MP-285

Micromanipulator System

(Including Model MP-285A Controller)

Reference Manual

Rev. 4.09 (20210122)

SUTTER INSTRUMENT®

One Digital Drive
Novato, CA 94949

Voice: 415-883-0128 Web: www.sutter.com
Fax: 415-883-0572 Email: info@sutter.com

CE EU Declaration of Conformity

Application of Council Directives:
2014/30/EU (EMC), 2014/35/EU (LVD), and 2015/863/EU (RoHS 3)

Manufacturer's Name:	Sutter Instrument Company
Manufacturer's Address:	One Digital Drive Novato, CA. 94949 USA Tel: +1 415 883 0128
Equipment Tested:	**MP-285** Micromanipulator System
Model(s):	**MP285/E** or **MP285A/E** (controller), **ROE285** (Rotary Optical Encoder) or **285JOY** (Joystick), & **MP285/M** (micromanipulator electromechanical)

Conforms to Standards:	EMC Emissions:	EN 61326-1, including: EN 55011: 2009 Class B; EN 61000-3-2: 2015, &	 EN 61000-3-3: 2014
	EMC Immunity:	EN 61000-4-2: 2009, EN 61000-4-4: 2012, EN 61000-4-6: 2014, EN 61000-4-11: 2004	EN 61000-4-3: 2011, EN 61000-4-5: 2014, EN 61000-4-8: 2010, &
	LVD (Safety):	EN 61010-1: 2010	
Tested/Verified By:	TUV/BABT Product Service 4855 Patrick Henry Dr., Building # 6 Santa Clara, CA 95054 USA Sutter Instrument		
Test Report(s):	A0022E01, A0022i01, AI102160, SI_EMC_MP-285_20160713, 20200520_REACH and RoHS Compliance Statement		

Sutter Instrument Company hereby declares that the equipment specified above was tested and conforms to the EU Directives and Standards listed above, and further certifies conformation to the requirements of the European Union's Restriction on Hazardous Substances in Electronic Equipment Directive 2015/863 (2011/65/EU Annex II) for RoHS 3.

Project Engineer:

Mark Flaming
President

SUTTER INSTRUMENT®

One Digital Drive, Novato, CA 94949 USA Phone: +1 415 883 0128 Fax: +1 415 883 0572
Email: info@sutter.com Web: http://www.sutter.com

DISCLAIMER

The **MP-285**-system consists of one controller (MP-285 or MP-285A), a user control device (ROE (MP-285/MP-285A) or Joystick (MP-285 only)), and one electromechanical micromanipulator device (MP-285/M). The purpose of the system is for the manipulation at the micro level of micropipettes and probes used in conjunction with a microscope. No other use is recommended.

This instrument is designed for use in a laboratory environment. It is not intended, nor should it be used in human experimentation or applied to humans in any way. This is not a medical device.

Do not open or attempt to repair the instrument. High voltages are present and inadvertent movement of the micromanipulator electromechanical could cause injury.

Do not allow unauthorized and/or untrained operative to use this device.

Any misuse will be the sole responsibility of the user/owner and Sutter Instrument Company assumes no implied or inferred liability for direct or consequential damages from this instrument if it is operated or used in any way other than for which it is designed.

SAFETY WARNINGS AND PRECAUTIONS

Electrical

- **Operate the MP-285 using 110-- 240 V AC., 50-60 Hz line voltage. This instrument is designed for use in a laboratory environment that has low electrical noise and mechanical vibration. Surge suppression is always recommended. The MP-285 controller has a manual voltage select switch as part of the power input connector on the rear panel. The MP-285A has automatic voltage switching and therefore does not have a voltage-select switch.**

- **Fuse Replacement: Replace only with the same type and rating:**

Table 1. MP-285 Mains fuse type and rating.

Mains Voltage Setting	Fuse: (Type: Time Delay, 5 x 20 mm, glass tube)	
	Rating	Manufacturer Examples
"110" (100 – 120 VAC)	2A, 250V (Time Delay)	Bussmann: GMC-2A/GDC-2A or GMC-2-R/S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)
"220" (200 – 240 VAC)	T1.0A, 250V	Bussmann: GDC-1A or S506-1A (RoHS) Littelfuse: 218 001 or 218 001.P (RoHS)

Table 2. MP-285A Mains fuse type and rating.

Mains Voltage	Fuse: (Type: Time Delay, 5 x 20 mm, glass tube)	
	Rating	Manufacturer Examples
100 - 240 VAC	2A, 250V (Time Delay)	Bussmann: GMC-2A/GDC-2A, or GMC-2-R/S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)

A spare fuse is in the power input module. Please refer to the fuse-replacement appendix for more details on fuse ratings and for instructions on how to change the fuse.

Avoiding Electrical Shock and Fire-related Injury

- ⚠ Always use the grounded power supply cord set provided to connect the system to a grounded outlet (3-prong). This is required to protect you from injury if an electrical hazard occurs.
- Do not disassemble the system. Refer servicing to qualified personnel.
- ⚠ To prevent fire or shock hazard do not expose the unit to rain or moisture.

Electromagnetic Interference

To comply with FDA and CE electromagnetic immunity and interference standards; and to reduce the electromagnetic coupling between this and other equipment in your lab always use the type and length of interconnect cables provided with the unit for the interconnection of the ROE, MP-285/M electromechanical device, and host computer via serial RS-232 interface, (see the Technical Specifications appendix for more details).

Operational

Failure to comply with any of the following precautions may damage this device.

- This instrument is designed for operation in a laboratory environment (Pollution Degree I) that is free from mechanical vibrations, electrical noise and transients.
- This unit is not designed for operation at altitudes above 2000 meters nor was it tested for safety above 2000 meters.
- ⚠ DO NOT CONNECT OR DISCONNECT THE CABLES BETWEEN THE CONTROLLER AND THE MECHANICAL UNITS WHILE POWER IS ON.

  Please allow at least 20 seconds after turning the unit off before disconnecting the mechanical units. Failure to do this may result in damage to the electronics.
- Operate this instrument only according to the instructions included in this manual.
- Do not operate if there is any obvious damage to any part of the instrument.
- ⚠ Operate only in a location where there is a free flow of fresh air on all sides. NEVER ALLOW THE FREE FLOW OF AIR TO BE RESTRICTED.
- ⚠ Do not operate this instrument near flammable materials. The use of any hazardous materials with this instrument is not recommended and if undertaken is done so at the users' own risk.
- ⚠ Do not attempt to operate the instrument with the manipulator shipping screws in place - severe motor damage may result.
- ⚠ Do not operate if there is any obvious damage to any part of the instrument. Do not attempt to operate the instrument with the manipulator shipping screws in place - severe motor damage may result. When transporting the mechanical manipulator, be sure to install the shipping screws supplied in their correct locations. Failure to do this may result in damage to the motors.

- ⚠ Never touch any part of the micromanipulator electromechanical device while it is in operation and moving. Doing so can result in physical injury (e.g., fingers can be caught and pinched between the moving parts of the micromanipulator).
- ⚠ As with all microinjection devices, sharp micropipettes can fly out of their holder unexpectedly. Always take precautions to prevent this from happening. Never loosen the micropipette holder chuck when the tubing is pressurized, and never point micropipette holders at yourself or others. Always wear safety glasses when using sharp glass micropipettes with pressure microinjectors.
- ⚠ Do not handle the manipulator mechanical while the power is on and take care to ensure no cables pass close to the mechanical manipulator.

Other

- Use this instrument only for microinjection purposes in conjunction with the procedures and guidelines in this manual.
- Retain the original packaging for future transport of the instrument.
- Some applications, such as piezo-impact microinjection call for the use of mercury in the micropipette tip. The use of any hazardous materials with any Sutter Instrument's instrument is not recommended and if undertaken is done so at the users' own risk.
- When transporting the mechanical manipulator, be sure to install the shipping screws supplied in their correct locations. Failure to do this may result in damage to the motors.
- This instrument contains no user-serviceable components — do not open the instrument casing. This instrument should be serviced and repaired only by Sutter Instrument or an authorized Sutter Instrument servicing agent.
- Sutter Instrument reserves the right to change specifications without prior notice.
- This device is intended only for research purposes.

Handling Micropipettes

⚠ Failure to comply with any of the following precautions may result in injury to the users of this device as well as those working in the general area near the device.

- The micropipettes used with this instrument are very sharp and relatively fragile. Contact with the pulled micropipette tips, therefore, should be avoided to prevent accidentally impaling yourself.
- Always dispose of micropipettes by placing them into a well-marked, spill-proof "sharps" container.

(This page intentionally left blank.)

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. GENERAL INFORMATION

1.1 About the Reference Manual

The MP-285 controller has been designed to fill the diverse needs of a wide range of applications in the life sciences, engineering, and commercial sectors. The MP-285 uses a system of menus, visualized on the controller display and accessed by keypad keystrokes, to provide for these varied uses. A Menu tree, consisting of the complete MP-285 menu hierarchy can be found in APPENDIX E.

The starting point for menu tree access is the Main Menu:

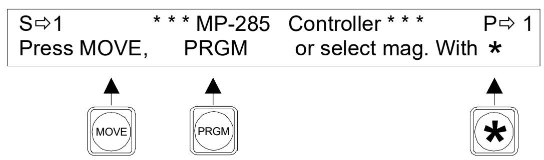


Figure 1-1. Using the Main Menu as a starting point for menu tree access.

The main menu is displayed soon after the controller is turned on (a startup screen is briefly displayed first). From the main menu you have access to three main branches of the menu tree using the keys <PRGM>, <MOVE> and <*>. A main fourth branch is accessed by pressing <MOVE> followed by <*>.

A large portion of the Reference Manual is organized according to the MP-285 menu structure. The following table lists the major manual sections and the keypad keystrokes used to reach the corresponding menu branch beginning at the Main (startup) Menu:

Table 1-1. Manual sections describing Main Menu keystrokes.

Manual Section	Keystrokes from Main Menu
Controller Configuration	<PRGM>
Movement Screen	<MOVE>
Robotic Movement	<MOVE><*>
Setup Selection	<*>

Within each of these manual sections, you will find a description of each sub-menu item as well as a systematic list of the keystrokes necessary to access and use each command.

In addition to sections describing menu items, the reference manual also has in-depth sections on input devices, manipulator installation, and the serial computer interface. The section describing input devices is presented first because some commands are specific to a particular input device and rely on a sound understanding of how the device works.

Several areas are not detailed here because they are covered in the Operations Manual. These are unpacking, basic setup, and the initial instruction on MP-285 operations for the beginner. If you are a first-time user of the MP-285, you may wish to consult the Operations Manual first. The Operations Manual presents the most important information in a concise fashion to limit the amount of time necessary for a first-time user to begin experimental micromanipulation with the MP-285. In contrast, the Reference manual covers all aspects of the MP-285 in depth and may be more useful to experienced MP-285 users.

2. INPUT DEVICES

The MP-285 is usually supplied with one of two input devices: a Rotary Optical Encoder (ROE) or a Joystick (MP-285 only). Much of the operation of the MP-285 is dependent on the type of input device and how the device interacts with the controller to make the manipulator move. The following section describes the operation of each input device. Several sections of the Reference Manual are split into sub-sections according to the type of input device being used.

Each input device has three separate input controls: one for each axis of the manipulator. The controls are designated by which input channel they control. Each of the three colored LEDs on the front panel of the MP-285 controller corresponds to an input channel. The table below details this correspondence:

Table 2-1. Input device indicator LED assignments on front panel.

Input Channel (Front panel LED color)	Input Device	
	ROE Knob	Joystick Movement
Yellow	Top	Twist
Red	Left	Left/Right
Green	Right	Up/Down

2.1 Rotary Optical Encoder (ROE)

The Sutter Rotary Optical Encoder controls movement of the manipulator by using a separate knob for each of the axes. Each knob turns the shaft of a digital encoder; a full turn of an encoder produces 512 pulses or “clicks”. The pattern and speed of the clicks produced by the ROE are directly related to how the knob is turned. The digital output of the ROE is translated by the MP-285 controller into signals that drive the micromanipulator stepper motors.

The precise relationship between knob movement and micromanipulator response will depend on several adjustable parameters that are discussed in other sections of this manual.

In addition to the knobs for each of the three axes, the ROE also has four buttons that allow easy access to four functions of the MP-285 controller. The buttons control the following functions:

- Changing from continuous to pulse movement mode
- Switching from course to fine microstep size
- Selecting either diagonal or normal movement mode
- Sending the manipulator to a predefined “Home” location

The following figure is presented so that you can familiarize yourself with the controls on the ROE.

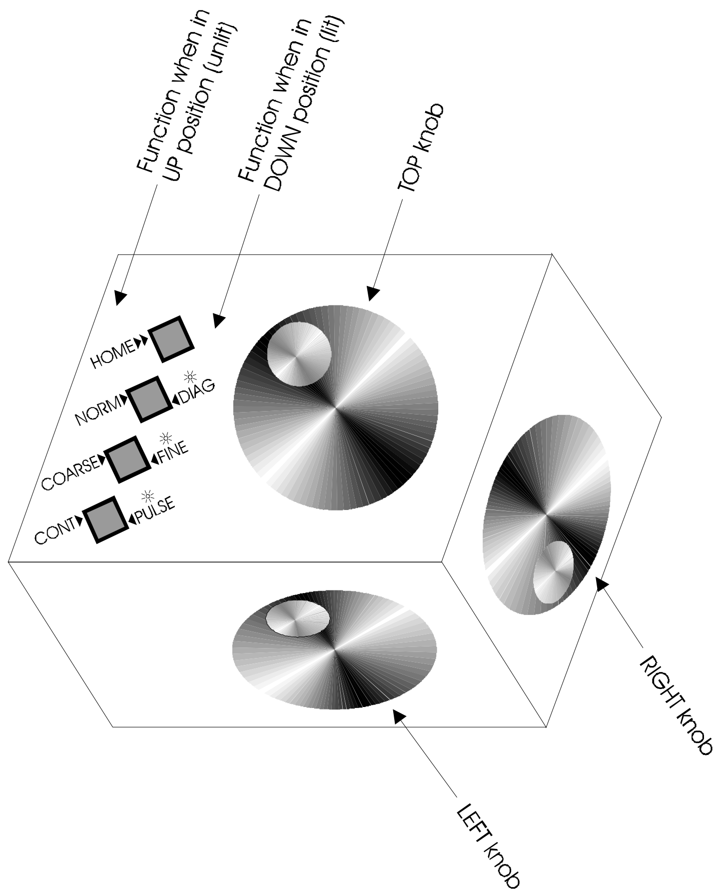


Figure 2-1. Rotary Optical Encoder (ROE).

2.2 Joystick (MP-285 controller only)

The joystick provided with the MP-285 is a three-axis controller. A pair of handle movements controls each axis' left/right deflections, up/down deflections or clockwise/counterclockwise twists. These movements are coupled to three potentiometers. The displacement of the joystick handle is transduced into an electrical analog output that is converted by the MP-285 controller into signals that drive the micromanipulator stepper motors. The relationship between handle deflection and response will not necessarily be a linear one. Joystick response characteristics are adjustable and are discussed in other sections of this manual.

The button on the end of the joystick handle is used for initiating movements when the controller is in Pulse movement mode.

The following figures show the controls present on the present and older versions of the Joystick.

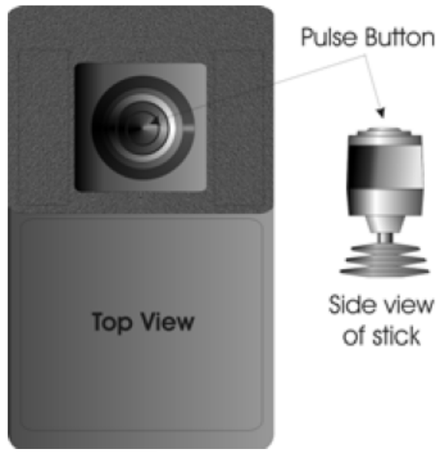


Figure 2-2. Joystick.

With the older version of the Joystick, the handle is also mechanically linked to return springs which, when enabled, will center the handle along its left/right and up/down axes. These springs can be disabled (partially or fully) by the slide switches located near the base of the handle (see Figure 2-3). If your joystick handle does not center itself, adjust these switches to enable the return springs.

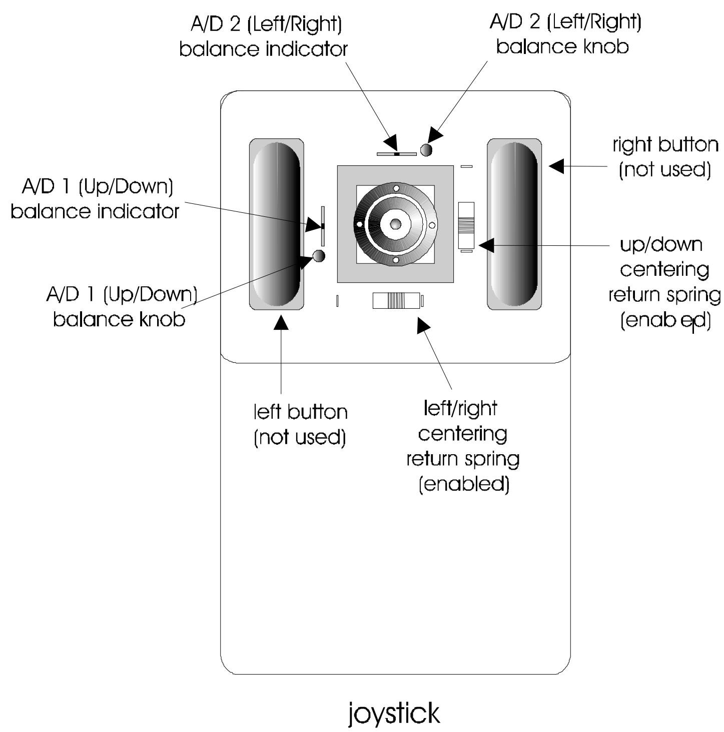


Figure 2-3. Joystick (older version).

2.3 Adjustment of Input Device Characteristics

As indicated, the response characteristics of each of the input devices are adjustable. The following list outlines the part of the controller menu and the corresponding part of this manual that deals with each adjustment.

- **Control of which input device channel is assigned to which manipulator stepper motor axis.** The MP-285 software allows full control over which ROE knob or pair of joystick moves is connected to each of the three physical axes of the manipulator. Furthermore, one can control the direction of movement commanded by a given input. Axes assignment is covered in the manual under Controller Configuration and is adjusted using the menu command **[PRGM\Setup\Axes]**.
- **Adjustment of the resolution of micromanipulator movement produced by the input device.** The distance and the speed at which the MP-285 manipulator moves in response to a given amount of input are controlled by several parameters. These are described in detail in the manual under Controller Configuration and are adjusted using the menu command **[PRGM\Continuous]** for continuous movement and the command **[PRGM\Pulse Mode]** for pulse movement. A subset of these parameters can also be adjusted during movement using the menu command **[MOVE\1,2,4 or 5]** as described in the Movement Screen section of the manual.
- **Set Dead Zone for Joystick (MP-285 controller only).** It is possible to set an amount of output from the joystick that will be ignored by the controller. This is necessary so that baseline signal is not interpreted as a movement command to be acted on by the controller. The adjustment of this parameter, referred to as the Dead Zone, is performed under the menu command **[PRGM\Setup\Hardware]** and is described in the Controller Configuration section of this manual.
- Finally, for the Joystick only, it is possible and occasionally necessary to make periodic adjustments to the hardware. This adjustment is detailed in APPENDIX B.

3. CONTROLLER CONFIGURATION [PRGM]

Many of the parameters that determine how the MP-285 controller responds to user input are completely adjustable. While many users will find no need to adjust these parameters, this section of the Reference Manual contains all the information necessary to understand and adjust all the parameters affecting controller function.

To enter the menu that will allow you to alter the configuration of your MP-285 controller do the following:

1. Turn on the power and observe the Main menu:

```
S⇨1        * * * MP-285   Controller * * *      P⇨ 1
Press MOVE,     PRGM       or select mag. with  *
```

2. Press <PRGM> on the controller keypad to bring up the Configuration menu:

Setup currently loaded ↓

```
█⇨Continuous        ⇨Pulse mode          Setup 1
 ⇨Setup             ⇨Save/Load
```

The configuration menu shows four possible sub menus that can be accessed from this location. It also gives the number of the currently loaded "Setup" where many of the controller parameters are stored.

Parameter settings saved in a Setup memory slot determine the response characteristics of the MP-285. The configuration menu and its sub menus contain commands for:

- Viewing and loading existing SETUPS (**[PRGM\Save/Load\View & Load]**).
- Adjusting SETUP parameters (**[PRGM\Continuous]**, **[PRGM\Pulse mode]** and **[PRGM\Setup]**).
- Saving adjusted parameters **[PRGM\Save/Load\Save & Clear]**.

The configuration menu also has provisions to adjust "global" parameters that are not affected by nor stored in the currently loaded SETUP. They are however, "remembered" when the controller is off. These functions include:

- Configuring the diagonal or 4th axis ([PRGM\Setup\4th axis]).
- Setting the location of the ABSOLUTE origin (**[PRGM\Setup\Axes\New origin]**).
- Modifying the controller to match various input devices (**[PRGM\Setup\Hardware]**).
- Configuring the RS-232 port (**[PRGM\Setup\Utilities\Baud rate]**)
- Configuring pause status in a Robotic routine (**[PRGM\Setup\Utilities\pause]**)

Finally, there are other utility functions available from within the configuration menu to allow the user to:

- Restore default SETUPS and global parameters (**[PRGM\Setup\Utilities\Reset all values]**)
- Troubleshoot input devices (**[PRGM\Setup\Utilities\AD test]**)
- Test serial port I/O (**[PRGM\Setup\Utilities\SIO test]**)
- Run the automatic centering routine (**[PRGM\Setup\Utilities\center axis]**)

The remainder of this section gives the details of how to alter each of the parameters previously listed. It is organized according to the sub-menu structure of the MP-285 controller.

3.1 Continuous Mode Sub-Menu [PRGM\Continuous]

When the **Continuous** movement mode is selected, input commands yield a continuous movement with variable velocity.

Adjustable parameters in the **Continuous mode sub-menu** vary with the type of input device connected to the MP-285 controller. The controller automatically identifies the input device and presents the appropriate menu. This is necessary because input devices can have either digital (Rotary Optical Encoder) or analog (Joystick) output.

The following discussion of the **Continuous mode sub-menu** is therefore organized by input device type. Joystick users may skip section I and go directly to section II.

3.1.1 I. FOR USERS OF THE ROTARY OPTICAL ENCODER (ROE)

The ROE produces digital output. Thus, the motion of the knobs is “directly” coupled to the micromanipulator’s movement. In continuous mode, micromanipulator velocity corresponds directly to knob turning velocity.

The movement of the ROE knob is “encoded” as digital pulses or **“clicks”**. There are 512 **clicks** for every full turn of a knob. Based on the settings of the controller, each **click** commands the execution of a given number of stepper motor **µsteps**.

To change the responsiveness of the ROE, you simply change the number of **µsteps/click** as follows:

1. Press <PRGM> from the main menu to access the configuration menu:

█ ⇨Continuous	⇨Pulse mode	Setup 1
⇨Setup	⇨Save/Load	

2. Press **<ENTR>** to access the Continuous mode submenu:

Number of µsteps/click
↓

7	=	1.4	µm/Click	@	0.2
Steps/click	=	0.28	µm/Click	@	0.04

3. Using the keypad, enter the desired number of µsteps/click.
4. Press **<ENTR>** to accept the new value or **<ESC>** to abort and restore the original value. Note that the right-hand portion of the screen updates to reflect the new value and calculates the number of microns/click for the two different available microstep sizes.
5. Press the **<ESC>** key to return to the **Programming menu**.

It is recommended that settings ≤10 be used. At very high **µsteps/click** settings (>20), it is possible to move the knob much faster than the micromanipulator can respond. In this case, ROE input is ignored.

Note that **µsteps/click** can also be set from within the Movement screen while in the Continuous mode (**[MOVE\1,2,4 or 5]**). See the Reference manual section on the Movement Screen Menu.

3.1.2 II. FOR USERS OF THE JOYSTICK (MP-285 controller only)

The MP-285 manipulator is designed so that the larger the deflection of the handle (on the joystick) the faster the movement of the manipulator. The controller handles this by converting the amount of deflection (a change in voltage divider) into an output of appropriate frequency (µsteps/second) applied to the micromanipulator stepper motors.

The conversion of the analog voltage from the divider to a signal of desired frequency is quantified by the parameters: **Speed Range, Offset, Jump to Max at… and Max Speed**. The four parameters are described in detail below and each can be altered within the **Continuous mode sub-menu** if a Joystick is attached.

To alter the four parameters:

6. Press **<PRGM>** from the main menu to access the configuration menu:

```
█⇨Continuous      ⇨Pulse mode       Setup 1
 ⇨Setup           ⇨Save/Load
```

7. Press **<ENTR>** to access the Continuous mode sub-menu:

cursor positioned to edit Speed range

```
Speed range  99        Jump to max at    6000
Offset       99        Max speed        10000
```

To edit the parameters:

1. Press **<TAB>** to move the cursor to the parameter you wish to change
2. Enter a new value using the number keys on the keypad
3. Press **<ENTR>** to accept the new value or <ESC> to restore the original value
4. Press **<ESC>** to return to the configuration menu.

A graphical description of each of the parameters follows.

3.1.2.1 Speed range

Speed range establishes the slope of the relationship between input device deflection (voltage) and micromanipulator velocity (µsteps/second). It can take any value from 1-99 and is dimensionless.

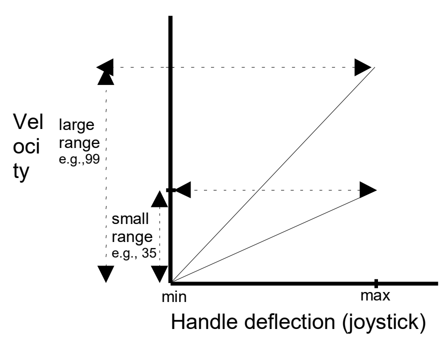


Figure 3-1. Speed range.

The functional effects of an increase in the **speed range** (i.e., slope of the velocity/deflection relationship) are:

- An increase in the range of velocities that can be obtained within the limits of an input device's deflection
- A decrease in the resolution of velocity control

Minimizing the speed range will give you a greater control of the micromanipulator velocity when working under magnification and very near a “target”. You can tailor this parameter to meet the specific requirements of your experimental application.

The **Speed range** can also be set from the **Movement screen** while in the Continuous mode (see **[MOVE\1,2,4 or 5]**).

3.1.2.2 Offset

Offset establishes the Y intercept of the Velocity/deflection relationship. It can take any value from 1-99 and is dimensionless.

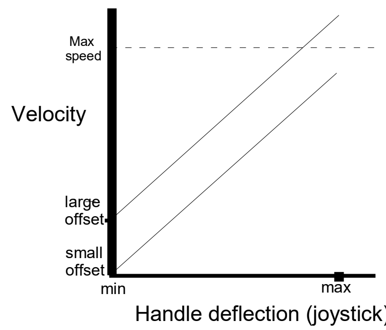


Figure 3-2. Speed offset.

The primary functional effect of increasing **offset** is to increase the initial velocity of the micromanipulator as input device deflection begins. An increase in the offset may also result secondarily in a decrease in the effective **Speed range** by raising the upper end of the range above the level of the **Max speed** (depending on the **Speed range** and **Max Speed** settings; see **Max speed** section). In general, adjustment of the **Offset** parameter allows the **Speed range** to be shifted to velocities that best match the experimental application and personal preference.

The Offset can also be set from the Movement screen while in the Continuous mode (see **[MOVE\1,2,4 or 5]**).

3.1.2.3 Jump to max at...

At the “high speed” end of broad speed ranges are velocities at which motor resonance can occur. This resonance may cause undesirable vibrations in the manipulator.

Jump to Max at... is a “cutoff” point meant to avoid operating the motors at undesirable stepper motor speeds which may cause vibration in the manipulator. Jump to Max at… can take any value from 1 to 9999. Its units are given in µsteps/second and are accurate only at settings < 5000.

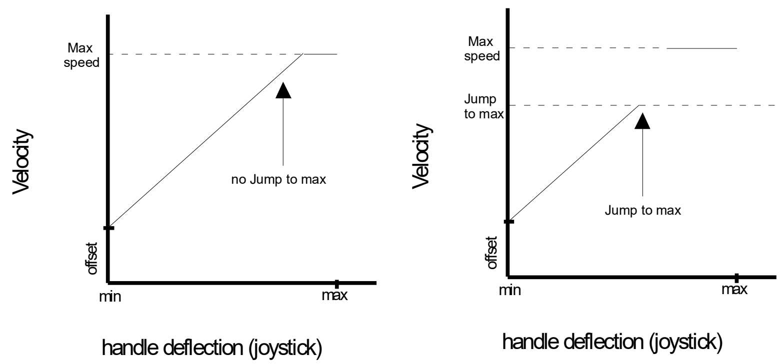


Figure 3-3. Jump to max speed.

The functional effect of the **Jump to max at…** setting is on **Speed range** width. An increase in the **Jump to max at…** value allows a broader **Speed range** before the velocity goes to **Max speed**. However, any advantage gained by the broad **Speed range** may be offset by motor resonance during the speed transition. Decreasing the **Jump to max at…** setting has the functional effect of narrowing the **Speed range**. This may be useful when working under high magnifications when occasional fast movements are still required for making an initial placement or final withdrawal of the micromanipulator.

3.1.2.4 Max speed

The max speed parameter sets the maximal speed. The units of this parameter are µsteps/second. This setting is very accurate below 1000 µsteps/second but becomes less accurate as velocity increases to 10,000 µsteps/second. Above 10,000 µsteps/second the setting can be quite inaccurate.

Max speed, in conjunction with **Jump to max at...**, define the micromanipulator's response at maximum deflections of the joystick.

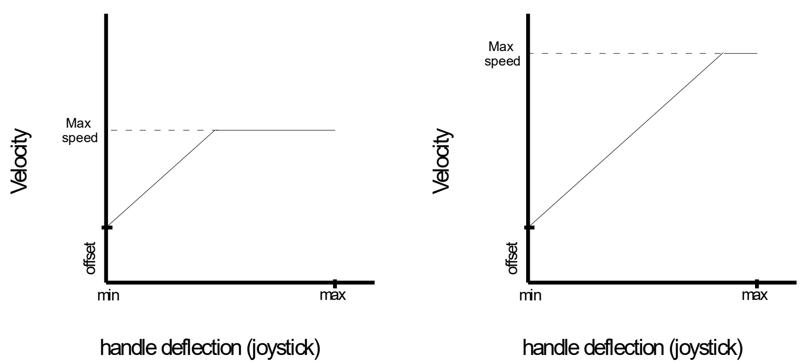


Figure 3-4. Max speed.

The functional effect of decreasing the **Max speed** setting is to lower the upper limit of velocity in response to maximum deflection of the input device and thereby reduce the **Speed range** without affecting the slope of the velocity/deflection relationship or the **Offset**. This is necessary, for example, to prevent excursions that are too fast for a given level of magnification. Increasing the **Max speed** setting has the opposite effect.

3.2 Pulse Mode Sub-Menu [PRGM\Pulse]

With the controller in **PULSE** mode an input command for any axis results in movement of a given length and speed. Using the pulse mode sub-menu, the user has control of both parameters.

Movement mode is selected by switching the “Cont./Pulse” control on the input device if using an ROE. If you are using a Joystick, movement mode is selected by pressing <MOVE> while in the Movement screen (joystick).

Pulse movement is initiated differently for each input device:

- Rotary Optical Encoder -produces 1 pulse per every 1/8th turn of a knob.
- Joystick - produces a pulse for each press of the button on the end of the joystick while the joystick is deflected.

In the Pulse mode sub-menu, you enter length of the pulse and pulse speed in terms of µsteps and µsteps/second respectively. The menu then automatically calculates and displays pulse length and speed in µm and µm/second. Note: the menu calculates and displays two values for each as they are both dependent on the microstep size (course=0.2µm/µstep or fine=0.04µm/µstep)

The number of µsteps/pulse and the number of µsteps per second are adjusted as follows:

1. From the main menu, press <PRGM> to access the configuration menu:

⇨Continuous	⇨Pulse mode	Setup 1
⇨Setup	⇨Save/Load	

2. <TAB > to “Pulse” and press <ENTR >to select the pulse mode submenu. You will see the following screen:

number of µsteps/pulse

20	Steps	4.0	µm	1000	µm/s	@	.2
5000	St/s	0.80	µm	200	µm/s	@	.04

number of µsteps/second

3. Using the numbers on the keypad, enter the desired value for the number of µsteps per PULSE and press <ENTR>
4. If you wish to set the number of µsteps per second, <TAB> to the lower line of the display, enter a new value and press <ENTR>
5. Press <ESC> to return to the configuration menu.

µsteps/pulse and the number of µsteps per second can also be set from within the Movement screen while in the Pulse mode (**[MOVE\1,2,4 or 5]**)

The length of each individual µstep is set by altering the “resolution of movement” setting (coarse or fine) and is adjusted as described in the **Movement Screen** section. This setting is continually displayed on the bottom center of the **Movement screen** (0.2 or 0.04 µm min, respectively).

