

MS173E User Manual**C-863 Mercury™****DC Motor Controller**

Release: 1.2.7

Date: 2008-04-24



This document describes the following product:

- **C-863**
DC Motor Controller, Single-Axis, Networkable



Physik Instrumente (PI) GmbH & Co. KG is the owner of the following company names and trademarks: PI®, PIMikroMove™, Mercury™, Mercury™ Step

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

Copyright 2008 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.

Document Number MS173E, Release 1.2.7
C-863_User_MS173E127.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 This manual is superseded by any new release.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 ISO / IEC Guide 22 and EN 45014

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

The manufacturer hereby declares that the product

Product Name: **Single-Channel DC Motor Controller**

Model Numbers: **C-863**

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

Electromagnetic Immunity: EN 61000-6-1

Safety (Low Voltage Directive): EN 61010-1

April 24, 2008
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-863 DC Motor Controller. It assumes that the reader has a fundamental understanding of basic servo systems, as well as motion control concepts and applicable safety procedures. The manual describes the physical specifications and dimensions of the C-863 DC 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-863 DC 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

1	Introduction	3
1.1	Overview.....	3
1.2	Prescribed Use	4
1.3	Safety Precautions	5
1.4	Unpacking.....	7
1.5	Accessories	7
1.6	Software Interfaces.....	8
1.6.1	Native Command Set	8
1.6.2	GCS Command Set.....	8
1.6.3	Available Software.....	8
2	Quick Start	11
2.1	Front and Rear Panel Elements	11
2.2	Connecting Mercurys™ with the Host PC	12
2.2.1	RS-232 from Host.....	13
2.2.2	USB from Host.....	13
2.3	Software Installation	13
2.3.1	USB Driver.....	14
2.3.2	GCS Software.....	14
2.3.3	Native-Command Software	14
2.4	Starting Operation	15
3	System Description	19
3.1	Control of Multiple Axes.....	19
3.2	LED Status Indicators.....	19
3.3	Power-up Settings	20
3.4	Cable Connections	20
3.5	Limit Signals	21
3.6	Reference Signal	22
3.7	Input/Output Lines	22
3.8	DIP Switch Settings	23
3.9	Firmware Update	24
4	Manual Control	27
4.1	Joystick.....	27
4.2	Pushbuttons.....	28
5	Macros	29

6	Troubleshooting	30
7	Technical Data	33
7.1	Specifications	33
7.2	Dimensions.....	34
7.3	Connector Pinouts.....	35
7.3.1	Motor-Connector.....	35
7.3.2	I/O Connector	36
7.3.3	Joystick Connector	36
7.3.4	Joystick Y-cable.....	37
7.3.5	RS-232 Connectors	37
7.3.6	USB Connector.....	38
7.3.7	Power Connector & Grounding Screw	39
8	Disposal	40

1 Introduction

The Mercury™ DC 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-863 replaces the C-862 Mercury™ servo-motor controller and supplements the successful C-663 Mercury™ Step Stepper Motor Controller. 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, including Mercury™ Step
- 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-863.10 Mercury™ Controller

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 C-863 Mercury™ DC motor controller shares its native programming language with its predecessor, the C-862 and with the Mercury™ Step stepper motor controller. Up to 16 Mercury™ controllers (DC and stepper) can be daisy chained and operated from one computer interface.

Flexible Automation

The C-863 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™ 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-863 Mercury™ Controller is intended to drive PI stages with DC motor or voice coil drives.

Observe the safety precautions given in this User Manual.

Operation other than instructed here may affect the safeguards provided.

C-863s conform to Measurement Category I (CAT I) and may not be used for Measurement Categories II, III or IV. Other use of the device (i.e. operation other than instructed in this Manual) may affect the safeguards provided.

C-863s meet the specifications of EN 61010 and are designed to operate under normal ambient conditions at least as listed here. More stringent conditions given in the specifications table on p. 33 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

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-863 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

Install and operate the C-863 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.



CAUTION

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

CAUTION

Never connect a stepper motor to a C-863 DC motor controller. Irreparable damage could result.



