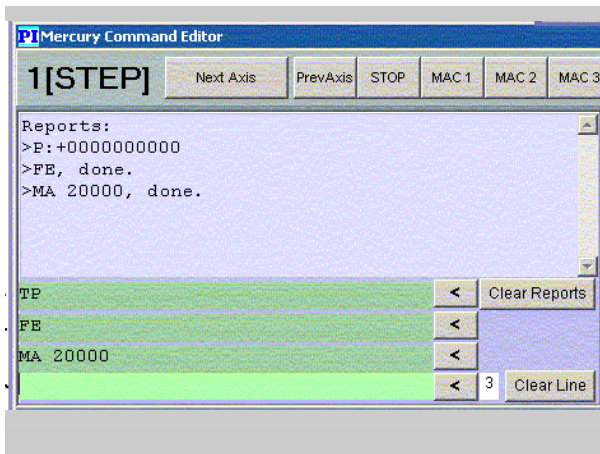


MS176E Software Manual

Native Commands

for Mercury™ Class Controllers

Release: 1.1.0 Date: 2009-08-07



This document describes software for use with the following products:

- C-663
Mercury Step™ Networkable Single-Axis Stepper Motor Controller
- C-863
Mercury™ Networkable Single-Axis DC-Motor Controller



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

Document Number MS176E,, Eco, Bro, Release 1.1.0
MercuryNativeCommands_MS176E.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 This manual is superseded by any new release.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.

About This Document

Users of This Manual

This manual assumes that the reader has a fundamental understanding of basic servo systems, as well as motion control concepts and applicable safety procedures.
This manual is designed to help the reader operate Mercury™ Class controllers using the native command set, including native controller macros.

This document is available as PDF file on the Mercury 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:

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 Mercury™ controller and the software tools which might be delivered with the controller are described in their own manuals (see below). All documents are available as PDF files via download from the PI Website (<http://www.pi.ws>).. For updated releases or other versions contact your Physik Instrumente Sales Engineer or write info@pi.ws.

User Manuals for hardware	Give dimensions, connections and specifications of the hardware components
MMCRun MS139E	Mercury Operating Software (native commands)
Mercury Native DLL & LabVIEW MS177E	Windows DLL Library and LabView VIs (native-command-based)
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 Commands MS163E	Mercury™ GCS Command descriptions
PIStageEditor _SM144E	Software for managing GCS stage-data database

1	Introduction	3
1.1	Software for Native Commands	3
1.1.1	Software Overview	3
1.1.2	Installation	4
1.1.3	Updates	4
1.2	Units in Native Commands	4
2	Quick Start	5
3	Operation	7
3.1	Addressing	7
3.1.1	Sending the Address Selection Code	7
3.1.2	Automatic Address Selection at Power-On	8
3.1.3	Address Selection Codes	8
3.2	Networking	9
3.3	Testing Communication	9
3.4	Setting Motion Control Parameters	10
3.4.1	Default Parameters for C-863 DC-Motor Controllers	10
3.4.2	Default Parameters for C-663 Stepper Motor Controllers	11
3.4.3	How to Set Motion Control Parameters	11
3.5	Operating Motors and Stages	13
3.5.1	Operating C-863 DC-Motor Controllers	13
3.5.2	Operating C-663 Stepper Motor Controllers	14
3.6	Joystick Control	15
3.6.1	Handling	15
3.6.2	Joystick Response Definition Tables	16
3.7	Trackball Control	18
3.8	Macro Storage on Controller	19
3.8.1	Defining a Macro	19
3.8.2	Executing (Starting) a Macro	19
3.8.3	Stopping a Macro	20
3.8.4	Limitations	20
3.8.5	Macro #0 (autostart macro)	21
3.8.6	Stand-Alone Operation Examples	21
4	System Details	29
4.1	Control Options—DC-Motor Controller Only	29
4.1.1	Position Control	29
4.1.2	Force Control	29
4.1.3	Position and Force Control	30
4.1.4	Notch Filter	30
4.1.5	Commands for Closed-Loop Control—Overview	31

4.2	On-Target Detection	31
4.3	Position Referencing	32
4.4	Limit Signals	32
4.4.1	Limit Switches.....	32
4.4.2	Soft Limits	33
4.5	Input/Output Lines	34
4.5.1	Overview.....	34
4.5.2	Trigger Output	34
5	Native Command Types	36
5.1	Address Selection Codes	36
5.2	Base Commands	37
5.3	Compound Commands.....	37
5.4	Single-Character Commands	38
5.5	Conditional Commands	38
5.6	Commands with Responses.....	38
5.7	Command Execution Mode	39
5.7.1	Direct Mode	39
5.7.2	Macro Execution Mode.....	39
6	Native Command Reference	40
6.1	Command and Response Formats.....	40
6.1.1	Command Execution	40
6.1.2	Command Codes.....	40
6.1.3	Report Commands	41
6.2	Base Commands in Alphabetic Order	42
6.3	Single-Character Commands	46
6.4	Base Command Reference in Alphabetic Order	47

1 Introduction

Mercury™ Class controllers include the C-663 Mercury Step™ open-loop, stepper motor controller as well as the C-863 Mercury™ DC-motor servo-controller.

With current firmware, it is possible to operate Mercury™ controllers with two ASCII command sets: the native command set and the PI General Command Set (GCS) . GCS support is currently provided via a Windows DLL which translates GCS-command-based function calls to the native commands. Either command set can be used to set operating modes, transfer motion parameters and to query system and motion values. See the Mercury™ GCS Command manual, MS163E, for a description of the GCS command set.

This manual covers only the native command set.

Most native Mercury™ commands begin with a two-letter mnemonic. Because the commands address only the *selected* controller, they do not themselves include controller or axis designators. The syntax of the native commands and a command reference in alphabetical order can be found in Section 6.4.

1.1 Software for Native Commands

1.1.1 Software Overview

The native ASCII command set is understood by the current Mercury™ firmware directly. With Mercury™ Class controllers, all motion of the connected motors and mechanical stages is software controlled. 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 the PI Website 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 and the connection is active). *PITerminal* also handles controller selection in a Mercury™ network (by device numbers ranging from 1 to 16). When used without the GCS DLL, firmware-native commands can be used with the connected controllers .
- *MMCRun* (operating software for Windows 95/98/2000/XP, NT and Vista) is the operating software for C-863 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 manual MS139 for a full description.

- MMC410.DLL: Windows DLL (see manual MS177 for details) facilitates many interfacing operations and data conversion tasks. Based on native command set.
- LabVIEW VIs: facilitates integrating these controllers in the LabVIEW environment. Uses the native-command MMC410.DLL above. (See manual SM177 for details)

1.1.2 Installation

If using only the native-command-based software, it is possible to bypass the Setup Wizard on the Mercury™ CD and to proceed as follows:

- 1 If using the USB interface, two FTDI USB drivers have to be installed (requires administrator rights on the host PC). When the Mercury™ is connected and powered up, Windows will discover the new hardware. Follow the on-screen instructions and show the Hardware Wizard the \USB_Driver directory on the Mercury™ CD
- 2 Copy the contents of the *MMCRun* directory to the host PC. To start *MMCRun*, double-click the EXE file of the same name

1.1.3 Updates

For firmware and software updates please see the C-863 or C-663 User manual. The current firmware revision of your Mercury™ controller is contained in the response to the VE command.

1.2 Units in Native Commands

The native command set uses *counts* or *steps* to measure linear or angular distances. Their length in physical units depends on the hardware attached to the controller. Counts are defined by the integrated encoders used for position feedback; steps refer to steps commanded by stepper motor controllers (PI stepper motors are not generally equipped with encoders). Conversion values are included with the technical data specification of the connected stage.

2 Quick Start

- 1 Set the device address:
Before power-on, verify correct setting of the address DIP switches (1 to 4) on the front panel of the Mercury™ controller.
Using a single controller, set its address to zero (switches 1 to 4 in ON (upper) position). Using multiple controllers in a daisy chain network, set each controller to an unique device address. See pages 7 and 9 for details.
- 2 Connect RS-232 or USB cable (not both!).
Connect the controller to a COM port or the USB bus on the host PC. Use the cables included.
If using multiple controllers, connect the first controller to the PC and daisy chain the other controllers via the short serial cables (C-862.CN).
Note that a USB connection will appear as an extra COM port when the controller is connected, powered up, and the USB drivers are installed from the product CD. Installing the USB drivers requires administrator rights on the host PC.
- 3 Stage connection:
Connect a suitable stage to the “DC Motor only” or “Stepper Motor only” socket. Do not connect DC-motor stages to C-663 stepper motor controllers, and do not connect stepper motor stages to C-863 DC motor controllers!
Connect the stage to the stage power supply if required.
- 4 Connect the controller power supply.

C-863 DC motor controllers:
+15 V is the nominal operating voltage (provided by the included C-890.PS power supply). The controller itself may work within the range of +12 to +30 V, but most of PI's motorized mechanics are equipped with 12 V DC-motors, so the +15 V supply will allow to operate the stages within their specifications. Using drives with 24 V DC-motors, a power supply with higher output voltage should be used.

C-663 stepper motor controllers:
+24 V is the nominal operating voltage. Use the included C-663.PS power supply.
- 5 Start the native Mercury™ operating software *MMCRun* or use your own software to send commands to the controller. Read more about *MMCRun* software in the subsequent sections and in the *MMCRun* manual (MS139E).
Note that *MMCRun* requires a baud rate setting of 9600 baud (DIP switches 5 and 6 on the Mercury™ front panel must be in upper (ON) position; changes require power cycling the device).

NOTES

If you use your own communication software, the controller has to be addressed before it responds to any commands. See "Addressing" on p. 7 for details. Addressing is done automatically if using the *MMCRun* operating software or the *PITerminal*.

The time lag for characters must not exceed 20 ms for firmware version 1.xx and 700 ms for firmware version 2.xx.

3 Operation

3.1 Addressing

Mercury™ controllers must be addressed (*selected*) to be able to respond to commands. After power-on or reset, a Mercury™ controller is not addressed (*deselected*).

You have two options for addressing:

- Send the valid address selection code over the communications interface, see Section 3.1.1
- For single controllers only: define an autostart macro containing an SC command (Select Controller, p. 72), see Section 3.1.2

The address is given by the “Addr” DIP switches (1 to 4) on the controller front panel. Addresses from 0 to 15 can be selected using the switches, see Section 3.1.3 below for the address code list.

Each controller in a daisy chain network has to be set to a different address using the DIP switches, and must be addressed with the corresponding unique address selection code.

A controller will remain *selected* until *deselected*. Deselection is done when receiving an address selection code with a different address sent over the communication interface, or when the controller is powered down or reset.

NOTE

Addressing is done automatically if using software from PI, e.g. the *MMCRun* operating software.

3.1.1 Sending the Address Selection Code

The address selection code consists of two characters. The first character is ASCII character #1, and the second character corresponds to the settings of the address DIP switches (1 to 4) on the front panel. Both characters are sent without additional termination character.

Example: If all 4 address switches are in upper (ON) position, then the address is set to 0, representing the ASCII character #48 (0x30). In this case the address selection code consists of the two characters ASCII 1 (no printable representation) and ASCII 48 (printable representation "0").

3.1.2 Automatic Address Selection at Power-On

If a single Mercury™ controller is to be powered up with enabled communication, there is a special command that can be used in the autostart macro (macro #0; MD0):

SC n (Select Controller n ; where n is the address given by the DIP switches 1 to 4, ranging from 0 to 15)

The SC n command (p. 72) in the autostart macro offers self-addressing at power-on. This might be helpful if the host PC is not capable to send none printable ASCII characters, or if no host PC is present at all.

This method can not be used with networked controllers.

3.1.3 Address Selection Codes

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

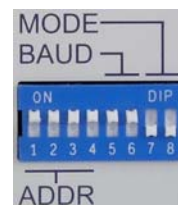


Fig. 1: slider up ON
slider down OFF

Factory settings are shown in bold. Changes require power cycling or resetting the device.

Address	DIP switch settings				Address selection code	
	SW1	SW2	SW3	SW4	1 st character	2 nd character
0	ON	ON	ON	ON	ASCII 01	ASCII 48
1	ON	ON	ON	OFF	ASCII 01	ASCII 49
2	ON	ON	OFF	ON	ASCII 01	ASCII 50
3	ON	ON	OFF	OFF	ASCII 01	ASCII 51
4	ON	OFF	ON	ON	ASCII 01	ASCII 52
5	ON	OFF	ON	OFF	ASCII 01	ASCII 53
6	ON	OFF	OFF	ON	ASCII 01	ASCII 54
7	ON	OFF	OFF	OFF	ASCII 01	ASCII 55
8	OFF	ON	ON	ON	ASCII 01	ASCII 56
9	OFF	ON	ON	OFF	ASCII 01	ASCII 57
10	OFF	ON	OFF	ON	ASCII 01	ASCII 65
11	OFF	ON	OFF	OFF	ASCII 01	ASCII 66
12	OFF	OFF	ON	ON	ASCII 01	ASCII 67
13	OFF	OFF	ON	OFF	ASCII 01	ASCII 68
14	OFF	OFF	OFF	ON	ASCII 01	ASCII 69
15	OFF	OFF	OFF	OFF	ASCII 01	ASCII 70

3.2 Networking

Up to 16 Mercury™ Class controllers can be networked with each other and controlled over the same RS-232* or USB interface. The controllers interconnect with an RS-232 bus architecture. The networking feature permits addressing each controller individually, based on the hardware address set in DIP switches 1 to 4 on the controller front panel.

Switching between the controllers in a network requires sending the address selection code, consisting of two characters (see Section 3.1 on p. 7). Each controller compares the address in the address selection code with its own address. If there is a match, the controller enables command interpretation and response, so as to respond to subsequent commands. If not, it disables all responses and ignores any subsequent transmissions except address selection codes.

Any motion sequence or operation started before receiving a disabling address selection code will continue to be executed, except for commands that issue reports over the communication interface. Thus each Mercury™ controller in the network can be selected, programmed to execute a desired operation or sequence of operations, and then deselected. The same or a different command sequence can then be sent to another controller.

Communication on the interface is always between the host computer and a *selected* Mercury™, with the other Mercurys™ in the *deselected* state. There is no direct communication from controller to controller (unless the digital I/O lines are interconnected and suitably programmed). The host program must handle address selection and motion sequencing among different controllers.

3.3 Testing Communication

Initial communications tests can be performed with either *MMCRun* (if the *Edition* window opens, communication has been established, see document MS139E) or *PITerminal* (on product CD). Note that *MMCRun* requires a baud rate setting of 9600 baud (DIP switches 5 and 6 on the Mercury™ front panel must be in upper (ON) position; changes require power cycling the device).

If you are using your own software, first send the proper *address selection code* to “wake up” the desired device (see “Addressing” on p. 7 for details). Then send the VE command (p. 84), followed by CR as termination character, to read the firmware version information.

With *MMCRun* just enter the VE command in the command line editor and press <ENTER> on your keyboard. Alternatively, you can click the *IDN?* button.

*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.

