num. array, 0), No. of digits (4), Error in (no error) C-880K005: VI only supported when called through PI_Multix.vi E-816: Only one axis per command allowed. It is necessary to wait a certain time before sending the next command to prevent it from being lost. F-206: No mix between F-206 axes X,Y,Z,U,V,W and separate axes A,B allowed
Output	Error out
Remarks	Moves specified axes to specified absolute positions. No. of digits is the number of digits after the decimal point in the position value(s) that will be sent. E-710: See also “NMOV.vi” in “Old commands.llb”.

2.2.11. MOV?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? can be FALSE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? can be FALSE E-516: If All axes? = TRUE, then Axis identifier? must be TRUE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE E-816: All axes? = FALSE, only one axis per command allowed. F-206: Command has different implementation, please use MOV?_old.vi M-8X0: Command has different implementation, please use MOV?_old.vi
Output	Target position, Error out
Remarks	Returns commanded target position.

2.2.12. MVR.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816
Input	System number (1), Axes to move (empty string array), Position values

(empty num. array, 0), No. of digits (4), Error in (no error)

C-880K005: VI only supported when called through PI_Multix.vi

E-816: Only one axis per command allowed. It is necessary to wait a certain time before sending the next command to prevent it from being lost.

Output: Error out

Remarks: Moves specified axes **relative** to current position. No. of digits is the number of digits after the decimal point in the position value(s) that will be sent.

E-710: See also "NMVR.vi" in "Old commands.llb".

2.2.13. ONT?.vi

Valid for: C-843, C-843.PM, C-848, C-865, C-880, E-516, E-816 (but must be present for all other systems also)

Input: System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes? = TRUE, then Axis identifier? must be TRUE

C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? must be TRUE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? must be TRUE

E-516: If All axes? = TRUE, then Axis identifier? can be FALSE.

E-816: All axes? = FALSE, only one axis per command allowed.

Output: Axis on target? (T/F), Error out

Remarks: Indicates whether or not queried axis is at target position.

2.2.14. POS?.vi

Valid for: C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0

Input: System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes?= TRUE, then Axis identifier? can be FALSE

C-843.PM: If All axes? = TRUE, then Axis identifier? can be FALSE

C-844: If All axes? = TRUE, then Axis identifier? can be FALSE

C-848: If All axes? = TRUE, then Axis identifier? can be FALSE

C-865: If All axes= TRUE, then Axis identifier? can be FALSE

C-880: If All axes? = TRUE, then Axis identifier? can be FALSE

C-880K005: VI only supported when called through PI_Multix.vi

E-516: If All axes? = TRUE, then Axis identifier? must be TRUE

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

E-816: All axes? = FALSE, only one axis per command allowed.

F-206: If All axes? = TRUE, then Axis identifier? must be TRUE

M-8X0: If All axes? = TRUE, then Axis identifier? must be TRUE

Output: Position, Error out

Remarks Returns position information (actual or target position, depending on system).

C-843: Returned position value is the actual position of the axis.

C-843.PM: Returned position value is the actual position of the axis.

C-844: Returned position value is the actual position of the axis.

C-848: Returned position value is the actual position of the axis.

C-865: Returned position value is the actual position of the axis.

C-880: Returned position value is the actual position of the axis.

E-516: Returned position value is the actual position of the axis.

E-710: Returned position value is the actual position of the axis.

E-816: Returned position value is the actual position of the axis.

F-206: Returned position value is the commanded target position for the axis.

M-8X0: Returned position value is the commanded target position for the axis.

2.2.15. SAI?.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816 (but must be present in “General command.llb” for all other systems also)

Input System number (1), Invert order? (F), SAI? ALL (F), Error in (no error)

C-843: Invert order should be FALSE, SAI? ALL is supported

C-843.PM: Invert order should be FALSE, SAI? ALL must be FALSE

C-844: Invert order should be FALSE, SAI? ALL must be FALSE

C-848: Invert order should be TRUE, SAI? ALL must be FALSE

C-865: Invert order should be FALSE, SAI? ALL is supported

C-880: Invert order should be TRUE, SAI? ALL must be FALSE to read all configured axis IDs and must be TRUE to get all physically defined axis IDs

C-880K005: VI only supported when called through PI_Multix.vi, SAI? ALL must be FALSE

E-516: Invert order should be FALSE, SAI? ALL must be FALSE

E-710: Invert order should be FALSE, SAI? ALL is supported

E-816: Invert order should be FALSE, SAI? ALL must be FALSE

Output Connected axes, Error out

Remarks Returns axis identifiers of all connected axes and writes them into Global2.vi. If SAI? ALL is TRUE, all physically available axes are returned, no matter if configured or not. Required by “Define connected axes.vi”

2.2.16. SPA?_Hex.vi

Valid for E-710

Input System number (1), Designator to query (empty string array), Parameter no. (hex) (empty num. array, 0), Without designators? (F), Error in (no error)

E-710: Use “HPA?.vi” to get valid parameter numbers and see description of SPA_Hex command for a description of valid parameter numbers.

Output Parameter value, Error out

Remarks Returns parameter values for queried designators and hex. parameter numbers. For axis-related parameters, Designator to query is the axis name; for piezo- or sensor-related parameters, the channel number; otherwise a parameter-related code. See GCS DLL Manual for available parameter numbers. If Without designators? is TRUE, all available parameter for all designators are returned. For parameter numbers which output a string, use “SPA?_Hex_String.vi”.

2.2.17. SPA?_Hex_String.vi

Valid for E-710

Input System number (1), Designator to query (empty string array), Parameter no. (hex) (empty num. array, 0), Without designators? (F), Error in (no error)

Output Parameter string, Error out

Remarks Returns parameter strings for queried axes and hex. parameter numbers. See “SPA_Hex_String.vi” for available parameter numbers. For axis-related parameters, Designator to query is the axis name; for piezo- or sensor-related parameters, the channel number; otherwise a parameter-related code. If Without designators? is TRUE, all available parameter for all designators are returned. For parameter numbers which require a numerical value as input, use “SPA?_Hex.vi”.

E-710: Use “HPA?.vi” to get valid parameter numbers

2.2.18. SPA_Hex.vi

Valid for E-710

Input System number (1), Designator to set (empty string array), Parameter value (hex) (empty num. array, 0), No. of digits (4), Parameter number (empty num. array, 0), Error in (no error)

E-710:

Please refer to the stage manual for valid parameter settings.

The following parameter numbers are valid:

- 0X02000000: Sensor Mechanic: Sensor/Analog enable 0 = Disabled 1 = Enabled
- 0X02000100: Sensor Mechanic: Sensor range factor
  0 = Board Range 3.00X 1 = Option 3.00X 21
  2 = Option 3.00X 31 3 = Option 3.00X 41
  4 = Option 3.00X 51 5 = Option 3.00X 61
  6 = Option 3.00X 71 7 = Board Range 2.13X
  8 = Option 2.13X 32 9 = Option 2.13X 42
  10 = Option 2.13X 52 11 = Option 2.13X 62
  12 = Option 2.13X 72 13 = Board Range 1.25X
  14 = Option 1.25X 43 15 = Option 1.25X 53
  16 = Option 1.25X 63 17 = Option 1.25X 73
  18 = Board Range 1.00X 19 = Option 1.00X 54
  20 = Option 1.00X 64 21 = Option 1.00X 74
  22 = Board Range 0.75X 23 = Option 0.75X 65
  24 = Option 0.75X 75 25 = Board Range 0.68X
  26 = Option 0.68X 76 27 = Board Range 0.56X
- 0X02000101: Sensor Mechanic: Board Gain 0 = Gain 0.5
  64 = Gain 1.0 128 = Gain 2.0 192 = Gain 3.0
- 0X02000102: Sensor Mechanic: Electrical poti selected

- 0X02000200: Sensor Mechanic: Sensor correction 0 order
- 0X02000300: Sensor Mechanic: Sensor correction 1st order
- 0X02000400: Sensor Mechanic: Sensor correction 2nd order
- 0X02000500: Sensor Mechanic: Sensor correction 3rd order
- 0X02000600: Sensor Mechanic: Sensor correction 4th order
- 0X05000000: Sensor Filter: Digital filter type 0 = No Filter 1 = IIR Filter 2 = FIR filter
- 0X05000001: Sensor Filter: Digital filter Bandwidth/Hz
- 0X05000002: Sensor Filter: Digital filter order
- 0X05000101: Sensor Filter: User filter parameter A0
- 0X05000102: Sensor Filter: User filter parameter A1
- 0X05000103: Sensor Filter: User filter parameter B0
- 0X05000104: Sensor Filter: User filter parameter B1
- 0X05000105: Sensor Filter: User filter parameter B2
- 0X07000000: Servo: Range min limit (µ)
- 0X07000001: Servo: Range max limit (µ)
- 0X07000200: Servo: Servo loop slew rate (axis unit/ms)
- 0X07000300: Servo: Servo loop P-Term
- 0X07000301: Servo: Servo loop I-Term
- 0X07000500: Servo: Position from sensor 1
- 0X07000501: Servo: Position from sensor 2
- 0X07000502: Servo: Position from sensor 3
- 0X07000503: Servo: Position from sensor 4
- 0X07000504: Servo: Position from sensor 5
- 0X07000505: Servo: Position from sensor 6
- 0X07000506: Servo: Position from sensor 7
- 0X07000507: Servo: Position from sensor 8
- 0X07000800: Servo: sensor ON/OFF start up 0 = Disabled 1 = Enabled
- 0X07000801: Servo: Servo enable 0 = Disabled 1 = Enabled
- 0X07000802: Servo: Auto-Zero start up 0 = Disabled 1 = Enabled
- 0X07000900: Servo: Tolerance
- 0X07000A00: Servo: Auto-Zero driving low voltage (V)
- 0X07000A01: Servo: Auto-Zero driving high voltage (V)
- 0X07000B00: Servo: Zoom Auto-Zero low voltage (V)
- 0X07000B01: Servo: Zoom Auto-Zero high voltage (V)
- 0X07000C00: Servo: Default position
- 0X07000C01: Servo: Default voltage
- 0X07010100: Servo: Syncronous
- 0X08000100: Servo output filter: Notch frequency of filter nr. 1
- 0X08000101: Servo output filter: Notch frequency of filter nr. 2
- 0X08000200: Servo output filter: Notch rejection of filter nr. 1
- 0X08000201: Servo output filter: Notch rejection of filter nr. 2
- 0X08000300: Servo output filter: Notch bandwidth of filter nr. 1
- 0X08000301: Servo output filter: Notch bandwidth of filter nr. 2
- 0X08000400: Servo output filter: Creep factor T1 (sec)
- 0X08000401: Servo output filter: Creep factor T2 (sec)
- 0X09000000: Output Matrix: Driving with piezo 1
- 0X09000001: Output Matrix: Driving with piezo 2
- 0X09000002: Output Matrix: Driving with piezo 3
- 0X09000003: Output Matrix: Driving with piezo 4
- 0X09000004: Output Matrix: Driving with piezo 5
- 0X09000005: Output Matrix: Driving with piezo 6
- 0X09000006: Output Matrix: Driving with piezo 7
- 0X09000007: Output Matrix: Driving with piezo 8
- 0X0C000000: Piezo: Output voltage low limit (V)
- 0X0C000001: Piezo: Output voltage high limit (V)
- 0X0E000100: System Global: Sensor sampling time

- 0X0E000200: System Global: Servo update time
- 0X0E000300: System Global: Triger-in enable (D_IN/Bit)
- 0X0E000400: System Global: DDL-License
- 0X0E000500: System Global: Last firmware version
- 0X0E000800: System Global: Trigger: -1 = Active High, 1 = Active Low
- 0X0E000900: System Global: Trigger pulse length: 1 = Short Pulse, 2 = Long Pulse
- 0X0F000100: System Mechanic: Stage type
- 0X0F000200: System Mechanic: Serial number
- 0X10000000: Fast Interface: DSP-Link input low limit
- 0X10000001: Fast Interface: DSP-Link input high limit
- 0X10000200: Fast Interface: PIO input low limit
- 0X10000201: Fast Interface: PIO input high limit
- 0X10000300: Fast Interface: Interface PIO control 0 = Disabled 1 = Enabled
- 0X10000303: Fast Interface: PIO on target line 4 function 0 = Trigger 8 = Axis 4 On-Target signal
- 0X12000000: Interface Parser: GPIB address
- 0X12000100: Interface Parser: Serial port default baud rate
- 0X13000000: Wave Generator: Generator running control 0 = All Disabled 16 = Single Run Enabled 48 = SYNC Run Enabled 80 = Run/DDL Enabled 240 = All Enabled
- 0X13000001: Wave Generator: Installed wave form
- 0X13000002: Wave Generator: Connected axis
- 0X13000003: Wave Generator: Wave generator cycles
- 0X13000101: Wave Generator: Curve shape 0 = No Function 16 = Sine Wave 32 = Ramp Wave 64 = Single Line
- 0X13000102: Wave Generator: Total wave form points
- 0X13000103: Wave Generator: Curve points
- 0X13000104: Wave Generator: Center point
- 0X13000105: Wave Generator: Speed-up/Slow-down points
- 0X13000106: Wave Generator: Curve start point
- 0X13000107: Wave Generator: Curve amplitude
- 0X13000108: Wave Generator: Curve offset
- 0X13000109: Wave Generator: Wave generator table rate
- 0X14000000: DDL: DDL table number
- 0X14000001: DDL: DDL repeat number
- 0X14000002: DDL: DDL gain constant
- 0X14000003: DDL: DDL gain change
- 0X14000004: DDL: DDL gain curve
- 0X14000005: DDL: Final DDL Gain
- 0X14000006: DDL: Final delay max (s)
- 0X14000007: DDL: Final delay min (s)
- 0X14000008: DDL: Time delay change rule
- 0X14000009: DDL: Final time delay
- 0X1400000A: DDL: DDL zero gain number
- 0X14000200: DDL: DDL report filter 0 = Disabled 1 = Enabled
- 0X14000201: DDL: DDL report filter band width factor

Output Hidden error (T/F), Error out

Remarks Sets parameters for commanded designators, waits 100 ms and queries ERR?. No. of digits is the number of digits after the decimal point in the parameter value(s) that will be sent. For axis-related parameters, Designator to set is the axis name; for piezo- or sensor-related parameters, the channel number; otherwise a parameter-related code. Hidden error is

TRUE if selected system reports error code ≠ 0. For parameter numbers which require a string as input use “SPA_Hex_String.vi”.

2.2.19. SPA_Hex_String.vi

Valid for E-710

Input System number (1), Designator to set (empty string array), Parameter number (hex) (empty num. array, 0), Parameter string (empty string array), Error in (no error)

E-710: the following parameter numbers are valid:

- 0X07000600: Servo: Axis name
- 0X07000601: Servo: Axis unit

Output Hidden error (T/F), Error out

Remarks Sets string parameters for commanded designators, waits 100 ms and queries ERR?. For axis-related parameters, Designator to set is the axis name; for piezo- or sensor-related parameters, the channel number; otherwise a parameter-related code. Hidden error is TRUE if selected system reports error code ≠ 0. For parameter numbers which require a numerical value as input, use “SPA_Hex.vi”.

2.2.20. STP.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516 (but must be present for E-710 also)

Input System number (1), Affected axes? (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: All axes? = TRUE, Axis identifier? = FALSE

C-843.PM: All axes? = TRUE, Axis identifier? = FALSE

C-844: All axes? = TRUE, Axis identifier? = FALSE

C-848: All axes? = TRUE, Axis identifier? = FALSE

C-865: All axes? = TRUE, Axis identifier? = FALSE

C-880: All axes? = TRUE, Axis identifier? = FALSE

E-516: If All axes? = TRUE, then Axis identifier? must be TRUE

Output Error out

Remarks Stops motion of specified axes. To stop a referencing routine (REF, MNL, MPL) or fast scan routine (FSC, FSA etc.), use “#24.vi”.