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

The voltage output on the “Motor +” and “Motor -” pins can be as high as the supply voltage used. If using a non-standard power supply for the Mercury™ and a motor that connects to these lines (i.e. without separate PWM amplifier), make sure the motor’s operating voltage will not be exceeded.



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—Maintain Compliance

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. Connection to a protective ground is required for compliance with applicable standards.



1.4 Unpacking

Unpack the C-863 Mercury™ controller with care. Compare the contents against the items covered by the order and against the packing list.

In C-863.10, the following components are included:

- Mercury™ Controller (C-863.10)
- Wide-range 15 V power supply (C-890.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 (14651) (EMI-suppression ferrite no longer required)
- Mercury™ Product CD with all software and manuals for Mercury™ Class products
- MS173E User Manual (this document) 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-170.PB	Pushbutton box with 4 buttons and 4 LEDs
C-170.IO	Connector for I/O socket (p. 36), with cable, open end
C-663.PS	24 V power supply, primarily for use in systems with 24 V DC motors not having PWM amplifiers; do not use with 12 V motors

1.6 Software Interfaces

The C-863 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). The DLL has many command-oriented functions, making it unnecessary to generate the GCS commands as ASCII strings. 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-863 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.

- *PI Terminal* 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)

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-863 from the National Instruments' LabVIEW environment (not included) using the Mercury™ GCS-DLL (see MS149E for details).

** 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.

- 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.


2 Quick Start

This Quick Start 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.

2.1 Front and Rear Panel Elements



Fig. 2: Front view of C-863 Mercury™


Front		
Labeling	Element Type	Purpose
RS-232 In	Sub-D 9(m)	Command routing (connects to PC or a networked Mercury™ class controller)
RS-232 Out	Sub-D 9(f)	Command routing in network (connects to a networked Mercury™ class controller)
	Mini-B type USB	Command routing (connects to host PC, do not connect when RS-232 IN is connected)
STA	LED green	Power and status
ERR	LED red/green	Command error: green: command OK red: command 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. 23)

CAUTION

The voltage output on the “Motor +” and “Motor -” pins can be as high as the supply voltage used. If using a non-standard power supply for the Mercury™ and a motor that connects to these lines (i.e. without separate PWM amplifier), make sure the motor’s operating voltage will not be exceeded.



Fig. 3: Rear view of C-863 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)
DC Motor only	Sub-D 15(f)	Motor/stage connection (I/O): DC motor only!
	Screw & washer	Protective ground connection

CAUTION

Never connect a stepper motor drive to a C-863 DC-motor controller. Irreparable damage could result.

2.2 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.

NOTE

Windows NT does not have USB support as standard. PI supports only the RS-232 interface for Windows NT. Unless you manage to get USB interfacing operational, the number of networkable devices may be limited to as few as 6 units by the current-sourcing capacity of the PC's COM port output stages.

Power up the C-863 by connecting it to a suitable power supply. The C-863 is ready for operation when the green power LED (STA) lights up.

2.2.1 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.

2.2.2 USB from Host

When you connect via the USB interface for the first time, be sure you are logged on the PC as a user having administrator rights. After the C-863 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.

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.

2.3 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. 8.

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

2.3.1 USB Driver

If using the USB interface, the USB driver must be installed as described in the section on connecting the USB interface, Section 2.2.2 above.

2.3.2 GCS Software

If you wish to use any of the GCS-based software, the Windows Setup installation procedure must be run. Proceed as follows:

- 1 Be sure to log in with administrator privileges and insert the CD in the host PC.
- 2 Insert the Mercury™ CD; if the Setup Wizard does not start automatically, start Setup from the root directory of the CD.
- 3 Follow the on-screen instructions. You can choose between *typical* and *custom* installation. Components covered by *typical* are LabVIEW™ drivers, DLLs and *PI*MikroMove™. *Typical* is recommended.

2.3.3 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; administrator privileges are not required.
- 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.

