This page is intentionally left blank.
Contents

About This Guide .......................................................................................................................... 9
Introduction .................................................................................................................................. 9
Chapters and Appendixes in This Guide .................................................................................... 9
Information about the DMM-0200 .............................................................................................. 10
About This Release .................................................................................................................... 10
Viewing the PDF Version ............................................................................................................ 10
Symbols ...................................................................................................................................... 10
Regulatory Compliance Certification ............................................................................................ 11
Contact Information .................................................................................................................. 12

Chapter 1 – Site Planning ............................................................................................................ 13
About This Chapter .................................................................................................................... 13
Introduction .................................................................................................................................. 13
Topics .......................................................................................................................................... 13
Customer Responsibilities ......................................................................................................... 14
Requirements for the Installation Site .......................................................................................... 14
Preliminary Steps ....................................................................................................................... 14
Site Requirements ...................................................................................................................... 15
Requirements for Receiving and Setup ....................................................................................... 15
Receiving ..................................................................................................................................... 15
Setup .......................................................................................................................................... 15
Specifications for Power, Communications, and Environment .................................................... 16
Electrical Power ........................................................................................................................... 16
Electrical receptacle .................................................................................................................... 16
Grounding ..................................................................................................................................... 16
Power Cord and Communications Cables ....................................................................................... 17
Operating Environment ............................................................................................................... 17
Where to Go Next ....................................................................................................................... 18

Chapter 2 – Technical Overview ................................................................................................ 19
About This Chapter .................................................................................................................... 19
Introduction .................................................................................................................................. 19
Topics .......................................................................................................................................... 19
Technical Description ................................................................................................................ 20
Chapter 5 – Running the DMM-0200 via PMX-2EX-SA Software

Introduction

Topics

Installing PMX Drivers

Where to Go Next

---

Chapter 3 – Installing the DMM-0200

Introduction

Topics

Installing the System

Modes of Operation

Connecting to the DMM-0200

Installing the Motor and Encoder Cables

Installing the I/O Cables

Installing the Power Cord and Communications Cables

Where to Go Next

---

Chapter 4 – Installing Application Software for the DMM-0200

Introduction

Topics

Software Description

Installing PMX-2EX-SA Software

Installing PMX Drivers

Where to Go Next

---

Chapter 2 – Motion Control Capabilities

Introduction

Topics

Encoder Input Connection

Digital Outputs

Limit, Home, and Digital Input

Interface Circuitry

Rear Panel LEDs

Rear Panel Connectors & Switches

Rear Panel Connector Pinouts

Rear Panel Jumper and Switch Selection

Where to Go Next

---

Chapter 1 – About This Chapter

Introduction

Topics

Description

Front Panel

Rear Panel

Where to Go Next

---

DMM-0200 Product User Guide

159 Swanson Road • Boxborough, MA 01719 • Tel: 508-475-3400 • Email: sales@dovermotion.com
Chapter 8 – Interactive Commands & USB Communications

Appendix
List of Tables

Table 1  Weights and Measurements .......................................................... 15
Table 2  Specifications for Electrical power .................................................. 16
Table 3  Specifications for Power and Communications Cables ........................ 17
Table 4  Specifications for the Operating Environment .................................... 17
Table 5  Rear Panel Connectors ..................................................................... 24
Table 6  Rear Panel Switches & Jumpers ....................................................... 24
Table 7  Pinouts – Serial Communications ................................................... 25
Table 8  Pinouts – USB Communication ....................................................... 25
Table 9  Pinouts – Encoder/Limits .................................................................. 26
Table 10 Pinouts – Motors ............................................................................ 26
Table 11 Pinouts – Encoder Out X & Y ............................................................ 27
Table 12 Pinouts – Digital I/O ......................................................................... 27
Table 13 Pinouts – Analog I/O & Joystick ...................................................... 28
Table 14 Current Switch Settings .................................................................. 29
Table 15 Jumper Selection – JP1 – JP8 ............................................................ 30
Table 16 Fuse LEDs ....................................................................................... 31
Table 17 Enable LEDs .................................................................................. 31
Table 18 Standard Programs ......................................................................... 57
Table 19 Standalone Run on Boot-Up ............................................................. 63
Table 20 Storing to Flash .............................................................................. 63
Table 21 Pulse Speed ..................................................................................... 92
Table 22 Motor Status ................................................................................... 93
Table 23 Polarity ............................................................................................ 97
Table 24 Digital Inputs .................................................................................. 98
Table 25 Digital Outputs .............................................................................. 98
Table 26 Enable Outputs .............................................................................. 99
Table 27 Joystick Control ............................................................................ 99
Table 28 Summary of Joystick Control Parameters ...................................... 100
Table 29 StepNLoop Closed Loop Control ................................................... 101
Table 30 StepNLoop Return Values .............................................................. 101
Table 31 StepNLoop Conditions ................................................................... 102
Table 32 ASCII Commands .......................................................................... 103
Table 33 Error Codes ................................................................................... 108
Table 34 Speed Settings .............................................................................. 111
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DMM-0200 Dimensions</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>DMM-0200 ISO View</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>DMM-0200 Front View</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>DMM-0200 Rear View</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td>Digital Inputs</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>Digital Outputs</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>Encoder Inputs</td>
<td>33</td>
</tr>
<tr>
<td>8</td>
<td>Analog Inputs</td>
<td>33</td>
</tr>
<tr>
<td>9</td>
<td>System Installation</td>
<td>36</td>
</tr>
<tr>
<td>10</td>
<td>Performax Installation</td>
<td>40</td>
</tr>
<tr>
<td>11</td>
<td>Driver Installation</td>
<td>41</td>
</tr>
<tr>
<td>12</td>
<td>DMM-0200 USB Port</td>
<td>44</td>
</tr>
<tr>
<td>13</td>
<td>PMX-2EX-SA Application</td>
<td>45</td>
</tr>
<tr>
<td>14</td>
<td>Main Control Screen</td>
<td>46</td>
</tr>
<tr>
<td>15</td>
<td>Status Screen (A)</td>
<td>47</td>
</tr>
<tr>
<td>16</td>
<td>Status Screen (B)</td>
<td>48</td>
</tr>
<tr>
<td>17</td>
<td>On-The-Fly-Speed Control (C)</td>
<td>48</td>
</tr>
<tr>
<td>18</td>
<td>Product Information (D)</td>
<td>49</td>
</tr>
<tr>
<td>19</td>
<td>Terminal (E)</td>
<td>49</td>
</tr>
<tr>
<td>20</td>
<td>On-The-Fly-Position Control (F)</td>
<td>49</td>
</tr>
<tr>
<td>21</td>
<td>Digital I/O Status (G)</td>
<td>49</td>
</tr>
<tr>
<td>22</td>
<td>About (H)</td>
<td>50</td>
</tr>
<tr>
<td>23</td>
<td>Setup (I)</td>
<td>50</td>
</tr>
<tr>
<td>24</td>
<td>Variables (J)</td>
<td>51</td>
</tr>
<tr>
<td>25</td>
<td>Program File Control (K)</td>
<td>52</td>
</tr>
<tr>
<td>26</td>
<td>Text Programming Box (L)</td>
<td>52</td>
</tr>
<tr>
<td>27</td>
<td>Compiler (M)</td>
<td>53</td>
</tr>
<tr>
<td>28</td>
<td>Terminal (N)</td>
<td>53</td>
</tr>
<tr>
<td>29</td>
<td>DMM-0200 Front Panel</td>
<td>56</td>
</tr>
<tr>
<td>30</td>
<td>Retrieving a Program</td>
<td>60</td>
</tr>
<tr>
<td>31</td>
<td>Text Window</td>
<td>61</td>
</tr>
<tr>
<td>32</td>
<td>USB Cable</td>
<td>90</td>
</tr>
<tr>
<td>33</td>
<td>Trapezoidal Velocity Profile</td>
<td>91</td>
</tr>
<tr>
<td>34</td>
<td>S-Curve Velocity Profile</td>
<td>91</td>
</tr>
<tr>
<td>35</td>
<td>Homing - Home Input Only (High Speed)</td>
<td>94</td>
</tr>
<tr>
<td>36</td>
<td>Homing - Home Input Only (High Speed and Low Speed)</td>
<td>94</td>
</tr>
<tr>
<td>37</td>
<td>Homing - Limit Only</td>
<td>95</td>
</tr>
<tr>
<td>38</td>
<td>Homing - Home and Z-Index</td>
<td>95</td>
</tr>
<tr>
<td>39</td>
<td>Homing - Z-Index Only</td>
<td>96</td>
</tr>
<tr>
<td>40</td>
<td>Joystick Control</td>
<td>99</td>
</tr>
<tr>
<td>41</td>
<td>Acceleration/Deceleration Profile</td>
<td>112</td>
</tr>
</tbody>
</table>
About This Guide