2.2.21. SVO.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0

Input System number (1), Without axis ID?(F), Axes to command (empty string array), Servo mode (empty bool. array, F), Error in (no error)

C-843: Without axis ID = FALSE

C-843.PM: Without axis ID = FALSE

C-844: Without axis ID = FALSE

C-848: Without axis ID = FALSE

C-865: Without axis ID = FALSE

C-880: Without axis ID = FALSE

E-516: Without axis ID = FALSE

E-710: Without axis ID = FALSE

E-816: Without axis ID = FALSE. Only one axis per command allowed.

F-206: Without axis ID = TRUE, only first field of Servo mode array is valid

M-8X0: Without axis ID = TRUE, only first field of Servo mode array is valid

Output Error out

Remarks Sets servo-control mode for given axes. If Without axis ID is TRUE, then Axes to command is ignored and first field of Servo mode array is used.

2.2.22. SVO?.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0

Input System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes? = TRUE, then Axis identifier? must be TRUE

C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE

C-844: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? can be FALSE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? can be FALSE

E-516: If All axes? = TRUE, then Axis identifier? must be TRUE

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

E-816: All axes? = FALSE, only one axis per command allowed.

F-206: All axes? = TRUE, Axis identifier? = FALSE

M-8X0: All axes? = TRUE, Axis identifier? = FALSE

Output Servo status (T/F), Error out

F-206: Only first field of servo status array is valid

M-8X0: Only first field of servo status array is valid

Remarks Returns servo status of queried axes.

2.2.23. VEL.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, F-206, M-8X0

Input System number (1), Without axis ID? (F), No. of digits (4), Axes to set (empty string array), Velocity values (empty num. array, 0), Error in (no error)

C-843: Without axis ID? = FALSE

C-843.PM: Without axis ID? = FALSE

C-844: Without axis ID? = FALSE

C-848: Without axis ID? = FALSE

C-865: Without axis ID? = FALSE

C-880: Without axis ID? = FALSE, for NanoCube axes command is not valid

C-880K005: VI only supported when called through PI_Multix.vi

E-516: Without axis ID? = FALSE

E-710: Without axis ID? = FALSE. Velocity unit is µm/ms.

F-206: F-206 platform velocity: Without axis ID? = TRUE; velocity of axes A and/or B: Without axis ID? = False; axes K,L,M: command not valid

M-8X0: M-8X0 platform velocity: Without axis ID? = TRUE; velocity of axes A and/or B: Without axis ID? = False

Output Error out, Hidden error

Remarks Sets velocity and checks for error. If Without axis ID? is TRUE, then Axes to set is ignored and first field of Velocity values array is used for velocity command. The velocity should not be set to 0. Number of digits is the number of digits after the decimal point in the velocity value(s) that will be sent. Hidden error is TRUE if selected system reports error code ≠ 0.

E-516: The VEL command saves the parameters in RAM only. To save the currently valid parameters to flash ROM, where they become the power-on defaults, you must run WPA.vi. Parameter changes not saved with WPA will be lost when the E-516 is powered off.

2.2.24. VEL?.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, F-206, M-8X0

Input System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes? = TRUE, then Axis identifier? must be TRUE

C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE

C-844: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? can be FALSE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? can be FALSE

C-880K005: VI only supported when called through PI_Multix.vi

E-516: If All axes? = TRUE, then Axis identifier? must be TRUE

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE. Velocity unit is µm/ms.

F-206: Velocity of F-206: All axes? = TRUE AND Axis identifier? = FALSE; velocity of axes A,B: All axes? must be FALSE; axes K,L,M: command not valid

M-8X0: Velocity of M-8X0: All axes? = TRUE AND Axis identifier? = FALSE; velocity of axes A,B: All axes? must be FALSE

Output Velocity, Error out

C-880: NanoCube axes will report velocity = 0

F-206: F-206 velocity: only first field of velocity array is valid

M-8X0: M-8X0 velocity: only first field of velocity array is valid

Remarks Returns velocity setting for specified axes.

2.3. PZT specific VIs ("PZT voltage.llb")

2.3.1. DPO.vi

Valid for	E-710
Input	System number (1), DPO axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Error out
Remarks	DDL processing parameter correction for specified axis. E-710: Command is available in command level 1 only (see "CCL.vi", "CCL?.vi")

2.3.2. DTC.vi

Valid for	E-710
Input	System number (1), DTC table (empty string array), All tables? (F), Error in (no error) E-710: All tables? must be FALSE.
Output	Error out
Remarks	Clears DDL table.

2.3.3. SVA.vi

Valid for	E-516, E-710, E-816
Input	System number (1), Axes to move (empty string array), PZT voltage (empty num. array, 0), No. of digits (4), Error in (no error) E-816: Only one axis per command allowed. It is necessary to wait a certain time before sending the next command to prevent it from being lost.
Output	Error out
Remarks	Sets absolute PZT voltage for specified axes. No. of digits is the number of digits after the decimal point in the voltage value(s) that will be sent.

2.3.4. SVA?.vi

Valid for	E-516, E-710, E-816
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) E-516: If All axes= TRUE, then Axis identifier? must be TRUE E-710: If All axes= TRUE, then Axis identifier? must be TRUE E-816: All axes? = FALSE, only one axis per command allowed.
Output	Commanded PZT voltage, Error out
Remarks	Returns commanded PZT voltage for queried axes.

2.3.5. SVR.vi

Valid for	E-516, E-710, E-816
Input	System number (1), Axes to move (empty string array), PZT voltage (empty num. array, 0), No. of digits (4), Error in (no error)

E-816: Only one axis per command allowed. It is necessary to wait a certain time before sending the next command to prevent it from being lost.

Output Error out

Remarks Sets relative PZT voltage for specified axes. No. of digits is the number of digits after the decimal point in the voltage value(s) that will be sent.

2.3.6. VOL?.vi

Valid for E-516, E-710, E-816

Input System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

E-516: If All axes? = TRUE, then Axis identifier? must be TRUE

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

E-816: All axes? = FALSE, only one axis per command allowed.

Output Current PZT voltage, Error out

Remarks Returns current PZT voltage for queried axes.

E-710: Axes to query are piezo channel numbers.

2.4. Special commands (“Special command.llb”)

2.4.1. #24.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, F-206, M-8X0 (but must be present for E-710 also)
Input	System number (1), Error in (no error) C-880K005: VI only supported when called through PI_Multix.vi
Output	Error out
Remarks	Stops motion by sending the single ASCII character 24.

2.4.2. #5.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880 (but must be present for all other systems also)
Input	System number (1), Error in (no error)
Output	Axis moving? (T/F), Error out
Remarks	Polls the motion status of the connected axes by sending the single ASCII character 5. Connected axes are read from Global2.vi and displayed on the front panel for assignment. Required by “General wait for movement to stop.vi” and “Wait for axes to stop.vi”. F-206: Different coding in answer, please use #5_old.vi M-8X0: Different coding in answer, please use #5_old.vi

2.4.3. #7.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710 (but must be present for E-516, F-206, M-8X0 also)
Input	System number (1), Error in (no error) C-880K005: VI only supported when called through PI_Multix.vi
Output	Ready? (T/F), String read, Error out
Remarks	Sends the single ASCII character 7 and returns the ready status of the controller. Sub-VI for “Wait for answer of longlasting command.vi”.

2.4.4. CCL.vi

Valid for	E-710
Input	System number (1), Password (100), Command level (0), Error in (no error) E-710: Command level can be 0 (only commands needed for normal operation are available) or 1 (all commands from command level 0 plus special commands for advanced users are available). Password for CCL 1 is “ADVANCED”.
Output	Error out, Hidden error
Remarks	If password is correct, this vi sets the command level of the controller and queries ERR?. Hidden error is TRUE if selected system reports error code ≠ 0. User “HLP?.vi” to determine which commands are available in the current command level.

2.4.5. CCL?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Command level, Error out
Remarks	Returns the current command level. See “CCL.vi” for further information.

2.4.6. CST.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Axis ID’s (empty string array), Stage names (empty string array), Error in (no error)
Output	Error out
Remarks	Assigns axes to stages and queries “ERR?”. With this command the stage assignment of the connected axes can be changed. Valid stage names can be listed with VST?.vi.

2.4.7. CST?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710, F-206, M-8X0
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? must be TRUE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? must be TRUE E-710: If All axes? = TRUE, then Axis identifier? can be FALSE F-206: If All axes? = TRUE, then Axis identifier? must be TRUE M-8X0: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Stage names, Error out
Remarks	Returns the name of the connected stage for queried axes.

2.4.8. DFF.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Axis (empty string array), Factor (empty num. array, 0), No. of digits (4), Error in (no error)
Output	Error out
Remarks	Defines scale factor which is applied to the basic unit (default is 1). E.g. 25.4 changes the physical unit from mm to inches. No. of digits is the number of digits after the decimal point in the factor value(s) that will be sent. *Example: The physical unit is mm and the scale factor is 1. The current position of a stage is 12. Now the scale factor is set to 3 with DFF. Reading the position gives 4 as result. A relative move of 1.5 causes the stage to move 4.5 mm.*

C-843: Factor can only be positive.

C-843.PM: Factor can only be positive.

C-848: Factor can only be positive.

C-865: Factor can only be positive.

C-880: Factor can only be positive.

E-710: Factor can only be positive.

2.4.9. DFF?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? can be FALSE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? can be FALSE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Factor, Error out
Remarks	Returns constant unit value for specified axes (e.g. 25.4 for inches).

2.4.10. DRC.vi

Valid for	E-710
Input	System number (1), Rec. table (0), Source ID (empty string), Rec. option (0), Trigger option (0), Error in (no error)
Output	Hidden error (T/F), Error out
Remarks	This VI configures the data recording, waits 100 ms and queries ERR?. See GCS DLL manual for available recording and trigger options. Hidden error is TRUE if selected system reports error code ≠ 0.

2.4.11. DRC?.vi

Valid for	E-710
Input	System number (1), Rec. table to query (empty num. array, 0), Error in (no error)
Output	Source ID (empty string array), Rec. option (empty num. array, 0), Error out
Remarks	This VI returns the data recording configuration (Source ID and Rec. option) for the queried record table.

2.4.12. DRR?.vi

Valid for	E-710
Input	System number (1), Rec. table IDs (Empty num. array, 0), xo (0), N (100), Nmax (1024), Without parameter? (FALSE), Error in (no error) E-710: Nmax = 32256.

Output	Data, Names, Sample time, Error out
Remarks	Returns N recorded data points. N must be less than or equal to Nmax. For large N values, communication timeout must be set long enough, otherwise a communication error may occur.

2.4.13. IMP.vi

Valid for	E-710
Input	System number (1), Axis to command (empty string), Impulse size (0), Delay (0), No. of digits (4), Error in (no error) E-710: Delay = 0.
Output	Error out
Remarks	Performs a single impulse-move (two equal moves in opposite directions in quick succession) from the current position with specified Impulse size (amplitude), and records fixed number of actual positions at specified intervals thereafter. If supported, Delay sets the number of servo loops between each position recording. No. of digits is the number of digits after the decimal point in the impulse amplitude values that will be sent. Controller saves a fixed number of position values, which can be read out with IMP?.vi. E-710: Controller saves 32,256 position values. Width of impulse and sampling interval taken from "Table Rate" parameter, set with "SPA_Hex.vi". Caution: "Table Rate" parameter influences Wave Generator also, not only IMP. Typically, IMP is used in open loop mode.For a single step-move, see "STE.vi".

2.4.14. IMP?.vi

Valid for	E-710
Input	System number (1), Query axis (empty string), xo (0), N (100), Nmax (1024), Error in (no error) E-710: Nmax = 32256.
Output	Impulse response, Error out
Remarks	Returns N saved impulse response points. N must be less than or equal to Nmax. For large N values, communication timeout must be set long enough, otherwise a communication error may occur.

2.4.15. INI.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710, F-206, M-8X0
Input	System number (1), INI axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? must be TRUE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? must be TRUE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

F-206: Initialize F-206: All axes? = TRUE, Axis identifier? = FALSE (separate axes A,B (if present) will not be initialized); initialize axes A,B: All axes? = FALSE, axes K,L,M: command not valid (NanoCube is initialized with F-206); VI will not wait for INI procedure to finish

M-8X0: Initialize M-8X0: All axes? = TRUE, Axis identifier? = FALSE, (separate axes A,B (if present) will not be initialized); initialize axes A,B: All axes? = FALSE; VI will not wait for INI procedure to finish. Use "INI hexaxes and wait until finished.vi" to initialize hexapod and/or separate axes and wait for the procedure to finish.

Output Error out

Remarks Initializes axes. System-specific: see individual GCS-DLL Manual for details.

C-844: It is necessary to wait a certain time – appr. 4 s – before sending the next command to prevent it from being lost.

2.4.16. RPA.vi

Valid for E-710

Input System number (1), Affected axes (empty string array), Parameter to restore (empty num. array), Error in (no error)

E-710: If Affected axes = empty array, all parameters for all axes are restored. See E-710 GCS DLL Manual for a list of available parameters.

Output Error out, Hidden error

Remarks Replaces the current values of Parameter to restore for Affected axes in the controller RAM with the values from non-volatile memory , waits 3000 ms and queries ERR?. If Affected axes is an empty array, RPA is sent without axis and parameter specification and controller restores all values for all axes. Hidden error is TRUE if selected system reports error code ≠ 0.

2.4.17. SAI.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710

Input System number (1), Old axis ID (empty string array), New axis ID (empty string array), Error in (no error)

Output Error out

Remarks Set axis identifier. With this command the axis identifiers of the connected axes can be changed. Only one character is allowed as axis ID. Valid axis IDs can be listed with TVI?.vi. Please run "Define connected axes.vi" with new axis IDs after renaming axes.

2.4.18. SEP?_Hex.vi

Valid for E-710

Input System number (1), Axes to query (empty string array), Parameter no. (hex) (empty num. array, 0), Without axes? (F), Error in (no error)

E-710: See E-710 GCS DLL Manual for a detailed list of available parameter numbers. Without axes? can be TRUE and will return all available parameter for all axes.

Output Parameter string, Error out

Remarks Returns parameter string from non-volatile memory for queried axis and hex. parameter number.

2.4.19. SEP_Hex.vi

Valid for E-710

Input System number (1), Password (empty string), Axis to set (empty string array), Parameter no. (hex) (empty num. array, 0), Parameter string (empty string array), Error in (no error)

E-710: See E-710 GCS DLL Manual for a detailed list of available parameter numbers

Output Hidden error (T/F), Error out

Remarks If password is correct, writes specified parameter values for specified hex. parameters of specified axes/channels to Eprom, waits 100 ms and queries ERR?. Hidden error is TRUE if selected system reports error code ≠ 0.

