

MS138E User Manual

C-663 Mercury™ Step

Stepper Motor Controller

Release: 1.3.0 Date: 2009-08-10



This document describes the following product:

- **C-663.10**
Stepper Motor Controller, Single-Axis,
Networkable



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

Physik Instrumente (PI) GmbH & Co. KG is the owner of the following company names and trademarks:
PI®, PIC®, PICMA®, PLine®, PIFOC®, PiezoWalk®, NEXACT®, NEXLINE®, NanoCube®, NanoAutomation®

The following designations are protected company names or registered trademarks of third parties:
Microsoft, Windows, LabView

Copyright 2009 by Physik Instrumente (PI) GmbH & Co. KG, Karlsruhe, Germany.
The text, photographs and drawings in this manual enjoy copyright protection. With regard thereto, Physik Instrumente (PI) GmbH & Co. KG reserves all rights. Use of said text, photographs and drawings is permitted only in part and only upon citation of the source.

First printing 2009-08-10
Document Number MS138E, Bro, Release 1.3.0
C-663_User_MS138E.doc

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 This manual is superseded by any new release.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

Declaration of Conformity

according to DIN EN ISO/IEC 17050:2005-01

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

The manufacturer hereby declares that the product

Product Name: **Stepper Motor Controller**

Model Numbers: **C-663**

Product Options: **all**

complies with the following European directives:

2006/95/EC, Low-voltage directive (LVD)

2004/108/EC, EMC Directive

The applied standards certifying the conformity are listed below.

Electromagnetic Emission: EN 61000-6-3, EN 55011 (05/98)+
A1(08/99) class A,
EN 55022 (09/98)

Electromagnetic Immunity: EN 61000-6-1

Safety (Low Voltage Directive): EN 61010-1

25 June 2009
Karlsruhe, Germany



Dr. Karl Spanner
President

About This Document

Users of This Manual

This manual is designed to help the reader to install and operate the C-663 Stepper Motor Controller. It assumes that the reader has a fundamental understanding of motion control concepts and applicable safety procedures.

The manual describes the physical specifications and dimensions of the C-663 Stepper Motor Controller as well as the software and hardware installation procedures which are required to put the associated motion system into operation.

This document is available as PDF file on the product CD. Updated releases are available for download from www.pi.ws or by email: contact your Physik Instrumente Sales Engineer or write info@pi.ws.

Conventions

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

WARNING

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



CAUTION

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



NOTE

Provides additional information or application hints.

Related Documents

The software tools and any stages which might be delivered with C-663 Stepper Motor Controller, are described in their own manuals. All documents are available as PDF files via download from the PI Website (www.pi.ws) or on the included product CD. For updated releases or other versions contact your Physik Instrumente Sales Engineer or write info@pi.ws.

MMCRUN MS139E	Mercury™ Operating Software (native commands)
Mercury™ Native DLL & LabVIEW MS177E	Windows DLL Library and LabView VIs (native-command-based)
Mercury™ Native Commands MS176E	Native Mercury™ Commands
Mercury™ GCSLabVIEW_MS149E	LabView VIs based on PI GCS command set
Mercury™ GCS DLL_MS154E	Windows DLL Library (GCS commands)
PIMikroMove™ User Manual SM148E	PIMikroMove™ Operating Software (GCS-based)
Mercury™ GCS Commands MS163E	Mercury™ GCS Commands
PIStageEditor _SM144E	Software for managing GCS stage-data database
GCS Data User SM146E	GCS array data format description

1	Introduction	3
1.1	Overview.....	3
1.2	Prescribed Use.....	4
1.3	Safety Precautions.....	5
1.4	Unpacking.....	6
1.5	Accessories.....	7
1.6	Software Interfaces.....	7
1.6.1	Native Command Set.....	8
1.6.2	GCS Command Set.....	8
1.6.3	Available Software.....	8
2	Quick Start	10
2.1	Software Installation.....	10
2.1.1	GCS Software.....	10
2.1.2	Native-Command Software.....	11
2.2	Starting Operation.....	11
3	Details of Operation	16
3.1	Operating Elements.....	16
3.1.1	Front Panel Elements.....	16
3.1.2	DIP Switch Settings.....	17
3.1.3	Rear Panel Elements.....	18
3.2	Supply Power Connection.....	18
3.3	Power-On Settings.....	19
3.4	Connecting Mercurys™ with the Host PC.....	20
3.4.1	RS-232 from Host.....	20
3.4.2	USB from Host.....	20
3.4.3	Networking.....	21
3.5	Limit Signals.....	22
3.5.1	Limit Switches.....	22
3.5.2	Soft Limits.....	22
3.6	Reference Signal.....	23
3.7	Input/Output Lines.....	23
3.7.1	Overview.....	23
3.7.2	Trigger Output.....	23
4	Updates	24
4.1	Firmware Update.....	24
4.2	Software Updates.....	27

5	Manual Control	28
5.1	Joystick.....	28
5.1.1	Joystick Control Accessories.....	28
5.1.2	Joystick Response Tables.....	29
5.1.3	Z-Axis Calibration with C-819.30 Joystick.....	29
5.2	Pushbutton Control.....	31
5.3	Trackball Control	31
6	Macros	32
7	Troubleshooting	33
8	Technical Data	36
8.1	Specifications	36
8.2	Dimensions.....	37
8.3	Connector Pinouts	38
8.3.1	Motor-Connector.....	38
8.3.2	I/O Connector	39
8.3.3	Joystick Connector	39
8.3.4	Joystick Y-cable.....	39
8.3.5	RS-232 Connectors	40
8.3.6	USB Connector.....	41
8.3.7	Power Connector & Grounding Screw	41
9	Disposal	42

1 Introduction

The Mercury™ Step stepper motor controller is the perfect solution for cost-effective and flexible motion control applications where a precision positioner is to be controlled by a PC or PLC (programmable logic controller). The C-663 supplements the successful Mercury™ DC motor servo-controllers. These products are Mercury™ Class controllers and as such share the same command sets and are internetworkable.

1.1 Overview

- RS-232 and USB Host Communication
- Stand-Alone Capability
- Network Capability for Multi-Axis Applications
- Compatible and Networkable with all other Mercury™ Class controllers
- Joystick Port for Manual Control
- Non-Volatile Macro Memory
- Parameter Changes on-the-fly
- TTL Inputs for Limit & Origin Switches
- Motor-Brake Control
- Programmable I/O Lines



Fig. 1: C-663.10 Mercury™ Step Controller

Microstepping of 1/16 full step (for up to 6400 steps/rev. with PI stepper motors) provides ultra-smooth, high-resolution motion.

Multi-Axis Control, Combination of DC & Stepper Motors

The networking feature allows the user to start out with one Mercury™ controller and add more units later for multiaxis setups.

The Mercury™ Step stepper motor controller shares its native programming language with the well-established Mercury™ DC-motor controllers. Up to 16 Mercury™ controllers (DC and stepper) can be daisy chained and operated from one computer interface.

Flexible Automation

The C-663 offers a number of features for performing automation and handling tasks in research and industry in a very cost-effective way. Programming is facilitated by the high-level mnemonic command language with macro and compound-command functionality. Macros can be stored in the non-volatile memory for later recall.

Stand-alone capability is provided by a user-programmable autostart macro to run automation tasks at power up (no run-time computer communication required!).

For easy synchronization of motion with internal or external trigger signals, four input and four output lines are provided. A joystick can also be connected for manual control.

User-Friendly: Comprehensive Software Package and Two Interface Options

Easy data interchange with laptop or PC is possible via the USB interface. To facilitate industrial applications, an RS-232 interface is also standard.

The included software supports networking of multiple controller devices. LabVIEW™ drivers and Windows DLLs allow for easy programming and integration into your system. Mercury™ Step controllers can also be operated using the PI General Command Set (GCS) via a DLL. PI-GCS allows interoperation of different PI controllers such as piezo drivers and multi-axis servo-controllers with minimal programming effort.

1.2 Prescribed Use

Based on its design and realization, the C-663 Mercury™ Step Controller is intended to drive PI stages with 2-phase stepper motors.

Observe the safety precautions given in this User Manual.

The C-663 may only be used for applications suitable according to the device specifications. Operation other than instructed in this User Manual may affect the safeguards provided.

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

The C-663 is a laboratory apparatus as defined by DIN EN 61010. It meets the following minimum specifications for safe operation. More stringent conditions given in the specifications table on p. 36 are, of course, also met.

- Indoor use
- Altitude up to 2000 m
- Temperature range 5°C to 40°C

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

1.3 Safety Precautions

Install and operate the C-663 controller only after you have read the operating instructions. Keep the manuals and any relevant Technical Notes readily available. If lost or damaged, new copies can be downloaded from www.pi.ws or obtained from PI. Read carefully the documentation of the included software components and of the mechanics used also.

Failure to heed warnings in this manual can lead to accidents causing bodily injury, damage to equipment or loss of warranty protection. Note that the C-663 does not contain any user-serviceable parts.



WARNING

All motion of the connected motors and mechanical stages is software controlled, and software may fail. Be aware that motorized stages generate large forces that may cause personal injury or other damage if mishandled.

CAUTION

■ If operating different Mercury™ controllers, do not mix up the 12 V, 15 V and 24 V power supplies!

CAUTION

■ Never connect a DC-motor drive to a C-663 Stepper Motor controller. Irreparable damage could result.

CAUTION

Commanding a velocity above the maximum possible for the stage will cause the motor to stall. Because stepper motors do not have position encoders, the position counter will continue to increment (it counts motor steps). The controller's motor position may not correspond with the actual motor position and this might endanger your application.