3.4 Setting Motion Control Parameters

Motion parameters are needed to adapt the Mercury™ control behavior to a specific stage or drive in order to get smooth motion and small position errors.

Mercury™ controllers come with factory-set default parameters. These values are active after power-on or reset as long as no autostart-macro is stored that overwrites them.

3.4.1 Default Parameters for C-863 DC-Motor Controllers

Parameter	Command to set this parameter	Factory Default Value
Velocity:	SV	45000 counts/s
Acceleration:	SA	400000 counts/s ²
P-term for position control:	DP	35
I-term for position control:	DI	0
D-term for position control:	DD	0
I-Limit for position control:	DL	2000
Control mode	FM	0 (position control)
P-term for force control:	FP	35
I-term for force control:	FI	0
D-term for force control:	FD	0
I-Limit for force control:	FL	2000
Low pass filter for additional sensor input	FT	10000 Hz
Notch filter frequency	FN	10000 Hz
Notch filter edge	FQ	0.8
Limit logic level:	LH, LL	LH (active high)
Limit enable:	LN, LF	LN (limits enabled)
Brake mode:	BN, BF	BN (brake on)
Upper soft limit:	JH	+100,000,000 counts
Lower soft limit:	JL	-100,000,000 counts
Settle time	WT	0 ms (i.e. on-target flag is set when calculated trajectory has finished)
Settle window	WW	5 counts

3.4.2 Default Parameters for C-663 Stepper Motor Controllers

Parameter	Command to set this parameter	Factory Default Value
Velocity:	SV	20000 steps*/s
Acceleration:	SA	250000 steps*/s ²
Current while moving:	DC	200 mA
Current while holding:	HC	100 mA
Time to hold:	HT	200 ms
Active limit level high/low:	LH, LL	LL (active low)
Limit ON/OFF:	LN, LF	LN (limits enabled)
Brake ON/OFF:	BN, BF	BN (brake on)

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

3.4.3 How to Set Motion Control Parameters

In most cases, the factory default values (see Sections 3.4.1 and 3.4.2) will not be suitable for the specific motor-driven mechanics used in your application. It is recommended that you overwrite the values with those suggested in the User manual of the mechanics.

Proper motion control parameter settings depend on the individual stage and drive connected. If a parameter set is found, it can be stored as a macro command for automatic execution upon power-on.

The following example shows how to change the parameter values for a C-863 DC-motor controller, but can—with appropriate parameters—also be applied for C-663 controllers.

Option 1: Set parameters "by hand"

Enter commands in the "Mercury Command Editor" of *MMCRun*:

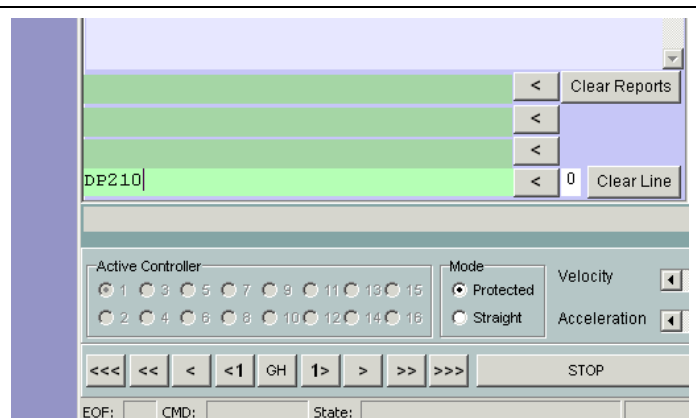
DP210 <Enter>

DI45 <Enter>

DD250 <Enter>

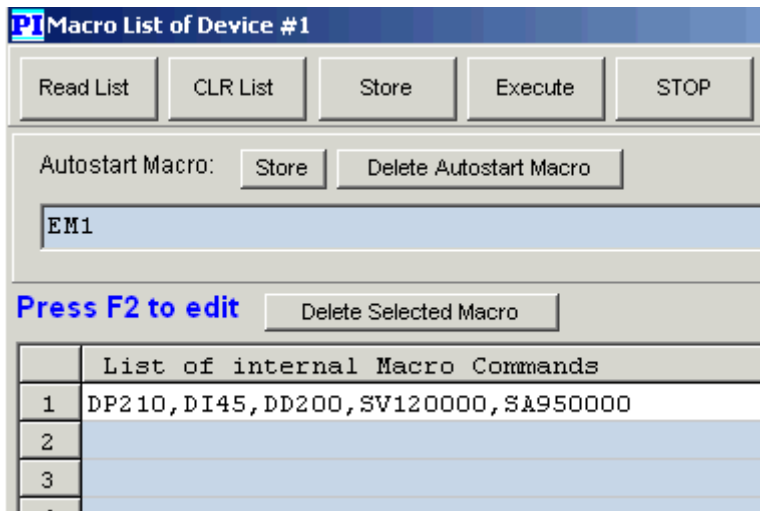
SV120000 <Enter>

SA950000 <Enter>



Option 2: Set parameters using an autostart macro

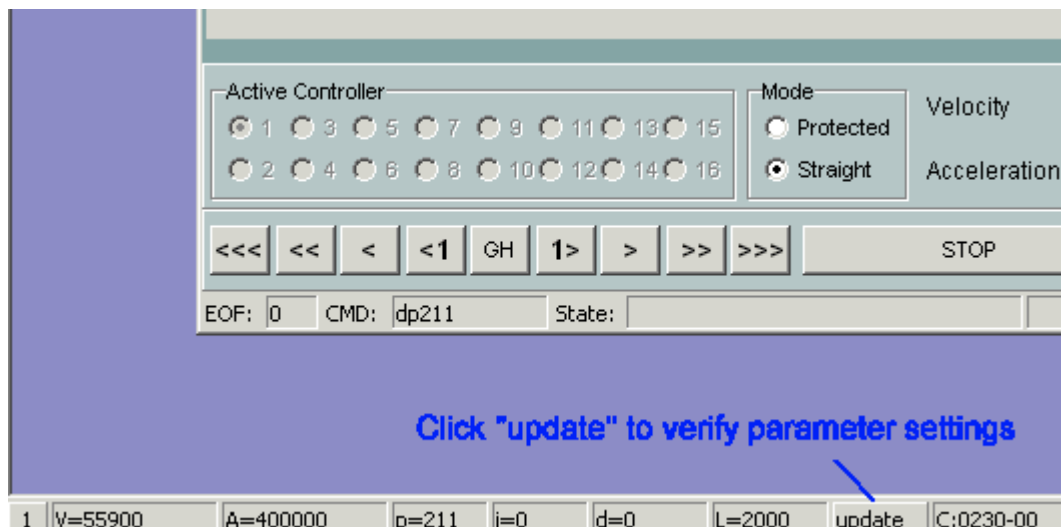
In *MMCRun*, define the parameter list as a macro command (macro 1 in the example). Then define an autostart macro which executes the parameter macro upon power-on (EM1 in this example). Press <Enter> on your keyboard to store.



List of internal Macro Commands	
1	DP210,DI45,DD200,SV120000,SA950000
2	
3	
4	

Now the controller is set up to load the proper motion parameters upon power-on. To make the parameters effective immediately, type EM1 (execute macro #1) in the command line or click the "Execute" button while the cursor is placed in the macro line #1.

Check the parameter settings displayed in the button status bar:



Click "update" to verify parameter settings

1	V=55900	A=400000	p=211	i=0	d=0	L=2000	update	C:0230-00
---	---------	----------	-------	-----	-----	--------	--------	-----------

If you do not use the *MMCRun* software, define the macro commands as follows:

Step1 : Define macro #1:

Store the parameters in macro #1:

MD1,DP210,DI45,DD250,SV120000,SA950000

Step 2: Define macro #0 (autostart macro):

Macro #0 calls macro #1

MD0,EM1

3.5 Operating Motors and Stages

3.5.1 Operating C-863 DC-Motor Controllers

When the controller is selected and the motion parameters are set, turn the motor on by sending the "MN" command (enables also the servo mode, i.e. closed-loop control). The color of the STA LED should change from red to green.

The Mercury™ controller now is ready for motion. It is recommended to start some test moves of the stage as described below.

Enter "MR1000". The stage should move 1000 encoder counts in positive direction. When it stops, enter "TP" to ask for the actual position. The position should be very close to 1000. Now enter "TT,TE" in order to read the target position and the position error. The TT (Tell Target) command reports where the stage should be (1000), and the TE (Tell Error) command reports the difference between the actual position and this target. If, for example, the reported position is 998, then the reported error will be -2. If the reported position would be 1002, the reported error would be 2.

To read all values at the same time, enter "TT,TP,TE". Now all three values are reported.

Now enter "MR-1000". The stage should move back to the zero position, also referred to as home position. Press the Enter key again. The stage should move another 1000 counts in negative direction. Press the Enter key again and again. The stage should move each time.

Enter "GH". The stage should return to the position where it was located at power-on (a "GH" command starts a motion to the home (zero) position, and upon power-on, the actual position of the stage is set to zero). While the stage is still in motion, enter the command "TE,TP" repeatedly. The difference between the actual position and the target is displayed, and you can watch how the stage approaches the target. Alternatively, you can enter "TP,WA100,RP" once.

Advanced programming: enter "MR1000,WS100,TE". The stage should move 1000 counts in positive direction. When the motion has finished, the position error (distance between actual position and target position) is reported.

To move the stage over a longer distance, enter "MR100000". Verify that the motion was performed successfully. Then press the Enter key again and verify that the stage moves the same distance in the same direction. Repeat this several times. Motion should continue each time the Enter key is pressed.

Set the acceleration to a lower value, such as "SA10000", and watch the velocity increasing and decreasing over a longer period. Set the velocity to various values and send MA or MR move commands with various targets.

Move the stage some distance away from the end, e.g. 100,000 counts, and try this command sequence:

```
MR50000,WS100,WA500,MR-50000,WS100,WA500,RP9.
```


The stage should move back and forth by 50,000 counts with a delay of half a second at each end. It should do this ten times (the command sequence is executed and then repeated nine times).

3.5.2 Operating C-663 Stepper Motor Controllers

When the controller is selected and the motion parameters are set, turn the motor on by sending the "MN" command. Note that C-663 stepper motor versions may have lost track of the position if they were in the "MF" state.

The Mercury™ controller now is ready for motion. It is recommended to start some test moves of the stage as described below.

Enter "MR1000". The stage should move 1000 steps in positive direction. When it stops, enter "TT,TP" to ask for the target position and the actual position. The actual position should be very close to 1000.

Now enter "MR-1000". The stage should move back to the zero position, also referred to as home position. Press the Enter key again. The stage should move another 1000 counts in negative direction. Press the Enter key again and again. The stage should move each time.

Enter "GH". The stage should return to the position where it was located at power-on (a "GH" command starts a motion to the home (zero) position, and upon power-on, the actual position of the stage is set to zero). While the stage is still in motion, enter the command "TT,TP" repeatedly. You can watch how the stage approaches the target. Alternatively, you can enter "TP,WA100,RP" once.

Advanced programming: enter "MR1000,WS100,TT,TP". The stage should move 1000 counts in positive direction. When the motion has finished, target position and actual position are reported.

To move the stage over a longer distance, enter "MR100000". Verify that the motion was performed successfully. Then press the Enter key again and verify that the stage moves the same distance in the same direction. Repeat this several times. Motion should continue each time the Enter key is pressed.

Set the acceleration to a lower value, such as "SA10000", and watch the velocity increasing and decreasing over a longer period. Set the velocity to various values and send MA or MR move commands with various targets.

Move the stage some distance away from the end, e.g. 100,000 steps, and try this command sequence:

MR50000,WS100,WA500,MR-50000,WS100,WA500,RP9.

The stage should move back and forth by 50,000 steps with a delay of half a second at each end. It should do this ten times (the command sequence is executed and then repeated nine times).

3.6 Joystick Control

3.6.1 Handling

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.

Mercury™ motor controllers offer convenient manual motion control by using analog joysticks connected to the “Joystick” socket. With the joystick mode enabled, both *selected* and *deselected* Mercury™s can be joystick-operated. When using an C-819.20 joystick device, its axes can be connected through the C-819.20Y cable to two compatible Mercury™ controllers. C-819.30 3-axis analog joysticks are equipped with separate connection cables for three controllers.

When a joystick is connected directly to a 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.

NOTES

If the Y-cable is used to connect a joystick to two controllers, the X-branch must be connected to a powered-up controller because joystick power is taken from that side.

Before a joystick can be operated correctly, a calibration routine may need to be performed. Activating the joystick before calibration may cause the motor to start moving even though the joystick is in the neutral position.

To calibrate the joystick X- or Y-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.

To calibrate the joystick Z-axis of C-819.30 3-axis joystick devices, a special procedure is required. See the Mercury™ User manual for details.

Commands for joystick handling:

Command	Function	Corresponding Report Command
JNn	Enables the joystick operation with a specified maximum velocity n. Range for n : 1000 to 500000 (unit is counts or steps per second)	JN?

	<p>Example: JN30000</p> <p>After this command is sent, the joystick is active. At full tilt angle the motor speed is 30,000 steps/s, as indicated.</p> <p>While joystick is active, move commands and trackball input are not accepted.</p>	
JF	Disables the joystick and reactivates processing of move commands	
JBn	Adds an offset n to the analog input provided by the joystick.	JB?
JTn	<p>Generates Predefined Joystick Response Table n.</p> <p>n = 0 Linear response table n = 1 Cubic response table n = 2 Linear response table inverted n = 3 Cubic response table inverted</p> <p>The cubic curve offers more sensitive control around the middle position and less sensitivity close to the maximum velocity.</p> <p>By default, the linear table n = 0 is loaded.</p>	
JHn	<p>Set upper soft limit</p> <p>Motion in positive direction does not exceed that position.</p>	JH?
JLn	<p>Set lower soft limit.</p> <p>Motion in negative direction does not exceed that position.</p>	JL?
TA5	Read joystick axis position, analog, from 0 to 255	
TA6	Read joystick button state, analog, from 0 to 255, high values indicate button pressed	

3.6.2 Joystick Response Definition Tables

Normally the joystick response is defined by loading a predefined response table using the JT command. The loaded response table will remain chosen even when the controller is reset.

For special purposes, it is possible to generate a user-defined joystick response table.

Commands for user-defined joystick tables (see Command Reference section for more details):

Command	Function	Corresponding Report Command
SI_n	Sets index pointer for joystick table. A new table value can then be written with the SJ command.	SI? Ask for the current index value
SJ_n	Sets the table value at active index to n. Range of n : 0 to 1024 Value 1024 represents maximum velocity, 0=standstill.	SJ? Ask for the current table entry.

Typical command sequence:

SI0,SJ0
SI1,SJ5
SI2,SJ10
SI3,SJ15 etc.

3.7 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 see User manual) on the Mercury™ controller. The lines are terminated by 10k to GND.