3.3 Setup Submenu [PRGM\Setup]

The **Setup submenu** is accessed through the configuration menu and allows access to several important parameters, operations, and information screens described in this section. To access the setup submenu:

1. From the main menu, press <PRGM> to access the configuration menu:
2. <TAB > 2X to "Setup" and press <ENTR >to select the pulse mode submenu. You will see the Setup submenu:

▌⇨ Axes	⇨ 4th axis
⇨ Hardware	⇨ Utilities

The remainder of this section is a detailed description of each of the items in the Setup submenu.

3.3.1 Axes [PRGM\Setup\Axes]

Press <ENTR> from the Setup submenu with the cursor on "Axes". You will see the Axes menu:

▌⇨ New	Axes:	◆0◆	◆2◆	◆4◆
origin	LEDs:	Green	Red	Yellow

This menu allows you to establish a New Absolute Origin, the position from which all movements are measured unless you have activated the relative display feature.

From the Axes menu, you may also set micromanipulator axis and motor direction that is assigned to a given channel from an input device.

3.3.1.1 <u>New origin</u>

To establish a New Origin (Absolute Origin):

1. From the movement screen, move the manipulator to the desired location
2. Access the Axes menu using the following sequence of keystrokes:
   Press <ESC> to return to the Main menu
   Press <PRGM> to bring up the configuration menu
   <TAB> 2X to "Setup" and press <ENTR>
   Finally, press <ENTR> to access Axes option.
3. You should be at the Axes menu with the cursor on "New Origin" (see diagram above). To reset the absolute origin to zero press <ENTR>.
4. Press <ENTR> two more times to answer yes to the queries "Reset axis origins now?" and "Are you sure?"
5. Press <ESC> 3X to get back to the main menu.

3.3.1.2 <u>Axis assignment</u>

To assign micromanipulator axis and motor direction for a given input channel:

1. From the Main menu, press <PRGM> to enter the configuration menu
2. <TAB> 2X to "Setup" and press <ENTR>
3. <TAB> to the input channel you wish to configure (i.e.: Green, Yellow, or Red). In the screen shown, the cursor is positioned to configure the Green input channel:

⇨ New	Axes:	◆0◆	◆2◆	◆4◆
origin	LEDs:	Green	Red	Yellow

4. Use the keypad to enter one of the numbers: 0, 1, 2, 3, 4 or 5. The meaning of these numbers is discussed next.

NOTE: The LED colors shown on this screen refer to the input channels of the controller. Each of these channels is hard wired to a motion control on the input device (i.e., knob on the rotary optical encoder or handle movement on the joystick). Please refer to Chapter 2 (INPUT DEVICES) at the beginning of this manual for the detailed correspondence.

Any motor can be assigned to any of the input channels and hence to any input device motion control. The following diagram uses a rotary optical encoder to illustrate the relationship between the three input device motion controls, the three controller input channels, and the three manipulator motors:

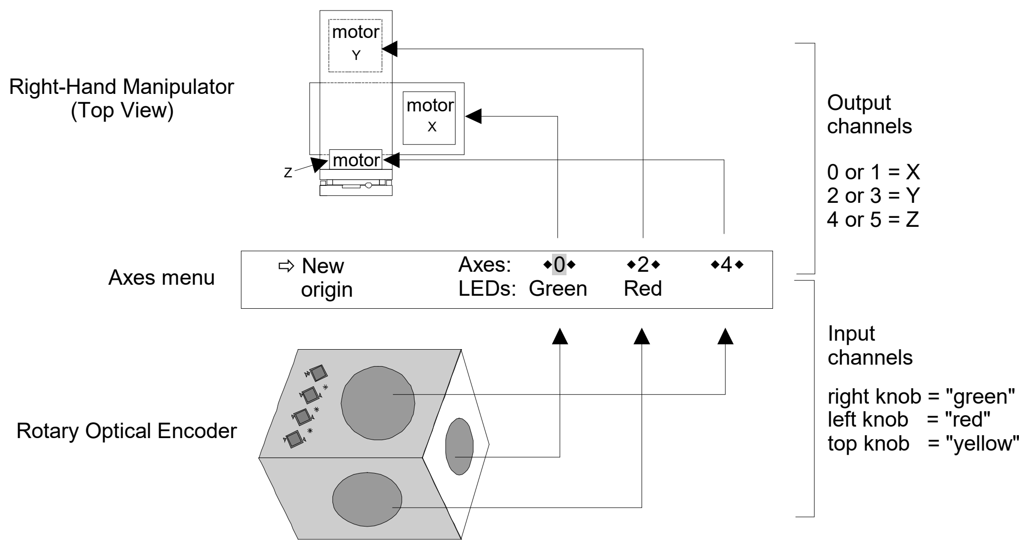


Figure 3-5. Axis assignment map.

Labels have been provided to allow you to mark the Input Device controls after they have been assigned to their respective micromanipulator motors.

Before attempting to change axes assignments it is important to understand the following:

- The numbers (0…5) used in the Axes menu refer to an MP-285 output channel and its polarity (direction of movement). Each pair of these channels is hard-wired to a motor: 0 and 1 = the X motor, 2 and 3 = the Y motor, and 4 and 5 = the Z motor (see Figure 3-6). It is important to remember that these relationships never change.

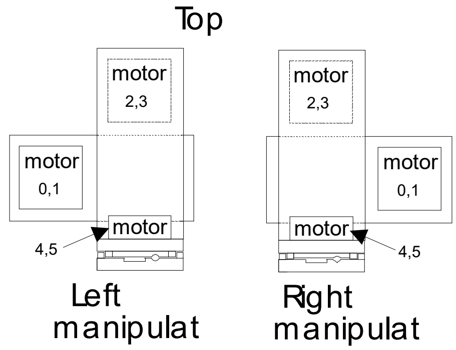


Figure 3-6. Values assigned to each motor.

- The LED colors on the Axes menu refer to the front panel LEDs and thus the input channels of the controller. Each channel is hard-wired to an input device control (a Rotary Optical Encoder knob, for example). This relationship never changes and is documented in Table 2-1.
- In the Axes menu, any output channel (i.e., any motor) can be assigned to any input channel (i.e., input device control).

NOTE: If you assign two or more input channels to a single output channel (motor) an undeclared error state exists. You will receive no error message but only one of the assigned input channels will cause motor movement. The other input channel(s) will have no effect.

To reiterate, you will alter the Axes menu in one of the following ways:

- To assign a specific motor to an input device axis, enter one of the two numbers corresponding to that motor (see Figure 3-5 and Figure 3-6) to the input device control being edited in the Axes menu.
- To reverse the response ("polarity" or "direction of travel") along an axis, alternate between the two numbers in any given pair corresponding to that motor.

These two types of alterations are best demonstrated by the following two examples:

In the factory installed SETUP #1 the "Green", "Red" and "Yellow" input channels are assigned 0,2,4, respectively.

⇨ New	Axes:	◆0◆	◆2◆	◆4◆
origin	LEDs:	Green	Red	Yellow

To reverse the polarity of the response (direction of travel on an axis) of the Green channel ONLY, the Axes menu assignments would be changed from 0,2,4 to 1,2,4 (shown below).

⇨ New	Axes:	◆1◆	◆2◆	◆4◆
origin	LEDs:	Green	Red	Yellow

To interchange the X-Y motor assignment on the input device controls (motor assigned to input control axis) the "X", "Y" and "Z" input device control assignments should be changed from 0,2,4 to 2,0,4 (as relative to factory installed SETUP #1).

⇨ New	Axes:	◆2◆	◆0◆	◆4◆
origin	LEDs:	Green	Red	Yellow

NOTE: The X, Y and Z coordinate labels on the *Movement screen* refer to the output channels of the MP-285 controller. As stated above, each output channel is hardwired to an axis' motor. The motors are then designated X, Y, and Z respectively. This relationship cannot be changed. This hard-wired arrangement may lead to confusion in certain configurations.

In a typical setup, the manipulator is mounted such that the X motor and the X output produce right-to-left movement and the Y motor and output produce front-to-back movement. Right-to-left movement is indicated on the **Movement screen** as changes in the X coordinate and front-to-back movement is indicated as changes in the Y coordinate.

A slightly less conventional setup uses the Right-Angle Adapter (285300) with the manipulator rotated 90 degrees. In this instance, the X and Y motors and their corresponding output channels are interchanged. On the **Movement screen**, the Y coordinate will change when the manipulator moves right-to-left, and the X coordinate will change as the manipulator moves front-to-back. As detailed in the previous section, the controller's interface to the input device can be modified so the desired input control produces movement along the desired axis. What cannot be changed however, is the X and Y coordinates on the Movement screen display.

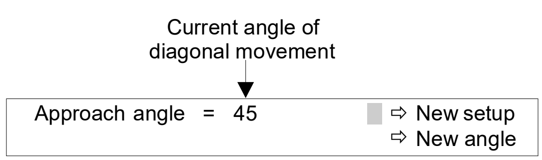
3.3.2 Fourth (4th) Axis Menu [PRGM\Setup\4th axis]

The **4th axis** option in the **Setup menu** allows one to set the angle and plane of movement that is executed when operating in the **Diagonal movement mode** (see **[MOVE\8]**).

Selection of the angle that best matches your needs can be made directly from the **New angle menu** (see **[PRGM\SETUP\4th axis\New Angle]** below) or by using the **New Setup** routine (see **[PRGM\SETUP\4th axis\New Setup]** below). Both functions are accessed from the **4th axis menu**. The default angle for diagonal movement is 45° and the default plane is defined by the X and Z-axes with the X-axis as the default major axis. The angle will remain the same until it is redefined by either of the two methods mentioned above. The plane in which the Diagonal movement will be executed will remain the same until the **New Setup** routine is run. During that routine, the two axes whose displacements are the greatest will be used to define a new plane and the largest movement of the two will define the major axis.

Once defined, the 4th axis is a global setting. The parameters are saved even when the controller is turned off and will not change as new setups are loaded.

To access the 4th axis menu, press <ENTR> from the Setup submenu with the cursor on "4th axis". You will see the 4th axis menu:



Immediately following is a description of the mechanics of changing the 4th axis angle by use of either the "New setup" or the "New angle" routines. Further description of how 4th axis angles and planes are defined and how to set up a 4th axis that is coaxial to a pipette angle is provided at the end of this section.

3.3.3 New Setup [PRGM\SETUP\4th axis\New setup]

When running a new setup for the 4th axis you may find it somewhat easier if you first reset the absolute origin on the controller. Once this is done, you can easily tell the direction of the movements you have made in the setup process. If you wish to reset the absolute origin before proceeding, follow the instructions in section **[PRGM\Setup\Axes\New origin]**. Note that this is optional; you may setup the 4th axis without resetting the origin.

To establish an angle and plane for Diagonal movement:

1. From the main menu, press <PRGM>
2. **<TAB>** 2X to “Setup” and press <ENTR>
3. **<TAB>** to “4th axis” and press <ENTR>
4. You will see the 4th axis menu. Press <ENTR> to accept “New setup”. You will see the following screen:

```
1) Press ENTR to begin.    2) move to start
 position, then press ESC when ready.
```

5. Follow the instructions. Press **<ENTR>** and you will find yourself in a screen identical to the movement screen.
6. Move to the start location of the diagonal that you wish to set up.
7. Press **<ESC>**. You will see the following screen:

```
1) Press ENTR              2) move to end
 position, then press ESC when ready.
```

8. Again, follow these instructions: press **<ENTR>** and you will find yourself in a screen identical to the movement screen.
9. Move to the end location of the diagonal that you wish to set up.
10. Press **<ESC>**. You will see the following screen:

```
   Accept new setup?          ⇨ Accept=ENTR
```

11. Press **<ENTR>** if you wish to accept the changes you have just made. If you wish to abort these changes, press **<ESC>** to return to the previous configuration.

3.3.4 New angle [PRGM\SETUP\4th axis\New angle]

To select the desired angle for Diagonal movement directly from the list of available (preset) angles (without changing the plane of diagonal movement):

1. From the main menu, press **<PRGM>**
2. **<TAB>** 2X to “Setup” and press **<ENTR>**
3. **<TAB>** to “4th axis” and press **<ENTR>**
4. You will see the 4th axis menu. **<TAB>** to “New angle and press **<ENTR>**. You will see the following screen:

```
   Select:   45 39 37 34 31 27 22 18 14 11
```

5. **< TAB>** to the desired angle and press **<ENTR>**. Press **< ESC>** to abort.

3.3.5 How the 4th Axis is defined

The controller uses the coordinates of the two points recorded using “New setup” to select the plane in which diagonal movement will be executed. The two axes in which there is the greatest displacement will define the plane of movement while the axis with the greatest displacement of the two will define the major axis. The controller will select the appropriate angle of motion in that plane.

The "New angle" routine allows you to directly select from 10 preset angles of movement in a previously defined plane. The angle selected establishes the 4th axis relative to the axis that had the greatest displacement ("major axis") the last time New setup was run. This means that the number of possible angles between 0 and 90° is actually 19.

The following diagram illustrates how changing the major axis affects diagonal movement when using the same angle (e.g., 22°):

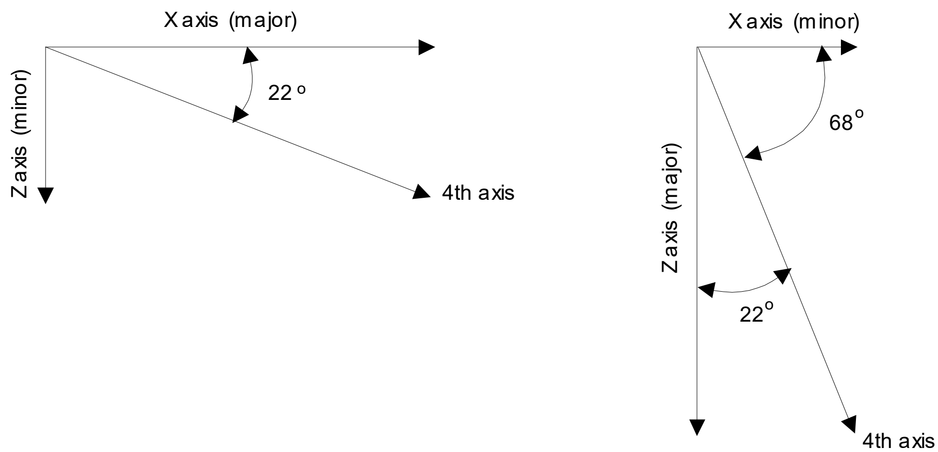


Figure 3-7. Relationship of 4th axis angle to the major axis.

In the left figure, 22° was selected as the angle when the X-axis was previously established as the "major" axis. The 4th axis is 22° below the horizontal. In the right figure, 22° was selected with the Z-axis established as the "major" axis. The 4th axis is set at 22° from the Z-axis and therefore 68° below the horizontal.

Note: The X-axis is the default "major" axis used by the controller any time before a New Setup routine has been run or after 45° has been selected from the New Angle menu. **Anytime you select an angle of 45° you will lose your current major axis.**

3.3.6 How to Setup a Diagonal to Provide Coaxial Movement of a Pipette

Combined use of the indexing ring to adjust the angle of the pipette (see the section on Installation) and the diagonal (4th axis) movement mode (see [PRGM\Setup\4th axis]) makes coaxial movement relatively simple. Keep in mind the angles that are available in the diagonal movement mode (see Fig. 9).

Matching a diagonal movement (4th axis) to a pre-determined headstage angle

1. Make note of the angle on the indexing ring indicated by the pointer once the headstage or pipette holder is set in the desired working position.
2. Set up the appropriate major axis and directions of movement using the New setup routine under the 4th axis menu (see [PRGM\Setup\4th axis\New setup]).

   During this step, do not worry about absolute coaxial movement (i.e., setting the exact angle) It is only necessary to define the correct major axis. The desired angle can be selected from the **New angle menu** in the next step. The following example demonstrates how the correct major axis is determined from the pointer's position:

In the illustration on the left in Figure 3-8, the angle at which the headstage has been positioned by trial is about 30° relative to the horizontal. For this angle and direction of movement, the X-axis should be made the “major axis” during the **New setup** routine. The figure on the right indicates the major axis for other angles and for both right and left-hand mounts.

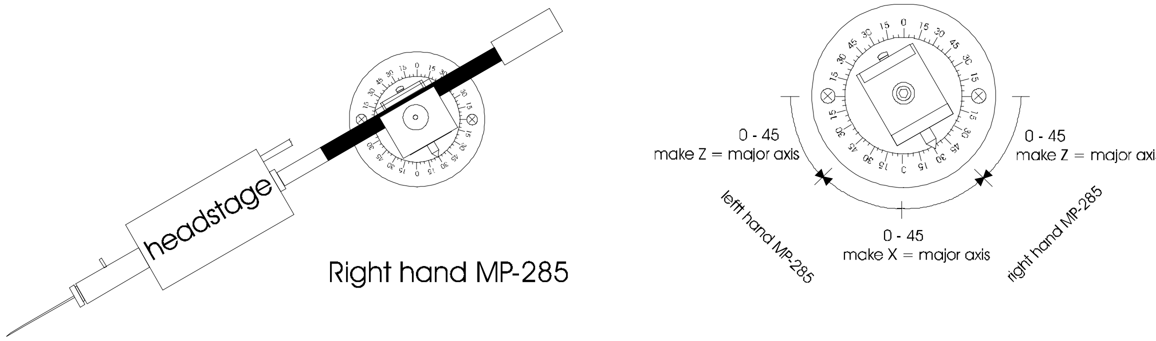


Figure 3-8. Determining the headstage angle.

3. Finally, go to the **New angle menu** and select the value that best approximates the desired angle. In the above example, 31° would be the best choice. Movement in the diagonal mode should now be along a “4th axis” that is approximately coaxial with the pipette.

Examine the pipette tip’s track under the microscope to confirm that its movement is approximately coaxial. It is possible that a satisfactory 4th axis angle is not available on the menu. In that case, you may wish to slightly adjust your angle of approach to match the closest angle available on the New angle menu. See the section on Installation for instructions on setting the rotary dovetail base detent to a specific angle.

3.3.7 Direction of Movement Along the Diagonal

Even with a correct angle of diagonal movement, you may find that the direction of movement along the diagonal is in the opposite polarity from what seems natural for the input you produced on the input device (especially if you are an ROE user).

The direction of movement along the diagonal is controlled by the direction of turn or twist on the ROE or Joystick. The two points: “start position” and “end position” which define the new 4th axis plane and angle also define a direction of diagonal movement. The following indicates the input device manipulation that will produce movement along the diagonal from the start position towards the end position:

- ROE — Counterclockwise turn of the upper knob
- Joystick — Clockwise twist of the stick

In certain configurations, it may be desirable to reverse the direction of movement along the Diagonal. For example, the direction of advance along a diagonal coaxial to a pipette may feel most natural when it occurs in the direction of turn (ROE) or twist (Joystick) that corresponds to an advancing right-hand screw. If the polarity of movement along the diagonal is not in the desired direction, it can be reversed by redoing the **New Setup** procedure and flipping the order of the “start position” and “end position.” Specifically, when the **New Setup** screen prompts you for the start position, move instead to the end position of your diagonal and when you are prompted for the end position move to the start position.

3.4 Hardware [PRGM\Setup\Hardware]

This menu allows you to:

- Set the **Dead zone** parameter.
- Choose the method by which certain features are controlled (**SW1→4**), depending on the input device being used.

3.4.1 Dead zone [PRGM\Setup\Hardware\Dead zone]

The Dead zone is the range of displacement near the neutral position of the joystick handle in which there is no response commanded (see Figure 3-9). The Dead zone has no meaning when using a digital input device such as the Rotary Optical Encoder.

The analog signal from each axis of the joystick is converted into a digital signal via individual A/D channels in the MP-285 controller. A digital value of 512 is produced if the joystick handle is at a neutral position (see [PRGM\Setup\Utilities\AD test]). Displacement of the joystick handle causes a corresponding increase or decrease depending on the direction of the handle displacement.

The Dead zone parameter defines the limits of a null area above and below 512 within which there will be 0 velocity. The value of the Dead zone setting is related to the digitized value of the input device signal. It is without units and can be in the range 0-512.

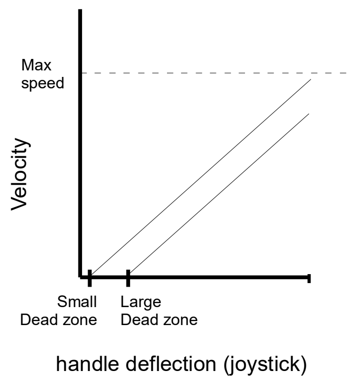


Figure 3-9. Dead zone.

Functionally, the Dead zone parameter sets the sensitivity of the joystick handle and makes control of the onset of movement more manageable. Secondarily, an increase in the Dead zone value decreases the effective range of speed values that can be obtained within the limits of joystick handle deflection.

3.4.1.1 <u>To set the Dead zone value:</u>

1. From the main menu, press <PRGM>
2. <TAB> 2X to “Setup” and press <ENTR>
3. <TAB> 2X to “Hardware” and press <ENTR>. You will see the following screen:

Cursor positioned for editing the Dead zone value

Dead zone 80	SW1 S SW2 S
(0=keypad 1=switch)	SW3 S SW4 S

4. Use the number keys to enter the desired value.
5. Press <ENTR> to accept or <ESC> to abort the edit.

The Dead zone parameter setting is global and remains unchanged when new Setups are loaded. The default Dead zone value is 80. Due to slight fluctuations in neutral position signals from analog input devices, Dead zone settings smaller than 80 are not recommended.

SW1→4 [PRGM\Setup\Hardware]

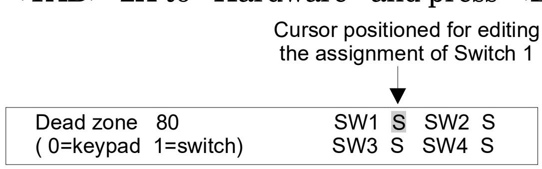
Several functions of the MP-285 controller can be accessed remotely via switches on the ROE. Remote switching is not possible if you are using the Joystick. The functions that can be controlled remotely on the ROE are as follows:

- Changing from Pulse to Continuous Movement mode
- Switching between Coarse to Fine Movement resolution
- Using Diagonal Movement mode
- Sending the unit to the home location

The controller keeps track of which input device is being used and assigns control of the above four functions to the appropriate location: input device switch (S) or keypad (K). These switch assignments can be viewed and edited from the Hardware menu. Manual switch assignment is sometimes necessary after a new type of input device has been connected to the controller.

To view or edit the switch assignments:

1. From the main menu, press <PRGM>
2. <TAB> 2X to "Setup" and press <ENTR>
3. <TAB> 2X to "Hardware" and press <ENTR>. You will see the following screen:



6. <TAB> to the Switch "SW1…SW4" you wish to edit.
7. Press <0> to assign control of that function to the keypad or <1> to assign control to the input device.

Table 3-1 shows the appropriate switch settings for each input device: Joystick and Rotary Optical Encoder (ROE):

Table 3-1. Switch settings.

	Joystick	ROE
Pulse/Continuous Movement mode SW1	K	S
Course/Fine Movement resolution SW2	K	S
Diagonal Movement mode SW3	Blank	S
Go home SW4	Blank	S

3.5 Utilities [PRGM\Setup\Utilities]

The Utilities menu is extensive. It contains sub-menus for control of seven different functions. Only three of these, “Info”, “Reset” and “Center axes” are of general use. The functions “Baud” and “SIO test” are of use to those who control the MP-285 with a computer over the serial interface. “AD test” is used by, and only appears for, those who have an analog input device (Joystick). Finally, “Pause” is used only by those who have the optional TTL input for Robotic movement triggering.

To access the Utilities functions, press <ENTR> from the Setup submenu with the cursor on “Utilities”. You will see the following menu:

```
⇨Baud    ⇨Info        ⇨SIO test        ⇨Reset
⇨Center axes     ⇨AD test      ⇨Pause
```

The remainder of this section describes each of the Utilities functions in the order that they appear in the Utilities screen shown above.

3.5.1 Baud [PRGM\Setup\Utilities\Baud]

To set the Baud rate and/or the Parity of the RS-232 serial port follow these steps:

1. From the main menu press <PRGM>
2. <TAB> 2X to “Setup” and press <ENTR> to accept
3. <TAB> 3X to “Utilities” and press <ENTR> to accept
4. Press <ENTR> to accept “BAUD” The following screen will appear:

```
Current Baud rate:         9600
1200  2400  4800  9600  19200               Parity Off
```

Cursor positioned for
selecting 1200 as the Baud rate

5. To change the Baud rate <TAB> to the desired setting and press <ENTR>
6. To change the Parity, <TAB> to the “Parity” option and press <ENTR> to toggle the parity on and off.
7. A confirmation prompt will appear above the Parity option:

Cursor positioned for
confirmation of the Parity change

```
Current Baud rate:          9600              PARITY ON?
1200  2400  4800  9600  19200                 Parity Off
```

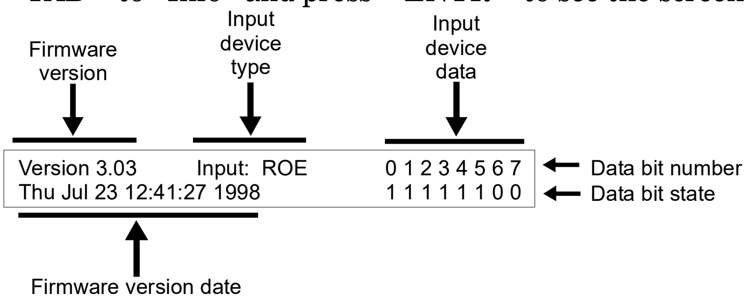
8. Press <ENTR> to accept the change or <ESC> to abort the change.

3.5.2 Info [PRGM\Setup\Utilities\Info]

The **Info** option on the **Utilities menu** displays information about the version of the firmware that is installed, the type of input device that is presently connected to the controller and the state of the serial data bits that convey input device switch positions to the controller.

To access the **Info screen:**

1. From the main menu press <PRGM>
2. <TAB> 2X to "Setup" and press <ENTR> to accept
3. <TAB> 3X to "Utilities" and press <ENTR> to accept
4. <TAB> to "Info" and press <ENTR> to see the screen:



When calling Sutter's technical support staff, it is often useful to know the version and date of the firmware installed in your controller.

The other two information fields displayed in the Info screen are useful when troubleshooting Input device problems; the Input device type should be correctly identified, and the Input device data display should indicate appropriate bit state changes as all input device switches (and analog input device controls) are exercised. Table 3-1 shows the input device switch (or control) associated with each data bit number.

Table 3-1. Info screen data bit meanings.

Input device type	Data Bit Number							
	0	1	2		3	4	5	
Joystick	handle up	handle left	handle ccw	right button			left button	handle button
ROE	home switch		coarse/fine switch				pulse/cont switch	diagonal switch

3.5.3 SIO test [PRGM\Setup\Utilities\SIO test]

The SIO or serial input/output test allows you to easily check the continuity and functionality of a serial link between your MP-285 and a computer. This function is quite useful when initially setting up for computer control of your manipulator. It allows you to send serial information in both directions over the RS232 interface.

The SIO test is best run while your MP-285 controller is connected to a serial port on your computer (i.e., COM1 or COM2). Typically, this will be the port that you intend to use to control your manipulator via a software program you write. SIO test works well in conjunction with a terminal emulator program such as Microsoft's HyperTerminal.

To use the SIO test:

1. Connect one of your computer's serial port(s) to the "Serial Port" on the back of the MP-285 controller using the supplied DB9 cable. Some computers have a serial port that use a DB25 connector. In this event, you may need a DB25 to DB9 converter (not included).
2. Configure your terminal emulator program for the following: TTY mode, 1200 Baud (bites per second), 8 data Bits, Parity Off (None), Stop bits 1, Flow control none and ECHO off. You will also need to tell the terminal emulator to communicate with the COM port you connected to the MP-285.
3. Turn on the MP-285. Using the Baud function shown previously, [PRGM\Setup\Utilities\Baud], set the Baud rate of the MP-285 controller to 1200 and make sure the Parity is set to off.
4. Press <ESC> from the Baud screen to return to the Utilities menu
5. <TAB> 2X to "SIO test" and press <ENTR> to select. You will see the following screen:

```
⇨ASCII      ⇨Jabber      ⇨Decimal
```

The functions "ASCII" and "Decimal" monitor serial input from your computer. The function "Jabber" produces a serial output string to your computer. All three work best at slower Baud rates (1200BPS). This does not mean that the controller cannot operate during normal communications at higher rates.

If your computer is on and a connection has been established, you can check the serial connection in one of three ways:

1. Press <ENTR> to access the "ASCII" function and press <ENTR> again to start the test. As you type characters on your computer keyboard, they should show up on the MP-285 display and also be echoed back to the terminal emulator window and displayed on the computer screen. Press any MP-285 keypad key to terminate the test
2. <TAB> to "Jabber" and press <ENTR>. Press <ENTR> again to start the test. You will see a continuous string of lowercase letters. This string should be produced on the window of your terminal emulator. Press any MP-285 keypad key to terminate the test.
3. <TAB> 2X to "Decimal" and press <ENTR>. Press <ENTR> two more times to bypass the warning and to start the test. This time, the controller prints out the three number ASCII code for each character you type at the terminal emulator. There is no echo back to the terminal so you will not see any characters on the terminal window.

If you are unable to establish contact with your computer over the serial link using the SIO test and a terminal emulator, you will need to correct this problem before attempting computer control. If you are having difficulties, check that you are properly connected to the COM port on the computer and have designated communications to the correct COM port in the terminal emulator program. Also, verify that you have correctly configured the serial port on the MP-285 controller.

If you have successfully established contact between your computer and MP-285 controller, you may wish to run the MP-285 PC Controller Program to get a feeling for the types of functions that can be controlled via the serial link. (see section “Computer Interface” later in this manual).

3.5.4 Reset [PRGM\Setup\Utilities\Reset]

This option allows all the **SETUP** memory slots to be restored to their original values, as they were set when the controller was shipped. **This operation will restore SETUPS (1 through 5) AND DELETE ANY SETUPS STORED IN MEMORY SLOTS 6 through 0.** If there are any custom SETUPS that you wish to restore after performing this operation, you will have to manually re-enter the parameter settings for those **SETUPS**.

The Reset operation will also erase any user entered robotic routines and restore the factory defaults.

Reset all values also reconfigures the following global parameters to their factory defaults:

- Current micromanipulator position is set as the absolute zero
- The **4th axis** diagonal is set to a 45° angle in the X-Z plane
- The RS-232 interface is set to a baud rate of 9600, Parity Off
- **Dead zone** is set to 80
- **Pause** is set to ignore TTL input

The entering of robotic routines and global parameters are described elsewhere in this manual.

To perform a “Reset”:

1. From the main menu press <PRGM>
2. <TAB> 2X to “Setup” and press <ENTR> to accept
3. <TAB> 3X to “Utilities” and press <ENTR> to accept
4. <TAB> 3X to “Reset” and press <ENTR> to see the screen:

```
WARNING.  ALL PROGRAMS AND SETUP VALUES
WILL BE RESET TO FACTORY  VALUES.  ENTR=OK
```

5. Press <ENTR> to reset all values or <ESC> to abort. You will receive one more confirmation prompt to which you must again either press <ENTR> or <ESC>.

3.5.5 Center axes [PRGM\Setup\Utilities\Center axes]

“Center axes” is a self-running routine that performs a task that you may need from time to time. First, the routine instructs the manipulator to move to the end of travel on each axis and records the coordinate of these points. Based on this information, the controller then calculates the midpoint of each axis and moves each axis to that location. Finally, with the manipulator still in central position, the controller does a reset of the absolute zero and thus gives the center axes location the absolute coordinate: 0,0,0.

Before you run the center axes routine, you must make sure that there is nothing impeding the full travel along each of the three axes. If anything inhibits movement, the center axes routine will give an incorrect result and may damage either your manipulator or some other part of your setup.

To run “Center axes” do the following:

1. From the main menu press <PRGM>
2. <TAB> 2X to “Setup” and press <ENTR> to accept
3. <TAB> 3X to “Utilities” and press <ENTR> to accept
4. <TAB> 4X to “Center axes” and press <ENTR>
5. Press **<ENTR>** to continue or **<ESC>** to abort. You will see the following screen:

```
Be patient.  This will take some time.
```

You will be returned to the Utilities menu when the routine is finished. If you then <ESC> back to the main menu and press <MOVE> you will find that you are at location 0,0,0 and if you look at the manipulator you will see that it is positioned at the center of travel for each axis.

3.5.6 AD test [PRGM\Setup\Utilities\AD test]

(Joystick only)

This option on the **Utilities** menu is useful for adjusting analog input devices (see APPENDIX B)

To view the **AD test** display:

1. From the main menu press <PRGM>
2. <TAB> 2X to “Setup” and press <ENTR> to accept
3. <TAB> 3X to “Utilities” and press <ENTR> to accept
4. <TAB> 5X to “AD test” and press <ENTR>. You will see the following screen:

Calc0	0	Calc1	0	Calc2	0	◄— Period of output signal to motor (µsec X 10)
Raw0	512	Raw1	512	Raw2	512	◄— Digitized input device signal (in neutral position)

The three values displayed on the bottom line are the digitized inputs from the analog input device. The value can range from 0 to 1023, as a function of direction and magnitude of input.