Introduction

This guide describes the installation of the DMM-0200 and the operation of it. The intended audiences are the customer, technician responsible for performing the installation and preparing the site beforehand, and the operator running the system.

This point forward, “About This Guide,” covers the following topics.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information about the DMM-0200</td>
<td>10</td>
</tr>
<tr>
<td>Symbols</td>
<td>10</td>
</tr>
<tr>
<td>Regulatory Compliance Certification</td>
<td>11</td>
</tr>
<tr>
<td>Contact Information</td>
<td>12</td>
</tr>
</tbody>
</table>

Chapters and Appendixes in This Guide

This guide contains the following chapters. It does not cover advanced topics such as debugging the system or editing configuration files.

<table>
<thead>
<tr>
<th>For</th>
<th>Go to page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1 Site Planning</td>
<td>13</td>
</tr>
<tr>
<td>Describes the responsibilities of the customer and the requirements for the installation site, and includes an installation process flow and installation procedure checklist.</td>
<td></td>
</tr>
<tr>
<td>Chapter 2 Technical Overview</td>
<td>19</td>
</tr>
<tr>
<td>Describes the functionality of the DMM-0200 and its main components.</td>
<td></td>
</tr>
<tr>
<td>Chapter 3 Installing the DMM-0200</td>
<td>35</td>
</tr>
<tr>
<td>Describes how to install the DMM-0200 into your system.</td>
<td></td>
</tr>
<tr>
<td>Chapter 4 Installing Application Software for the DMM-0200</td>
<td>39</td>
</tr>
<tr>
<td>Explains how to install application software on your PC.</td>
<td></td>
</tr>
<tr>
<td>Chapter 5 Running the DMM-0200 via PMX-2EX-SA Software</td>
<td>43</td>
</tr>
<tr>
<td>Explains how to run the DMM-0200 using only the GUI on your PC.</td>
<td></td>
</tr>
<tr>
<td>Chapter 6 Running the DMM-0200 via Front Panel Buttons</td>
<td>55</td>
</tr>
<tr>
<td>Explains how to run the DMM-0200 using only the front panel buttons.</td>
<td></td>
</tr>
<tr>
<td>Chapter 7 Editing Programs</td>
<td>59</td>
</tr>
<tr>
<td>Explains how to edit stored programs.</td>
<td></td>
</tr>
<tr>
<td>Chapter 8 Interactive Commands &amp; USB Communications</td>
<td>88</td>
</tr>
<tr>
<td>Describes the USB communications and ASCII commands to interactively communicate to the DMM-0200.</td>
<td></td>
</tr>
<tr>
<td>Appendix</td>
<td>110</td>
</tr>
<tr>
<td>Supporting documentation for the DMM-0200 manual</td>
<td></td>
</tr>
</tbody>
</table>
Information about the DMM-0200

About This Release

This version of the operations manual supports the DMM-0200 GEN1 motion controller.

Viewing the PDF Version

The PDF version of this guide provides a number of ways to navigate through the content. Blue text indicates links to related topics. You can also do the following:

- Click the Bookmarks tab in the left pane to view the list of bookmarks and click any bookmark to navigate to that topic. If the Bookmarks tab is not visible, choose View> Navigation> Bookmarks from the menu bar.
- Click on the Contents bookmark. Pick an entry in the Contents and click either the entry text or its page number to go to that topic.
- Scroll to the end of the Contents to the List of Figures. Pick any entry in the list and click either the figure number and title or its page number to go to that topic.
- Scroll to the end of the List of Figures to the List of Tables. Pick any entry in the list and click either the table number and title or its page number to go to that topic.
- Click on the Index bookmark. Pick any index entry and click its page number to go to that topic.

Symbols

This guide uses the following symbols.

- **Note:** Indicates neutral or positive information that emphasizes or supplements important points of the main text.

- **Caution:** Advises users that failure to take or avoid a specified action could result in loss of data.

- **Warning:** Advises users that failure to take or avoid a specified action could result in physical harm to the user or hardware.
Regulatory Compliance Certification

[TESTING FOR COMPLIANCE IS IN PROCESS.]

The DMM-0200 hardware and its components meet or exceed the requirements of the following regulatory agencies. Applicable labels indicating compliance with these requirements appear on the hardware.
Contact Information

DMM-0200 Contact Information

The Business Unit of Dover Motion in Boxborough, MA is the manufacturer of the DMM-0200. The location of the unit is:

Dover Motion
159 Swanson Rd.
Boxborough, MA 01719

The Dover Motion Website is located at:

http://www.Dovermotion.com

If you have questions or comments, contact:
Phone: 508-475-3400
Email: sales@dovermotion.com

Or for Sales, contact:

Eastern U.S. & Canada
Phone: 508-475-3400
Email: eastsales@dovermotion.com

Western U.S. & Canada
Phone: 508-475-3400
Email: westsales@dovermotion.com
Chapter 1
Site Planning

About This Chapter

Introduction

This chapter describes the responsibilities of the customer and the requirements for the installation site.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Responsibilities</td>
<td>14</td>
</tr>
<tr>
<td>Site Requirements</td>
<td>15</td>
</tr>
<tr>
<td>Specifications for Power, Communications, and Environment</td>
<td>15</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>18</td>
</tr>
</tbody>
</table>
Customer Responsibilities

Requirements for the Installation Site

For installation of the DMM-0200, you are responsible for supplying the following:

- An electrical receptacle available for power to the system. See “Electrical Receptacle” on page 16 for specifications.