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

Command	Function	Corresponding Report Command
JAn	Enable trackball mode with increment length of n. Each signal transition will shift the target by n counts. The trackball mode can be disabled by sending the JF command While the trackball mode is active, move commands or joystick control are not accepted.	JA?
SBn	C-863 DC motor versions only. Enables a filter for trackball input. The parameter n determines how often the same signal level must be read before the signal transition is accepted. This suppresses transient noise and allows for stable reading. Default value: n=0 (disabled)	SB?
JF	Disables trackball mode and returns to normal command mode.	
SVn	Set the maximum velocity n for the target shift.	
SAn	Set the acceleration n for target shifts.	

Command sequence example:

JA50 Enables trackball mode. Each trackball signal transition shifts the target by 50 counts.

JF Disables trackball mode.

3.8 Macro Storage on Controller

Macros can be a most powerful tool for the programmer. A *macro command* is a Base Command or a Compound Command stored inside the controller. Up to 32 macros can be stored in non-volatile memory on each Mercury™ controller.

The native-command macro storage facility has the following features:

- Each macro can contain up to 16 base commands
- Macros are identified by the numbers 0 to 31
- Macro 0, if defined, is the Autostart Macro, which is executed automatically upon power-on or reset
- Macros are executed on the controller where they are stored, so commands in a macro may address only the axis and/or I/O channels associated with that controller (there is no command-interface communication between controllers). Interaction between separate axes is conceivable only through suitable programming and hardwiring of I/O lines
- Once started, a controller macro can continue even if the controller is in the *deselected* state

3.8.1 Defining a Macro

To define a macro command, use an MD*n* (Macro Definition) command as the *first* instruction in a compound command followed by a comma-separated list of base commands. *n* sets the macro number.

Note that the system gives no warning if you define (and overwrite) an existing macro.

Example 1:

```
MD1,MR5000 CR
```

With this command string, the base command "MR5000" is stored as macro command #1. The move command is not executed until the macro command is executed.

Example 2:

```
MD3,MR1000,WS100,MR-1000,WS100,RP5 CR
```

With this command, the compound command "MR1000,WS100,MR-1000,WS100,RP5" is stored as macro #3.

3.8.2 Executing (Starting) a Macro

To execute (call up, run) the macro defined above, just issue the command EM3. Unlike MD# commands, EM# commands can be used anywhere in compound commands.

3.8.3 Stopping a Macro

Macro commands terminate either naturally when they have completely executed or they can be stopped when the controller is addressed and receives any character except a single-character command, **CR**, or an address selection code.

Example 3:

Assuming the macro **"MR500,WS200,RP10000"** is running and will not terminate until 10000 moves of 500 counts each are completed. If you want to stop this sequence, just send a character, e.g. "x" without **CR** and the macro stops executing.

The macro can be started again by **"EMn"** where n is the number of the macro.

3.8.4 Limitations

Up to 32 macro sequences, each holding up to 16 base commands, can be stored in the Mercury™ Class controller's flash memory.

31 macro sequences (numbers 1 to 31) are available for general use. Macro #0 is special: it is called the *autostart macro* and, if defined, is run automatically upon power-up or reset (see next section).

Macro commands may be stored in any order, but you may prefer to number them sequentially as they are entered, because the system gives no warning if you define (and overwrite) an existing macro.

A macro command can call other macros, but if the calling macro is to continue after the called macro completes, the called macro must not contain any macro calls. For example, MC1 could call MC2, but MC2 could not then call MC3 and still be able to return to complete the remainder of MC1. A macro may call as many other macro commands as desired, as long as each one called does not call another. If there is no need to return to the calling command, then macros may call macros without limitation. It is sometimes desirable to define a complex motion sequence in a macro which calls one macro to set important parameters such as torque, gain, or velocity, then calls one or more others to perform the motion.

Example: **MD1,EM2,EM3,EM4,EM5,EM6**

NOTE

A macro command can call other macros, but if the calling macro is to continue after the called macro completes, the called macro must not contain any further macro calls. Up to 32 macro sequences can be stored.

If execution of any of the commands is to be conditional, depending on a run-time condition, then those commands must be grouped together at the end of a macro (see the xxx command descriptions). This restriction makes the use of the EM (execute macro) command in macros even more useful.

3.8.5 Macro #0 (autostart macro)

Macro #0 is a special macro command. If a command sequence is stored as Macro #0, it will be executed automatically immediately after a system reset or power-up. This allows specification of a motion program that is automatically executed when power is applied, and is the mechanism by which stand-alone operation is implemented.

Macro #0 may also be used to set "power-up default" parameters, either before manual/computer control is begun or before transferring control to other macros for stand-alone operation.

Macro #0 is defined by the "MD0,xxx" command, where xxx represents a comma-separated list of base commands e.g.:

MD0,MR50000,WS100,GH,WS100,RP4CR

Macro #0 can be erased by:

RZCR

The contents of macro zero can be read by

TZCR

3.8.6 Stand-Alone Operation Examples

Mercury™ controllers offer the extremely useful feature of *autonomous macro execution*, meaning that positioning tasks can be programmed for execution at power-on, even if there is no PC connected. The macro language supports *conditional command execution*, which, in conjunction with the digital I/O lines and a small push button box (C-170.PB), provides virtually unlimited operational flexibility.

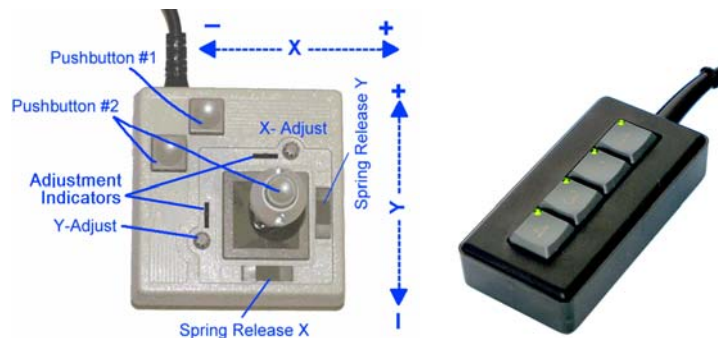


Fig. 2: C-819.20 Joystick and C-170.PB Pushbutton box with LEDs for Mercury™ Controllers

Mercury™ motor controllers offer convenient manual motion control by using pushbuttons and analog joysticks.

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.

With the joystick mode enabled, both *selected* and *deselected* Mercurys™ can be joystick-operated. The two joystick axes can be connected through the C-819.20Y cable to two Mercury™ controllers.

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.

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 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.

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.

The following examples illustrate some techniques that are useful in macro programming to make the Mercury™ a stand-alone controller under operator control.

In the descriptions below, parameters that are said to be “programmed” are set before operation from a host PC. Other variable behavior is determined at run time by an operator using the push button box and with no PC connected.

Example 1:

Autostart macro, initialization and “endless” loop functionality.

[MC00] EM5,EM13

[MC05] BF,DP200,DI,DD200,SV80000,SA400000,FE2,WS100,DH,SV250000

[MC13] EM14,EM15,EM16,EM17,WA20,RP,RP

- | | |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Macro #0 | Autostart-macro that is executed automatically at power up, independent of whether a PC is connected or not. Using it as a switch to call one or two other macros makes the program easier to understand and maintain. |
| Macro #5 | Initialization tailored for M-511.PD stages. It disengages the motor brake, sets the P-I-D parameters and starts a reference search in the correct direction towards the sensor (FE2). When the reference is found, the position is defined as zero and the velocity is set to 250,000 counts/second. |
| Macro #13 | Long-duration repetition macro, At most every 20 ms, the sequence of macros named in the EM commands (14, 15, 16, 17) is executed (their content is not shown here). The RP,RP makes the sequence repeat more than 4 billion times (in our scale, forever). |

Example 2:

These macros illustrate conditional command execution. The macro executes from left to right but the XN and XF commands cause the macro to exit immediately if the state of the specified input line (button) does not meet the required condition (high for XN, low for XF):

[MC15] XN1,XN2,SV30000,WF1,WF2

The above macro sets the speed to 30,000 if button #1 and #2 are pressed together. Macro exits when both are released.

[MC13] XN1,WA30,XF2,SV10000,WF1

The above macro sets speed to 10,000 if button #1 is pressed, unless button 2 is pressed within 30 ms (because that is probably an attempt to press both at once).

[MC14] XN2,WA30,XF2,SV20000,WF2

Sets the speed to 20,000 if button #2 is pressed unless button 1 is pressed within 30 ms

[MC16] XN3,SV70000,CP8,WA200,XN3,SV130000,CP12,WA200,XN3,SV190000,CP14,WA200,XN3,SV250000,CP15,WF3

Button 3 sets one of 5 speeds, depending on how long it is held in. LEDs indicate the progression. Every 200 ms the speed is reset and the number of LEDs lit is changed accordingly. Note that the most significant bit for CP corresponds to the bottom LED.

Example 3:

The following macros illustrate various types of moves that are useful under operator control:

[MC17] XN1,MR-9999999,WF1,AB1

Button 1 causes a continuous move in the negative direction. The target is out of range, so the move will continue until interrupted with the AB1 command. That command will only be executed after the button initiating the move is released.

[MC19] XN2,WS0,MR500

Button 2 causes repeated step moves of 500 counts until the button is released. Differs from a continuous move in that the end position will be a multiple of 500 from the start. Placing the WS before the move allows the rest of the loop to complete before the move has finished.

[MC18] XN3,MR5000,WF3

Button 3 causes a step move of 5000 counts in the positive direction. The WF3 command ensures that there will be only one step per press of the button.

[MC18] XN4,CF1,CF2,MA-50000,EM19,CN1,EM20,EMxx

[MC19] XF1,MR10,WS0,RP10000

[MC20] XF1,CN2

Input 4 initiates a move to -50000 followed by a step move of 10 x 10,000 counts that can be interrupted (with a resolution of 10 counts) by a signal on input 1. When the step move stops, digital output #1 is set high; if the move went to the end, then output #2 will also be set. Note that it is no longer possible to return to the macro that called Macro 18 because of the nesting limit. The solution is to chain to the calling macro explicitly with EM.

A multi-dimensional scan can be arranged by properly interconnecting the IO lines of the Mercury™ controllers controlling the different axes.

Example 4:

This large macro set is designed to allow fully autonomous operation in any of four modes. The desired mode is selected by pressing the corresponding button upon start-up; thereafter the buttons function as defined for the mode chosen, as can be seen from the macro descriptions.

- | | |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mode 1: | The motor moves at one of two programmed speeds as long as one of the buttons is pressed. The types of motion associated with the four buttons are “positive-fast,” “positive-slow,” “negative-fast,” and “negative-slow.” |
| Mode 2: | The motor moves to pre-defined positions at a programmed speed and acceleration. |
| Mode:3: | The motor moves by long or short increments at a programmed speed |
| Mode 4: | Same as mode 2, but with a different speed setting. |

// Mercury™ Macro File

[MC00] EM+9,EM+10
 [MC09] BF,DP200,DI,DD200,SV80000,SA400000,FE2,WS100,DH,SV250000
 [MC10] EM11,EM12,EM13,EM14
 [MC11] XN1,SV70000,EM15
 [MC12] XN2,SV250000,EM20
 [MC13] XN3,EM26
 [MC14] XN4,SV120000,EM20
 [MC15] EM16,EM17,EM18,EM19,WA20,RP,RP
 [MC16] XN1,MR9999999,WF1,AB1
 [MC17] XN2,MR10000,WF2
 [MC18] XN3,MR-10000,WF3
 [MC19] XN4,MR-9999999,WF4,AB1
 [MC20] EM25,EM21,EM22,EM23,EM24,WA100,RP,RP
 [MC21] XN1,MA100000,CP1,WF1
 [MC22] XN2,MA50000,CP2,WF2
 [MC23] XN3,MA-50000,CP4,WF3
 [MC24] XN4,MA-100000,CP8,WF4
 [MC25] XN2,XN3,GH,CP6,WF2,WF3
 [MC26] EM27,EM28,EM29,EM30,WA30,RP,RP
 [MC27] XN1,MR4000,WS50,WA20
 [MC28] XN2,MR2000,WS50,WA20
 [MC29] XN3,MR-2000,WS50,WA20
 [MC30] XN4,MR-4000,WS50,WA20

These macros can be stored permanently in the Mercury™ Controller. The data file itself is an ASCII file and can be stored using the file manager of the *Mercury™ Move Software*.

Macro #0 Autostart-macro that is executed automatically at power up, independent of whether a PC is connected or not. It calls two other macro commands, macro #9 and macro #10.

Overall initialization

- Macro #9 Tailored for M-511.PD stages. It disengages the motor brake, sets the P-I-D parameters and starts a reference search in the correct direction towards the sensor (FE2). When the reference is found, the position is defined as zero and the velocity is set to 250,000 counts/s.
- Macro #10 Calls the sequence of macros #11, #12, #13 and #14. This is used to find a button pressed at power up, or just after the reference position is found. If a button is just pressed, another macro is called. If not, Mercury™ goes to normal operation and waits for command input.

Macros that set up and jump to the sections corresponding to each mode

- Macro #11 Only executed if button #1 is pressed. If the button is pressed, the velocity is set to 70,000 and macro #15 is executed.
- Macro #12 Only executed if button #2 is pressed. If the button is pressed, the velocity is set to 250,000 and macro #20 is executed.
- Macro #13 Only executed if button #3 is pressed. If the button is pressed, macro #26 is executed.
- Macro #14 Only executed if button #4 is pressed. If the button is pressed, the velocity is set to 120,000 and macro #20 is executed.

Macros used for Mode 1

Macro #15	Long-duration repetition macro, Every 20 ms the sequence of macros #16, #17, #18 and #19 is executed. The RP,RP makes the sequence repeat more than 4 billion times (in our scale forever).
Macro #16	Only executed if button #1 is pressed. As long as the button is pressed, the motor is moved in positive direction.
Macro #17	Only executed if button #2 is pressed. The motor moves 10,000 steps (positive). When the button is released, the macro is exit.
Macro #18	Only executed if button #3 is pressed. The motor moves 10,000 steps (negative). When the button is released, the macro is exit.
Macro #19	Only executed if button #4 is pressed. As long as the button is pressed, the motor is moved in negative direction.

Macros used for modes 2 and 4

Macro #20	Long-duration repetition macro, Every 100 ms the sequence of macros #25, #21, #22, #23 and #24 is executed. The RP,RP makes the sequence repeat more than 4 billion times (in our scale forever).
Macro #21	If button #1 is pressed, the motor moves to position 100,000. The digital output pattern is set to 1 (LED #1 ON). The macro exits when the button is released.
Macro #22	If button #2 is pressed, the motor moves to position 50,000. The digital output pattern is set to 2 (LED #2 ON). The macro exits when the button is released.
Macro #23	If button #3 is pressed, the motor moves to position -50,000. The digital output pattern is set to 4 (LED #3 ON). The macro exits when the button is released.
Macro #24	If button #4 is pressed, the motor moves to position -100,000. The digital output pattern is set to 8 (LED #4 ON). The macro exits when the button is released.
Macro #25	If button #2 and button #3 are pressed at the same time, the motor goes home. The digital output pattern is set to 6 (LED #2 and #3 ON). The macro exits when both buttons are released.

