# TRIO™ MP-235

## Three-Axis Motorized Micromanipulator System

With 25mm X & Y Axes, Tilting 50mm D Axis, Synthetic Z Axis, and USB Interface

## Operation Manual

Rev. 2.31k(20201124) (FW v2.3)

**2**

(The picture on the cover page shows a TRIO MP-235 ROE/controller and a TRIO MP-235/M micromanipulator. The Sutter Instrument IPA Headstage shown mounted on the micromanipulator is not included in the TRIO MP-235 Series system.)
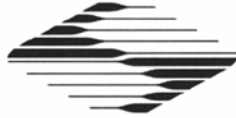
# $C\!\in$ EU Declaration of Conformity

## Application of Council Directives:
## 2014/30/EU (EMC), 2014/35/EU (LVD), and 2011/65/EU (RoHS 2)

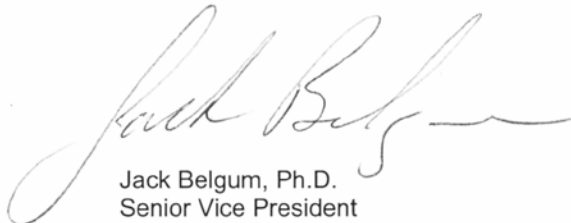| | |
|---|---|
| **Manufacturer's Name:** | Sutter Instrument Company |
| **Manufacturer's Address:** | One Digital Drive<br>Novato, CA. 94949 USA<br>Tel: +1 415 883 0128 |
| **Equipment Tested:** | **TRIO MP-235** 3-Axis Motorized Micromanipulator System |
| **Model(s):** | TRIO MP-235 system consisting of<br><br>**TRIO-235/E** (controller & ROE (Rotary Optical Encoder) for user control), with Power adapter (100-240 VAC to 24VDC) (same as, and tested as, a QUAD/E<br><br>**TRIO-235/M** (motorized micromanipulator electromechanical, subset of the QUAD/M), |

**Conforms to Standards:**

EMC Emissions:  EN 61326-1:2013, including:
EN 55011: 2009 Group 1, Class A;
EN 61000-3-2:2015, & EN 61000-3-3:2014

EMC Immunity:  EN 61000-4-2:2009,          EN 61000-4-3:2011,
EN 61000-4-4:2012,          EN 61000-4-5:2014,
EN 61000-4-6:2014,          EN 61000-4-8:2010, &
EN 61000-4-11:2004

LVD (Safety):  EN 61010-1:2010

**Tested/Verified (as a QUAD system on which the TRIO MP-235 is based) by:**

ITC Engineering Services, Inc
9959 Calaveras Road, PO Box 543
Sunol, CA 94586-0543 USA
Tel. +1 925 862 2944          Fax: +1 925 862 9013
Email: itcemc@itcemc.com       Web: www.itcemc.com

Sutter Instrument

**Test Report(s):**  20140120−01R1−Micromanipulator, 20140120-01
SI_EMC_QUAD_20160713

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 2011/65/EU (RoHS 2).

Project Engineer:

Jack Belgum, Ph.D.
Senior Vice President

## SUTTER INSTRUMENT ®

# DISCLAIMER

The **TRIO MP-235** consists of one electromechanical micromanipulator device and one ROE (Rotary Optical Encoder) with integrated controller. 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.

Unless otherwise indicated in this manual or by Sutter Instrument Technical Support for reconfiguration, do not open or attempt to repair the instrument.

Do not allow an 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 TRIO MP-235 using 110 – 240 VAC., 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

- ⚠ NOTE: There are no user-replaceable fuses in the TRIO MP-235 system.

- ⚠⚡ The TRIO MP-235 system's power supply consists of an external AC to DC switching power adapter. If the external power adapter is damaged due to a mains over or under voltage, it must be replaced.

- ⚠⚡ GROUNDING/EARTHING: Proper grounding protects the ROE/controller electronics, reduces/eliminates electromagnetic interference, and improves the safety of the system operator. The ROE/controller provides a socket (labeled GROUND) that accepts a banana plug attached to a suitably gauged insulated wire, the other end of which (alligator clip) connects to a solid, proper ground.

## Avoiding Electrical Shock and Fire-related Injury

- Always use the grounded power cord provided to connect the system's power adapter to a grounded/earthed mains outlet (3-prong). This is required to protect you from injury in the event that 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/EU 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 for interconnecting the electro-mechanical devices and ROE/controller (refer to Technical Specifications 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.

- ⚠️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 so 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.

- ⚠️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 operate if there is any obvious damage to any part of the instrument. Do not attempt to operate the instrument with the TRIO MP-235/M electromechanical manipulator shipping tape in place or severe motor damage may result. When transporting the mechanical manipulator, be sure to reinstall the shipping tape (using masking tape or equivalent only) to the original 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).

- ⚠️If the TRIO MP-235 system is used in a microinjection environment, please observe the following. As with most micromanipulation 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 tubing.

- ⚠️Take care to ensure no cables pass close to the TRIO MP-235/M electromechanical micromanipulator within the spherical movement limits of all its axes combined.

## Other

- Retain the original packaging for future transport of the instrument.

- Sutter Instrument reserves the right to change specifications without prior notice.

- Use of this instrument is for research purposes only.

## 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. Avoid contact with micropipette tips to prevent accidentally impaling oneself.

- Always dispose of micropipettes by placing them into a well-marked, spill-proof "sharps" container.

# TABLE OF CONTENTS

## TABLE OF FIGURES

# TABLE OF TABLES

# 1. INTRODUCTION

## 1.1 Structure of the TRIO MP-235 Documentation Package

The TRIO MP-235 3-Axis Micromanipulator System is comprised of a ROE/controller, a power adapter, and a TRIO MP-235/M stepper-motor-based electromechanical micromanipulator. This manual consists of four parts: This chapter, Introduction, which provides an overview and general description of the TRIO MP-235 system; Chapter 2, Installation, which describes how to install, set up, and configure all components of the system; Chapter 3, Operations, which describes how to operate the TRIO MP-235; Chapter 4, Maintenance, describes how to perform routine and other maintenance of the TRIO MP-235; and Chapter 5, Reconfiguration, describes the reconfiguration possibilities of the TRIO MP-235 system.
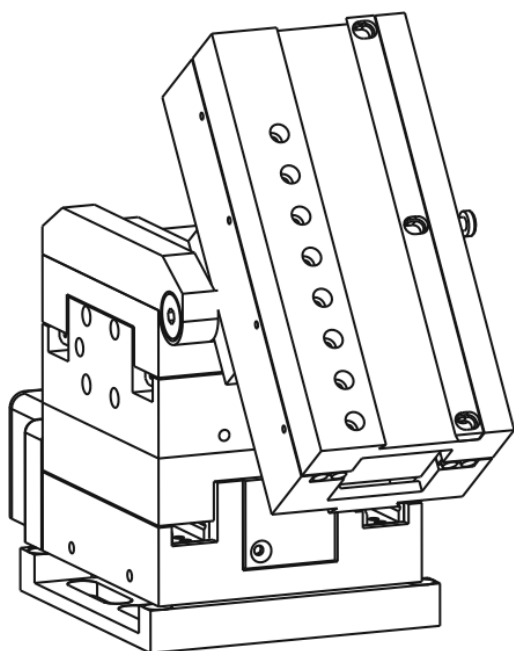
TRIO MP-235/M
ELECTROMECHANICAL
MICROMANIPULATOR
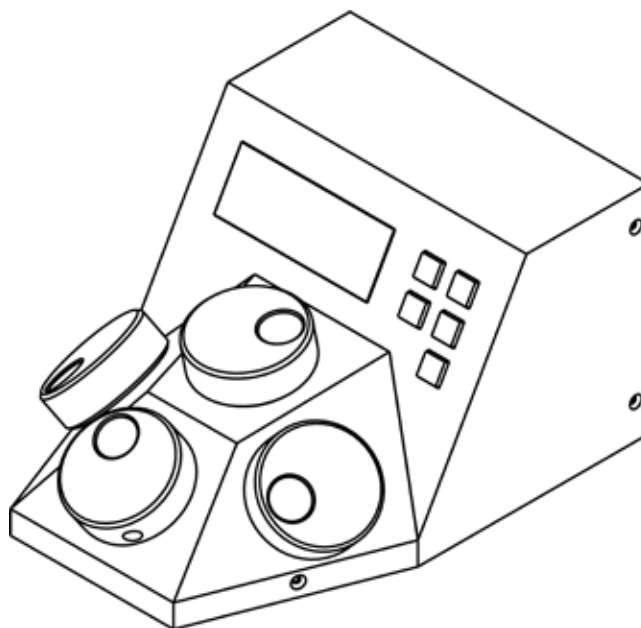
TRIO MP-235/E
ROE/CONTROLLER

Figure 1-1. The TRIO MP-235 system

## 1.2 Components of the TRIO MP-235 System

Carefully remove all components from the shipping container. In addition to this manual, the following should be included:

- TRIO MP-235 ROE Rotary Optical Encoder input device with built-in controller and external power adapter.
- TRIO MP-235/M electromechanical micromanipulator

- 25-pin DSUB cable (connects the ROE/controller to the TRIO MP-235/M electromechanical micromanipulator).
- Power adapter
- Power adapter AC mains cable appropriate for your location
- Ground/Earth cable
- USB Cable

IMPORTANT

Once the TRIO MP-235 system has been unpacked, remove the shipping tape from the various locations on the TRIO MP-235/M electromechanical micromanipulator. The shipping tape must be removed before operating the TRIO MP-235 system. If you need to transport the TRIO MP-235/M in the future, reapply 2 to 3-inch pieces of masking tape to the same locations. Once the tape has been removed, handle the TRIO MP-235/M with care. The mechanisms can be damaged if any of the axes are inadvertently moved without the tape in place.