E-710: Command is available in command level 1 only (see “CCL.vi”, “CCL?.vi”)

2.4.20. STA?.vi

Valid for C-848, C-880, C-880K005 (but must be present in Special command.llb for all other systems also)

Input System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-848: If All axes? = TRUE, then Axis identifier? can be FALSE

C-880: If All axes? = TRUE, then Axis identifier? can be FALSE

C-880K005: VI only supported when called through PI_Multix.vi

Output Axis status, Error out

C-848, C-880:

The status word for each axis is a 16-bit register containing the following information (bit encoding is 0 = LSB, 15 = MSB):

Bit #	Description
0	Motion complete flag. This bit is set (1) when the axis trajectory has completed. This flag is only valid for the S-curve, trapezoidal, and velocity contouring profile modes.
1	Wrap-around condition flag. This bit is set (1) when the axis has reached one end of its travel range and has wrapped to the other end of the travel range. Specifically, when traveling in a positive direction past the position +1,073,741,823, the axis will wrap to position -1,073,741,824, and vice-versa. The bit can be reset with the CLR command.
2	Breakpoint reached flag. This bit is set (1) when one of the breakpoint conditions has occurred.
3	Index pulse received flag. This bit is set (1) when an index pulse has been received.
4	Motion error flag. This bit is set (1) when the maximum position error is exceeded. This bit can only be reset when the axis is no longer in a motion error condition
5	Positive limit switch flag. This bit is set (1) when the positive limit switch goes active.
6	Negative limit switch flag. This bit is set (1) when the negative limit switch goes active.

7 Command error flag. This bit is set (1) when an erroneous command has been received by the motion control chip.

8* Servo-control on/off status (1 indicates on, 0 indicates off).

9* Axis on/off status (1 indicates on, 0 indicates off). The C-848 always has the axis ON.

10* In-motion flag. This bit is continuously updated and indicates whether or not the axis is in motion: 1 indicates axis is in motion, 0 not in motion.

11* Reserved (may contain 0 or 1)

12*,13* Current axis # (13 bit = high bit, 12 bit = low bit). Axis encoding is as follows:

Bit 13	Bit12	MC Axis	C-848 Axis
0	0	1	A
0	1	2	B
1	0	3	C
1	1	4	D

14,15 Reserved (may contain 0 or 1)

C-880K005:

The status word for each axis is a 16-bit register containing the following information (bit encoding is 0 = LSB, 15 = MSB):

Bit # Description

0 Motion complete flag. Set to 1 when motion is completed. SetMotionCompleteMode determines if this bit is based on the trajectory generator position or the encoder position.

1 Wrap-around condition flag. This bit is set (1) when the actual (encoder) position wraps from maximum allowed position to minimum or vice versa.

2 Breakpoint 1 reached flag. This bit is set (1) when breakpoint 1 is triggered.

3 Capture received flag. This bit is set (1) when a position caputre occures.

4 Motion error flag. This bit is set (1) when a motion error occurs

5 Positive limit switch flag. This bit is set (1) when the positive limit switch goes active.

6 Negative limit switch flag. This bit is set (1) when the negative limit switch goes active.

7 Instruction error flag. This bit is set (1) when an instruction error occurs.

8-10 Reserved, may be 0 or 1.

11 Commutation error flag. This bit is set (1) when a commutation error occurs.

12-13 Reserved, may be 0 or 1.

14 Breakpoint 2 reached flag. This bit is set (1) when breakpoint 2 is triggered.

15 Reserved, may be 0 or 1.

Remarks Returns axis status (integer). Required by “General wait for movement to stop.vi” and “Wait for axes to stop.vi”.

2.4.21. STE.vi

Valid for	C-843, C-843.PM, C-848, C-865, C-880, E-710
Input	System number (1), Axis to command (empty string), Step size (0), Delay (0), No. of digits (4), Error in (no error) C-843: Delay = 0. C-843.PM: Delay = 0. C-865: Delay = 0. E-710: Delay = 0.
Output	Error out
Remarks	Performs a step-move from, and back to, the current position with specified step size (amplitude). If supported, Delay sets the number of servo loops between position recording. No. of digits is the number of digits after the decimal point in the step size (amplitude) values that will be sent. Controller saves a definite number of position values which can be read out with STE?.vi. Use “General wait for movement to stop.vi” before calling “STE?.vi” to make sure that motion has finished before reading back the saved values. For an impulse-move, see “IMP.vi”. C-843: Controller saves up to 32,640 position values for all 4 channels in sum. C-843.PM: Controller saves up to 32,640 position values for all 4 channels in sum. C-848: Controller saves 1024 position values. C-865: Controller saves up to 32,640 position values. C-880: Controller saves 1024 position values. E-710: Controller saves 8192 position values. “Table Rate” parameter, set with SPA, is used as sampling interval instead of Delay. Caution: Table Rate parameter influences Wave Generator, not only IMP.

2.4.22. STE?.vi

Valid for	C-843, C-843.PM, C-848, C-865, C-880, E-710
Input	System number (1), Query axis (empty string), xo (0), N (100), Nmax (1024), Error in (no error) C-843: Nmax = 32640. C-843.PM: Nmax = 32640. C-848: Nmax = 1024. C-865: Nmax = 32640 C-880: Nmax = 1024. E-710: Nmax = 8192.
Output	Step response, Error out
Remarks	Returns N saved step response points. N must be less than or equal to Nmax. For large N values, communication timeout must be set long enough, otherwise a communication error may occur.

2.4.23. TNR?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Number of Rec. tables, Error out
Remarks	Returns the number of recording tables.

2.4.24. TPC?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Number of piezo channels, Error out
Remarks	Returns the number of available piezo channels.

2.4.25. TVI?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Invert order, Error in (no error) C-843: Invert order must be FALSE C-843.PM: Invert order must be FALSE C-844: Invert order must be FALSE C-848: Invert order should be TRUE C-865: Invert order must be FALSE C-880: Invert order should be TRUE E-710: Invert order must be FALSE
Output	Valid axis IDs, Error out
Remarks	Get valid axis identifiers. Should be called before axes are renamed with SAI.vi.

2.4.26. VST?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Error in (no error)
Output	Available stages, Error out
Remarks	Returns the names of available stages connected to the controller.

2.4.27. WPA.vi

Valid for	E-516, E-710, E-816
Input	System number (1), Password (100), Affected axes (empty string array), Parameter to save (empty num. array), Error in (no error) E-516: Affected axes and Parameter to save = empty array E-710: If Affected axes = empty array, all parameters for all axes are saved. See E-710 GCS DLL Manual for a list of available parameters. E-816: Affected axes and Parameter to save = empty array
Output	Error out, Hidden error

Remarks If password is correct, this vi writes current settings of Parameter to save for Affected axes to non-volatile memory of the controller, waits 3000 ms and queries ERR?. If "Affected axes" is an empty array, WPA is sent without axis and parameter specification. Hidden error is TRUE if selected system reports error code ≠ 0.

E-516: The WPA command saves the currently valid parameters listed below to flash ROM, where they become the power-on defaults. Parameter changes not saved with WPA will be lost when the E-516 is powered off. If current parameter values are incorrect, the system may malfunction. Be sure that you have the correct parameter settings before using the WPA command. Communication interface, enabled channels and display format, averaging (AVG), drift compensation mode (DCO), velocity control mode (VCO) and velocity (VEL), offset and gain for position and output voltage display, mode and tolerance for on-target reading (SPA), position limits (NLM, PLM), voltage limits (VMA, VMI), macros and default macro setting.

E-710: Command is available in command level 1 only (see “CCL.vi”, “CCL?.vi”)

2.5. Old commands and commands with alternate implementations (“Old commands.llb”)

2.5.1. #5_old.vi

Valid for F-206, M-8X0 (but must be present for all other systems also)

Input System number (1), Error in (no error)

Output Overall system moving? (T/F), Sep. Axis 1 moving? (T/F), Sep. Axis 2 moving? (T/F), Error out

Remarks Polls the motion status of the F-206/M-8X0 and/or up to 2 additional connected axes by sending the single ASCII character 5. Required by “General wait for movement to stop.vi”.

2.5.2. NMOV.vi

Valid for E-710

Input System number (1), Axes to move (empty string array), Position values (empty num. array, 0), No. of digits (4), Error in (no error)

Output Error out

Remarks Moves specified axes to specified absolute positions. No. of digits is the number of digits after the decimal point in the position value(s) that will be sent.

E-710: **Caution: The use of “NMOV.vi” is the same as that of the GCS command VI “MOV.vi”, but it is not a GCS command VI!**

This function is faster than “MOV.vi” but

- does not check range limits and servo states
- does not move the axes synchronously.

When the commanded target is outside the range limits, the axis will stop at its physical limit. To go back to normal operation, command the axis to a valid position using “MOV.vi”.

When servo is off, the axis does not move.

The commanded target can be queried with “MOV?.vi”.

Note: Due to the emulation of the native E-710 command set, the execution

of “MOV.vi” is noticeable slower than those of the native firmware commands. Therefore “NMOV.vi” is provided for applications which require quickest possible response to motion commands.

2.5.3. NMVR.vi

Valid for E-710

Input System number (1), Axes to move (empty string array), Position values (empty num. array, 0), No. of digits (4), Error in (no error)

Output Error out

Remarks Moves specified axes to specified relative positions. No. of digits is the number of digits after the decimal point in the position value(s) that will be sent.

E-710: **Caution: The use of “NMVR.vi” is the same as that of the GCS command VI “MVR.vi”, but it is not a GCS command VI!**

This function is faster than “MVR.vi” but

- does not check range limits and servo states
- does not move the axes synchronously.

When the commanded target is outside the range limits, the axis will stop at its physical limit. To go back to normal operation, command the axis to a valid position using “MOV.vi”.

When servo is off, the axis does not move.

The commanded target can be queried with “MOV?.vi”.

Note: Due to the emulation of the native E-710 command set, the execution of “MVR.vi” is noticeable slower than those of the native firmware commands. Therefore “NMVR.vi” is provided for applications which require quickest possible response to motion commands.

2.5.4. Wait for hexapod system axes to stop.vi

Valid for F-206, M-8X0 (but must be present for all other systems also)

Input System number (1), All axes? (T), Axes to wait for (empty string array), Stop refnum (F), Local stop (F), Error in (no error)

To wait for the hexapod to stop, only one hexapod axis (X, Y, Z, U, V or W) needs to be commanded, because the VI cannot distinguish between the different hexapod axes.

F-206: Axes to wait for can be any of X, Y, Z, U, V, W, A, B, K, L, M

M-8X0: Axes to wait for can be any of X, Y, Z, U, V, W, A, B

Output Error out

Remarks This vi waits for the specified axes of a PI hexapod system (hexapod axes X, Y, Z, U, V, W and separate axes A, B) to stop using #5 polling. If a NanoCube axis (K, L or M) is commanded, the VI will return immediately. If one of the hexapod axes (X, Y, Z, U, V or W) is commanded, it will wait for all six hexapod axes to stop. It returns immediately if a communications error occured, or if Local stop or Stop refnum is TRUE. When using as a sub-VI, use Refnum stop to stop VI from caller. Required by “General wait for movement to stop.vi”.

2.6. File handling VIs ("File handling.llb")

2.6.1. GetDataFormat.vi

Valid for	C-880, E-710, F-206, M-8X0
Input	IOSource (Read (F)/Write (F), Path (empty path), ArrayName (empty string), Datastream (empty string)), Error in (no error)
Output	Header out (Separator, NDim, Remarks), DataOK, Found Header, Data Type, NData, Names out, Sample time, Error out
Remarks	This vi checks the format of a data file. See separate TN and control descriptions in the diagram for more information.

2.6.2. TableIO.vi

Valid for	C-880, E710, F-206, M-8X0
Input	IOSource (Read (F)/Write (F), Path (empty path), ArrayName (empty string), Datastream (empty string)), Header in (Separator (\t), NDim (0), Remarks (empty string)), Names in (empty string array), Table in (empty 2D num. array), Sample time in (0), (Error in (no error)
Output	Datastream out, Header out (Separator, NDim, Remarks), Names out, Table out, Sample time out, Error out
Remarks	This vi reads or writes data files in table format. See separate TN and control descriptions in the diagram for more information. Sub-VI for "DRR?.vi".

2.7. Limit- and reference-specific commands ("Limits.llb")

2.7.1. ATZ.vi

Valid for	E-710
Input	System number (1), Affected axes (empty string array), All axes? (F), Low voltage parameter (empty num. array), Error in (no error)
Output	Error out
Remarks	Performs an automatic zero-point calibration for the specified linear axes (see the E-710 User Manual for details), waits until this procedure has finished and returns whether ATZ was successful or not. Select NaN as Low voltage parameter if you want the controller to use the stored values it has. The home position is reset to default by ATZ.

2.7.2. DFH.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710
Input	System number (1), DFH axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? must be TRUE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880K005: VI only supported when called through PI_Multix.vi

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

Output Error out

Remarks Defines the current position of the DFH axes as home position. Their position value is set to 0. The home position is reset to default by ATZ, REF, MNL, MPL or similar commands.

2.7.3. DFH?.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710

Input System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes? = TRUE, then Axis identifier? must be TRUE

C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE

C-844: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? can be FALSE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? can be FALSE

C-880K005: VI only supported when called through PI_Multix.vi

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

Output Home position, Error out

Remarks Returns the difference between the current home position and the absolute or initial zero point (default home position) for each of the queried axes.

2.7.4. GOH.vi

Valid for C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710

Input System number (1), GOH axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error)

C-843: If All axes? = TRUE, then Axis identifier? must be TRUE

C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE

C-844: If All axes? = TRUE, then Axis identifier? must be TRUE

C-848: If All axes? = TRUE, then Axis identifier? must be TRUE

C-865: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880: If All axes? = TRUE, then Axis identifier? must be TRUE

C-880K005: VI only supported when called through PI_Multix.vi

E-710: If All axes? = TRUE, then Axis identifier? must be TRUE

Output Error out

Remarks Moves specified axes to their home positions.

2.7.5. TMN?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? can be FALSE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? can be FALSE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Minimum travel limit, Error out
Remarks	Returns minimum (low-end) travel limit (position of negative limit switch, or value of negative soft limit, if set, whichever is higher).

2.7.6. TMX?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) C-843: If All axes? = TRUE, then Axis identifier? must be TRUE C-843.PM: If All axes? = TRUE, then Axis identifier? must be TRUE C-844: If All axes? = TRUE, then Axis identifier? must be TRUE C-848: If All axes? = TRUE, then Axis identifier? can be FALSE C-865: If All axes? = TRUE, then Axis identifier? must be TRUE C-880: If All axes? = TRUE, then Axis identifier? can be FALSE E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Maximum travel limit, Error out
Remarks	Returns maximum (high-end) travel limit (position of positive limit switch or value of positive soft limit, if set, whichever is lower).

2.8. Commands for Optical or Analog Signals ("Optical or Analog Input.llb")

2.8.1. TAD?.vi

Valid for	E-710
Input	System number (1), Sensors to query (empty string array), All sensors? (F), Sensor identifier? (T), Error in (no error) E-710: If All sensors? = TRUE, then Sensor identifier? must be TRUE.
Output	AD value, Error out
Remarks	Returns AD value for the specified sensor number. E-710: Command is available for 4-channel version only and in command level 1 only (see "CCL.vi", "CCL?.vi")

2.8.2. TNS?.vi

Valid for	E-710
Input	System number (1), Sensors to query (empty string array), All sensors? (F), Sensor identifier? (T), Error in (no error) E-710: If All sensors? = TRUE, then Sensor identifier? must be TRUE.
Output	Nom. sensor value, Error out
Remarks	Returns norminized sensor value for the specified sensor number. E-710: Command is available in command level 1 only (see "CCL.vi", "CCL?.vi")

2.8.3. TSC?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Number of sensor channels, Error out
Remarks	Returns the number of available sensor channels.

2.8.4. TSP?.vi

Valid for	E-710
Input	System number (1), Sensors to query (empty string array), All sensors? (F), Sensor identifier? (T), Error in (no error) E-710: If All sensors? = TRUE, then Sensor identifier? must be TRUE.
Output	Sensor position, Error out
Remarks	Returns sensor position for the specified sensor number. E-710: Command is available in command level 1 only (see "CCL.vi", "CCL?.vi")

2.9. Wave-Generator-Specific Commands ("WaveGenerator.llb")

2.9.1. #9.vi

Valid for	E-516, E-710
Input	System number (1), Error in (no error)
Output	WG running? (T/F), Error out
Remarks	Polls to determine whether a wave generator is running for any of the connected axes by sending the single ASCII character 9. Connected axes are read from Global2.vi and displayed on the front panel for assignment.