When the joystick is at its neutral position (no movement is being commanded), the value should be near 512. However, there is an inherent instability to this signal and fluctuations of ±20 are not unusual (see APPENDIX B for instructions on adjusting this neutral position value). The Dead zone parameter setting establishes the thresholds above and below the neutral position value (512) beyond which the signal must increase or decrease before an output to the motor will be initiated (see **[PRGM\Setup\Hardware\Dead zone]**).

The values displayed on the upper line represent the period (in µsec X 10) with which discrete signals are being sent to the micromanipulator motor to effect µstep movements. The further the joystick handle is deflected, the smaller the period value will become. This faster frequency translates to a higher motor velocity.

The specific relationship between the input values (bottom line) and the period length (top line) is shaped by the parameter settings that adjust the response characteristics for analog input devices (**Speed range, Offset, Max speed and Jump to max at...**). Changes in those parameters will result in changes in the pattern of output values that are consistent with the "Velocity/Input device deflection" graphs shown in the section on Controller Configuration (see **[PRGM\Continuous]**).

3.5.7 Pause [PRGM\Setup\Utilities\Pause]

This function is only useful if one has a controller modified to trigger robotic routines from an external TTL pulse. In such a unit, a transition from a low to high logic level can be used to release a robotic routine from an inserted pause (See the Robotics section for information on inserting a pause in a robotic routine).

In the standard controller, a pause in a robotic routine is ended by a key press on the MP-285 controller keypad. In a TTL triggerable controller, the pause can be ended by either a key press or the external logic signal. The "Pause" utility configures the controller to look for the TTL pulse and allows one to insert a delay between the TTL trigger and the release from the robotic pause.

To configure the Pause function:

1. From the main menu press <PRGM>
2. <TAB> 2X to "Setup" and press <ENTR> to accept
3. <TAB> 3X to "Utilities" and press <ENTR> to accept
4. <TAB> 6X to "Pause" and press <ENTR>. You will see the following screen:

```
Enter delay after pause in ms        0
0 disables external pause / delay / continue
```

6. Use the number keys to enter the desired delay in milliseconds. If you enter a delay of <0>, the controller will ignore the external trigger signal.

When setting up the delay function, be aware that the controller has a built-in delay of about 150ms. You cannot move faster than this after the logic level transition. If you wish to move after a delay greater than 150ms, the number you enter in the screen should be your desired delay less 150ms.

3.5.8 Save/Load menu [PRGM\Save/Load]

Using its **Save/Load** capacity, the MP-285 controller can store up to 10 sets of operating parameters or **SETUPS** that tailor the manipulator's response to personal preference and experimental demands. These parameters are discussed elsewhere (**[MOVE]**, **[PRGM\Continuous]**, **[PRGM\Pulse]** and **[PRGM\Setup]**). From the **Save/Load menu** you can also **clear** the settings from a memory slot or choose to **view** the values before loading (or clearing) a **SETUP**.

The following is a list of the parameters for which settings are saved in each **SETUP** memory slot (**SETUP**-specific parameters). Note that most of these parameters relate only to certain movement modes and/or input devices.

Analog input devices:
- Speed range (for Continuous mode using a Joystick)
- Offset (for Continuous mode using a Joystick)
- Jump to max at...(for Continuous mode using a Joystick)
- Max speed (for Continuous mode using a Joystick)

Analog & digital input devices:
- msteps/pulse (for Pulse mode)
- msteps/second (for Pulse mode)

Movement mode (Continuous/Pulse)
Movement resolution (Course/Fine or 0.2/0.04mm/step)

Digital input device (continuous mode):
- msteps/click (for Continuous mode using an ROE)

Axes assignments

There are five default SETUPS loaded in memory slots 1 through 5 at the factory. The default values in Setups 1 through 5 are listed below. **SETUPS** can be loaded by using the **[*]** key, by using the keys numbered 1 through 5 while in the **Main menu** (See Setup Selection). Alternatively, you can load using the Save/Load menu.

3.5.8.1 <u>To access the Save/Load menu:</u>

1. From the main menu, press <PRGM> to access the configuration menu:

```
█⇨Continuous      ⇨Pulse mode      Setup 1
 ⇨Setup           ⇨Save/Load
```

2. <TAB > 3X to “Save/Load” and press <ENTR> to select the Save submenu. You will see the Save/Load submenu:

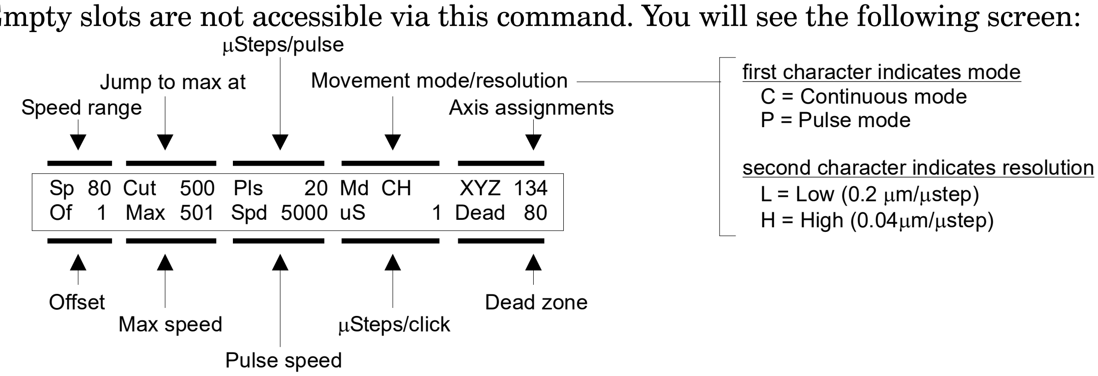
```
Setup        █⇨ Save/Clear   1 2 3 4 5 6 7 8 9 0
is now 1      ⇨ View/Load    * * * * *
```

The remainder of this section describes how to use “Save/Clear” and “View/Load” followed by a table that gives the default values of the factory installed setups #1 through #5.

3.6 View/Load [PRGM\Save/Load\View/Load]

To View and/or Load the settings saved in a memory slot (e.g., #5):

1. From the main menu, press <PRGM>
2. <TAB> 3X to “Save/Load” and press <ENTR>
3. <TAB> to “View/Load” and press <ENTR>
4. <TAB> to the number of the saved setup you wish to View and/or Load and press <ENTR>. Note that there is an asterisk only under Setup slots that contain Setups. Empty slots are not accessible via this command. You will see the following screen:



5. At this point you have two options:

To Load the Setup being viewed, press <ENTR>. You will see a warning screen to tell you that you are about to overwrite the Setup that is currently loaded. If the current Setup is not saved in another memory slot, or you are not sure you may wish to press <ESC> and save it before loading the new setup (see “Save/Clear” below for instructions). If you do not need to save the current setup, simply press <ENTR>.

Alternatively, to View the next Setup, without loading the setup you are viewing, press <ESC> followed by <ENTR>. If you are already viewing the last stored Setup, you will be bounced to the “Save/Clear” function.

3.7 Save/Clear [PRGM\Save/Load\Save/Clear]

While using the micromanipulator, you may have altered the settings on one or more of the parameters in the initially loaded SETUP. **First, recognize that until you need to load a different SETUP you do not need to take any save action**. The altered SETUP will remain in non-volatile memory (the controller can be turned on and off)

When and if you find you need to load a different SETUP, you have three options:

- Discard the altered SETUP

  or

- Save the altered SETUP in the same memory slot (after clearing the old version of the SETUP)

  or

- Save the altered SETUP in a different memory slot (retaining the old version of the SETUP in its original memory slot).

To **discard** the altered Setup is straightforward. Anytime you attempt to Load a new Setup over a Setup that has be altered from its previous configuration, you will be warned that the current setup is either “not saved” or “about to be overwritten”. If you wish to discard your altered setup, simply ignore the warning.

To Save the altered Setup in **the same** memory slot, follow these instructions:

1. From the main menu, press <PRGM>
2. <TAB> 3X to “Save/Load” and press <ENTR>
3. Press <ENTR> to accept “Save/Clear”
4. The cursor will be on the Setup slot that was altered. To overwrite the original copy press <ENTR>.
5. You will be prompted with the question “ERASE #?” where # is the occupied slot. Press <ENTR> to erase and <ESC> to abort the save at that location. This operation performs a “Clear” of the memory slot. **You have not yet saved the altered setup!**
6. To save the altered Setup in this slot, press <ENTR> 2X more.

Note that the “**Clear**” can be executed by stopping at step 5 if you wish to purge an undesired Setup.

To Save the altered Setup in a **different** memory slot, follow these instructions:

1. From the main menu, press <PRGM>

2. <TAB> 3X to "Save/Load" and press <ENTR>
3. Press <ENTR> to accept "Save/Clear"
4. The cursor will be on the Setup slot that was altered. <TAB> to a free memory slot (no asterisk) and press <ENTR>

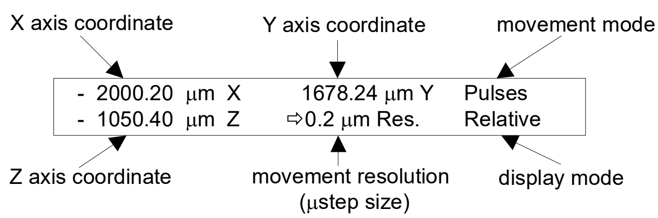
4. MOVEMENT SCREEN [MOVE]

The MP-285 is driven by stepper motors whose inherent electromechanical design provides outstanding resolution and dynamic range of movement while remaining virtually driftless at rest. The minimal unit of distance that the MP-285 motors can travel is referred to as a "microstep" (µstep) . All movements greater than the distance of a single µstep are the result of the execution of a series of µsteps. Functionally, a short µstep length provides the smoothest, slowest, movement while longer µsteps provide faster, less smooth movement. The firmware installed in the MP-285 controller lets you select between two preset µstep lengths: 0.04 µm, a "fine" resolution movement, useful while working under high magnification, and 0.2 µm, a "coarse" resolution movement for rapid initial positioning of the micromanipulator. Selection of these settings is described below (see **[MOVE\3]** and **[MOVE\6]**).

Turn on the **POWER** switch on the front panel of the MP-285 controller. As the controller boots up, it will briefly display a start-up screen giving model and manufacturer information. Immediately after that, the **MAIN MENU** will appear:

```
S⇨1          * * * MP-285   Controller * * *        P⇨ 1
Press MOVE,     PRGM       or select mag. with  *
```

To initiate any movement using the input device, you must first press < **MOVE>** on the keypad of the MP-285 (refer to the **Movement Screen** branch of the enclosed menu tree). The **Movement screen** displays the manipulator's position in µm:



The X, Y and Z on this screen ALWAYS refer to the same manipulator motor (axis) regardless of the motor assignment to Input device control.

The **Movement Screen** also gives the following information:

- The movement resolution or minimum distance of each microstep (0.2 or 0.04 µm Res.).
- The mode by which the micromanipulator will move: Continuous movement of variablecontinuous mode velocity (top right corner of the screen blank) or Pulse movement of constant velocity ("Pulses" in the top right corner of the screen).
- An indication of whether the X, Y and Z coordinates displayed on the Movement Screen are measured with respect to either an ABSOLUTE Origin (lower right corner of the screen blank) or a RELATIVE Origin ("Relative" in the lover right corner).
- An indication of whether the manipulator is in Diagonal movement mode (the phrase "µm Res." will be replaced by the word "Diagonal")

Once <MOVE> has been selected from the **MAIN MENU** and the **Movement screen** is displayed, you can use the input device to move the micromanipulator. If there is no change in the coordinate as you use the input device, make the following checks:

1. Make certain that all the cables are properly connected.
2. Press <ESC> to return to the MAIN MENU and then go back to the Movement screen by pressing <MOVE> again.
3. Turn the controller off and on.

The following section gives a description of the keypad keys and/or input device buttons that are active when the Movement screen is displayed. In addition to the keys described here, the <*> key provides access to a broad range of Robotic functions. These functions are described in the section of the Reference manual entitled "Robotic Movements".

4.1 Course/Fine Movement Resolution [MOVE\3] and [MOVE\6]

To set the movement resolution to 0.2 µm/µstep (coarse) while the **Movement screen** is displayed press <**3**> on the keypad or when using a Rotary Optical Encoder press the **coarse/fine** button on the Input device (up and unlit = coarse resolution).

To set the movement resolution to 0.04 µm/µstep (fine) while the **Movement screen** is displayed press <**6**> on the keypad or when using a Rotary Optical Encoder press the **coarse/fine** button on the Input device (down and lit = fine resolution).

Any time fine control of the micromanipulator is required (for example anytime you are working under a microscope) you should use the **fine** movement resolution (0.04 µm/µstep) setting.

4.2 Continuous/Pulse Movement Modes [MOVE\MOVE]

To toggle between continuous and pulse modes of movement while the **Movement Screen** is displayed use the switch on the ROE. If you are using a joystick, press <MOVE> on the keypad while in the movement screen.

4.3 Input Device Response Adjustment [MOVE\1,2,4 or 5]

The MP-285 controller allows you to adjust several parameters that alter the responsiveness of the micromanipulator to commands from the input device. Several of these can be rapidly adjusted while in the Movement screen. The following is a brief definition of each of these parameters and a description of their adjustment. This section is organized by movement mode (CONTINUOUS vs. PULSE). Under Continuous, organization is by input device.

4.3.1 Adjustment of Parameters affecting PULSE Movement

When the controller is set to **PULSE** movement mode (see above), a single input command will result in a single pulse of programmed length and speed. If you are in the Movement screen, you can change both these parameters:

1. Make sure your controller is in Pulse movement mode. If this is true, you will see the word "Pulses" displayed in the upper right-hand corner of the movement screen. If you are not in Pulse mode, follow the instructions in section "Continuous/Pulse Movement Modes" to switch to Pulse Mode
2. From the Movement screen press either the <1>, <2>, <4> or <5> key. You will see the following screen:

number of µsteps/pulse

↓

20 Steps	4.0	µm	1000	µm/s	@.2
5000 St/s	0.80	µm	200	µm/s	@.04

3. On the left, the screen gives the pulse length and speed in µSteps and µSteps/s, respectively (note the screen omits the µ here). On the right, the screen calculates the true pulse length and speed (in µm and µm/s for each µstep size (.2 and .04µm).
4. The pulse length and speed are adjusted using the <1>, <2>, <4> and <5> keys. <1> and <4> function as up and down arrows to increase and decrease pulse length while <2> and <5> act to increase and decrease pulse speed. True pulse length and speed are constantly updated on the right half of the screen.
5. After you have adjusted, press <ENTR> to return to the movement screen.

Pulse length and speed can also be set from within the Configuration Menu (see section on **PRGM\Pulse).**

4.3.2 Adjustment of Parameters effecting CONTINUOUS Movement (ROE users)

When in **CONTINUOUS** movement mode, input commands from the ROE yield a continual movement. The velocity of the movement is directly related to the velocity of the knob. Movement of the ROE knob is "encoded" as digital pulses or **"clicks** (512 **clicks** per full turn of the ROE knob). Each **click** commands a defined number of µsteps (a definable value). The micromanipulator's response to turning an ROE knob depends on two parameters: the number of **µsteps/click** and the length of each individual µstep.

To change the number of **µsteps/click while in the movement screen:**

1. Make sure your controller is in Continuous movement mode. If this is true, you will **not** see the word "Pulses" displayed in the upper right-hand corner of the movement screen. If you are not in Continuous mode, follow the instructions in section Continuous/Pulse movement modes to switch to Continuous mode.
2. From the Movement screen press either the <1>, <2>, <4> or <5> key. You will see the following screen:

Number of µsteps/click

↓

7	=	1.4	µm/Click	@	0.2
Steps/click	=	0.28	µm/Click	@	0.04

3. On the left, the screen gives the number of µsteps/click (Note: the screen omits the µ here). On the right, the length/click (in µm/click) for each µstep size (.2 and .04µm) is calculated.
4. Press <1> on the keypad to increase the µsteps/click or <4> to decrease. The calculated values update instantaneously
5. Press <ENTR> to return to the movement screen.

The length of each individual µstep can be set by altering the "resolution of movement" setting (coarse or fine) and is adjusted as detailed in section Course/Fine **Movement Resolution**.

µsteps/click can also be set from within the Configuration Menu (see section on **PRGM\Continuous).**

It is recommended that settings ≤10 be used. Increasing the number of **µsteps/click** results in faster movement but there are practical limitations on this parameter. For example, at very high **µsteps/click** settings (>20) it is possible to move the knob much faster than the micromanipulator can respond. This results in input commands being ignored or skipped. Furthermore, the control of the velocity of those movements can be very poor as even the slowest of knob rotations produce near maximal velocities.

Further refinement of the sensitivity of the ROE knobs. For some experimental applications the Rotary Optical Encoder control knobs may still seem too "sensitive" even at the minimal movement resolution settings (**µsteps/click = 1 and µm/µstep = 0.04**). It is possible to achieve continuous-like movement with a less "sensitive" feel to the knob by using the pulse movement mode. The basic idea is to distribute the execution of a few (e.g., 5) µsteps in a Pulse movement over a relatively long period of time (e.g., 0.5 sec). In this way, a relatively large knob movement at a very slow rate (e.g., 1/8 of a turn every 0.5 sec) will command a relatively small number of microsteps. This should result in a less sensitive "feel" to the knob when very slow movements are required.

The settings in the menu shown below are a good starting point for trying this method. The settings can be made by using the **[PRGM\Setup\Pulse mode]** commands to access the **Pulse mode menu**. Then you can further refine the values within the **Movement screen** (when the Pulse mode has been selected) by pressing the **1,2,4 or 5 keys**.

number of µsteps/pulse

5 Steps	1.0 µm	2 µm/s	@.2
10 St/s	0.20 µm	0 µm/s	@.04

number of µsteps/second

Using this approach, the minimal increment of movement is calculated for the two resolutions and for the settings shown above is 0.2 µm (at fine resolution of 0.04 µm/µstep). It will take 0.5 seconds to execute each pulse (at the rate of 10µsteps/second) and a new pulse will be executed every 1/8 revolution of the ROE knob.

If the knob is turned slowly, to allow execution of each pulse, the total distance moved per full turn of the knob would be 1.6 µm (8 X 0.2 µm). For greater or less knob sensitivity, simply increase or decrease the # µsteps/pulse and proportionally increase or decrease the velocity (µsteps/second).

4.4 Adjustment of Parameters effecting CONTINUOUS Movement (Joystick users)

In **CONTINUOUS** movement mode, input commands yield a continuous movement. The velocity of movement depends on the amount of displacement of the joystick handle. An A/D interface in the MP-285 controller converts analog command signals from the input device into a digital output to the manipulator motors. The firmware installed in the MP-285 controller allows you to adjust four parameters related to this A/D conversion of the input signal. For a full discussion of the adjustable parameters and how they affect the A/D conversion see the section in the Controller Configuration section of this manual.

Two of the four parameters can be adjusted from within the movement screen. They are "Speed Range and "Offset". **Speed Range** establishes the range of speeds that can be obtained within the limits of the Force Sensitive Resistor or joystick displacement. The maximum (broadest) range is assigned 99 and the minimum (narrowest) is 1. **Offset** is the minimal speed at which the movement will start. The maximum **Offset** value is 99 and the minimum is 1.

To change **Speed range** and **Offset while in the movement screen:**

1. Make sure your controller is in Continuous movement mode. If this is true, you will **not** see the word "Pulses" displayed in the upper righthand corner of the movement screen. If you are not in Continuous mode, follow the instructions in section Continuous/Pulse movement modes to switch to Continuous mode.
2. From the Movement screen press either the <1>, <2>, <4> or <5> key. You will see the following screen which gives the values for both Speed range and Offset:

To change	Speed range=99	Offset=99
press keys ▶	1 or 4	2 or 5

3. The parameters are adjusted using the <1>, <2>, <4> and <5> keys. <1> and <4> function as up and down arrows to increase and decrease Speed range while <2> and <5> act to increase and decrease Offset.
4. After you have adjusted, press <ENTR> to return to the movement screen.

Speed Range and **Offset** can also be adjusted using the **[PRGM\Continuous]** commands to access the **Continuous mode menu**. Two other parameters that affect the input device response: **Cutoff** and **Max speed** can only be adjusted using **PRGM\Continuous**. These commands are covered in the Controller Configuration section of the manual where a graphical representation of how the four parameters relate to manipulator movement can be found.

4.5 "4th axis" or Diagonal movement [MOVE\8]

The MP-285's Diagonal mode makes precise movement of a micromanipulator along a diagonal path (or "4th axis") by simultaneously activating two axes. In **DIAGONAL** mode the controller automatically moves the micromanipulator along a predetermined "4th axis" by appropriate simultaneous commands to two stepper motors. The 4th axis is specified using the **New setup** routine (**[PRGM\Setup\4th axis\New setup]**). This procedure is covered in the Controller Configuration section of the manual. **DIAGONAL** movements can be executed while in **CONTINUOUS** or **PULSE** modes and in either course or fine resolution.

To toggle the DIAGONAL mode on and off, press **<8>** on the keypad while in the **Movement screen** (or use the switch on the ROE). The word "Diagonal" will appear in the bottom/middle position of the **Movement screen** when this function is active, replacing "µm Res.":

- 2000.20 µm X	1678.24 µm Y	Pulses
- 1050.40 µm Z	⇨0.2 Diagonal	Relative

▲ 4th axis ON

To execute movement in the DIAGONAL mode use the control on the input device which is typically configured for the Z-axis:

- Rotary Optical Encoder — turn the upper knob
- Joystick — twist the stick

4.6 Setting Relative origin [MOVE\0] and Relative Display mode [MOVE\ENTR]

The MP-285 position is measured with respect to a point called the **ABSOLUTE Origin**. Before shipment, the ABSOLUTE Origin is set at the point where all three micromanipulator axes are at the middle of their range of motion. If desired, you can establish a different ABSOLUTE Origin (see section on **PRGM\Setup\Axes\New origin**).

The MP-285 controller can also measure distance with respect to a second or **RELATIVE Origin**. This feature allows for convenient measurement of the distance between two points on a specimen <u>without resetting the ABSOLUTE Origin</u>. To use this feature, simply set a RELATIVE Origin while in the movement screen.

At any time, you can revert to the ABSOLUTE Display mode in which the position coordinates will once again indicate the micromanipulator's distance from the ABSOLUTE Origin.

To set the **RELATIVE Origin**

1. Press <0> on the keypad at any time while the **Movement screen** is displayed
2. Press <ENTR> when prompted. This position will remain in memory as the Relative origin until the **<0><ENTR>** key sequence is repeated. When you press <0><ENTR> the display will also switch to Relative Display mode:

-	0.00 µm X	0.00 µm Y	Pulses
-	0.00 µm Z	⇨0.2 Diagonal	Relative

Relative Display Mode is ON

3. To toggle back to Absolute mode and/or to turn on Relative mode without resetting the Relative origin, simply press <ENTR> while in the Movement screen. Coordinates are automatically changed to the appropriate values.

5. ROBOTIC MOVEMENTS [MOVE*]

5.1 General Information

The MP-285 micromanipulator can make two classes of preprogrammed or robotic movements.

The first class, Movement to a "Home" location, is highly structured. To perform a home movement the user first designates a home location. At any point thereafter, if a home movement is initiated the controller commands the manipulator to move via the shortest possible route. While the speed of the movement can be controlled, there is no control over the path taken. This function is often used to repeatedly return a micropipette tip to some "rally" point (i.e., the HOME position) from other locations.

The second class of robotic movement is much more general. Using the Learn function, the controller "records" a path of movement that may contain up to 55 separate moves. After recording, the series of movements can be executed once, looped, or reversed. Furthermore, the moves can be executed in the absolute coordinate system used by the controller or can be executed as relative moves with respect to the location of the point where the path was started. Up to 10 individual programmed robotic series can be saved and recalled.

The MP-285 is an "open loop" system. There is no feedback of position information between the manipulator and controller except at the end of travel (EOT). The controller calculates the position of the manipulator based on the commands that it has sent. Even with this limitation, you will find that the MP-285 has a high degree of repositioning accuracy operating in robotic movement mode.

If you are use the Rotary Optical Encoder (ROE) as an input device to move the manipulator between robotic moves, you may encounter difficulty with repositioning using robotic movement. When a knob is rapidly turned, it is possible for the ROE/microprocessor combination to command more microsteps than are actually moved. This will lead to the display of inappropriate position coordinates. This will have no functional consequence during normal operations, but it will have a confounding effect when using the robotic features.

If you use the robotic feature and your input device is an ROE, it is **imperative** that you observe the following recommendations:

- Never use greater than 7 µsteps/click (see the Controller Configuration section of the manual for a description of this operating parameter).
- Avoid rapid spinning of the ROE knobs.

Accessing MP-285 Robotic Functions:

1. From the main menu press <MOVE> followed by <*>. The Robotics menu will appear:

```
█⇨Home   ⇨New Home    ⇨Execute     ⇨ Learn
 ⇨Show home           ⇨ Select program
```

2. Press <TAB> to select the desired function and press <ENTR>.

The first half of this section gives the details on the three robotic commands related to the Home function: “Home”, “New Home”, and “Show home”. This is followed by a more comprehensive description of generalized robotic movement and then details on each of the associated commands: “Execute”, “Learn”, and “Select program”.

5.2 Home Functions

5.2.1 Home [MOVE*\Home]

The keypad commands are only necessary if you are using a Joystick. ROE users simply push the Home button on the ROE box to execute a move to the home location (for details see below).

To move the micromanipulator to the home position when using a Joystick:

1. From the movement screen, press <*>
2. Press <ENTR> to select the default (i.e. Home)
3. The micromanipulator will robotically move to the designated Home by the shortest path.
4. The Movement screen will be restored.

To move the micromanipulator to the home position when using an ROE:

If the **HOME** button on the ROE is up, in the off or “unlit” position, simply press the switch to activate the “Go Home” robotic movement. If the switch is down, in the on or “lit” position, you must toggle it off and on to issue the “Go Home” command. To reduce the possibility of inadvertently issuing a “Go home” command, it is advised to leave the button in the down position.

5.2.2 New home [MOVE*\New home]

To set a New Home position:

1. While in the movement screen move to the location you wish to establish as HOME
2. Press <*> to enter the robotics menu
3. <TAB> to “New Home” and press <ENTR>. You will see the following screen:

█⇨ Accept new home (TAB to edit)	⇨ spd = 2000 μm/s ⇨ res = .20 μm/Step

At this point, you have the option of setting both the speed and the resolution that will be used during the robotic move to the Home location.

To toggle between low and high resolutions:

1. <TAB> 2X to “res” and press <ENTR>. The screen above will change and read “—TAB to toggle—" in the lower left corner
2. Following the direction, press <TAB> to change the resolution from whatever is currently displayed to the other resolution. You will be warned that the change in resolution automatically changed the speed.

To change the speed:

1. <TAB> to “spd” and press <ENTR>
2. Use the number keys on the keypad to enter the desired speed in μm/s
3. Press <ENTR> to accept the new speed or <ESC> to abort the edit.

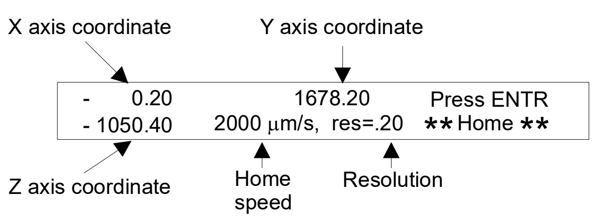
A few words on setting the speed and resolution of the home move:

- Changing the resolution of the move automatically makes a change in the speed that is calculated and displayed by the controller. For this reason, it makes the most sense to change resolution first and then set speed.
- Because the controller commands the manipulator in µsteps/s, and because there is a finite rate to the number of µsteps that can be issued, you will not be able to reach as fast speeds in high resolution. Therefore, if you desire the highest speed of movement it must be made at low (.20µm/µstep) resolution.
- In most applications, the resolution of robotic moves will have little consequence to the user. You may notice that moves at higher resolution are slightly smoother, however if you make a low- and high-resolution move at the same final speed you will probably find the differences very slight.
- The maximal speeds that can be commanded at both resolutions are 1330µm/s at 0.04µm/µstep and 6550 (MP-285) or 3000 (MP-285A) µm/s at 0.20µm/µstep.

5.3 Show home [MOVE*\Show home]

To review the ABSOLUTE coordinates of the current HOME position, the Home speed and the home resolution:

1. From the movement screen press <*> to access the ROBOTICS menu
2. <TAB> 4X to "Show home" and press <ENTR>. You will see the following screen:



Note that the coordinates shown in **Show Home** are ABSOLUTE. If you have your display on RELATIVE mode, the coordinates displayed when you reach the Home position can be different from those shown here.

5.4 Generalized Robotic Movements

The remaining commands in the Robotic Menu are directly related to generalized robotic movement: "Learn", "Execute", and "Select program".

The Learn routine is used to enter a new robotic program. It functions like a recorder which is turned on and then fed a series of moves simply by moving the manipulator to the desired location using the input device and then telling the recorder to capture the move. If desired, you can enter a different speed and resolution for each move. It is also possible to tell the controller to "Pause" during a robotic series and wait for a releasing keystroke before continuing.

The Execute command is used to run a robotic program. It provides you with several different modes of robotic program execution. A program can be executed once or multiple times in a loop. The program can be executed in the order it was recorded or in reverse order. If you pause in the middle of a program you have the option of resuming from the point you paused, or you can go back to the start. Finally, any program can be executed either in absolute mode, which measures coordinates with respect to the absolute origin of the controller or in relative mode, which treats each move as a differential move with respect to the point in manipulator space that program execution was started.

Finally, the Select program command allows you to perform various housekeeping tasks with respect to learned programs. For example, you may select and view a previously entered program or save a newly entered program.

The remainder of this section gives the details on how to use each of the commands related to generalized robotic movement.

5.5 Execute [MOVE*\Execute]

This command is used to EXECUTE the robotic movements programmed in the Learn routine (for details on programming robotic movements, see the Learn section that follows). To access the Execute functions:

1. From the movement menu press <*>
2. <TAB> 2X to key to “Execute” and press <ENTR>. You will see the following screen:

```
█⇨Do once   ⇨Loop   ⇨Reverse
 ⇨Abs/Rel        Now:  Absolute,  Program 1
```

   The lower right portion of the screen indicates that Program 1 is the currently selected robotic routine and that Absolute execution mode has been selected.

   At this point, you have several Execution options. You may execute Program 1 a single time using “Do once”. The program may be run in a cyclical fashion using “Loop”. Finally, the program may be run in opposite direction using “Reverse”. The procedure for all three of these operations is identical. Simply press <TAB> to go to the desired execution option. What follows is detailed here for “Do once”:

3. If the cursor is still on “Do once” simply press <ENTR>, if not <TAB> to “Do once” and press <ENTR>. You will see the following two screens in sequence:

```
Executing.    To stop:
press key & hold until end of move          1
```

```
1000.00              1000.00
   0.00                                     2
```

The values in the second screen may be different if you are running a program other than the factory default Program 1. The number in the lower righthand corner indicates the move the controller is executing where #0 is the first move of the program. The three numbers in the left hand and middle portions of the screen indicate the location of the manipulator at the beginning of the present move. Additional screens like the second will appear; one for each move in the program. If the move is quick (fast speed and/or short distance), the screen may only be visible for a fraction of a second.

If the robotic program being run contains a pause, like the factory default Program 1, a screen will be reached like the following:

```
Execution paused.          Press ESC to stop.
Press another key to continue.             4
```

As is obvious from this screen, you have two choices:

- If you press <ESC>, the robotic routine will be terminated, and you will be bounced back to the movement screen.
- If you press any other key, the robotic routine will continue its execution with any remaining steps.

Robotic program execution using the options "Loop" or "Reverse" is accomplished identically to one-time execution: simply <TAB> to the desired option and press <ENTR>. During execution, screens will advance in a similar fashion to that described for "Do once" above except in either a reversed direction or in a continuous cycle without an end point.

To end "Loop" execution: hold down any key until the end of a move. You will enter into a pause mode from where you have a chance to exit.

5.6 Resuming from a pause [MOVE*\Execute\Resume]

Anytime you exit a robotic routine while it is in progress, whether from a programmed pause or by holding a key until the end of a move, you will be given the opportunity to Resume execution of the robotic routine the next time you go to the Execute menu. Under these circumstances, the Execute screen will look like this:

```
█⇨Do once   ⇨Loop   ⇨Reverse   ⇨Resume
 ⇨Abs/Rel        Now:  Absolute,  Program 1
```

The option "Resume" has been added. If you select this option, using the <TAB> key and pressing <ENTR>, you will start robotic program execution of the indicated program at whatever move you exited during the previous execution.

Resume is preserved when the controller is turned off. The Resume option will remain in place until the program is run to completion or a new robotic program is selected.