**Warning:** If you use a different type of power cord, plug, or receptacle, other than those specified in the guide, you are responsible for having a qualified electrician install them.

**Warning:** If the system voltage is different than what is specified in the guide, you are responsible for having a qualified electrician rewire the system power supplies PWS1-4 and the main transformer T1 for that voltage.

- A suitable bench area for the DMM-0200. See bench specifications in the Table 1.
- One 2.0 USB cables (optional).

**Note:** The USB cable is not required for the initial installation since the system software is already installed on the DMM-0200 when shipped. This cable is for customer use.

Preliminary Steps

Perform the following preliminary steps to prepare for the installation:

1. Complete the Site Requirements Checklist. See Site Requirements section.

**Note:** Do not attempt to perform an installation unless the requirements specified in this guide have been carried out.
Site Requirements
Requirements for Receiving and Setup

Receiving

When you receive your DMM-0200, do the following:

1. Inspect the package for damage.

Caution: If you suspect damage, take pictures of the location where you suspect there is damage. A representative of the shipping carrier may need to be present during the unpacking process for insurance purposes.

Setup

Select a location where you want the DMM-0200 to be unpacked and installed. Please ensure the following parameters are met: Table 1 Weights and Measurements

<table>
<thead>
<tr>
<th>Specification</th>
<th>DMM-0200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>2.24 Kg (approx. 4.95 lbs)</td>
</tr>
<tr>
<td>Dimensions (maximum):</td>
<td></td>
</tr>
<tr>
<td>Width</td>
<td>24.06 cm (approx. 9.47 inches)</td>
</tr>
<tr>
<td>Depth</td>
<td>20.05 cm (approx. 8.08 inches)</td>
</tr>
<tr>
<td>Height</td>
<td>12.86 cm (approx. 5.06 inches)</td>
</tr>
</tbody>
</table>

Figure 1 DMM-0200 Dimensions

24.06 cm
20.05 cm
12.86 cm
Specifications for Power, Communications, and Environment

Electrical Power

The DMM-0200 requires a main power source, as specified in Table 2. Ensure that the AC power source is easily accessible.

Table 2 Specifications for Electrical power

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>90-264VAC ± 10%</td>
</tr>
<tr>
<td>Frequency</td>
<td>47-63 Hz</td>
</tr>
<tr>
<td>Current (at nominal voltage, 110VAC)</td>
<td>Typical, maximum, inrush surge: No more than 11 amps.</td>
</tr>
</tbody>
</table>

Electrical receptacle

The DMM-0200 requires a NEMA 5-15 receptacle in North America. The electrical receptacle must be located within 2.5 meters (approx. 8 feet) of the power cord input on the system.

Note: The electrical receptacle must have a dedicated 1.2kVA power line and ground.

Grounding

Certain types of electrical noise are greatly exaggerated by poor or improper electrical ground connections. To prevent these problems, it is desirable to have a dedicated circuit and ground for use by the DMM-0400.

Note: For grounding wiring, the DMM-0200 uses internal AC ground connections with green/yellow 16AWG wiring.

Warning: Do not connect the DMM-0200 to the same dedicated line and ground that is used to power a device with a high current.

Warning: Use qualified personnel for installation of all electrical fixtures, and ensure that all installations follow local bylaws.
Power Cord and Communications Cables

Table 3 lists the suggested lengths for power cord and communications cables for the DMM-0200. The DMM-0200 shipment includes only the power cord.

Table 3 Specifications for Power and Communications Cables

<table>
<thead>
<tr>
<th>Cable Type</th>
<th>Length¹</th>
<th>Maximum Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMM-0200 power cord (supplied):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEMA5-15 (North America-specific)</td>
<td>2.4 meters (8 feet) 15 Amp</td>
<td>3.6 meters (12 feet)</td>
</tr>
<tr>
<td>Communications cables (not supplied):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USB 2.0 cable A-B</td>
<td>3 meters (10 feet)</td>
<td>30 meters (100 feet)</td>
</tr>
</tbody>
</table>

Warning: If you use a different type of power cord, plug, or receptacle, other than those specified in this guide, you are responsible for having a qualified electrician install them.

Operating Environment

Table 4 lists the specifications for the environment in which the DMM-0200 will be fully operational and compliant with its performance specifications.

Table 4 Specifications for the Operating Environment

<table>
<thead>
<tr>
<th>Specification</th>
<th>Acceptable Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>15-30˚ C  59-86˚ F</td>
</tr>
<tr>
<td>Relative humidity</td>
<td>20-80% non-condensing</td>
</tr>
<tr>
<td>Altitude</td>
<td>0 to 2,000 meters above sea level 0 to 6,600 feet above sea level</td>
</tr>
</tbody>
</table>

¹ The cord and cable lengths specified in this table are suggested lengths. You must supply the cord or cable if you need to use anything longer than these (see the column, Maximum length).
Where to Go Next

Continue with the next chapter, *Chapter 2 - Technical Overview*. The next section will allow you to become familiar with the functionalities, capabilities, and main components of the DMM-0200.
Chapter 2
Technical Overview

About This Chapter

Introduction

This chapter describes the functionalities, capabilities, and main components of the DMM-0200 from a technical perspective. This chapter should provide the user with more detail and familiarity regarding the operation DMM-0200.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Description</td>
<td>20</td>
</tr>
<tr>
<td>Front Panel</td>
<td>22</td>
</tr>
<tr>
<td>Rear Panel</td>
<td>23</td>
</tr>
<tr>
<td>Rear Panel Connector Pinouts</td>
<td>25</td>
</tr>
<tr>
<td>Rear Panel Jumper and Switch Selection</td>
<td>29</td>
</tr>
<tr>
<td>Rear Panel LEDs</td>
<td>31</td>
</tr>
<tr>
<td>Interface Circuitry</td>
<td>32</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>34</td>
</tr>
</tbody>
</table>
Technical Description

Motion Control Capabilities

The DMM-0200 controller (along with the DMM-0400) is a family of programmable, motion controllers. It is highly customizable and can control up to four axis of motion.

A wide range of applications using stepper motors can be controlled using the DMM-0200. The versatile yet low cost and highly efficient design of the DMM-0200 allows the user to satisfy a wide range of motion control requirements.

See below for part number information:

Ordering Information:

DMM - 02#0

Number of Step Axes (1 - 2)

Example:

DMM - 0220

2 step axes

The DMM-0200 can be ordered with 1 or 2 step axes per controller. Please contact Dover Motion Sales group for further information.

Additional axis of motion (up to 4 axes) is available with the DMM-0400 controller.
Description

The DMM-0200 can control stepper motors. The figures in this section will describe the axis configurations for the DMM-0200 as well as interfacing with the controller.

Caution: To avoid overheating, always operate the DMM-0200 with the enclosure cover on to allow for proper air flow through it.

Note: Removing the enclosure cover is only necessary for troubleshooting.