The maximum velocity depends on various influences like operating voltage, phase current setting and mechanical load. Datasheet values are for orientation only and may not work under all conditions.

Check out the maximum possible velocity for your individual application!

CAUTION

Do not enable a joystick axis here when no joystick is connected to the controller hardware. Otherwise the corresponding controller axis may start moving and could damage your application setup.

CAUTION

Never connect the RS-232-IN and USB connectors of the same controller to a PC at the same time, as damage may result.

CAUTION

Because the unit is not grounded over the power supply, a grounding screw is provided at the lower left corner of the rear panel for connecting the metal case to a protective ground.

1.4 Unpacking

Unpack the C-663 Mercury™ Step controller with care. Compare the contents against the items covered by the order and against the packing list.

The C-663.10 includes the following components:

- Mercury™ Step Controller
- Wide-range, 24 V power supply (C-663.PS)
- Power-supply line-voltage cable

- RS-232 null-modem cable for PC connection (C-815.34, 3 m)
- RS-232 straight-through networking cable (C-862.CN, 28 cm)
- USB cable (type A to mini-B) for PC connection (000014651)
- Mercury™ Product CD with all software and manuals for Mercury™ Class products (C-663.CD)
- MS138E User Manual (this document) in printed form
- MS163E Mercury™ GCS Commands Manual in printed form
- MS176E Mercury™ Native Commands Manual in printed form

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

1.5 Accessories

The following items are not included but can be ordered separately:

Order Number

C-815.38	Stage/motor cable, 3 m (sub-D 15 m/f)
C-862.CN2	Long straight-through networking cable for interconnecting Mercury™ Class controllers, 180 cm
C-819.20	Analog joystick, 2 axes
C-819.20Y	Y-cable for connecting 2 Mercurys™ to joystick
C-819.30	3-axis analog joystick for Mercury™ controller
C-170.PB	Pushbutton box with 4 buttons and 4 LEDs
C-170.IO	Connector for I/O socket (p. 39), with cable, open end
C-663.PS	24 V power supply

1.6 Software Interfaces

The C-663 Mercury™ shares many features and commands with other Mercury™ class controllers from PI. It is possible to use either the Mercury™ native ASCII command set or the PI General Command Set (GCS) to operate Mercury™ Class controllers. The commands are used to set operating modes, transfer motion parameters and to query system and motion values.

1.6.1 Native Command Set

The native ASCII command set is understood by the controller directly. It can be used with virtually any terminal-emulator software and with *MMCRun*. Most native Mercury™ commands begin with a two-letter mnemonic. The syntax of the native commands and the detailed command descriptions can be found in the Mercury™ Class Native Commands software manual, MS176E.

1.6.2 GCS Command Set

GCS (General Command Set) is the PI standard command set, offering compatibility between different controllers. With current Mercury™ firmware, GCS command support is implemented by a Windows DLL which translates the GCS commands to the native commands (for details, see the Mercury™ GCS DLL manual, MS154E, and the Mercury™ GCS Commands manual, MS163E). Most GCS commands and the corresponding DLL function calls are characterized by a three-letter mnemonic.

NOTES

Although the GCS DLL has a gateway for sending native commands directly, mixing native and GCS commands is not recommended. GCS move commands, for example, may not work properly after the position is changed by a native command.

GCS commands can be typed directly into the *Command entry* window of GCS-compatible PI user-interface software, like *PIMikroMove*™ (see the *PIMikroMove*™ manual on the product CD).

1.6.3 Available Software

With the C-663 DC Motor Controller, all motion of the connected motors and mechanical stages is programmed or controlled by software. To offer maximum flexibility, software interfaces at a number of different levels are provided and documented. Most of the individual programs and driver libraries are described in separate manuals. Updated releases are available on www.pi.ws or via email: contact your PI Sales Engineer or write info@pi.ws.

- *PITerminal* is a Windows program which can be used as a simple terminal with almost all PI controllers. It supports both direct and via-GCS-DLL connection to Mercury™ controllers via RS-232 and USB (the USB link also looks like a COM port to host software when the USB drivers are installed). When the GCS-DLL connect option is used, command entered in the Terminal window are actually sent to the GCS-DLL, where they are interpreted as GCS commands. When the *Connect* button is used instead, current Mercury™ firmware expects commands from the native command set..

Native-Command-Set Software

- *MMCRun* (operating software for Windows™ 95/98/2000/XP and NT) is an operating software for C-863 and C-862 Mercury™, and C-663 Mercury™ Step controllers. *MMCRun* allows immediate operation of the motion system. It features easy commanding and macro programming of Mercury™ class controllers. See the *MMCRun Manual, MS139E*, for a full description.
- *MMC410.DLL* Windows DLL facilitates many interfacing operations and data conversion tasks for programmers of custom applications. This DLL is based on the native command set. (See the *Mercury Native DLL and LabView Manual, MS177*, for details)
- *LabVIEW VIs*: facilitate integrating Mercury™ class controllers in the LabVIEW environment. Uses the native-command *MMC410.DLL* above. (See the *Mercury Native DLL and LabView Manual, MS177*, for details)
- *MMC_Z-Joy* is a software tool required to calibrate the Z-axis of a C-819.30 3-axis analog joystick. It calculates a special response table and saves it to the corresponding controller

GCS-Based Software**

- *PIMikroMove™* (application for Microsoft Windows platforms) is operating software for this and many other PI controllers. With *PIMikroMove™* you can start your motion system—host PC, controller and stage(s)—immediately without writing customized software. *PIMikroMove™* offers motion control displays and features that in many cases make it unnecessary to deal with ASCII command formats. It also has a complete command input facility, which lets you experiment with various GCS commands easily. *PIMikroMove™* uses the GCS DLL described below to command the controller or controller network.
- GCS LabVIEW drivers to communicate with the C-663 from the National Instruments' LabVIEW environment (not included) using the Mercury™ GCS-DLL (see *MS149E* for details).
- GCS DLL (Windows DLL Library): The Mercury™ General Command Set Dynamic Link Library (Mercury™ GCS DLL) is an intermediate layer providing easy access to the controller from Windows programs. The use of the DLL and the functions it contains is described in a separate manual (*MS154E*). Most of the DLL functions correspond directly with the commands of the PI General Command Set.

NOTE

The GCS LabVIEW drivers and the GCS library are also available for Linux operating systems (kernel 2.6, GTK 2.0, glibc 2.4).

** Software is based on the PI General Command Set and requires the Mercury GCS DLL to translate the GCS commands into commands that can be understood by Mercury™ Class native-command firmware.

2 Quick Start

2.1 Software Installation

It is most convenient to test the system with PI software, even if custom software is planned later. *PIMikroMove™*, *MMCRun* or *PITerminal* can be used (on the product CD). For more information on these programs see Section 1.6, p. 7.

This section describes the procedures necessary to install the various programs.


NOTE

The PI host software is improved continually. It is therefore recommended that you visit the PI website (www.pi.ws) regularly to see if updated releases of the software are available for download. Updates are accompanied by information (readme files) so that you can decide if updating makes sense for your application. You need a password to see if updates are available and to download them. This password is provided on the Mercury™ CD in the Mercury Releasenews PDF file in the \Manuals directory. See "Software Updates" (p. 27) for download details.

2.1.1 GCS Software

If you wish to use any of the GCS-based software, proceed as follows.

Windows operating systems:

- 1 Insert the Mercury™ CD in your host PC
- 2 If the Setup Wizard does not open automatically, start it from the root directory of the CD with the  icon
- 3 Follow the on-screen instructions and select the "typical" installation. Typical components are LabView drivers, GCS DLL, PIMikroMove™

Linux operating systems:

- 1 Insert the Mercury™ CD in the host PC
- 2 Open a terminal and go to the /linux directory on the Mercury™ CD
- 3 Log in as superuser (root)
- 4 Start the install script with `./INSTALL`
Keep in mind the case sensitivity of Linux when typing the command
- 5 Follow the on-screen instructions. You can choose the individual components to install

If the installation fails, make sure you have installed the kernel header files for your kernel.

The *PIStages2.dat* stage database file needed by the GCS-based host software is installed in the *...\PI\GcsTranslator* directory. In that directory, also the *MercuryUserStages2.dat* database will be located which is created automatically the first time you connect stages in the host software (i.e. the first time the VST? or CST functions of the GCS library are used).

The location of the PI directory is that specified upon installation, usually in *C:\Documents and Settings\All Users\Application Data (Windows XP)* or *C:\ProgramData (Windows Vista)*. If this directory does not exist, the EXE file that needs the stage databases will look in its own directory.

2.1.2 Native-Command Software

The Windows Setup routine can also be used to install the native-command-based software. Setup puts it in subdirectories of the installation directory and adds the *MMCRun* and the Mercury™ firmware update programs to the *PI* program group under *Start→Programs→PI→Mercury*

If using *only* the native-command software, it is possible to bypass the Setup Wizard as follows:

- 1 Insert the CD in the host PC.
- 2 If the Setup Wizard starts automatically, close or ignore it
- 3 Open a Windows Explorer and go to the CD drive
- 4 Copy the contents of the *Mercury_Native_Software* directory to a directory on the host PC.
- 5 Access the programs you need from that directory. To start the *MMCRun* host software, for example, go to that directory, open the *MMC_Run* subdirectory and double-click the *MMCRun* “.EXE” file there.

2.2 Starting Operation

This section assumes that you wish to control one or more Mercury™ Class controllers together from a single RS-232 or USB port on a host computer and are connecting a motorized axis to each controller. Be sure to refer to the User Manuals of all other hardware in the system.