5.7 Absolute/Relative Robotic Program Execution [MOVE*\Execute\Abs/Rel]

It may not be obvious that the first move of a robotic sequence is a move from the current location of the manipulator to the first program location. The nature of the first move is thus a function of the absolute position of the manipulator at the start of execution. If your robotic series is not executing in the way you expect, this may be the reason. For example, if you reset the Absolute Origin and find that your robotic routine no longer executes as expected, it may be because the original robotic routine was dependent on a set of absolute coordinates that were lost when the absolute origin was moved.

In order to separate Robotic functioning from the absolute coordinate system of the manipulator, the MP-285 controller allows Relative Robotic Program execution. In this mode, the controller treats all movements as changes rather than as movements to absolute locations. To do this, the controller makes the first robotic program location equivalent to the position of the manipulator at the start of program execution. All subsequent moves are then treated as differential moves from this location. In doing so, the absolute coordinate reference is suspended.

The easiest way to understand relative execution is by example. Imagine that you want to use the robotic capability of the MP-285 to methodically impale and microinject many cells. The controller can easily be taught, via the "Learn" function, to make a patterned impalement motion. One would bring a candidate cell into the center of the microscopic field of view and then teach the controller the impalement motion. To run such a robotic routine in Absolute mode, each cell must be brought to the central location under the injection pipette. This might not be the best way to quickly inject many cells. Alternatively, you can tell the controller to execute in Relative mode. This will allow you to manually position the pipette close to the cell using your input device and then begin the robotic routine. The controller will execute the robotic routine in this new location, not in the center of your field of view, as it would do in Absolute mode.

5.7.1 To toggle between Absolute and Relative modes of Robotic program execution:

1. From the movement screen press <*>
2. <TAB> 2X to "Execute" and press <ENTR>
3. <TAB> 3X (or 4X if "Resume" is present) to "Abs/Rel" and press <ENTR>
4. As is stated on the screen that appears, "Press <TAB> to toggle vector mode". "Vector" is MP-285 speak for movement. At the bottom of the screen, you will see the current mode of Robotic program execution. When the screen displays the mode you desire, press <ENTR>.
5. You will be back in the Execution menu. In the lower right, the correct Execution mode (Absolute or Relative depending on your selection) will be displayed.
6. You may now execute the program using any of the three execution modes "Do once", "Loop" or "Reverse". You may also choose to "Resume" a previously paused robotic routine. Note that this gives you the option of running part of a routine in one mode, (Absolute or Relative), pausing the routine, switching to the other mode (Relative or Absolute) and then resuming the routine.

The default Robotic movement mode is Absolute. Anytime you load a new program, the mode is switched back to Absolute. Furthermore, powering the controller off/on will also reset the robotic mode back to absolute.

5.8 Learn [MOVE*\Learn]

Learn is used to **"record"** robotic movements that can then be executed later. The MP-285 is capable of storing up to 10 robotic programs or routines, each of which can contain up to 55 moves. Alternatively, the controller can save a single routine (in Program#1) of 500 moves.

In order to use the Learn routine, Program location #1 should be empty; new programs can only be entered into this location. If there is a program currently in location #1 that you wish to save, you must first save it to a different location before entering the new robotic series (see "Select program"). If you attempt to write a new program to an occupied location #1, the controller will alert you that you will overwrite the existing program.

In addition to entering a new robotic program, it is also possible to append new moves to the end of an existing series. This is accomplished by loading the program you wish to modify into Program location #1 and then choosing the option in the Learn routine that allows you to add to the existing program rather than overwrite it.

To use the Learn routine:

1. In the movement screen, use your input device to move the manipulator to the desired first location of your new robotic series. Press <*>to access the Robotics menu
2. <TAB> 3X to "Learn" and press <ENTR>. Unless you previously cleared Program location #1, you will see the following screen:

```
WARNING:  Program 1 contains vectors.
Press ESC to quit or another to continue.
```

3. If you need to save the contents of Program 1, press <ESC>, if you do not, or you want to add new moves (vectors) to the robotic routine stored in Program location #1 press any other key to continue. You will see the following screen:

```
█ ⇨ Program 1: Add vectors after  6 - or -
  ⇨ Start new program 1 with vector no. 1
```

4. You have the option to add moves (vectors) at the end of the existing program (in this case the current program consists of 6 moves) or you can overwrite and start a new program. Use the <TAB> key to select the desired option and press <ENTR>. In either case you will get to a screen like the following:

```
█ ⇨ Add vector after    0      ⇨ End program
  ⇨ Insert pause      ⇨ Go home  ⇨ Delete
```

You will return to this screen once for every move you add to your robotic program. Each time you add a move, the "0" will increment (i.e., 1,2,3…). In addition to adding moves, you have several other options at this point ("End program", "Insert pause", "Go home" and "Delete"). These options are addressed next.

If you choose at this point to enter the current manipulator location as a move (vector): Do the following:

1. <TAB> to "Add vector after 0" (may be another number) and press <ENTR>. You will see the following screen:

```
█ ⇨ Accept vector        ⇨  spd = 6550 µm/s
    (TAB to edit)        ⇨  res  = .20  µm/Step
```

2. At this point, you can set both the resolution (minimal µstep size) and speed for the current move. You may set a different speed and resolution for each move. The default values are the values from the last programmed step you entered.

To toggle between low and high resolutions:

1. <TAB> 2X to "res" and press <ENTR>. The screen above will change and read "—TAB to toggle—" in the lower left corner
2. Following the direction, press <TAB> to change the resolution from what is currently displayed to the other resolution. You will be warned that the change in resolution automatically changes the speed.

To change the speed:

1. <TAB> to "spd" and press <ENTR>
2. Use the number keys on the keypad to enter the desired speed in /s
3. Press <ENTR> to accept the new speed or <ESC> to abort the edit.

Once you have set the speed and resolution to your desire, you must record the move or "Accept vector". The cursor will move to the correct location after either a speed or a resolution change, so all you have to do is press <ENTR>.

You will be return to a screen that looks identical to the Movement screen except that it will display an "L" in the lower right-hand corner to remind you that you are in the middle of the Learn routine and not in a standard movement screen.

1. Move to the next location in your developing routine and press <*> to begin recording this move. You will again see the screen:

```
█ ⇨ Add vector after     1       ⇨ End program
  ⇨ Insert pause      ⇨ Go home   ⇨ Delete
```

2. Notice that the "Add vector after #" portion has incremented by 1. To add the current manipulator location as another move (vector) press <ENTR>. You will again see a screen where you can change the speed and resolution of the move.
3. You can change the speed or resolution or simply press <ENTR> to record the move.
4. Repeat steps 1 thru 3 until you have entered all desired moves into your robotic routine.
5. To end the Learn routine, press <*> from the Movement screen within the Learn function and then <TAB> to "End program" and press <ENTR>.

A screen will briefly be displayed which tells you how many instructions (moves) were recorded. You will then be returned to the standard Movement screen.

You can use several options during the entry of moves into a robotic routine. They are the other functions you have seen on this screen:

```
█ ⇨ Add vector after     1       ⇨ End program
  ⇨ Insert pause      ⇨ Go home   ⇨ Delete
```

- **Insert pause** lets you put a break in the robotic routine. The manipulator will stop moving and will not move further until released by a key press or an external TTL signal*. (*for controllers that have this option installed)
- **Delete** allows you to delete the most recently added program step
- **Go home** allows one to quickly move the manipulator to the current home location
- **End program** is used to exit the Learn routine. There is no other way to exit the Learn routine.

A few words on setting the speed and resolution of moves in your robotic routine:

- Changing the resolution of the move automatically makes a change in the speed that is calculated and displayed by the controller. For this reason, it makes the most sense to change resolution first and then set speed.
- The controller commands the manipulator in µsteps/s and because there is a finite rate to the number of µsteps that can be issued, you will not be able to reach as fast speeds in high resolution. If you desire the highest speed of movement, it must be made at low (.20µm/µstep) resolution.
- In most applications, the resolution of robotic moves will have little consequence to the user. You may notice that moves at higher resolution are slightly smoother, however if you make a low- and high-resolution move at the same final speed you will probably find the differences very slight.
- The maximal speeds that can be commanded at both resolutions are 1330µm/s at 0.04µm/µstep and 6550 (MP-285) or 3000 (MP-285A) µm/s at 0.20µm/µstep.

Finally, you may find it much easier, when you enter a new robotic series via the Learn routine, to first set your absolute origin to zero (see [PRGM\Setup\Axes\New origin]. If you perform the reset just prior to entering the first coordinate of the series, you will have an easy point of reference for making the rest of the moves in your series.

5.9 Saving, Loading and Viewing Robotic Programs [MOVE*\Select program]

The commands accessible within “Select program” allow you to perform several housekeeping tasks with respect to robotic programs. These include **saving** a newly entered program, **loading** a previously entered program so it can be executed, and **viewing** the moves in a robotic routine prior to execution.

To access the “Select program” commands:

1. From the movement screen, press <*>
2. <TAB> 5X to “Select program” and press <ENTR>. You will see the following screen:

```
Program      █⇨ Save / Clear     1234567890
is  now  1     ⇨ Load   ⇨ View    *    *
```

This screen shows the commands available to you. On the left side of the screen the Robotic program currently loaded (ready for execution) is given. On the right side of the screen is an indicator of the robotic program storage slots (1, 2, 3…9 and 0). Asterisks indicate slots that are occupied.

The commands “Save/Clear” allow you to take the currently loaded program and save it in one of the unoccupied slots. This command will most often be used to save a newly written program. Alternatively, it may also be used to save the program in location #1 before writing a new program there. If you wish to save to an occupied slot, the controller will instruct you to “Clear” the slot first.

The commands “Load” and “View” operate on the programs in the slots on the right. Load is the command that takes a saved program and makes it accessible for execution (running). After loading a program from slot *X*, the left-hand side of this screen will read, “Program is now *X*”.

The View command gives you a chance to preview the moves in a robotic program saved in one of the slots prior to running.

The details on how to use each of these commands are as follows:

5.9.1 To use “Save/Clear”:

1. From the Movement screen, press <*>
2. <TAB> 5X to “Select program” and press <ENTR>
3. Press <ENTR> to access “Save/Clear”. The screen will change to the following:

```
Program        " Save / Clear "    1234567890
is  now  1                         *    *
```

4. Use <TAB> to move to an empty slot and press <ENTR>. You should see an asterisk appear under the slot you chose.
5. If you choose to save in an occupied slot (e.g. #6 in the above screen), the query “ERASE ?6 ENTR = OK” will appear in the center of the screen. If it is okay to erase, press <ENTR> else press <ESC>. If you press <ENTR>, you will be queried a second time. Again, you must press <ENTR> to continue. At this point, you will have cleared the old

slot, and its asterisk will be gone. However, you have not yet saved the program in the newly empty slot. To complete the save, you must press <ENTR> again, <TAB> to the now empty slot and press <ENTR> a fourth time. The asterisk will reappear signaling that the new program has been saved.

Note, you can only "Save" the currently loaded program. If you have just finished a Learn routine, the currently loaded program will be #1. If you are about to use the Learn commands to enter a new routine (by default into location #1) and you may need to save the program in location 1 first. Before *X* can be saved, you have to make sure it is loaded (i.e. the righthand side of the screen must say "Program is now *X*").

5.9.2 To use "Load" :

1. From the Movement screen, press <*>
2. <TAB> 5X to "Select program" and press <ENTR>
3. <TAB> to "Load" and press <ENTR>. The screen will change as follows:

```
Program        " Load "         1234567890
is  now  1                      *    *
```

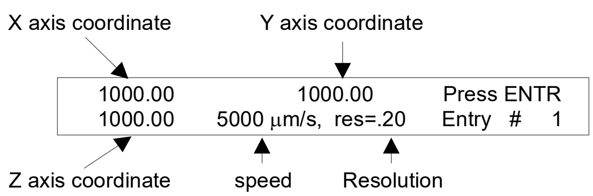
4. <TAB> to the program slot you wish to Load and press <ENTR>. The left-hand side of the screen will now display the number of the program just loaded.
5. To execute the newly loaded program, press <ESC> (back to the movement screen), <*> (back to the robotics menu), <TAB> 2X to "Execute" and press <ENTR>. Follow the instructions as detailed in section "Execute [Move*\Execute]".

5.9.3 To use "View" :

1. From the Movement screen, press <*>
2. <TAB> 5X to "Select program" and press <ENTR>
3. <TAB> 2X to "View" and press <ENTR>. The screen will change as follows:

```
Program        " View "         1234567890
is  now  1                      *    *
```

4. <TAB> to the program you wish to View and press <ENTR>. You will see a screen like the following:



The screen displays the first move or "entry" of the Robotic program you chose to view. As can be seen, the coordinates, speed and resolution that were recorded by each entry during the LEARN routine are displayed.

5. To view each successive move or entry in the program, press <ENTR>.
6. After the last entry has been displayed, press <ENTR> again to return to the "Select program" screen. Note that View does not load the program; it simply displays the steps of the program on screen. One must use the "Load" command to load programs.

6. SETUP SELECTION [*]

The MP-285 controller allows you to adjust several parameters that alter the responsiveness of the micromanipulator to commands from the input device being used. The current settings are remembered when the controller is turned off. Furthermore, up to 10 alternative configurations can be stored as **SETUPS** that can be selected from the **Main Menu**.

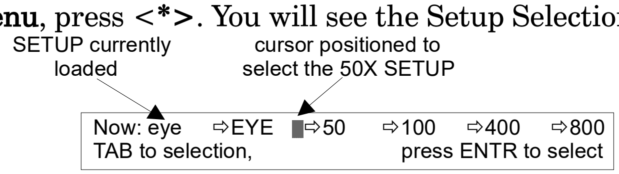
As delivered, the MP-285 controller has 5 resident **SETUPS** in locations 1 through 5 (see Table 6-1). These were loaded when the unit was assembled, and they represent a wide range of input device response. You may find that one of these **SETUPS** delivers close to the type of responsiveness that you desire from the MP-285. You may eventually want to change one or more of the characteristics of a factory **SETUP** and save the reconfigured version. A thorough discussion of input device parameters and their alteration can be found in the manual section **Controller Configuration.** The methods for saving a user configured set of parameters as a new SETUP is detailed in the same section (see **[PRGM\Save/Load]**).

The 5 resident **SETUPS** which were loaded into the first five memory slots can be loaded using the **Setup selection menu**. These five **SETUPS** are labeled in this menu as **EYE, 50, 100, 400 and 800** referring to the power of magnification at which you might be working under a microscope. Alternatively, setups can be selected rapidly using their corresponding number keys, 1 through 5. All 5 Setups are listed in the table at the end of this section.

The original 5 resident SETUPS can be restored at any time (see **[PRGM\Setup\Utilities\Reset all values]**). WARNING: restoring factory setups will erase setups 6, 7, 8, 9 and 0 as well!

6.1 To use the Setup selection menu (select factory installed SETUPS by label)

1. From the **MAIN Menu**, press **<*>**. You will see the Setup Selection menu:



2. If any changes have been made to the current SETUP, A warning screen will appear instead. If you wish to save the current setup, press <ESC> and follow the instructions for Save\Load setup under Controller Configuration. If you do not wish to save the current setup, press <ENTR>
3. If the warning screen does not appear and/or you do not need to save, <TAB> to select the desired setup by label “EYE”, “50”, “100”, “400” or “800”. The table at the end of this section gives the parameters of each setup by Label.
4. Press <ENTR> to select the chosen setup.
5. Press <ESC> to get back to the MAIN MENU. Press <MOVE> to start using the selected SETUP.

6.1.1 To rapidly select any SETUP from the Main menu:

1. From the **Main menu**, press the numeral key corresponding to the **SETUP** you wish to use (**1** through **9** or **0**).
2. The Main menu screen will change accordingly in the top right corner to indicate the newly loaded **SETUP** number.

SETUP currently loaded

```
S⇨1          * * * MP-285   Controller * * *        P⇨ 1
Press MOVE,     PRGM       or select mag. With  *
```

3. If any changes have been made to the current SETUP, A warning screen will appear. If you wish to save the current setup, press <ESC> and follow the instructions for Save/Load setup under Controller Configuration. If you do not wish to save the current setup, press <ENTR>
4. To start using the selected **SETUP**, press <**MOVE**> to advance to the **Movement Screen**.

Table 6-1. Version 3.04 Factory Installed Setups.

	Program 1	**Program 2**	**Program 3**	**Program 4**	**Program 5**
Program alias	Eye	50X	100X	400X	800X
Speed range	99	99	99	90	80
Offset	99	50	20	20	1
Jump to max at	6000	3000	2500	1000	500
Max speed	10000	2500	5000	1000	500
µ steps/pulse	20	10	5	2	1
µ steps/second	5000	5000	5000	5000	5000
µ steps/click	20	10	5	1	1
Axes assignments	024	134	134	134	134
Mode/Resolution	CL	CL	CL	CH	CH

7. MANIPULATOR INSTALLATION

7.1 Mounting the MP-285: General Considerations

Sutter Instrument Company realizes that many experimental applications in which the MP-285 will be used may be relatively hostile due to the use of saline solutions in close proximity to the instrument. For that reason, we have used, whenever possible, corrosion-resistant materials to build the MP-285. Even so, **THE MP-285 IS NOT CORROSION-PROOF**. Installation and use of the unit should be conducted so as to avoid direct contact with ANY solutions or aerosols. Particular attention should be given to protecting the motors! If a spill does occur, it should be immediately and thoroughly cleaned. Wipe the affected area with distilled water to remove any salts or other corrosive material and then wipe the area dry. Sutter Instrument Company assumes no responsibility for damage to the micromanipulator caused by a failure to conform to these recommendations.

The base of the MP-285 is also the mounting surface for the bearing ways of the X-axis. **It is imperative that it not be subjected to torsional stresses that may cause misalignment and malfunctioning of the bearings**. We have found that when the MP-285 base plate is pressed firmly against most mounting surfaces; torsional stress can be transmitted to the base. **In order to assure that there are no stresses applied to the base, you should insert a precision-thickness washer on each of the 4 mounting screws.** The washer should be inserted between the MP-285 base and your mounting surface. This will prevent the stressing of the manipulator. We have included a supply of 4 appropriate washers with your MP-285.

When mounting the MP-285 there should be no physical impediments to the full travel of the three axes. Pay attention to the screws or clamps used to secure the base of the micromanipulator to make certain that the X-axis slide can move to the end of its travel without obstruction.

7.2 Minimizing Noise

We are aware of two potential noise sources for users coupling their MP-285 with high-gain, high input impedance, electrophysiological recording amplifiers. The first is associated with the manipulator and its connecting cable from the controller. Under certain circumstances, these may act as an antenna concentrating electric field noise that originates from nearby electrical equipment (computer monitors, and fluorescent lights are the most notorious offenders) and bringing it into close proximity with recording apparatus. **Grounding the manipulator will largely eliminate this noise source.** Ground tabs are located on each motor housing for this purpose. It should be noted that the manipulator produces negligible electrical noise when it is not moving because it is powered by a linear power supply with no AC current present.

The second potential source of noise is associated with the magnetic field radiated from the power supply transformer in the controller. Certain electrophysiological amplifiers are more susceptible to generating noise when exposed to this magnetic field. **This noise can be minimized or eliminated by allowing a maximal distance between your MP-285 controller and your amplifier.** It may be possible to decrease this noise source further by placing a soft steel plate or a piece of high permeability metal between the controller and the amplifier.

7.3 Mounting the MP-285: Specifics

The base of the MP-285 has 4 mounting holes laid out perpendicular to one another on 2.5" centers. The holes accommodate M3 metric or # 4 English machine screws. A template (drawn to scale) can be fashioned based on the footprint shown in Figure 7-1 and used to drill holes in a suitable mounting surface.

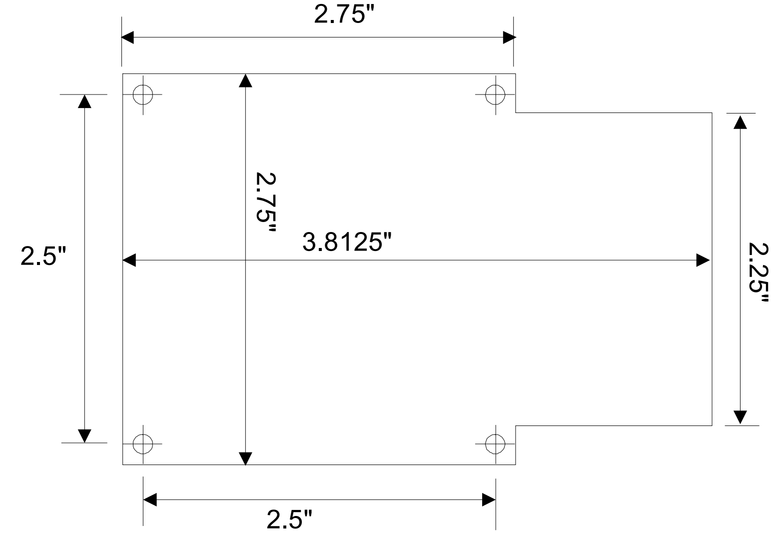


Figure 7-1. MP-285 footprint (not to scale).

This hole pattern is also compatible with mounting the MP-285 onto the MP-285 mounting adapter plate, part #285210 (see Figure 7-2) using M3-.5 x 14mm metric screws or onto the 285RBI rotary base (see Figure 7-3). Regardless of the mounting surface, use a precision thickness washer on each screw between the MP 285 base and the mounting surface. While it is possible to insert and tighten all four mounting screws with the MP 285 in one position, you may find it convenient to connect and power up the manipulator and position the X-axis for the best access to the screw holes.

7.3.1 Mounting Adapter Plate (285210)

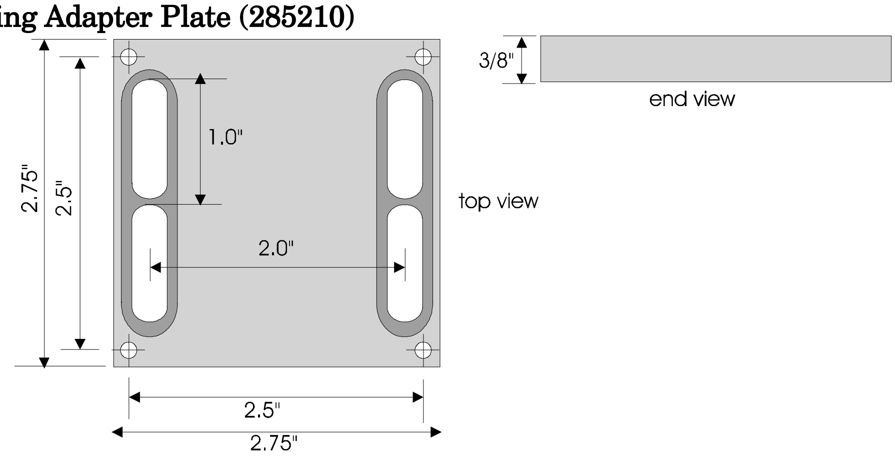


Figure 7-2. Mounting adapter plate (285210).

The Mounting Adapter plate is designed to be mounted on to any surface providing either ¼" or #10 holes on 1" centers. Because there is limited access to the screw slots once the 285210 is connected to the manipulator, the adapter must be surface mounted before attaching the

7.3.2 Rotary Base (285RBI)

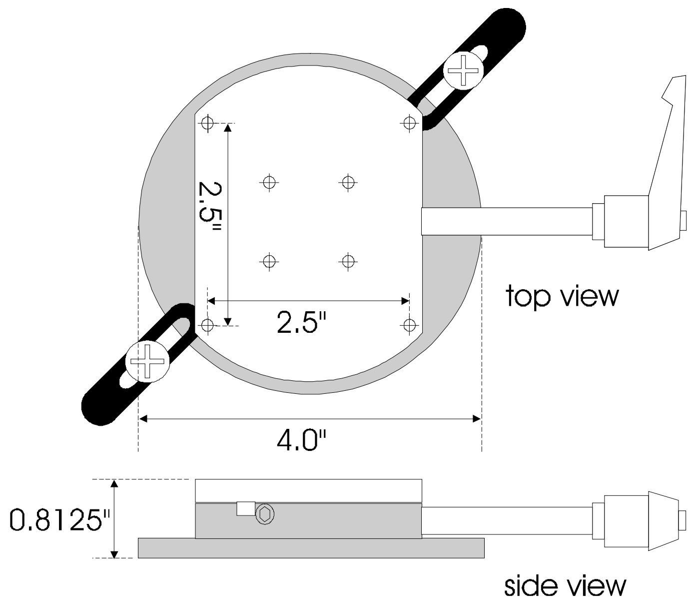


Figure 7-3. Rotating base (RBI285).

The Rotating Base can be clamped to an appropriate surface such as a microscope stage or one of the optional MT series micromanipulator-stands offered in the Sutter catalog. Each of the two "toe" clamps included with the 285RBI are meant to be screwed down using one ¼" - 20 x ¾" (or equivalent) bolt and flat washer. The rotary base should be clamped to a surface with appropriately sized holes, spaced to maximize the overlap of the toe clamp with the base plate of the 285RBI. Before tightening the toe-clamp bolts, make certain that the 90° rotation of the upper plate corresponds to the desired working range of the manipulator. Rotate the bottom plate if this is not the case. The relative position of the handle used to lock the 285RBI on the stem may also need to be adjusted (so that it clears the surface on which the base is installed). **The handle is spring-loaded on its stem and can be turned independently if pulled outward first:**

1. Turn the handle clockwise to lock the base and prevent the stem of the handle from rotating.
2. Pull the handle outward (away from the stem).
3. Turn the handle to the desired "locked" position and let go. It is spring-loaded to return it to its full inward stop where it should engage with the stem. If this doesn't happen, turn it slightly in one direction or the other until the handle engages with the stem.

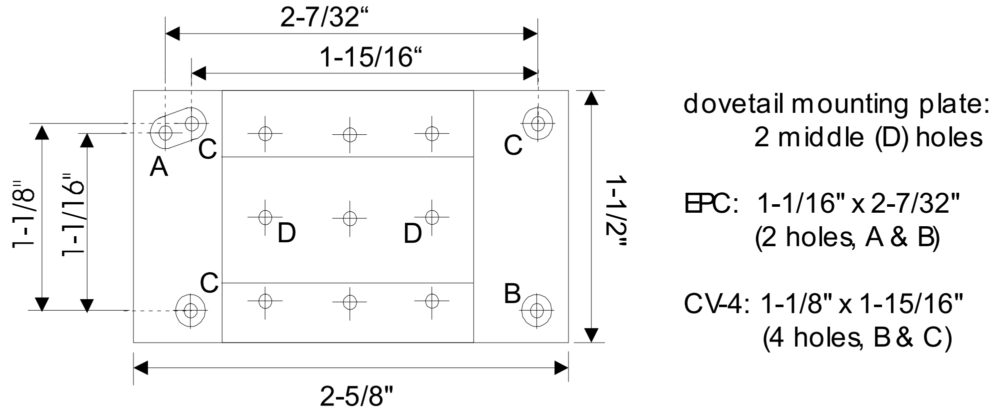
7.4 Mounting pipette holders and headstages

The mounting hardware included with the MP-285 is designed to eliminate the need for custom machining of headstage and pipette holders. Furthermore, the MP-285 will directly accommodate Axon Instruments' new 203B headstage.

The rotating **dovetail base** is reversible if you need to re-configure the MP-285 for use on the opposite side of your experimental set-up. The vertical position of the **rotating dovetail assembly** relative to the Z-axis slide assembly can also be adjusted in ½-inch intervals. Removal and re-installation of the **rotating dovetail assembly** is detailed in the section subtitled **Reversing the configuration of the brake plate.**

7.5 Plate-mounted Devices

The most stable way to mount a headstage is to fasten it directly to the Z axis slide. For most voltage clamps it is also essential to electrically isolate the headstage from the manipulator. The standard **dovetail mounting plate** included with the MP-285 has a plastic plate that provides electrical isolation while allowing for a direct mount of Axon Instruments' CV-4 headstage or Heka Electronics' EPC-7, 8, or 9 headstages.



The user can drill additional holes to allow the plate-mount installation of other headstages.

7.5.1 To install the CV-4 or EPC-7, 8 or 9 headstages onto the standard dovetail mounting plate:

1. Remove the plastic isolation plate provided with the headstage (4 screws for the CV-4 and 2 screws for the EPC-7, 8 or 9) and save the screws.
2. Line up the appropriate holes on the plastic portion of the Sutter dovetail mounting plate with the holes on the headstage and fasten using the original screws.
3. Slide the dovetail mounting plate/headstage assembly into the dovetail base and secure it by tightening the dovetail clamp set screw (see Figure 7-4).

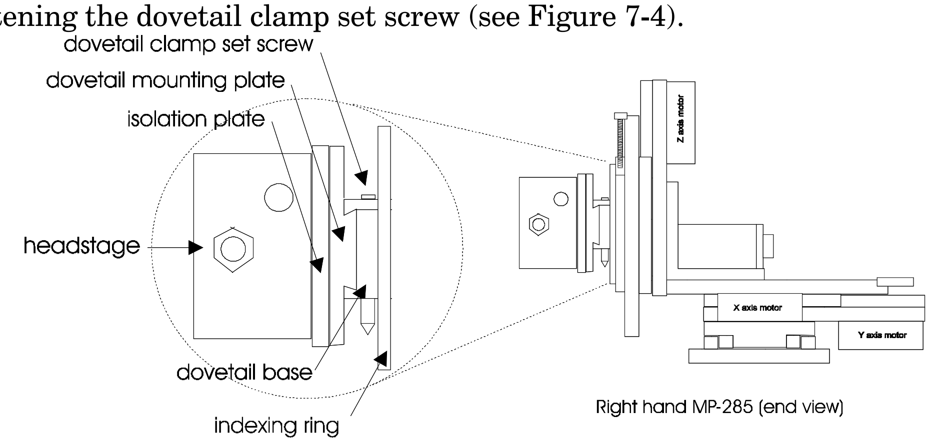


Figure 7-4. Detail of the headstage surface-mounting system.

An **optional 4" dovetail mounting plate** (Figure 7-5, right) is also available for installing longer headstages or for extending a directly mounted headstage closer to (or away from) the cell or tissue being studied.

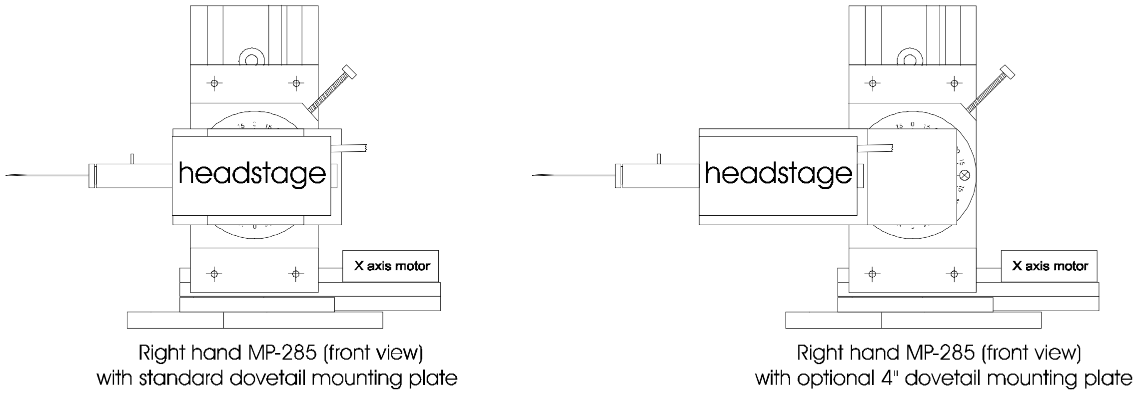


Figure 7-5. Comparison of the standard and 4” dovetail mounting plate (left and right figures, respectively).

7.6 Rod-Mounted Devices

Rod-type pipette holders and rod-mounted headstages are easily installed onto the Z-axis slide of the MP-285 using the **dovetail rod holder** provided with this unit (see Figure 7-6).

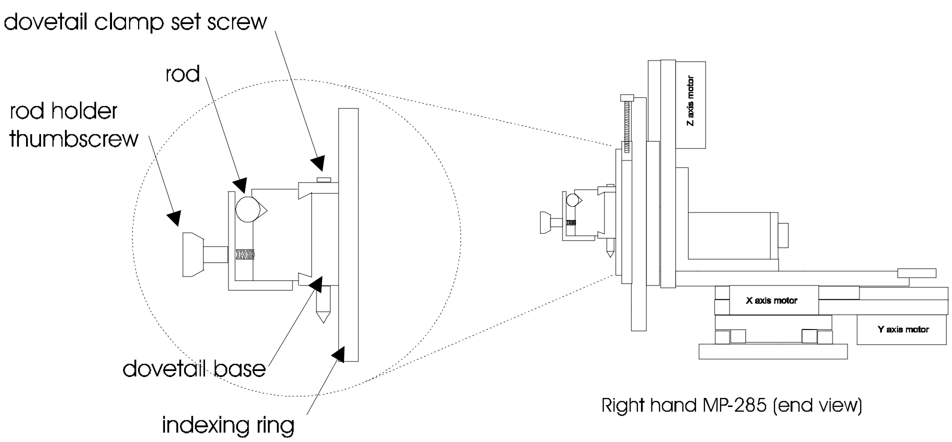


Figure 7-6. Detail of the rod holder mounting system.