2.9.2. DDL.vi

Valid for	E-710
Input	System number (1), Table number (1), Offset (0), No. of digits (6), Values (empty num. array), Error in (no error) E-710: Table number can be 1 to 8. First stored value in table has index 1, index 0 is the DDL repeat number. Maximum 32768 values can be stored.
Output	Error out
Remarks	Loads the specified values to the specified DDL Table. Please refer to the GCS DLL Manual for units and restrictions.

2.9.3. DDL?.vi

Valid for	E-710
Input	System number (1), Table number (1), Offset (0), N (100), Error in (no error) E-710: Table number can be 1 to 8. First stored value in table has index 1, index 0 is the DDL repeat number. Maximum 32768 values can be stored.
Output	Values, Error out
Remarks	Returns N values from specified DDL Table (in the controller memory). For large N values, communication timeout must be set long enough, otherwise a comm.error may occur.

2.9.4. GWD?.vi

Valid for	E-516, E-710
Input	System number (1), Query axis (empty string), Xo (0), N (10), Nmax (64), Error in (no error) E-710: Query axis is identical with wave table number. Nmax must be less than the length of the defined wave.
Output	Waveform points, Error out
Remarks	Returns N waveform sequence points. Nmax is the maximum number of points that can be read at once. If N > Nmax, then the VI will query GWD? more than once to read all N values.

2.9.5. TLT?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Number of tables, Error out
Remarks	Returns the number of DDL tables available in the controller.

2.9.6. TWC.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Error out
Remarks	Clears the trigger wave settings for the waveform.

2.9.7. TWG?.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Number of wave generators, Error out E-710: Number of wave generators = 2
Remarks	Returns the number of available wave generators.

2.9.8. TWS.vi

Valid for	E-710
Input	System number (1), Wavepoint (empty num. array, 0), Trigger value (empty num. array, 0), Error in (no error) E-710: Wavepoint can be 1-16000. Trigger value is bitmapped: • Bit 0: trigger line 1: 0 not active, 1 active • Bit 1: trigger line 2: 0 not active, 1 active • Bit 2: trigger line 3: 0 not active, 1 active • Bit 3: trigger line 4: 0 not active, 1 active • Bit 8: If = 0, then the Trigger values apply to corresponding Wavepoints only. If = 1, then the Trigger value applies to all points between the last point set by this command and the corresponding Wavepoint point.
Output	Hidden error (T/F), Error out
Remarks	Sets trigger values for point(s) on the waveform and checks for error. Trigger value is bit-mapped. Hidden error is TRUE if selected system reports error code ≠ 0.

2.9.9. WAV.vi

Valid for	E-516, E-710
Input	System number (1), Affected axes (empty string array), Waveform (SIN), No. of digits (3), Xo (0), N (1000), Add? (+) (F), Append? (&) (F), SIN Parameters (5,7692,0,0,5), TAN Parameters (10,6283,0,0,0), POL Parameters (0,0,0.01,0,0,0,0), RAMP Parameters (500,0,5,0), SIN_P Parameters (500,5,0), LIN Parameters (0,5,0), PNT Parameters (0.001,0.002,...,0.01), Nmax for PNT (5), Segment length (0), Error in (no

error)

E-516: Waveform can be: SIN, TAN, POL or PNT.
The following parameters are equivalent (see GCS DLL manual):
Affected axes = szAxes, Xo = nStart, N = nLength, Add? = Add,
PNT Parameters = pPoints (PNT parameters is an array consisting of single data points which will be sent in groups of Nmax for PNT data points.)

Valid for SIN: Affected axes, No. of digits, Xo, N, Add?, SIN Parameters (A, N, Xo, Phi and B)

Valid for TAN: Affected axes, No. of digits, Xo, N, Add?, TAN Parameters (A, N, Xo, Phi and B)

Valid for POL: Affected axes, No. of digits, Xo, N, Add?, POL Parameters (Xo, Ao to A5)

Valid for PNT: Affected axes, No. of digits, Xo, N, Add?, Nmax for PNT, PNT Parameters (Ao to An-1)

E-710: Affected axis is identical with wave table number.
Waveform can be: SIN_P, RAMP, LIN or PNT.

The following parameters are equivalent (see GCS DLL manual):
Affected axes = szWaveTableIds, Xo = iOffsetOfFirstPointInWaveTable,
N = iNumberOfPoints, Append? = iAddAppendWave,
C = iCenterPointOfWave, J = iNumberOfSpeedUpDownPointsInWave,
A = dAmplitudeOfWave, O = dOffsetOfWave,
Segment length = iSegmentLength, PNT Parameters = pdWavePoints (PNT parameters is an array consisting of single data points which will be sent in groups of Nmax for PNT data points.)

O (dOffsetOfWave) is only valid if a wave segment is being concatenated to an existing wave (Append? = TRUE).

Valid for SIN_P: Affected axes, No. of digits, Xo, N, Append?, Segment length, SIN_P Parameters (C, A and O)

Valid for RAMP: Affected axes, No. of digits, Xo, N, Append?, Segment length, RAMP Parameters (C,J,A and O)

Valid for LIN: Affected axes, No. of digits, Xo, N, Append?, Segment length, LIN Parameters (J,A and O)

Valid for PNT: Affected axes, No. of digits, Xo, N, Append?, Segment length, Nmax for PNT, PNT Parameters (Ao to An-1). Nmax for PNT must be ≤ Wave storage max. value set with WMS.vi. Segment length must be zero for PNT.

The length of the wave to define must be less than the value defined with "WMS.vi".

Output Hidden error (T/F), Error out

Remarks Defines or modifies a stored waveform for one or more axes. Please refer to the GCS DLL manual for a description of waveform parameters, units and restrictions. No. of digits is the number of digits after the decimal point in the parameter value(s) that will be sent. Hidden error is TRUE if selected system reports error code ≠ 0.

2.9.10. WAV?.vi

Valid for	E-710
Input	System number (1), Axes to query (empty string array), Parameter numbers (empty num. array, 0), Error in (no error) E-710: Axis to query is identical with wave table number. The following parameter number is valid: • 1: Number of waveform points for currently defined wave
Output	WAV Parameter value, Error out
Remarks	Returns waveform parameter values for queried axes and parameter numbers.

2.9.11. WCL.vi

Valid for	E-710
Input	System number (1), WCL axes (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Error out
Remarks	Clears waveform associated with specified axis. E-710: Does also clear DDL table.

2.9.12. WGO.vi

Valid for	E-516, E-710
Input	System number (1), Affected axes (empty string array), WGO Parameter (empty num. array, 0), Error in (no error) E-710: Affected axis is identical with wave generator number.
Output	Hidden error (T/F), Error out
Remarks	Enables, disables, and sets wave generator output mode. Parameter is bit-mapped. Hidden error is TRUE if selected system reports error code ≠ 0. See GCS DLL Manual for parameter definition. E-710: Only two wave generators can run simultaneously.

2.9.13. WGO?.vi

Valid for	E-516, E-710
Input	System number (1), Axes to query (empty string array), , All axes? (F), Axes identifier? (T), Error in (no error) E-516: If All axes? = TRUE, then Axis identifier? can be FALSE E-710: Axis to query is identical with wave generator number. If All axes? = TRUE, then Axis identifier? must be TRUE
Output	WGO parameter, Error out
Remarks	Returns WGO parameter for specified axes.

2.9.14. WGR.vi

Valid for	E-710
Input	System number (1), Error in (no error)
Output	Error out
Remarks	Starts a new recording.

2.9.15. WMS.vi

Valid for	E-710
Input	System number (1), Without axis ID? (F), Axes to set (empty string array), Wave storage max. (empty num. array, 0), Error in (no error) E-710: Without axis ID? = FALSE. A maximum number of 63488 samples can be defined for all axes in total.
Output	Error out, Hidden error
Remarks	Sets maximum size of wave-point storage available for given axes and checks for error. If Without axis ID? is TRUE, then Axes to set is ignored and first field of Wave storage max. array is used as number of wave points for every axis. Hidden error is TRUE if selected system reports error code ≠ 0.

2.9.16. WMS?.vi

Valid for	E-710
Input	System number (1), Axes to query (empty string array), All axes? (F), Axis identifier? (T), Error in (no error) E-710: If All axes? = TRUE, then Axis identifier? must be TRUE
Output	Wave storage max., Error out
Remarks	Returns maximum size of wave-point storage for specified axes.

2.10. Joystick-specific VIs (“Joystick.llb”)

2.10.1. Calculate joystick scaling.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	XPos min (0), YPos min (0), XPos max (65535), YPos max (65535), Resolution factor X (150), Resolution factor Y (150)
Output	XPos min scaled, YPos min scaled, XPos max scaled, YPos max scaled, XPos Center scaled, YPos Center scaled, Scaling factor X, Scaling factor Y
Remarks	Sub-VI for operation with a joystick connected to the game port of the host computer. Calculates joystick position scaling. If Resolution factor * = *Pos max, maximum resolution is achieved. *Pos min and *Pos max depend on the Windows joystick calibration.

2.10.2. Read joystick.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	Joystick ID (0), Error in (no error)
Output	XPos, YPos, Button 1 pressed (T/F), Button 2 pressed (T/F), Error out
Remarks	Sub-VI for operation with a joystick connected to the game port of the host computer. Reads joystick position and button status for a standard 2-button 2-axis joystick. Install joystick driver and calibrate joystick in the Windows control panel before running this VI.

2.10.3. Scale joystick data.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	XPos (0), YPos (0), XPos Center (0), YPos Center (0), Dead band X (0), Dead band Y (0), Factor X (0), Factor Y (0)
Output	XPos scaled, YPos scaled
Remarks	Sub-VI for operation with a joystick connected to the game port of the host computer. Scales joystick position. Use output value from "Calculate joystick scaling.vi" for Factor *. Dead band * is the maximum scaled position value that does not result in any motion.

2.11. Support VIs ("Support.llb")

Support VIs are sub-VIs for command VIs which make certain programming tasks more convenient. They can also be used for building main programs.

Caution: Please do not change these VIs, as that might cause the command VIs that use them to fail.

2.11.1. Analyse input string for terminal.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	String new (empty string), Last string sent (empty string)
Output	String out, Out not equal to in? (T/F)
Remarks	This VI is a sub-VI for "PI Terminal.vi". It analyses String new and returns it in String out if it is not empty and does not contain a "#" at the beginning. In case of an empty new string, Last string sent is returned. If String new contains a "#" character, the corresponding ASCII character is returned.

2.11.2. Assign booleans from string to axes.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Queried axes (empty string array), All axes queried? (F), Input string (empty string), Error in (no error)
Output	Booleans(T/F), Error out
Remarks	This VI assigns numerical values from input string to boolean values for queried axes. If All axes? is TRUE, connected axes are read from Global2.vi and displayed on the front panel for assignment. Example: An input string like "A=0SpaceLinefeedB=1Linefeed" or "0SpaceLinefeed1Linefeed" will be converted to an output array consisting of two values "FALSE; TRUE".

2.11.3. Assign DRC values.vi

Valid for	E-710
Input	Input string (empty string), Queried Rec. table (empty num. array, 0), Error in (no error)
Output	Source ID (empty string array), Rec. option (empty num. array, 0), Error out
Remarks	This VI assigns values (Source ID and Rec. option) from Input string to Queried Rec. tables. Sub-VI for DRC?.vi.

2.11.4. Assign SPA values from string to axes.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816
Input	Input string (empty string), Queried axes (empty string array), Parameter number (empty num. array, 0), Error in (no error)
Output	Parameter values, Parameter strings, Error out
Remarks	This VI assigns numerical values / strings from input string to queried axes and parameter numbers. Sub-VI for "SPA?.vi", "SPA?_String.vi" and SEP?.vi".

2.11.5. Assign SPA_Hex values from string to axes.vi

Valid for	E-710
Input	Input string (empty string), Queried axes (empty string array), Parameter number (hex) (empty num. array, 0), Error in (no error)
Output	Parameter values, Parameter strings, Error out
Remarks	This VI assigns numerical values / strings from input string to queried axes and hex. parameter numbers. Sub-VI for “SPA?_Hex.vi”, “SPA?_Hex_String.vi”.

2.11.6. Assign values from string to axes.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Queried axes (empty string array), All axes queried? (F), Input string (empty string), Error in (no error)
Output	Values, Strings, Error out
Remarks	This VI assigns numerical values and/or single lines from input string to queried axes. If All axes? is TRUE, connected axes are read from Global2.vi and displayed on the front panel for assignment.

2.11.7. Build command substring.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Affected axes (empty string array), No. of digits (4), Parameters (empty num. array, 0)
Output	Command substring
Remarks	This VI builds a command substring by combining axis identifier and parameter. No. of digits is the number of digits after the decimal point in the parameter value(s) that will be sent. Example: For Affected axes = A; B, Parameters = 1.2342; 2.3 and No. of digits = 3 the resulting string is “SpaceA1.234SpaceB2.300”.

2.11.8. Build num command substring.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	No. of digits (4), Num 1 (empty num. array, 0), Num 2 (empty num. array, 0)
Output	Command substring
Remarks	This VI builds a command substring by combining Num1, Space and Num2. No. of digits is the number of digits after the decimal point in the Num 1/2 value(s) that will be sent. Example: For Num 1 = 1.24; 3.25456, Num 2 = 5.0; 7.4321 and No. of digits = 3 the resulting string is “Space1.240Space5.000Space3.255Space7.432"

2.11.9. Build query command substring.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0

Input	System number (1), Axes to query in (empty string array), Query all axes? (F), With space? (F), Axis identifer? (T),
Output	Command substring, Axes to query out, Number of rows
Remarks	This VI builds a query command substring. If All axes? is TRUE, connected axes are read from "Global2.vi" and returned in Axes to query out, otherwise Axes to query out is identical with Axes to query in. Number of rows is size of the Axes to query out array. If Axis identifier? is FALSE, command substring is an empty string (e.c. for systems which accept commands like POS? without axis IDs). If With space? is TRUE, a space character is added between the axes identifiers. Example: If axes A;B;C;D are connected to the system to command, Axes to query in is A;B;D, Query all axes? is TRUE and Use Axis identifier? is TRUE, resulting Command substring is "ABCD", Number of rows is 4 and Axes to query out is A;B;C;D. If With space? is TRUE, the resulting Command substring is "A B C D".

2.11.10. Build SPA_Hex command substring.vi

Valid for	E-710
Input	Axes to set (empty string array), No. of digits (4), Parameter number (hex) (empty num. array, 0), Parameter values (empty num. array, 0)
Output	SPA_Hex command substring
Remarks	This VI builds a command substring for the SPA_Hex command. No. of digits is the number of digits after the decimal point in the parameter value(s) that will be sent. Sub-VI for "SPA_Hex.vi".

2.11.11. Build SPA query command substring.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816
Input	Axes to query (empty string array), Parameter number (empty num. array, 0)
Output	Command substring, Number of rows
Remarks	This VI builds an SPA? Command substring. Axes and parameters are combined into a substring. Number of rows is size of Axes to query array. Sub-VI for "SPA?.vi" and "SEP?.vi" .

2.11.12. Build SPA_Hex query command substring.vi

Valid for	E-710
Input	Axes to query (empty string array), Parameter number (hex) (empty num. array, 0)
Output	Command substring, Number of rows
Remarks	This VI builds an SPA?_Hex Command substring. Axes and parameters are combined into a substring. Number of rows is size of Axes to query array. Sub-VI for "SPA?_Hex.vi" .