- 1 Set device address:
Make sure each controller is set to a suitable device address. The factory default setting of the address DIP switches is all ON, corresponding to device number 1. Each Mercury™ Class controller in a network must be set to a unique address. See “DIP Switch Settings” on p. 17 for details on setting device numbers.

- 2 Interconnect controller or network and host PC:
Between the first controller and the PC use either the USB cable or the null-modem RS-232 cable, but not both.
If there are to be additional controllers in the network, chain them from RS-232 Out to the RS-232 In of successive units. Use C-862.CN straight-through RS-232 cables (M-F) between the units.
Each controller on the network must be set to the same baud rate (DIP switches 5 and 6 on the C-663) and it must be the same as that assumed by the host software.

CAUTION

Never connect a DC-motor drive to a C-663 Stepper Motor controller. Irreparable damage could result.

- 3 Connect each motorized axis to the corresponding controller.
- 4 Connect a protective ground to the grounding screw on the rear panel of each controller.
- 5 Connect power to each of the controllers. If there are C-862 (black) Mercurys™ in the system, be especially careful when connecting the power supplies.

The status LED (label STA) now will glow green for normal operation. This means that the motor output current is at the hold value upon power-on.

Note that when using the USB interface for the first time, two FTDI USB drivers now must be installed on the host PC. These drivers are provided on the Mercury™ CD in the \USB_Driver directory. Follow the on-screen instructions. Installing the USB drivers requires administrator rights on the host PC.

- 6 Start the PI host software you wish to use and set it up for the stage(s) connected to your system. The sample screen shots that follow are from *PI MikroMove™*, though *MMCRun* can be used instead; more details on the host software operation are given in the respective software manual.

NOTE

A USB connection will appear as an extra COM port when the controller is connected, powered up, and the USB drivers are installed.

The baud rate used by the host must be the same as that set on the DIP switches, even if the USB interface is used!

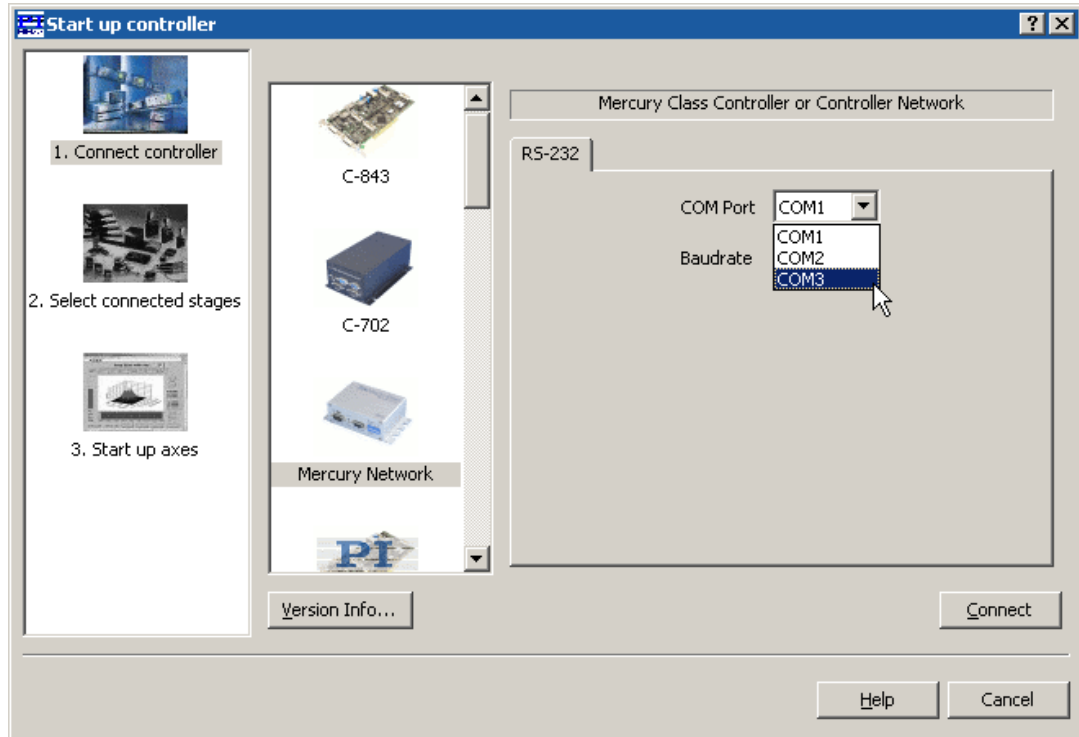


Fig. 2: PIMikroMove™ Start up Controller screen, Step 1 with “extra” COM port for USB connection being selected

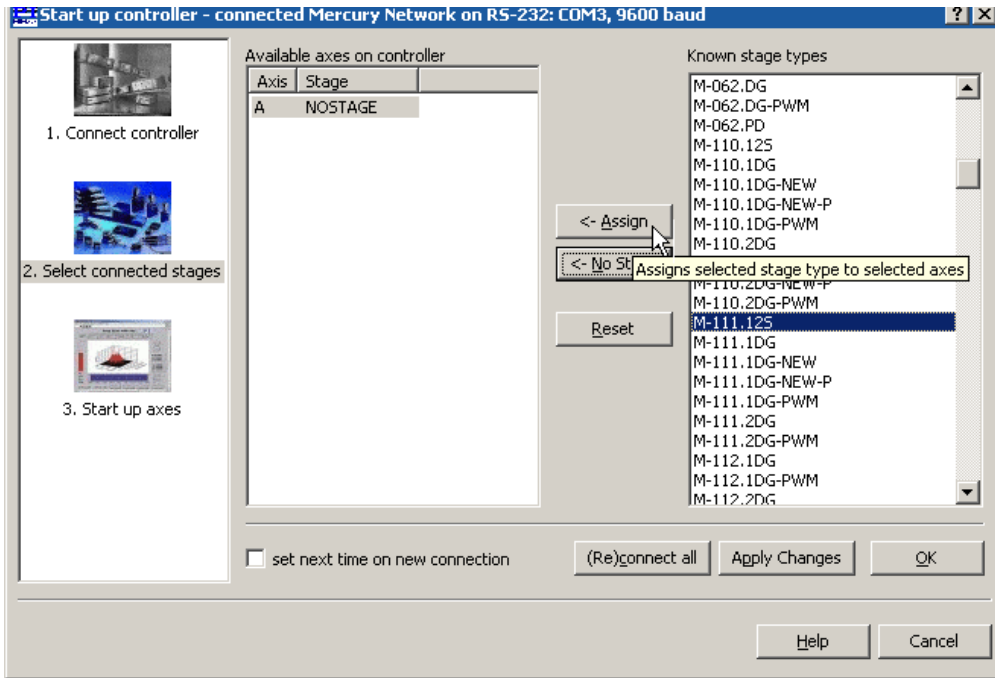


Fig. 3: PIMikroMove™ Start up controller: Step 2: select stage and be sure to click Assign before moving to Step 3 with OK

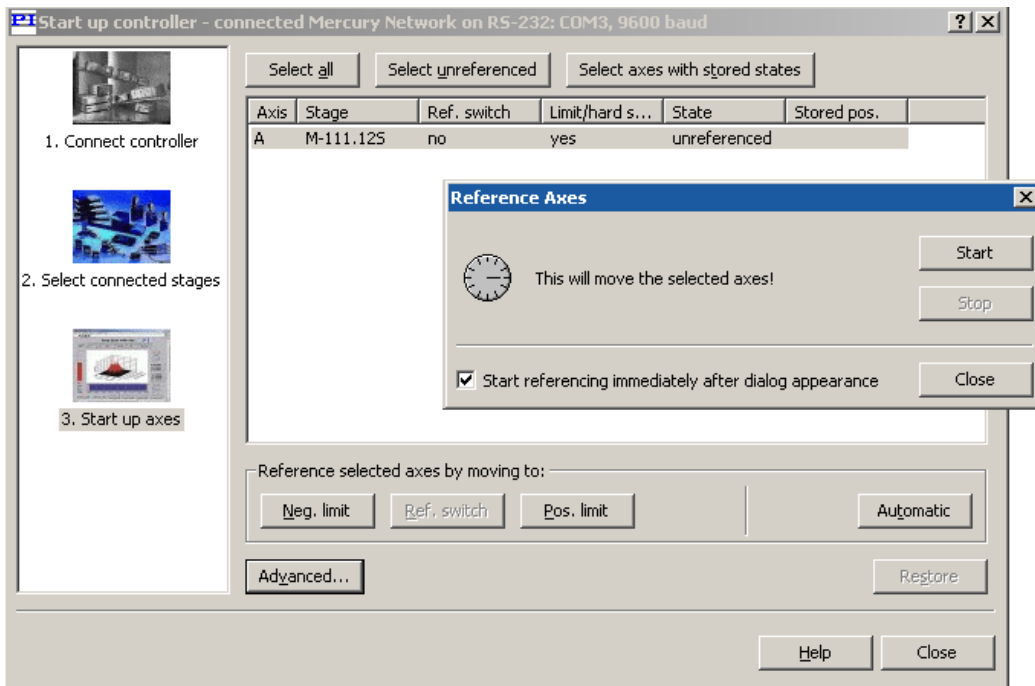


Fig. 4: PIMikroMove™ Start up controller, Step 3: Reference the axis

- After communication has been established, make sure the motion parameters of each axis are set as required for the mechanics. Starting parameters will be set if the software was informed of the type or class of stage connected (as in Fig. 3) . Otherwise it may be necessary to set the stage parameter values directly. Appropriate starting values are given in

the User manual for the stage.