The **dovetail rod holder** slips into the **dovetail base** and is secured by tightening the **dovetail clamp set screw**. Rods are secured in the **dovetail rod holder** by tightening the **rod holder thumbscrew** (see Figure 7-6).

Although some voltage clamp headstages are meant to be rod-mounted, it is highly recommended that, whenever possible, headstages be surface-mounted as described above. Plastic mounting rods are often used in rod-mounted systems to electrically isolate the headstage from the manipulator. Unfortunately, plastic is very sensitive to heat and can be a major source of “drift”. If you wish to rod-mount your headstage, remember that it is important to maintain as rigid a system as possible between the rod holder and the tip of the pipette.

7.7 Adjusting the Rotating Dovetail Base

The **rotating dovetail base** allows the pipette holder or headstage to be tilted up for convenient pipette changes and then back down to a working position that is defined by an adjustable detent. That detent position can be set by eye, using an actual position that you wish to maintain, or it can be set using the compass ring, based on a specific angle that you wish to approximate. The following instructions describe how to position the detent.

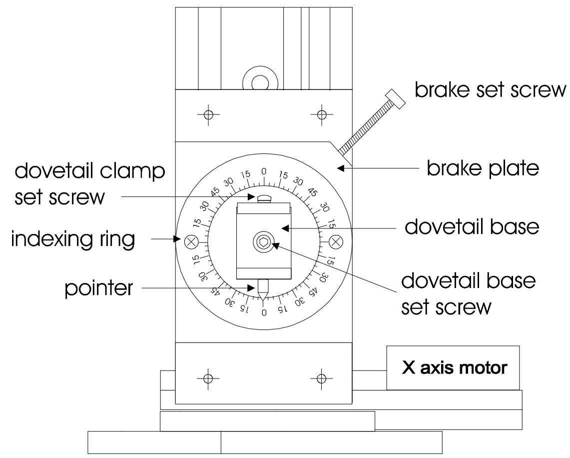
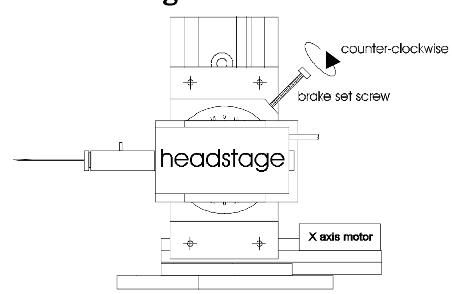


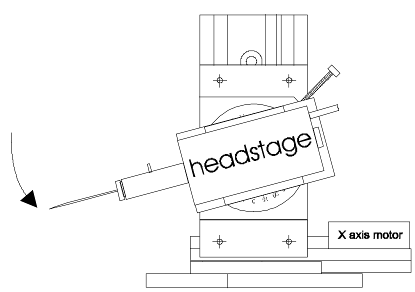
Figure 7-7. Detail of rotating dovetail base (front view).

7.7.1 To set the detent to a desired working position or angle:

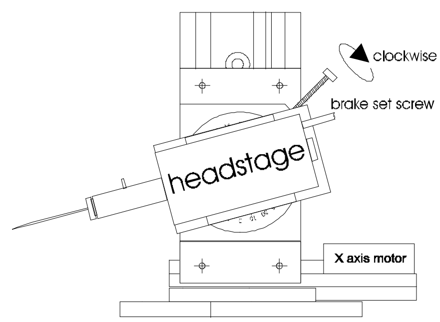
1. Release the brake by turning the brake set screw counterclockwise



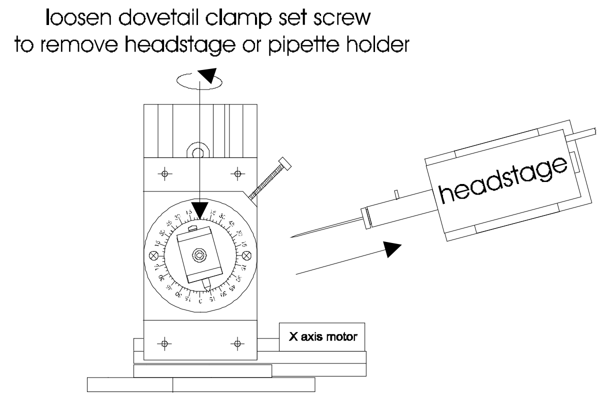
2. Twist the headstage until the rotating dovetail base is stopped in its detent (remove the headstage or pipette holder if they prevent this rotation)



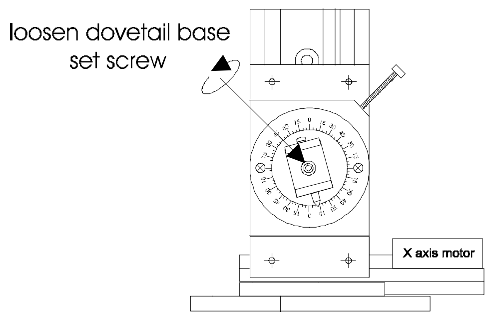
3. Firmly tighten the brake by turning the **brake set screw** clockwise



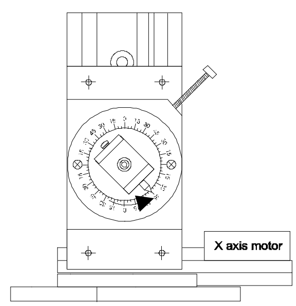
4. Remove the headstage or pipette holder (if installed) from the **dovetail base**



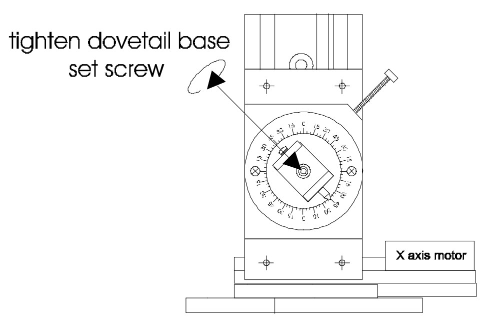
5. Loosen the **dovetail base set screw** so that the base is free to rotate (but not so loose as to allow movement from its own weight)



6. Rotate the **dovetail base** to the desired working position by eye or set it to a specific angle using the indexing ring as a reference



7. Carefully tighten the **dovetail base set screw**, avoiding rotation of the **dovetail base** in the process



8. Replace the headstage or pipette holder into the dovetail base and tighten the dovetail clamp set screw

The detent should now be set at the desired angle.

7.7.2 Further information about the brake set screw:

- You must loosen the **brake set screw** to rotate the headstage or pipette holder to or from the detent position.
- The **brake set screw** must be tightened for stable holding in any position, including when it is in the detent position

7.7.3 Further information about the relationship between the dovetail base, the brake disc and the dovetail set screw that secures them together:

- During routine movements to and from the detent position, the **dovetail base set screw** must be tight to prevent the **dovetail base** from rotating independently of the brake disc. They must move together during routine use to maintain the desired relative position of the detent.

While adjusting the desired angle for the detent, position (steps 5, 6 and 7, above) the **brake set screw** must be tight to prevent the **brake disc** from moving with the **dovetail base**. Any movement of the **brake disc** during these steps will cause an incorrect positioning of the detent.

7.8 Optional Mounting Configurations

This section outlines several optional mounting configurations. Under certain circumstances, these may be useful extensions of the standard configurations detailed in “Installation”.

7.8.1 Horizontal offset

In your experimental set-up there may be physical hindrances that prevent the MP-285 from getting close enough to the desired working area. Examples of such obstructions include condensers or differential interference contrast filters on inverted microscopes and nosepieces on upright microscopes. In such cases, it is useful to offset the headstage mounting system horizontally from the manipulator. This can be done using the optional HORIZONTAL EXTENSION PLATE (285310) that can be added by the user at any time. The plate can be installed in either of two positions to give a horizontal offset of 1.75” or 2.375”.

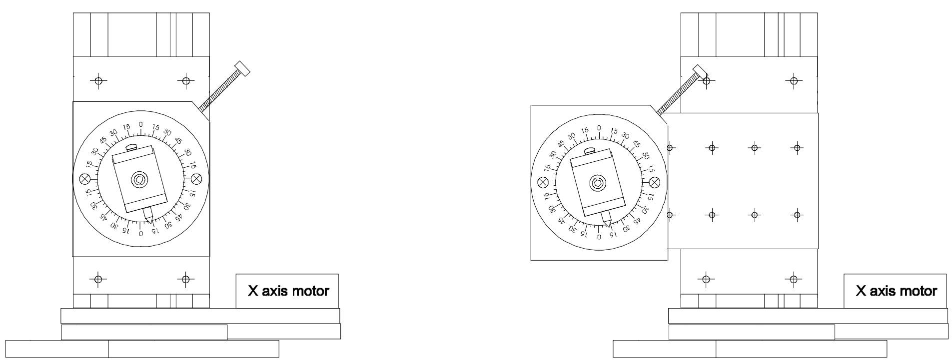


Figure 7-8. Comparison of MP-285 with and without horizontal extension plate (right and left figures, respectively).

7.8.2 Vertical offset

In many experimental set-ups a very flat approach angle is required to get the micropipette into the desired working area. This is especially true when using an upright scope. Often, the desired angle of approach can be accomplished if the rotating dovetail assembly is screwed to the lowest position on the Z-axis slide. However, in some cases the lowest position on the Z-axis slide will still be too high to allow the desired angle. Using the optional VERTICAL EXTENSION PLATE (285305), it is possible to vertically offset the rotating dovetail assembly even further. This piece can be added by the user at any time. The plate can be installed in one of four positions to give additional vertical offset of 0.5” up to 2.0” in 0.5” increments. This same plate can also be used to provide additional height over the same range (0.5” to 2.0”) (see Figure 7-8).

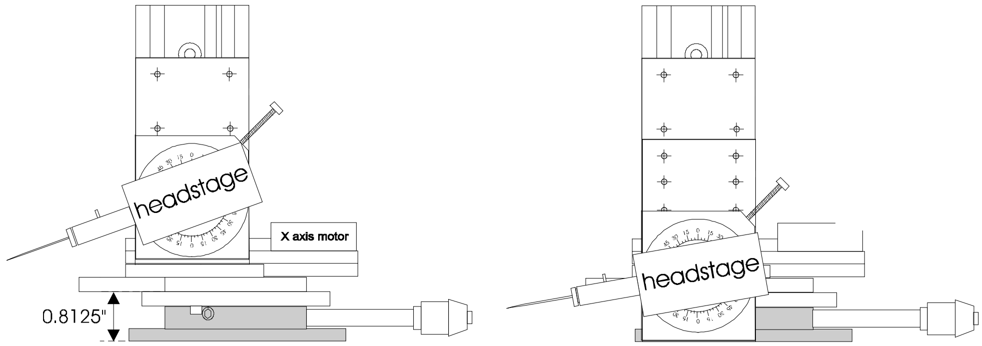


Figure 7-9. Comparison of MP-285 with and without vertical extension plate (right and left figures, respectively).

7.9 Mounting the Manipulator Rotated 90° from the Stage Plate

In some experimental set-ups there may be physical hindrances to the rear of the manipulator that impede the installation, translation or rotation of the MP-285. This is known to be the case when the MP-285 is used with an Olympus Model BX50 microscope. In such cases, it is useful to mount the manipulator with the “Y axis” oriented right-to-left (see Figure 7-10).

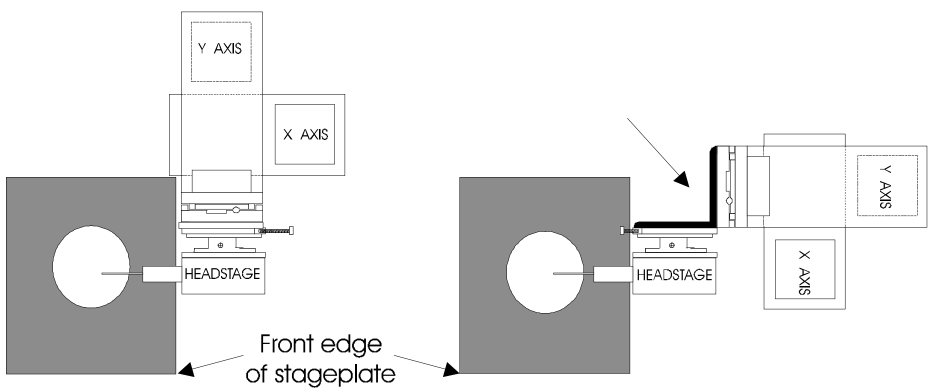


Figure 7-10. Top view of MP-285 mounted without and with a right angle headstage adapter (arrow).

The right angle headstage adapter allows the headstage to be correctly oriented relative to the working area (stage plate) even though the Z-axis slide is now 90º from its normal position. The right angle headstage adapter (285300) can be added by the user at any time. The plate is drilled to allow installation of the headstage mounting hardware on one “leg” of the adapter while the other “leg” is screwed down to the Z-axis slide.

7.10 Mounting the MP-285 Elevated at an Angle

The MP-285 is normally installed with the base plate mounted horizontally. Other orientations can be used but care must be taken to assure proper loading of the cable/spring slide assembly of the axis slide assemblies. If you plan to use other orientations, you may wish to contact Sutter Instrument Technical Support (415-883-0128) for help in planning your installation.

One side of each slide assembly is connected to the other by a cable. The extended portion of the cable shortens as the motor winds the cable onto a capstan and lengthens when the motor reverses direction to allow the cable to unwind. A spring is also stretched between the two sides of the slide assembly to oppose the force exerted by the cable (see Figure 7-11).

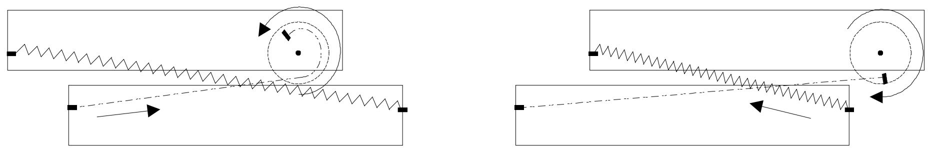


Figure 7-11. Cable/spring loading of the driven slide.

In a standard installation, the Z axis slide assembly is oriented vertically and the “moving” side of the slide assembly (the driven slide) “hangs” from the cable (see Figure 7-12).

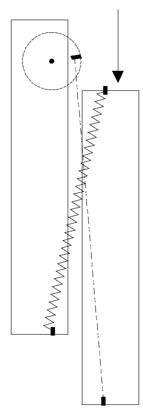


Figure 7-12. Driven slide (arrow) orientation.

In this way, the gravitational load is carried by the cable rather than by the spring. The same principle should be observed if you choose to install the MP-285 with either the X or Y-axis oriented vertically. If the X-axis is mounted vertically, about 90% of the mass of the manipulator is “suspended” by the linkage between the two sides of that slide assembly.

The manipulator should always be oriented so that **the cable**, rather than the spring, is supporting that load. In Figure 7-13, the X-axis assembly is oriented vertically using a “right hand” MP-285; for simplicity, the springs are not depicted. On the left, the gravitational load is being carried by the spring. On the right, the cable is carrying that load. Note that when mounted in the correct orientation, the X-axis motor will be down.

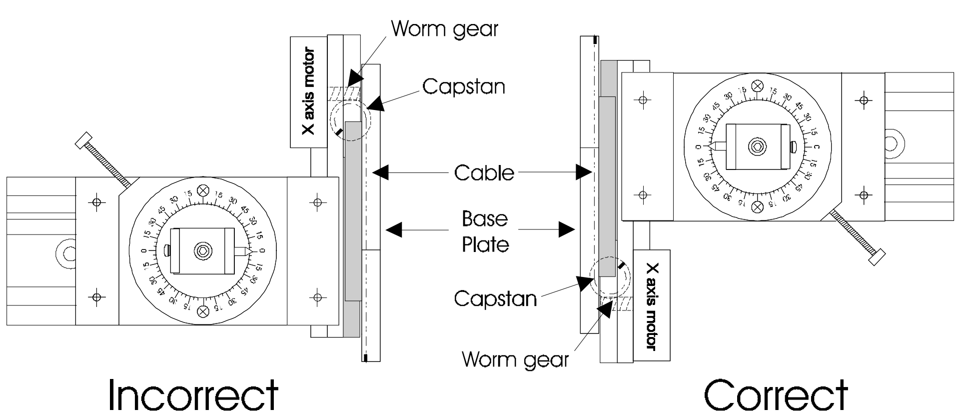


Figure 7-13. Mounting the MP-285 with the X-axis oriented vertically.

Figure 7-14 illustrates the correct orientation when installing the MP-285 with the Y-axis vertically using a "right hand" MP-285. Note that in this case, when mounted correctly, the Y-axis motor is on the top.

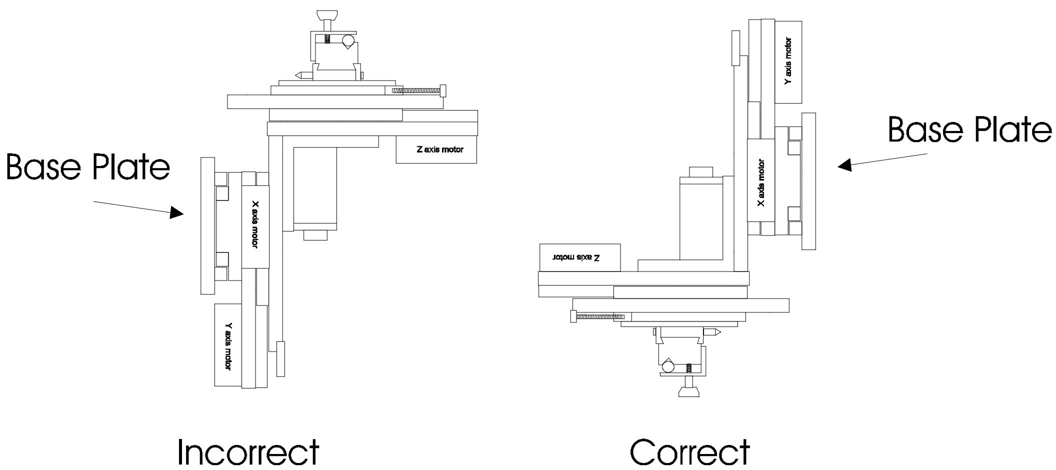


Figure 7-14. Mounting the MP-285 with the Y-axis oriented vertically.

7.11 Reversing MP-285 Handedness

Note: Over time, several design changes have been made resulting in minor modifications in the assembly of the MP-285 micromanipulator. Because of these differences, the handedness of some manipulators can be more easily changed than others. It is recommended that you contact Sutter Instrument Technical Support (415-883-0128) before you attempt to change the handedness of your manipulator.

The MP-285 has an asymmetric design with respect to the orientation of the X-axis slide and the brake set screw on the rotating dovetail mounting assembly (see Figure 7-15).

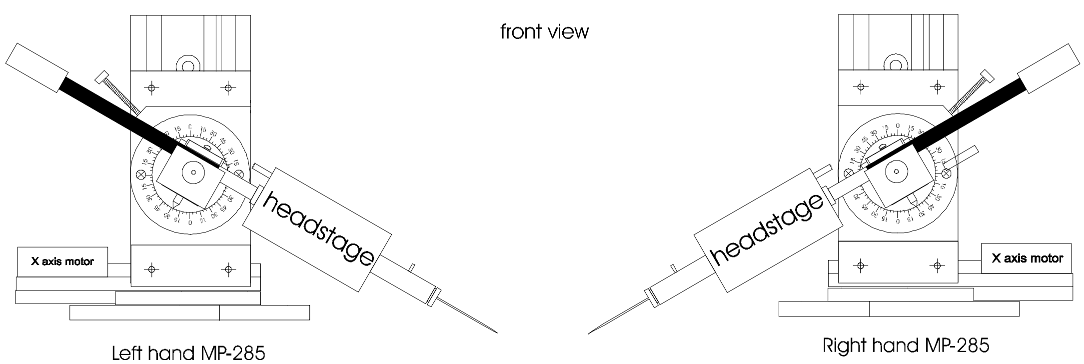


Figure 7-15. Right- and left-hand manipulators.

The handedness of any unit can be converted to allow its use on the opposite side of your setup. This requires reversing the orientation of the bottom-most (X-axis) slide assembly, its wiring harness, and the brake plate (located on the Z-axis slide).

7.12 Reversing the configuration of the X-axis slide assembly and its wiring harness

1. To avoid manually moving the manipulator slide assemblies and possibly damaging the movement, install the 6 shipping screws as described in APPENDIX C of this manual before proceeding.
2. Identify the DSUB connector cover. It is located on top of the Y-axis slide assembly, immediately behind the Z-axis slide assembly (see Figure 7-16).

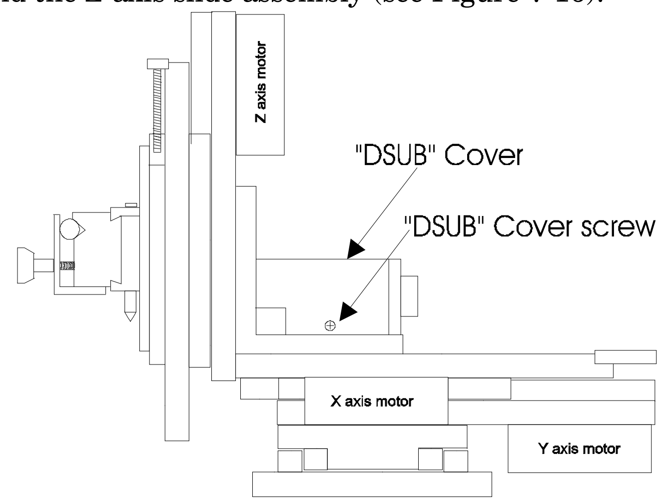


Figure 7-16. DB-25 connector (D-sub) cover removal.

3. Remove the 2 screws (one on either side) that hold the cover down and then lift the cover off.
4. Identify the four hex (Allen) "cap" screws that mount the Y-axis slide assembly on top of the X-axis slide assembly (indicated by arrows in Figure 7-17). Using a 2.5mm Allen wrench, remove those 4 screws. The Allen wrench that was shipped with the MP-285 for removing the shipping screws is the correct size for these screws as well.

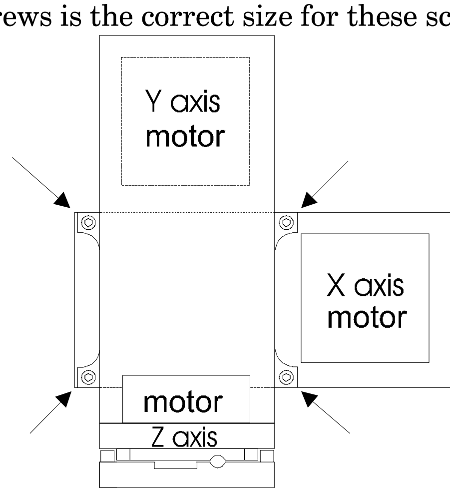


Figure 7-17. Y-axis mounting screws indicated by arrows (Right hand MP-285, top view).

5. Slip the X-axis slide assembly out from under the rest of the micromanipulator in the direction of the X-axis motor and wiring harness.
6. Bring the X-axis slide assembly over the top of the micromanipulator and slip it under Y-axis slide assembly from the other side.
7. Make certain that the wiring harness for the X-axis motor is not twisted or restrained.
8. Re-attach the Y-axis and X-axis assemblies using the 4 Allen "cap" screws.

9. Replace the DSUB connector cover with its two screws.
10. Remove the shipping screws.

7.13 Reversing the configuration of the brake plate

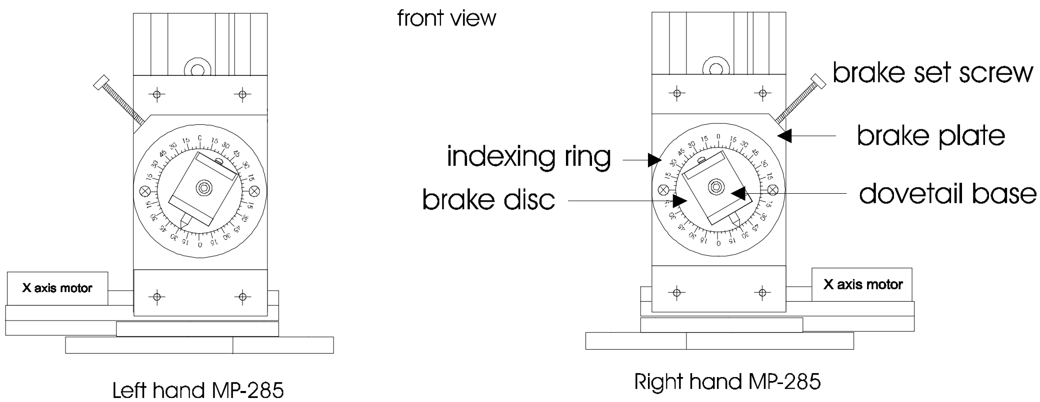


Figure 7-18. Brake assembly parts.

1. Remove the two screws that hold the indexing ring onto the brake plate and lift the ring off.
2. Pull the dovetail base/brake disc assembly straight out to remove it from the brake plate.
3. Remove the 3 screws (previously concealed by the indexing ring) that hold the brake plate to the Z-axis slide assembly.
4. Flip the brake plate over and re-attach it to the Z-axis slide assembly with its 3 screws.
5. Replace the dovetail base/brake disc assembly into the brake plate.
6. Re-attach the indexing ring to the brake plate using its 2 screws.
7. The brake set screw will now be oriented for use from the opposite side of the micromanipulator.

(This page intentionally left blank.)

8. EXTERNAL CONTROL

NOTE: Unless otherwise specified, all "MP-285" references refer equally to Model MP-285 and Model MP-285A controllers.

Controlling the MP-285 or MP-285A externally via computer is accomplished by sending commands between the computer and the equivalent connector on the rear of the controller: **SERIAL** (RS-232, 9-pin DSUB (MP-285 or MP-285A) or **USB** (MP-285A only).

8.1 The SERIAL (RS-232) Interface

RS-232-C, minimal 3-wire (Ground, Transmit, & Receive), 9-pin D-Shell connector (labeled "SERIAL" on the rear panel of the controller.

Table 8-1. Serial RS-232 (DB9 connector) port settings.

Property	Setting
Data ("Baud") Rate (bps (bits per second))	19200, **9600*,** 4800, 2400, 1200
Data Bits	**8**
Stop Bits	**1***, 1.5, 2
Parity	**None***, Even, Odd
Flow Control	**None**

*** Default**

NOTE: The data rate can be selected via the MP-285 controller's display and keypad. The default data rate (9600 bps) is recommended for most applications. The Parity can also be configured to be "ON", although the OFF ("None") setting is recommended and is the default.

8.2 The MP-285A USB Interface

Controlling the MP-285A externally via computer is accomplished by sending commands over the USB interface between the computer and the USB connector on the rear panel of the MP-285A controller/ROE. The USB device driver for Windows is downloadable from Sutter Instrument's web site (www.sutter.com). The MP-285A requires USB CDM (Combined Driver Model) Version 2.10.00 or higher. The CDM device driver for the MP-285A consists of two device drivers: 1) USB device driver, and 2) VCP (Virtual COM Port) device driver. Install the USB device driver first, followed by the VCP device driver. The VCP device driver provides a serial RS-232 I/O interface between a Windows application and the MP-285A. Although the VCP device driver is optional, its installation is recommended even if it is not going to be used. Once installed, the VCP can be enabled or disabled.

The CDM device driver package provides two I/O methodologies over which communications with the controller over USB can be conducted: 1) USB Direct (D2XX mode), or 2) Serial RS-232 asynchronous via the VCP device driver (VCP mode). The first method requires that the VCP device driver not be installed, or if installed, that it be disabled. The second method requires that the VCP be installed and enabled.

8.3 Virtual COM Port (VCP) Serial Port Settings

The following table lists the required RS-232 serial settings for the COM port (COM3, COM5, etc.) generated by the installation or enabling of the VCP device driver.

Table 2. MP-285A USB-VCP interface serial port settings.

Property	Setting
Data ("Baud") Rate (bps (bits per second))	9600
Data Bits	8
Stop Bits	1
Parity	None
Flow Control [1]	"Hardware" or RTS/CTS

The settings shown in the above table can be set in the device driver's properties (via the Device Manager if in Windows) and/or programmatically in your application.

8.4 Protocol and Handshaking

Most command sequences have a terminator: ASCII CR (Carriage Return; 13 decimal, 0D hexadecimal) (see the *MP-285 external-control commands* table). All commands return an ASCII CR (Carriage Return; 13 decimal, 0D hexadecimal) to indicate that the task associated with the command has completed. When the controller completes the task associated with a command, it sends ASCII CR back to the host computer indicating that it is ready to receive a new command. If a command returns data, the last byte returned is the task-completed indicator.

8.5 Command Sequence Formatting

Each command sequence consists of at least one byte, the first of which is the "command byte". Those commands that have parameters or arguments require a sequence of bytes that follow the command byte. No delimiters are used between command sequence arguments, and command sequence terminators are used in most cases. Although most command bytes can be expressed as ASCII displayable/printable characters, the rest of a command sequence must generally be expressed as a sequence of unsigned byte values (0-255 decimal; 00 – FF hexadecimal, or 00000000 – 11111111 binary). Each byte in a command sequence being transmitted to the controller must contain an unsigned binary value. Attempting to code command sequences as "strings" is not advisable. Any command data being returned from the controller must also be received and initially treated as a sequence of unsigned byte values. Groups of contiguous bytes can later be combined to form larger values, as appropriate (e.g., 2 bytes into 16-bit "word", or 4 bytes into a 32-bit "long" or "double word"). For the MP-285 controller, all axis position values (number of microsteps) are stored as "long" (or "signed long") 32-bit positive or negative values, and each is transmitted and received to and from the controller as four contiguous bytes.

[1] While the Flow Control property for the RS-232 DB9 interface is always set to "None", it must be set to "Hardware" or RTS/CTS signaling for the virtual serial port via the USB-VCP device driver.

8.6 Axis Position Command Parameters

All axis positional information is exchanged between the controller and the host computer in terms of microsteps. Conversion between microsteps and microns (micrometers) is the responsibility of the software running on the host computer (see *Microns/microsteps conversion* table for conversion factors).

Microsteps are stored as positive or negative 32-bit values ("long" (or optionally, "signed long") for C/C++; "I32" for LabVIEW).

The 32-bit value consists of four contiguous bytes, with a byte/bit-ordering format of Little Endian ("Intel") (most significant byte (MSB) in the first byte and least significant (LSB) in the last byte). If the platform on which your application is running is Little Endian, then no byte order reversal of axis position values is necessary. Examples of platforms using Little Endian formatting include any system using an Intel/AMD processor (including Microsoft Windows and Apple Mac OS X).

If the platform on which your application is running is Big Endian (e.g., Motorola PowerPC CPU), then these 32-bit position values must have their bytes reverse-ordered after receiving from, or before sending to, the controller. Examples of Big-Endian platforms include many non-Intel-based systems, LabVIEW (regardless of operating system & CPU), and Java (programming language/environment). MATLAB and Python (script programming language) are examples of environments that adapt to the system on which each is running, so Little-Endian enforcement may be needed if running on a Big-Endian system. Some processors (e.g., ARM) can be configured for specific endianess.

8.7 Microsteps and Microns (Micrometers)

All coordinates sent to and received from the controller are in microsteps. To convert between microsteps and microns (micrometers), use the following conversion factors (multipliers):

Table 8-1. Micron/microstep conversion.

System/Device	From/To Units	Conv. Factor
MP-285/M micromanipulator*	μsteps → μm	0.04
	μm → μsteps	25
MT-800 XY-Translator (**MT-2000** or **MT-2078** system):	μsteps → μm	0.05
	μm → μsteps	20

* Applicable also to **3DMS-285**, **MT-78, & MT-88** stages, & **MOM** & **SOM** microscope objectives

For accuracy in your application, type these conversion factors as "double" (avoid using the "float" type as it lacks precision with large values). When converting to microsteps, type the result as a 32-bit "long", "signed long", or "I32" integer. When converting to microns, type the result as "double" (64-bit double-precision floating-point values).

8.7.1 Ranges and Bounds

Table 8-1. Ranges and bounds.

Device	Axis	Len. (mm)	Origin	Microns (Micrometers (μm))	Microsteps (μsteps)
MP-285/M **	X, Y, & Z	25 mm	**COT***	**-12,500 – 12,500**	**-200,000–200,000**
			BOT	0 – 25,000	0 – 400,000
MT-800	X & Y	22 mm	**COT***	**-11,000 – 11,000**	**-140,800–140,800**
			BOT	0 – 22,000	0 – 281,600
	Z	25 mm	**COT***	**-12,500 – 12,500**	**-200,000–200,000**
			BOT	0 – 25,000	0 – 400,000

** Factory default.*

*** Applies also to 3DMS/M, MP-78/M, & MP-88/M stages, and SOM & MOM objective movers.*

NOTE: Origin is a physical position of travel that defines the center of the absolute position coordinate system (i.e., absolute position 0).
Physical Positions: BOT (Beginning Of Travel), COT (Center Of Travel), & EOT (End Of Travel).
In the MP-285, the Origin can be set to any physical position (factory default is COT).

NOTE: The MP-78/M (stage) and MT-800 (XY-translator) do not have a Z-axis motor. In either case, the controller's Z axis can be optionally connected to a motor of a different device (e.g., focus drive).