Figure 2 DMM-0200 ISO View
Front Panel

Figure 3 DMM-0200 Front View

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>POWER</td>
<td>On/Off power switch</td>
</tr>
<tr>
<td>2</td>
<td>START</td>
<td>Start motion program</td>
</tr>
<tr>
<td>3</td>
<td>STOP</td>
<td>Stop motion program</td>
</tr>
<tr>
<td>4</td>
<td>PROGRAM SELECT</td>
<td>Selects the motion program number</td>
</tr>
</tbody>
</table>
Rear Panel

There are 2 axis of motion – X & Y
The connectors for axis X are shown in detail here.

The connectors and LEDs for Y is identical to the X axis.

Figure 4 DMM-0200 Rear View
Rear Panel Connectors & Switches

The following table lists the connector positions on the rear panel and the type of connector for each.

Table 5 Rear Panel Connectors

<table>
<thead>
<tr>
<th>Connector Position</th>
<th>Description</th>
<th>Connector Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>J4-serial x</td>
<td>Serial data communications, Axis 1, X-axis</td>
<td>DE-9-M</td>
</tr>
<tr>
<td>J21-serial y</td>
<td>Serial data communications, Axis 2, Y-axis</td>
<td>DE-9-M</td>
</tr>
<tr>
<td>J10-ENC/LIM X</td>
<td>Encoder/Limits, Axis 1 (positions and limits for X-axis)</td>
<td>HD-15-M</td>
</tr>
<tr>
<td>J24-ENC/LIM Y</td>
<td>Encoder/Limits, Axis 2 (positions and limits for Y-axis)</td>
<td>HD-15-F</td>
</tr>
<tr>
<td>J11-MOTORX</td>
<td>Motor, Axis 1 (motor for X-axis)</td>
<td>DE-9-F</td>
</tr>
<tr>
<td>J26-MOTOR Y</td>
<td>Motor, Axis 2 (motor for Y-axis)</td>
<td>DE-9-F</td>
</tr>
<tr>
<td>J25-ENC OUT X/Y</td>
<td>Encoder out, Axis 1 &amp; 2, X/Y axis</td>
<td>DA-15-M</td>
</tr>
<tr>
<td>J18-DIGITAL I/O</td>
<td>Opto-isolated inputs and outputs</td>
<td>DB-25-F</td>
</tr>
<tr>
<td>J19-ANALOG I/O JOYSTICK</td>
<td>Joystick connection</td>
<td>DA-15-F</td>
</tr>
<tr>
<td>USB</td>
<td>USB communications, all axis</td>
<td>USB</td>
</tr>
<tr>
<td>AC IN</td>
<td>AC Power</td>
<td>NEMA-5-15P</td>
</tr>
</tbody>
</table>

The following table lists the switches and jumpers on the rear panel.

Table 6 Rear Panel Switches & Jumpers

<table>
<thead>
<tr>
<th>Switch/Jumper Position</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1 CUR X</td>
<td>Current limit DIP switch for X axis stepper</td>
</tr>
<tr>
<td>SW2 CUR Y</td>
<td>Current limit DIP switch for Y axis stepper</td>
</tr>
<tr>
<td>JP1</td>
<td>Rear panel SPDT relay enable/disable</td>
</tr>
<tr>
<td>JP2</td>
<td>N/A</td>
</tr>
<tr>
<td>JP3</td>
<td>Front panel PROGRAM SELECT enable/disable</td>
</tr>
<tr>
<td>JP4</td>
<td>Front panel PROGRAM SELECT enable/disable</td>
</tr>
<tr>
<td>JP5</td>
<td>Front panel PROGRAM SELECT enable/disable</td>
</tr>
<tr>
<td>JP6</td>
<td>Front panel PROGRAM SELECT enable/disable</td>
</tr>
<tr>
<td>JP7</td>
<td>Front panel START button enable/disable</td>
</tr>
<tr>
<td>JP8</td>
<td>Front panel STOP button enable/disable</td>
</tr>
</tbody>
</table>
Rear Panel Connector Pinouts

Pinouts – Serial Communications (J4-SERIAL X, J21-SERIAL Y)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>DE-9-M Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>RX</td>
<td>Receive data</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>TX</td>
<td>Transmit data</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Ground</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>N/C</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>N/C</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7 Pinouts – Serial Communications

**Note**: The two serial connectors on the rear panel are used for setting up the individual drivers for each axis of motion. **This is already done at Dover Motion.**

Under normal circumstances, the user should not need to communicate to the individual drivers and therefore should not need to connect to these four serial ports.

Should you need to communicate to the individual drivers, please contact the Dover Motion Service Department for assistance.

Pinouts – USB Communication

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>USB-B Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VCC</td>
<td>+5VDC</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>D-</td>
<td>Data-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>D+</td>
<td>Data+</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Ground</td>
<td></td>
</tr>
</tbody>
</table>

Table 8 Pinouts – USB Communication

**Note**: The user should only need to communicate to the DMM-0200 using the USB connection. The USB connection provides full control of the DMM-0200.
### Pinouts – Encoder/Limits (J10-ENC/LIM X, J24-ENC/LIM Y)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>HD-15-F Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+VLIM</td>
<td>Limit power (+5VDC)</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>+LIM</td>
<td>Positive limit</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>-LIM</td>
<td>Negative limit</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CHASSIS</td>
<td>Earth ground</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Logic ground</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>+5VENC</td>
<td>Encoder power (+5VDC)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ENC A</td>
<td>Channel A</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ENC /A</td>
<td>Complement of channel A</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ENC B</td>
<td>Channel B</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ENC /B</td>
<td>Complement of channel B</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ENC Z</td>
<td>Channel Z</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ENC /Z</td>
<td>Complement of channel Z</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>HOME</td>
<td>Home switch</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>GND</td>
<td>Logic ground</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>CHASSIS</td>
<td>Earth ground</td>
<td></td>
</tr>
</tbody>
</table>

Table 9 Pinouts – Encoder/Limits

### Pinouts – Motors (J11-MOTOR X, J26-MOTOR Y)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>DE-9-F Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>STP A</td>
<td>Stepper motor coil A</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>N/C</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>Ground</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>STP B</td>
<td>Stepper motor coil B</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>STP /A</td>
<td>Stepper motor coil /A</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>FAULT</td>
<td>Motor fault</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>STP B/</td>
<td>Stepper motor coil /B</td>
<td></td>
</tr>
</tbody>
</table>

Table 10 Pinouts – Motors
### Pinouts – Encoder Out X & Y (J25-ENC OUT X/Y)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>DA-15-M Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ENC AY</td>
<td>Y axis – Encoder A channel</td>
<td>7 15</td>
</tr>
<tr>
<td>2</td>
<td>ENC /AY</td>
<td>Y axis – Encoder /A channel</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ENC BY</td>
<td>Y axis – Encoder B channel</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ENC /BY</td>
<td>Y axis – Encoder /B channel</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ENC ZY</td>
<td>Y axis – Encoder Z channel</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ENC /ZY</td>
<td>Y axis – Encoder /Z channel</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ENC /ZX</td>
<td>X axis – Encoder A channel</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ENC ZX</td>
<td>X axis – Encoder /A channel</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ENC BX</td>
<td>X axis – Encoder B channel</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ENC /BX</td>
<td>X axis – Encoder /B channel</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ENC /AX</td>
<td>X axis – Encoder Z channel</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>ENC AX</td>
<td>X axis – Encoder /Z channel</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>GND</td>
<td>Ground</td>
<td>1 8</td>
</tr>
<tr>
<td>15</td>
<td>GND</td>
<td>Ground</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pinouts – Digital I/O (J18-DIGITAL I/O)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin Number</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>21</td>
</tr>
<tr>
<td>22</td>
</tr>
<tr>
<td>23</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>25</td>
</tr>
</tbody>
</table>