Note that upon stage selection, *PIMikroMove*[™] sends an INI command which initializes the axes.

- 8 Make sure that the controller knows the absolute position of each axis (i.e. that the axis is referenced). If the software did not prompt for referencing (as in Fig. 4), it may be assuming that the current position is 0. Arrange for the axis to move toward and stop at a limit or reference switch. See the respective software or commands manual for details.

CAUTION

Commanding a velocity above the maximum possible for the stage will cause the motor to stall. Because stepper motors do not have position encoders, the position counter will continue to increment (it counts motor steps). The controller's motor position may not correspond with the actual motor position and this might endanger your application.

The maximum velocity depends on various influences like operating voltage, phase current setting and mechanical load. Datasheet values are for orientation only and may not work under all conditions.

Check out the maximum possible velocity for your individual application!

- 9 Start some test moves of the axis:
In the *PIMikroMove*[™] main window, for example, you can perform steps of a predefined size by pressing the associated arrow buttons for the axis


3 Details of Operation

3.1 Operating Elements

3.1.1 Front Panel Elements



Fig. 5: Front view of C-663 Mercury™

Front		
Labeling	Element Type	Purpose
RS-232 In	Sub-D 9(m)	Serial connection to host PC or to previous controller in a daisy-chain network
RS-232 Out	Sub-D 9(f)	Serial connection to next controller in a daisy-chain network
	Mini-B type USB	Universal Serial Bus for connection to host PC, do not connect when RS-232 IN is connected
STA	LED green/off	Controller status: green: motor on (motor output current is at the hold value) off: motor off or firmware update mode (selected by DIP switch 8)
ERR	LED red/off	Error status: red: error. The error status is cleared (LED is turned off) when the error status is read by command off: no error
Mode, Baud, Addr	8-bit DIP switch	Setting device number, RS-232 baud rate, limit-switch and firmware update modes (see DIP Switch Settings, p. 17)

3.1.2 DIP Switch Settings

Switch 1 to 4:	Address / device number (16 possible combinations)
Switch 5 and 6:	Baud rate
Switch 7	Limit switch mode
Switch 8	Firmware update mode

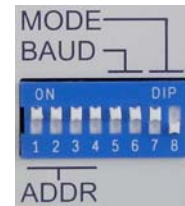


Fig. 6: slider up ON
slider down OFF

Factory settings are shown in bold.

Dev. No.	Switch Value	SW1	SW2	SW3	SW4
1	0	ON	ON	ON	ON
2	1	ON	ON	ON	OFF
3	2	ON	ON	OFF	ON
4	3	ON	ON	OFF	OFF
5	4	ON	OFF	ON	ON
6	5	ON	OFF	ON	OFF
7	6	ON	OFF	OFF	ON
8	7	ON	OFF	OFF	OFF
9	8	OFF	ON	ON	ON
10	9	OFF	ON	ON	OFF
11	10	OFF	ON	OFF	ON
12	11	OFF	ON	OFF	OFF
13	12	OFF	OFF	ON	ON
14	13	OFF	OFF	ON	OFF
15	14	OFF	OFF	OFF	ON
16	15	OFF	OFF	OFF	OFF

Baud rate*	SW5	SW6
9600	ON	ON
19200	ON	OFF
38400	OFF	ON

*Other settings are fixed at 8 data and 1 stop bit, no parity; Internal buffers are used so there is no handshake required; *MMCRun* supports only 9600 baud

HW Limit Switch Mode*	SW7
Limits active-low	ON
Limits active-high	OFF

* Hardware limit switch mode must agree with software setting

Firmware Update Mode	SW8
Update	ON
Normal operation	OFF

NOTES

Modified DIP switch settings come into effect after the next power-on or reset.

In the native command set the controller's "address" is the same as the "switch value".

In the GCS command set, the controller's "address" is the same as the "device number".

3.1.3 Rear Panel Elements

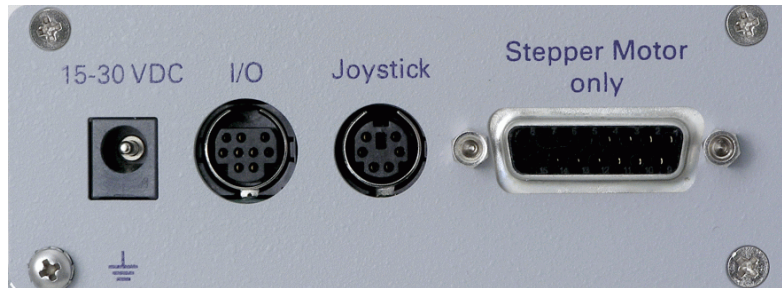



Fig. 7: Rear view of C-663 Mercury™

Rear		
Labeling	Element Type	Purpose
15-30 VDC	Barrel connector	Supply power input, center positive
I/O	Mini DIN connector, 9-pin	Digital I/O and analog input
Joystick	Mini DIN connector, 6-pin	Analog joystick (input)
Stepper Motor only	Sub-D 15(f)	Motor/stage connection (I/O): stepper motor only!
	Screw & washer	Protective ground connection

CAUTION

Never connect a DC-motor drive to a C-663 stepper-motor controller. Irreparable damage could result.



3.2 Supply Power Connection

The C-663 comes with a 24 V wide-range-input power supply that can be used with line voltages from 100 VAC to 240 VAC at 50 or 60 Hz.

To power on the C-663, proceed as follows:

- 1 Because grounding is not assured over the power connection, connect the C-663 chassis to a protective ground via the labeled screw on the rear panel (see figure below)



- 2 Connect the included wide-range power supply to the "15-30 VDC" connector of the C-663

- 3 Connect the AC power cord of the power supply to the wall socket. The status LED (label STA) on the C-663 front panel now will glow green (motor on, i.e. motor output current is at the hold value) for normal operation. If the STA LED does not glow, check the DIP switch settings on the C-663 front panel: all LEDs stay off when the C-663 is in firmware update mode (DIP switch 8 in ON position)

3.3 Power-On Settings

The C-663 Mercury™ Step has the following power-on settings:

Parameter	Factory Default Value	Unit
Velocity:	20000	steps*/s
Acceleration:	250000	steps*/s ²
Current while moving:	200	mA
Current while holding:	100	mA
Time to hold:	200	ms
Active limit level high/low:	active low	
Limit ON/OFF:	limits enabled	
Brake ON/OFF:	brake on	

* The C-663 uses microsteps = 1/16 of the full steps given in the motor hardware documentation

Values can be set at any time by command (e.g. in a macro). *PIMikroMove*™ sends the commands to set appropriate values automatically when a known stage is connected. *MMCRun* does the same when a class of stage is selected on the start screen.

In many cases even the values provided for the stage by the software or by the stage User Manual will not be optimum for your application.

If customized start-up values for any parameters are required, the autostart macro of the controller (Macro #0) can be used to set them. It is not possible to store modified parameters as new power-on defaults directly.

3.4 Connecting Mercurys™ with the Host PC

CAUTION

Never connect the RS-232-IN and USB connectors of the same controller to a PC at the same time, as damage may result.

Use either the RS-232 or the USB interface to connect the controller or controller network to the host PC. Interconnect any additional Mercury™ Class controllers being networked with straight-through RS-232 cables chaining from the RS-232 OUT to the RS-232 IN connectors of each in turn. Each controller in a network must be set to a unique address using DIP switches 1 to 4.

The baud rate used by the host must be the same as that set on the DIP switches 5 and 6, even if the USB interface is used. The default baud rate is 9600 baud (DIP switches 5 and 6 in ON position). Other Mercury™ settings are fixed at 8 data and 1 stop bit, no parity; internal buffers are used so there is no handshake required.

NOTES

Modified DIP switch settings come into effect after the next power-on or reset.

Note that MMCRun supports only 9600 baud.

3.4.1 RS-232 from Host

If you use RS-232, connect the C-815.34, RS-232 null-modem cable (F-F) between the "RS-232 IN" socket to the desired COM port of the host computer.

3.4.2 USB from Host

When you connect via the USB interface for the first time, after the C-663 is powered up, a message will appear saying that new hardware has been detected. Follow the on-screen instructions and show the Hardware Wizard the \USB_Driver directory on the Mercury™ CD. Installing the USB drivers requires administrator rights on the host PC.

NOTE

The USB drivers will make the USB interface appear to all software on the host PC as a new COM port. That port will be present only when the controller is connected via USB and powered-up. Even when using USB, it is necessary to set the baud rate consistently on the PC and all connected devices.

With current firmware, it may be necessary to power-cycle the controller while the host PC is on to establish communication with it.

3.4.3 Networking

C-663 Mercury™ step controllers can be networked with each other or, for example, with C-863 Mercury™ DC-motor controllers.

Up to 16 controllers can be connected to one RS-232 or USB port. The controllers interconnect with an RS-232 bus architecture (daisy-chain) via straight-through networking cables (order number C-862.CN), chaining from the RS-232 OUT to the RS-232 IN connectors of each in turn.



Fig. 8: C-862.CN Mercury™ Class network cable, length 28 cm, other lengths available on request

The RS-232 output stages of some PCs may not be capable of driving more than 6 units; if this is a problem use USB to interface with the PC.

Each controller on the network must be set to a unique address, but they do not need to be in any particular order. With the C-663, the controller address is set in the first four DIP switches on the front panel (see p. 17). Furthermore, all controllers in a daisy chain network must be set to the same baud rate using the DIP switches 5 and 6.

The networking feature does not support simultaneous ASCII communication with the various controllers, but simultaneous motion of the connected axes and/or macro execution is possible.