Travel Speed: The following table shows the selectable travel speeds for single-, double-, and triple-axis movements for supported devices using orthogonal move commands.

Table 2. Travel speeds.

Resolution	Speed Range (microns/sec)
Low (coarse: 0.2 μm/μstep (10 μsteps/step))	0 – 3000*
High (fine: 0.04 μm/μstep (50 μsteps/step))	0 – 1310

** CAUTION: Although the absolute maximum microns/sec. speed allowable in low (coarse) resolution is 6,550, it is essential that a speed no higher than 3,000 be used with the MP-285A model controller.*

8.8 External Control Commands

Each command consists of one byte. Those commands that have parameters or arguments require a sequence of bytes that follow the command byte. No delimiters are used between command sequence arguments. Every command and command sequence ends with a terminator byte containing an ASCII CR (13 decimal, 0D hexadecimal).

8.8.1 Get Current Position ('c') Command

This command is used to obtain the current position (X, Y, & Z coordinates) of the manipulator or stage. The command sequence consists of 2 bytes as shown in the following table. The data returned consists of 13 bytes: 12 bytes containing X, Y, & Z position values in microsteps (4 bytes each), followed with the task-complete indicator (1 byte).

Table 8-3. Get Current Position (‘c’) command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Get Current Position (‘c’)	Tx	All	2	0	99	63	0110 0011	0099		‘c’	Returns the current positions (μsteps) of X, Y, & Z axes.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		13		Three 4-byte (32-bit) values (current positions in μsteps of X, Y, & Z), + 1 byte for completion indicator.						
				0 (4)							X–axis position
				4 (4)							Y–axis position
				8 (4)							Z–axis position
				12	13	0D	0000 1101			<CR>	Task-completion indicator

8.8.1 Move to Specified Position (‘m’) Command

This command instructs the controller to move all three axes to the position specified. The command sequence consists of 14 bytes: Command byte followed by three sets of four bytes containing position information in microsteps for X, Y, and Z, and ending with the terminator. Return data consists of 1 byte (task-complete indicator), which occurs after the move is complete.

Table 8-4. Move to specified position (‘m’) command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Move to Specified Position (‘m’)	Tx	All	14	0	109	6D	0110 1100	0109		‘m’	Moves to specified position (μsteps) (see *Ranges* table)
				1 (4)							X μsteps
				5 (4)							Y μsteps
				9 (4)							Z μsteps
				13	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

8.8.1 Setting Resolution and Velocity (‘V’) Command

This command instructs the controller to move all three axes to the position specified. The command sequence consists of 4 bytes: Command byte, followed by 2 bytes containing resolution and velocity information, and ending with the terminator. Return data consists of 1 byte (task-complete indicator).

Table 8-5. Set resolution and velocity ('V') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Set Velocity & Resolution ('V')	Tx	All	4	0	86	56	0101 0110	0086		'V'	Command (Note: Uppercase 'V') (see *Resolution & Velocity* note)
				1 (2)	0 – 4095 or 32, 768 – 35, 768	0000 – 051E or 8000 – 8BB8	00000000 00000000 – 00000101 00011110 or 10000000 00000000 – 10001011 10111000	One **unsigned** short (16-bit) integer (2 bytes) containing both resolution and velocity values. MSB (Bit 15) contains resolution setting; remaining bits (14–0) contains velocity value. **Resolution (Bit 15)**: 0 = Low (coarse: 0.2 µm/µstep (10 µsteps/step)) 1 = High (fine: 0.04 µm/µstep (50 µsteps/step)) **Velocity (Bits 14-0)**: Low Res.: 0 – 6550 (MP-285) or 3000 (MP-285A) µm/sec High Res.: 0 – 1310 µm/sec			
				3	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

The Set Resolution & Velocity ('V') command unsigned 16-bit value can be easily composed mathematically using the following formula:

unsigned short ResSpeed = (Res * 0x8000) + Speed

where "ResSpeed" is the final unsigned 16-bit value (Little Endian bit order), "Res" is the resolution (0 for Low; 1 for High), 0x8000 hexadecimal (Base 16) (32,768 decimal) as a multiplier positions the resolution (0 or 1) to Bit 15 (the high order bit), and then the "Speed" value is simply added to occupy Bits 14 through 0. The "unsigned short" is a C/C++ type definition that ensures that "ResSpeed" is a 16-bit variable that holds only positive values.

8.8.1 Set Origin ('o') Command

This command sets the Origin position along the line of travel for each axis. The command sequence consists of 2 bytes: Command byte, followed by the terminator. Return data consists of 1 byte (task-complete indicator).

Table 8-6. Set origin ('o') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Set Origin ('o')	Tx	All	2	0	111	6F	0110 1111	0111		'o'	Sets the Absolute Origin to the current position.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

The Origin is set to a physical position of travel to define absolute position 0. The factory default physical Origin position is center of travel (COT). This means that all higher positions (towards end of travel (EOT)) are positive values, and all lower positions (towards beginning of travel (BOT)) are negative values. The Origin can also be changed (via the controller's front panel display/keypad.

CAUTION: When changing the Origin from its factory default, it is not possible to obtain the new Origin's physical position via an external control command. If changing the Origin's physical position via the external control 'o' command, it is recommended that the external application keep careful track of all Origin changes, and automatically adjust its view of the absolute position coordinate system according to the current Origin's physical position.

8.8.1 Set Absolute Mode ('a') Command

This command sets the nature of the positional values specified with the Move ('m') command as absolute positions as measured from the point of origin (Position 0). The command sequence consists of 2 bytes: Command byte, followed by the terminator. Return data consists of 1 byte (task-complete indicator).

Table 8-7. Set Absolute Mode ('a') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Set Absolute Mode ('a')	Tx	All	2	0	97	61	0110 0001	0097		'a'	Sets movement mode to Absolute. Each 'm'-command axis value represents an absolute position. (Note: No VFD update)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

Current position (via the 'c' command) reports absolute positions of each axis. Moving to a new position (via the 'm' command) is specified with absolute position values when in Absolute mode ('a' command) or with relative values (relative to the current position) when in Relative mode ('b' command).

CAUTION: In an external control program, care should be taken to ensure that the Absolute/Relative mode state be updated upon a mode change and kept track of, as it is not possible to obtain the current mode from the controller. In addition, any computational relative positioning made in an external program while in Absolute mode must ensure that relative positions are accurately translated to correct absolute positions before initiating a move command.

8.8.1 Set Relative Mode ('b') Command

This command sets the nature of the positional values specified with the Move ('m') command as relative positions as measured from the current position (absolute position returned by the Get Current Position ('c') command). The command sequence consists of 2 bytes: Command byte, followed by the terminator. Return data consists of 1 byte (task-complete indicator).

Table 8-8. Set Relative Mode (‘b’) command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Set Relative Mode (‘b’)	Tx	All	2	0	98	62	0110 0010	0098		‘b’	Sets movement mode to Relative. Each ‘m’-command axis value represents a position relative to the current position. (Note: No VFD update)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

Current position (via the ‘c’ command) reports absolute positions of each axis. Moving to a new position (via the ‘m’ command) is specified with absolute position values when in Absolute mode (‘a’ command) or with relative values (relative to the current position) when in Relative mode (‘b’ command).

CAUTION: In an external control program, care should be taken to ensure that the Absolute/Relative mode state be updated upon a mode change and kept track of, as it is not possible to obtain the current mode from the controller. In addition, any computational relative positioning made in an external program while in Absolute mode must ensure that relative positions are accurately translated to correct absolute positions before initiating a move command.

8.8.1 Interrupt Move (^C) Command

This command interrupts and stops a move in progress that originally initiated by the Move (‘m’) command. The command sequence consists of 1 byte: Command byte (no terminator). Return data consists of 1 byte if movement is not in progress, or 2 bytes (‘=’ (move-in-progress indicator) and task-complete indicator.

Table 8-9. Interrupt Move (^C) command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Interrupt Move (^C)	Tx	All	1	0	3	03	0000 0011	0003	^C	**<ETX>**	Interrupts an ‘m’-command initiated move in progress
	Rx		1	0	61	3D	0011 1011	0061		‘=’	Move in progress indicator
			2	0	13	0D	0000 1101	0013		<CR>	Task-completion indicator
	Rx			0	13	0D	0000 1101	0013		<CR>	Task-completion indicator (movement was not in progress)

8.8.1 Refresh VFD (‘n’) Command

This command refreshes the VFD (Vacuum Fluorescent Display) of the controller. The command sequence consists of 2 bytes: Command byte and terminator. Return data consists of 1 byte: task-complete indicator.

Table 8-10. Refresh VFD ('n') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Refresh VFD[2] Display ('n')	Tx	All	2	0	110	6E	0110 0110	0101		'n'	Refreshes the controller's display (X, Y, & Z coordinates only)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	0	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator

8.8.1 Reset Controller ('r') Command

This command resets the controller. The command sequence consists of 2 bytes: Command byte and terminator. Return data consists of 1 byte: task-complete indicator.

Table 8-11. Reset controller ('r') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Reset Controller ('r')	Tx	All	2	0	114	72	0111 0010	0114		'r'	Resets the controller.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	0	13	0D	0000 1101	0013		<CR>	Task-completion indicator

8.8.1 Get Status ('s') Command

This command gets status information from the controller and returns it in fixed-sized block of data. The command sequence consists of 2 bytes: Command byte and terminator. Return data consists of 33 bytes: 32 bytes of information and task-complete indicator.

Table 8-12. Get Status ('s') command.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (Len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Get Status ('s')	Tx	All	2	0	115	73	0111 0011	0115		's'	Returns status information
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		33	0							Status data – see *Status Data Structure* table.
				32	13	0D	0000 1101	0013		<CR>	Task-completion indicator

Please refer to section **Status Return Data** for information on the status data structure

8.9 Notes

1. **Task-Complete Indicator:** All commands will send back to the computer the "Task-Complete Indicator" to signal the command and its associated function in controller is

[2] Vacuum Florescence Display

complete. The indicator consists of one (1) byte containing a value of 13 decimal (0D hexadecimal), and which represents an ASCII CR (Carriage Return).

2. **Intercommand Delay:** A short delay (usually around 2 ms) is recommended between commands (after sending a command sequence and before sending the next command).
3. **Clearing the I/O Send & Receive Buffers:** Clearing (purging) the transmit and receive buffers of the I/O port immediately before sending any command is recommended. Note that this clearing of the buffers affects only the computer-side I/O; it does not (necessarily) clear the buffers on the controller side, requiring, when necessary, to reset/power-cycle the controller. Following the rules described will generally avoid problems with getting garbage data in the I/O buffers of both the computer and controller (i.e., using exact number of bytes for both command sequences and return data (as per the Commands table), never sending a command before the previous command is finished with its task, etc.).
4. **Positions in Microsteps and Microns:** All positions sent to and received from the controller are in microsteps (μsteps). See Microns/microsteps conversion table) for conversion between μsteps and microns (micrometers (μm)).

   Declaring position variables in C/C++:
   ```
   /* current position for X, Y, & Z */
   long   cp_x_us, cp_y_us, cp_z_us;  /* microsteps */
   double cp_x_um, cp_y_um, cp_z_um;  /* microns */
   /* specified (move-to) position for X, Y, & Z */
   long   sp_x_us, sp_y_us, sp_z_us;  /* microsteps */
   double sp_x_um, sp_y_um, sp_z_um;  /* microns */
   ```
   Use the same convention for other position variables the application might need.

   Declaring the microsteps/microns conversion factors in C/C++:
   ```
   /* conversion factors for MP-285/M based config. */
   double us2umCF = 0.04; /* microsteps to microns */
   double um2usCF = 25;   /* microns to microsteps */
   /* conversion factors for MT-800 config. */
   double us2umCF = 0.05; /* microsteps to microns */
   double um2usCF = 20;   /* microns to microsteps */
   ```
   NOTE: In an MP-285A-based system configured for an MP-78 stage or MT-800-based XY translator (MT-2078), the Z axis may be configured for different conversion factors (e.g., if Z is wired to a separate device such as a focus drive). In such cases, make sure the appropriate microsteps/microns conversion factors are used for Z while using the standard factors for X and Y.

   Converting between microsteps and microns in C/C++:
   ```
   /* converting X axis current position */
   cp_x_um = cp_x_us * us2umCF;  /* microsteps to microns */
   cp_x_us = cp_x_um * um2usCF;  /* microns to microsteps */
   ```
   Do the same for Y and Z, and for any other position sets used in the application.

5. **Ranges and Bounds:** See Ranges and Bounds table for exact minimum and maximum values for each axis of each compatible device that can be connected. All move commands include positive or negative values for positions. All positions are absolute as measured from the Origin position as set in the controller for all axes of the attached device. The

factory default Origin position is the physical center position (between beginning of travel and end of travel) of the device. In application programming, it is important that positional values be checked (>= minimum and <= maximum) to ensure that a position is within the bounds of travel before it is sent to the controller.

6. **Absolute Positioning System Origin:** The Origin is set to a physical position of travel to define absolute position 0. The factory default physical Origin position is center of travel (COT). This means that all higher positions (towards end of travel (EOT)) are positive values, and all lower positions (towards beginning of travel (BOT)) are negative values. The Origin can be changed (via the controller's front panel display/keypad or via the Origin ('o') command sent from an external program.

   CAUTION: When changing the Origin from its factory default, it is not possible to obtain the new Origin's physical position via an external control command. If changing the Origin's physical position via the external control 'o' command, it is recommended that the external application keep careful track of all Origin changes, and automatically adjust its view of the absolute position coordinate system according to the current Origin's physical position.

7. **Absolute vs. Relative Positioning:** Current position (via the 'c' command) report absolute positions of each axis. Moving to a new position (via the 'm' command) is specified with absolute position values when in Absolute mode ('a' command) or with relative values (relative to the current position) when in Relative mode ('b' command).

   CAUTION: In an external control program, care should be taken to ensure that the Absolute/Relative mode state be updated upon a mode change and kept track of, as it not possible to obtain the current mode from the controller. In addition, any computational relative positioning made in an external program while in Absolute mode must ensure that relative positions are accurately translated to correct absolute positions before initiating a move command.

8. **Position Value Typing:** All positions sent and received to and from the controller are in microsteps and consist of 32-bit integer values (four contiguous bytes). Position values can be either positive or negative, so the type must be "signed". Although each positional value is transmitted to, or received from, the controller as a sequence of four (4) contiguous bytes, for computer application computational and storage purposes each should be typed as a signed integer ("long" or "signed long" in C/C++; "I32" in LabVIEW, etc.). Note that in Python, incorporating the optional NumPy package brings robust data typing like that used in C/C++ to your program, simplifying coding and adding positioning accuracy to the application.

9. **Position Value Bit Ordering:** All 32-bit position values transmitted to, and received from, the controller must be bit/byte-ordered in "Little Endian" format. This means that the least significant bit/byte is last (last to send and last to receive). Byte-order reversal may be required on some platforms. Microsoft Windows, Intel-based Apple Macintosh systems running Mac OS X, and most Intel/AMD processor-based Linux distributions handle byte storage in Little-Endian byte order so byte reordering is not necessary before converting to/from 32-bit "long" values. LabVIEW always handles "byte strings" in "Big Endian" byte order irrespective of operating system and CPU, requiring that the four bytes containing a microsteps value be reverse ordered before/after conversion to/from a multibyte type value (I32, U32, etc.). MATLAB automatically adjusts the endianess of multibyte storage entities to that of the system on which it is running, so explicit byte

reordering is generally unnecessary unless the underlying platform is Big Endian. If your development platform does not have built-in Little/Big Endian conversion functions, bit reordering can be accomplished by first swapping positions of the two bytes in each 16-bit half of the 32-bit value, and then swap positions of the two halves. This method efficiently and quickly changes the bit ordering of any multibyte value between the two Endian formats (if Big Endian, it becomes Little Endian, and if Little Endian, it becomes then Big Endian).

10. **Travel Lengths and Durations:** "Move" commands might have short to long distances of travel. If not polling for return data, an appropriate delay should be inserted between the sending of the command sequence and reception of return data so that the next command is sent only after the move is complete. This delay can be auto calculated by determining the distance of travel (difference between current and target positions) and rate of travel. This delay is not needed if polling for return data. In either case, however, an appropriate timeout must be set for the reception of data so that the I/O does not time out before the move is made and/or the delay expires.
11. **Z-Axis Usage in 2-Axis Systems:** On an MT-800, MP-78, or MP-88 system, the Z axis can be used as a focus drive, with a conversion factor that may be custom according to the make and model of the microscope being used.
12. **Setting Resolution & Velocity:** The Set Resolution & Velocity ('V') command unsigned 16-bit value can be easily composed mathematically using the following formula:

    *unsigned short ResSpeed = (Res * 0x8000) + Speed*
13. where "ResSpeed" is the final unsigned 16-bit value (Little Endian bit order), "Res" is the resolution (0 for Low; 1 for High), 0x8000 (32,768 decimal) as a multiplier positions the resolution (0 or 1) to Bit 15 (the high order bit), and then the "Speed" value is simply added to occupy Bits 14 through 0. The "unsigned short" is a C/C++ data type definition that ensures that "ResSpeed" is a 16-bit variable that holds only positive values.
14. **Move Interruption:** A command should be sent to the controller only after the task of any previous command is complete (i.e., the task-completion terminator (CR) is returned). One exception is the "Interrupt Move" (^C) command, which can be issued while a command-initiated move is still in progress.

8.10 Status Return Data

Table 8-1. Status data structure (as returned by Get Status ('s') command).

Offset	Length	Name	Description							
			Bit	**Name**	**Description**	**Values**				
						Binary-Coded Decimal (BCD)				**Dec. Digit**
						3	**2**	**1**	**0**	
0	8 bits	**FLAGS**	**0-3**	**SETUP #**	Currently loaded setup number coded in BCD (decimal digit 0-9)	0	0	0	0	0
						0	0	0	1	1
						0	0	1	0	2
						0	0	1	1	3
						0	1	0	0	4
						0	1	0	1	5

Offset	Length	Name	Description							
						0	1	1	0	6
						0	1	1	1	7
						1	0	0	0	8
						1	0	0	1	9
						1 (Set)		**0 (Clear)**		
			4	**ROE_DIR**	Last ROE direction	Negative		Positive		
			5	**REL_ABS_F**	Display origin	Absolute		Relative		
			6	**MODE_F**	Manual mode flag	Continuous		Pulse		
			7	**STORE_F**	Setup condition	Stored		Erased		
1	Byte	**UDIRX**	User-defined values for motor axis directions. Valid values: 0-5							
2	Byte	**UDIRY**								
3	Byte	**UDIRZ**								
4	Word	**ROE_VARI**	Microsteps per ROE click							
6	Word	**UOFFSET**	User-defined period start value							
8	Word	**URANGE**	User-defined period range							
10	Word	**PULSE**	Number of microsteps per pulse							
12	Word	**USPEED**	Adjusted pulse speed microsteps per sec.							
14	Byte	**INDEVICE**	Input device type							
15	8 bits	**FLAGS_2**	**Bit**	**Name**	**Description**	**1 (Set)**		**0 (Clear)**		
			0	**LOOP_MODE**	Program loops	Do loops		Execute once		
			1	**LEARN_MODE**	Learn mode status	Learning now		Not learning		
			2	**STEP_MODE**	Resolution (microsteps/step)	50		10		
			3	**SW2_MODE**	Joystick side button	Enabled		Disabled		
			4	**SW1_MODE**	Enable FSR/Joystick	Enabled		Cont/Pulse (keypad)		
			5	**SW3_MODE**	ROE switch	Enabled		Disabled		
			6	**SW4_MODE**	Switches 4 & 5	Enabled		Disabled		
			7	**REVERSE_IT**	Program sequence	Reverse		Normal		
16	Word	**JUMPSPD**	"Jump to max at" speed							
18	Word	**HIGHSPD**	"Jumped to" speed							
20	Word	**DEAD**	Dead zone, not saved							
22	Word	**WATCH_DOG**	Programmer's function (analog input for overload protection)							
24	Word	**STEP_DIV**	Microns ←→ Microsteps conversion factor. See Note 2.							
26	Word	**STEP_MUL**	Microns ←→ Microsteps conversion factor. See Note 2.							
28	Word	**XSPEED**	Velocity (microns/sec., Bits 14 - 0) & resolution (0 or 1, Bit 15). See Note 3.							
30	Word	**VERSION**	Firmware version. See Note 4.							
32	Byte		End of received data terminator (ASCII CR (13 decimal or 0D hexadecimal))							

NOTES:

1. All values are stored in Little-Endian bit order. All byte values are ordered Bit 7 through Bit 0. All “word” (16-bit) values are ordered Bit 15 through Bit 0. To reverse the bit order of word values to Big Endian, swap positions of both bytes (least significant byte becomes most significant and most significant becomes least significant).
2. **STEP_DIV and STEP_MUL:** Both contain 16-bit values used as factors for converting positional values between microns and microsteps. See the Microns/microsteps conversion factors table for what the values need to be for conversions. Position values in microns are typically stored in “double” (for double-precision floating-point) data type variables, while positions in microsteps are stored as 32-bit signed integer variables data-typed as “signed long”. “double” and “signed long” (or just “long”) are C/C++ data types. Both conversion factors as copied or derived from STEP_DIV and STEP_MUL should be stored in “double” data type variables so they can be used as multipliers to facilitate accurate conversions between double-precision microns and 32-bit integer microsteps.
   In the C/C++ examples below “status_data_block” is the address of the data returned by the ‘s’ command, and “double” is the data type for double-precision floating-point variables.

```
/* define both conversion factors as double-precision floating point
variables */
double um2usCF, us2umCF;
```

   MP-285: STEP_DIV contains the microsteps/micron conversion factor. STEP_MUL contains the microns/microstep conversion factor (the reciprocal of STEP_DIV) * 100.

```
/* Get microsteps/microns conversion factor */
us2umCF = (double)((unsigned short)status_data_block[24]);
/* Get microns/microstep conversion factor */
um2usCF = (double)((unsigned short)status_data_block[26]) / 100;
```

   For example, if the controller is configured for an MP-285/M micromanipulator or derived device (3DMS-285 or MP-78 stage, or MOM or SOM objective mover), then STEP_DIV contains 25 and STEP_MUL contains 4 (0.04 (the actual reciprocal of STEP_DIV * 100).

   If the controller is configured for an MT-800 XY Translator, then STEP_DIV contains 20 and STEP_MUL contains 5 (0.05 * 100).

   MP-285A: Both STEP_DIV and STEP_MUL contain the distance travelled by ten microsteps, expressed in nanometers (where 1 nanometer = 0.001 micron). To get the length of one microstep in nanometers, divide the field’s value by 10 and then again by 1000. To get the number of microsteps required to move one micron, take the reciprocal of the length of one microstep (in microns).

```
/* microns/microstep conversion factor: Divide by 10 for
nanometers, then by 1000 for microns */
um2usCF = (double)((unsigned short)status_data_block[26]) / 10000;
/* microsteps/micron: Reciprocal of microns/microstep */
us2umCF = 1 / um2usCF;
```

   For example, if the controller is configured for an MP-285/M micromanipulator or derived device (3DMS-285 or MP-78 stage, or MOM or SOM objective mover), then both fields contain 400, meaning 10 microsteps = 400 nanometers. The length of one microstep is therefore 400/10 = 40 nanometers, or 400/10000 = 0.04 microns. The number of microsteps needed to move one micron is 1/0.04 = 25 microsteps. Thus, the conversion factors for the MP-285/M are 0.04 microns/microstep and 25 microsteps/micron.

3. **XSPEED**: Contains an unsigned 16-bit value ("unsigned short" or "WORD") with both the Resolution (Low or High) and the speed (microns/sec) encoded within it. The Resolution is stored in the high-order bit (Bit 15), and the speed is stored in the remaining bits (Bits 14 through 0). Extracting both values can be done in the following way (C/C++):

```
/* "status_data_block" is the name of the address of the data
returned by the 's' command. "unsigned" is a data type prefix that
indicates positive numbers only */
unsigned short XSPEED, Speed, B15; /* 16-bit variables */
unsigned char Res;                 /* 8-bit variable */
/* read 16 bits from "status_data_block" at Index (offset) 28 */
XSPEED = (unsigned short)status_data_block[28];
Res    = 0;      /* assume Low Res */
Speed  = XSPEED; /* assume Low Res speed */
B15    = 0x8000; /* Bit 15 position value (32758 dec.) */
if (XSPEED >= B15)  /* if High Res . . . */
{
    Res   = 1;             /* set Res to High */
    Speed = (XSPEED - B15) /* extract High Res speed */
}
```

VERSION: Contains the version of the controller's firmware * 100. To extract the version, divide by 100 (e.g., 302 / 100 = 3.02 (3 is the major version number and 02 is the minor version)).

```
/* Get the version as a 16-bit positive integer value */
unsigned short VERSION =
            (unsigned short)status_data_block[30];
/* major version integer */
unsigned short ver_major = VERSION / 100;
/* minor version integer */
unsigned short ver_minor = VERSION % 100;
/* full version floating-point value */
float Ver = ((float)VERSION) / 100;
```

8.11 Error Reporting

Errors are reported as ASCII numerals to the host and on the controller display at the upper right-hand corner. These are:

Table 8-1. Error codes.

Code	Value			Error	Description
ASCII Char.	Dec.	Hex	Binary		
0	48	30	00110000	**SP Over-run**	The previous character was not unloaded before the latest was received
1	49	31	00110001	**Frame Error**	A valid stop bit was not received during the appropriate time period
2	50	32	00110010	**Buffer Overrun**	The input buffer is filled, and CR has not been received
4	51	34	00110011	**Bad Command**	Input cannot be interpreted – command byte not valid

8	56	38	00111000	**Move Interrupted**	A requested move was interrupted by input on the serial port. This code is ORed with any other error code. The value normally returned is “<”, i.e., ‘8’ (38h) ORed with ‘4’ (34h) = ‘<’ (3Ch). ‘4’ is reported on the vacuum fluorescent display.
					
					`'8' | '0' = '8'  (38h | 30h = 38h)`
					`'8' | '1' = '9'  (38h | 31h = 39h)`
					`'8' | '2' = ':'  (38h | 32h = 3Ah)`
					`'8' | '3' = ';'  (38h | 33h = 3Bh)`
					`'8' | '4' = '<'  (38h | 34h = 3Ch)`

8.12 Setting up for Serial Communication

First, use the 9-pin serial port cable provided with the MP-285 to connect the “serial port” of your computer to that of the MP-285 controller. Next configure your terminal emulator (e.g., HyperTerminal in Microsoft Windows (9X and above) to the following settings (or their equivalent):

- TTY mode
- Echo typed characters locally only (do not echo input to the computer serial port back to the controller)
- Data (“Baud”) Rate: 9600 bps
- 8 data bits, no parity, 1 stop bit
- COM port - set to the port to which you have connected the MP-285 controller
- Flow Control: Set to “None” if using the DB9 SERIAL port (9-pin serial cable) or set to “Hardware” if using the USB-VCP port (USB cable) (MP-285A only).

A simple test can now be made to confirm that the RS-232 cable is properly connected, and the computer terminal emulator is properly configured. The “o” command can serve this purpose, but it is important to know that **THIS COMMAND RESETS THE ABSOLUTE ORIGIN OF THE MP-285 CONTROLLER!** It is possible to minimize the relative effect of this command by moving the manipulator to a very short distance from the **ABSOLUTE ORIGIN** (e.g., X=0.04 µm, Y=0.04 µm, Z=0.04 µm) before issuing the "o" command from the remote computer. Now, type "o" (followed by a RETURN). The MP-285 CONTROLLER display should reset to X=0.00 µm, Y=0.00 µm, Z=0.00 µm. There may also be a change in the overall appearance of the display. The original display configuration can be restored by entering an "n" (followed by a RETURN) from the computer keyboard or by pressing the reset button on the front panel of the MP-285 CONTROLLER.

You may also want to try the serial interface test "SIO test" that is built into the MP-285 controller. For directions on how to use this function, see the subheading "SIO test [PRGM\Setup\Utilities\SIO test]" in the Controller Configuration section of the manual.

If these tests of the serial port connection fail, try another COM port assignment and/or recheck your computer's configuration to make certain that they conform to the above specifications.

8.13 Interface Programs

Now that you have the cable connected and port configured, you may want to try one of the following:

- Try the Visual C++ MP-285 interface program supplied with your manipulator.
- Run one of the simple QuickBasic or PowerBASIC program fragments on the following pages.
- If you are on a non-Wintel platform, you can download sample programs from the Sutter Instrument web site at http://www.sutter.com/.

The programs and code fragments are provided only as examples of how the data stream can be handled to and from the controller and are not mature software packages. The NEW285.bas PowerBASIC code found in the Software section of the Sutter Instruments website can be useful as a starting point for developing a more advanced PowerBASIC interface or software in MODULA-2 and C programming languages, as well.