Macros used for Mode 3

Macro #26	Long-duration repetition macro, Every 30 ms the sequence of macros #27, #28, #29 and #30 is executed. The RP,RP makes the sequence repeat more than 4 billion times (in our scale forever).
Macro #27	If button #1 is pressed, the motor moves 4,000 steps, waits until the position is reached, then waits for another 20 ms and repeats the move as long as the button remains depressed.
Macro #28	If button #2 is pressed, the motor moves 2,000 steps, waits until the position is reached, then waits for another 20 ms and repeats the move as long as the button remains depressed.
Macro #29	If button #3 is pressed, the motor moves -2,000 steps, waits until the position is reached, then waits for another 20 ms and repeats the move as long as the button remains depressed.
Macro #30	If button #4 is pressed, the motor moves -4000 steps, waits until the position is reached, then waits for another 20 ms and repeats the move as long as the button remains depressed.

4 System Details

4.1 Control Options—DC-Motor Controller Only

The C-863 Mercury™ DC-motor controller features closed-loop control of the connected mechanics (= operation with servo ON). Closed-loop control can be enabled with the MN command (p. 69) and disabled by the MF command (p. 69).

The following control options can be selected using the FM command (p. 55):

- **Position control**, power-on default, see p. 29
- **Force control** (with firmware revision 2.21 and newer), see p. 29
- **Position and force control** (with firmware revision 2.21 and newer), see p. 30

4.1.1 Position Control

Position control is the power-on default. The control variable is the position of the stage. The sensor feedback used by the control-loop is that of the incremental position encoder in the stage.

The position-control-loop features dedicated parameters (P-, I-, D-term, I-limit) which can be set by command.

See Section 3.4 on p. 10 for factory default settings and for how to set the parameters.



Fig. 3: Position control algorithm, with sensor feedback (position encoder in the stage) and PID correction

4.1.2 Force Control

With firmware revision 2.21 and newer, it is possible to control the force or another alternative variable measured by an additional sensor connected to the “Joystick” socket, e.g. a force sensor.

This option can be used, for example, to maintain the force the stage imposes upon an object. If the force increases, the stage moves back, and if the force decreases, the stage moves forward until the current force matches the target force again.

Sensor signal range: -10 V to +10 V

Sensor signal input: "Joystick" socket, pin 2

Resolution: 12 bit

The force control-loop features dedicated parameters (P-, I-, D-terms, I-limit) which can be set by command.

The low pass filter for the sensor input can be set via the FT command. The frequency range is 40 Hz to 10,000 Hz.

See Section 3.4 on p. 10 for factory default settings and for how to set the parameters.

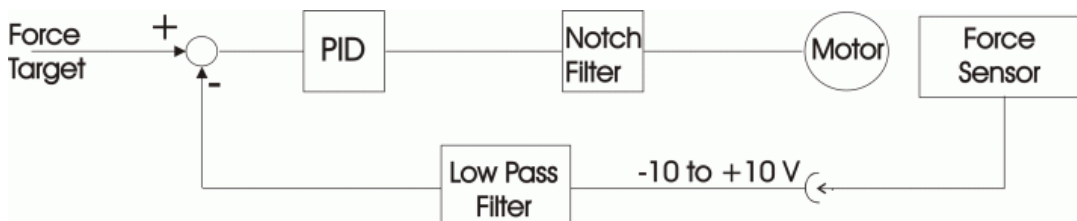


Fig. 4: Force control algorithm with PID correction and notch filter. The feedback of the additional (force) sensor is corrected by a low pass filter.

4.1.3 Position and Force Control

With firmware revision 2.21 and newer, position control with an embedded (force) control-loop is possible, i.e. the position and an additional variable are controlled. The sensor feedback used for position-control is that of an incremental position encoder in the mechanics, while the sensor feedback used for the embedded control-loop is that of an additional sensor connected to the “Joystick” socket, e.g. a force sensor. See Sections 4.1.1 and 4.1.2 for further details.

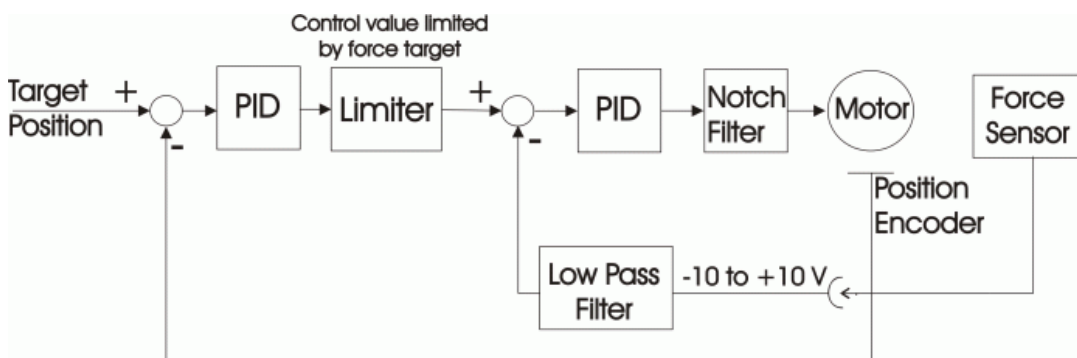


Fig. 5: Control algorithm maintaining position and force, with feedback of position encoder and additional (force) sensor; two control loops with separate PID corrections; the force target is used as limit for the control value resulting from the position control loop

4.1.4 Notch Filter

Mechanical resonances of the system exaggerate the response to certain frequencies in the control signal. The C-863 has a notch filter on the output of the control-loop to compensate for resonances in the mechanics by reducing the corresponding frequency components in the control signal. The frequency range for the notch filter is 40 Hz to 10,000 Hz. Notch filter frequency and edge can be

set by the FN and FQ commands. Default values are 10,000 Hz and a notch edge of 0.8.

4.1.5 Commands for Closed-Loop Control—Overview

Command	Function	Corresponding Report Command
DDn	Set servo d-term for position control to n	GD
DIn	Set servo i-term for position control to n	GI
DLn	Set servo i-term limit for position control to n	GL
DPn	Set servo p-term for position control to n	GP
FMn	Set Control Mode, 0 = position control (default), 1 = force control, 2= position and force control	FM?
FAn	Set force target range: $-1000 < n < +1000$	FA?
FPn	Set servo p-term for force control to n	FP?
FIn	Set servo i-term for force control to n	FI?
FLn	Set i-term limit for force control to n	FL?
FDn	Set servo d-term for force control to n	FD?
FOn	Set force offset to n	FO?
FTn	Set low pass filter	FT?
FNn	Set notch filter frequency	FN?
FQn	Set notch filter Q-value	FQ?
MAn	Move absolute to target position n	
MF	Motor servo off (disable closed-loop control)	
MN	Motor servo on (enable closed-loop control)	
MRn	Move relative by distance n, target position = n + last valid target	
TP	Tell position (of encoder)	
TR	Tell input signal of additional (force) sensor	
TT	Tell target position	

4.2 On-Target Detection

By default, the on-target flag is set when the motion trajectory has finished. It is assumed that the motor then has reached the target position. But in fact, the motor can still be in the physical settling process so that the actual motor position may not yet match the target with count accuracy.

With C-863 DC motor controllers, you can combine the on-target flag with the physical motor position. To do this, use WW (set settle window) and WT (set settle

time) to define the appropriate conditions. With WW, a settle window is defined which is centered around the target position. The on-target status becomes "true" when the actual position stays in this window for at least the settle time (set with WT). By default, the settle time is set to 0 (WT0) which means that the axis is on target when the calculated motion trajectory has finished, irrespective of the settle window settings.

To report the on target state, use the TS command.

4.3 Position Referencing

Incremental encoder signals can detect minute position changes, but are inherently unable to indicate absolute position. Hence the position counter in the software and firmware is set to 0 upon power-up or reboot, and a reference signal detection mechanism must be used to determine the absolute physical position of the stage. A position reference switch (sometimes called "origin sensor") working in conjunction with a capture mechanism of the motion processor can provide this information.

The FE, FE1, FE2 and FE3 (find rising/falling edge) commands are used to start a reference signal search run:

Command	Start the reference position search in:
FE	positive direction
FE1	negative direction
FE2	Automatic reference search for PI stages
FE3	Automatic reference search for older PI stages

4.4 Limit Signals

4.4.1 Limit Switches

During operation, limit sensors (switches) can be used to stop motion at the end of the allowable travel range. 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 compatible 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).

On the C-663 and C-863, the limit switch signals are interpreted by both the controller hardware and software. DIP switch 7 changes the hardware interlock between active-high (OFF) and active-low (ON).

For the software configuration the following commands are used with all Mercurys™:

LH	Limit switch logic active high	p. 67
LL	Limit switch logic active low	p. 67

In addition, it is possible to disable software limit switch evaluation completely:

LN	Limit switch operation ON	p. 67
LF	Limit switch operation OFF	p. 67

NOTE

If the hardware and software limit switch configurations are not compatible, no motion is possible. Standard PI stepper motor stages have active-low limit switches, whereas PI DC motor stages have active-high limit switches.

4.4.2 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. Damage could, for example, occur if the stage would run into the hard stop at full speed since the limit switches in the stage stop the stage without deceleration.

When a position beyond a soft limit is commanded with MA or MR, the stage moves and is stopped smoothly at the corresponding soft limit. Note that moves started with FE are not affected by the soft limits.

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.

The deceleration applied is that programmed with the SA command.

Command	Function	Corresponding Report Command
JHn	Set upper soft limit Motion in positive direction does not exceed that position.	JH?
JLn	Set lower soft limit. Motion in negative direction does not exceed that position.	JL?

4.5 Input/Output Lines

4.5.1 Overview

Mercury™ controllers offer 4 digital outputs and 4 inputs for either digital or (8-bit) analog use. A set of commands is available to handle these I/O lines:

Command	Function
CN	Channel ON
CF	Channel OFF
CP	Channel pattern defining digital outputs
TC	Tell channel (digital input)
TA	Tell analog input value
WN	Wait channel ON
WF	Wait channel OFF
XN	Execute if channel ON
XF	Execute if channel OFF

Note that there are commands which allow conditional exits from macros depending on the digital and joystick inputs. This makes possible pushbutton control and/or interaxis communication in stand-alone mode, i.e. without a PC connection.

4.5.2 Trigger Output

You can program the digital output line 4 of a Mercury™ controller to trigger other devices (pin 8 of the "I/O" socket, see User manual for pinout).

The trigger mode selection and the settings for trigger position(s) can be made by command.

Command	Function	Corresponding Report Command
ZMn	Set trigger mode n n=0 Trigger OFF n=1 Single trigger pulse: A trigger pulse is written when the axis has reached the trigger position given by ZPn. n=2 Multiple trigger pulses: The first trigger pulse is written when the axis has reached the trigger position given by ZPn. The next trigger pulses each are written when the axis position equals the sum of the last valid trigger position and the increment value given	ZM? Report e.g. : "ZM:0"

	by ZIn. n=3 Trigger "grid" mode: A trigger pulse is written whenever the axis has covered the distance given by ZIn (the value is used for both motion in positive and negative direction)	
ZPn	Set trigger position n n can be positive or negative.	ZP? Report e.g.: "ZP:10000"
ZIn	Set trigger position increment n n can be positive or negative	ZI? Report e.g.: "ZI:100"

Command sequence example:

DH20000 Set current position to 20,000
ZP26000 Set trigger position to 26,000
ZI100 Set trigger increment to 100
SV50000 Set velocity to 50,000 counts/s
SA400000 Set acceleration to 400,000 counts/s²
ZM2 Set trigger mode to "multiple"
MR100000 Start a relative move by 100,000 counts in positive direction

This sequence provokes the following:

The stage starts at position 20,000 and moves to 120,000 with defined velocity and acceleration. When crossing position 26,000 a trigger signal is sent and the next trigger position is set to 26,100, where the next trigger is sent and so on.

The trigger increment can also be negative. In that case the stage should run in negative direction, otherwise the decremented trigger position is never reached.

5 Native Command Types

Over 50 commands are available for programming the Mercury™ controller. Commands are used to set parameters, to control motion and to read values from the controller. They can be classified as follows:

- **Address Selection Code:** 2 ASCII characters beginning with 01 (ctrl-A); instructs the specified controller to enter the *selected* state and all non-specified controllers to enter the *deselected* state. This is the only command that can be sent over the interface and affect controllers in the (power-on default) *deselected* state.

All the following command types are ignored by controllers in the *deselected* state.

- **Base Commands:** (e.g. "MR5000")
Single commands, consisting of a mnemonic optionally followed by an argument. Execution begins immediately after reception of the CR character marking the end of the command.
- **Compound Commands** (e.g. "MR5000,WS100,GH")
Series of base commands separated by commas. Sequential execution starts upon receiving the CR, which marks the end of the compound command.
- **Single-Character Commands** (e.g. 0x27)
One character without terminator.
Immediate report or action
- **Conditional Commands** (e.g. "XN1,MR5000")
For use in compound commands only; affect the interpretation of the base commands following them in the compound command.

5.1 Address Selection Codes

The address selection code consists of two characters. The first character is ASCII character #1, and the second character corresponds to the settings of the address DIP switches (1 to 4) on the front panel. Both characters are sent without additional termination character

This sequence instructs the specified controller to enter the *selected* state and all non-specified controllers to enter the *deselected* state. This may occur during execution of another command or macro. It is the only command which can affect controllers in the *deselected* state.

In the *deselected* state, all commands except address selection codes are ignored, and all reports (e.g. from running macros) suppressed.

Because Mercurys™ power-up in the *deselected* state, an address selection code is typically the first sequence sent. Note that using the SC command, a controller can be selected from within its autostart macro. See "Addressing" on p. 7 for further details.

5.2 Base Commands

A *base command* is one command followed by a terminating `CR` (ASCII character 10). Note that PI GCS software uses `LF` as command terminator. PI software, including PITerminal, automatically switches termination character depending on the controller attached.

The command is executed immediately after the `CR` is received. The command will be repeated once for each `CR` that is received immediately after it is entered. This feature makes it very easy to continuously execute the same command by simply holding down the `Enter` key. Both uppercase and lowercase characters are valid, and spaces are allowed.

Examples of base commands:

MR2000	Move motor 2,000 counts/steps relative to the present target
DP250	Set p-term of servo filter for position control to 250 (C-863 DC motor versions only)
MN	Set motor in ON state
GH	Send motor to home position (go home)
MA20000	Send motor to absolute position 20,000
TP	Reports position as: "P:+0000000000"

5.3 Compound Commands

A *compound command* is a series of base commands separated by commas. It is thus possible to string together several commands before terminating them with a `CR` (Carriage Return). These base commands will then be executed sequentially.

The syntax for a compound command is:

`CMD[n], CMD[n], ..., CR`

Examples:

```
MR2000,WS100,MR-4300
mr5000,ws100,wa500,ma12000,ws100,wa800,tp
MA3500, WS100,WA120,tp
```


The compound command in the following example instructs the motor to move 1000 steps/counts in the positive direction, wait in that position 500 milliseconds, return to the original position, wait 1 second, and then repeat the sequence 5 times.