The GCS and native command sets present different communications models at the user interface. See the respective Commands manual for details. With current firmware, the hardware model is that of the native command set—i.e. the GCS commands are translated by the GCS DLL to commands in the native command set and the proper controllers are selected using the native addressing mechanism with its address selection codes.

All controllers connected can execute individual macro commands at the same time, but there is no direct communication from controller to controller, only from controller to the PC. The host program must handle address selection and motion sequencing among different controllers. This communication model imposes certain limits for path interpolation and multi-axis motion control as well as for conditional motion execution.

See the respective Commands manual for programming examples.

3.5 Limit Signals

3.5.1 Limit Switches

During operation, limit sensors (switches) can be used to stop motion at the end of the allowable travel range and/or to provide absolute position information (referencing); see the commands manual for the command set you are using for details of commands that use these signals. Each of the two switches will interrupt motion in a particular direction. If positioning equipment from PI is used, the limit switches or sensors are pre-wired for operation with Mercury™ controllers.

The Mercury™ controller can be configured to accept either an active-high or an active-low stop signal from the limit sensors (both must be the same). As default low signals are used to inhibit movement.

The limit switch signals are interpreted by both the controller hardware and software. With DIP switch 7, the hardware interlock can be changed between active-high (OFF) and active-low (ON). For the software configuration, see the Commands manual of the command set you are using (native: Lx commands, GCS: SPA).

NOTE

If the hardware and software limit switch configurations are not compatible, no motion is possible. PI stepper motor stages have active-low limit switches, so SW 7 must be ON.

3.5.2 Soft Limits

It is possible to set "soft limits" which establish a "safety distance" which the stage will not enter on both ends of the travel range. Soft limits halt the motion of given axes smoothly, i.e. with regard to the current valid deceleration. They can protect the stage from damage which could occur if it runs into the hard stop at full speed because the limit switches in the stage stop the stage without deceleration.

Depending on the command set you are using, the behaviour of the stage differs when a position beyond a soft limit is commanded:

- Native Command Set: The stage moves and is stopped smoothly at the corresponding soft limit.
- GCS Command Set: The stage will not move, and an appropriate error code will be set.

Soft limits are also applied during joystick-controlled motion.

The soft limits always refer to the current home position (zero position). This means that changing the home position will also shift the soft limits.

For details see the Commands manual of the command set you are using.

3.6 Reference Signal

A position reference switch (sometimes called “origin sensor”) working in conjunction with the capture mechanism of the motion processor can be used to provide absolute position information (referencing).

The Mercury™ Step controller accepts signals from a direction-sensing reference switch. That signal is expected to be high when the axis position is on one side of the reference position and low when the position is on the other side.

See the Commands manual for the command set you are using for details of commands which can use the reference signal.

3.7 Input/Output Lines

3.7.1 Overview

C-663 Mercury™ Step controllers offer 4 digital outputs and 4 inputs for either digital or analog (8-bit resolution) use (see p. 39 for pinout of the I/O socket).

A set of commands is available to handle these I/O lines (for descriptions see the respective native or GCS Commands manual).

Note that there are commands which allow conditional command execution depending on the digital inputs. This makes possible, for example, pushbutton or trackball control (see pages 31 and 31), inter-axis triggering in stand-alone mode (i.e. without a PC connection) or trigger output for control of external devices (see below).

3.7.2 Trigger Output

You can program the digital output line 4 of the C-663 to trigger other devices (pin 8 of the “I/O” socket, see p. 39).

The following trigger modes are supported by the C-663:

- Single trigger pulse: a trigger pulse is written when the axis has reached a certain position
- Multiple trigger pulses: the first trigger pulse is written when the axis has reached a certain position, and then a trigger pulse is written whenever the axis has covered a given distance

The trigger mode selection and the settings for trigger position and distance can be made by command. For details and examples, see the Commands manual for the command set you are using.

4 Updates

4.1 Firmware Update

The current firmware revision of your Mercury™ Step controller is contained in the response to the following commands:

- Native command set: VE
- GCS command set: VER?

It is possible to update the Mercury™ Step firmware over the RS-232 PC interface.

The controller architecture with separate boot loader and firmware proper is designed to prevent aborted updates from locking up the device. However, it is still recommended that firmware updates be executed with care and not be made the subject of experimentation. Because of reliability problems with certain hardware, the USB interface is not recommended for firmware update.

NOTE

Firmware update does not erase macros stored on the controller.

To update the firmware, proceed as follows:

1. Connect the Mercury™ to be updated to the host computer via RS-232; (use of the USB interface is not recommended)
2. Set DIP switch 8 to ON (firmware update mode) and power cycle the device; both LEDs will remain dark
3. In Windows Explorer, go to a directory on the harddrive containing both the MMC_update program executable and the new firmware image (if necessary, create it and copy the files there)
4. Make sure the new firmware image filename begins with "C663" and has the ".hex" extension (rename it, if necessary)
5. Start the MMC_update program

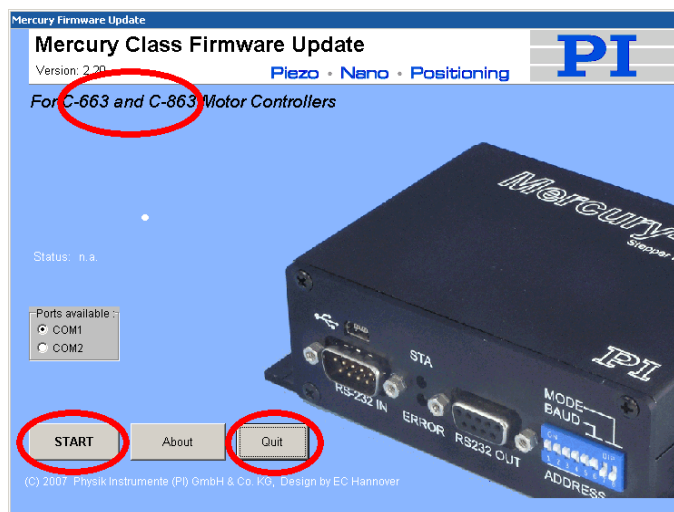


Fig. 9: MMC_update opening screen; supported controller models circled

6. Make sure that your controller is named in the headline; the controller depicted on the screen may differ (as in screenshot above)
7. Make sure the proper COM port is selected (use of the USB virtual COM port is not recommended)
8. Click *START* to proceed to the main screen

NOTE

MMC_Update cannot determine the exact type of controller connected. If Mercury™ Step firmware is loaded in a Mercury™ DC motor version or vice-versa, the unit will not start. In such a case, try rerunning *MMC_update* with the correct firmware type.

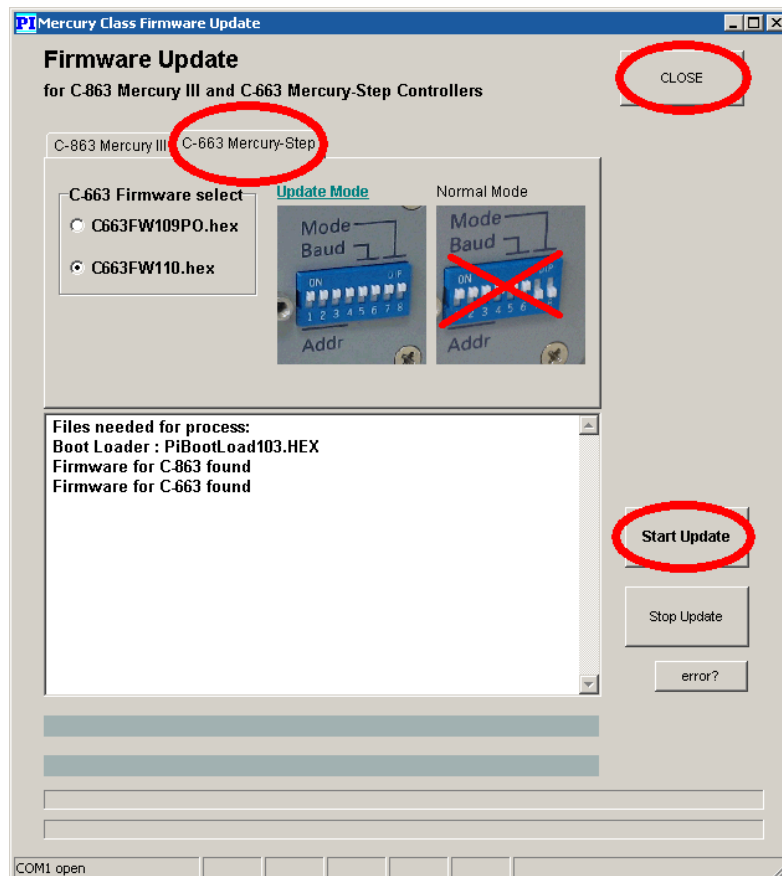


Fig. 10: MMC_update main screen; with C-663 tab and Start Update button circled

9. Make sure the C-663 tab card is on top.
10. Make sure the proper firmware file is selected in the *C-663 Firmware select* pane. There is a radio button there for each file in the same directory as the *MMC_update* executable that begins with "C663" and has the "hex" extension.

NOTE

Using the *Stop Update* button is not recommended.

11. Click *Start Update* once and once only.
 The four bars above the status line will provide progress information, and log messages will appear in the field above them (see table below).
 If an error occurs, power cycle the controller and repeat steps 9-11
12. When the update is complete, return to the opening screen with *CLOSE* and then exit the program with *Quit* to release the COM port.

Messages

The following table lists and explains some of the messages that may appear during the update process.