2.11.13. Build SPA_String command substring.vi

Valid for	C-843, C-843.PM, C-865, E-710
Input	Axes to set (empty string array), Parameter number (empty num. array, 0),

	Parameter values (empty string array)
Output	SPA_String command substring
Remarks	This VI builds a command substring for the SPA_String and SEP command. Sub-VI for “SPA_String.vi” and “SEP.vi”.

2.11.14. Build SPA_Hex_String command substring.vi

Valid for	E-710
Input	Axes to set (empty string array), Parameter number (hex) (empty num. array, 0), Parameter values (empty string array)
Output	SPA_Hex_String command substring
Remarks	This VI builds a command substring for the SPA_Hex_String command. Sub-VI for “SPA_Hex_String.vi”.

2.11.15. Build stringplusnum substring.vi

Valid for	All systems
Input	Sequence (String1String2String3Value1Value2), String1 (empty string array), String2 (empty string array), String3 (empty string array), Value1 (empty num. array, 0), Value2 (empty num. array, 0), No. of digits Value1 (6), No. of digits Value2 (6), Input selection (T,T,T,T,F), Error in (no error)
Output	Substring, Error out
Remarks	This vi builds a command substring by combining up to three strings and two values in the given order.

2.11.16. Build WAV command substring.vi

Valid for	E-516, E-710
Input	Axes to set (empty string array), Waveform (SIN), No. of digits (3), Xo (0), N (1000), Add? (+) (F), SIN Parameters (5,7692,0,0,5), TAN Parameters (10,6283,0,0,0), POL Parameters (0,0,0.01,0,0,0,0), PNT Parameters (0.001,0.002...0.01),
Output	WAV command substring
Remarks	This vi builds a command substring for the WAV command. No. of digits is the number of digits after the decimal point in the parameter value(s) that will be sent. Sub-VI for “WAV.vi”.

2.11.17. Commanded axes connected?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Commanded axes (empty string array), Error in (no error)
Output	Hidden error (T/F), Error out
Remarks	This VI checks if Commanded axes are a subset of all connected axes (read from “Global2.vi”) and returns Hidden error TRUE if this is not the case. Connected axes are defined by “Define connected axes.vi”. White space strings in Commanded axes are ignored.

2.11.18. Commanded stage name available?.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Commanded stages (empty string array), Error in (no error)
Output	Hidden error (T/F), Error out
Remarks	This VI checks if Commanded stages is a subset of all available stages and returns Hidden error TRUE if this is not the case. Available stages are defined by “VST?.vi”.

2.11.19. Convert num array to string.vi

Valid for	All systems
Input	Number of digits (4), Num. values (empty num. array)
Output	Output string
Remarks	This vi converts an array of numerical values to a space separated output string. The difference to LabVIEW's native Array to Spreadsheet String function is that no carriage return or newline is added.

2.11.20. Count occurrences in string.vi

Valid for	All systems
Input	Input string (empty string), Expression (empty string)
Output	Occurrences
Remarks	This VI counts, how often an expression occurs in a string.

2.11.21. Define axes to command from boolean array.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Axes to query (empty string array), Command axis? (empty bool. array, F)
Output	Axes to command, Remaining axes
Remarks	This VI returns only those axis IDs from the Axes to query array in the Axes to command array which have a boolean value TRUE in the Command axis? array, and all remaining axes in the Remaining axes array.

2.11.22. GCSTranslateError.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	Error in (no error)
Output	Error out, GCS Error?, Error description
Remarks	Returns if error in contains a GCS error code and if this is the case, it displays the corresponding error message and appends it to source in error out.

2.11.23. General wait for movement to stop.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0

Input	System no. (1), Axes to wait for (empty string array), All axes? (T), Polling cycle time, ms (1), Additional wait time, ms (0), Error in (no error) E-710: VI will not wait, use appropriate Additional wait time to wait for commanded moves. F-206: VI will not wait for INI procedure to complete. M-8X0: VI will not wait for INI procedure to complete.
Output	Error out
Remarks	This VI waits for the specified axes to stop. An additional wait time can be specified. The wait method depends on the system to command. “Define connected systems.vi” must be run before running this vi. Requires “Wait for axes to stop.vi”, “#5.vi”,“STA?.vi”, “#5_old.vi”, “ONT?.vi” and “Wait for hexapod system axes to stop.vi” to be present.

2.11.24. Get arrays without blanks.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	String array in (empty string array), Values in (empty num. array), Booleans in (empty bool. array, F), Array size in (0)
Output	String array out, Values out, Booleans out, Array size out
Remarks	Returns the string array and related values and boolean arrays without white space string fields.

2.11.25. Get lines and values from string.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	Array size (0), Input string (empty string)
Output	Numerical values, Strings
Remarks	This VI returns numerical values and single lines from input string without any axis assignment. If number of lines/values (Array size) is known, algorithm is faster, otherwise Array size = 0 should be used. Sub-VI for “VST?.vi” and “STE?.vi”.

2.11.26. Get lines from string.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	Array size (0), Input string (empty string)
Output	Strings
Remarks	This VI returns single lines from input string. If number of lines (Array size) is known, algorithm is faster, otherwise Array size = 0 should be used. Sub-VI for “VST?.vi”.

2.11.27. Get string array size without blanks.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	String array (empty string array)
Output	Corrected array size
Remarks	This VI returns the size of a string array without counting white space strings.

2.11.28. How often does string contain regular expression.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Regular expression (empty string), String (empty string)
Output	Number
Remarks	This VI returns a count of the occurances of a regular expression in a string.

2.11.29. Longlasting one-axis command.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710, F-206
Input	System number (1), Axis to command (empty string), Command (empty string), Axis and value? (F), Value (NaN), Error in (no error)
Output	Answer (T/F), Error out
Remarks	This VI sends a command (like REF, MNL or MPL), polls with #7 for controller-ready signal and returns original (boolean) command response.

2.11.30. Manual VMO.vi

Valid for	C-844, C-848, C-848.PM, C-865, C-880, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Axes to command (empty string array), Minimum pos. (empty num. array), Maximum pos. (empty num. array), Position values (empty num. array, 0), Error in (no error)
Output	Move possible (T/F), Error out
Remarks	Virtual movement. Indicates whether a move to the specified position is possible or not by checking if the commanded position value is within the given position range. Axes will NOT be moved.

2.11.31. Return single characters from string.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, F-206, E-516, E-710, E-816, M-8X0
Input	Input string (empty string), Invert order (F), Error in (no error)
Output	Character array (empty string array), Error out
Remarks	Get single characters from input string.

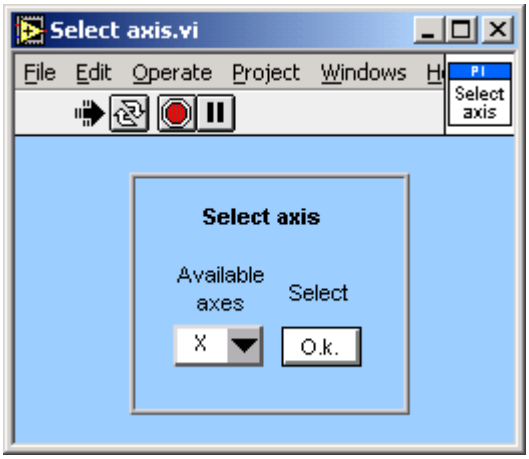
2.11.32. Round with options.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, F-206, E-516, E-710, E-816, M-8X0
Input	No. of digits to round to (2), Round mode selection (Round to nearest), Numeric in (0), Num array in (empty num. array)
Output	Numeric out, Num array out
Remarks	Rounds Numeric in and Num array in according to No. of digits to round to and Round mode selection.

2.11.33. Select axis.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, F-206, E-516, E-710, E-816, M-8X0

Input	System number (1)
Output	Selected axis, Index of axis in Global2
Remarks	This VI reads all connected axes from Global2 and writes them into a menu ring control for selection. The selected axis and it's index in Global2 are returned.



2.11.34. Selection to string array.vi

Valid for	All systems
Input	Selection array (empty Menu Ring array, 0), String input (empty string array)
Output	String array
Remarks	This vi returns a string array which contains strings according to the selected value of String input. Example: For Selection array = (2,0,1) and String input = (A,B,C) the resulting String array is (C,A,B).

2.11.35. Select values for chosen axes.vi

Valid for	C-843, C-843.PM, C-848, C-865, C-880, C-880K005, E-710
Input	Queried axes (empty string array), Values (empty num. array), Axes subset (empty string array)
Output	Values subset
Remarks	This VI returns only values for the given axes subset.

2.11.36. Select with boolean array input.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	Size (0), T string (empty string), F string (empty string), T/F (empty boolean array)
Output	String array out
Remarks	This vi returns a string array of a given size with T string and F string, depending on the boolean value at the corresponding index of T/F.

2.11.37. Substract axes array subset from axes array.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0

Input	Axes to query (empty string array), Axes subset (empty string array)
Output	Axes to command
Remarks	This VI returns only these axes IDs from the Axes to query array which are **not** present in the Axes subset array. Needed by “Define axes to command from boolean array.vi”.

2.11.38. Unbundle/bundle interface clusters for PI Terminal.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-516, E-710, E-816, F-206, M-8X0
Input	System number (1), Interface configuration (RS232, 1000, COM1, 57600), DLL interface configuration (C-843, Board, 1), Flow control (All FALSE, x13, x11, x0)
Output	Interface, RS232 configuration system, GPIB configuration system, DLL for device, DLL interface
Remarks	This VI is a sub-VI for “PI Terminal.vi”. It unbundles Interface configuration and DLL interface configuration and returns the cluster contents in a different composition which is used by “PI Terminal.vi”.

2.11.39. Wait for answer of longlasting command.vi

Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, C-880K005, E-710, but must be present for E-516, F-206, M-8X0 too
Input	System number (1), Stop refnum (F), Local stop (F), Error in (no error)
Output	Answer (T/F), Error out
Remarks	This VI waits for the answer of commands like REF, MPL, MNL or scanning routines using #7 polling and stops if answer has come, Stop refnum or Local stop is TRUE, or if a communications error occured. Sub-VI for Long-lasting, one-axis commands and controller-algorithm commands. Requires “#7.vi” to be present. When using as a sub-VI, use Stop refnum to stop VI from caller.

2.11.40. Wait for axes to stop.vi

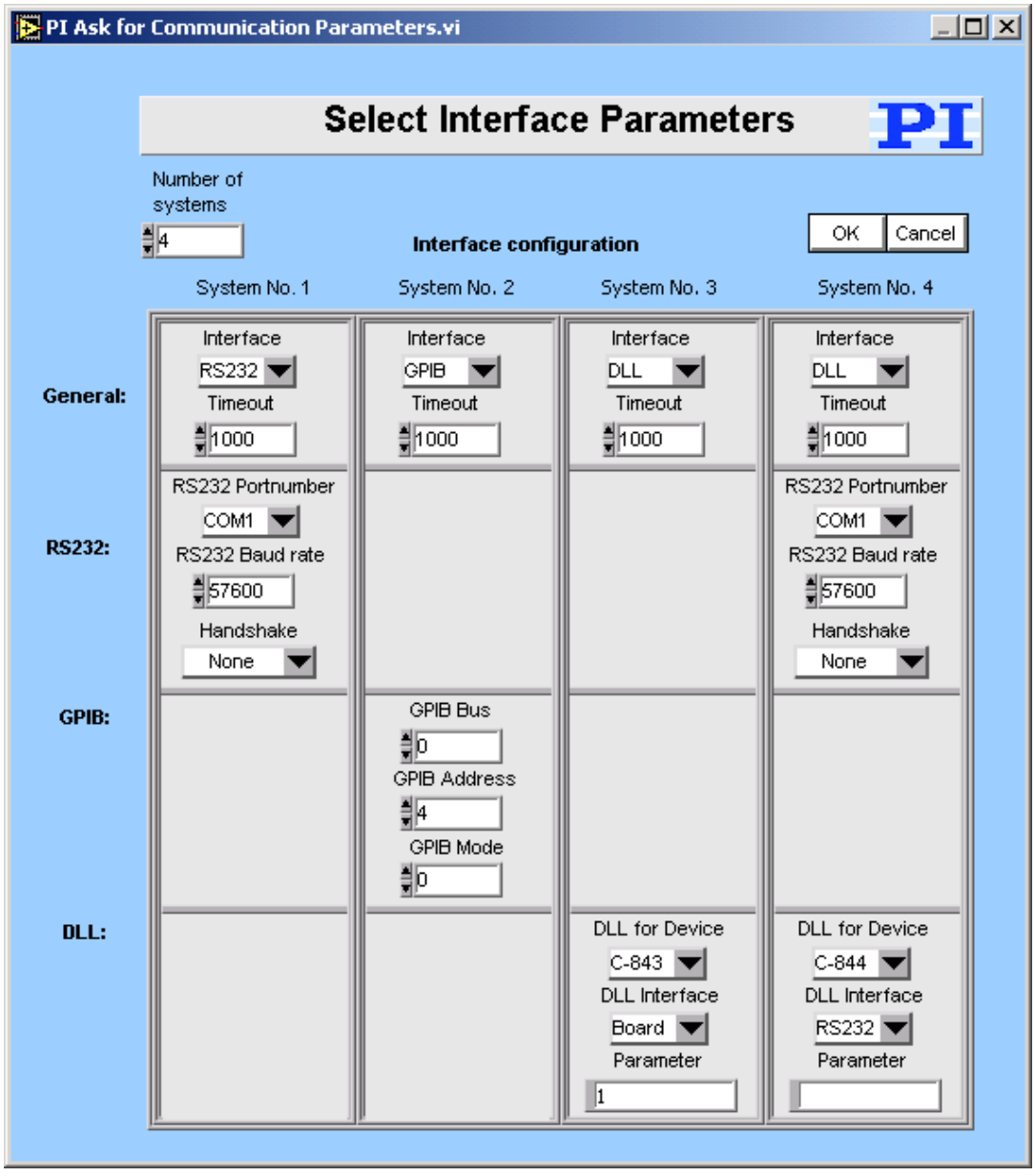
Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880 (but must be present in Support.llb for all other systems also)
Input	System number (1), Axes to wait for (empty string array), With status bit polling? (F), Polling cycle time, ms (400), Stop refnum (F), Local stop (F), Error in (no error) C-843: With status bit polling? = FALSE C-843.PM: With status bit polling? = FALSE C-844: With status bit polling? = FALSE C-880: With status bit polling? = TRUE
Output	Error out
Remarks	This VI waits for the specified axes to stop using #5 polling. It also stops if a communication error occured, Stop refnum or Local stop is TRUE. Requires “STA?.vi” to be present. Required by “General wait for movement to stop.vi”. When using as a sub-VI, use Stop refnum to stop VI from caller.

3. High Level VIs

3.1. Terminal.vi

The terminal VI is a stand-alone application. It first asks the user to specify the full configuration (number of controlled systems, RS-232, GPIB or DLL communication, communications parameters), then it establishes a connection with a selected system. This will work for all PI devices which support the PI General Command Set, or at least follow the same syntax rules and support the *IDN? and ERR? commands.

After starting the VI, the interface parameters of the systems with which to communicate must be selected. For this reason, “PITerminal.vi” calls “PI Ask for Communication Parameters.vi”. Select here the number of connected PI systems that you want to communicate with. For each system, select the appropriate interface parameters.