Table 11 Pinouts – Encoder Out X & Y

Table 12 Pinouts – Digital I/O

\^ Jumper JP1 enables and disables the relay function through digital output 8. Please see Jumper section for detailed description of the functionality.
### Pinouts – Analog I/O & Joystick (J19-ANALOG I/O JOYSTICK)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Function</th>
<th>Description</th>
<th>DA-15-F Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+VANALOG</td>
<td>Analog power (+5VDC)</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>STICK P/B</td>
<td>Joystick - Stick pushbutton, Digital input 1</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>ANALOG X</td>
<td>Joystick X analog input, Analog input 1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ANALOG Y</td>
<td>Joystick Y analog input, Analog input 2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>LEFT P/B</td>
<td>Joystick - Left pushbutton⁵, Digital input 2</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ANALOG Z</td>
<td>Joystick Z analog input, Analog input 3</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>RIGHT P/B</td>
<td>Joystick - Right pushbutton⁶, Digital input 3</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Table 13 Pinouts – Analog I/O & Joystick

¹ Jumper JP8 enables and disables the left pushbutton control from the joystick. Please see Jumper section for detailed description of the functionality.

² Jumper JP7 enables and disables the right pushbutton control from the joystick. Please see Jumper section for detailed description of the functionality.
Rear Panel Jumper and Switch Selection

Current Switch Settings – SW1 CUR X, SW2 CUR Y

<table>
<thead>
<tr>
<th>DS4</th>
<th>DS3</th>
<th>DS2</th>
<th>DS1</th>
<th>Current (A)</th>
<th>DIP Switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2.8</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2.6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2.4</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2.2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1.8</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

Table 14 Current Switch Settings

These DIP switches set the maximum current rating for the stepper motor. Please refer to your motor data sheet to set these DIP switches.
### Jumper Selection – JP1 – JP8

<table>
<thead>
<tr>
<th>Jumper</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JP1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Output 8 to rear panel</td>
<td>Installing a jumper on JP1 1-2 will disable the SPDT Relay and Output 8 will be routed to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for SPDT Relay</td>
<td>Installing a jumper on JP1 2-3 will enable the SPDT Relay and Output 8 will be used to switch the relay</td>
</tr>
<tr>
<td>JP2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 8 to rear panel</td>
<td>Installing a jumper on JP2 1-2 will route Input 8 to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 to disable Input 8</td>
<td>Installing a jumper on JP2 2-3 will disable Input 8</td>
</tr>
<tr>
<td>JP3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 7 to rear panel</td>
<td>Installing a jumper on JP3 1-2 will disable the front panel THUMBWHEEL button and Input 7 will be routed to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel thumbwheel</td>
<td>Installing a jumper on JP3 2-3 will enable the front panel THUMBWHEEL button and Input 7 will not be available on J18</td>
</tr>
<tr>
<td>JP4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 6 to rear panel</td>
<td>Installing a jumper on JP4 1-2 will disable the front panel THUMBWHEEL button and Input 6 will be routed to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel thumbwheel</td>
<td>Installing a jumper on JP4 2-3 will enable the front panel THUMBWHEEL button and Input 6 will not be available on J18</td>
</tr>
<tr>
<td>JP5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 5 to rear panel</td>
<td>Installing a jumper on JP5 1-2 will disable the front panel THUMBWHEEL button and Input 5 will be routed to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel thumbwheel</td>
<td>Installing a jumper on JP5 2-3 will enable the front panel THUMBWHEEL button and Input 5 will not be available on J18</td>
</tr>
<tr>
<td>JP6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 4 to rear panel</td>
<td>Installing a jumper on JP6 1-2 will disable the front panel THUMBWHEEL button and Input 4 will be routed to J18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel thumbwheel</td>
<td>Installing a jumper on JP6 2-3 will enable the front panel THUMBWHEEL button and Input 4 will not be available on J18</td>
</tr>
<tr>
<td>JP7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 3 to rear panel</td>
<td>Installing a jumper on JP7 1-2 will disable the front panel STOP button and Input 3 will be routed to J18 &amp; J19</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel Stop button</td>
<td>Installing a jumper on JP7 2-3 will enable the front panel STOP button and Input 3 will not be available on J18 &amp; J19</td>
</tr>
<tr>
<td>JP8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1-2 for Input 2 to rear panel</td>
<td>Installing a jumper on JP8 1-2 will disable the front panel START button and Input 2 will be routed to J18 &amp; J19</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2-3 for front panel Start button</td>
<td>Installing a jumper on JP8 2-3 will enable the front panel START button and Input 2 will not be available on J18 &amp; J19</td>
</tr>
</tbody>
</table>

Table 15 Jumper Selection–JP1 – JP8
### Rear Panel LEDs

#### Fuse LEDs

<table>
<thead>
<tr>
<th>LED</th>
<th>Description</th>
<th>Possible States</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuse X</td>
<td>Indicates that the fuse on X axis is ok</td>
<td>Not illuminated, Fuse is blown</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Illuminated green, No error</td>
</tr>
<tr>
<td>Fuse Y</td>
<td>Indicates that the fuse on Y axis is ok</td>
<td>Not illuminated, Fuse is blown</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Illuminated green, No error</td>
</tr>
</tbody>
</table>

Table 16 Fuse LEDs

#### Enable LEDs

<table>
<thead>
<tr>
<th>LED</th>
<th>Description</th>
<th>Possible States</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENA X</td>
<td>Indicates that the axis is enabled</td>
<td>Not illuminated, Axis is not enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Illuminated green, Axis is enabled</td>
</tr>
<tr>
<td>ENA Y</td>
<td>Indicates that the axis is enabled</td>
<td>Not illuminated, Axis is not enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Illuminated green, Axis is enabled</td>
</tr>
</tbody>
</table>

Table 17 Enable LEDs
Interface Circuitry

Limit, Home, and Digital Input

To trigger the opto-isolated digital inputs, sink the digital input signal to the ground of the corresponding opto-supply.

Note: Alarm input for TB9 version is 5V TTL type.

Figure 5 Digital Inputs

Digital Outputs

For the opto-isolated outputs, the digital output signal will source from VS when the signal is turned on.

Figure 6 Digital Outputs
Encoder Input Connection

Both single-ended and differential quadrature encoder inputs are accepted.

When using single-ended encoders, use the /A, /B, and /Z inputs.

+5V supply and Ground signals are available to power the encoder. Make sure that the total current usage is less than 200mA for the +5V.

The maximum encoder frequency is 5MHz.