1.3 Overview

1.3.1 Features

- Three independent axes – 50mm travel in the diagonal axis for coaxial pipette movement, and 25mm travel in X and Y axes. Travel for the virtual Z axis utilizes a cosine function of the angle chosen, using the X and D axes.
- Sub-micron 100nm resolution
- Digital display indicates coordinates in relative or absolute
- User-friendly, fanless compact controller with ROE preserves bench space
- Push button control of multiple functions – work, home, Lock, pulse and relative
- Robotic home and work-position moves for easy automated pipette exchange

1.3.2 Description

While the axes provide X and Y orthogonal motion typical of most motorized manipulators, Sutter has introduced a diagonal axis with the **TRIO MP-235** so one can move the electrode coaxially at the exact desired angle of approach. This axis also significantly extends the range of travel (50mm) for the system.

The **TRIO MP-235's** ROE provides fine control of electrode position and the rate of rotation of ROE dials for each axis determines the speed of travel. The finest step size is less than 100nm. Five conveniently located buttons on the ROE provide control of all the basic functions you will need in normal operation (Work, Home, Lock, Relative, and Pulse).

Press and hold WORK (for 3 seconds) to quickly store a work position, tap HOME to move all axes to an initial location that is useful for changing electrodes, or press and hold the HOME button (for 3 seconds) to memorize a new HOME position.

When you are ready to record data, the motor drive electronics can be suppressed by pressing the LOCK button. In the LOCK mode, the display turns red and ROE input is locked out to avoid any accidental motion.

Pressing and holding the RELATIVE button for three seconds at any location causes the display coordinates to all zeroes. When activating relative mode, the display turns blue. To return to viewing the absolute coordinates, tap the RELATIVE button to toggle back.

Finally, tapping the PULSE button causes a 3µm advance in the diagonal. This rapid burst of forward motion can assist in sharp electrode cell penetration.

All the electronics, except for a small power supply, are housed within the **TRIO MP-235** ROE and no separate controller or computer is required.

External computer control of the **TRIO MP-235** is possible via the USB connector mounted on the controller/ROE's rear panel. The controller's internal software is programmed with a defined set of commands allowing for a wide range of micromanipulator/stage movements as programmed in software residing in an external computer connected via USB.

## 2. INSTALLATION

When installing the TRIO MP-235 system for the first time, it is recommended that the components of the system be installed in the following order: TRIO MP-235/M electromechanical micromanipulator first, followed by the TRIO MP-235/E ROE/Controller.

### 2.1 Mounting Instructions

The following sections describe how to mount the TRIO MP-235/M manipulator to a stand using the mounting adapter plate, how to adjust the pipette angle and how to mount different headstages.

#### 2.1.1 Mounting the TRIO MP-235/M to the Stand or Platform

The TRIO MP-235/M attaches to the mounting adapter plate using four M3.5x6 hex head locking screws.
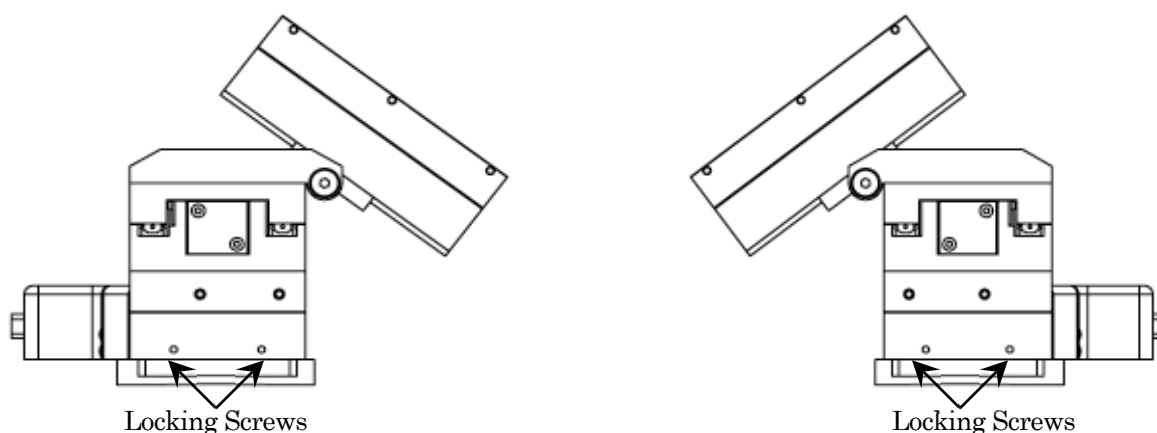


Figure 2-1. Side view of TRIO MP-235/M showing mounting adapter plate and lock screws.

The TRIO MP-235/M is shipped with the adapter plate in place. It is attached using four tapered pegs, along with four locking screws.

To remove it, first loosen the four hex screws that secure the manipulator to the pegs in the adapter plate. The rear pair is in a similar location in the back of the manipulator. Once the locking screws are sufficiently loosened, lift the TRIO MP-235/M upwards from the adapter plate.

Before attaching the adapter plate to the TRIO MP-235/M, you need to decide where to position the manipulator on your stand/platform. The stand can be any flat surface carrying ¼-20, 10-32, or M6 holes on one-inch centers (such as a Sutter Instrument MT-series stand or MD series platform).

Examine the space of the platform onto which installation is to take place. Attach the control cable to TRIO MP-235/M and move the entire unit around on the platform until the precise desired position is determined. A small bag containing the necessary hardware to attach the TRIO MP-235 to the stand is included.
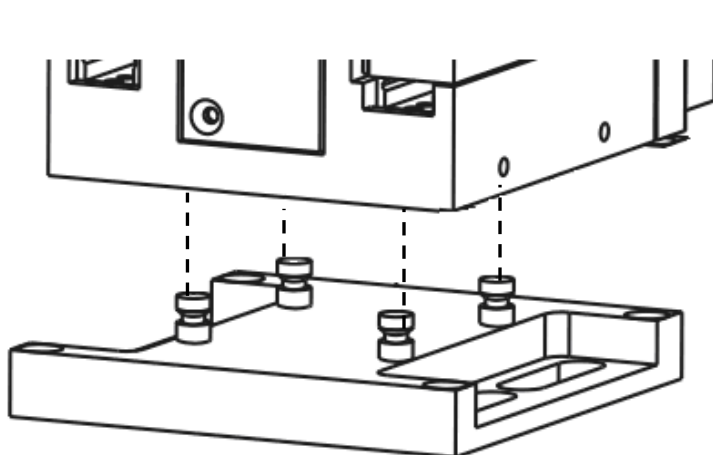


Figure 2-2. Mounting the TRIO MP-235/M on the Adapter Plate

Once the plate is mounted, align the pegs on top of the plate with the holes in the manipulator, push the X-axis firmly onto the plate, and re-tighten the locking hex set screws.

## 2.2 Headstage Mounting

Sutter IPA headstage, Axon headstages 203B or CV-7, and the Heka EPC-10 headstage have an integral dovetail that fits directly into the rotary dovetail slide bracket on the TRIO MP-235/M. The dovetail slide bracket on the TRIO MP-235/M also supports older Axon and Heka headstages when using the 4'' dovetail extension.

Rod-mounted headstages and micro tools are accommodated by the use of a rod clamp that fits into the dovetail (not shown). All the headstage adapters and mounting hardware are included with the manipulator and are shipped in a zip lock plastic bag.

## 2.3 Other Accessories

One or more accessories may have been ordered and received for mounting the TRIO MP-235/M and/or modifying the headstage mount to the manipulator (i.e., rotating base, microscope stage mount, gantry, dovetail extension, etc.). Setup of these accessories is normally covered in documentation accompanying the accessory.

## 2.4 Electrical Connections and Initial Operating Instructions

Initially, you may want to simply connect the TRIO MP-235/M micromanipulator and the ROE/Controller together and try some gross movements in order to get a feel for the controls and how to make simple movements. It is perfectly acceptable to set the manipulators in the middle of a bench top, make all electrical connections and then observe each unit's movement by eye.

⚠ CAUTION: Unless the TRIO MP-235/M micromanipulator electromechanical baseplate is firmly bolted down to a breadboard or solidly to a firm surface, the TRIO MP-235/M is likely to tip over when fully extending all of its axes, especially if its loaded with a headstage that extends beyond the TRIO MP-235/M's current center of gravity.

Upon deciding to directly install the TRIO MP-235 system in your rig, it is useful to follow the initial setup procedure to learn how to move the units to allow easy access to the mounting screws.