Example:

```
MR1000,WS100,WA500,MR-1000,WS100,WA1000,RP5
```

Once this compound command is entered, it remains in the buffer until replaced by another non-blank line of input and can be re-executed by sending a carriage return `CR` alone (blank line).

5.4 Single-Character Commands

Single-character commands are used to generate a report or carry out an action by sending only one character without a following `CR`. This allows faster requests (saving the second character and the `CR`), but the main advantage of single-character commands is that they are executed immediately even when the controller is in the process of executing a macro or compound command, without interrupting a running sequence (except for the “motor stop” command “!”).

Find a complete overview of single-character commands in the Command Reference section on p. 46.

5.5 Conditional Commands

Conditional commands are normally used in compound or macro commands. The state of an input line decides whether the following command(s) are executed or not.

Example: `XN4,MR5000`

The relative move is only executed if digital input #4 is ON (high).

5.6 Commands with Responses

Some commands cause the controller to send a string of data to the host computer, be it a position, servo-control parameters, or other information. These commands, also called *report commands*, usually begin with a T (tell) or G (get).

A compound command or macro containing a report command will also issue the corresponding report when the report command is executed. The reports are suppressed, however, if controller is put in the *deselected* state.

5.7 Command Execution Mode

Base or compound commands can be executed in one of two modes:

- Direct mode, where the command comes in to the *selected* controller over the active interface,
- Macro execution mode, where commands in macros stored on the controller are being executed.

The execution mode does not affect the way the controller responds to single-character commands or *address selection codes* sent over the interface. Nor does the selection state affect operations in macro execution mode, except that reports are suppressed when the controller is *deselected*.

5.7.1 Direct Mode

If there is no Macro #0 defined, the controller goes into direct mode upon power-on or reset .

There are only two ways to place it in macro execution mode:

- Call a macro (executing an EM n command where n is a defined macro)
- Make sure Macro #0 is defined and restart (e.g. with RT).

5.7.2 Macro Execution Mode

When in macro execution mode, the controller returns to direct mode under the following circumstances:

- Controller is in *selected* state and receives any character other than single-character command or the two characters of an address selection code
- Controller is power-cycled or reset with no Macro #0 defined
- Macro which called another macro has finished
- Macro not called by another macro has finished

To achieve a full understanding of how this works, see the EM command description.

6 Native Command Reference

This section describes commands supported by firmware 1.06 and higher.

All single commands fall into one of two classes, those which evoke responses (*report* commands) and those which do not. *Report* commands report the requested values in a defined format.

6.1 Command and Response Formats

6.1.1 Command Execution

Upon power-on, all Mercury™ Class controllers execute the startup macro (if present) and, unless selected from inside the macro, remain in the *deselected* state. That means the controller ignores everything on the command interface except for *address selection codes*.

Whenever an *address selection code* is sent, the controller with the matching address goes into the *selected* state and all other controllers into the *deselected* state. The controller stays in the *selected* state as long as no other, non-matching *address selection code* is received.

Upon reception of any character except an *address selection code* or a single-character command, the selected controller will interrupt any running macro or compound command and start to interpret the input as a command.

Upon reception of a single-character command, the selected controller will execute the command immediately, even during execution of a macro or compound command.

6.1.2 Command Codes

Mercury™ commands use a more or less mnemonic code of 1 to 3 characters to identify the type of operation; the code is followed by pertinent data value(s), if necessary.

NOTE

All commands are terminated by the termination character CR (carriage return, ASCII character decimal 13). Exceptions are single-character commands and address selection codes which are executed immediately without termination.

Examples:

MR5000 CR	Move Relative 5000 counts or (micro)steps
TP CR	Tell Position, causes the report to be sent

For example, “TP” (Tell Position) by itself is adequate to display the motor position, but “MR” (Move Relative) alone would be useless, because the system would not know how far you wished to move. Some commands which take a data value will assume a particular value if none is specified.

MR, for example, must be followed by a minimum of 1 and a maximum of 9 digits to accommodate the allowable range of motion.

6.1.3 Report Commands

Report commands, which cause the Mercury™ controller to emit a string of data, be it a position, target or other information usually begin with a “T” for “Tell” or “G” for “GET”.

Single-character commands generate the same report strings as the equivalent base command would:

Example:

"%" equals TSCR

Report strings generated by a report command are terminated by the three-character sequence CR LF ETX (ASCII 13, 10 , 03)

Examples of report strings :

Command: TT	Report:	T:+0000035500	CR	LF	ETX
Command: TL	Report:	L:+0000000120	CR	LF	ETX

NOTE

The command and report terminators (CR LF ETX) will be omitted in most of the examples in this manual

6.2 Base Commands in Alphabetic Order

Commands with white background apply to all controller versions, lines with light gray background apply to C-663 stepper motor versions only, and lines with dark gray background apply to C-863 DC motor versions only.

Com-mand	Function	Valid For (if empty, command is valid for all controllers)	Corresponding Report Command	Report Identifier	Page
AB	Abort motion: Stop abruptly				47
AB1	Abort motion: Stop smoothly				47
BF	Set brake OFF (and/or deactivate some motors)				48
BN	Set brake ON				48
CF	Channel OFF				48
CNn	Channel ON				48
CP	Channel pattern defining digital outputs				49
CS	Report checksum			C:	49
DCn	Define drive current	C-663			49
DD	Define derivative gain for position control	C-863	GD		50
DH	Define home				50
DI	Define integral gain for position control	C-863	GI		50
DL	Define integral limit for position control	C-863	GL		51
DP	Define proportional gain for position control	C-863	GP		51
EM	Execute macro				52
FA	Set Absolute Target for Force Control	C-863	FA?		52
FA?	Get Absolute Target for Force Control	C-863		FA:	52
FD	Set D-Term for Force Control	C-863	FD?		53
FD?	Get D-Term for Force Control	C-863		FD:	53
FE	Find edge (find reference position)				53
FI	Set I-Term for Force Control	C-863	FI?		54
FI?	Get I-Term for Force Control	C-863		FI:	55
FL	Define integral limit for force control	C-863	FL?		55
FL?	Get integral limit for force control	C-863		FL:	55
FM	Set Control Mode	C-863	FM?		55
FM?	Get Control Mode	C-863		FM:	56
FN	Set Notch Filter Frequency	C-863	FN?		56

Com- mand	Function	Valid For (if empty, command is valid for all controllers)	Corresponding Report Command	Report Identifier	Page
FN?	Get Notch Filter Frequency	C-863		FN:	56
FO	Set Offset to Sensor Input of Additional Sensor	C-863	FO?		57
FO?	Get Offset of Additional Sensor Input	C-863		FO:	57
FP	Set P-Term for Force Control	C-863	FP?		57
FP?	Get P-Term for Force Control	C-863		FP:	58
FQ	Set Notch Filter Edge	C-863	FQ?		58
FQ?	Get Notch Filter Edge	C-863		FQ:	58
FS	Set Force Sign	C-863			59
FS?	Get Force Sign	C-863			59
FT	Set Low Pass Filter Frequency	C-863	FT?		59
FT?	Get Low Pass Filter Frequency	C-863		FT:	60
GD	Get D-Term for Position Control	C-863		D:	60
GH	Go home				60
GI	Get I-Term for Position Control	C-863		I:	60
GL	Get I-Limit for Position Control	C-863		M:	61
GP	Get P-Term for Position Control	C-863		G:	61
HC	Set hold current	C-663			61
HT	Set hold time	C-663			62
JA	Enable Trackball Mode		JA?		62
JA?	Get Trackball Activation State			JA:	63
JB	Set Joystick Bias		JB?		63
JB?	Get Joystick Bias			JB:	63
JC	Clear Soft Limits				63
JF	Set joystick OFF				65
JH	Set Upper Soft Limit		JH?		64
JH?	Get Upper Soft Limit			JH:	64
JL	Set Lower Soft Limit		JL?		64
JL?	Get Lower Soft Limit			JL:	65
JN	Set joystick ON				65
JT	Select joystick table				66
LF	Limit switch operation OFF				67
LH	Limit switch logic active high				67
LL	Limit switch logic active low				67
LN	Limit switch operation ON				67

Com- mand	Function	Valid For (if empty, command is valid for all controllers)	Corresponding Report Command	Report Identifier	Page
MA	Move absolute				68
MD	Macro definition				68
MF	Motor servo off				69
MN	Motor servo on				69
MR	Move relative				69
RM	Remove (erase) specified macro				70
RMA LL	Remove all macros and parameters				70
RP	Repeat from beginning of line				70
RT	Reset system (like power-on reset)				71
RZ	Remove (erase) Macro 0 (autostart macro)				71
SA	Set acceleration				71
SB	Set Filter for Trackball (Digital Input)	C-863	SB?		72
SB?	Get Filter Settings for Trackball (Digital Input)	C-863		SB:	72
SC	Select controller				72
SF	Set Filter Value for Encoder Input	C-863	SF?		73
SF?	Get Filter Value for Encoder Input	C-863		SF:	73
SI	Select Index for Joystick table				73
SJ	Set joystick table value				73
SM	Set Maximum Following Error	C-863	SM?		74
SM?	Get Setting for Maximum Following Error	C-863		SM:	74
ST	Stop motion smoothly and move back				75
SV	Set velocity				75
TA	Tell analog input value			An:	75
TB	Tell board address			B:	76
TC	Tell channel (digital input)			H0n:	76
TD	Tell Dynamic target	C-863		N:	77
TE	Tell error	C-863		E:	77
TF	Tell following error	C-863		F:	77
TI	Tell iteration number			X:	78
TL	Tell programmed acceleration			L:	78
TM	Tell macro contents				78
TP	Tell position			P:	79

Com- mand	Function	Valid For (if empty, command is valid for all controllers)	Corresponding Report Command	Report Identifier	Page
TR	Tell Current Value of Alternative Sensor	C-863		R :	79
TS	Tell status			S :	80
TT	Tell target position			T :	83
TV	Tell current velocity			V :	83
TY	Tell programmed velocity			Y :	83
TZ	Tell macro zero				84
VE	Display version number				84
WA	Wait absolute time				84
WF	Wait channel OFF				85
WN	Wait channel ON				85
WS	Wait stop				85
WT	Set settle time	C-863	WT?		86
WT?	Get settle time	C-863		WT :	86
WW	Set settle window	C-863	WW?		86
WW?	Get settle window	C-863		WW :	87
XF	Execute if channel OFF				87
XN	Execute if channel ON				87
ZI	Set trigger increment		ZI?		88
ZI?	Get trigger increment			ZI :	88
ZM	Set trigger mode		ZM?		89
ZM?	Get trigger mode			ZM :	89
ZP	Set trigger position		ZP?		89
ZP?	Get trigger position			ZP :	90

6.3 Single-Character Commands

Single-character commands are used to generate a report or initiate an action by sending only one character without terminator. This allows faster requests (saving the second character and the `CR`), but the main advantage of single-character commands is that they are interpreted immediately and executed asynchronously. They can thus be used during execution of macros or compound commands without interrupting a running sequence.

Char	Equivalent Command	ASCII decimal	ASCII hex	Report generated	Comment
"+" *	TE*	43	0x2B	E:xxx	Tell Position Error
"(" *	TF*	40	0x28	F:xxx	Tell Following Error
"%"	TS	37	0x25	S:xxx	Tell Status
"#"	TC	35	0x23	H00:xxx	Tell Channel
"' "	TP	39	0x27	P:xxx	Tell Position
"!"	AB	33	0x21	(none)	Motor stop
"\"	(none)	92	0x5C	0 or 1	moving status
"/"	TA2	47	0x2F	A2:xxxx	Tell Analog 2
"&"	TA1	38	0x26	A1:xxxx	Tell Analog 1
")"	TA4	41	0x29	A4:xxxx	Tell Analog 4

* commands not available for C-663 Mercury™ Step

NOTE

Single-character commands are sent without any terminator. As soon as such a character is recognized, the command is executed.

6.4 Base Command Reference in Alphabetic Order

NOTE

All commands have to be sent with the termination character CR (carriage return)

All report strings are terminated by the sequence: CR LF ETX (Carriage Return + Line Feed + EndOfText)

AB Abort motion (abruptly)

Stops the motor immediately.

The motion profile velocity is set immediately to zero and the previously programmed target position is set to the current position.

AB is used for immediate stopping. If running at high speeds you may take into account that stopping the motor abruptly may cause the motor to lose track of the position. This command will cause a mechanical shock to the system because no deceleration ramp is used. Use the AB1 or ST commands if you want to stop the motor smoothly.

Example:

AB	Aborts the motion of the motor immediately
-----------	--------------------------------------------

See also: AB1, "!" (single-character command) and ST

AB1 Abort motion (smoothly)

Stops the motor smoothly with programmed deceleration ramp.

The motion profile velocity is reduced to zero using the current set acceleration value, and the target position is set to the current position.

AB1 is used for smooth stopping while the motor moves at high speed.

Example:

AB1	Aborts the motion of the motor smoothly
------------	-----------------------------------------

See also: AB, "!" (single-character command) and ST

BF Brake OFF

Deactivates the active-low motor brake output line, (motor connector pin 1, set to 5 V) releasing the motor brake, if present and connected.

With many .DD and .PD stages, the BF state (5 V) is required to enable the motor amplifier, whether the stage is equipped with a brake or not.

See also: BN

BN Brake ON

Activates the active-low motor brake output line (motor connector pin 1, set to 0 V) causing a motor brake, if present and connected, to engage. The factory default power-on state is 0 V (brake on). T

With active drive stages (.DD and .PD models), the BN state (0 V) will disable the motor amplifier even if no brake is installed.

See also: BF

CF n Channel OFF (range n : 1..4)

Sets digital I/O output channel n to OFF (low, 0V). No other channels are affected.

Command: CF n

Parameter: n indicates the digital I/O output channel, can be 1, 2, 3 or 4.

Report: none

see also CN, CP

Example:

CF3	sets digital I/O output channel #3 to zero
------------	--------------------------------------------

CN n Channel ON (range n : 1...4)

Sets digital I/O output channel n to ON (high, +5V). No other channels are affected.

Command: **CN n**

Parameter: n indicates the digital I/O output channel, can be 1,2,3 or 4.

Report: none

See also: CF, CP

CP n Channel Pattern (range n : 0 to 15)

Sets digital output channels 1 to 4 according to the digits in the binary representation of value of n (bitmapped). $n=0$ sets all channels to low (0V), $n=15$ sets all channels to high (+5V). $n=3$ sets channels 1 and 2 high, channels 3 and 4 low, and so on.

Command: **CP n**

Parameter: n is a channel mask for digital I/O output channels 1 to 4, if a bit is set, the corresponding channel is ON.

Report: none

see also: CN,CF

Examples:

CP5	Sets channels 1 and 3 high, 2 and 4 low (5 = 0101 _b)
CP15	Sets all channels high
CP0	Sets all channels low

CS CheckSum

Reports the current firmware version.

Command	Report
CS	C:0230-15

DC n Define Drive Current

C-663 Mercury Step™ only: sets the motor phase current (drive current) for moving state.