![Encoder Inputs Diagram](image)

Figure 7 Encoder Inputs

Analog Inputs

8 x 10-bit analog inputs are available on the DMM-0200. Use AI[1-8] command to read the analog input value. Range is from 0-5000 mV.

![Analog Inputs Diagram](image)

Figure 8 Analog Inputs
Where to Go Next

Continue with the next chapter, Chapter 3 - Installing the DMM-0200. Now that user has become familiar with the technical capabilities of the DMM-0200, the next section will outline the steps needed to install the DMM-0200 into your system.
Chapter 3
Installing the DMM-0200

About This Chapter

Introduction

This chapter describes how to hook up the DMM-0200 to your Dover Motion stages.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installing the System</td>
<td>36</td>
</tr>
<tr>
<td>Modes of Operation</td>
<td>36</td>
</tr>
<tr>
<td>Connecting to the DMM-0200</td>
<td>37</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>38</td>
</tr>
</tbody>
</table>
Installing the System

Figure 9 System Installation

Modes of Operation

1. **PC Controlled** – The DMM-0200 can be controlled using a PC and USB communications. Dover Motion provides a GUI for running and programming the DMM-0200. The software is called PMX-2EX-SA. See Chapter 5.

2. **Stand Alone Operation** – The DMM-0200 can be operated stand alone. No external communications are required. Motion can be initiated using the front panel START & STOP buttons and the PROGRAM SELECT thumbwheel. See Chapter 6.
Connecting to the DMM-0200

Installing the Motor and Encoder Cables

**Caution:** Before plugging in any motor and encoder cables, verify the mechanical operation of your Dover Motion stage. Verify that the stage moves freely as intended and that there is no potential for damage or personal injury.

**Note:** If you have any questions regarding the mechanical operation of your stage or the connecting to the DMM-0200, please call the Dover Motion customer service department:

Phone: 508-475-3400

Email: sales@dovermotion.com

Plug in the motor and encoder cables to the appropriate rear panel connectors. Plug in your motor and encoder connections before plugging in the power cable.

Installing the I/O Cables

**Caution:** Before plugging in the I/O cables, verify that your I/O devices operate as intended and there is no potential for damage or personal injury.

Plug in the I/O cables to the appropriate rear panel connectors. Plug in your I/O connections before plugging in the power cable.

Installing the Power Cord and Communications Cables

The DMM-0200 comes with a 2.4 meter (approx. 8 foot) power cord. You must supply the USB 2.0 cable.

To install the power cord, insert the female connector on the power cord into the male receptacle on the rear of the system enclosure. The USB 2.0 cable plugs into the USB connector on the rear panel. See the Rear Panel section for the locations of these connectors.

**Note:** After plugging in the power cord, the DMM-0200 is ready to be turned on.
Where to Go Next

Continue with the next chapter, *Chapter 4 - Installing the Application Software for the DMM-0200*. After hooking up the DMM-0200 to the stages, the user should now be able to install the application software, **PMX-2EX-SA Software** and start communicating to the DMM-0200. The next section describes the steps needed to properly install the application software.

Note: Using the PMX-2EX-SA Software is not required in order to communicate to the DMM-0200. The user may use Labview, Matlab, or a number of other software to communicate to the DMM-0200 through USB connection.
Chapter 4
Installing Application Software for the DMM-0200

About This Chapter

Introduction

This chapter describes the Application GUI software you need to run the DMM-0200 via a terminal.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Description</td>
<td>40</td>
</tr>
<tr>
<td>Installing PMX-2EX-SA Software</td>
<td>40</td>
</tr>
<tr>
<td>Installing PMX Drivers</td>
<td>41</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>42</td>
</tr>
</tbody>
</table>
Software Description

The software that is used to communicate and to run the DMM-0200 is the PMX-2EX-SA software. This software provides the user with an intuitive graphical user interface for running and programming the DMM-0200. The user also needs to install the PMX drivers to complement the software.

Note: Using the PMX-2EX-SA Software is not required in order to communicate to the DMM-0200. The user may use Labview, Matlab, or a number of other software to communicate to the DMM-0200 through USB. See the USB Communications section for communication details and associated ASCII commands.

Installing PMX-2EX-SA Software

GUI (PMX-2EX-SA) Install

Locate the CD that was shipped with your DMM-0200. From your Windows folder double click the “Performax_Installation_1.54” and follow the on screen instructions. See Figure 10.

Figure 10 Performax Installation
Installing PMX Drivers

Locate the CD that was shipped with your DMM-0200. From your Windows folder double click the “Arcus_Drivers_and_Tools_Setup_1.06” and follow the on screen instructions. See Figure 11.

![Figure 11 Driver Installation](image-url)
Where to Go Next

Continue with the next chapter, Chapter 5 - Running the DMM-0200 via PMX-2EX-SA Software. This next section describes how to use the software that was installed in section 4. Details and functions about the various screens are described.
Chapter 5
Running the DMM-0200 via PMX-2EX-SA Software

About This Chapter

Introduction
This chapter describes how to communicate with the DMM-0200 via USB connection to the user’s PC and how to use the PMX-2EX-SA software to control motion.

Topics
This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connecting a USB Cable</td>
<td>44</td>
</tr>
<tr>
<td>Establishing Communication with the DMM-0200</td>
<td>45</td>
</tr>
<tr>
<td>Using the Main Control Screen</td>
<td>46</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>54</td>
</tr>
</tbody>
</table>
Connecting a USB Cable

To communicate with the DMM-0200 via a terminal you will need a USB 2.0 cable not supplied with this DMM. See Table 3 for details of cable specifications.

Once you have located a USB cable, plug the “B” end into the DMM-0200 USB port. See Figure 12. Then plug the “A” end in any open USB port of your PC.

Figure 12 DMM-0200 USB Port
Establishing Communication with the DMM-0200

Locate the “SOFT-EXE-PMX-2EX-SA-114” icon on your desktop or locate it in your START menu.

Select the device you want to communicate with. Choose USB for USB communications for the DMM-0200.
Using the Main Control Screen

With the DMM-0200 GUI, you can control all motion via one “Main Control Screen”, see Figure 14. For additional information not found in this manual reference the “PMX-4EX-SA Manual” which is included on the shipping CD and installed onto your PC with this GUI.