Further notes:

- The approach used for handling the data stream in the PowerBASIC programs uses UNIONs. The UNION allows the bytes in the data stream to be simultaneously "TYPED" as LONG INTEGERs (for display of and operations on the data from the MP-285 controller) or as STRINGs of BYTES (for sending coordinates back to the MP-285 controller).
- Sutter Instrument Company welcomes your feedback about any successes or problems that you may experience in developing serial communication software for the MP-285. With your permission, we will put your comments and suggestions on our website for the benefit of other scientists who might gain from your efforts.

```
          'QuickBASIC "getit" ROUTINE

'This program is written in QuickBasic 4.5. It will get the position
'coordinates from the MP-285, (in usteps), convert them to um's and
```

```
'then print them to the screen.

CLS

'First, open the appropriate COM port (without a handshaking protocol)
OPEN "COM2:9600,N,8,1,CD0,CS0,DS0" FOR RANDOM AS #1

'Next send the "get position" command and a CR to the MP-285
PRINT #1, "c";
PRINT #1, CHR$(13);

'Finally, for each axis convert 4 bytes from the INPUT BUFFER into
'a LONG INTEGER using the CVL function, multiply that value by .04 to
'convert it from usteps to um's and then PRINT it to the screen. All
'three of these steps are contained in one line of code below.
PRINT "MP-285 COORDINATES"
PRINT USING "X = +#####.## um "; .04 * (CVL(INPUT$(4, 1)))
PRINT USING "Y = +#####.## um "; .04 * (CVL(INPUT$(4, 1)))
PRINT USING "Z = +#####.## um "; .04 * (CVL(INPUT$(4, 1)))
'This line dumps the last byte (a CR) from the buffer
endchar$ = INPUT$(1, 1)

CLOSE #1
END

          'QuickBASIC "goto" ROUTINE

'This program is written in QuickBASIC 4.5. It will prompt you to
'enter the coordinates (in um's) to which you want the manipulator to
'move, convert the values to usteps and then send them to the MP-285

CLS

'First, open the appropriate COM port (without a handshaking protocol)
OPEN "COM2:9600,N,8,1,CD0,CS0,DS0" FOR RANDOM AS #1

'The next step will prompt you to input the coordinates in um's
INPUT "X = ", X
INPUT "Y = ", Y
INPUT "Z = ", Z

'This step will send the MOVE command to the MP-285 followed by a 13 byte
'data stream. The first 12 bytes in the data stream are three 4-byte
'strings representing the LONG INTEGERS entered above.
'Those values are reduced here
'to bytes by the MKL$ function after they have been converted
'from um's to
'usteps (* 25). The last byte is a CR.
PRINT #1, "m";
PRINT #1, MKL$(X * 25);
PRINT #1, MKL$(Y * 25);
PRINT #1, MKL$(Z * 25);
```

```
PRINT #1, CHR$(13);
CLOSE #1
END


 'PowerBASIC "getit" ROUTINE (Pbgetit.bas)

'This program is written in PowerBASIC 3.0. It will get the position
'coordinates from the MP-285 (in usteps), convert them to um's and
'then print them to the screen.

CLS
'Open the appropriate COM port (without a handshaking protocol).
OPEN "COM2:9600,N,8,1,CD0,CS0,DS0" FOR RANDOM AS #1

UNION fourbytes                'creates a data "TYPE" to allow the
  longfield AS LONG            '4 bytes describing an axis' position
  fourstring AS STRING * 4     'to be arrayed as a "LONG" integer and
END UNION                      'as a "Fixed-length" string called
                               'longfield and fourstring, respectively

DIM XVAL AS fourbytes          'dimensions the arrays to
DIM YVAL AS fourbytes          'which the byte values of the
DIM ZVAL AS fourbytes          'coordinates will be written and
                               'declares their "TYPE" (a UNION
                               'called fourbytes in this instance)

PRINT #1, "c"                  'requests the coordinates
PRINT #1, CHR$(13)

XVAL.fourstring = INPUT$(4,#1)  'reads the coordinates (4 bytes
YVAL.fourstring = INPUT$(4,#1)  'for each) into the appropriate
ZVAL.fourstring = INPUT$(4,#1)  'array

PRINT "MP-285 COORDINATES"
PRINT USING "X = +#####.## um ";XVAL.longfield * .04 'converts the ustep
PRINT USING "Y = +#####.## um ";YVAL.longfield * .04 'values to um's and
PRINT USING "Z = +#####.## um ";ZVAL.longfield * .04 'prints to screen

endchar$ =  INPUT$(1,#1)       'dumps the last byte (a CR) from the
                               'buffer


 'PowerBASIC "goto" ROUTINE (Pbgoto.bas)

'This program is written in PowerBASIC 3.0. It will prompt you to enter
'the coordinates (in um's) to which you want the manipulator to move,
'convert those values to usteps and then send them to the MP-285.

CLS
'Open the appropriate COM port (without a handshaking protocol).
OPEN "COM2:9600,N,8,1,CD0,CS0,DS0" FOR RANDOM AS #1
```

```
UNION fourbytes                  'a data "TYPE" created to allow the
  longfield AS LONG              '4 bytes describing an axis' position
  fourstring AS STRING * 4       'to be arrayed as a "LONG" integer and
END UNION                        'as a "Fixed-length" string called
                                 'longfield and fourstring, respectively

DIM XVAL AS fourbytes            'dimensions the arrays to
DIM YVAL AS fourbytes            'which the coordinates in the
DIM ZVAL AS fourbytes            'buffer will be written and
                                 'declares their "TYPE" (a UNION
                                 'called fourbytes in this instance)

INPUT "X = ",X                   'prompts you to input the coordinates
INPUT "Y = ",Y                   '(in um's) of the position to which
INPUT "Z = ",Z                   'you want the manipulator to move

XVAL.longfield = X * 25          'converts the values entered from
YVAL.longfield = Y * 25          'um's to usteps and enters them into
ZVAL.longfield = Z * 25          'LONG INTEGER array named longfield

PRINT #1, "m";                   'sends the MOVE command to the MP-285
PRINT #1, XVAL.fourstring;       'followed by the 3 strings of bytes
PRINT #1, YVAL.fourstring;       '(4 each) that represent the LONG
INTEGERS
PRINT #1, ZVAL.fourstring;       'put into the longfield array in the last
PRINT #1, CHR$(13);              'step (a CR ends the data stream)
```

8.14 The Visual C++ MP-285 Interface

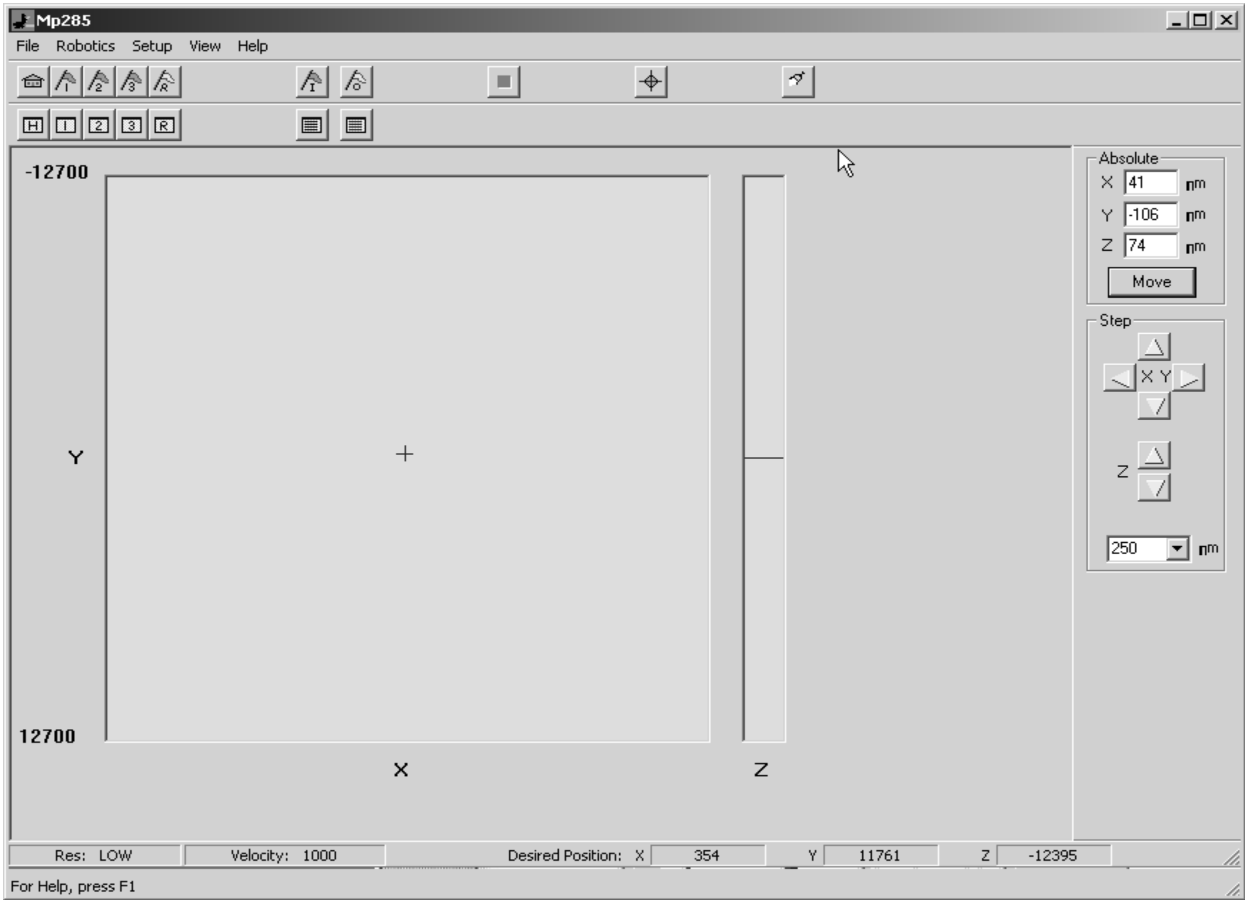
8.14.1 Preface

Sutter Instrument Company has created a Visual C++ program with an object-oriented interface to control the MP-285 micromanipulator via a host computer. This brief introduction is intended for users who are already familiar with operating the MP-285.

The software is designed for executing micromanipulator movements and controlling various functions of the MP-285 via a serial interface. If you wish to customize or modify the interface, the source code is available upon request.

8.14.2 Installing the Visual C++ PC Controller

1. Windows 95 or greater should be installed and running on a PC, 486DX or faster.
2. Insert the disk label MP-285 Controller Program into A: drive.
3. Click "Start", "Run...", and type "a:\mp285.exe" then click "OK"
4. Follow the setup directions as they appear. An icon will be created in Start\Programs
5. Once setup is complete, make sure the MP-285 is connected properly to a serial port on the computer.
6. Turn on the MP-285 first, then go to Start\Programs and click on the MP-285 icon.
7. Select the correct COM port and baud rate and click OK.
8. The MP-285 workspace should appear:



8.14.3 MP-285 PC Controller User Interface

On the right-hand side of the window, the **Absolute** box with three panels displays the current location of the manipulator. Each panel corresponds to one of the three axes and indicates the position, in µm, relative to the absolute position read from the controller when the software began communication. In each box, you can manually specify the next position you want to move to along each axis. The move is then performed when you click on the "move" button below.

If you want to determine the function of a button, simply position the mouse cursor over the button and a pop-up box will appear describing the button's function.

The **Step** box allows you to move the manipulator in specified increments along each axis. The drop-down box allows you to specify how many µm the manipulator moves each time you click on one of the direction buttons.

The five buttons on the left side of the top bar send the manipulator to the **Home** position, **Position 1**, **Position 2**, **Position 3**, and **Return to previous position** respectively. Directly beneath each of these buttons is a corresponding button used to **Define** where 'Home', 'Position 1', 'Position 2', etc. are in the field of possible positions.

The next two buttons allow quick access to two pre-programmed robotic movement routines. The corresponding buttons beneath are used to define the routines.

The **Stop** button to the right becomes red only when it is active, i.e. when the manipulator is in the process of moving to another position. This button allows you to stop the movement of the manipulator.

The button to the right of the stop button is used to **Update Display Coordinates** after the manipulator has received input commands from a manual input device such as the Rotary Optical Encoder (ROE). When input from a manual control device is processed and executed, the display on the MP-285 controller will update automatically with the new position while the PC interface will not. Pressing this button will cause the coordinates on the PC interface to update.

The two boxes at the bottom left of the window display the resolution (**Res**) and **Velocity** at which the manipulator is set to move.

When the cursor is moved into the main workspace, the mouse cursor becomes a black cross. With a left click of the mouse, you can move the manipulator relative to the current position, represented by the smaller cross. The large block represents the X and Y axes while the **narrow strip** along the right represents the Z-axis.

Along the bottom, three boxes show the current **Cursor Position** along the axes so that the user can move the manipulator accurately and quickly to a new position. The X and Y boxes will update when the cursor is moved in the large block and the Z box will update when the cursor is moved within the narrow strip to the right.

The **Robotics** menu at the top of the screen lets you create, edit, and execute preset programs stored in the controller or on the PC (**Robotics/Host**), and transfer programs back and forth from the PC to the controller for editing and execution. (**Robotics/Controller**).

The **Setup** menu lets you fine-tune your manipulator so that you can select the specifications that are appropriate for your experiments. Here, you can choose a specific **Velocity** and **Resolution (High or Low)**. Under **Communications**, you can specify Serial Port and Baud Rate. **Hardware** allows one to choose a new origin and reset the Controller. Please refer to the main MP-285 manual before using this command.

Additional information is available through the Help menu. For tech support and information about the MP-285 PC Control Interface, choose **Help** or **About** from the menu bar.

APPENDIX A. LIMITED WARRANTY

- Sutter Instrument Company, a division of Sutter Instrument Corporation, limits the warranty on this instrument to repair and replacement of defective components for two years from date of shipment, provided the instrument has been operated in accordance with the instructions outlined in this manual.
- Abuse, misuse, or unauthorized repairs will void this warranty.
- Warranty work will be performed only at the factory.
- The cost of shipment both ways is paid for by Sutter Instrument during the first three months this warranty is in effect, after which the cost is the responsibility of the customer.
- The limited warranty is as stated above and no implied or inferred liability for direct or consequential damages is intended.
- Consumables are exempt from this warranty.
- An extended warranty for up to three additional years can be purchased at the time of ordering, or until the original warranty expires. For pricing and other information, please contact Sutter Instrument.

(This page intentionally blank.)

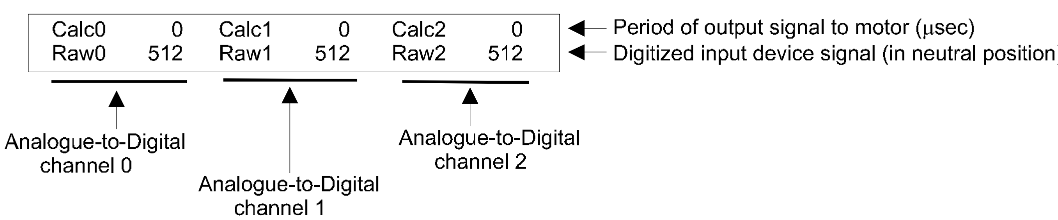
APPENDIX B. BALANCING INPUT DEVICE POTENTIOMETERS

NOTE: This section on the Joystick input device applies only to systems based on the original MP-285 controller – the MP-285A controller does not support the Joystick input device.

B.1. Joystick

The handle of the joystick provided with the MP-285 is directly linked to three potentiometers, one for each axis of movement. As the handle is moved or twisted it turns the stem on one or more of those potentiometers and changes their electrical resistance. Such a change in any one axis' potentiometer alters the voltage across that component and serves to transduce the mechanical motion of the joystick handle into an electrical signal ranging from 0 to 5 volts. A neutral position of the handle produces an intermediate voltage of 2.5 volts. Voltage deviation from the neutral position signal results in corresponding increases in micromanipulator velocity, while the direction that the voltage changes (above or below 2.5 volts) encodes the direction of micromanipulator movement. The voltage value of each axis on the joystick is an analog signal that is converted by the MP-285 controller to a digital value of 0-1023, with an intermediate value of 512 corresponding to an input voltage of 2.5 volts.

To view the "digitized" value of the inputs from each axis of the joystick access the **AD test** screen (**[PRGM\Setup\Utilities\AD test]**).



The lower row of values should read close to 512 when the handle is in a neutral position. **Fluctuation is normal.** With careful observation of the display it should be possible to determine the displayed median value for each A/D channel. This should be about 512 while the joystick is in its neutral position. Deflection or twisting of the handle will cause Raw values to vary up or down, depending on the axis and direction of handle movement.

The median "Raw" digital value on the AD test display may deviate from 512 in one or more of the A/D channels while the joystick is in its neutral position. This deviation can be ignored if ≤ 20 units. If this deviation plus any fluctuation becomes too large, it may exceed the limits of the **Dead zone** setting. As a result, a movement command will be issued even though the joystick is not being deflected.

The joystick shipped with the MP-285 allows you to correct the deviation in the neutral position signal voltage by adjusting the position of the joystick handle relative to the potentiometer stems (i.e., balance the potentiometers). Two axes, the up/down- and the left/right-handle deflections, can be balanced. These correspond to the A/D channels 1 and 2, respectively. The potentiometer associated with the twist motion of the handle is factory set and cannot be adjusted.

To adjust the neutral position digitized voltage value to 512 (i.e., balance the potentiometer) access the A/D test screen. Identify the balance knob next to the joystick handle (see Figure 8-1) that corresponds to the handle axis that is "out of balance". Turn the balance knob until the average raw digital value is as close to 512 as possible.

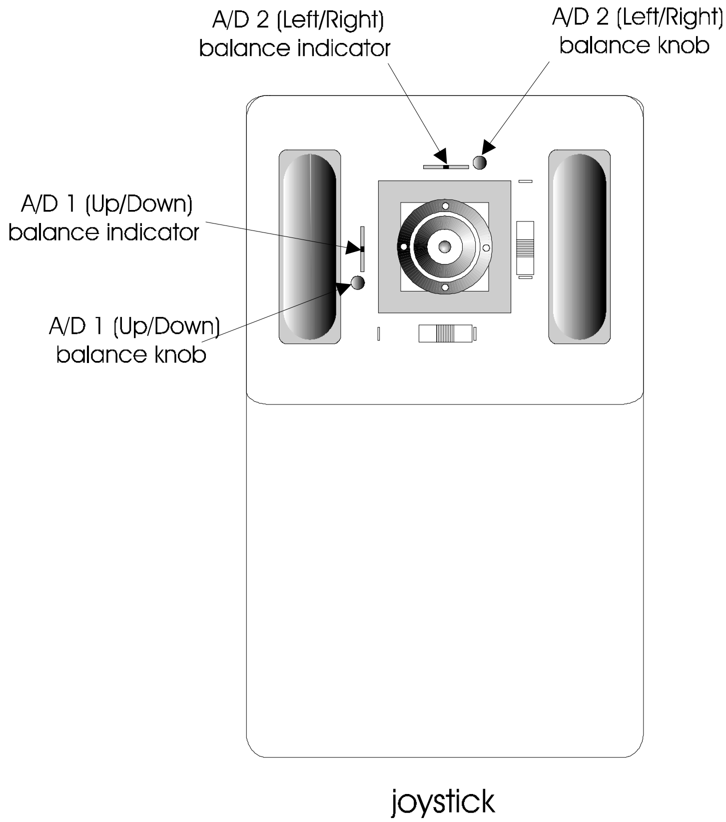


Figure 8-1. Potentiometer balance adjusters on the joystick.

APPENDIX C. TROUBLESHOOTING/FUSE REPLACEMENT

This Appendix contains information on basic troubleshooting, fuse replacement, and a detailed description of how to secure and repackage the manipulator for safe transport to Sutter Instrument Company for repair.

C.1. Troubleshooting

Several types of problems are described along with corresponding suggestions for correcting these problems. As with all our products, Sutter Instrument Company offers unlimited technical support for users of the MP-285 in the event that you encounter a problem that is not described here or if the corrective measures suggested are not effective.

The micromanipulator won't move in any direction along any axis and the display does not indicate any change in the coordinates.

- Make certain that the movement screen is being displayed. If it is not, press the MOVE key on the keypad.
- Check all cable connections.
- Press the ESC key on the keypad to display the MAIN MENU. Then press the MOVE key to return to the Movement screen.
- Press the RESET button on the front panel of the MP-285 controller.
- Turn the power on and off.

The micromanipulator won't move in either direction along one axis (even though the display indicates a change in the coordinates). All other axes work fine.

- Check to see if the slide's excursion is obstructed. Mounting screws, components of a microscope, or other nearby devices could be preventing travel.

The micromanipulator will not move in one direction along one of the axes and the display does not indicate a change in the coordinates for that direction. The micromanipulator moves fine in the other direction along the same axis and all other axes work fine.

- Check to see if the slide is at the endpoint of its excursion. Located on one edge of each axis are index marks to indicate the position of the driven slide relative to the driver. When the center marks of the opposing indexes are aligned with one another, the driven slide is at the middle of its 1-inch (25mm) excursion (see Figure 8-2).

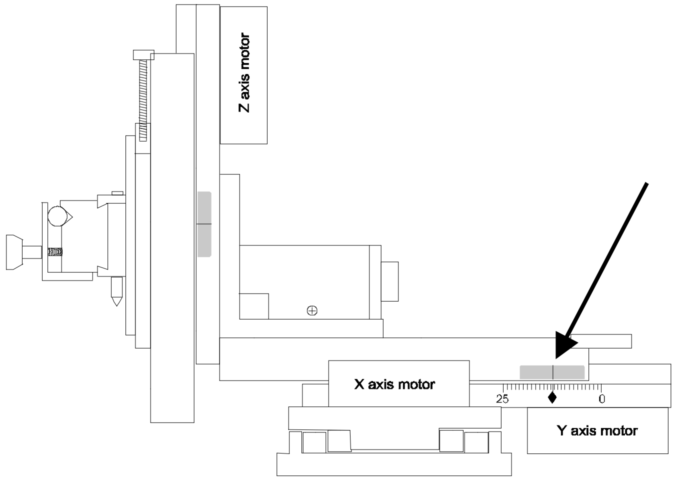


Figure 8-2. Mid-excursion index mark.

The slide for one axis of the micromanipulator is at its limit of excursion (see Figure 8-2), it won't move in either direction and the display indicates a change in the coordinates for only one of the directions. All other axes work fine.

- There is no action possible by the user. Call Sutter Instrument Company for technical support

While in the Movement screen and using a Joystick, the 3 and 6 keys do not toggle the movement resolution (0.2/0.04 µm) and/or the [MOVE] key does not toggle the movement mode (Continuous/Pulse).

- Check the switch settings on the Hardware menu ([PRGM\Setup\Hardware]). These should read as indicated in the example shown below if you are using a joystick. If they do not, set them accordingly (see the [PRGM\Setup\Hardware] section in this manual).

```
Dead zone   80                      SW1  K   SW2  K
( 0=keypad  1=switch)               SW3      SW4
```

- Press the RESET button on the front panel of the MP-285 controller

While in the Movement screen (and when using a Force Sensitive Resistor Pad), the MOVE key does not toggle the movement mode (Continuous/Pulse).

- Check the switch settings on the Hardware menu ([PRGM\Setup\Hardware]). These should read as indicated in the example shown below if you are using a Force Sensitive Resistor Pad. If they do not, set them accordingly (see the [PRGM\Setup\Hardware] section in this manual).

```
Dead zone   80                      SW1  S   SW2
( 0=keypad  1=switch)               SW3      SW4
```

- Press the RESET button on the front panel of the MP-285 controller.

C.2. Fuse Replacement

If the controller fails to power up when the power switch is turned on (i.e., if display and fans do not come on), check the line power fuse to see if it has blown. The fuse is in the fuse holder on the power entry module on the back of the controller. To remove the fuse holder first unplug the power cord from the power entry module. This will reveal a slot just under the edge of the fuse holder. Use a screwdriver to pry the holder straight out of the power entry module.

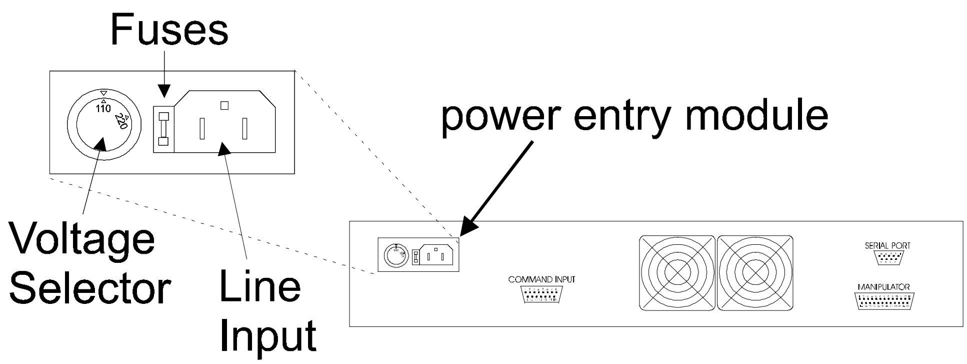


Figure C-3. Power entry module.

The fuse that is readily visible in the fuse holder when you take it out is the one that is "active" when the holder is installed. A spare fuse is also stored within the fuse holder. The spare fuse is concealed in a compartment as shown in Figure C-4. To remove the spare fuse, press down on the end of the compartment to push it out of the other end. The old fuse can serve as a convenient tool for pushing the spare fuse compartment out. Replace the active fuse with the spare and re-install the fuse holder and power cord. If the MP-285 controller fails to power up with the new fuse installed, call Sutter Instrument technical support personnel for assistance.

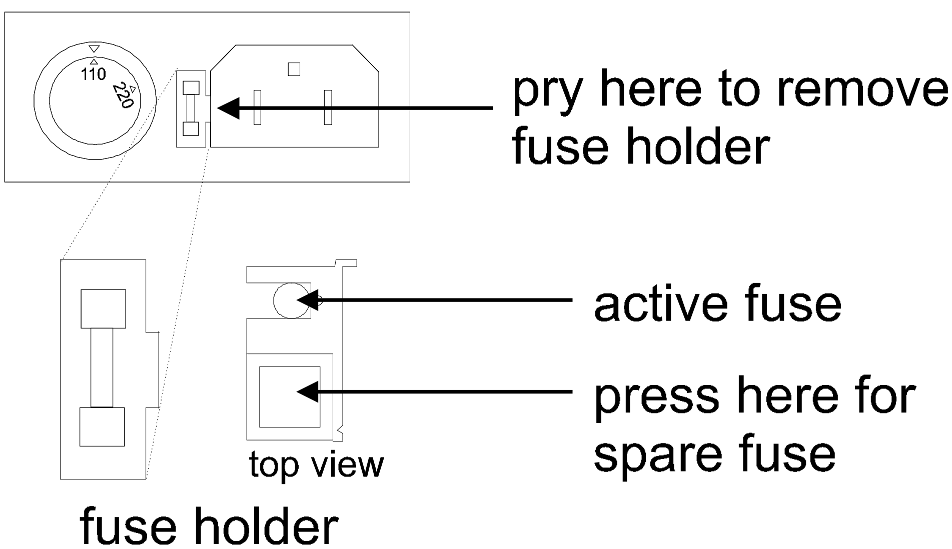


Figure C-4. Fuse holder.

Make certain that the type and rating of the fuse being replaced are as indicated in the table below, according to the mains voltage setting being used.

Table C-1. Mains fuse type and rating.

Mains Voltage Setting	Fuse (Type: Time Delay, 5 x 20 mm, glass tube)	
	Rating	Manufacturer Examples
"110" (100 – 120 VAC)	2A, 250V (Time Delay)	Bussmann: GMC-2A, GMC-2-R (RoHS), GDC-2A, or S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)
"220" (200 – 240 VAC)	T1.0A, 250V	Bussmann: GDC-1A or S506-1A (RoHS) Littelfuse: 218 001 or 218 001.P (RoHS)

Table C-2. MP-285A Mains fuse type and rating.

Mains Voltage Setting	Fuse (Type: Time Delay, 5 x 20 mm, glass tube)	
	Rating	Manufacturer Examples
100 - 240 VAC	2A, 250V (Time Delay)	Bussmann: GMC-2A, GMC-2-R (RoHS), GDC-2A, or S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)

C.3. Repacking the MP-285 for Shipping

It is hoped that the user, or the user in conjunction with Sutter technical support, will be able to diagnose and repair most MP-285 malfunctions. However, we realize that this is not always the case.

Several components within the mechanical portion of the MP-285 micromanipulator are quite susceptible to damage if the unit is not properly secured during shipping. The following section details the necessary procedure for securing the MP-285 mechanical either should you need to ship it for repair or in the event it must be moved for other reasons. It is recommended that you follow this procedure whenever the manipulator will be moved in any way other than a hand carry.

If possible, please use the original shipping container for packing the MP-285 system. If it is necessary to use a replacement shipping container keep in mind that the MP-285 should be 1) enclosed in a bag to keep it clean during shipping, 2) well protected in the box by bubble wrap (or equivalent) and 3) surrounded by a foam enclosure (or equivalent). **Do not rely on foam "peanuts" alone!**

To install the shipping screws:

1. Locate the 6 shipping screws, the 3 shipping screw warning tags and the 2.5mm hex (Allen) wrench that came with the unit from the factory. If it is necessary to replace the screws, you will need six M3 x .5 x 18 mm machine screws.
2. Use the input device to move the X-axis slide until the shipping screw holes are lined up. **DO NOT MOVE THE SLIDES MANUALLY TO ALIGN THE HOLES!**

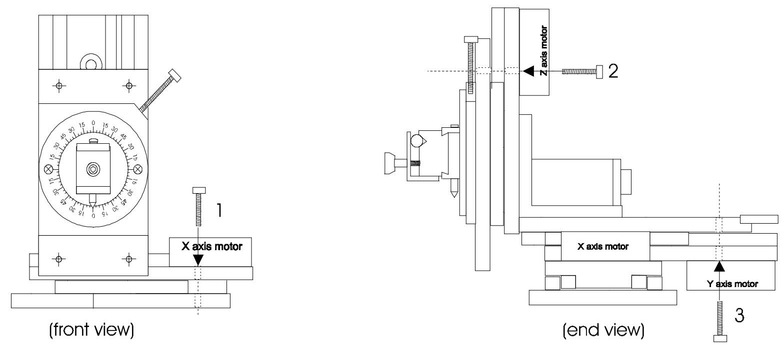


Figure C-5. Location and orientation of shipping screw holes.

3. Put the warning tag on a shipping screw and insert it into the holes as indicated above. If the screw does not turn easily into the threads remove it and adjust the slide using the input device to better align the holes. Repeat this procedure as needed. **DO NOT USE THE INPUT DEVICE TO MOVE A SLIDE WHILE THE SCREWS ARE IN THE HOLES! DO NOT OVER TIGHTEN THE SCREW. YOU ONLY NEED TO TURN IT UNTIL THE HEAD OF THE SCREW IS JUST SEATED.**
4. Now put the second screw for the X-axis through the hole on the other end of the warning tag and install that screw in the shipping screw holes along the opposite edge of the slide.
5. Temporarily label (with tape, for example) the X-axis control(s) of the input device as a reminder not to actuate the X motor once the shipping screws are installed into the X-axis slides.

(This page intentionally left blank.)

APPENDIX D. TECHNICAL SPECIFICATIONS

Travel:	1 inch (25.4 mm) in all three axes
Resolution:	
Low	0.2 µm / µstep
High	0.04 µm / µstep
Maximum speed:	3 mm / sec. (3,000 µm / sec) at Low Resolution 1.31 mm /sec (1,310 µm / sec) at High Resolution
Long term stability:	< 10 nm/hour @ 24°C
Drive mechanism:	Precision worm gear/capstan
Serial interface:	RS-232; DB9; Data rates: 1200, 2400, 4800, 9600, 19200 bps, 8 data bits, 1 stop bit, no parity, no flow control
USB-VCP interface (MP-285A only)	USB with VCP (Virtual COM Port) device driver (RS-232C); Data rate: 9600 bps, 8 data bits, 1 stop bit, no parity, RTS/CTS or "Hardware" flow control
Dimensions (H x W x D):	
Mechanical	5 x 4 x 5.25 in 12.5 x 10 x 13.5 cm
Controller (rackmount)	4 x 19 x 12.25 in 10 x 48.25 x 31 cm
Controller (tabletop)	4 x 16 x 12.25 in 10 x 40.5 x 31 cm
Weight:	
Mechanical	2 lb., 14 oz. 1.3 kg
Controller	10 lbs. 4.5 kg

Electrical:

Input voltage (Mains): MP-285 (switchable):

100 – 120 VAC, 50/60 Hz
200 – 240 VAC, 50/60 Hz

MP-285A (autoswitch)

100 - 240 VAC, 50-60Hz

Power Consumption: 150 Watts max.

Mains fuse (rear of cabinet):

Table D-1. MP-285 Mains fuse type and rating.

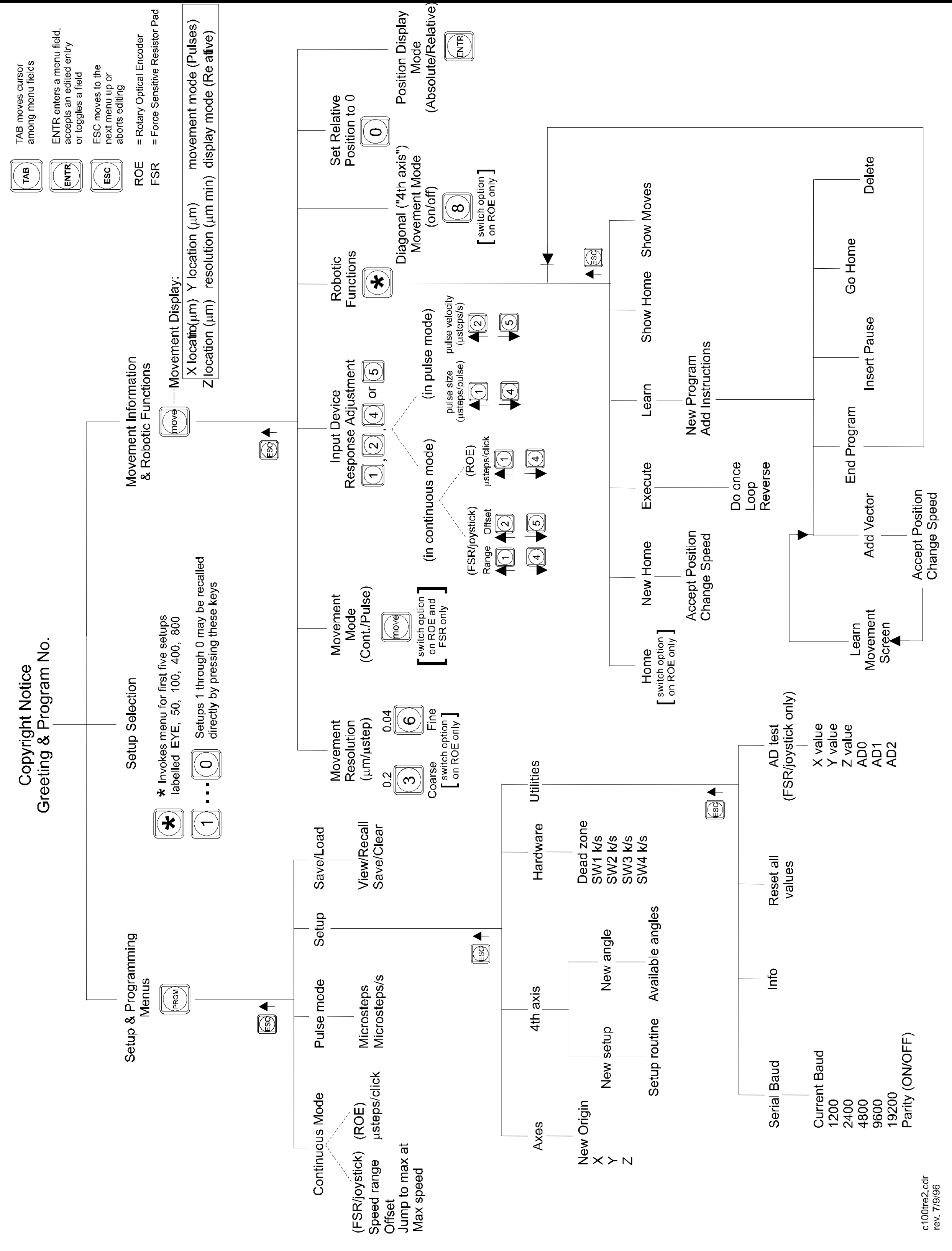
Mains Voltage Setting	Fuse (Type: Time Delay, 5 x 20mm, glass tube)	
	Rating	Manufacturer Examples
"110" (100 – 120 VAC)	2A, 250V (Time Delay)	Bussmann: GMC-2A, GMC-2-R (RoHS), GDC-2A, or S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)
"220" (200 – 240 VAC)	T1.0A, 250V	Bussmann: GDC-1A or S506-1A (RoHS) Littelfuse: 218 001 or 218 001.P (RoHS)

Table D-2. MP-285A Mains fuse type and rating.

Mains Voltage Setting	Fuse (Type: Time Delay, 5 x 20 mm, glass tube)	
	Rating	Manufacturer Examples
100 - 240 VAC	2A, 250V (Time Delay)	Bussmann: GMC-2A, GMC-2-R (RoHS), GDC-2A, or S506-2A (RoHS) Littelfuse: 239 002 or 239.002.P (RoHS)

Power cord: 10A, 250V,
with safety ground plug

APPENDIX E. ANNOTATED MENU TREE



(This page intentionally left blank.)