1. With the power switch on the back of the ROE in the OFF (0) position, connect the power adapter's 24VDC cable to the POWER receptacle.
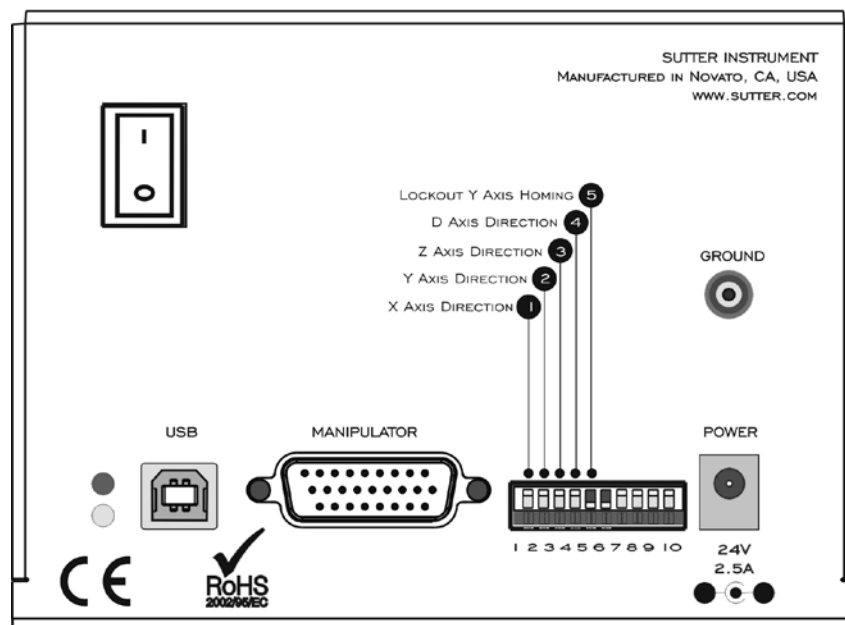


Figure 2-3. Rear of TRIO MP-235 ROE/Controller cabinet

2. With the power OFF (rear panel switch in the "0" position), connect a well-grounded/earthed wire to the GROUND banana plug receptacle.
3. With the power OFF, connect the male end of the DB-HD-26 cable to the MANIPULATOR connector on the ROE, the other end of which is connected to the TRIO MP-235/M micromanipulator electromechanical. **(See cautionary note below.)**
4. Verify that the ten switches on the rear of the ROE are set as desired.
5. Power up the system by moving the power switch on the rear of the ROE to the "1" position.

*⚠ *CAUTION:  NEVER CONNECT OR DISCONNECT THE ROE/CONTROLLER FROM THE TRIO MP-235/M WHILE THE POWER IS ON!*

## 2.5 ROE/Controller Rear Panel Controls and Configuration

### 2.5.1 Power Switch

The power switch for the TRIO MP-235 system is located on the rear panel of the ROE/controller. At power up, the microprocessor in the ROE/controller scans the attached equipment and configures the system accordingly.

### 2.5.2 Configuration Switches
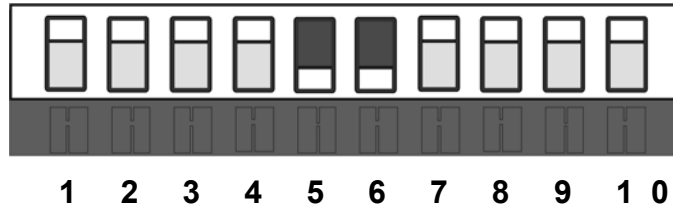


Figure 2-4. Configuration switches on rear of TRIO MP-235 ROE/Controller unit (switch positions shown are factory defaults).

### *2.5.2.1 Switches 1, 2, 3, and 4*

The first four switches set the directionality for each of the four axis-movement knobs.

Table 2-1. Configuring the directionality of each axis-movement knob.

| Switch # | Axis | Knob Rotation Directionality for Forward (+) Movement | |
|---|---|---|---|
| | | Clockwise | Counterclockwise |
| 1 | X | OFF (Up)* | ON (Down) |
| 2 | Y | OFF (Up)* | ON (Down) |
| 3 | Z | OFF (Up)* | ON (Down) |
| 4 | D | OFF (Up)* | ON (Down) |

* Factory default (typical setting for right-hand-mounted manipulator).

### *2.5.2.2 Switch 5: Locking Out Y Movement During Homing*

Configures whether the Y axis is locked out while homing.

Table 2-2. Configuring the Homing Y-Movement Lock Out.

| Switch # | Homing Y Movement Lock Out | |
|---|---|---|
| | Enabled | Disabled |
| 5 | OFF (Up) | ON (Down)* |

* Factory default (recommended normal operation setting).

### 2.5.2.3 Switch 6: Sensor Test (Firmware < v2.2)

Table 2-3. Configuring the Sensor Test.

| Switch # | Sensor Test | |
|---|---|---|
| | **Enabled \*\*** | **Disabled** |
| **6** | OFF (Up) | **ON (Down)\*** |

\* Factory default (do not change unless requested to by Sutter Instrument Technical Support).

*** CAUTION: To avoid damage to the micromanipulator or stage, DIP Switch 6 (Sensor Test) must always be set to OFF (DOWN).**

### 2.5.2.4 Switch 6: Calibration Homing on Power On (Firmware v2.2+)

Configures whether calibration homing occurs or not on power on (FW v2.2+).

Table 2-4. Calibration Homing on Power On (Firmware v2.2+).

| Switch # | Calibration Homing on Power On | |
|---|---|---|
| | **None (No calibration)** | **Calibration occurs** |
| **6** | OFF (Up) | **ON (Down)\*** |

\* Factory default (recommended normal operation setting).

### 2.5.2.5 Switches 7 through 10

These four switches are reserved for all controllers programmed with a firmware version below v2.2. For firmware v2.2 and above, the switches are defined in the following paragraphs.

### 2.5.2.6 Switch 7: Speed of Manual Movement (Firmware v2.2+)

This switch determines the speed at which movement takes place when manipulated by the ROE rotational knobs for any axis. The OFF (up) position configures the speed to "normal" (factory default). Set the switch to ON (down) for slower speed in situations where the micromanipulator or stage is burdened with heavier loads. This speed setting affects only manual movement using the rotation knobs of the ROE; moves initiated by external command (see Chapter 4) are not affected.

Table 2-5. Speed of manual movement (Firmware v2.2+).

| Switch # | Speed of Manual Movement | |
|---|---|---|
| | **Normal** | **Slower\*\*** |
| **7** | **OFF (Up)\*** | ON (Down) |

\* Factory default (recommended normal operation setting).

\*\* Use the ON (down) setting for slower speed when the micromanipulator or stage is burdened with heavier loads.

### *2.5.2.7 Switch 8: Reserved (Firmware <= v2.2)*

This switch is reserved for future use. The factory default position is OFF (up).

### *2.5.2.8 Switch 8: ROE Axis Movement Knobs Active – All vs. D Only (Firmware v2.3+)*

This switch determines if all axis-movement knobs of the ROE are active or only the D-axis knob.

Table 2-6. ROE Axis movement knobs active – all vs. D only (Firmware v2.3+).

| Switch # | ROE Axis Movement Knobs Active – All vs. D Only | |
| --- | --- | --- |
| | **All Active (X, Y, Z, & D)** | **D Only (X, Y, & Z inactive)\*\*** |
| 8 | **OFF (Up)\*** | ON (Down) |

\* Factory default (recommended normal operation setting).

\*\* Use the ON (down) setting for when it is desired to have only the D-axis movement knob active, while the rest (X, Y, & Z) are inactive.

### *2.5.2.9 Switch 9: Y-Axis Travel Length (Firmware v2.2+)*

This switch informs the ROE/controller as to the travel length of the Y axis in the connected micromanipulator eletromechanical or stage. The length of the Y axis on the TRIO MP-235/M electromechanical micromanipulator is 25mm, so Switch 9 should be set to OFF (up) (factory default for a standard TRIO MP-235 system). For a variant model with 12.5mm of travel in the Y axis, switch 9 must be set ON (down).

Table 2-7. Y-Axis travel length (Firmware v2.2+).

| Switch # | Y-Axis Travel Length | |
| --- | --- | --- |
| | **25mm** | **12.5mm \*\*** |
| 9 | **OFF (Up)\*** | ON (Down) |

\* Factory default (recommended normal operation setting).

\*\* Use the ON (down) setting for half-length Y axis (12.5mm).

*CAUTION: Always be certain that the position of Switch 9 correctly matches the physical length of travel of the Y axis on the connected device. Setting the switch to the OFF (up) position for 25mm could result in equipment damage if the attached device's Y axis length of travel is less than 25mm.*

### *2.5.2.10 Switch 10: X Axis Travel Length (Firmware v2.2+)*

This switch informs the ROE/controller as to the travel length of the X axis in the connected micromanipulator eletromechanical or stage. The length of the X axis on the TRIO MP-235/M electromechanical micromanipulator is 25mm, so Switch 10 should be set to OFF (up) (factory default for a standard TRIO MP-235 system). For a variant model with 12.5mm of travel in the X axis, switch 10 must be set ON (down).