Figure 14
1. Current pulse position (X/Y axis).
2. Current encoder position (X/Y axis).
3. Current speed (X/Y axis).
4. Motor status (X/Y axis).
   i. IDLE – Motor is not moving.
   ii. ACCEL – Motor is accelerating.
   iii. CONST – Motor is moving at constant speed.
   iv. DECEL – Motor is decelerating.
5. StepNLoop value (X/Y axis).
   i. NA – StepNLoop is disabled.
   ii. IDLE – Motor is not moving.
   iii. MOVING – Motor is moving.
   iv. CORRECTING – Motor is attempting to correct its position.
   v. STOPPING – Motor is stopping using deceleration.
   vi. ABORTING – Motor is stopping without deceleration.
   vii. JOGGING – Motor is jogging.
   viii. HOMING – Motor is homing using the home switch.
   ix. L-HOMING - Motor is homing using the limit switch.
   x. Z-HOMING – Motor is homing using the Z-index.
   xi. ERR-RANGE – The error range has been exceeded.
   xii. ERR-ATMPT – The maximum number of attempts has been made to correct the position.
   xiii. ERR-STALL – The motor has stalled.
   xiv. ERR-LIM – A limit switch has been hit.
6. StepNLoop delta value (X/Y axis).
7. –Limit, +Limit, Home input status (X/Y axis).
8. Z encoder index channel status (X/Y axis).
9. Clear status button and StepNLoop status(X/Y axis).
10. Move mode.
   i. ABS mode: On individual and interpolated move commands, motor will move to target position.
   ii. INC mode: On individual and interpolated move commands, motor will increase its position by the target position amount.
Control (B)

1. Global high speed, low speed, and acceleration values are entered here (X/Y axis). To give each axis individual speed parameters, enter HSPD[axis], LSPD[axis], and ACC[axis] into the command line in the “Terminal” section.
2. Select X/Y axis. Selection of both axes will result in synchronous movement.
3. Target position entered here (X/Y axis).
4. Enable driver power for the indicated motor (X/Y axis).
5. H+/H– – Home the motor at high speed using only the home sensor.
6. RSTOP/ISTOP – Stop the motor with deceleration using RSTOP. Stop the motor immediately using ISTOP.
7. RP/RE – Reset the position/encoder position.
8. ABS/INC – Set the move mode to absolute or incremental.
9. J+/J– – Jogs the motor in the positive or negative direction.
10. HL+/HL– – Home the motor at high speed and low speed using only the home sensor.
11. Z+/Z– – Only encoder index channel used for homing.
12. L+/L– – Home the motor using only the limit sensor.
13. ZH+/ZH– – Both encoder index and home sensor used for homing.
14. DAT/ABS – Move the motor to position zero by using DAT. Move the motor to the target position by using ABS.

On-The-Fly-Speed Control (C)

1. Select X/Y axis.
2. Select destination speed of the axis.
3. Select the acceleration used during an on-the-fly speed change.
4. Select the SSPD mode for the axis. See On-The-Fly Speed section for details.
5. Set the SSPD mode for the axis.
6. Set on-the-fly speed change. Acceleration will be taken from the “Accel” field.
Product Information (D)

Figure 18 Product Information (D)

Terminal (E)

Figure 19 Terminal (E)

1. Send commands to the PMX-2EX-SA through this terminal
2. Replies from the PMX-2EX-SA will be shown here.

On-The-Fly-Position Control (F)

Figure 20 On-The-Fly-Position Control (F)

1. Select X/Y axis.
2. Set the new target position of the axis.
3. Perform an on-the-fly position change.

Digital Input/Output (G)

Figure 21 Digital I/O Status (G)

a. Digital input status for DI1-DI8.
b. Digital output status for DO1-DO8.
c. Analog input status for AI1 and AI2 [0-5000 mV].
About (H)

Displays the current software and firmware versions

Setup (I)

- Polarity:
  a. Set direction/pulse/home/Z-index polarity for X/Y axis
  b. Set s-curve enable/disable for X/Y axis
  c. Set the encoder multiplier to 1X/2X/4X for X/Y axis
  d. Limit - Set the limit input polarity
  e. DO - Set the digital output polarity
  f. EO - Set the enable output polarity
  g. DI - Set the digital input polarity
  h. SA Err - Set the return jump line for standalone error handling
2. Boot Up
   a. DO Boot/EO Boot - Set the digital and enable output configuration on boot up
   b. Auto Run - Have the specified standalone program run on boot up.
3. Homing Parameters
   a. LCA - Set the limit correction amount for the specified axis
   b. HCA - Set the home correction amount for the specified axis
   c. Return to Zero - Return to zero after a homing routine.
5. StepNLoop Parameters (X/Y axis). See StepNLoop section for further details.
6. Device Name – Set the name of the device. Must be in the range of 2EX00 to 2EX99.
7. Misc
   a. Ignore Error - Set the IERR register to ignore the limit error status
   b. Enable Decel - Set the EDEC register to enable unique deceleration
8. STORE - Permanently store the downloaded parameters to flash memory.
9. Download/Upload - Download the current settings to the unit or upload the current settings from the unit.
10. Upload/Download parameters to and from RAM
11. Open/Save parameters to file.

**Variables (J)**

1. Current values of variables that cannot be stored to flash.
2. Current values of variables that can be stored to flash.
3. Send commands to the PMX-2EX-SA through this terminal.
Program File Control (K)

1. Open - Open a standalone program
2. Save - Save a standalone program
3. New - Clear the standalone program editor

Text Programming Box (L)

1. Text Program – Text box for writing and editing a standalone program.
2. Opens a larger Program Editor window for easier programming.
3. Clear Code Space – Clear the code space on the PMX-2EX-SA.
Compiler (M)

Figure 27 Compiler (M)

a. Compile - Compile code in text programming box into assembly level code that the PMX-2EX-SA can understand.
b. Download - Download the compiled code into memory. Note that the text based code must be compiled before download.
c. Upload - Upload standalone code that is currently on your PMX-2EX-SA. This automatically translates assembly level language to readable text-based code.
d. View - View compiled code for easy cutting and pasting.

Program Control (N)

Figure 28 Terminal (N)

1. Run – Standalone program is run.
2. Stop – Program is stopped.
3. Pause – Program that is running can be stopped.
4. Cont – Program that is paused can be continued.
5. XThread - Open the Standalone Program Control for all standalone programs.
6. Index - Current line of code that is being executed.
7. Status of standalone program:
   i. Idle – Program is not running.
   ii. Running – Program is running.
   iii. Paused – Program is paused.
   iv. Error – Program is in an error state.
Where to Go Next

Continue with the next chapter, Chapter 6 - Running the DMM-0200 via Front Panel Buttons. The next section describes how to run the DMM-0200 in standalone mode using the front panel buttons to control motion. Descriptions of the default motion programs installed on the DMM-0200 are also described.
Chapter 6
Running the DMM-0200 via Front Panel Buttons

About This Chapter

Introduction

This chapter describes how to run the DMM-0200 via the Front Panel Thumb Wheel and Start/ Jog+, Stop/ Jog- buttons for standalone operation. The sample motion programs installed from the factory are also described in greater detail.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front Panel Buttons</td>
<td>56</td>
</tr>
<tr>
<td>DMM-0200 Stored programs</td>
<td>57</td>
</tr>
<tr>
<td>Executing Stored Programs</td>
<td>57</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>58</td>
</tr>
</tbody>
</table>
Front Panel Buttons

The DMM-0200 front panel is made up of a Thumb wheel, Start/ Jog+, and Stop/ Jog- buttons, see Figure 29. These buttons can be used to run stored programs in the DMM-0200s memory. No USB communication to the DMM-0200 is required for standalone operation. The DMM-0200 will function as a completely standalone motion controller.

Figure 29 DMM-0200 Front Panel
DMM-0200 Stored programs

The DMM-0200 is shipped with 7 standard programs that can be executed via the Thumbwheel and Start/Stop buttons. See Table 18.