C-843:	Interface = DLL, DLL for Device = C-843, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).
C-843.PM:	Interface = DLL, DLL for Device = C-843.PM, DLL Interface = Board, Parameter = Board number (1 for first C-843 board).
C-844:	Interface = DLL, DLL for Device = C-844, DLL Interface = RS232 or GPIB, Parameter = empty string, RS232 baud rate = 9600
C-865:	Interface = DLL, DLL for Device = C-865, DLL Interface = RS232, Parameter = empty string, RS232 baud rate = set as appropriate
C-880:	Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.
C-848:	Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.
C-880K005:	Interface = RS232, Input and output HW handshake must be FALSE.
E-516:	Interface = RS232 or GPIB, RS232: Input and output HW handshake must be TRUE.
E-710:	Interface = DLL, DLL for Device = E-710, DLL Interface = RS232 or GPIB, Parameter = empty string
E-816:	Interface = RS232 (supports only RS-232 communication), Input and output HW handshake must be TRUE.
F-206:	Interface = RS232 or GPIB, The error status will not be cleared by this VI. The first ERR? query will report a hidden error with error code 1, which will be cleared during system initialization (INI). RS232: Input and output handshake settings must be FALSE.
M-8X0:	Interface = RS232 or GPIB. RS232: Input and output handshake settings must be FALSE.

If the chosen timeout value is greater than 300 ms, it will automatically be set to 300 ms for a fluid program operation.

In the upper window ("Send") the user can enter commands which will be transmitted to the chosen device one line at a time when the ENTER key is pressed.

All controller responses are displayed in the Receive response window, which can be cleared by pressing the Clear Receive Window button or F2.

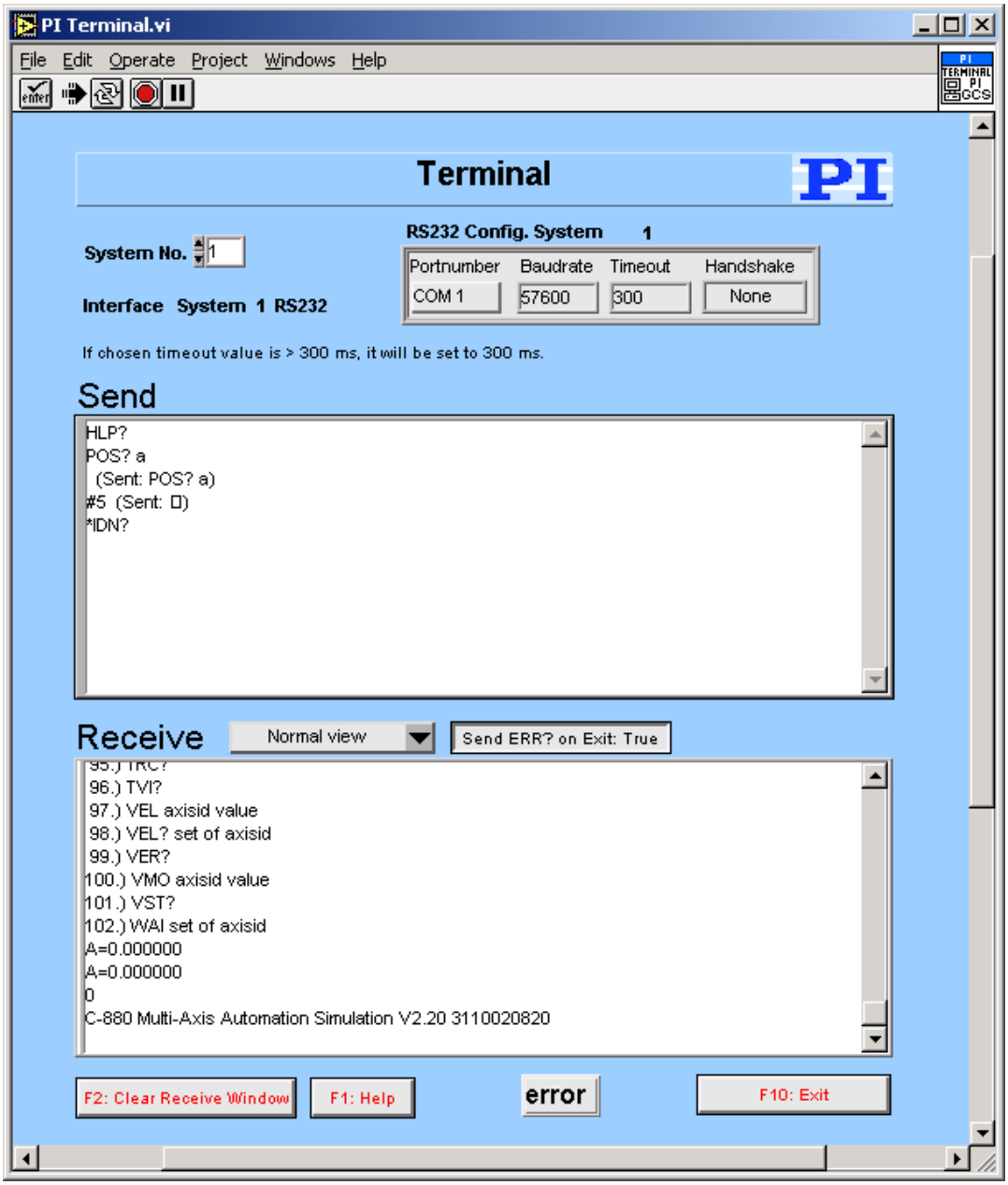
The view style of the Receive window can be changed to Show all characters or Hex View using the menu ring above the Receive window.

Exit or F10 will terminate the terminal application.

To send the last command again, just press the ENTER key again. The next line will then show the following entry: "(Send: *cmd*)" with *cmd* being the command from the line before, which was resent.

When the terminal application has just been started, pressing ENTER without entering a command will send "*IDN?" to the chosen system.

New commands can only be inserted into the last line of the Send window. The user can scroll through the history of the Send window using the scroll bar or the cursor up/down keys, but cannot change the history or resend commands by pressing ENTER unless in the last line. Pressing ENTER will always resend the last command, no matter where the cursor is positioned. Selecting text and using copy and paste (Ctrl+C, Ctrl+V) works for single lines, if only the contents of one single line (the command text) is selected and copied, not the full line (including the LineFeed) or multiple lines.



Many of PI's General Command Set compatible devices support single-byte commands. For example, the user can stop a fast scan of a C-880 or F-206 by sending an ASCII 24 (decimal). To enter this command into the Send window simply type a "#" followed by the decimal value of the byte to be sent, e.g. enter "#24" and presses ENTER to stop a fast scan. An entry "(Send: *)" will be added to the original command with * being the corresponding ASCII character of the single byte sent.

Pressing F1 or the Help button will pop up a help window. To return to the terminal application, press Esc. If Send ERR? on Exit? is TRUE, an “ERR?” query is sent to the device when Exit is pressed to prevent the controller from keeping an error condition produced during the use of the terminal application.

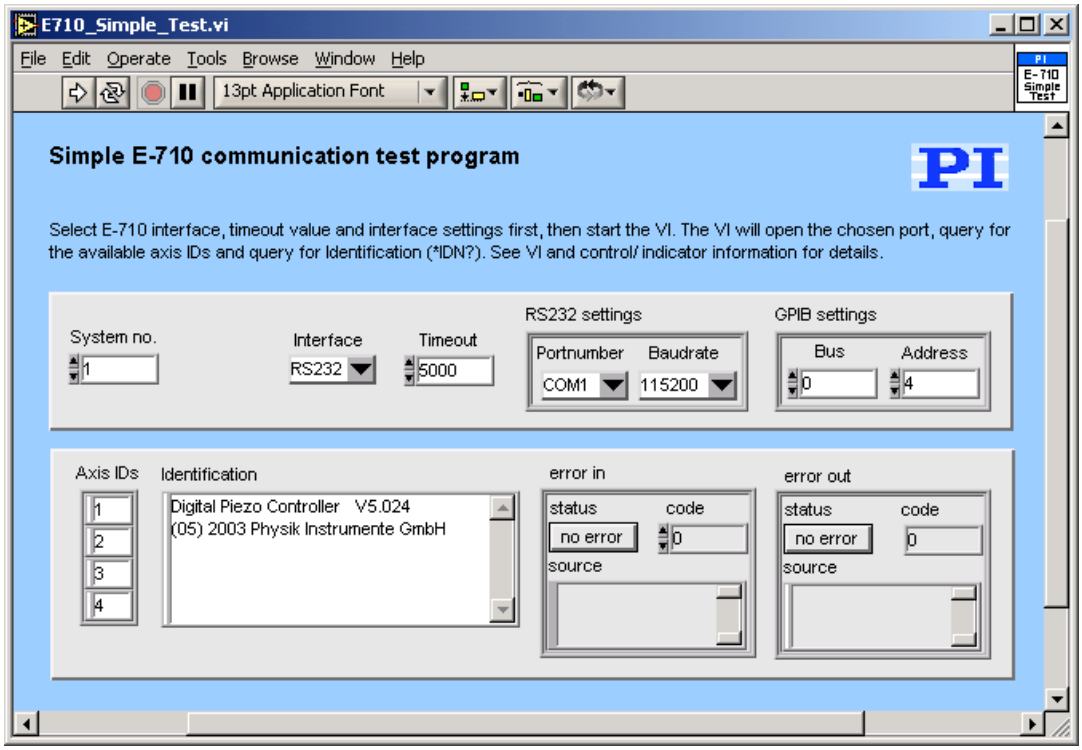
3.2. E710_Simple_Test.vi

This simple test VI is a stand-alone sample application. Use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

Specify

- System no. (= 1 in a one-system configuration),
- Interface (= RS232 or GPIB)
- Timeout value (in milliseconds)
- and interface settings first.

Then start the VI. The VI will open a connection to the E-710 and query the controller for its identification string and the available axis IDs. The diagram shows how to combine the driver and support VIs for these tasks.



3.3. E710_Configuration_Setup.vi

This VI performs a fully automatic initialization of the selected system (global settings and automatic zero-point calibration of the stages) in the LabVIEW environment. Use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

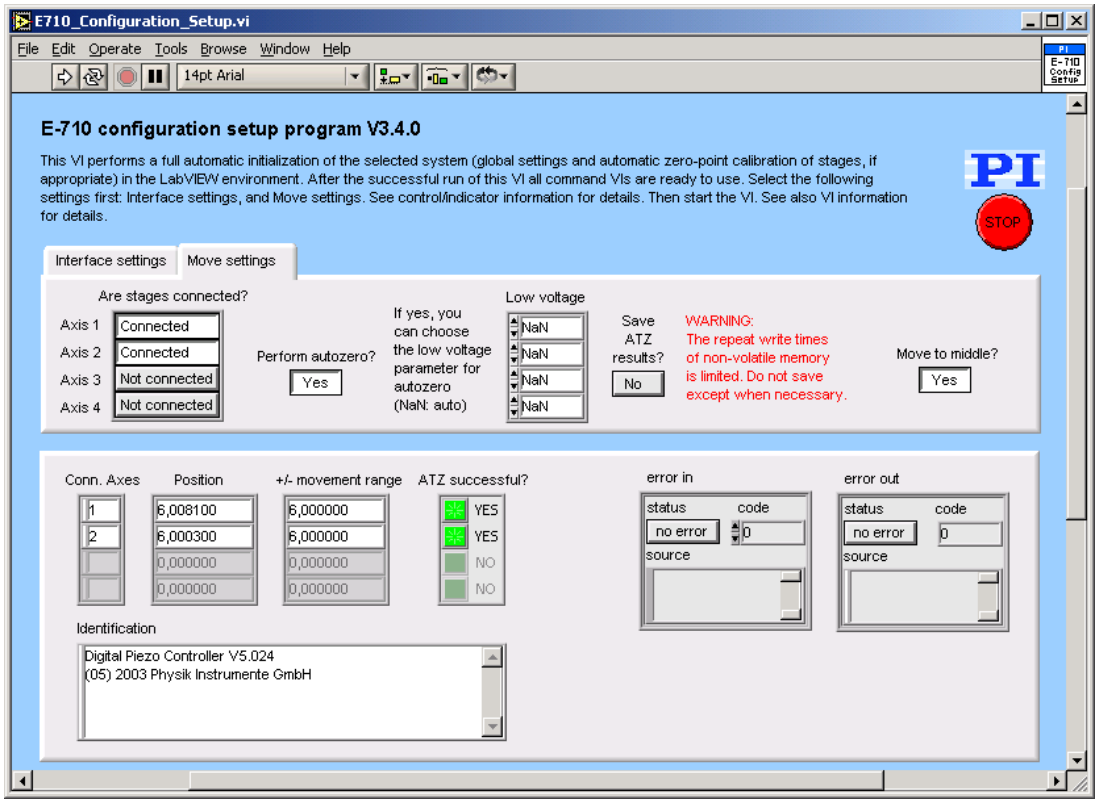
After the successful run of this VI, all command VIs are ready to use. Specify the correct parameters first:

- System No.: 1 in a one-system-only configuration
- Interface (= RS232 or GPIB)
- Timeout value (in milliseconds)
- Interface settings
- To which axes stages are connected
- Whether auto zero is to be performed
- If auto zero is to be performed, whether the low voltage parameter is to define manually (Low voltage = I32 value) or automatically (Low voltage = NaN)
- If all axes are to be moved to the middle position of their travel range.
- Whether the servo status of all axes is to change to TRUE. If FALSE is selected here, the servo status will not be changed.

Then start the VI.

"E710_Configuration_Setup.vi" performs the following initialization tasks:

1. Runs "PI Open Interface of one system.vi" to open a connection to the controller.
2. Runs "*IDN?.vi" to query for the controller identification string.
3. Defines the selected system to be "E-710"
4. Runs "SAI?.vi" to query for axis IDs of available axes.
5. Runs CST.vi to define axes with no stages connected as "NOSTAGE" and axes with stages connected as "ID-STAGE".
6. Runs "Define connected axes.vi" with Read from controller = FALSE and Connected axes = subset of available axes depending on Are stages connected?.
7. Runs "INI.vi"
8. Runs "ATZ.vi". if Perform autozero? is TRUE
9. Runs "POS?.vi"
10. Runs "ERR?.vi" to query the controller for its error status.
11. Reads the position range (TMN?, TMX?).
12. If Move to middle? is TRUE, moves each axis to the middle position of its range (MOV).
13. Runs "POS?.vi" to query for the position of all axes.
14. Runs "ERR?.vi" to query the controller for its error status.
15. Runs "GCSTranslateError.vi" to append the error message which corresponds with a GCS error number returned by "ERR?.vi" to Source from Error out.



Use this VI as the initialization VI for the E-710 in your application.

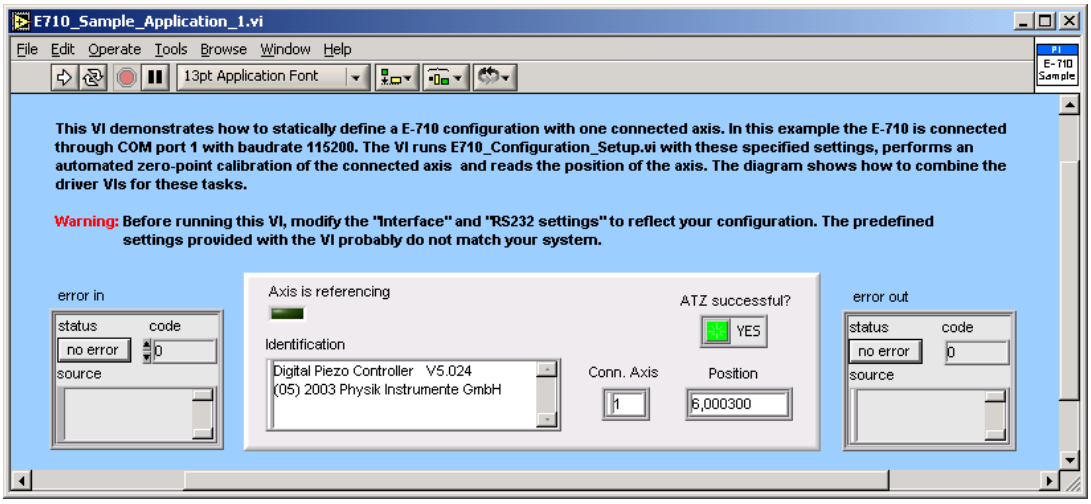
When using as a sub-VI, connect Stop refnum terminal to stop VI from caller.