Table 2-8. Length of X Axis (Firmware v2.2+).

| Switch # | Length of X Axis | |
|---|---|---|
| | 25mm | 12.5mm ** |
| 10 | OFF (Up)* | ON (Down) |

* Factory default (recommended normal operation setting).

** Use the ON (down) setting for half-length (12.5mm) X axis.

*CAUTION: Always be certain that the position of Switch 10 correctly matches the physical length of travel of the X axis on the connected device. Setting the switch to the OFF (up) position for 25mm could result in equipment damage if the attached device's X axis length of travel is less than 25mm.*

# 3. OPERATIONS

## 3.1 Main Controls and Indicators on the ROE/Controller
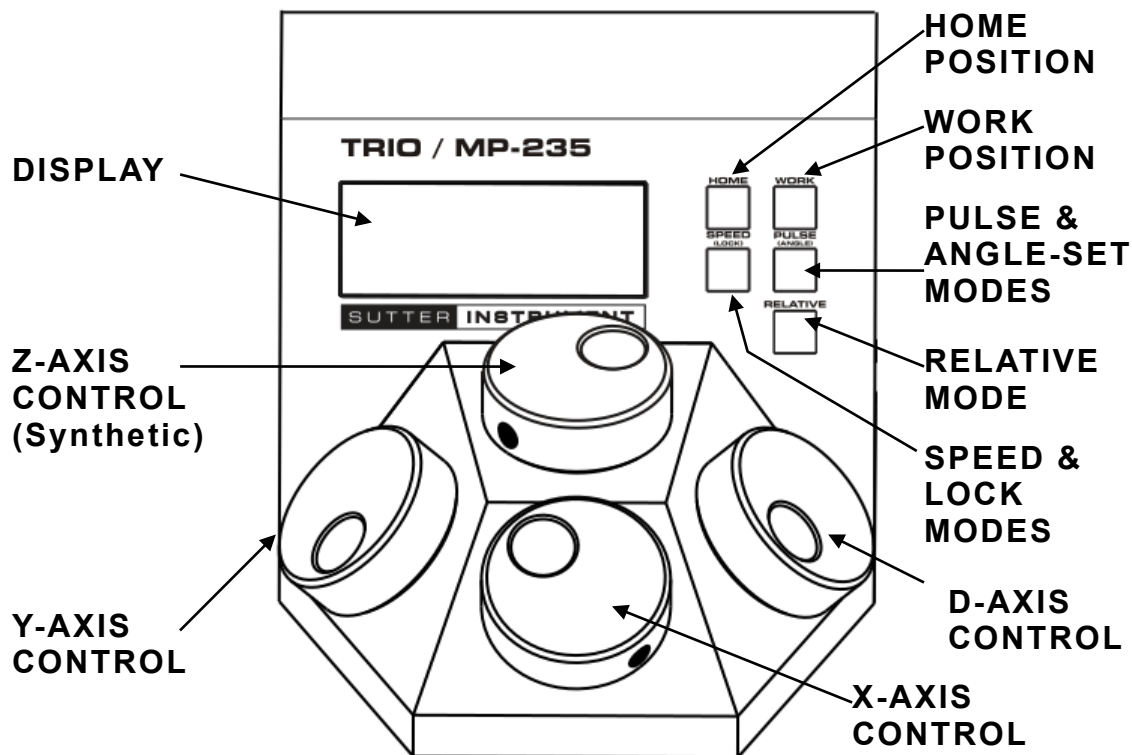


Figure 3-1. Front view of the TRIO MP-235 ROE/Controller

## 3.2 Display

### 3.2.1 Initial Startup

```
      SUTTER
  INSTRUMENT CORP.
```
(Text in Green)

Figure 3-1. LCD Display showing startup screen.

When starting the TRIO MP-235 system for the first time or if the HOME position has not yet been defined (saved), the values of all four axes will be 1,000 micrometers (microns).

```
  TRIO / MP-235
    REV 2.3
```
(Text in Green)
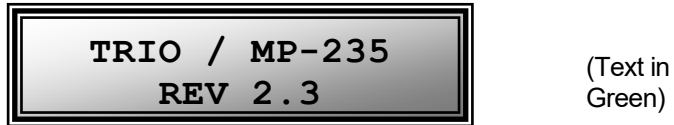
Figure 3-2. Startup screen

```
X= 1000 Y= 1000
D= 1000 >=
```
(Text in Green)

Figure 3-3. Factory default startup (Home) position

## 3.3 Control Operations

### 3.3.1 Maximum Positive Position Values:

Move the dial of an axis clockwise until its position value stops incrementing. The following table lists the maximum position value (in microns) for each axis.

Table 3-1. Maximum positive position value of each axis

| Axis | Maximum Position Value (in microns) |
|------|-------------------------------------|
| X | 25,000 |
| Y | 25,000 |
| D | 50,000 |

```
X=25000   Y=25000
D=50000   >=
```
(Text in Green)
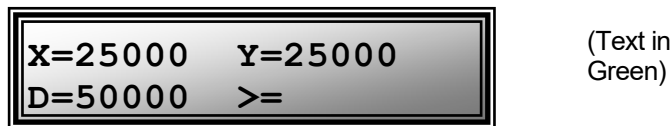
Figure 3-4. Maximum positive values

### 3.3.2 Setting Position for HOME or WORK

To set position, hold down HOME or WORK button for 3 seconds until beep sounds.

### 3.3.3 Setting the Angle of the Pipette/Headstage Holder

To change the angle of the holder, first loosen the set screw at the top of the rotary dovetail bracket, rotate the holder to the desired angle, and then retighten the set screw.

Measure the angle of the holder. (Tip: Many smart phones have an app with a level that can assist the user.)

### 3.3.4 Operating the Virtual Z Axis

The TRIO MP-235 consists of three physical axes, X, Y and D. A cosine function utilizing X and D axes and the angle of the holder has been implemented to create a virtual Z-axis. Best results are obtained when using it between 10° and 90°.

Zero (0°) is set with the diagonal being parallel to the table and 90° is set with the diagonal being perpendicular to the table.

To set the angle measured above, on the ROE hold PULSE down for 3 seconds. The screen will be red until the display indicates in green: "Select the angle in use (0-90)".

Use the D dial on the ROE to set the value of the angle. Once this value is dialed in, do not touch the ROE knob for 8-10 sec. The virtual D angle will now be set.

### 3.3.5 Moving to the Home Position (HOME)

```
X=       0  Y=  1524
D=       0  >=       0
```
(Text in Red)
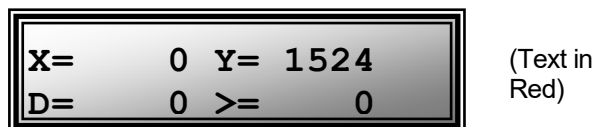
Figure 3-5. Moving to Home position (screen is amber while moving)

If the Home position has not yet been defined and saved, the Home position values for all axes will default to 1,000 microns, as shown in the following figure.

```
X=  1000    Y=  1000
D=  1000    >=
```
(Text in Green)
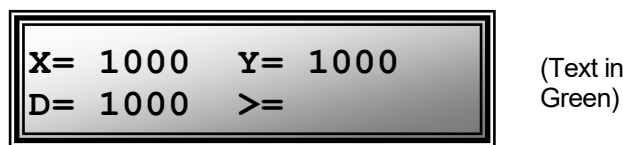
Figure 3-6. Factory default Home position

If the Home position has been previously defined (saved), pressing HOME will make a move to the defined home position (see example in the following figure).

```
X=1868    Y=1524
D=1686    >=25
```
(Text in Green)
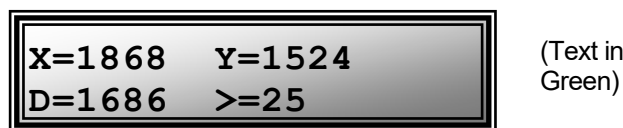
Figure 3-7. Example Home position defined and saved

To move to the Home position, press HOME. If the current position before pressing HOME is greater than the Home position, the movement will be as follows:

NOTE: Movement to the Home position works only if X coordinates of the HOME position are less than the WORK position.

1. Movement begins by retracting the D axis (at the angle currently set) away from the sample.

2. Movement then continues along the X axis toward the Home position.

3. The final movement is along the Y-axis towards the operator and away from the microscope.

NOTE: Step 3 occurs only if "Y-Lockout" is disabled. Otherwise, no movement along the Y-axis occurs.

### 3.3.6 Moving to the Work Position (WORK)

```
X=2868   Y=2524
D=1686   >=25
```
(Text in Green)
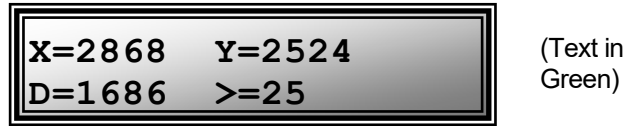
Figure 3-8. Example Work position

To move to the Work position, press the WORK button. If the current position before pressing WORK is less than the Work position, the movement will be as follows:

1. Movement travels along the Y-axis away from the operator and towards the microscope.

2. Movement is then made along the X axis toward the sample.

3. Travel then continues along the diagonal until reaching its end-of-travel point.

NOTE: Step 1 occurs only if "Y-Lockout" is disabled. Otherwise, movement begins with Step 2.