2.4 Starting Operation

- 1 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," p. 23 for details on setting device numbers.
- 2 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. Between the first controller and the PC use either the USB cable or the null-modem RS-232 cable, but not both. Each controller on the network must be set to the same baud rate (DIP switches 5 and 6 on the C-863) and it must be the same as that assumed by the host software.



CAUTION

Never connect a stepper motor drive to a C-863 DC motor controller. Irreparable damage could result.



CAUTION

The voltage output on the "Motor +" and "Motor -" pins can be as high as the supply voltage used. If using a non-standard power supply for the Mercury™ and a motor that connects to these lines (i.e. without separate PWM amplifier), make sure the motor's operating voltage will not be exceeded.

- 3 Connect each motorized axis to the corresponding controller.



CAUTION

Always respect the power supply voltage markings. Never connect a 24 V power supply to an old (black) C-862 Mercury™

- 4 Connect a protective ground to the grounding screw on the rear panel
- 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.
- 6 The status LED (label STA) will glow green for normal operation.

- 7 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 *PIMikroMove™*, 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!

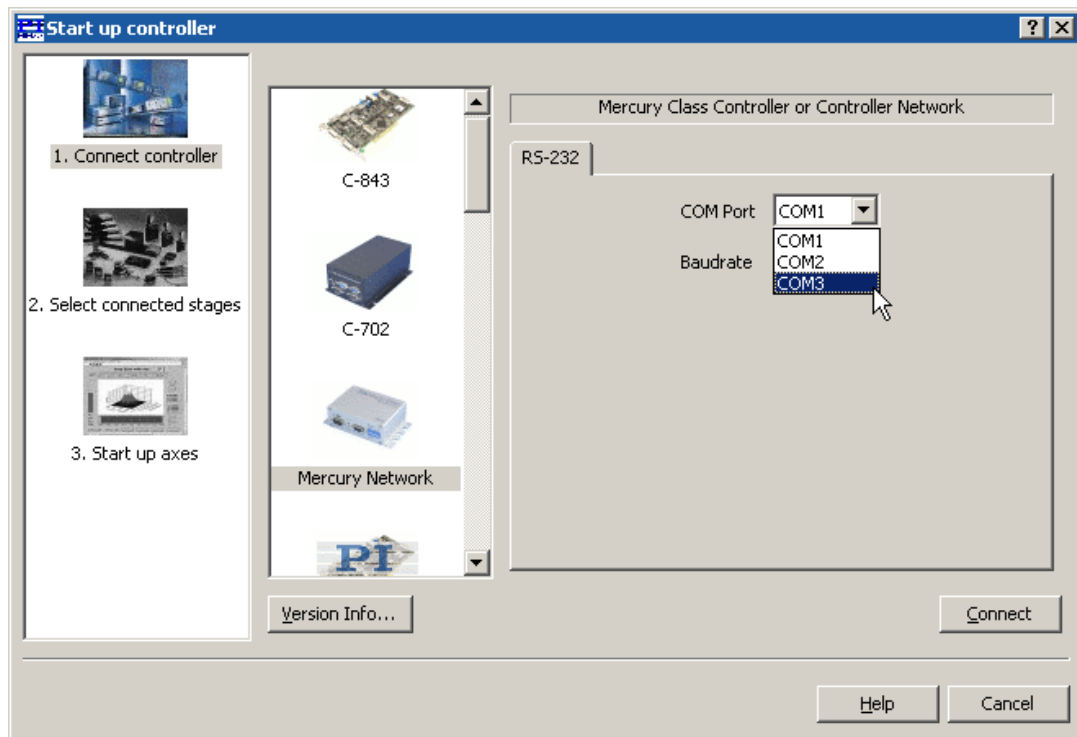


Fig. 4: *PIMikroMove™* Start up Controller screen, Step 1 with “extra” COM port for USB connection being selected