As the initialization is a complex procedure which uses a large number of sub-VIs, E710_Configuration_Setup.vi is password-protected, meaning that you cannot see or modify the diagram. In this way, the full initialization is packed into one single and fully tested procedure which you simply insert into your own application program. For security reasons as well as your convenience, we recommend that you not modify this VI.

3.4. E710_Sample_Application_1.vi

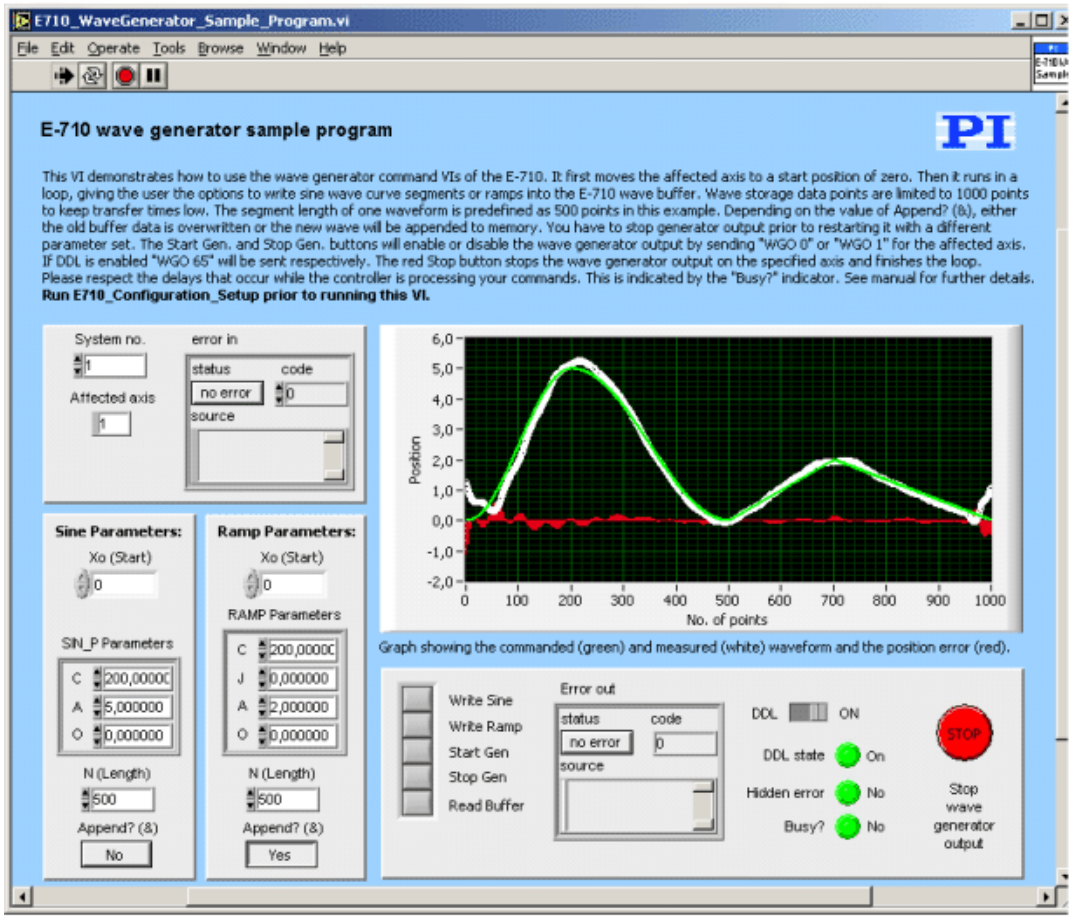
This VI demonstrates how to statically define an E-710 configuration with one connected axis. In this example the E-710 is connected through COM port 1 with baudrate 57600. The VI runs E710_Configuration_Setup.vi with these specified settings, performs an automated zero-point calibration of the connected axis and reads the position of the axis. The diagram shows how to combine the driver VIs for these tasks.

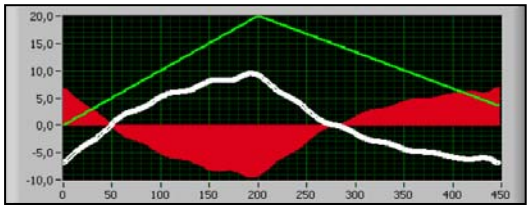
Warning: Before running this VI, modify the "Interface" and "RS232 settings" to reflect your configuration. The predefined settings provided with the VI probably do not match your system.



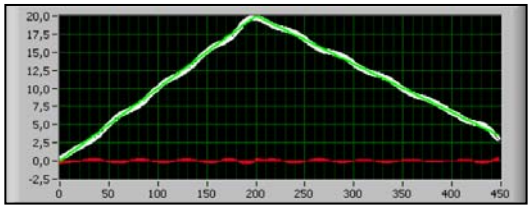
3.5. E710_WaveGenerator_Sample_Program.vi

This VI demonstrates how to use the wave generator command VI's of the E-710. It first moves the affected axis to a start position of zero. Then it runs in a loop, giving the user the options to write sine wave curve segments or ramps into the E-710 wave buffer. Wave storage data points are limited to 1000 points to keep transfer times low. The segment length of one waveform - which normally defines the frequency - is predefined as 500 points in this example. Depending on the value of Append? (&), either the old buffer data is overwritten or the new wave will be appended to memory. Please take into account that you can append at most two waveforms since buffer size is limited to 1000 points as mentioned above.





Position graph with DDL off



Position graph with DDL on

Adjust the following controls to modify the shape of the waveforms.

N (Length): Waveform (i.e. the points to be modified) goes from 0 to Length. Datapoints from Length to 500 are filled with zeros.

Xo (Start): Position where the first point of the waveform is located.

C: Center point of the wave. If C = Length/2 then the wave is symmetrical.

A: Amplitude of the waveform. F(x) = A*sin(x) + O

O: Offset of the waveform. F(x) = A*sin(x) + O

J: Jerk. Means curvature in the transition to or from standstill.

You have to stop generator output prior to restarting it with a different parameter set. Otherwise you get a Hidden error. The Start Gen. and Stop Gen. buttons will enable or disable the wave generator output by sending "WGO 0" or "WGO 1" for the affected axis. The red Stop button stops the wave generator output on the specified axis and finishes the loop.

Please respect the delays that occur while the controller is processing your commands. This is indicated by the Writing... indicator.

In case of an error a button Clear Error will appear and the control buttons above will be disabled. Pressing this button will clear the error status and enable the control buttons again.

With DDL ON you can achieve significantly better position accuracy. Compare the target (green), response (white) and error (red) signals on the graph. Since this is a very sensitive mechanism we recommend starting with low amplitudes and adjusting the parameters with caution. The DDL button works only if the controller supports DDL. Contact PI for information on activation of DDL support.

See Appendix B for details about the use of the E-710 wave generator.

Run E710_Configuration_Setup prior to running this VI.

3.6. Joystick Operation Sample Program.vi

This VI can be used to control 2 axes of a connected system with a standard 2-button, 2-axis joystick connected to the game port of the host computer. The absolute value of the joystick position is converted into velocity values for the two stages connected to the system being commanded. Two velocity levels for each axis can be specified, e.g. one for fast, rough positioning and one for slow, fine positioning. Joystick button 2 switches between these levels. The sign of the joystick position determines whether the move command issued contains the positive or negative travel limit (read automatically if Read travel range from controller? is TRUE) of the corresponding axis. When the joystick is "in the middle position", the velocity of the corresponding axis is set to zero.

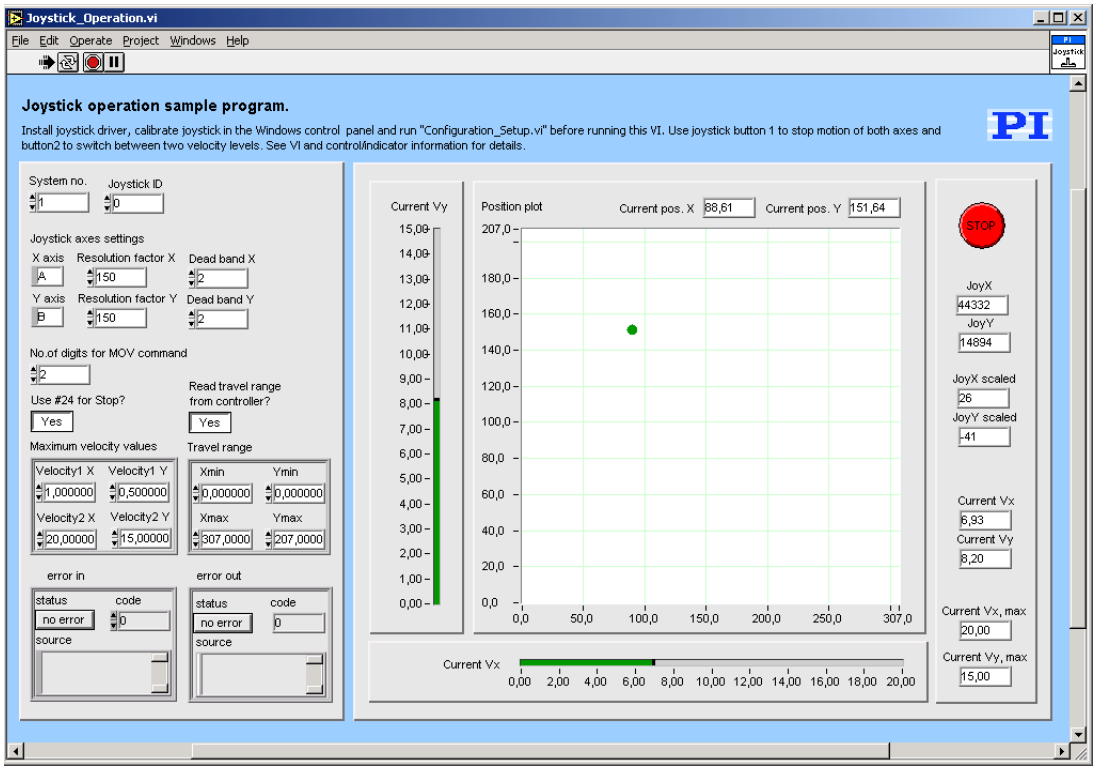
Dead band * is the maximum size of the scaled joystick position value that does not result in any motion.

Resolution factor * determines the joystick resolution.

No. of digits is the number of digits after the decimal point in the position values that will be sent by the MOV command.

If joystick button 1 is pressed, a stop command (STP or #24, depending on Use 24 for stop, is sent to the controller. The diagram shows how to combine the driver and support VIs for these tasks.

Important: Install joystick driver and calibrate joystick in the Windows Control Panel before running this VI.



Valid for	C-843, C-843.PM, C-844, C-848, C-865, C-880, E-710
Input	System number (1), Joystick ID (0), X axis (empty string), Y axis (empty string), Resolution factor X (150), Resolution factor Y (150), Dead band X (2), Dead band Y (2), No. of digits (2), Use #24 for stop (TRUE), Read from controller (T), Maximum velocity values (2, 20, 0.5, 15), Travel range (0, 10, 0, 5), Error in (no error)

C-843: Use #24 for stop = TRUE, Read from controller = TRUE;
Run “C843_Configuration_Setup.vi” prior to running this VI.

C-843.PM: Use #24 for stop = TRUE, Read from controller = TRUE;
Run “C843_PM_Configuration_Setup.vi” prior to running this VI.

C-844: Use #24 for stop = TRUE, Read from controller = TRUE
Run “C844_Configuration_Setup.vi” prior to running this VI.

C-848: Use #24 for stop = TRUE, Read from controller = TRUE
Run “C848_Configuration_Setup.vi” prior to running this VI.

C-865: Use #24 for stop = TRUE, Read from controller = TRUE;
Run “C865_Configuration_Setup.vi” prior to running this VI. Y Axis must be empty string.

C-880: Use #24 for stop = TRUE, Read from controller = TRUE
Run “C880_Configuration_Setup.vi” prior to running this VI.

E-710: Use #24 for stop = FALSE, Read from controller = TRUE.
Default position unit is µm, default velocity unit is µm/ms. Joystick button 1 is not supported (controller does not support #24 or STP).
Run “E710_Configuration_Setup.vi” prior to running this VI.

Use the *Help→Show Context Help* menu sequence in the LabVIEW environment to display the *Context Help* window with the VI and control/indicator descriptions.

4. PI Systems Currently Supported by This Driver Set

Product	works with LabVIEW driver version	if product firmware version is equal to or newer than
C-843	2.01 – 2.02	MC-DLL 1.0.2.2
	2.05 – 2.06	MC-DLL 1.0.2.3
	3.1.2., 3.1.2a	MC-DLL 1.0.2.3
	3.4.3	MC-DLL 1.0.2.8
C-843.PM	3.1.0	MC-DLL 1.0.2.5
C-848	3.0.2	1.0
C-865	3.3.0	MC_C865.dll 1.0
C-880	1.1	2.00
	1.2	2.10
	2.04	2.20
	2.05 – 2.06	2.21
	3.2.0	2.40
C-880K005	2.06	1.0
C-880K006	2.06	1.0
C-880K007	2.06	1.0
E-516	1.0 – 2.02	DSP V3.01, MCU V5
	2.05 – 2.06	DSP V3.11, MCU V5
	3.4.2	DSP V3.30, MCU V5
E-710 3- & 4-channel versions	3.4.0	5.027
	3.4.4 (a, b)	5.0.33, 6.0.33
E-710 6-channel	3.4.4 (a, b)	2.13
E-816	2.01 – 2.06	2.02
F-206	1.1 – 2.06	Fhx0035
M-840	2.03 – 2.06	Hex0037
	2.2.0	Hex0045
	3.0.1	Hex0050
	3.1.1	Hex0051
M-850	2.03 – 2.06	Hex0040
	3.0.1	Hex0050
	3.1.1	Hex0051

5. Appendix A

Error codes are not unambiguous, but can result from a PI error message or LabVIEW internal error code. In addition to the list below see National Instruments error codes.