### 3.3.7 Setting Absolute/Relative Coordinates Mode (RELATIVE)

The RELATIVE button toggles between Relative and Absolute coordinate systems. The default coordinate system on power up is Absolute, with the coordinates on the screen shown in green. To switch to relative coordinates, press the RELATIVE button once. To reset the current position to all zeroes, depress the RELATIVE for 3 seconds or until a beep is heard, and then release the button. This resets the current position to all zeroes.

Press RELATIVE once (briefly for < 2 sec.)

```
X=- 893   Y= 1524
D= 1686   >= 30
```
(Text in Blue)
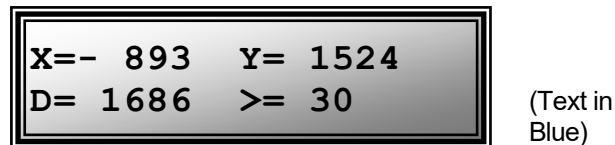
Figure 3-9. Relative mode

Depress RELATIVE for 3 sec. or until beep sounds

```
X=-     0   Y=     0
D=     0   >=     0
```
(Text in Blue)
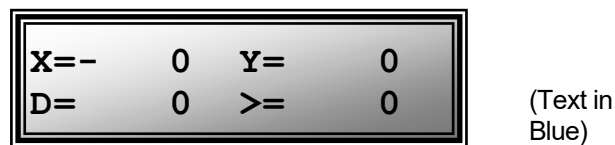
Figure 3-10. Relative mode

Pressing RELATIVE briefly while in Relative mode, returns displayed coordinates back to Absolute mode

```
X=- 893   Y= 1524
D= 1686   >= 30
```

(Text in Green)

Figure 3-11. Absolute mode

### 3.3.8 Mode Indications

The TRIO MP-235 system has three modes of operation: Absolute coordinates, Relative coordinates, and Lock mode. The display turns color for each specific mode, as shown in the following table.

Table 3-2. Screen colors and modes

| Screen Color | Mode | Example |
|---|---|---|
| Green | Absolute Coordinates | X= 1868 Y= 1524<br>D= 1686 >= 27 |
| Blue | Relative Coordinates | X=-    0 Y=    0<br>D=    0 >=    0 |
| Red | Knobs disabled during move to Home or Work position, while in Lock mode. | X=    0 Y= 1524<br>D=    0 >=    0 |

### 3.3.9 Speed Control and ROE Knob Movements (SPEED)

The rate at which the ROE axis knobs move the electromechanical can be adjusted with the SPEED button. Each press of the button cycles through four speeds: 0 (normal) through 3 (fastest).

### 3.3.10 Movement Knobs Disabling and Lock Mode ([SPEED]/LOCK)

Axis-movement knobs are disabled during movements to Home, Work, or while in Lock Mode (display is in red).

### 3.3.11 Pausing Home Movements (HOME (while moving to Home))

After Move to Home has been initiated, and while the move is in progress, pressing HOME a second time pauses the manipulator. Pressing HOME again resumes movement.

### 3.3.12 Pausing Work Movements (WORK (while moving to Work))

After Move to Work has been initiated, and while the move is in progress, pressing WORK a second time pauses the manipulator. Pressing WORK again resumes movement.

### 3.3.13 Pulse Mode and D-Axis Movement (PULSE)

Pulse mode advances the D axis in 2.85 $\mu$m steps. Each press of the PULSE button increments the Diagonal axis by one 2.85-$\mu$m step beyond the current position. This feature can be used to penetrate tough or resistant tissue.

# 4. EXTERNAL CONTROL

## 4.1 General

Controlling the TRIO MP-235 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 TRIO MP-235 controller/ROE. The USB device driver for Windows is downloadable from Sutter Instrument's web site (www.sutter.com). The TRIO MP-235 requires Sutter Instrument's USB CDM (Combined Driver Model) Version 2.10.00 or higher. The CDM device driver 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 TRIO MP-235. 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.

## 4.2 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 4-1. USB-VCP interface serial port settings.

| Property | Setting |
|---|---|
| Data ("Baud") Rate (bits per second (bps)) | 57600 |
| Data Bits | 8 |
| Stop Bits | 1 |
| Parity | None |
| Flow Control | None |

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.

## 4.3 Protocol and Handshaking

Command sequences do not have terminators. 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.

## 4.4 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 not used. 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 transmitted to the controller must contain an unsigned binary value. Attempting to code command sequences as "strings" is not advisable. Any command data returned by the controller should be initially treated as a sequence of unsigned byte values upon reception. 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 TRIO MP-235, all axis position values (number of microsteps) are stored as "unsigned long" 32-bit positive-only values, and each is transmitted and received to and from the controller as four contiguous bytes.

## 4.5 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 32-bit values ("long" (or optionally, "signed long"), or "unsigned long" for C/C++; "I32" or "U32" for LabVIEW). "Unsigned" means the value is always positive; negative values are not allowed. The positive-only values can also be stored in signed type variables, in which case care must be taken to ensure that only positive values are exchanged with the controller.

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.

## 4.6 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 4-2. Microns/microsteps conversion.

| System/Device | From/To Units | Conversion Factor (multiplier) |
|---|---|---|
| TRIO MP-235 with TRIO MP-235/M micromanipulator | $\mu$steps $\rightarrow$ $\mu$m | 0.09375 |
| | $\mu$m $\rightarrow$ $\mu$steps | 10.66666666667 |

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 "unsigned long" (C/C++), "uint32" (MATLAB), or "U32" (LabVIEW) integer (positive only) value. When converting to microns, type the result as a "double" (C/C++, MATLAB) or "DBL" (LabVIEW) 64-bit double-precision floating-point value.

## 4.1 Ranges and Bounds

Table 4-3. Ranges.

| Device | Axis | Millimeters | Microns | Microsteps |
|---|---|---|---|---|
| TRIO MP-235/M | X | 0 – 25 | 0 – 25,000 | 0 – 266,667 |
| | Y | 0 – 25 | 0 – 25,000 | 0 – 266,667 |
| | D | 0 – 50 | 0 – 50,000 | 0 – 533,334 |

## 4.2 Commands

### 4.2.1 Get Current Position ('c' or 'C') Command

This command is used to obtain the current position (X, Y, & D coordinates) of the manipulator. The command sequence consists of one byte as shown in the following table. Thirteen bytes are returned: X, Y, & D position values in microsteps (4 bytes each), and completion indicator (1 byte).

Table 4-4. Get Current Position command.

| Command | Tx/- Delay/- Rx | Ver. | Total Bytes | Byte Offset (Len.) | Value Dec. | Value Hex. | Value Binary | Alt- key- pad # | Ctrl- char | ASCII def./- char. | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Get Current Position and Angle ('c' or 'C') | Tx | All | 1 | 0 | 99 or 67 | 63 or 43 | 0110 0011 or 0100 0011 | 0099 or 0043 | | 'c' or 'C' | Returns the current positions ($\mu$steps) of X, Y, & D axes |
| | Rx. | All | 14 | | Three 4-byte (32-bit) values (current positions in $\mu$steps of X, Y, & D), + 1 byte for completion indicator. See *Ranges* table for minimum and maximum values. | | | | | | |
| | | | | 0 (4) | | | | | | | X pos. in $\mu$steps |
| | | | | 4 (4) | | | | | | | Y pos. in $\mu$steps |
| | | | | 8 (4) | | | | | | | D pos. in $\mu$steps |
| | | | | 12 | 13 | 0D | 0000 1101 | | ^M | <CR> | Completion indicator |

## 4.2.2 Move to Controller-Defined HOME Position ('h') Command

This command moves to the position last defined by the HOME key. Movement is equivalent to pressing the HOME key (D first, followed by X & Y together, as if moving away from a work position).

Table 4-5. Move to controller-defined HOME position ('h') 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 HOME Position ('h') | Tx | | 1 | 0 | 104 | 68 | 0110 1000 | 0104 | | 'h' | Moves to the position saved for the controller's HOME button. |
| | Rx | | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |

## 4.2.3 Move to Controller-Defined WORK Position ('w') Command

This command moves to the position defined last defined for the WORK key. Movement is equivalent to pressing the WORK key (X & Y together first, followed by Z, as if moving towards a work position).

Table 4-6. Move to controller -defined WORK position ('w') 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 WORK Position ('w') | Tx | | 1 | 0 | 119 | 77 | 0111 0111 | 0119 | | 'w' | Moves to the position saved for the controller's WORK button. |
| | Rx | | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |

## 4.2.4 Move to Specified X-Axis Position ('x' or 'X') Command

This command moves to a specified position for only the X-axis.

Table 4-7. Move to specified X-axis position ('x' or 'X') 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 X axis Position ('x' or 'X') | Tx | All | 5 | 0 | 120 or 90 | 78 or 5A | 0111 1000 or 0101 1010 | 0120 or 0090 | | 'x' or 'X' | Move X axis to specified position (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | X $\mu$steps |
| | Rx | | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |

## 4.2.5 Move to Specified Y-Axis Position ('y' or 'Y') Command

This command moves to a specified position for only the Y-axis.