Message	Possible Explanations	Action to Take
Can not detect baud rate / Verify switch 8 ON?	Not in firmware update mode	Set DIP switch 8 set to ON and power cycle controller; both LED should be dark
	RS-232 cable not connected, or connected to wrong port	Connect cable; restart software and select another port
	Wrong RS-232 cable connected to wrong RS-232 connector	Connect a null-modem cable from host PC serial port to RS-232 in of the Mercury™ to be updated
	Mercury™ not powered up	Connect to power
	More than one instance of MMC_update running on the same COM port	Close all instances and reopen a single instance—note that minimized instances may dock themselves to a corner of the desktop
	Another program has control of the COM port; exit the other program completely	Close all other programs that use the COM port
Can not connect bootloader	You double-clicked <i>Start Update</i>	Click <i>Start Update</i> only once
Error while sending boot loader	Communications error	
Error: 0	You clicked the <i>Error?</i> button (Error code 0 indicates no error, or that the error has been cleared by the software)	No action necessary
File xxx will be loaded Start_update: 0	Normal course of events	
Copy boot loader	Normal course of events	After this point, clicking <i>Stop update</i> is only recommended if you see you are loading firmware for the wrong device type
Connect Bootloader Erase Flash Send Data Block Update Complete	Normal course of events	

4.2 Software Updates

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

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

- 1 On the www.pi.ws front page, click on *Download/Support* in the *Service* section on the left
- 2 On the *Download/Support* page, click on *Manuals and Software*
- 3 On the *PI Download Server* page, enter the Username and the Password which are provided in the MercuryReleasenews xxxxx.pdf on the Mercury CD and click on *Login*
- 4 Click on *Download* in the navigation bar across the top
- 5 Click on the *C Motion Controllers* category
- 6 Click on *C-663*
- 7 Click on *Software* (if you click on *Documents* you will get the latest manuals)
- 8 Click the download button below the latest CD-Mirror (includes the manual versions that were with the release)

5 Manual Control

5.1 Joystick

C-663 Mercury™ Step controllers offer convenient manual motion control by using analog joysticks (order numbers C-819.20, C-819.30) connected to the “Joystick” socket (pinout on p. 39). It is the velocity of the controlled axes that is determined by the joystick position. See the Commands manual for the command set you are using for the joystick-related commands (e.g. for activation/deactivation of joystick control).

CAUTION

Do not enable a joystick axis in the software when no joystick is connected to the controller hardware. Otherwise the corresponding controller axis may start moving and could damage your application setup.

5.1.1 Joystick Control Accessories

- C-819.20 Analog joystick, 2 axes
- C-819.20Y Cable to connect one C-819.20 joystick to two Mercury™ Class controllers. Each controller sees only one axis and the states of that axis' buttons
- C-819.30 3-axis analog joystick for Mercury™ controller

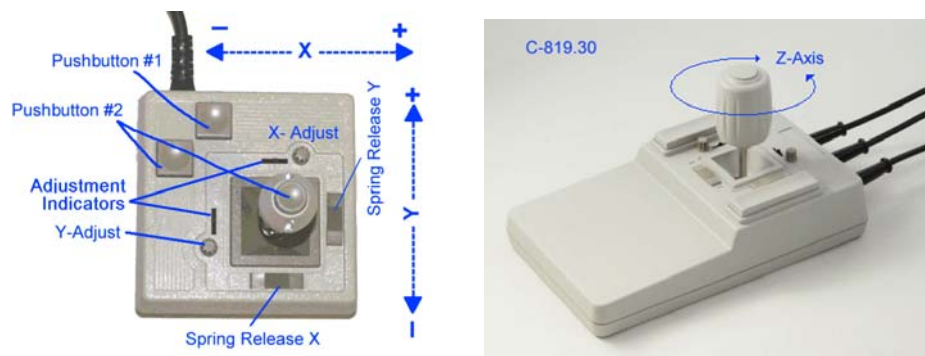


Fig. 11: C-819.20 (left) and C-819.30 (right) joysticks for Mercury™ Controllers

NOTES

While C-819.30 is equipped with separate connection cables for three controllers, you need a C-819.20Y cable to connect C-819.20 to two different Mercury™ class controllers.

If a C-819.20Y Y-cable is used, joystick power is taken from the X-axis branch, so it must be connected to a powered-up controller.

5.1.2 Joystick Response Tables

Normally the joystick response is defined by loading a predefined response table. For special purposes, it is possible to generate a user-defined joystick response table.

See the Commands manual for the command set you are using for details.

See also Section 5.1.3 below for a special application.

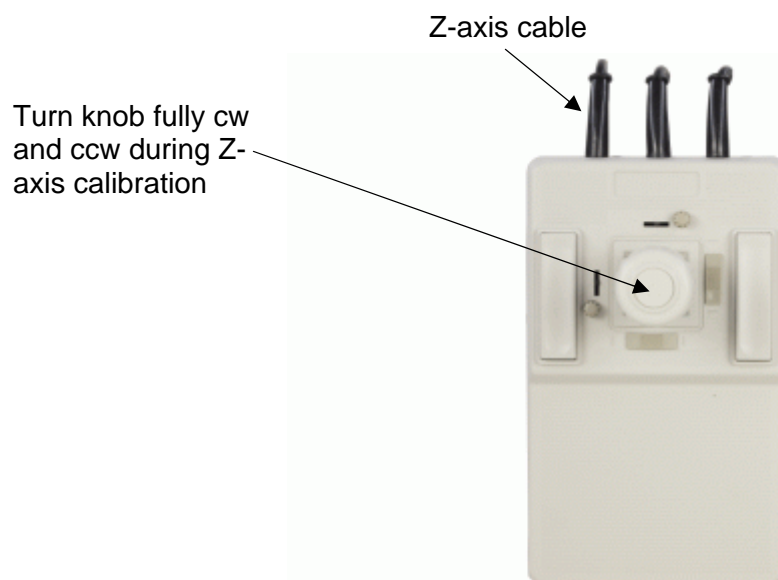
5.1.3 Z-Axis Calibration with C-819.30 Joystick

The C-819.30 3-axis analog joystick can be connected to up to three C-663 or C-863 Mercury™ controllers via separate connection cables.

The working principle of the joystick X- and Y-axis is the same as with conventional 2-axis devices. The Z-axis of C-819.30 is operated by turning the knob at the end of the stick clockwise or counterclockwise.

All three joystick axes are spring centered so that the stick returns to its center position when released. The center position should correspond to zero velocity so that the connected stages are at rest. To ensure that the center position corresponds to zero velocity, C-819.30 provides adjustment knobs for the X- and Y-axis. For the Z-axis, a special calibration procedure is required since the Z-axis features a different response behaviour and has no adjustment knob.

The Z-axis calibration procedure is done using the *MMC_Z-Joy* software tool (*MMC_Z-Joy.exe* is provided on the Mercury™ CD). Based on the joystick Z-axis output, this tool generates a new cubic response table that enables the stage to reach its maximum velocity, combined with high sensitivity at low velocities. The new response table is saved permanently in the Mercury™ controller until it is overwritten by a new joystick-response definition.



Proceed as follows:

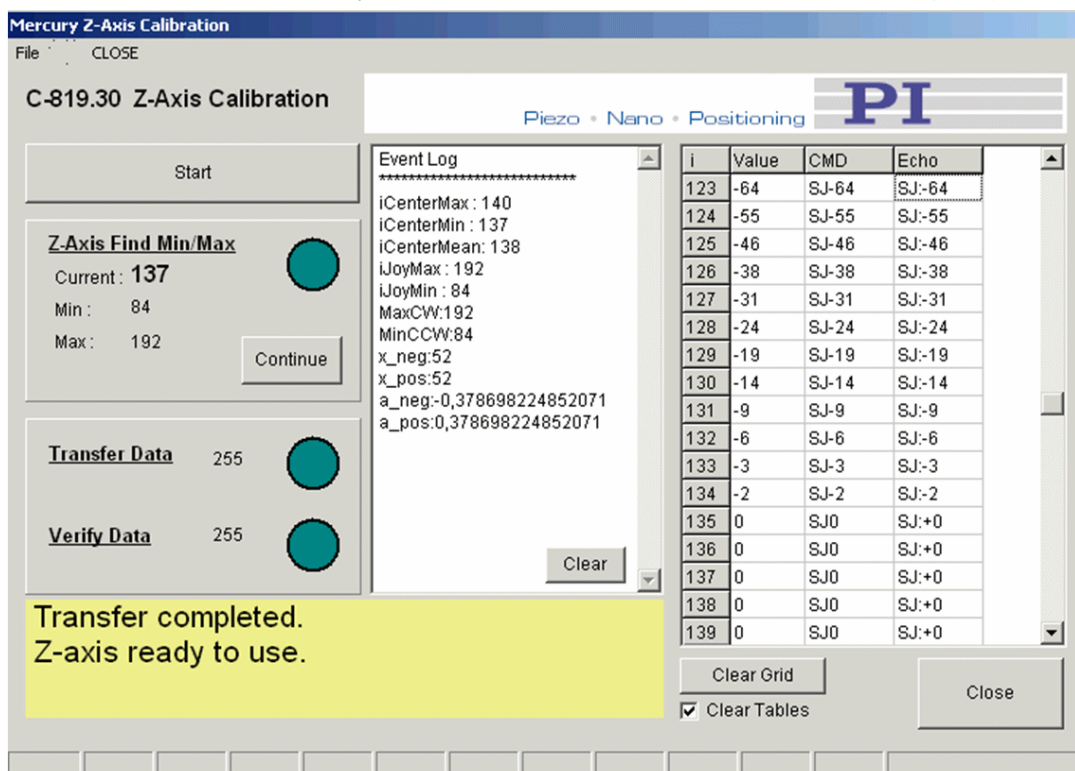
1. Connect the Z-axis cable of the joystick to the Mercury™ controller which is to be operated by the joystick Z-axis.