Parameter n : Current in mA

Example:

DC400	Sets the drive current to 400 mA.
--------------	-----------------------------------

See also: HC, HT

DDn Define Derivative gain for Position Control (0 < n < 32,767)

C-863 DC-motor versions only: Sets the gain to be applied to the derivative term in the PID algorithm for the position control loop. The primary purpose of this term is to increase damping and reduce overshoot at the end of motion.

Range: 0 to 65,535

The factory default value is 0

Corresponding report command: GD

Examples:

DD200	Set the D-term to 200
--------------	-----------------------

GD	Report:
	D: +0000000200

See also: DI, DP, DL

DHn Define Home

Defines the current motor position as *n* (home position will be *n* counts/steps away).

Example:

DH	Defines the current position as 0.
DH20000	Defines the current position as 20000

DIn Define Integral gain for Position Control

C-863 DC-motor versions only: Sets the gain to be applied to the integral term in the PID algorithm for the position control loop. The primary function of this term is to overcome friction-induced static errors.

Range: 0 to 65,535

The factory default value is 0

Corresponding report command: GI

Examples:

DI45	Set the I-term to 45
GI	Report: I: +0000000045

See also: DL, DD, DP

DLn Define Integral Limit for Position Control

C-863 DC-motor versions only: Limits the amount of contribution by the integral gain term for the position control loop.

Range: 0 to 65,535

The factory default value is 2,000

Corresponding report command: GL

Examples:

DL2000	Set the I-limit to 2,000
GL	Report: M: +0000002000

See also: DI, DD, DP

DPn Define Proportional gain for Position Control

C-863 DC-motor versions only: Sets the slope of the proportional relationship between the position error and the motor voltage for the position control loop. The higher the gain value set, the greater the stiffness of the position coupling, meaning that a small error value causes a proportionally larger motor current driving the motor towards the target.

The default gain value usually ensures stable operation. The optimum value depends on friction, inertia, motor power, and the resolution of the encoder. It must be determined by the user.

If the error reported by an axis after completing its motion is excessive, the gain value may be increased in small increments until the error is within acceptable limits. If the axis becomes unstable and begins to oscillate, the gain must be reduced until the oscillation stops.

Range: 0 to 65,535

The factory default value is 35.

Corresponding report command: GP

Examples:

DP220	Set the P-term to 220
--------------	-----------------------

GP

Report:

G: +0000000220

See also: DD, DI, DL

EM n Execute Macro Command n (range n : 1 to 31)

Executes the macro command stored under the indicated macro number n . The macro command may contain another EM command so that they can work like a program loop. The macro execution stops when the macro sequence has finished or when any new command is received.

If there is no macro defined for the number n , no action will be taken.

Example:

EM3	Execute macro #3
------------	------------------

See also: MD, TM, TZ

FA n Set Absolute Target for Force Control

C-863 DC-motor versions only: If the C-863 is used in “force control” or “position and force control” mode, this command defines the force target that shall be reached and maintained by varying the position.

Range: 0 to 1024

Corresponding report command: FA?

Example:

FA225	Set force target to 225
--------------	-------------------------

See also: FD, FI, FM, FO, FP, FN, FQ, FT, TR

FA? Get Absolute Target for Force Control

C-863 DC-motor versions only: Reports the force target set with FA.

Example:

FA?	Report:
	FA: +225 (Force target is set to 225)

See also: FD, FI, FM, FO, FP, FN, FQ, FT, TR

FD n Set D-Term for Force Control

C-863 DC-motor versions only: If the C-863 is used in “force control” or “position and force control” mode, this command defines the D-term for the force control algorithm.

Range: 0 to 65,535

The factory default value is 0

Corresponding report command: FD?

Example:

FD150	Set force D-term to 150
--------------	-------------------------

See also: FA, FI, FM, FO, FP, FN, FQ, FT, TR

FD? Get D-Term for Force Control

C-863 DC-motor versions only: Reports the D-term set with FD for the force control algorithm.

Example:

FD?	Report:
	FD: +150 (Force D-term is set to 150)

See also: FA, FI, FM, FO, FP, FN, FQ, FT, TR

FE n Find Edge (range n : 0 to 3)

Searches for the reference (origin) position.

The motor moves until a transition on the reference line is detected. In conjunction with a physical reference switch, FE can be used to move to a known (origin) position. The physical origin point is where the reference input line detects a transition from GND to +5 V or vice versa. Most of PIs mechanical stages have a reference sensor that can be used with this feature.

The FE functionality considers directional hysteresis effects and permits to find the reference point precisely irrespective of the starting position.

Moves started with FE are not affected by soft limits set with JH and JL.

The meaning of the parameter n is:

n=0:	Search starts in positive direction
n=1:	Search starts in negative direction
n=2:	Search starts in positive direction if the reference line input level is high. Otherwise the search starts in negative direction. <i>Use this option when referencing Pi standard stages.</i>
n=3:	Search starts in positive direction if the reference line input level is low. Otherwise the search starts in negative direction.

Examples:

FE0	Causes motor to move in positive direction until the reference signal changes state. If the reference input is high when the command is issued, the motor runs toward the positive limit until the input changes to low, and vice versa.
FE1	Causes the motor to move in a negative direction until the reference input changes state.
FE2	Causes standard PI-stages to move from all starting positions towards the reference point and to approach it in correct direction.
FE3	Can be used with stages with inverted reference signal levels (none PI-standard).

FI_n Set I-Term for Force Control

C-863 DC-motor versions only: If the C-863 is used in “force control” or “position and force control” mode, this command defines the I-term for the force control algorithm.

Range: 0 to 65,535

The factory default value is 0

Corresponding report command: FI?

Examples:

FI50	Set force I-term to 350
-------------	-------------------------

See also: FA, FD, FM, FO, FP, FN, FQ, FT, TR

FI? Get I-Term for Force Control

C-863 DC-motor versions only: Reports the I-term set with FI for the force control algorithm.

Examples:

FI?	Report :
	FI: +350 (Force I-term is set to 350)

See also: FA, FD, FM, FO, FP, FN, FQ, FT, TR

FLn Define Integral Limit for Force Control

C-863 DC-motor versions only: Limits the amount of contribution by the integral gain term for the force control loop.

Range: 0 to 65,535

The factory default value is 2,000

Corresponding report command: FL?

Examples:

FL2000	Set the force I-limit to 2,000
---------------	--------------------------------

See also: FA, FD, FI, FM, FO, FP, FN, FQ, FT, TR

FL? Get Integral Limit for Force Control

C-863 DC-motor versions only: Reports the limit set with FL for the integral gain term of the force control loop.

Example:

FL?	Report:
	FL: +2000

See also: FA, FD, FI, FM, FO, FP, FN, FQ, FT, TR

FMn Set Control Mode

C-863 DC-motor versions only: This command defines the control mode.

n = 0: Position control (force control is OFF), see p. 29 for details

n = 1: Force control (position control is OFF), see p. 29 for details

n = 2: Position control and force control are ON, see p. 30 for details

Corresponding report command: FM?

Examples:

FM1	Set control mode to force control
------------	-----------------------------------

See also: FA, FD, FI, FO, FP, FN, FQ, FT, TR, FS

FM? Get Control Mode

C-863 DC-motor versions only: Reports the control mode set with FM.

Examples:

FM?	Report:
	FM:1 (Force control mode is ON)

See also: FA, FD, FI, FO, FP, FN, FQ, FT, TR, FS

FNn Set Notch Filter Frequency

C-863 DC-motor versions only: The C-863 has a notch filter on the output of the control-loop to compensate for resonances in the mechanics by reducing the corresponding frequency components in the control signal. The FN command sets the frequency of the notch filter.

Range: 40 to 10,000 Hz

The factory default value is 10,000 Hz.

Corresponding report command: FN?

Examples:

FN8000	Set notch filter frequency to 8000 Hz
---------------	---------------------------------------

See also: FA, FD, FI, FO, FP, FM, FQ, FT, TR

FN? Get Notch Filter Frequency

C-863 DC-motor versions only: Reports the frequency of the notch filter set with FN.

Examples:

FN? Report :
FN: 8000 (Notch filter frequency is set to 8000 Hz)

See also: FA, FD, FI, FO, FP, FM, FQ, FT, TR

FO_n Set Offset to Input of the Additional Sensor

C-863 DC-motor versions only: If the C-863 is used in “force control” or “position and force control” mode, this command sets an offset to the input of the additional (force) sensor.

Note that this offset is already included if you query the current value of the additional sensor with the TR command.

Corresponding report command: FO?

Examples:

FO-20 Set offset for the additional sensor to -20

See also: FA, FD, FI, FP, FM, FN, FQ, FT, TR

FO? Get Offset of Additional Sensor Input

C-863 DC-motor versions only: Reports the offset for the additional (force) sensor input set with FO.

Examples:

FO? Report:
FO: -20 (Offset of additional sensor is set to -20)

See also: FA, FD, FI, FP, FM, FN, FQ, FT, TR

FP_n Set P-Term for Force Control

C-863 DC-motor versions only: If the C-863 is used in “force control” or “position and force control” mode, this command defines the P-term for the force control algorithm.

Range: 0 to 65,535

The factory default value is 35.

Corresponding report command: FP?

Example:

FP150	Set force P-term to 150
--------------	-------------------------

See also: FA, FD, FI, FO, FM, FN, FQ, FT, TR

FP? Get P-Term for Force Control

C-863 DC-motor versions only: Reports the P-term set with FP for the force control algorithm.

Example:

FP?	Report:
	FP : +150 (Force P-term is set to 150)

See also: FA, FD, FI, FO, FM, FN, FQ, FT, TR

FQn Set Notch Filter Edge

C-863 DC-motor versions only: The C-863 has a notch filter on the output of the control-loop to compensate for resonances in the mechanics by reducing the corresponding frequency components in the control signal. The FQ command sets the edge of the notch filter, multiplied by 100.

The factory default value is 80 which corresponds to a notch filter edge of 0.8.
Corresponding report command: FQ?

Examples:

FQ50	Set notch filter edge to 0.5
-------------	------------------------------

See also: FA, FD, FI, FP, FO, FM, FN, FT, TR

FQ? Get Notch Filter Edge

C-863 DC-motor versions only: Reports the notch filter edge set with FQ.

Example:

FQ?	Report :
	FQ : 80

FSn Set Force Sign

C-863 DC-motor versions only: This command inverts the sign of the additional (force) sensor value, i.e. the sign of the value reported with TR. The sign affects the motion direction of the stage when the force control loop is active.

With the default setting (FS0), the force control loop causes the stage to move back (negative direction) if the force increases, and if the force decreases, the stage will move forwards (positive direction) until the current force matches the target force again. With FS you can invert the motion direction of the stage so that increasing force would cause the stage to move forwards (positive direction).

n = 0 = positive sign of the force sensor value:

Increasing force causes the stage to move back (in negative direction)

n = 1 = negative sign of the force sensor value

Increasing force causes the stage to move forwards (in positive direction)

Default value is n = 0.

Corresponding report command: FS?

Example:

FS1	Invert the sign of the force sensor value (can be read with TR)
------------	-----------------------------------------------------------------

See also: TR, FM, FA, FT

FS? Get Force Sign

C-863 DC-motor versions only: Reports the sign set with FS for the additional (force) sensor value (the sign affects the value reported with TR).

Example:

FS?	Report:
	FS : 0 The sign of the force sensor value is positive

See also: TR, FM, FA, FT

FTn Set Low Pass Filter Frequency

C-863 DC-motor versions only: The C-863 has a low pass filter on the input of the additional (force) sensor used for the “force control” or “position and force control” modes. The FT command sets the frequency of the low pass filter.

Range: 40 to 10,000 Hz

The factory default value is 10,000 Hz.

Corresponding report command: FT?

Example:

FT5000 Set low pass filter frequency to 5000 Hz

See also: FA, FD, FI, FO, FP, FM, FN, FQ, TR

FT? Get Low Pass Filter Frequency

C-863 DC-motor versions only: Reports the frequency of the low pass filter set with FT.

Example:

FT? Report :
 FT: 5000 (Notch filter frequency is set to 5000 Hz)

See also: FA, FD, FI, FO, FP, FM, FN, FQ, TR

GD Get D-Term for Position Control

C-863 DC-motor versions only: Reports the D-term set with DD for the position control algorithm.

Example:

GD Report:
 D: +0000000010 (P-term for position control is set to 10)

See also: DD

GH Go Home

Causes the motor to move to the currently defined zero position. Equivalent to an MA0 (Move to zero position) command.

Example:

GH	Moves motor to zero position.
-----------	-------------------------------

GI Get I-Term for Position Control

C-863 DC-motor versions only: Reports the I-term set with DI for the position control algorithm.

Example:

GI Report:
 I: +0000000130 (I-term for position control is set to 130)

See also: DI

GL Get Integration Limit for Position Control

C-863 DC-motor versions only: Reports the integration limit set with DL for the position control algorithm.

Example:

GL Report:
 M: +0000002000 (I-limit for position control is set to 2000)

See also: DL

GP Get P-Term for Position Control

C-863 DC-motor versions only: Reports the P-term (proportional gain) set with DP for the position control algorithm.

Example:

GP Report:
 G: +0000000035 (P-term for position control is set to 35)

See also: DP

HC Set Hold Current

C-663 stepper motor versions only: Sets the hold current for the stepper motor. The hold current becomes active after a programmable time after a move has terminated. Normally the hold current is about 25% of the drive current and this allows to keep the temperature of the stepper motor down, close to room temperature.

Example:

HC250	Set hold current to 250 mA
--------------	----------------------------

See also: DC, HT

HT Set Hold Time

C-663 stepper motor versions only: Sets the hold time, that is the delay time between completion of a move and the activation of the hold current.

Example:

HT500	Set hold time to 500 ms
--------------	-------------------------

See also: DC, HC

JAn Enable Trackball Mode

This command enables the trackball mode with an increment length of n counts. The trackball mode is disabled by the JF command.

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 see User manual) on the Mercury™ controller. The lines are terminated by 10k to GND.

While the trackball mode is active, move commands or joystick control are not accepted.

Corresponding report command: JA?

Example:

JA10	Enable trackball mode with increment length of 10 counts, i.e. each signal transition will shift the target by 10 counts. To suppress transient noise of the trackball input signal, a digital input filter can be activated by the SB command.
JA?	Report: JA:10
JF	Disable trackball mode
JA?	Report: JA:0 , trackball is disabled

See also: JF, SB

JA? Get Trackball Activation State

Reports the trackball mode increment length set with JA. If the increment length is 0, trackball mode is disabled. See Section 3.7 on p. 18 for further details.

Example:

JA?	Report: JA: 0 (Trackball increment length is 0, i.e. trackball is disabled)
-----	--------------------------------------------------------------------------------

See also: JA, JF, SB

JBn Set Joystick Bias

This command adds an offset n to the analog input provided by the joystick. This allows to correct the neutral (center) value of the joystick reading, e.g. if the actual joystick reading is 120, with bias 8 the reading used would be 128.

The factory default value is 0.

Corresponding report command: JB?