Table 4-8. Move to specified Y-axis position ('y' or 'Y') 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 Y axis Position ('y' or 'Y') | Tx | All | 5 | 0 | 121 or 91 | 79 or 5B | 0111 1001 or 0101 1011 | 0121 or 0091 | | 'y' or 'Y' | Move Y axis to specified position (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | Y $\mu$steps |
| | Rx | | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |

## 4.2.6 Move to Specified D-Axis Position ('d' or 'D') Command

This command moves to a specified position for only the D-axis.

Table 4-9. Move to specified D-axis position ('d' or 'D') 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 D axis Position ('d' or 'D') | Tx | All | 5 | 0 | 100 or 68 | 64 or 44 | 0110 0100 or 0100 0100 | 0100 or 0068 | | 'd' or 'D' | Move D-axis to specified (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | D $\mu$steps |
| | Rx | | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |

## 4.2.7 External Command 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 Send/Receive Buffers:** Clearing (purging) the transmit and receive buffers of the I/O port immediately before sending any command is recommended.

4. **Positions in Microsteps and Microns:** All positions sent to and received from the controller are in microsteps ($\mu$steps). See *Microns/microsteps conversion* table) for conversion between $\mu$steps and microns (micrometers ($\mu$m)).

*Declaring position variables in C/C++:*
```
/* current position for X, Y, & D */
unsigned long  cp_x_us, cp_y_us, cp_d_us; /* microsteps */
double         cp_x_um, cp_y_um, cp_d_um; /* microns */
/* specified (move-to) position for X, Y, & D */
unsigned long  sp_x_us, sp_y_us, sp_d_us; /* microsteps */
double         sp_x_um, sp_y_um, sp_d_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 the TRIO MP-235/M based config. */
double us2umCF = 0.09375;        /* microsteps to microns */
double um2usCF = 10.66666666667; /* microns to microsteps */
```

*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 & D, 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 must include positive values only for positions – negative positions must never be specified. All positions are absolute as measured from the physical beginning of travel of a device's axis. In application programming, it is important that positional values be checked (>= 0 and <= max.) to ensure that a negative absolute position is never sent to the controller and that end of travel is not exceeded. All computational relative positioning must always resolve to accurate absolute positions.

*Declaring minimum and maximum absolute position variables in C/C++:*
```
/* minimum and maximum positions for X, Y, & D */
double min_x_um, min_y_um, min_d_um;  /* minimum microns */
double max_x_um, max_y_um, max_d_um;  /* maximum microns */
```
*Set minimum and maximum absolute positions for each axis – see Ranges & Bounds table.*
```
/* initialize all minimum positions in microns*/
min_x_um = 0;
min_y_um = 0;
min_d_um = 0;
/* initialize all maximum positions in microns*/
/* TRIO MP-235/M */
max_x_um = 25000;
max_y_um = 25000;
max_d_um = 50000;
```

6. **Absolute Positioning System Origin:** The Origin is set to a physical position of travel to define absolute position 0. The physical Origin position is fixed at beginning of travel (BOT). This means that all higher positions (towards end of travel (EOT)) are positive values; there are no lower positions and therefore no negative values are allowed.

7. **Absolute vs. Relative Positioning:** Current position ('c') and move commands always use absolute positions. All positions can be considered "relative" to the Origin (Position 0), but all are in fact absolute positions. Any position that is considered to be "relative" to the current position, whatever that might be, can be handled synthetically by external programming. However, care should be taken to ensure that all relative position calculations always result in correct positive absolute positions before initiating a move command.

*Declaring relative position variables in C/C++:*
```
/* relative positions for X, Y, & D */
double rp_x_um, rp_y_um, rp_d_um;  /* microns */
/* initialize all relative positions to 0 after declaring them */
rp_x_um = rp_y_um = rp_d_um = 0;
```

*Enter any positive or negative value for each relative position (e.g., rp_x_um = 1000; rp_y_um = 500; rp_d_um = -500 … etc.*

*For each axis, check to make sure that the new resultant absolute position (to which to move) is within bounds. Reset the relative position to 0 if not. If relative value is negative, its positivized value must not be greater than the current position. Otherwise, if positive, adding current position with relative position must not exceed the maximum position allowed. If out of bounds, resetting relative position to 0 allow the remaining conversions and movement to resolve without error.*

```
/* check to make sure that relative X is within bounds */
if ( ( rp_x_um < 0 && abs(rp_x_um) > cp_x_um ) ||
     (cp_x_um + rp_x_um > max_x_um) )   /* out of bounds? */
    rp_x_um = 0; /* yes, so reset relative pos. to 0 */
```

*Repeat the above bounds check for each of the remaining axes.*

*For each axis, calculate new absolute position in microns and then convert to microsteps before issuing a move command.*

```
/* convert X relative position to absolute position */
sp_x_um = cp_x_um + rp_x_um; /* add relative pos. to current pos. */
/* convert new absolute X position in microns to microsteps */
sp_x_us = sp_x_um * um2usCF;
```

*Repeat for each of the remaining axes as required before issuing 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 in microsteps are always positive, so data type must be an "unsigned" integer that can hold 32 bits of data. 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 an unsigned 32-bit integer ("unsigned long" in C/C++, "uint32" in MATLAB, "U32" in LabVIEW, etc.).

    Position values in microns (micrometers or $\mu$m) should be data typed as double-precision floating point variables ("double" in C/C++ and MATLAB, "DBL" 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. **Angle Setting & Movement:** Although the set angle command allows for a range of 0° to 90°, the effective range that allows full movement is 1° to 89° (>0° and <90°). If 0° or 90°, Z or X axis fails to move, causing single- and multi-axis movement commands to fail. The ideal range for smooth movement is 10° to 80°. Factory default is 30°.

# 5. MAINTENANCE

Routine cleaning of the TRIO MP-235 system is required to prevent excessive dust accumulations. Wipe all exterior surfaces with a dry, soft, cotton cloth.

Periodically inspect all cables and connections to make sure that all connections are made well and that all connectors are well and evenly seated.

# 6. RECONFIGURATION

## 6.1 Changing the Rotary Knob Functions on the ROE/Controller

The axis motor assignment of each axis control knob on the ROE can be changed by opening the ROE/Controller cabinet as seen in the figure below and changing cables to appropriate connectors.
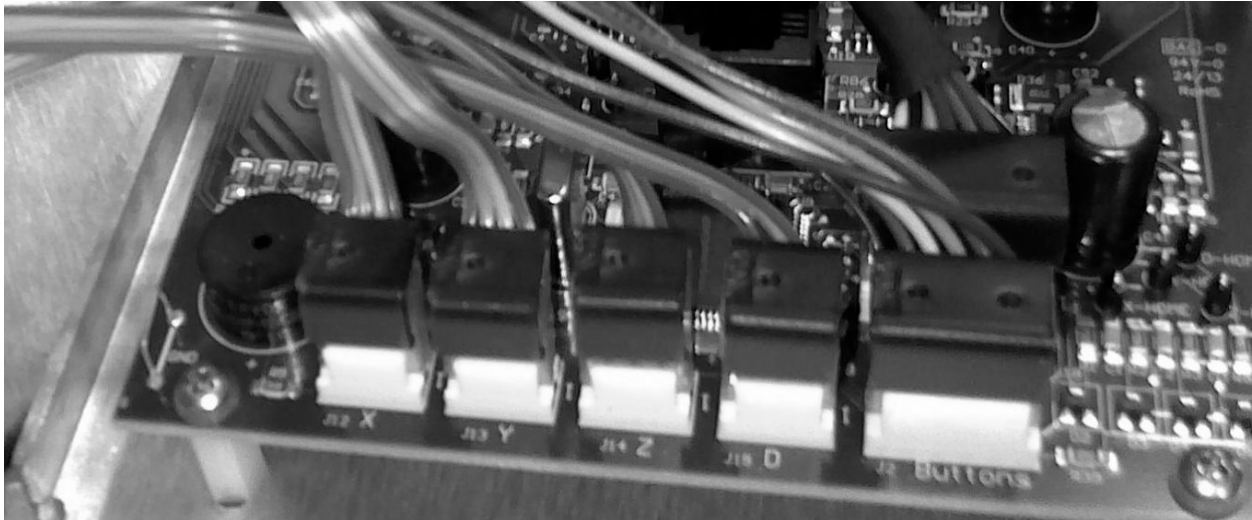


Figure 6-1. Locations of the axis connectors inside the ROE/Controller

# 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.

- 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.

# APPENDIX B. ACCESSORIES

| | |
|---|---|
| **W621 150** | Ground cable |
| **285204** | 4-inch dovetail extension |
| **285210** | Mounting adapter plate |
| **225RBI** | Rotating base |
| **221165** | Z-axis vertical extension |
| **BR-AW** | Rod holding clamp for XenoWorks® injectors (for rod OD 2 – 4 mm) |
| **MP-ROD** | Rod holder (for rod 6.25 mm or larger) |
| **MP-RISER-0.5**[1] | ½-inch riser |
| **MP-RISER-1.0** | 1-inch riser |
| **MT-78-FS** | Fixed platform stage with imperial/standard holes, chamber insert, and gantry supports |
| **MT-78-FS/M6** | Same as MT-78-FS with M6 tapped holes |
| **MD Series** | Micromanipulator platform |
| **MT-75** | Standard gantry-stand 8.7 to 13.4 in (22.1 to 33.9 cm) |
| **MT-75S** | Short gantry-stand 6.7 to 9.6 in (16.9 to 24.4 cm) |

---

[1] Risers may be combined to achieve desired height.