APPENDIX F. EXTERNAL CONTROL COMMANDS REFERENCE

NOTE: Unless otherwise specified, all "MP-285" references refer equally to Model MP-285 and Model MP-285A controllers.

Controlling the MP-285 or MP-285A externally via computer is accomplished by sending commands between the computer and the equivalent connector on the rear of the controller: **SERIAL** (RS-232, 9-pin DSUB (MP-285 or MP-285A) or **USB** (MP-285A only).

The SERIAL (RS-232) Interface: RS-232-C, minimal 3-wire (Ground, Transmit, & Receive), 9-pin D-Shell connector (labeled "SERIAL" on the rear panel of the controller.

Table F-1. Serial RS-232 (DB9 connector) port settings.

Property	Setting
Data ("Baud") Rate (bps (bits per second))	19200, **9600***, 4800, 2400, 1200
Data Bits	**8**
Stop Bits	**1***, 1.5, 2
Parity	**None***, Even, Odd
Flow Control	**None**

*** Default**

NOTE: The data rate can be selected via the MP-285 controller's display and keypad. The default data rate (9600 bps) is recommended for most applications. The Parity can also be configured to be "ON", although the OFF ("None") setting is recommended and is the default.

The MP-285A USB Interface: Controlling the MP-285A externally via computer is accomplished by sending commands over the USB interface between the computer and the USB connector on the rear panel of the MP-285A controller/ROE. The USB device driver for Windows is downloadable from Sutter Instrument's web site (www.sutter.com). The MP-285A requires USB CDM (Combined Driver Model) Version 2.10.00 or higher. The CDM device driver for the MP-285A consists of two device drivers: 1) USB device driver, and 2) VCP (Virtual COM Port) device driver. Install the USB device driver first, followed by the VCP device driver. The VCP device driver provides a serial RS-232 I/O interface between a Windows application and the MP-285A. Although the VCP device driver is optional, its installation is recommended even if it is not going to be used. Once installed, the VCP can be enabled or disabled.

The CDM device driver package provides two I/O methodologies over which communications with the controller over USB can be conducted: 1) USB Direct (D2XX mode), or 2) Serial RS-232 asynchronous via the VCP device driver (VCP mode). The first method requires that the VCP device driver not be installed, or if installed, that it be disabled. The second method requires that the VCP be installed and enabled.

Virtual COM Port (VCP) Serial Port Settings: The following table lists the required RS-232 serial settings for the COM port (COM3, COM5, etc.) generated by the installation and enabling of the VCP device driver.

Table F-2. MP-285A USB-VCP interface serial port settings.

Property	Setting
Data ("Baud") Rate (bps (bits per second))	9600
Data Bits	8
Stop Bits	1
Parity	None
Flow Control [3]	"Hardware" or RTS/CTS

The settings shown in the above table can be set in the device driver's properties (via the Device Manager if in Windows) and/or programmatically in your application.

Protocol and Handshaking: Most command sequences have a terminator: ASCII CR (Carriage Return; 13 decimal, 0D hexadecimal) (see the *MP-285 external-control commands* table). All commands return an ASCII CR (Carriage Return; 13 decimal, 0D hexadecimal) to indicate that the task associated with the command has completed. When the controller completes the task associated with a command, it sends ASCII CR back to the host computer indicating that it is ready to receive a new command. If a command returns data, the last byte returned is the task-completed indicator.

Command Sequence Formatting: Each command sequence consists of at least one byte, the first of which is the "command byte". Those commands that have parameters or arguments require a sequence of bytes that follow the command byte. No delimiters are used between command sequence arguments, and command sequence terminators are used in most cases. Although most command bytes can be expressed as ASCII displayable/printable characters, the rest of a command sequence must generally be expressed as a sequence of unsigned byte values (0-255 decimal; 00 – FF hexadecimal, or

[3] While the Flow Control property for the RS-232 DB9 interface is always set to "None", it must be set to "Hardware" or RTS/CTS signaling for the virtual serial port via the USB-VCP device driver.

00000000 – 11111111 binary). Each byte in a command sequence being transmitted to the controller must contain an unsigned binary value. Attempting to code command sequences as “strings” is not advisable. Any command data being returned from the controller must also be received and initially treated as a sequence of unsigned byte values. Groups of contiguous bytes can later be combined to form larger values, as appropriate (e.g., 2 bytes into 16-bit “word”, or 4 bytes into a 32-bit “long” or “double word”). For the MP-285 controller, all axis position values (number of microsteps) are stored as “long” (or “signed long”) 32-bit positive or negative values, and each is transmitted and received to and from the controller as four contiguous bytes.

Axis Position Command Parameters: All axis positional information is exchanged between the controller and the host computer in terms of microsteps. Conversion between microsteps and microns (micrometers) is the responsibility of the software running on the host computer (see *Microns/microsteps conversion* table for conversion factors).

Microsteps are stored as positive or negative 32-bit values (“long” (or optionally, “signed long”) for C/C++; “I32” for LabVIEW).

The 32-bit value consists of four contiguous bytes, with a byte/bit-ordering format of Little Endian (“Intel”) (most significant byte (MSB) in the first byte and least significant (LSB) in the last byte). If the platform on which your application is running is Little Endian, then no byte order reversal of axis position values is necessary. Examples of platforms using Little Endian formatting include any system using an Intel/AMD processor (including Microsoft Windows and Apple Mac OS X).

If the platform on which your application is running is Big Endian (e.g., Motorola PowerPC CPU), then these 32-bit position values must have their bytes reverse-ordered <u>after</u> receiving from, or <u>before</u> sending to, the controller. Examples of Big-Endian platforms include many non-Intel-based systems, LabVIEW (regardless of operating system & CPU), and Java (programming language/environment). MATLAB and Python (script programming language) are examples of environments that adapt to the system on which each is running, so Little-Endian enforcement may be needed if running on a Big-Endian system. Some processors (e.g., ARM) can be configured for specific endianess.

Microsteps and Microns (Micrometers): All coordinates sent to and received from the controller are in microsteps. To convert between microsteps and microns (micrometers), use the following conversion factors (multipliers):

Table F-3. Microns/microsteps conversion.

Device	From/To Units	Conv. Factor
MP-285/M* micromanipulator	microsteps → microns	0.04
	microns → microsteps	25
MT-800 (**MT-20xx**) series translators	microsteps → microns	0.05
	microns → microsteps	20

** Applies also to 3DMS/M & MP-x8-series stages, and MOM & SOM microscope objective movers*

For accuracy in your application, type these conversion factors as “double” (avoid using the “float” type as it lacks precision with large values). When converting to microsteps, type the result as a 32-bit “long”, “signed long”, or “I32” integer. When converting to microns, type the result as “double” (64-bit double-precision floating-point values).

Ranges and Bounds:

Table F-4. Ranges and bounds.

Device	Axis	Len. (mm)	Origin	Microns (Micrometers (μm))	Microsteps (μsteps)
MP-285/M, 3DMS, MP-78, MOM, SOM	X, Y, & Z	25 mm	**COT***	**-12,500 – 12,500**	**-200,000 – 200,000**
			BOT	0 – 25,000	0 – 400,000
MT-800	X & Y	22 mm	**COT***	**-11,000 – 11,000**	**-140,800 – 140,800**
			BOT	0 – 22,000	0 – 281,600
	Z	25 mm	**COT***	**-12,500 – 12,500**	**-200,000 – 200,000**
			BOT	0 – 25,000	0 – 400,000

** Factory default.*

NOTE: Origin is a physical position of travel that defines the center of the absolute position coordinate system (i.e., absolute position 0).
Physical Positions: BOT (Beginning Of Travel), COT (Center Of Travel), & EOT (End Of Travel).
In the MP-285, the Origin can be set to any physical position (factory default is COT).

NOTE: The MP-x8-series stage and MT-800 (MT-20xx series) translator do not have a Z-axis motor. In either case, the controller’s Z axis can be optionally connected to a motor of a different device (e.g., focus drive).

Travel Speed: The following table shows the selectable travel speeds for single-, double-, and triple-axis movements for supported devices using orthogonal move commands.

Table F-5. Travel speeds.

Resolution	Speed Range (microns/sec)
Low (coarse: 0.2 µm/µstep (10 µsteps/step))	0 – 3000*
High (fine: 0.04 µm/µstep (50 µsteps/step))	0 – 1310

** CAUTION: Although the absolute maximum microns/sec. speed allowable in low (coarse) resolution is 6,550, it is essential that a speed no higher than 3,000 be used with the MP-285A model controller.*

Command Reference: The following table lists all the external-control commands for the MP-285.

Table F-6. MP-285[A] external-control commands.

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Get Current Position ('c')	**Tx**	All	2	0	99	63	0110 0011	0099		**'c'**	Returns the current positions (µsteps) of X, Y, & Z axes.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		13		Three 4-byte (32-bit) values (current positions in µsteps of X, Y, & Z), + 1 byte for completion indicator.						
				0 (4)							X–axis position
				4 (4)							Y–axis position
				8 (4)							Z–axis position
				12	13	0D	0000 1101			<CR>	Task-completion indicator
Move to Specified Position ('m')	**Tx**	All	14	0	109	6D	0110 1100	0109		**'m'**	Moves to specified position (µsteps) (see *Ranges* table)
				1 (4)							X µsteps
				5 (4)							Y µsteps
				9 (4)							Z µsteps
				13	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator
Set Velocity & Resolution ('V')	**Tx**	All	4	0	86	56	0101 0110	0086		**'V'**	Command (Note: Uppercase 'V') (see *Resolution & Velocity* note)
				1 (2)	0 – 4095 or 32,768 – 35,768	0000 – 051E or 8000 – 8BB8	00000000 00000000 – 00000101 00011110 or 10000000 00000000 – 10001011 10111000				One **unsigned** short (16-bit) integer (2 bytes) containing both resolution and velocity values. MSB (Bit 15) contains resolution setting; remaining bits (14 –0) contains velocity value. **Resolution (Bit 15)**: 0 = Low (coarse: 0.2 µm/µstep (10 µsteps/step)) 1 = High (fine: 0.04 µm/µstep (50 µsteps/step)) **Velocity (Bits 14-0)**: Low Res.: 0 – 6550 (MP-285) or 3000 (MP-285A) µm/sec High Res.: 0 – 1310 µm/sec
				3	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator
Set Origin ('o')	**Tx**	All	2	0	111	6F	0110 1111	0111		**'o'**	Sets the Absolute Origin to the current position.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator

Command	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Description
					Dec.	Hex.	Binary				
Set Absolute Mode (‘a’)	**Tx**	All	2	0	97	61	0110 0001	0097		**‘a’**	Sets movement mode to Absolute. Each ‘m’-command axis value represents an absolute position. (Note: No VFD update)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator
Set Relative Mode (‘b’)	**Tx**	All	2	0	98	62	0110 0010	0098		**‘b’**	Sets movement mode to Relative. Each ‘m’-command axis value represents a position relative to the current position. (Note: No VFD update)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx			1	13	0D	0000 1101	0013		<CR>	Task-completion indicator
Interrupt Move (^C)	**Tx**	All	1	0	3	03	0000 0011	0003	^C	**<ETX>**	Interrupts an ‘m’-command initiated move in progress
	Rx		1	0	61	3D	0011 1011	0061		**‘=’**	Move in progress indicator
			2	0	13	0D	0000 1101	0013		<CR>	Task-completion indicator
	Rx			0	13	0D	0000 1101	0013		<CR>	Task-completion indicator (movement was not in progress)
Refresh VFD[1] Display (‘n’)	**Tx**	All	2	0	110	6E	0110 0110	0101		**‘n’**	Refreshes the controller’s display (X, Y, & Z coordinates only)
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	0	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator
Reset Controller (‘r’)	**Tx**	All	2	0	114	72	0111 0010	0114		**‘r’**	Resets the controller.
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	0	13	0D	0000 1101	0013		<CR>	Task-completion indicator
Get Status (‘s’)	**Tx**	All	2	0	115	73	0111 0011	0115		**‘s’**	Returns status information
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		33	0							Status data – see *Status Data Structure* table.
				32	13	0D	0000 1101	0013		<CR>	Task-completion indicator

[1] Vacuum Florescence Display

NOTES:

1. **Task-Complete Indicator:** All commands will send back to the computer the "Task-Complete Indicator" to signal the command and its associated function in controller is complete. The indicator consists of one (1) byte containing a value of 13 decimal (0D hexadecimal), and which represents an ASCII CR (Carriage Return).
2. **Intercommand Delay:** A short delay (usually around 2 ms) is recommended between commands (after sending a command sequence and before sending the next command).
3. **Clearing the I/O Send & Receive Buffers:** Clearing (purging) the transmit and receive buffers of the I/O port immediately before sending any command is recommended. Note that this clearing of the buffers affects only the computer-side I/O; it does not (necessarily) clear the buffers on the controller side, requiring, when necessary, to reset/power-cycle the controller. Following the rules described will generally avoid problems with getting garbage data in the I/O buffers of both the computer and controller (i.e., using exact number of bytes for both command sequences and return data (as per the *Commands* table), never sending a command before the previous command is finished with its task, etc.).
4. **Positions in Microsteps and Microns:** All positions sent to and received from the controller are in microsteps (μsteps). See *Microns/microsteps conversion* table) for conversion between μsteps and microns (micrometers (μm)).

   Declaring position variables in C/C++:
   ```
   /* current position for X, Y, & Z */
   long   cp_x_us, cp_y_us, cp_z_us;  /* microsteps */
   double cp_x_um, cp_y_um, cp_z_um;  /* microns */
   /* specified (move-to) position for X, Y, & Z */
   long   sp_x_us, sp_y_us, sp_z_us;  /* microsteps */
   double sp_x_um, sp_y_um, sp_z_um;  /* microns */
   ```
   Use the same convention for other position variables the application might need.

   Declaring the microsteps/microns conversion factors in C/C++:
   ```
   /* conversion factors for MP-285/M based config. */
   double us2umCF = 0.04; /* microsteps to microns */
   double um2usCF = 25;   /* microns to microsteps */
   /* conversion factors for MT-800 config. */
   double us2umCF = 0.05; /* microsteps to microns */
   double um2usCF = 20;   /* microns to microsteps */
   ```
   NOTE: In an MP-285A-based system configured for an MP-78 stage or MT-800-based XY translator (MT-2078), the Z axis may be configured for different conversion factors (e.g., if Z is wired to a separate device such as a focus drive). In such cases, make sure the appropriate microsteps/microns conversion factors are used for Z while using the standard factors for X and Y.

   Converting between microsteps and microns in C/C++:
   ```
   /* converting X axis current position */
   cp_x_um = cp_x_us * us2umCF;  /* microsteps to microns */
   cp_x_us = cp_x_um * um2usCF;  /* microns to microsteps */
   ```
   Do the same for Y and Z, and for any other position sets used in the application.
5. **Ranges and Bounds:** See *Ranges and Bounds* table for exact minimum and maximum values for each axis of each compatible device that can be connected. All move commands include positive or negative values for positions. All positions are absolute as measured from the Origin position as set in the controller for all axes of the attached device. The factory default Origin position is the physical center position (between beginning of travel and end of travel) of the device. In application programming, it is important that positional values be checked (>= minimum and <= maximum) to ensure that a position is within the bounds of travel before it is sent to the controller.
6. **Absolute Positioning System Origin:** The Origin is set to a physical position of travel to define absolute position 0. The factory default physical Origin position is center of travel (COT). This means that all higher positions (towards end of travel (EOT)) are positive values, and all lower positions (towards beginning of travel (BOT)) are negative values. The Origin can be changed (via the controller's front panel display/keypad or via the Origin ('o') command sent from an external program.

   CAUTION: When changing the Origin from its factory default, it is not possible to obtain the new Origin's physical position via an external control command. If changing the Origin's physical position via the external control 'o' command, it is recommended that the external application keep careful track of all Origin changes, and automatically adjust its view of the absolute position coordinate system according to the current Origin's physical position.
7. **Absolute vs. Relative Positioning:** Current position (via the 'c' command) report absolute positions of each axis. Moving to a new position (via the 'm' command) is specified with absolute position values when in Absolute mode ('a' command) or with relative values (relative to the current position) when in Relative mode ('b' command).

   CAUTION: In an external control program, care should be taken to ensure that the Absolute/Relative mode state be updated upon a mode change and kept track of, as it not possible to obtain the current mode from the controller. In addition, any computational relative positioning made in an external program while in Absolute mode must ensure that relative positions are accurately translated to correct absolute positions before initiating a move command.
8. **Position Value Typing:** All positions sent and received to and from the controller are in microsteps and consist of 32-bit integer values (four contiguous bytes). Position values can be either positive or negative, so the type must be "signed". Although each positional value is transmitted to, or received from, the controller as a sequence of four (4) contiguous bytes, for computer application computational and storage purposes each should be typed as a signed integer ("long" or "signed long" in C/C++; "I32" in LabVIEW, etc.). Note that in Python, incorporating the optional NumPy package brings robust data typing like that used in C/C++ to your program, simplifying coding and adding positioning accuracy to the application.
9. **Position Value Bit Ordering:** All 32-bit position values transmitted to, and received from, the controller must be bit/byte-ordered in "Little Endian" format. This means that the least significant bit/byte is last (last to send and last to receive). Byte-order reversal may be required on some platforms. Microsoft Windows, Intel-based Apple Macintosh systems running Mac OS X, and most Intel/AMD processor-based Linux distributions handle byte storage in Little-Endian byte order so byte reordering is not necessary before converting to/from 32-bit "long" values. LabVIEW always handles "byte strings" in "Big Endian" byte order irrespective of operating system and CPU, requiring that the four bytes containing a microsteps value be reverse ordered before/after conversion to/from a multibyte type value (I32, U32, etc.). MATLAB automatically adjusts the endianess of multibyte storage entities to that of the system on which it is running, so explicit byte reordering is generally unnecessary unless the underlying platform is Big Endian. If your development platform does not have built-in Little/Big Endian conversion functions, bit reordering can be accomplished by first swapping positions of the two bytes in each 16-bit half of the 32-bit value, and then swap positions of the two halves. This method efficiently and quickly changes the bit ordering of any multibyte value between the

two Endian formats (if Big Endian, it becomes Little Endian, and if Little Endian, it becomes then Big Endian).

10. **Travel Lengths and Durations:** "Move" commands might have short to long distances of travel. If not polling for return data, an appropriate delay should be inserted between the sending of the command sequence and reception of return data so that the next command is sent only after the move is complete. This delay can be auto calculated by determining the distance of travel (difference between current and target positions) and rate of travel. This delay is not needed if polling for return data. In either case, however, an appropriate timeout must be set for the reception of data so that the I/O does not time out before the move is made and/or the delay expires.

11. **Z-Axis Usage in 2-Axis Systems:** On an MT-800, MP-78, or MP-88 system, the Z axis can be used as a focus drive, with a conversion factor that may be custom according to the make and model of the microscope being used.

12. **Setting Resolution & Velocity**: The Set Resolution & Velocity ('V') command unsigned 16-bit value can be easily composed mathematically using the following formula:

    *unsigned short ResSpeed = (Res * 0x8000) + Speed*

    where "ResSpeed" is the final unsigned 16-bit value (Little Endian bit order), "Res" is the resolution (0 for Low; 1 for High), 0x8000 (32,768 decimal) as a multiplier positions the resolution (0 or 1) to Bit 15 (the high order bit), and then the "Speed" value is simply added to occupy Bits 14 through 0. The "unsigned short" is a C/C++ data type definition that ensures that "ResSpeed" is a 16-bit variable that holds only positive values.

13. **Move Interruption:** A command should be sent to the controller only after the task of any previous command is complete (i.e., the task-completion terminator (CR) is returned). One exception is the "Interrupt Move" (^C) command, which can be issued while a command-initiated move is still in progress.

Table F-7. Status data structure (as returned by the Get Status ('s') command.

<table>
<tr><th>Offset</th><th>Length</th><th>Name</th><th colspan="8">Description</th></tr>
<tr><td rowspan="18">0</td><td rowspan="18">8 bits</td><td rowspan="18">FLAGS</td><td>Bit</td><td>Name</td><td>Description</td><td colspan="5">Values</td></tr>
<tr><td rowspan="12">0-3</td><td rowspan="12">SETUP #</td><td rowspan="12">Currently loaded setup number coded in BCD (decimal digit 0-9)</td><td colspan="4">Binary-Coded Decimal (BCD)</td><td rowspan="2">Dec. Digit</td></tr>
<tr><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>2</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>3</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>4</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>5</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>6</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>7</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>8</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>9</td></tr>
<tr><td></td><td></td><td></td><td colspan="2">1 (Set)</td><td colspan="3">0 (Clear)</td></tr>
<tr><td>4</td><td>ROE_DIR</td><td>Last ROE direction</td><td colspan="2">Negative</td><td colspan="3">Positive</td></tr>
<tr><td>5</td><td>REL_ABS_F</td><td>Display origin</td><td colspan="2">Absolute</td><td colspan="3">Relative</td></tr>
<tr><td>6</td><td>MODE_F</td><td>Manual mode flag</td><td colspan="2">Continuous</td><td colspan="3">Pulse</td></tr>
<tr><td>7</td><td>STORE_F</td><td>Setup condition</td><td colspan="2">Stored</td><td colspan="3">Erased</td></tr>
<tr><td>1</td><td>Byte</td><td>UDIRX</td><td colspan="8" rowspan="3">User-defined values for motor axis directions. Valid values: 0-5</td></tr>
<tr><td>2</td><td>Byte</td><td>UDIRY</td></tr>
<tr><td>3</td><td>Byte</td><td>UDIRZ</td></tr>
<tr><td>4</td><td>Word</td><td>ROE_VARI</td><td colspan="8">Microsteps per ROE click</td></tr>
<tr><td>6</td><td>Word</td><td>UOFFSET</td><td colspan="8">User-defined period start value</td></tr>
<tr><td>8</td><td>Word</td><td>URANGE</td><td colspan="8">User-defined period range</td></tr>
<tr><td>10</td><td>Word</td><td>PULSE</td><td colspan="8">Number of microsteps per pulse</td></tr>
<tr><td>12</td><td>Word</td><td>USPEED</td><td colspan="8">Adjusted pulse speed microsteps per sec.</td></tr>
<tr><td>14</td><td>Byte</td><td>INDEVICE</td><td colspan="8">Input device type</td></tr>
<tr><td rowspan="5">15</td><td rowspan="5">8 bits</td><td rowspan="5">FLAGS_2</td><td>Bit</td><td>Name</td><td>Description</td><td colspan="2">1 (Set)</td><td colspan="3">0 (Clear)</td></tr>
<tr><td>0</td><td>LOOP_MODE</td><td>Program loops</td><td colspan="2">Do loops</td><td colspan="3">Execute once</td></tr>
<tr><td>1</td><td>LEARN_MODE</td><td>Learn mode status</td><td colspan="2">Learning now</td><td colspan="3">Not learning</td></tr>
<tr><td>2</td><td>STEP_MODE</td><td>Resolution (microsteps/step)</td><td colspan="2">50</td><td colspan="3">10</td></tr>
<tr><td>3</td><td>SW2_MODE</td><td>Joystick side button</td><td colspan="2">Enabled</td><td colspan="3">Disabled</td></tr>
</table>

Offset	Length	Name	Description				
			4	SW1_MODE	Enable FSR/Joystick	Enabled	Cont/Pulse (keypad)
			5	SW3_MODE	ROE switch	Enabled	Disabled
			6	SW4_MODE	Switches 4 & 5	Enabled	Disabled
			7	REVERSE_IT	Program sequence	Reverse	Normal
16	Word	**JUMPSPD**	"Jump to max at" speed				
18	Word	**HIGHSPD**	"Jumped to" speed				
20	Word	**DEAD**	Dead zone, not saved				
22	Word	**WATCH_DOG**	Programmer's function (analog input for overload protection)				
24	Word	**STEP_DIV**	Microns ←→ Microsteps conversion factor. See Note 2.				
26	Word	**STEP_MUL**	Microns ←→ Microsteps conversion factor. See Note 2.				
28	Word	**XSPEED**	Velocity (microns/sec., Bits 14 - 0) & resolution (0 or 1, Bit 15). See Note 3.				
30	Word	**VERSION**	Firmware version. See Note 4.				
32	Byte		End of received data terminator (ASCII CR (13 decimal or 0D hexadecimal))				

NOTES:

1. All values are stored in Little-Endian bit order. All byte values are ordered Bit 7 through Bit 0 (left to right). All "word" (16-bit) values are ordered Bit 15 through Bit 0 (left to right). To reverse the bit order of word values to Big Endian, swap positions of both bytes (least significant byte becomes most significant and most significant becomes least significant).

2. **STEP_DIV and STEP_MUL:** Both contain 16-bit values used as factors for converting positional values between microns and microsteps. See the *Microns/microsteps conversion factors* table for what the values need to be for conversions. Position values in microns are typically stored in "double" (for double-precision floating-point) data type variables, while positions in microsteps are stored as 32-bit signed integer variables data-typed as "signed long". "double" and "signed long" (or just "long") are C/C++ data types. Both conversion factors as copied or derived from STEP_DIV and STEP_MUL should be stored in "double" data type variables so they can be used as multipliers to facilitate accurate conversions between double-precision microns and 32-bit integer microsteps.
In the C/C++ examples below "status_data_block" is the address of the data returned by the 's' command, and "double" is the data type for double-precision floating-point variables.

```
/* define both conversion factors as double-precision floating
point variables */
double um2usCF, us2umCF;
```

MP-285: STEP_DIV contains the microsteps/micron conversion factor. STEP_MUL contains the microns/microstep conversion factor (the reciprocal of STEP_DIV) * 100.

```
/* Get microsteps/microns conversion factor */
us2umCF = (double)((unsigned short)status_data_block[24]);
/* Get microns/microstep conversion factor */
um2usCF = (double)((unsigned short)status_data_block[26]) / 100;
```

For example, if the controller is configured for an MP-285/M micromanipulator or derived device (3DMS-285 or MP-78 stage, or MOM or SOM objective mover), then STEP_DIV contains 25 and STEP_MUL contains 4 (0.04 (the actual reciprocal of STEP_DIV * 100).

If the controller is configured for an MT-800 XY Translator, then STEP_DIV contains 20 and STEP_MUL contains 5 (0.05 * 100).

MP-285A: Both STEP_DIV and STEP_MUL contain the distance travelled by ten microsteps, expressed in nanometers (where 1 nanometer = 0.001 micron). To get the length of one microstep in nanometers, divide the field's value by 10 and then again by 1000. To get the number of microsteps required to move one micron, take the reciprocal of the length of one microstep (in microns).

```
/* microns/microstep conversion factor: Divide by 10 for
nanometers, then by 1000 for microns */
um2usCF = (double)((unsigned short)status_data_block[26]) / 10000;
/* microsteps/micron: Reciprocal of microns/microstep */
us2umCF = 1 / um2usCF;
```

For example, if the controller is configured for an MP-285/M micromanipulator or derived device (3DMS-285 or MP-78 stage, or MOM or SOM objective mover), then both fields contain 400, meaning 10 microsteps = 400 nanometers. The length of one microstep is therefore 400/10 = 40 nanometers, or 400/10000 = 0.04 microns. The number of microsteps needed to move one micron is 1/0.04 = 25 microsteps. Thus, the conversion factors for the MP-285/M are 0.04 microns/microstep and 25 microsteps/micron.

3. **XSPEED**: Contains an unsigned 16-bit value ("unsigned short" or "WORD") with both the Resolution (Low or High) and the speed (microns/sec) encoded within it. The Resolution is stored in the high-order bit (Bit 15), and the speed is stored in the remaining bits (Bits 14 through 0). Extracting both values can be done in the following way (C/C++):

```
/* "status_data_block" is the name of the address of the data
returned by the 's' command. "unsigned" is a data type prefix that
indicates positive numbers only */
unsigned short XSPEED, Speed, B15; /* 16-bit variables */
unsigned char Res;                 /* 8-bit variable */
/* read 16 bits from "status_data_block" at Index (offset) 28 */
XSPEED = (unsigned short)status_data_block[28];
Res    = 0;      /* assume Low Res */
Speed  = XSPEED; /* assume Low Res speed */
B15    = 0x8000; /* Bit 15 position value (32758 dec.) */
if (XSPEED >= B15)  /* if High Res . . . */
{
    Res   = 1;             /* set Res to High */
    Speed = (XSPEED - B15) /* extract High Res speed */
}
```

VERSION: Contains the version of the controller's firmware * 100. To extract the version, divide by 100 (e.g., 302 / 100 = 3.02 (3 is the major version number and 02 is the minor version)).

```
/* Get the version as a 16-bit positive integer value */
unsigned short VERSION =
           (unsigned short)status_data_block[30];
/* major version integer */
unsigned short ver_major = VERSION / 100;
/* minor version integer */
unsigned short ver_minor = VERSION % 100;
/* full version floating-point value */
float Ver = ((float)VERSION) / 100;
```

Table F-8. Error codes.

ASCII Char.	Value			Error	Description
	Dec.	Hex.	Binary		
0	48	30	00110000	SP Overrun	The previous character was not unloaded before the latest was received
1	49	31	00110001	Frame Error	A valid stop bit was not received during the appropriate time period
2	50	32	00110010	Buffer Overrun	The input buffer is filled, and CR has not been received
4	51	34	00110011	Bad Command	Input cannot be interpreted – command byte not valid

8	56	38	00111000	Move Interrupted	A requested move was interrupted by input on the serial port. This code is ORed with any other error code. The value normally returned is "<", i.e., '8' (38h) ORed with '4' (34h) = '<' (3Ch). '4' is reported on the vacuum fluorescent display.
					`'8' | '0' = '8'   (38h | 30h = 38h)`
					`'8' | '1' = '9'   (38h | 31h = 39h)`
					`'8' | '2' = ':'   (38h | 32h = 3Ah)`
					`'8' | '3' = ';'   (38h | 33h = 3Bh)`
					`'8' | '4' = '<'   (38h | 34h = 3Ch)`

Table F-9. MP-285 & MP-285A Programmed robotic move external commands.

Name	Tx/-Delay /-Rx	Ver.	Total Bytes	Byte Offset (len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Details
					Dec.	Hex.	Binary				
Download Program ('d') (to the controller)	Tx	All	2 + (n * 12) + 1	0	100	64	0110 0100	0100		'd'	Downloads a sequence of vectors to the controller to be stored in a specified program number.
				1	1-10	01-0A	0000 0001 – 0110 1010	0001 – 0010			Program number (1-byte unsigned integer): 1 – 10.
				2	1 – 99 (n)	01 – 63	0000 0001 – 0110 0011	0001 – 0099			Number of vectors (n) in the program (1-byte unsigned integer)
					v=1. Loop: While n > 0, 12 bytes are sent for Vector v						
				+1 (4)							Vector v X–axis distance in μsteps
				+5 (4)							Vector v Y–axis distance in μsteps
				+9 (4)							Vector v Z–axis distance in μsteps
					Decrement n and loop for next vector if n > 0; increment v. Else, exit loop						
				+1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	1	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator
Execute Stored Program ('k')	Tx	All	3	0	107	6B	0110 1011	0107		'k'	Executes (runs) a specified program stored on the controller.
				1	1-10	01-0A	0000 0001 – 0110 1010	0001 – 0010			Program number (1-byte unsigned integer): 1 – 10.
				2	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	1	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator
Upload Program ('u') (to the computer)	Tx	All	2 + (n * 12) + 1	0	100	64	0110 0100	0100		'u'	Uploads to the computer a sequence of vectors stored in controller's specified program number.
				1	1-10	01-0A	0000 0001 – 0110 1010	0001 – 0010			Program number (1-byte unsigned integer): 1 – 10.
				2	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1 + (n	0	1-99	01-63	0000 0001	0001			Number of vectors in the program

Name	Tx/-Delay/-Rx	Ver.	Total Bytes	Byte Offset (len.)	Value			Alt-key-pad	Ctrl-char	ASCII def./-char.	Details
					Dec.	Hex.	Binary				
			* 12) + 1		(n)		– 0110 0011	– 0099			(1-byte unsigned integer), referenced in this table as “n”.
					v=1. Loop: While n > 0, 12 bytes are returned for Vector v						
				+1 (4)							Vector v X–axis distance in µsteps
				+5 (4)							Vector v Y–axis distance in µsteps
				+9 (4)							Vector v Z–axis distance in µsteps
					Decrement n and loop for next vector if n > 0; increment v. Else, exit loop						
				+1	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator
Continue After Pause (‘e’)	**Tx**	All	2	0	101	65	0110 0101	0101		**‘e’**	Command
				1	13	0D	0000 1101	0013	^M	<CR>	Terminator
	Rx		1	0	13	0D	0000 1101	0013	^M	<CR>	Task-completion indicator

NOTES:

1. “Download” means sending a program from the computer to the controller. “Upload” means sending a program from the controller to the computer.
2. Each vector is 36 bytes (three sets of 12 bytes, each consisting of three contiguous 32-bit signed values (4 bytes each) for X, Y, and Z, in that order).

NOTES:

(This page intentionally left blank.)

INDEX

NOTES

NOTES