Notes:

It is recommended to mark Z-axis cable and controller since they belong together after the calibration procedure. Interchanging joystick axis and controller requires a new calibration procedure.

It is not necessary to connect a stage to the controller for the calibration procedure.

2. Connect the Mercury™ controller to the PC.
3. Power up the Mercury™.
4. Start the MMC_ZJoy software tool on the PC via its executable (.exe) file.



C-819.30 Z-Axis Calibration

Start

Z-Axis Find Min/Max

Current: **137**

Min: 84

Max: 192

Continue

Transfer Data 255

Verify Data 255

Transfer completed.
Z-axis ready to use.

Event Log

```
*****
iCenterMax: 140
iCenterMin: 137
iCenterMean: 138
iJoyMax: 192
iJoyMin: 84
MaxCW:192
MinCCW:84
x_neg:52
x_pos:52
a_neg:-0,378698224852071
a_pos:0,378698224852071
```

i	Value	CMD	Echo
123	-64	SJ-64	SJ:-64
124	-55	SJ-55	SJ:-55
125	-46	SJ-46	SJ:-46
126	-38	SJ-38	SJ:-38
127	-31	SJ-31	SJ:-31
128	-24	SJ-24	SJ:-24
129	-19	SJ-19	SJ:-19
130	-14	SJ-14	SJ:-14
131	-9	SJ-9	SJ:-9
132	-6	SJ-6	SJ:-6
133	-3	SJ-3	SJ:-3
134	-2	SJ-2	SJ:-2
135	0	SJ0	SJ:+0
136	0	SJ0	SJ:+0
137	0	SJ0	SJ:+0
138	0	SJ0	SJ:+0
139	0	SJ0	SJ:+0

Clear

Clear Grid

Clear Tables

Close

5. Click *Start* to enter the main window, then click *Start* again to initiate the calibration procedure.
6. When the *Z-Axis Find Min/Max* pane starts blinking, turn the Z-axis knob on the joystick fully clockwise and then fully counterclockwise (i.e. to the mechanical end stops).
7. Click *Continue*. Now the joystick response table is calculated, transmitted and verified. The procedure has finished when the *Z-axis ready to use* message is displayed.
8. Click *Close* to quit the software tool.

The Mercury™ Step controller is now prepared to handle the joystick Z-axis. The corrected response curve is stored permanently in the controller. If you want to

undo the special adjustment and use the controller with a "normal" joystick axis, reload a suitable, predefined response curve by command. See the Commands manual of the command set you are using for details.

5.2 Pushbutton Control

The C-170.PB pushbutton box connects to the "I/O" socket (pinout on p. 39) of the Mercury™ Step controller. It allows applying TTL signals to the input lines and displays the state of the output lines on LEDs.

You can use the pushbutton box to execute a command or macro conditionally, depending on whether a pushbutton is pressed or not. See the Commands manual of the command set you are using for macro details and examples.



Fig. 12: C-170.PB pushbutton box with LEDs for Mercury™ Controllers

5.3 Trackball Control

A trackball device can be used to set the target position. Connect the digital TTL signals A and B (also referred to as quadrature signals) provided by the trackball to the digital input lines 3 and 4 of the "I/O" socket (pinout on p. 39) on the Mercury™ Step controller. The lines are terminated by 10k to GND.

Each signal transition shifts the target position by a given amount in positive or negative direction. This makes possible extremely fine motion control with high position resolution.

See the Commands manual of the command set you are using for details regarding trackball control.

6 Macros

The Mercury™ controller macro feature allows defining command sequences and storing them permanently in non-volatile memory in the controller. Each defined macro can be executed later by command. A Start-up macro can be defined that will be executed automatically every time the controller is started, making possible customized stand-alone operation without a host computer.

If you are working with one of the PI operating software interfaces (*PIMikroMove™* or *MMCRun*), it is also possible to store macros on the host PC (known as *host macros*). See the corresponding software manual for details.

The native and GCS command sets have differing macro architectures. In each, however, a macro consists of valid commands in the corresponding command set. The Mercury™ GCS macro capability is implemented for Mercurys™ with native firmware via the GCS DLL. That DLL translates macros between the two command sets and macro architectures. See the Command Manual for the command set you are using for details.

Examples of what is possible with an autostart macro sequence and a pushbutton box and/or joystick to make the Mercury™ a stand-alone controller under operator control include the following:

- Execute commands that set proper operating parameters
- Execute motion commands
- Execute commands that wait for motion to complete
- Light one or more LEDs
- Execute a command or macro conditionally, depending on whether a pushbutton or joystick button is pressed or not
- Execute a command or macro conditionally, depending on the response to certain other commands
- Execute a command or macro conditionally, depending on the position of the connected joystick axis
- With custom cabling, execute a command or macro conditionally, depending on the state of an output from a different Mercury™ or other external signal

The techniques generally employed include:

- Using separate macros for different initialization and operating functions
- Calling one macro from another, perhaps conditionally
- Invoking repeated execution of a macro
- Wiring outputs from one Mercury™ to the inputs of another

See the Commands manual for the command set you are using for details.

7 Troubleshooting

NOTE

The controller must be power-cycled to read in new DIP switch settings. Power-cycling will return internal parameters to their power-on values.

Problem (1):	When the Mercury™ Step is powered up, both front-panel LEDs stay dark (normally upper LED lights green).
Solution:	<p>Make sure that:</p> <ul style="list-style-type: none"> ■ Switch #8 on the front panel is in lower position (firmware update OFF) ■ There is no start-up macro switching the motor off ■ Power supply with suitable voltage is connected and operating properly
Problem (2):	Axis starts to move before you send any move commands.
Solution:	<ul style="list-style-type: none"> ■ The autostart macro is active. Solution: Erase the autostart macro ■ Joystick is connected but not calibrated. Before a joystick can be operated correctly, a calibration routine may need to be performed. Activating the joystick before calibration may cause the motor axis to start moving even though the joystick is in the neutral position. To calibrate a joystick axis turn the corresponding “Adjust” knob on the joystick until the motor stops. If using <i>MMCRun</i> this procedure is facilitated by clicking <i>Adjust</i> in the <i>Joystick</i> pane. See the <i>MMCRun</i> software manual for details. For calibration of the joystick Z-axis, a special calibration procedure is required, see p. 29 for details. ■ Joystick was activated but no joystick is connected. Disable joystick

Problem (3):	When the PI host software starts, no communication with the Mercury™ can be established. Error message "No controller found" or "Could not connect...".
Solution:	<p>Make sure that:</p> <ul style="list-style-type: none"> ■ COM port is not being used by another program ■ Baud rate setting correct (for 9600 baud: SW 5&6:ON). Note that <i>MMCRun</i> requires 9600 baud. ■ Upper LED lights green? If it stays dark, see problem (1) ■ Firmware has current version level ■ For RS-232 connection: C-815.34 null-modem cable is being used ■ Either RS-232 or USB cable is connected, but not both
Problem (4):	The device does not respond to your own terminal program.
Solution:	<p>Make sure that:</p> <ul style="list-style-type: none"> ■ Communication can be established with PI software ■ You have sent the proper <i>address selection code</i> or that there is an SC command in the autostart macro with the proper unit address. The device address is set with DIP switches 1 to 4 on the front panel. See "Addressing" in the Native Commands manual for details. ■ Baud rate settings on controller and host match
Problem (5):	The controller communicates and reports position values, but the connected stage or motor does not move.
Solution:	<ul style="list-style-type: none"> ■ Make sure that the motor brake is set to OFF. Some stages without a physical brake deactivate the motor when the brake signal is in the default ON state ■ Verify that the limit-switch-polarity and stage-has-limit-switches axis parameters (depends on stage type) are set properly (in native command set, commands: Lx, in GCS SPA) ■ Verify that the hardware-interlock limit switch polarity is properly set (DIP switch 7 ON for active-low, for standard PI stepper motor drives) ■ The controller might be in the joystick or trackball mode where move commands are not accepted. To enable motion by move commands, disable the joystick or trackball mode (in native command set: JF; in GCS: JON or SPA)

Problem (6)	Commands are not recognized.
Solution:	<ul style="list-style-type: none"> ■ In <i>MMCRun</i>, try using "straight" instead of "protected" command mode. Perhaps the command in question is too new to be recognized by this version of the program ■ Make sure multi-character commands are followed by a valid termination character (<code>CR</code> for native, <code>LF</code> for GCS command set)
Problem (7)	The controller does not seem to react properly to limit signals from the stage.
Solutions:	<ul style="list-style-type: none"> ■ Verify that the "limit-switch-polarity" and "stage-has-limit-switches" axis-parameters (depends on stage type) are set properly (in native command set: Lx commands; in GCS: SPA) ■ Non-PI stage limit signal not compatible with Mercury™ limit signal input (e.g. insufficient source or sink current). Contact PI and stage manufacturer to determine where the problem lies ■ Verify that the hardware-interlock limit switch polarity is properly set (DIP switch 7 ON for active-low, for standard PI stepper motor drives)
Problem (8)	The stage does not react to joystick motions.
Solution:	<p>Make sure that the joystick is properly connected and enabled.</p> <p>A joystick calibration procedure routine may be required: see "Joystick" on p. 28 for details.</p>
Problem (9)	The stage starts to move but stops and the audio output changes.
Solution:	<p>Commanding a velocity above the maximum possible for the stage: this will cause the motor to stall. Because stepper motors do not have position encoders, the position counter will continue to increment. The controller's motor position may not correspond with the actual motor position and this might endanger your application.</p> <p>The maximum velocity depends on various influences like operating voltage, phase current setting and mechanical load. Datasheet values are for orientation only and may not work in all applications. Note that the C-663 uses microsteps which are 1/16 the size of the full steps given in the hardware datasheets.</p> <p>Check out the maximum possible velocity for your individual application!</p>