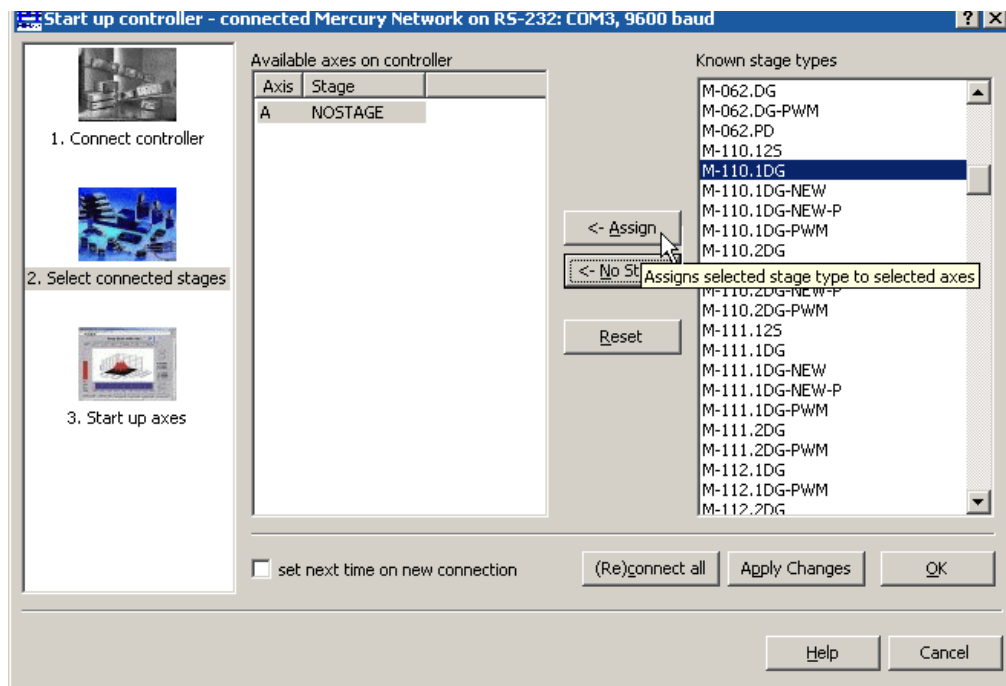


Fig. 5: PIMikroMove™ Start up controller: Step 2: select stage and be sure to click Assign before moving to Step 3 with OK