# APPENDIX C. TECHNICAL SPECIFICATIONS

| | |
|---|---|
| Travel | 25mm on X & Y; 50mm on D |
| Resolution | Minimal microstep size is 62.5 nanometers per microstep. Display has single micron resolution. |
| Speed | 3 mm/sec. maximum |
| Long Term Stability | < 1 micron/hour drive mechanism |
| Electrical: | |
|     Power Adapter: | Meanwell GS60A24-P1J |
|       Input (Mains) | 100 - 240 VAC, 50/60 Hz, 1.4A |
|       Output (to controller) | 24V DC, 2.5A, 60W Max. (see following table for cable specs) |
|     System Power consumption | 60-Watts maximum |
|     Mains fuses | None replaceable (power protection built into the Power Adapter) |
|     Cables | (Refer to the following tables for a description of all possible cables.) |

Table C-1. TRIO MP-235 cables and receptacles/connectors.

| Controller Rear Panel Port Connector/Receptacle | Cable Connector Types | Connects to … | Cable Type | Cable Max. Length |
|---|---|---|---|---|
| **Power Adapter 3-pin male connector** | ◄—3-pin power standard (female) \| 3-pin male—► (Geographical region dependent) | **Mains power source.** | **10A, 250V, with safety ground plug** | **3 meters (approx. 10 feet)** |
| **ROE/Controller Cabinet: MANIPULATOR (26-Pin HD DSUB female receptacle)** | ◄—HD DB-26 male \| HD DB-26 female—► (Straight-through) | **TRIO MP-235/M** | **Minimum of 26 awg stranded wire with 500 Volt.** | **3 meters (approx. 10 feet)** |
| **Power Adapter** | ◄—(fixed) \| ID 2.1 x OD 5.5 mm Barrel Plug (male)—► | **ROE/Controller Cabinet: POWER receptacle (center pin positive)** | **UL1185 18AWG** | **1.8 meters (approx. 6 feet)** |
| **ROE/Controller Cabinet: GROUND (1-pin Banana-style female receptacle)** | ◄—Banana male \| —►Alligator clip (hooded) | **a ground/earth source (user determined)** | | |
| **ROE/Controller Cabinet: USB** | ◄—A \| —►B | **Computer USB port** | | |

# APPENDIX D. QUICK REFERENCE

## D.1. Manual Operation
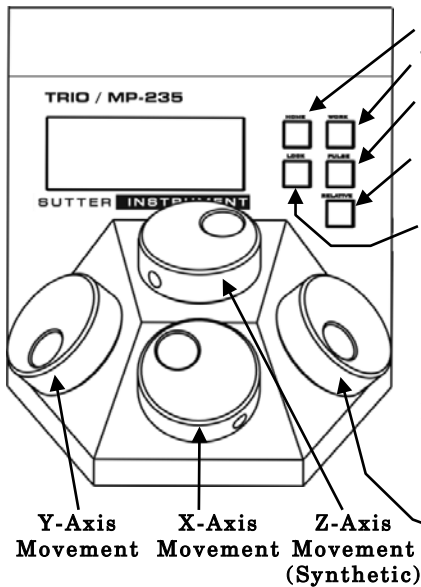


**HOME:** Move to defined home position. Press again to pause/resume.

**WORK:** Move to defined work position. Press again to pause/resume.

**PULSE:** Advances diagonal axis in 2.85 $\mu$m steps. Hold 3-sec. to enter **ANGLE SET** mode (active for 10 sec.) – use Knob D to change angle (0 – 90 degrees).

**RELATIVE:** Toggles between **Relative** and **Absolute** position moves. Hold 3-sec. to set the relative mode origin to the current absolute position.

**SPEED:** Cycles through Speed 0 (normal) through 3 (slowest). Hold 3 sec. to enter **LOCK** mode.

**Setting Home/Work Pos., Relative Mode Origin Pos., & Angle:** To set position, hold down HOME, WORK, & RELATIVE buttons for 3 seconds until beep sounds. Likewise, set angle with LOCK.

**Screen-color mode indications:** Green = Absolute position; Blue = Relative position; Red = Movement in progress or in quiet (**LOCK**) mode; knobs disabled.

**Movement Knobs Disabling and Quiet (LOCK) Mode:** Movement knobs are disabled during movement to Home, Work, external movement command, or while in quiet (LOCK) Mode.

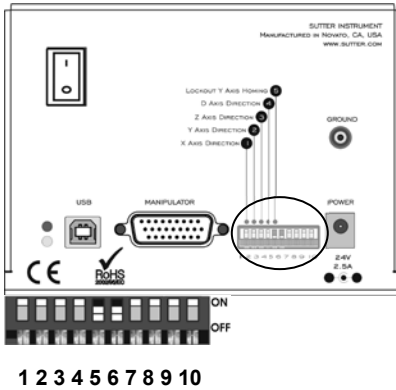**D-Axis Movement;** selects angle in degrees when in **ANGLE SET** mode.

**Y-Axis Movement**  **X-Axis Movement**  **Z-Axis Movement (Synthetic)**

## D.2. Configuration



**1 2 3 4 5 6 7 8 9 10**

Table D-1. Configuration Switches 1 – 5.

| Sw # | Definition | State | Setting | Position |
|---|---|---|---|---|
| 1 | X-Axis Knob Rotation for Forward (+) Movement | **Clockwise** | **Off\*** | **Up\*** |
| | | Counter | On | Down |
| 2 | Y-Axis Knob Rotation for Forward (+) Movement | **Clockwise** | **Off\*** | **Up\*** |
| | | Counter | On | Down |
| 3 | Z-Axis Knob Rotation for Forward (+) Movement | **Clockwise** | **Off\*** | **Up\*** |
| | | Counter | On | Down |
| 4 | D-Axis Knob Rotation for Forward (+) Movement | **Clockwise** | **Off\*** | **Up\*** |
| | | Counter | On | Down |
| 5 | Y Axis Lock Out for Homing | Enabled | Off | Up |
| | | **Disabled** | **On\*** | **Down\*** |

\* Normal operation (factory default).

Table D-2. Configuration Switches 6 – 10 (Ver. <2.2)

| Sw # | Definition | State | Setting | Position |
|---|---|---|---|---|
| 6 | Sensor Test\*\* (see Caution) | Enabled | Off | Up |
| | | **Disabled** | **On\*** | **Down\*** |
| 7 - 10 | Reserved | | **Off\*** | **Up\*** |

\*\***CAUTION: To avoid damage to the micromanipulator or stage, DIP Switch 6 (Sensor Test) must always be set to ON (down).**

Table D-3. Config. Switches 6 - 10 (Ver. 2.2 & 2.3+)

| Sw # | Definition | State | Setting | Position |
|---|---|---|---|---|
| 6 | Calibration Homing on Power On | None | Off | Up |
| | | Calibrate | On | Down\* |
| 7 | SPEED Select or PULSE button mode | Speed Select | Off\* | Up\* |
| | | Pulse | On | Down |
| 8 | Reserved (Ver =<2.2) | | Off\* | Up\* |
| | ROE Axis knobs active – all vs. D only (Ver =>2.3) | All Active | Off\* | Up\* |
| | | D only | On | Down |
| 9 \*\*\* | Y-axis travel length (25 or 12.5 mm) | 25mm | Off\* | Up\* |
| | | 12.5mm | On | Down |
| 10 \*\*\* | X-axis travel length (25 or 12.5 mm) | 25mm | Off\* | Up\* |
| | | 12.5mm | On | Down |

\*\*\***CAUTION: To avoid possible equipment damage, always be certain that Switch 9 and 10 settings correctly match the physical length of travel of the Y & X axes on the connected device.**

## D.3. External Control

Controlling the TRIO MP-235 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 TRIO MP-235 controller/ROE. The USB device driver for Windows is downloadable from Sutter Instrument's web site (www.sutter.com). The TRIO MP-235 requires Sutter Instrument's USB CDM (Combined Driver Model) Version 2.10.00 or higher. The CDM device driver 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 TRIO MP-235. 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 or enabling of the VCP device driver.

Table D-4. USB-VCP interface serial port settings.

| Property | Setting |
|---|---|
| Data ("Baud") Rate (bits per second (bps)) | 57600 |
| Data Bits | 8 |
| Stop Bits | 1 |
| Parity | None |
| Flow Control | None |

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:** Command sequences do not have terminators. 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 not used. 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 transmitted to the controller must contain an unsigned binary value. Attempting to code command sequences as "strings" is not advisable. Any command data returned by the controller should be initially treated as a sequence of unsigned byte values upon reception. 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 TRIO MP-235, all axis position values (number of microsteps) are stored as "unsigned long" 32-bit positive-only 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 32-bit values ("long" (or optionally, "signed long"), or "unsigned long" for C/C++; "I32" or "U32" for LabVIEW). "Unsigned" means the value is always positive; negative values are not allowed. The positive-only values can also be stored in signed type variables, in which case care must be taken to ensure that only positive values are exchanged with the controller.