100	PI LabVIEW driver reports error. See source control for details.
0	No error
1	Parameter syntax error
2	Unknown command
3	Command length out of limits or command buffer overrun
4	Error while scanning
5	Unallowable move attempted on unreferenced axis, or move attempted with servo off
6	Parameter for SGA not valid
7	Position out of limits
8	Velocity out of limits
9	Attempt to set pivot point while U,V and W not all 0
10	Controller was stopped
11	Parameter for SST or for one of the embedded scan algorithms out of range
12	Invalid axis combination for fast scan
13	Parameter for NAV out of range
14	Invalid analog channel
15	Invalid axis identifier
16	Unknown stage name
17	Parameter out of range
18	Invalid macro name
19	Error while recording macro
20	Macro not found
21	Axis has no brake
22	Axis identifier specified more than once
23	Illegal axis
24	Incorrect number of parameters
25	Invalid floating point number
26	Parameter missing

27	Soft limit out of range
28	No manual pad found
29	No more step response values
30	No step response values recorded
31	Axis has no reference sensor
32	Axis has no limit switch
33	No relay card installed
34	Command was not allowed for selected stage(s)
35	No digital input installed
36	No digital output configured
37	No more MCM responses
38	No MCM values recorded
39	Controller number invalid
40	No joystick configured
41	Invalid axis for electronic gearing, axis can not be slave
42	Position of slave axis is out of range
43	Slave axis cannot be commanded directly when electronic gearing is enabled
44	Calibration of joystick failed
45	Referencing failed
46	OPM (Optical Power Meter) missing
47	OPM (Optical Power Meter) not initialized or cannot be initialized
48	OPM (Optical Power Meter) Communication Error
49	Move to limit switch failed
50	Attempt to reference axis with referencing disabled
51	Selected axis is controlled by joystick
52	Controller detected communication error
53	MOV! motion still in progress
54	Unknown parameter
55	No commands were recorded with REP
100	PI LabVIEW driver reports error. See source control for details.
200	No stage connected
201	File with axis parameters not found
202	Invalid axis parameter file
203	Backup file with axis parameters not found
204	PI internal error code 204

205	SMO with servo on
206	uudecode: incomplete header
207	uudecode: nothing to decode
208	uudecode: illegal UUE format
209	CRC32 error
210	Illegal file name (must be 8-0 format)
211	File not found on controller
212	Error writing file on controller
213	VEL command not allowed in DTR Command Mode
214	Position calculations failed
301	Send buffer overflow
302	Voltage out of limits
303	Attempt to set voltage when servo on
304	Received command is too long
305	Error while reading/writing EEPROM
306	Error on I2C bus
307	Timeout while receiving command
308	A lengthy operation has not finished in the expected time
309	Insufficient space to store macro
310	Configuration data has old version number
311	Invalid configuration data
333	Internal hardware error
555	BasMac: unknown controller error
1000	Too many nested macros
1001	Macro already defined
1002	Macro recording not activated
1003	Invalid parameter for MAC
1004	PI internal error code 1004
2000	Controller already has a serial number
4000	Sector erase failed
4001	Flash program failed
4002	Flash read failed
4003	HW match code missing/invalid
4004	FW match code missing/invalid
4005	HW version missing/invalid
4006	FW version missing/invalid

-1	Error during com operation (could not be specified)
-2	Error while sending data
-3	Error while receiving data
-4	Not connected (no port with given ID open)
-5	Buffer overflow
-6	Error while opening port
-7	Timeout error
-8	There are more lines waiting in buffer
-9	There is no interface open with the given ID
-10	The event for the notification could not be opened
-11	The function was not implemented (e.g. only RS-232 communication provides this feature and it was called for IEEE488)
-12	Error while sending echoed data
-13	IEEE488: System error
-14	IEEE488: Function requires GPIB board to be CIC
-15	IEEE488: Write function detected no listeners
-16	IEEE488: Interface board not addressed correctly
-17	IEEE488: Invalid argument to function call
-18	IEEE488: Function requires GPIB board to be SAC
-19	IEEE488: I/O operation aborted
-20	IEEE488: Non-existent interface board
-21	IEEE488: Error performing DMA
-22	IEEE488: I/O operation started before previous operation completed
-23	IEEE488: No capability for intended operation
-24	IEEE488: File system operation error
-25	IEEE488: Command error during device call
-26	IEEE488: Serial poll status byte lost
-27	IEEE488: SRQ remains asserted
-28	IEEE488: Return buffer full
-29	IEEE488: Address or board locked
-30	RS-232: Use of 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	RS-232: Error configuring the COM port
-32	Error dealing with internal system resources (events, threads, ...)
-33	A DLL or one of the required functions could not be loaded

-34	FTDIUSB: invalid handle
-35	FTDIUSB: device not found
-36	FTDIUSB: device not opened
-37	FTDIUSB: IO error
-38	FTDIUSB: insufficient resources
-39	FTDIUSB: invalid parameter
-40	FTDIUSB: invalid baud rate
-41	FTDIUSB: device not opened for erase
-42	FTDIUSB: device not opened for write
-43	FTDIUSB: failed to write device
-44	FTDIUSB: EEPROM read failed
-45	FTDIUSB: EEPROM write failed
-46	FTDIUSB: EEPROM erase failed
-47	FTDIUSB: EEPROM not present
-48	FTDIUSB: EEPROM not programmed
-49	FTDIUSB: invalid arguments
-50	FTDIUSB: not supported
-51	FTDIUSB: other error
-1001	Unknown axis identifier
-1002	Number for NAV out of range--must be in [1,10000]
-1003	Invalid value for SGA--must be one of 1, 10, 100, 1000
-1004	Controller sent unexpected response
-1005	No manual control pad installed, calls to SMA and related commands are not allowed
-1006	Invalid number for manual control pad knob
-1007	Axis not currently controlled by a manual control pad
-1008	Controller is busy with some lengthy operation (e.g. reference move, fast scan algorithm)
-1009	Internal error--could not start thread
-1010	Controller is (already) in macro mode--command not valid in macro mode
-1011	Controller not in macro mode--command not valid unless macro mode active
-1012	Could not open file to write or read macro
-1013	No macro with given name on controller or macro is empty
-1014	Internal error in macro editor
-1015	One of the arguments given to the function is invalid (empty string, index out of range, ...)

-1016	Axis identifier is already in use for a connected stage
-1017	Invalid axis identifier
-1018	Could not access array data in COM server
-1019	Range of array does not fit the number of parameters
-1020	Parameter ID given to SPA or SPA? is not valid
-1021	Number for AVG out of range--must be >0
-1022	Number of samples given to WAV out of range
-1023	Generation of wave failed
-1024	Motion error while axis in motion
-1025	Controller is (already) running a macro
-1026	Configuration of PZT stage or amplifier failed
-1027	Current settings are not valid for desired configuration
-1028	Unknown channel identifier
-1029	Error while reading/writing wave generator parameter file
-1030	Could not find description of wave form. Maybe WG.INI is missing?
-1031	The WGWaveEditor DLL function was not found at startup
-1032	The user cancelled a dialog
-1033	Error from C-844 Controller
-1034	DLL necessary to call function not loaded, or function not found in DLL
-1035	The open parameter file is protected and cannot be edited
-1036	There is no parameter file open
-1037	The selected stage does not exist
-1038	There is already a parameter file open. Please close this file before openig a new file
-1039	DLL required to call function not loaded, or function not found in DLL
-1040	The version of the connected controller is invalid
-1041	Parameter could not be set with SPA--parameter on controller undefined!
-1042	The maximum number of wave definitions has been exceeded
-1043	The maximum number of wave generators has been exceeded
-1044	There is no wave defined for specified axis
-1045	Wave output to axis already stopped/started
-1046	Not all axes could be referenced
-1047	Could not find parameter set required by frequency relation
-1048	Command ID given to SPP or SPP? is not valid
-1049	A stage name given to CST is not unique
-1050	A unencoded file transferred did not start with begin followed by the

	proper filename
-1051	Could not create/read file on host PC
-1052	Checksum error when transfering a file to/from the controller
-1053	The PiStages.dat database could not be found. This file is required to connect a stage with the CST command
-1054	No wave being output to specified axis
-1055	Invalid password
-1056	Error during communication with OPM (Optical Power Meter), maybe no OPM connected
-1057	WaveEditor: Error during creation of wave, wrong number of parameters
-1058	WaveEditor: Frequency out of range
-1059	WaveEditor: Error during creation of wave, wrong index for integer parameter
-1060	WaveEditor: Error during creation of wave, wrong index for floating point parameter
-1061	WaveEditor: Error during creation of wave, could not calculate value
-1062	WaveEditor: Component to display graph is not installed

6. Appendix B

6.1. E-710 Wave Generator and DDL

6.1.1. Wave Generator Basics

With the E-710, it is possible to create arbitrary waveforms and to output them for up to two axes (E-710 is equipped with two wave generators).

The waveforms are stored in wave tables. Each axis has its own wave table. To address a wave table, the appropriate axis ID must be used (e.g. if you want to address the wave table of axis Z, the wave table ID must be Z). The total number of points available for all wave tables is 63488 (62464 with 6-axis versions). These points can be flexible assigned to the wave tables (see WMS.vi), but due to memory restrictions, the number of tables to which the points are distributed must not exceed 4 (3 with E-710 3-axis versions). The assignment is valid until the E-710 is powered down or WMS.vi is called again.

Waveforms can be created based on predefined "curve" shapes. Additionally you can freely define curve shapes. The waveform can be made up by concatenating a number of “segments”. See the WAV.vi functions for more information. The waveforms are stored in the wave tables until the E-710 is powered down or the number of points per table is reset. Even if WMS.vi is called for only one wave table, the waveforms should be defined again for all tables.

A waveform can be output a fixed number of times, or repeated indefinitely (see WGO.vi). When the wave generator output is stopped and restarted, it will continue with the first point of the waveform. The output values will always be interpreted as positions, so that the servo must be on during wave generator operation. Note that these target values are relative positions, i.e. they are added to any other current target contributions coming from move commands and / or from the analog input (for details see E-710 User Manual). To address a wave generator, you have to use the ID of the axis for which the generator shall be started (e.g. if you want to start wave generator output for axis Z, the wave generator ID must be Z). Wave generator output will continue even if LabVIEW or the program from which it was started is quit.

Dynamic Digital Linearization (DDL) is standard on 6-axis E-710s and available as an option on 3- and 4-axis units. It can be used to reduce residual tracking error in dynamic applications, i.e. while the wave generators run. Using DDL involves gathering data in tables in the controller during an initialization phase and then applying that data during subsequent wave generator operation. See WGO.vi, DDL.vi, DDL?.vi and DTC.vi.

Each time a wave generator is started, data recording starts automatically for the corresponding axis (read the data with DRR?.vi). The data for the individual axes is written to separate record tables. The record configuration can be done with DRC.vi. Recording ends when the record table content has reached the maximum number of points (8192 per table). Recording can be restarted with WGR.vi.

6.1.2. How to Use the Wave Generator

NOTES

Be sure that you have set correct waveform sequence before enabling wave output to avoid unpredictable stage response, such as overflow and vibration.

Using the wave generators is as follows (See also E710_WaveGenerator_Sample_Program.vi):

1. Set the maximum number of wave points for the wave tables with WMS.vi.
2. Define the waveform using the WAV.vi control inputs (if necessary create the waveform by concatenating multiple segments).
3. Optionally: Check the waveform:
   After you sent the waveform definition to the wave table, it is always a good idea to check it by reading back the waveform sequence from the E-710 before actually outputting it. This can be done with GWD?.vi.
4. Optionally: If you want to change the table rate for wave generator and data recording, set parameter 0x13000109 using SPA.vi. This parameter is available in command level 1 (see CCL.vi) and can be set in RAM only (not in EEPROM).
5. Optionally: If you want to output trigger signals during the wave generator output, configure the trigger lines with TWS.vi.
6. Set servo on with SVO.vi for the axes for which you want to start the wave generator output. Wave generator output is only possible in closed-loop operation because the wave points are interpreted as target positions (but not as voltages as it would be required for open-loop operation).
7. Start the wave generator output and hence the motion of the axis with WGO.vi. Different start modes can be set with WGO.vi separately for each wave generator (= axis), for example, DDL initialization or DDL usage during the wave generator output.
   When starting the wave generator, recording is started automatically, and the data can be read with DRR?.vi.
8. Optionally: Restart recording with WGR.vi.
9. Stop the wave generator output with WGO.vi.

You can check the wave generator activation status with #9.vi.

6.1.3. Examples: Wave Generation and Output

The following examples show the wave generator usage based on the control inputs of the VIs of this GCS LabVIEW driver set.

General:

In this example, axis 1 was renamed to X to illustrate the interrelation between axis ID and wave generator ID / wave table ID.

Command	Comment
WMS X 3000	Set the maximum number of points for the wave table belonging to axis X to 3000
WAV X ….	Define a waveform for wave table X (axis X)
SVO X 1	Servo is switched ON for axis X
WGO X 1	Start output of wave generator X (axis X) immediately (synchronized by interrupt)
WGO X 0	Stop output of wave generator X (axis X)

Waveform examples:

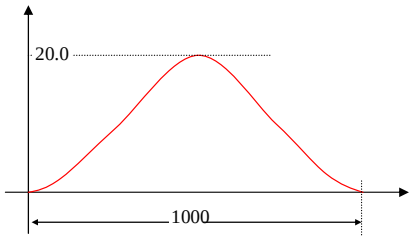
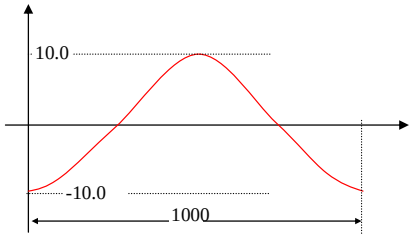
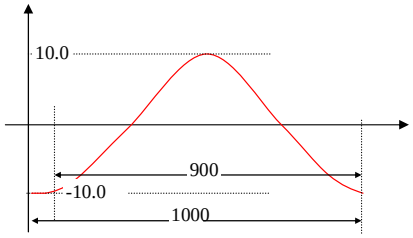
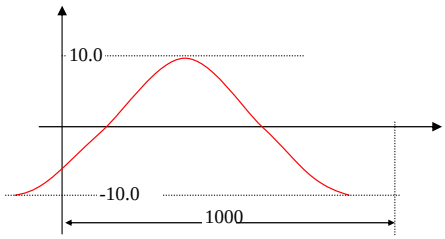
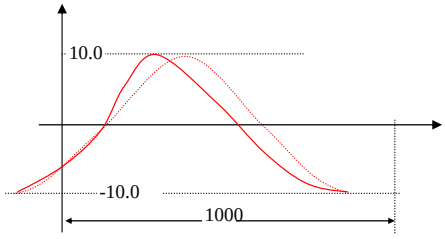
The examples below are all construed for axis 1, and the table rate is 1 (default).

NOTES

The table rate can be changed via the parameter 0x13000109 (only in RAM)—this parameter sets the number of servo-loop cycles to be used for wave generator and data recording

operations. Settings other than 1 make it possible to cover longer time periods with a limited number of points. By default, the servo update time is 200 µs, i.e. 5000 points = 1 s.

The offset in the waveform (O (=*dOffsetOfWave)* set by WAV.vi) is ignored in the following cases:
- for the first segment of a waveform which consists of multiple segments
- when a waveform replaces the previous wave table content (Append? (=*iAddAppendWave)* = 0).

Description	Commands	Wave Form
Sine wave: 5 Hz, Amp_{p-p} = 20 µm	WAV 1 SIN_P 1000 20.0 WGO 1 1	
Sine wave: 5 Hz, Amp_{p-p} = 20 µm Offset = -10 µm (O (=*dOffsetOfWave))*	WAV 1 SIN_P 1000 20.0 -10.0 WGO 1 1	
Sine wave: 5 Hz, Amp_{p-p} = 20 µm Offset = -10 µm, (O (=*dOffsetOfWave))* 100 zeros (1-100)	WAV 1 SIN_P 1000 20.0 -10.0 900 WGO 1 1	
Sine wave: 5 Hz, Amp_{p-p} = 20 µm Offset = -10 µm, (O (=*dOffsetOfWave))* Phase shift 72°	WAV 1 SIN_P 1000 20.0 -10.0 1000 -200 WGO 1 1	
Sine wave: 5 Hz, Amp_{p-p} = 20 µm Offset = -10 µm, (O (=*dOffsetOfWave))* Phase shift 72° Type: asymmetrical	WAV 1 SIN_P 1000 20.0 -10.0 1000 -200 100 WGO 1 1	

7. Index

End of document