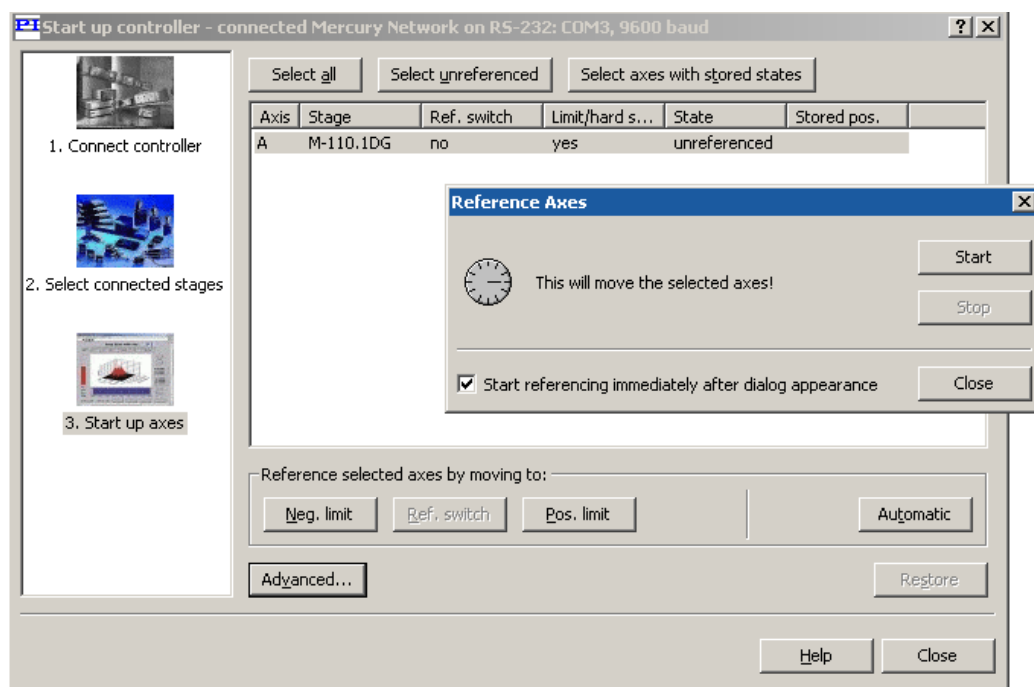


Fig. 6: PIMikroMove™ Start up controller, Step 3: Reference the axis

- 8 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. 5) . Otherwise it may be necessary to set the

stage parameter values directly. Appropriate starting values are given in the User manual for the stage.

- 9 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. 6), 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.

3 System Description

3.1 Control of Multiple Axes

C-863 and C-862 Mercury™ controllers can be networked with each other or with C-663 Mercury™ Step stepper 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.* Each controller on the network must be set to a unique address. With the C-863, the controller address is set in the first four DIP switches on the front panel (see p. 23).

The networking feature does not support simultaneous ASCII communication with the various controllers, but simultaneous motion of the connected axes, servo-control 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 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.

3.2 LED Status Indicators

When power is to the controller is turned on, the status LED (labeled STA) will glow green for normal operation.

The lower LED (labeled ERR) signalizes command errors and should be off unless a command syntax error was found. The error status is cleared when the error status is read by command.

*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.

3.3 Power-up Settings

The C-863 Mercury™ has the following power-up settings:

Setting	Power-up value	Unit
Velocity:	45000	counts/s
Acceleration:	400000	counts/s ²
P-term:	35	
I-term:	0	
D-term:	0	
I-Limit:	2000	
Limit logic level	active high	
Limit enable	limits enabled	
Brake mode	brake on	

Values can be set at any time by command (e.g. in a macro). *PI MikroMove™* 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 from 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-up defaults directly.

3.4 Cable Connections

- The motor control signals are all on the 15-pin sub-D “DC Motor” connector; they include the motor control lines (outputs) and the encoder and limit switch signals (inputs). This connector is compatible with all PI stages having DC motors and current voice coil drives. Stages with detachable cables come with the C-815.38 Motor Cable for connection to the controller. Stepper motor drives use the same connector but are not compatible. Permanent damage can occur if stage and controller types are not compatible.
- The host PC connects to one of the Mercury™ Class controllers on the network via RS-232 or USB. The “RS-232 OUT” connector of that controller can be used to connect an additional controller with a straight-through networking cable (order number C-862.CN). Up to 16 Mercurys™ can be

daisy-chained from RS-232 OUT to RS-232 IN in this manner^{*}. Note that unique controller addresses must be set for network operation, but they do not need to be in any particular order.



Fig. 7: C-862.CN Mercury™ Class network cable, length 28 cm, other lengths available on request

- Power (15 to 30 VDC, 15 V recommended) must be supplied at the power in connector. Be sure never to connect a 24 V power supply to the older 12-15 V, C-862 Mercury™ DC-motor controller.



CAUTION

The voltage output on the “Motor +” and “Motor -” pins can be as high as the supply voltage used. If using a non-standard power supply for the Mercury™ and a motor that connects to these lines (i.e. without separate PWM amplifier), make sure the motor’s operating voltage will not be exceeded.

3.5 Limit Signals

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 high 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

^{*} The RS-232 output circuitry of some PCs may not be capable of driving more than 6 Mercurys™; if this is a problem use USB to interface with the PC.

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 DC motor stages have active-high limit switches, so SW 7 must be OFF.

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™ DC motor 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 a details of commands which can use the reference signal.

3.7 Input/Output Lines

C-863 Mercury™ controllers offer 4 digital outputs and 4 inputs for either digital or analog (8-bit resolution) use (see p. 36 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 and joystick inputs. This makes possible pushbutton control and/or inter-axis triggering in stand-alone mode, i.e. without a PC connection.