Example:

JB8	Adds 8 to the joystick reading.
-----	---------------------------------

See also: JN, JF, JT

JB? Get Joystick Bias

Reports the joystick bias set with JB.

Example:

JB?	Report: JB: +20 (Joystick bias is set to 20)
-----	-------------------------------------------------

See also: JN, JF, JT

JC Clear Soft Limits

This command sets the upper and lower soft limits to maximum values (+/- 2,147,483,647) so that the motion range of the stage is not limited.

Example:

JC	Clear (disable) all soft limits
----	---------------------------------

See also: JL, JH

JHn Set Upper Soft Limit

Set upper soft limit. Motion in positive direction does not exceed that position.

When a position beyond a soft limit is commanded with MA or MR, the stage moves and is stopped smoothly at the corresponding soft limit. Note that moves started with FE are not affected by the soft limits.

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.

The maximum value is +2,147,483,647 counts.

Corresponding report command: JH?

Example:

JH250000	Set upper soft limit to 250000 counts
-----------------	---------------------------------------

See also: JL, JC

JH? Get Upper Soft Limit

Report upper soft limit set with JH.

Example:

JH?	Report:
	JH: +250000 (The upper soft limit is set to 250000 counts)

See also: JL, JC

JLn Set Lower Soft Limit

Set lower soft limit. Motion in negative direction does not exceed that position.

When a position beyond a soft limit is commanded with MA or MR, the stage moves and is stopped smoothly at the corresponding soft limit. Note that moves started with FE are not affected by the soft limits.

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.

The maximum value is -2,147,483,647 counts.

Corresponding report command: JL?

Example:

JL-120000	Set lower soft limit to -120000 counts
------------------	----------------------------------------

See also: JH, JC

JL? Get Lower Soft Limit

Report lower soft limit set with JL.

Example:

JL?	Report:
	JL: -120000 (The lower soft limit is set to -120000 counts)

See also: JH, JC

JF Joystick OFF

This command disables joystick and trackball operation and returns to normal command input mode.

Example:

JF	Set joystick or trackball OFF
-----------	--------------------------------------

See also: JN, JA

JN Joystick ON

This command enables joystick operation with a maximum velocity of n counts/s or steps/s.

When the joystick is fully deflected (joystick reading 0 or 255) the motor moves with velocity n .

While the joystick mode is active, move commands or trackball control are not accepted.

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.

Example (Mercury™ Step):

JN20000	Set joystick ON with a maximum speed of 20000 steps/s
JN?	Reports: JN: 20000 (joystick is enabled)
JF	Disable joystick mode
JN?	Report: JN: 0 (joystick is disabled)

See also: JF, JT

JTn Set Joystick Table

Generate predefined joystick response table n (see "Joystick Control" on p. 15 for more information).

The parameter n determines the table to be loaded:

n = 0: Linear response table

n = 1: Cubic response table (factory default setting)

n = 2: Inverted linear response table

n = 3: Inverted cubic response table

The response table selected with JT is permanently stored in the controller until it is overwritten by another JT command or by generating a user-defined joystick response table using the SI and SJ commands.

Executing the JT command takes about 2 seconds. When completed the "Table done" report is sent.

Examples:

JT1	Set joystick table to cubic response table
JT3	Set joystick table to inverted cubic response table

See also: JN, JF, SJ

LF Limit switch OFF

Disables software limit switch operation.

The LN and LF commands affect only the software. The LF command should only be used when hardware limit switches are not installed.

This does not affect the hardware interlock in the controller circuitry, if present (C-663 and C-863). Make sure the hardware limit switch polarity is properly set on the controller: DIP switch 7 ON for active-low, OFF for active-high. PI DC motor drives typically have active-high limit switches, stepper motor drives, active-low. See the hardware User manuals for complete information.

Irrespective of the limit switch activation status, soft limits can be set using the JH and JL commands.

See also: LN, LL, LH, JH, JL

LH Limit switch active HIGH

Sets the controller to expect both limit switch inputs to be active-high (PI DC motor drives typically have active-high limit switches). When a limit switch input is greater than 3 V and limit switches are enabled with LN, motion in the corresponding direction will be terminated.

Make sure the hardware limit switch polarity is properly set on the controller: Typically DIP switch 7 ON for active-low, OFF for active-high.

See also: LL, LN, LF

LL Limit switch active LOW

Sets the controller to expect both limit switch inputs to be active-low (PI stepper motor drives typically have active-low limit switches). When a limit switch input is less than 1 volt and limit switches are enabled with LN, motion in the corresponding direction will be terminated.

Make sure the hardware limit switch polarity is properly set on the controller: Typically DIP switch 7 ON for active-low, OFF for active-high.

See also: LH, LN, LF

LN Limit switch operation ON

Enables software limit switch operation. When a limit switch is encountered during motion, motion is halted and is no longer possible in that direction as long as the switch remains closed. The target is changed to the position at which the limit switch was encountered. Movement in the reverse direction is not affected.

This does not affect the hardware interlock in the controller circuitry, if present (C-663 and C-863). Make sure the hardware limit switch polarity is properly set on the controller: DIP switch 7 ON for active-low, OFF for active-high. PI DC motor drives typically have active-high limit switches, stepper motor drives, active-low. See the hardware User manuals for complete information.

Irrespective of the limit switch activation status, soft limits can be set using the JH and JL commands.

See also: LF,LH,LL, JH, JL

MA n Move Absolute ($-1,073,741,824 < n < 1,073,741,823$)

Starts a move to absolute position n . If soft limits are set with JH and/or JL and the commanded position n is beyond a soft limit, the motion will be stopped smoothly at the soft limit. Note that the soft limits refer to the current home (zero) position. Hence they are shifted if the home position is changed.

MA commands are not accepted as long as the joystick or trackball mode are activated.

Motion is counted in counts or (micro)steps.

Example:

MA30000	Starts the motor to go to position 30000
----------------	------------------------------------------

MD n Macro Definition (range n : 1 to 31)

Defines a new macro command. Defining more than one macro with the same value of n will result in the loss of all but the last macro so defined. To define a macro, use MD followed by the desired macro number and a comma, and then write the command (base or compound command).

Examples:

MD1,MR50000,WS100,GH	Defines macro #1
MD2,TT,TP	Creates a macro command to Tell Target, then Tell Position and assigns it to number 2.
EM2	Executes macro #2, (Same as entering TT,TP)

See also: TM, RM, RZ

MF Motor OFF

Shuts off the motor current.

On C-863 DC motor versions this command turns the servo loop off; the motor no longer holds its position actively and may be moved freely. The color of the STA LED on the C-863 front panel changes from green to red when receiving an MF command.

The MF command is used to prevent unwanted movement (servo dither) or to allow for manual positioning of the unit. The motor position is still monitored in the MF status and may be queried (e.g. by the TP command).

Be careful when using this command on C-663 stepper motor versions because the controller may lose track of the motor position.

Examples:

MF CR	Sets motor servo-control to OFF
----------------------------------------------------------------------------	---------------------------------

See also : MN (Motor ON)

Servo control can be enabled again anytime by MN.

MN Motor ON

Sets the motor current back to the hold value.

On C-863 DC motor versions, MN automatically reenables servo control and sets the target to the current position so the activation of the servo loop is jerk-free. The color of the STA LED on the C-863 front panel changes from red to green when receiving an MN command.

Note: C-663 stepper motor versions may have lost track of the position if they were in the MF state.

Examples:

MN	Sets motor servo-control to ON
-----------	--------------------------------

See also : MF (Motor OFF)

MR n Move Relative

Initiates a move of relative distance of n counts/steps from the current target position. n may be either a positive or negative number, and is added algebraically to the current target to obtain the new target. The resulting absolute target position

must be between + and -1,073,741,823.

Motion is counted in encoder counts (DC motor versions) or (micro)steps (stepper versions).

If soft limits are set with JH and/or JL and the resulting absolute target position is beyond a soft limit, the motion will be stopped smoothly at the soft limit. Note that the soft limits refer to the current home (zero) position. Hence they are shifted if the home position is changed.

MR commands are not accepted as long as the joystick or trackball mode are activated.

Examples:

MR5000	Motor moves 5000 cts or steps in positive direction
MR-330	Motor moves 330 cts or steps in negative direction
MR2000,WS100,MR-1200	Motor moves 2000 cts or steps in positive, then 1200 in negative direction

See also: MA

RM Remove Macro

Used to initialize the memory reserved for one or all macro commands.

Clears (erases) one or all macros.

Example:

RM	Removes all stored macros except Macro 0
RMn	Removes macro n

RMALL Remove Macros and Parameters

Removes all macros stored, including Macro 0, and resets all parameters to their start-up values.

Example:

RMALL	Removes all macros and resets parameters
--------------	------------------------------------------

RPn Repeat n times

Causes the command string to repeat *n* times.

Limitation: $n \leq 32568$

If n is not specified, the command(s) in the string repeats indefinitely.

The repeat loop may be interrupted by sending any character except for a single-character command or a 2-character address selection code.

Example:

TP,WA250,RP12	The command TP (tell position) is repeated 12 times.
----------------------	------------------------------------------------------

RT Reset

Restarts the internal firmware operation, as if from a power-off condition.

All parameter values are restored to their defaults. If the autostart macro (Macro #0) is defined, it will be executed.

This command can not be used in a macro command.

RZ Remove Macro Zero

Used to remove the autostart macro (macro #0) from memory.

Example:

RZ	Removes the autostart macro
-----------	-----------------------------

SAn Set Acceleration

Sets the acceleration and deceleration rate in counts or (micro)steps per second squared.

Typical acceleration values:

C-663: 10,000 to 100,000; default 20,000 steps

C-863: 50,000 to 2,000,000; default 400,000 counts

Corresponding report command: TL

Example:

SA60000	Sets the acceleration and deceleration to 60,000 counts or steps / s ²
----------------	-----------------------------------------------------------------------------------

SB n Set Filter for Trackball (Digital Input)

C-863 DC-motor versions only: Enables a filter for trackball input. The parameter n determines how often the same signal level must be read before the signal transition is accepted. This suppresses transient noise and allows for stable reading.

Default value: $n = 0$ (disabled)

Corresponding report command: SB?

Example:

SB10	Sets the filter value to 10 (number of readings)
-------------	--------------------------------------------------

See also: JA

SB? Get Filter Setting for Trackball (Digital Input)

C-863 DC-motor versions only: Reports the number of readings set with SB.

Example:

SB?	Report: SB: +10 (The filter value is set to 10 (number of readings))
------------	--------------------------------------------------------------------------------

See also: JA

SC n Select Controller (0 ≤ n ≤ 15)

Puts the controller whose address is n (set with the DIP switches 1 to 4) in the *selected* state (communication enabled). Controllers with none matching address setting are set to *deselected* state (communication disabled).

This command is only allowed in the Macro 0 (autostart macro).

Example:

SC0	If run on the controller with address 0, puts it in the selected state
------------	------------------------------------------------------------------------

MD0, SC0	Assuming the controller's address is 0, then after future power-ons or resets, the device will be ready to communicate without waiting for its address selection code.
-----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------

SF*n* Set Filter Value for Encoder Input

C-863 DC-motor versions only: This command sets a low pass filter for encoder signals. Because the update rate for encoder reading is very high, noisy signals might cause faulty evaluation. The parameter *n* determines how often the same signal value must be read before it is accepted. This suppresses transient noise and allows for stable reading. Set the SF value to values of 20 to 50 to eliminate fast transients and to ensure correct counting.

Default setting is 0 (= disabled), range is $0 \leq n \leq 255$

Corresponding report command: SF?

Examples:

SF20	Set the encoder reading verification counter to 20.
-------------	-----------------------------------------------------

SF? Get Filter Value for Encoder Input

C-863 DC-motor versions only: Reports the filter setting made with SF.

Examples:

SF?	Report: SF: +20 (The encoder reading verification counter is set to 20)
------------	-----------------------------------------------------------------------------------

SI*n* Set Joystick Table Index

For user-defined joystick tables, the table value for each table index (0 to 255) can be set by command. With the SI command the table index is defined and the next write command (SJ) will write to that index location.

Example:

SI200	Sets the table index to 200
SI?	Report: I0:200

See also: SJ

SJ*n* Set Joystick Table Value

For user defined joystick tables: The table value *n* is written to the index position previously defined by an SI command. Table values range from -1024 to 1024.

With the SJ? command, the table value can be read.

Example:

SJ150	Stores the value 150 at specified table index.
SJ?	Report: SJ: +150
SI0 SJ-1024 SI1 SJ-1020 SI2 SJ-1016 ...	Command sequence to write the joystick table. The index runs from 0 to 255.

See also: SI

SMn Set Maximum Following Error (0 < n < 32,767)

C-863 DC motor versions only: Sets the maximum allowable error between the dynamic target and the actual position. May be changed as often as desired to provide maximum protection to the system. The normal following error can be monitored during motion with the TF command. For maximum system safety, use the SM command to limit following error to a value slightly above that required for normal operation.

Note that during acceleration phases the position error is larger than during constant move phase.

Range of n: 0 to 2³²

Default value: 2000

Corresponding report command: SM?

Example:

SM800	Set the maximum following error to 800
--------------	----------------------------------------

SM? Get Setting for Maximum Following Error

C-863 DC motor versions only: Reports the setting for the maximum allowable error made with SM.

Example:

SM?	Report: SM: +800
------------	----------------------------

ST Stop Motion

Stops the motor smoothly.

The controller reads the motor position when the command is received and moves the motor smoothly to this position using the current acceleration values.

Depending on the programmed acceleration rate, the motor will overshoot and then move back to the captured position.

The target is set to the current position.

Example:

ST	Stops the motor smoothly and pulls back.
-----------	------------------------------------------

See also: AB, AB1

SVn Set Velocity

Sets the speed to which the motor will be accelerated during subsequent moves. The value *n* is given in counts or (micor)steps per second.

On a DC motor, if the load changes, the controller attempts to maintain the velocity by varying the motor drive signal.

Corresponding report command: TY

Example:

sv60000	Sets the velocity of motion to 60,000 counts or steps per second.
----------------	-------------------------------------------------------------------

TAn Tell Analog Input

Analog values can be read through

n indicates the input channel.

If *n* = 0, the report is the same as with TA

n = 1 to 4 gives the input channels 1 to 4 (pins 1 to 4 of the "I/O" socket) with a resolution of 8 bits

n = 5 gives the joystick input (range 0 to 5 V) with a resolution of 8 bits

n = 6 the joystick button input with a resolution of 8 bits

Examples:

TA1	Report: A1:195
------------	-----------------------

TA	Report: A:185 190 220 230
-----------	-------------------------------------------------------------------

TB Tell Board Address (Mercury™ address setting)

Reports the address of the currently selected Mercury™ controller. The address is set with the DIP switches 1 to 4 on the controller front panel.

See "Addressing" on p. 7 for more information.

TB	B:0
-----------	------------

TCn Tell Channel

Reads the digital input channels 1 to 4 (pins 1 to 4 of the "I/O" socket).