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 D-5. Microns/microsteps conversion.

| System/Device | From/To Units | Conversion Factor (multiplier) |
|---|---|---|
| TRIO MP-235 with TRIO MP-235/M micromanipulator | $\mu$steps $\to$ $\mu$m | 0.09375 |
| | $\mu$m $\to$ $\mu$steps | 10.66666666667 |

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 "unsigned long" (C/C++), "uint32" (MATLAB), or "U32" (LabVIEW) integer (positive only) value. When converting to microns, type the result as a "double" (C/C++, MATLAB) or "DBL" (LabVIEW) 64-bit double-precision floating-point value.

Table D-6. Ranges and bounds.

| Device | Axis | Millimeters | Microns | Microsteps |
|---|---|---|---|---|
| TRIO MP-235/M | X & Y | 25 | 0 – 25,000 | 0 – 266,667 |
| | D | 50 | 0 – 50,000 | 0 – 533,334 |

**Command Reference:** The following table lists all the external-control commands for the TRIO MP-235.

Table D-7. TRIO MP-235 external control commands.

| Command | Tx/-Delay/-Rx | Ver. | Total Bytes | Byte Offset (Len.) | Value Dec. | Value Hex. | Value Binary | Alt-key-pad # | Ctrl-char | ASCII def./-char. | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Get Current Position and Angle ('c' or 'C') | Tx | All | 1 | 0 | 99 or 67 | 63 or 43 | 0110 0011 or 0100 0011 | 0099 or 0043 | | 'c' or 'C' | Returns the current positions ($\mu$steps) of X, Y, & D axes |
| | Rx. | All | 14 | | Three 4-byte (32-bit) values (current positions in $\mu$steps of X, Y, & D), + 1 byte for completion indicator. See *Ranges* table for minimum and maximum values. | | | | | | |
| | | | | 0 (4) | | | | | | | X pos. in $\mu$steps |
| | | | | 4 (4) | | | | | | | Y pos. in $\mu$steps |
| | | | | 8 (4) | | | | | | | D pos. in $\mu$steps |
| | | | | 12 | 13 | 0D | 0000 1101 | | ^M | <CR> | Completion indicator |
| Move to HOME Position ('h') | Tx | All | 1 | 0 | 104 | 68 | 0110 1000 | 0104 | | 'h' | Moves to the position saved for the controller's HOME button. |
| | Rx | All | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |
| Move to WORK Position ('w') | Tx | All | 1 | 0 | 119 | 77 | 0111 0111 | 0119 | | 'w' | Moves to the position saved for the controller's WORK button. |
| | Rx | All | 1 | 0 | 13 | 0D | 0000 1101 | | | <CR> | Completion indicator |
| Move to specified X axis Position ('x' or 'X') | Tx | All | 5 | 0 | 120 or 90 | 78 or 5A | 0111 1000 or 0101 1010 | 0120 or 0090 | | 'x' or 'X' | Move X axis to specified position (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | X $\mu$steps |

| Command | Tx/-Delay/-Rx | Ver. | Total Bytes | Byte Offset (Len.) | Dec. | Hex. | Binary | Alt-key-pad # | Ctrl-char | ASCII def./-char. | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Rx | All | 1 | 0 | 13 | 0D | 0000 1101 | | | \<CR\> | Completion indicator |
| Move to specified Y axis Position ('y' or 'Y') | Tx | All | 5 | 0 | 121 or 91 | 79 or 5B | 0111 1001 or 0101 1011 | 0121 or 0091 | | 'y' or 'Y' | Move Y axis to specified position (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | Y µsteps |
| | Rx | All | 1 | 0 | 13 | 0D | 0000 1101 | | | \<CR\> | Completion indicator |
| Move to specified D axis Position ('d' or 'D') | Tx | All | 5 | 0 | 100 or 68 | 64 or 44 | 0110 0100 or 0100 0100 | 0100 or 0068 | | 'd' or 'D' | Move D-axis to specified position (see *Ranges* table) |
| | | | | 1 (4) | | | | | | | D µsteps |
| | Rx | All | 1 | 0 | 13 | 0D | 0000 1101 | | | \<CR\> | Completion indicator |

## 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 Send/Receive Buffers:** Clearing (purging) the transmit and receive buffers of the I/O port immediately before sending any command is recommended.

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, & D */
   unsigned long  cp_x_us, cp_y_us, cp_d_us; /* microsteps */
   double         cp_x_um, cp_y_um, cp_d_um; /* microns */
   /* specified (move-to) position for X, Y, & D */
   unsigned long  sp_x_us, sp_y_us, sp_d_us; /* microsteps */
   double         sp_x_um, sp_y_um, sp_d_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 the TRIO MP-235/M based config. */
   double us2umCF = 0.09375;        /* microsteps to microns */
   double um2usCF = 10.66666666667; /* microns to microsteps */
   ```

   *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 & D, 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 must include positive values only for positions – negative positions must never be specified. All positions are absolute as measured from the physical beginning of travel of a device's axis. In application programming, it is important that positional values be checked (>= 0 and <= max.) to ensure that a negative absolute position is never sent to the controller and that end of travel is not exceeded. All computational relative positioning must always resolve to accurate absolute positions.

   *Declaring minimum and maximum absolute position variables in C/C++:*
   ```
   /* minimum and maximum positions for X, Y, & D */
   double min_x_um, min_y_um, min_d_um;  /* minimum microns */
   double max_x_um, max_y_um, max_d_um;  /* maximum microns */
   ```
   *Set minimum and maximum absolute positions for each axis – see Ranges & Bounds table.*
   ```
   /* initialize all minimum positions in microns*/
   min_x_um = 0;
   min_y_um = 0;
   min_d_um = 0;
   /* initialize all maximum positions in microns*/
   /* TRIO MP-235/M */
   max_x_um = 25000;
   max_y_um = 25000;
   max_d_um = 50000;
   ```

6. **Absolute Positioning System Origin:** The Origin is set to a physical position of travel to define absolute position 0. The physical Origin position is fixed at beginning of travel (BOT). This means that all higher positions (towards <u>end</u> of travel (EOT)) are positive values; there are no lower positions and therefore no negative values are allowed.

7. **Absolute vs. Relative Positioning:** Current position ('c') and move commands always use absolute positions. All positions can be considered "relative" to the Origin (Position 0), but all are in fact absolute positions. Any position that is considered to be "relative" to the current position, whatever that might be, can be handled synthetically by external programming. However, care should be taken to ensure that all relative position calculations always result in correct positive absolute positions before initiating a move command.

*Declaring relative position variables in C/C++:*
```
/* relative positions for X, Y, & D */
double rp_x_um, rp_y_um, rp_d_um;  /* microns */
/* initialize all relative positions to 0 after
declaring them */
rp_x_um = rp_y_um = rp_d_um = 0;
```

*Enter any positive or negative value for each relative position (e.g., rp_x_um = 1000; rp_y_um = 500; rp_d_um = -500 … etc.*

*For each axis, check to make sure that the new resultant absolute position (to which to move) is within bounds. Reset the relative position to 0 if not. If relative value is negative, its positivized value must not be greater than the current position. Otherwise, if positive, adding current position with relative position must not exceed the maximum position allowed. If out of bounds, resetting relative position to 0 allow the remaining conversions and movement to resolve without error.*
```
/* check to make sure that relative X is within
bounds */
if ( ( rp_x_um < 0 && abs(rp_x_um) > cp_x_um ) ||
     (cp_x_um + rp_x_um > max_x_um) )   /* out of
bounds? */
    rp_x_um = 0; /* yes, so reset relative pos.
to 0 */
```
*Repeat the above bounds check for each of the remaining axes.*

*For each axis, calculate new absolute position in microns and then convert to microsteps before issuing a move command.*
```
/* convert X relative position to absolute
position */
sp_x_um = cp_x_um + rp_x_um; /* add relative pos.
to current pos. */
/* convert new absolute X position in microns to
microsteps */
sp_x_us = sp_x_um * um2usCF;
```
*Repeat for each of the remaining axes as required before issuing 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 in microsteps are always positive, so data type must be an "unsigned" integer that can hold 32 bits of data. 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 an unsigned 32-bit integer ("unsigned long" in C/C++, "uint32" in MATLAB, "U32" in LabVIEW, etc.).

Position values in microns (micrometers or $\mu$m) should be data typed as double-precision floating point variables ("double" in C/C++ and MATLAB, "DBL" 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. **Angle Setting & Movement:** Although the set angle command allows for a range of 0° to 90°, the effective range that allows full movement is 1° to 89° (>0° and <90°). If 0° or 90°, Z or X axis fails to move, causing single- and multi-axis movement commands to fail. The ideal range for smooth movement is 10° to 80°. Factory default is 30°.

# NOTES:

# NOTES:

# INDEX

# <u>NOTES</u>