8 Technical Data

8.1 Specifications

	C-663.10
Function	Stepper motor controller, stand-alone capability
Drive type	2-phase stepper motor
Channels	1
Motion and control	
Microstep resolution	1/16 full step (6400 steps/rev with most PI stages)
Trajectory profile modes	Trapezoidal, point-to-point
Limit switch inputs	2xTTL signals, programmable logic & hardware interlock
Reference (origin) switch input	1 x TTL signal
Motor brake output	1 x TTL out, programmable
Electrical properties	
Phase (drive) current	0 to 1000 mA, programmable, default 200 mA; safe limit depends on motor
Interfaces and operation	
Communication interfaces	RS-232 (bus architecture), USB
Motor connector	Sub-D 15 (f)
Controller network	Up to 16 units on single interface*
I/O lines	4 analog/digital in, 4 digital out
Command set	Mercury™ native command set, GCS (via DLL)
User operating software	MMCRun, PIMikroMove™
Software drivers	GCS (PI General Command Set) DLL, GCS LabVIEW drivers, native Mercury™ DLL, native LabVIEW drivers
Manual control	Joystick, Y-cable for 2D-motion; pushbutton box
Miscellaneous	
Operating voltage	15 to 30 V DC
Operating temperature range	0 to 50°C
Mass	0.3 kg
Dimensions	130 x 76 x 40 mm

* 16 with USB; 6 with RS-232 (depending on RS-232 output driver of the PC)

8.2 Dimensions

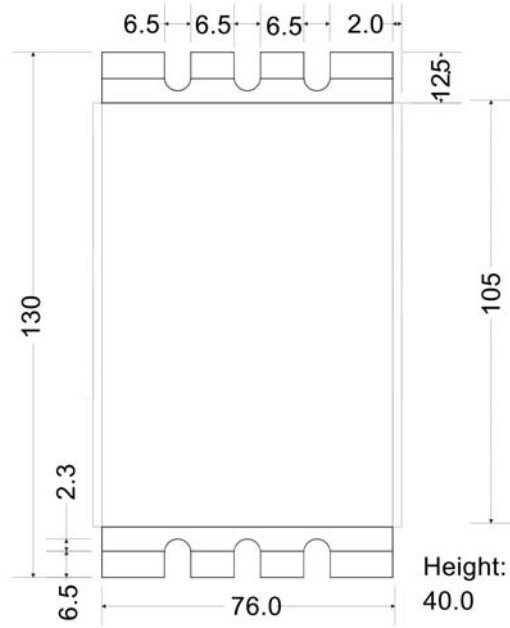


Fig. 13: C-663 dimensions in millimeters

8.3 Connector Pinouts

8.3.1 Motor-Connector

CAUTION

DC-motor drives use the same connector as stepper-motor stages but are not compatible. Permanent damage can occur if stage and controller types are not compatible.

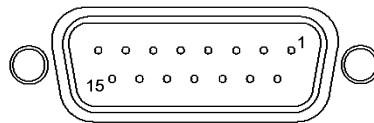


Fig. 14: Sub-D 15(f) motor connector

Pin	Signal direction	Function
1	output	Motor phase 1A
9	output	Motor phase 1B
2	output	Motor phase 2A
10	output	Motor phase 2B
3		not used (optional enc. A)
11		not used (optional enc. A-)
4		not used (optional enc. B)
12		not used (optional enc. B-)
5		n.c.
13	output	Brake signal, active low
6	output	+5V
14	input	Limit, positive, active low
7		GND
15	input	Position reference
8	input	Limit, negative, active low

8.3.2 I/O Connector

Pin	Signal direction	Function
1	input	Input 1 (analog 0 to +5V / digital, TTL)
2	input	Input 2 (analog 0 to +5V / digital, TTL)
3	input	Input 3 (analog 0 to +5V / digital, TTL)
4	input	Input 4 (analog 0 to +5V / digital, TTL)
5	output	Output 1 (digital, TTL)
6	output	Output 2 (digital, TTL)
7	output	Output 3 (digital, TTL)
8	output	Output 4 (digital, TTL)
9	output	Vcc (+5V)

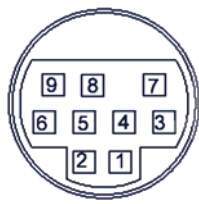


Fig. 15: Mini DIN 9-pin I/O connector

8.3.3 Joystick Connector

Pin	Signal direction	Function
1		GND
2		n.c.
3	output	Vcc (3.3 V)
4	input	Input: Joystick analog 0 to 3.3 V
5		n.c.
6	input	Input: Joystick Button #1

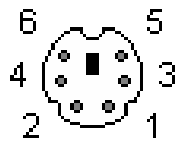


Fig. 16: Mini DIN 6-pin (PS2-style) joystick connector

8.3.4 Joystick Y-cable

The joystick Y-cable (C-819.20Y) maps the second joystick-axis (Y) position and button signals to the joystick inputs for the second controller as shown below:

Joystick Pin	Signal	Controller 1 (X) Pin	Controller 2 (Y) Pin
1	GND	1	1
2	Button 2: 0 or 3.3 V		6
3	Vcc (3.3 V supply input)	3 (3.3 V output)	n.c.
4	X-axis position, 0 to 3.3 V	4	
5	Y-axis position, 0 to 3.3 V		4
6	Button 1: 0 or 3.3 V	6	

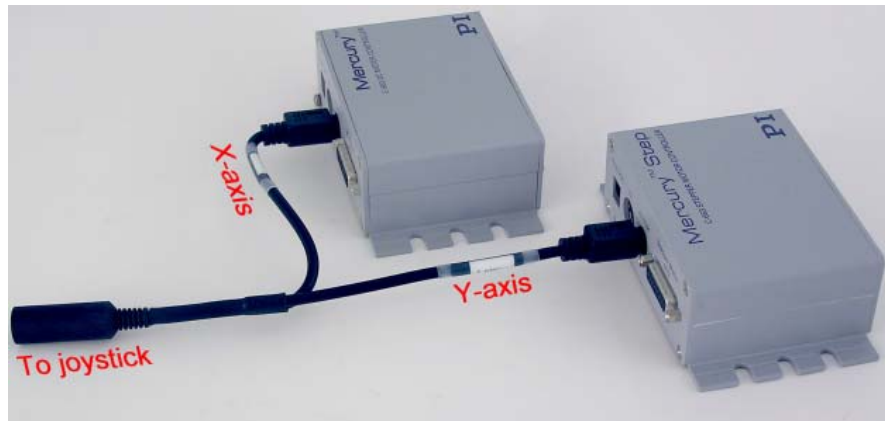


Fig. 17: C-819.20Y joystick Y-cable with 2 controllers

8.3.5 RS-232 Connectors

Connector Labels: RS-232 IN and RS-232 OUT

Connector Types: Sub-D, 9-pin, male for OUT, female for IN

The IN and OUT lines are permanently bussed together, straight-through. Only when the USB interface is active, does the controller assert signals on the Rx/D (receive) line, pin 2, otherwise that line is driven by the host PC only. This is the reason an RS-232-only network may be limited to as few as 6 units.

Note that the controller RS-232 bus is wired as a DTE and, if connected to a PC, requires a cross-over (null-modem) cable

RS-232 IN and RS-232 OUT Connectors

Pin on all Mercury™ Connectors	Signal name on all Mercury™ Connectors*	Signal direction	Function
1			n.c.
2	RxD*	PC to controller**	Commands
3	TxD*	Controller** to PC	Reports (responses)
4			n.c.
5	GND		GND
6			n.c.
7			n.c.
8			n.c.
9			n.c.

*The RS-232 connection with the PC is via null-modem cable, so the connected signal names on the PC side are reversed.

** If the PC connection is via USB, then the Mercury™ connected to the PC copies everything received from the host over USB to the Mercury™ RxD line of *both* its RS-232 connectors. It also copies everything it sees on the Mercury™ TxD line to the host via USB.

CAUTION

Never connect the RS-232-IN and USB connectors of the same controller to a PC at the same time, as damage may result.



Fig. 18: Mini type-B USB connector (top) and sub-D 9-pin RS-232 (bottom)

8.3.6 USB Connector

Industry-standard mini type-B USB connector.

8.3.7 Power Connector & Grounding Screw

Power and protective ground connections are on the rear panel:

Connector	Direction	Function
Barrel connector: Center	input	15 to 30 V DC
Barrel connector: Outside	input	Power GND
Grounding screw (lower left)	⏏	Protective ground

CAUTION

Because the unit is not grounded over the power supply, a grounding screw is provided at the lower left corner of the rear panel for connecting the metal case to a protective ground.

9 Disposal

In accordance with EU directive 2002 / 96 / EC (WEEE), as of 13 August 2005, electrical and electronic equipment may not be disposed of in the member states of the EU mixed with other wastes.

To meet the manufacturer's product responsibility with regard to this product, Physik Instrumente (PI) GmbH & Co. KG will ensure environmentally correct disposal of old PI equipment that was first put into circulation after 13 August 2005, free of charge.

If you have such old equipment from PI, you can send it to the following address postage-free:

Physik Instrumente (PI) GmbH & Co. KG
Auf der Römerstr. 1
76228 Karlsruhe, Germany