3.8 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

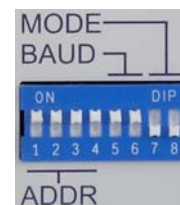


Fig. 8: slider up ON
slider down OFF

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	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; <i>MMCRun</i> 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

NOTE

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".

3.9 Firmware Update

It is possible to update the Mercury™ 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 "C863" and has the ".hex" extension (rename it, if necessary)
5. Start the MMC_update program

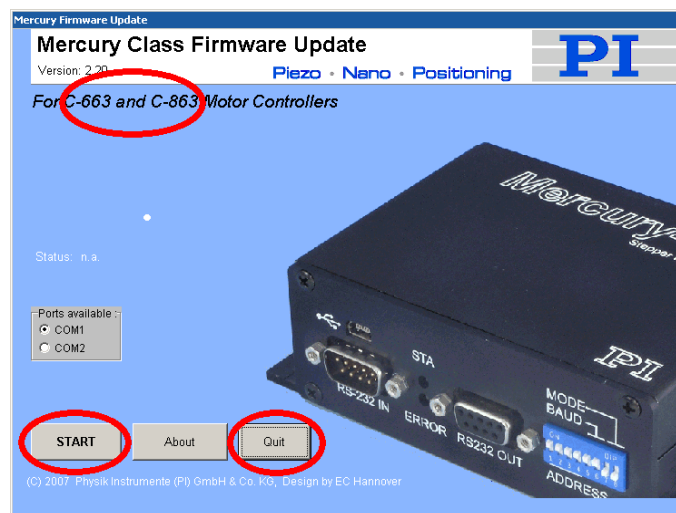


Fig. 9: MMC_update opening screen; supported controller models circled

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.

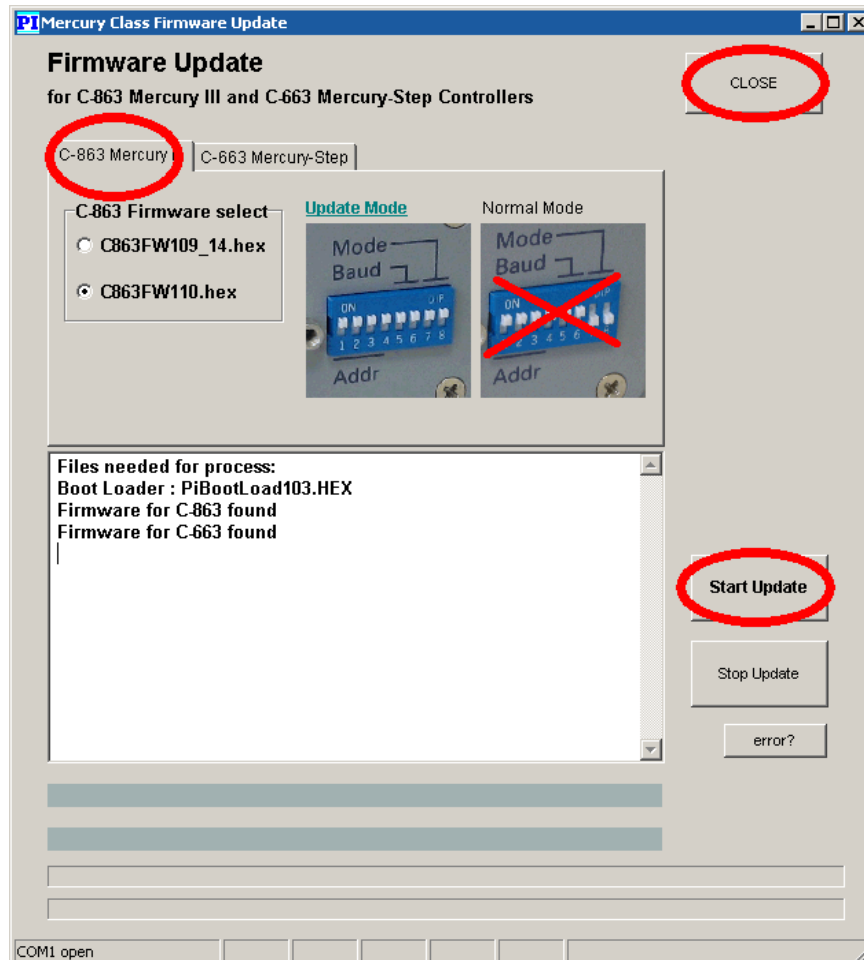


Fig. 10: MMC_update main screen; with C-863 tab and Start Update button circled

9. Make sure the C-863 tab card is on top.
10. Make sure the proper firmware file is selected in the *C-863 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 Manual Control

4.1 Joystick



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.