Range of n: 0 to 4

if n = 0 then the status of all 4 channels is reported as a hexadecimal number from 0 to F with each bit corresponding to the status of one of the channels.

(LSB = ch 1)

Examples:

Command:	Report
TC1	"H01:1" (channel 1 high)
TC2	"H02:0" (channel 2 low)
TC0	"H00:F" (all channels 1 to 4 high)
TC0	"H00:0" (all channels 1 to 4 low)
TC0	"H00:5" (channels 1 and 3 high, 2 and 4 low)
TC	Same as TC0

(Digital channels can also be read with the single character command "#")

TD Tell Dynamic target

C-863 DC motor versions only: Reports the instantaneous value of the dynamic target. As the motor is moved along the programmed path to the (final) target, a "dynamic target" is used to define the trajectory and control the position at each instant along the way.

Examples:

Command:	Report
TD	"N:+0000126317"

TE Tell Error

C-863 DC motor versions only: Reports the position error of the motor, as determined by subtracting the actual position from the target position.

Examples:

Command:	Report
TE	"E:+0000000015"
TE,WA200,RP99	The report is delivered 5 times a second for 100 times

TF Tell Following error

C-863 DC motor versions only: Reports the difference between the dynamic target and the actual position. During motion, it is normal for the actual position to lag behind the dynamic target position by some amount, usually dependent on the programmed motion parameters.

If the programmed velocity is higher than physically possible for the system, or if an obstruction has been encountered, the Following Error will increase. If the obstruction is temporary, the servo-controller will attempt to reduce the following error to zero when the obstruction has been overcome. If the condition is not temporary, the following error will typically increase until the programmed limit is reached

TI Tell Iterations

Reports the current value of the repeat counter. It can be used in a repeat loop to determine how many loops are still to perform.

Example:

MR100,WS100,WA250,TI,RP99

The motor will make repetitive moves of 100 steps, with a delay of 0.25 seconds between steps, for a total of 100 times. The TI command will report the number of iterations remaining to be performed after each iteration. Note that this command must be in a repeat loop to be useful.

Report1:	"X:+0000000000" (First TI is executed before the number of loops is specified!)
Report2	"X:+0000000099" and so on.

TL Tell Acceleration

Reports acceleration value setting (not the current acceleration). This is the acceleration the controller will use at the beginning and end of a move.

Example:

Command:	Report
TL	L:+0000170000

See also: SA (Set Acceleration)

TM[n] Tell Macros (0 ≤ n ≤ 31)

Displays one or all currently stored macro commands. If n = 0 or not specified, all macros (except the autostart macro) will be displayed. To display the autostart macro, use the TZ command. Since macros may be defined in any order, the TM command is useful for confirming the existence of, as well as listing the contents of a macro.

Example:

Command:	Report
TM1	MC001 MR55555
TM0	MC001 MR55555 MC002 MA120000

See also: TZ (Tell Macro 0)

TP Tell Position

Reports the absolute position of the motor. TP may be used to monitor motion during both motor ON and motor OFF status. With stepper motor versions, only commanded motion is accounted for.

(Also implemented as single-character command “'”, apostrophe)

Command:	Report
TP	"P:+0000005555"
TP,WA100,RP	causes the controller to report the position every 100 ms

TR Tell Current Value of Additional (Force) Sensor

C-863 DC motor versions only: This command reports the current value of the additional sensor used for the force control loop, i.e. the input signal on pin 2 of the "Joystick" socket. This can be, for example, a force sensor measuring the current force the stage imposes on an object. The input voltage range is -10 to +10 V. The voltage value is converted by a 12 bit A/D converter, resulting in a range of -2048 to 2048 for the reported value.

The value reported with TR includes the offset set with FO, and was already filtered by the low pass filter set with FT. Furthermore, the sign settings made with FS (Set Force Sign) are already applied to the signal.

Example:

TR	Report :
	R:+0000000158

See also: FA, FD, FI, FP, FO, FM, FN, FQ, FT, FS

TS Tell Status

Reports the status of the system, its motion and limit switch states.

(Also implemented as single character command "%")

The status report differs for C-863 DC motor versions and C-663 stepper motor versions.

C-863 Mercury™ DC motor controllers:

Example:

Command:	Report
TS	"S:04 02 00 03 02 00"

The status report string consists of 6 blocks of 2 characters each with the format:

S:B1 B2 B3 B4 B5 B6

Each block of 2 characters represents one byte (8 bit).

The first character in the block represents bits 4 to7 while the second character in the block represents bits 0 to 3.

Block 6 contains an error code.

Block 1	System status	Bit 0: not used Bit 1: not used Bit 2: On Target Bit 3: Reference Search Success	2 nd character
		Bit 4: not used Bit 5: not used Bit 6: not used Bit 7: Motor servo loop OFF	1 st character
Block 2	Internal operation flags	Bit 0: not used Bit 1: internal use (busy) Bit 2: not used Bit 3: not used	2 nd character
		Bit 4: Macro running Bit 5: not used Bit 6: not used Bit 7: not used	1 st character
Block 3	Motor loop flags	Bit 0: not used Bit 1: not used Bit 2: Move direction polarity Bit 3: not used	2 nd character
		Bit 4: not used Bit 5: not used Bit 6: not used Bit 7: not used	1 st character
Block 4	Signal Lines Status	Bit 0: Limit sensors ON Bit 1: Limit sensors active HIGH Bit 2: Find reference operation in progress Bit 3: Brake ON	2 nd character
		Bit 4: Joystick ON Bit 5: not used Bit 6: not used Bit 7: not used	1 st character
Block 5	Signal Lines Inputs	Bit 0: not used Bit 1: Reference signal (input) Bit 2: Positive limit signal (input) Bit 3: Negative limit signal (input)	2 nd character
		Bit 4: DIO 1 Bit 5: DIO 2 Bit 6: DIO 3 Bit 7: DIO 4	1 st character
Block 6	Error Codes	00: no error 01: RS-232 timeout 02: RS-232 overflow 03 Macro storage full 04 Macro out of range 05 Macro wrong com 06 Wrong Command 07 Hard stop 08 not defined. 09 Position following error 0A Move attempt while servo off 0B Move attempt while joystick ON	

C-663 Mercury™ Step stepper motor controllers:

Example:

Command:	Report
TS	S:03 07 00

The status report string consists of 3 blocks of 2 characters each with the format:

S:B1 B2 B3

Each block of 2 characters represents one byte (8 bit).

The first character in the block represents bits 4 to7 while the second character in the block represents bits 0 to 3.

Block 6 contains an error code.

Block 1	System Status	Bit 4: Macro running Bit 5: Motor OFF Bit 6: Brake ON Bit 7: Drive current active	1 st character
		Bit 0: Ready Bit 1: On target Bit 2: Reference drive active Bit 3: Joystick ON	2 nd character
Block 2	Signal Line Status	Bit 4: Digital input 1 Bit 5: Digital input 2 Bit 6: Digital input 3 Bit 7: Digital input 4	1 st character
		Bit 0: Limit negative Bit 1: Reference signal Bit 2: Limit positive Bit 3: no function	2 nd character
Block 3	Error codes	00: No Error 01: RS-232 timeout 02: RS-232 overflow 03: Macro storage full 04: Macro out of range 05: Wrong macro command 06: Command error	

TT Tell Target

Reports the target position. This is the absolute position to which the motor was commanded. During motion this target may differ from the dynamic target used by the controller to maximize conformance to the proper motion profile (see the TF command, p. 77)

The target position may be specified directly with the MA (Move Absolute) and several other commands, or indirectly with the MR (Move Relative) command.

Example:

Command:	Report
TT	T: +0000005000

TV Tell current velocity

Reports the current profile velocity in c/s or steps/s.

Note that this command can not be used to trace the physical velocity of the motor. It report the current velocity of the motion profile calculated by the profile generator. During motion, outside the acceleration and deceleration phases, the value reported will equal the velocity programmed with SV.

Example:

Command:	Report
TV	V: +0000016777

See also: TY, SV

TY Tell programmed velocity

Reports the current programmed velocity setting (not the current velocity). This value can be changed with the SV command.

Example:

Command:	Report
TY	Y: +0000060000

See also: TV, SV

TZ Tell macro zero

Reads the autostart macro (Macro #0).

The autostart macro, as defined by the "MD0,xxx" command, is automatically executed upon power-on or reset.

Example:

Command:	Report
TZ	MC000 MR5000,WS100,GH

See also: RZ

VE Version report

Reports the copyright notice and revision level of the installed firmware.

Example:

Command:	Report
VE	(c)2009 Physik Instrumente(PI) Karlsruhe, C-863, Ver. 2.30, 2009-21-07

WAn Wait Absolute (0 < n < 65,535)

Inserts a wait period of n milliseconds before executing the next command.

Example:

Command:	Function
MR2000,WA3000,MR-2000	This command line will move the motor by 2,000 counts or steps, then, 3 seconds after the start of the move, the motor will move back 2,000 counts or steps. Note that the wait period of 3 seconds includes the time the motor is moving.
MR2000,WS3000,MR-2000	This command line will move the motor by 2,000 counts or steps, then, after terminating the move it waits for 3000 ms until it moves back for 2000 counts or steps.

WF n Wait for channel n OFF (range of n : 1 to 4)

Waits for digital I/O channel n to be OFF.

Can be used for command sequencing. It waits until input channel n is OFF before continuing program execution.

Example:

Command:	Function
WF2,MR5000	This command waits until input channel 2 is OFF (low), then starts the relative move for 5000 counts or steps.

WN n Wait for channel n ON (range of n : 1 to 4)

Waits for digital I/O channel n to be ON.

Can be used for command sequencing. It waits until input channel n is ON before continuing program execution.

Example:

Command:	Function
WN2,MR5000	This command waits until input channel 2 is ON (high), then starts the relative move for 5000 counts or steps.

WS n Wait for motor stop

Waits until the motor is on target (i.e. on-target flag in the TS report is true) and then waits for another n milliseconds before continuing to the next command.

If the parameter n is omitted, the default wait time of 1000 ms is used.

See also “On-Target Detection” on p. 31.

Examples:

Command:	Function
MR5000,WS100,RP	Moves 5000 counts or steps, then waits until the motor has reached its target, then waits for another 100 ms before repeating the command line

See also: WT, WW

WT_n Set Settle Time

C-863 DC-motor versions only: This command sets the settle time to *n* milliseconds.

Used for on-target detection: The on-target status becomes "true" when the actual position stays in the settle window (set with WW) for at least the settle time. If the settle time is set to 0 (**WT0**), then the axis is on target when the calculated motion trajectory has finished, irrespective of the actual position.

Factory default: 0

Corresponding report command: WT?

Command:	Function
WT50	Set the settle time to 50 ms. The current position must be for at least 50 ms in the settle window before the on-target status becomes true.

See also: WW, WS, TS

WT? Get Settle Time

C-863 DC-motor versions only: This command reports the settle time set with WT in milliseconds.

WT?	Report:
	WT : 50

See also: WW, WS, TS

WW_n Set Settle Window

C-863 DC-motor versions only: This command sets the settle window to *n* counts.

The settle window is centered around the target position. The on-target status becomes "true" when the actual position stays in this window for at least the settle time (set with WT). If the settle time is set to 0 (**WT0**), then the axis is on target when the calculated motion trajectory has finished, irrespective of the settle window settings.

Factory default: 5

Corresponding report command: WW?

Command:	Function
WW2	Set the settle window to 2 counts. The actual motor position must stay inside a window of ± 2 count around the target for at least the settle time before the on-target flag is set.

See also: WT, WS, TS

WW? Get Settle Window

C-863 DC-motor versions only: This command reports the settle window set with WW in counts.

WW?	Report:
	WW: 5

See also: WT, WS, TS

XF n Execute if OFF

Execute the remainder of the command line only if input channel n is OFF (0 V).

Range of n : 1 to 6

See XN for channel assignment.

Example:

Command:	Function
XF3,MR5000	This command moves the motor by 5000 counts/steps only if input channel #3 is low (GND).

XN n Execute if ON

Execute the remainder of a compound command only if input channel n is ON (+5 V).

Range of n : 1 to 6

Channel assignment:

1 to 4 : Digital input channel 1 to 4 (pins 1 to 4 of the "I/O" socket)

5: Joystick input

6: Joystick button

Command:	Function
XN3,MR5000	This command moves the motor by 5000 counts or steps only if the input channel #3 is high (+5 V).
TP,XN1,TF	After reading the current position, the profile error is read only if digital input #1 is high (+5 V).
XN1,XN2,TP	The TP command is executed only when digital input channels #1 and #2 are high at the same time.

See also: XF, WN, WF

ZIn Set Trigger Increment

This command defines the position distance (in counts or steps) for consecutive trigger signal outputs on the digital output line 4 (pin 8 of the "I/O" socket, see C-863 or C-663 User manual for pinout). The increment value can be positive or negative.

The setting is effective if the trigger mode is set to 2 (multiple trigger pulses) or 3 (trigger "grid" mode) with ZMn.

Corresponding report command: ZI?

Example:

ZI25000	Set the trigger increment to 25000 counts or steps
----------------	----------------------------------------------------

See also: ZM, ZP

ZI? Get Trigger Increment

Reports the position distance (in counts or steps) set with ZIn for consecutive trigger signal outputs.

Example:

ZI?	Report: ZI : +25000
------------	-------------------------------

See also: ZM, ZP

ZM*n* Set Trigger Mode

Defines the trigger output mode *n* for the digital output line 4 (pin 8 of the “I/O” socket, see C-863 or C-663 User manual for pinout).

n = 0 Trigger OFF (factory default)

n = 1 Single trigger pulse:

A trigger pulse is written when the axis has reached the trigger position given by ZP*n*.

n = 2 Multiple trigger pulses:

The first trigger pulse is written when the axis has reached the trigger position given by ZP*n*. The next trigger pulses each are written when the axis position equals the sum of the last valid trigger position and the increment value given by ZIn.

n = 3 Trigger “grid” mode:

A trigger pulse is written whenever the axis has covered the distance given by ZIn (the value is used for both motion in positive and negative direction)

Corresponding report command: ZM?

Example:

ZM2	Set the trigger mode to 2 (multiple trigger pulses)
------------	-----------------------------------------------------

See also: ZP, ZI

ZM? Get Trigger Mode

Reports the trigger mode set with ZM*n*.

Example:

ZM?	Report: ZM : 2
------------	--------------------------

See also: ZP, ZI

ZP*n* Set Trigger Position

This command defines the position (in counts or steps) where the first trigger pulse is to be output on the digital output line 4 (pin 8 of the “I/O” socket, see C-863 or C-663 User manual for pinout). The setting is applied if the trigger mode is set to 1 (single trigger pulse) or 2 (multiple trigger pulses) with ZM*n*.

The trigger position value can be positive or negative.

Corresponding report command: ZP?

Example:

ZP30000	Set the trigger position to 30000
----------------	-----------------------------------

See also: ZM, ZI

ZP? Get Trigger Position

Reports the trigger position (in counts or steps) set with ZPn.

Example:

ZP?	Report: ZP : +30000
------------	-------------------------------

See also: ZM, ZI