C-863 Mercury™ motor controllers offer convenient manual motion control by using analog joysticks (order number C-819.20). With joystick mode enabled, any number of current-production Mercury™ class controllers can be joystick-operated, even while another Mercury™ class controller in the network is under computer control. The two axes of each joystick can be connected through the C-819.20Y cable to two different Mercury™ class controllers.

NOTE

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

When a joystick is connected directly to the controller (not to the host PC), it is the *velocity* (not the position or motion of the target) of the controlled axes that is determined by the joystick position. For use of the joystick on the host PC, see the corresponding software manual (e.g. SM148 for *PIMikroMove*™).

NOTE

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 MMCRun, this procedure is facilitated by clicking *Adjust* in the *Joystick* pane. See the *MMCRun* software manual for details.

Joystick control accessories:

- C-819.20 Analog Joystick
- C-819.20Y Cable to connect one Joystick to two current-production Mercury™ Class controllers. Each controller sees only one axis and the states of that axis' buttons

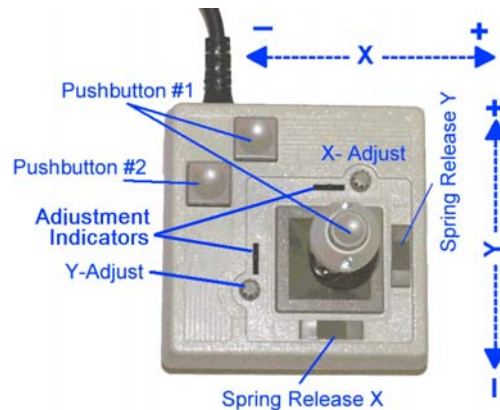


Fig. 11: C-819.20 Joystick for Mercury™ Controllers

4.2 Pushbuttons

The C-170.PB pushbutton box connects to the Mercury™ I/O connector. It allows applying TTL signals to the input lines and displays the state of the output lines on LEDs.



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

5 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.

6 Troubleshooting

NOTE

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

Problem (1):	When the Mercury™ 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) ■ Power supply with suitable voltage is connected and operating properly
Problem (2):	Axis starts to move before you send any move commands.
Possible cause:	<ul style="list-style-type: none"> ■ The autostart macro is active. Solution: Erase the autostart macro ■ Joystick is connected but not calibrated. See Section 4 on p. 27 for details ■ Joystick was activated but no joystick is connected. Disable joystick
Problem (3):	<p>When the PI host software starts, no communication with the Mercury™ can be established. Error message "No controller found" or "Could not connect..."</p>
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) ■ Upper LED lights green? If not, see problem (1) ■ Firmware has current version level ■ For RS-232 connection: C-815.32 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 (see Native Commands manual for explanation) ■ 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 OFF for active-high, for standard PI DC motor drives)
Problem (6)	Commands are not recognized.
Solution	<ul style="list-style-type: none"> ■ In MMCRUN, 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 (CR for native, LF 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, commands: Lx, 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 OFF for active-high, for standard PI DC 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 the respective Software manual for details</p>

7 Technical Data

7.1 Specifications

	C-863.10
Function	DC motor servo-controller, 1 channel
Drive type	DC motor or voice coil
Channels	1
Motion and control	
Controller type	P-I-D servo-controller
Trajectory profile modes	Trapezoidal, point-to-point
Encoder input	A/B quadrature TTL level, differential as per RS-422; 20 MHz
Stall detection	Automatic motor stop when programmable position error is exceeded
Limit switch inputs	2 x TTL signals, programmable logic & hardware interlock
Reference (origin) switch input	1 x TTL signal
Motor brake output	1 x TTL out, programmable
Electrical properties	
Output power	3 A x supply voltage, or as limited by power supply (30 W with standard 15 V, 2 A C-890.PS)!
Output voltage	0 to \pm supply voltage (15 V with standard C-890.PS); motor used must be compatible with supply voltage
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	Mercury™ GCS (PI General Command Set) DLL, GCS LabVIEW™ drivers, native Mercury™ DLL, native LabVIEW drivers

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.

Manual control (optional)	Joystick, Y-cable for 2D-motion; pushbutton box
Miscellaneous	
Operating voltage	15 to 30 VDC, 15 V recommended, if higher voltage used, connected motor must be compatible
Current draw	80 mA + motor current (3 A max.)
Operating temperature range	5 to 50°C
Mass	0.3 kg
Dimensions	130 x 76 x 40 mm

7.2 Dimensions

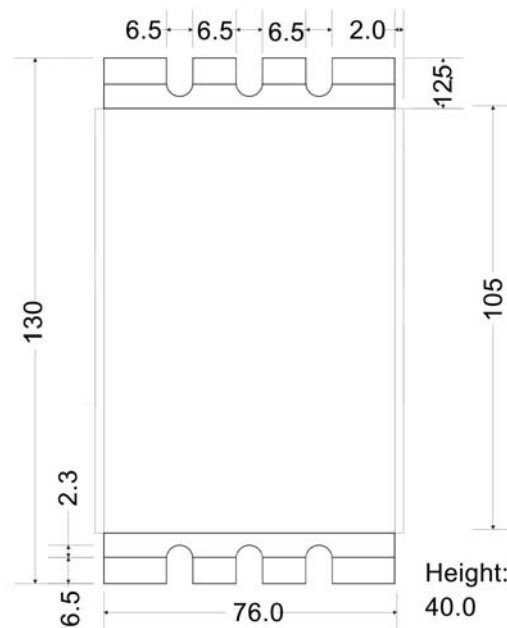


Fig. 13: C-863 dimensions in millimeters

7.3 Connector Pinouts

7.3.1 Motor-Connector

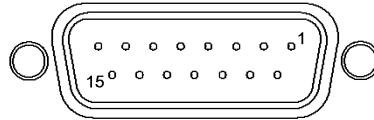


Fig. 14: Sub-D 15(f) motor connector

Pin	Pin	Function
1		Programmable output (brake control) (0 or + 5V)
	9	Motor – (differential, power PWM*)
2		Motor + (differential, power PWM*)
	10	Power GND
3		PWM magnitude (TTL-level output)
	11	PWM sign (motion direction, TTL-level output)
4		+5 V (output)
	12	Negative limit signal (input)
5		Positive limit signal (input)
	13	Reference signal (input)
6		Limit GND
	14	Encoder: A(+) / ENCA (input)
7		Encoder: A(-) (input)
	15	Encoder: B (+) / ENCB (input)
8		Encoder: B (-) (input)

* PWM (pulse-width modulation) ON-voltage magnitude on these lines is close to the supply voltage.



CAUTION

The voltage output on the “Motor +” and “Motor -” pins can be as high as the supply voltage used. If using a non-standard power supply for the Mercury™ and a motor that connects to these lines (i.e. without separate PWM amplifier), make sure the motor’s operating voltage will not be exceeded.

7.3.2 I/O Connector

Pin	Signal	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

7.3.3 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

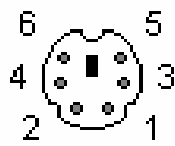


Fig. 16: Mini DIN 6-pin (PS2-style) joystick connector

7.3.4 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 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

7.3.5 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 RxD (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)


7.3.6 USB Connector

Industry-standard mini type-B USB connector.

7.3.7 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—Maintain Compliance

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. Connection to a protective ground is required for compliance with applicable directives.

8 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