Note: The programs shipped with the DMM-0200 are SAMPLE programs. These programs are not intended to be the customer’s final solution. They are standard Dover Motion programs that may not be exactly suitable for the customer’s application. The user should create their own motion programs for specific motion profiles and control.

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Function</th>
<th>Pushing the START Button will...</th>
<th>Pushing the STOP Button will...</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Joystick Control</td>
<td>Enable external joystick control</td>
<td>Disable external joystick control</td>
<td>When the joystick control is enabled, starting another motion program will automatically disable the joystick and run the selected motion program. To enable the joystick again, the user must start program 0 again.</td>
</tr>
<tr>
<td>1</td>
<td>X Axis Jogging</td>
<td>Jog the X Axis positive</td>
<td>Jog the X Axis negative</td>
<td>Holding the START (or STOP) button will jog the axis. Releasing the START (or STOP) button will stop the jogging.</td>
</tr>
<tr>
<td>2</td>
<td>Y Axis Jogging</td>
<td>Jog the Y Axis positive</td>
<td>Jog the Y Axis negative</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>XY Homing</td>
<td>Homes all axis</td>
<td>Stop the homing routing</td>
<td>The homing program will home all axis sequentially one after another. The axis will jog to the negative limit, jog off the negative limit by 1 mm, and zero the position counter. The resulting position is the home or zero position.</td>
</tr>
<tr>
<td>6</td>
<td>XYZU Cycling</td>
<td>Cycles all axis</td>
<td>Stops the cycling</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Available for customer use</td>
<td>Start program 7</td>
<td>Stop program 7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Available for customer use</td>
<td>Start program 8</td>
<td>Stop program 8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Available for customer use</td>
<td>Start program 9</td>
<td>Stop program 9</td>
<td></td>
</tr>
</tbody>
</table>

Table 18 Standard Programs

Executing Stored Programs

To execute one of the stored programs, select that program number on the thumbwheel and then depress the Start button.

To stop the program, press the Stop button.
Where to Go Next

Continue with the next chapter, Chapter 7 - Editing Programs. Using either front panel buttons or the application software to control the DMM-0200, the DMM-0200 calls motion programs to run the stages. The next section describes how to edit those programs.
Chapter 7
Editing Programs

About This Chapter

Introduction

This chapter describes how to edit motion programs with the GUI. This chapter also describes the programming language for programs in the DMM-0200.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Editing a Stored Program</td>
<td>60</td>
</tr>
<tr>
<td>Standalone Program Specification</td>
<td>62</td>
</tr>
<tr>
<td>Programming Language Specification</td>
<td>64</td>
</tr>
<tr>
<td>Sample Programs</td>
<td>83</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>87</td>
</tr>
</tbody>
</table>
Editing a Stored Program

To edit a stored program with the GUI, you first need to open the program. You can either Upload the program from the controller’s memory or open a program saved on your PC. Within the GUI you can hit the Open or Upload button. See Figure 30.

Figure 30: Retrieving a Program
The program will now be in the Text window. The user can now edit the program as needed. After editing the programs, hit the Compile button and then the Download button to compile and download the new programs into the DMM-0200 memory. To run the new programs, hit the run button. See Figure 31.

Figure 31: Text Window

Note: When you upload the saved programs from the controller’s memory, you will be uploading Dover Motion’s standard motion programs described in the DMM-0400 Stored Programs section.

If you want to keep Dover Motion’s standard programs, you will have to edit the uploaded file (which contains Dover Motion’s standard programs) to include your new motion programs.

If you do not want to keep Dover Motion’s standard programs, Clear Code Space, create your own motion programs, and download to the controller’s memory.
Standalone Program Specification

Standalone Program Specification:
Memory size: 1,275 assembly lines.
Note: Each line of pre-compiled code equates to 1-4 lines of assembly lines.

WAIT Statement
When writing a standalone program, it is generally necessary to wait until a motion is completed before moving on to the next line. In order to do this, the WAIT statement must be used. See the examples below:

In the example below, the variable V1 will be set immediately after the X10000 move command begins; it will not wait until the controller is idle.

```
X10000 ;* Move to position 0
V1 = 100
```

Conversely, in the example below, the variable V1 will not be set until the motion has been completed. V1 will only be set once the controller is idle.

```
X10000 ;* Move to position 0
WAITX ;* Wait for the move to complete
V1 = 100
```

Multi-Threading
PMX-2EX-SA supports the simultaneous execution of up to 2 standalone programs. Programs 0, 1 are controlled via the SR0 and SR1 commands respectively. For examples of multi-threading, please refer to the Example Stand-alone Programs section.

Note: Sub-routines can be shared by different threads.

Error Handling
If an error occurs during standalone execution (i.e. limit error), the program automatically jumps to SUB 31. If SUB 31 is NOT defined, the program will cease execution and go to error state. If SUB 31 is defined by the user, the code within PMX-2EX-SA Manual page 42 rev 2.2 SUB 31 will be executed. The return jump line will be determined by value of the 13th-bit of the POL register. If the value is 0, the return jump line will be the line that caused the error. Otherwise, the return jump line will be line 0.

Calling subroutines over communication
Once a subroutine is written into the flash, they can be called via USB communication using the GS command. The subroutines are referenced by their subroutine number [0-31]. If a subroutine number is not defined, the controller will return with an error.
Standalone Run on Boot-Up
Standalone can be configured to run on boot-up using the **SLOAD** command. See description below:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Standalone Program 0</td>
</tr>
<tr>
<td>1</td>
<td>Standalone Program 1</td>
</tr>
</tbody>
</table>

Table 19 Standalone Run on Boot-Up

Storing to Flash
The following items are stored to flash:

<table>
<thead>
<tr>
<th>ASCII Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DN</td>
<td>Device name</td>
</tr>
<tr>
<td>DOBOOT</td>
<td>DO configuration at boot-up</td>
</tr>
<tr>
<td>EDEC</td>
<td>Unique deceleration enable</td>
</tr>
<tr>
<td>EOB OOT</td>
<td>EO configuration at boot-up</td>
</tr>
<tr>
<td>IERR</td>
<td>Ignore limit error enable</td>
</tr>
<tr>
<td>HCA, HCA[axis]</td>
<td>Home Correction Amount</td>
</tr>
<tr>
<td>LCA, LCA[axis]</td>
<td>Limit Correction Amount</td>
</tr>
<tr>
<td>POL[axis]</td>
<td>Polarity settings</td>
</tr>
<tr>
<td>SCV[axis]</td>
<td>S-curve enable</td>
</tr>
<tr>
<td>SL[axis], SLR[axis], SLE [axis], SLT[axis], SLA[axis]</td>
<td>StepNLoop parameters</td>
</tr>
<tr>
<td>JO, JF, JV[1-6], JL[1-8]</td>
<td>Joystick settings</td>
</tr>
<tr>
<td>RZ</td>
<td>Return to zero (homing)</td>
</tr>
<tr>
<td>SLOAD</td>
<td>Standalone program run on boot-up parameter</td>
</tr>
<tr>
<td>TOC</td>
<td>Time-out counter reset value</td>
</tr>
<tr>
<td>V32-V63</td>
<td>Note that on boot-up, V0-V31 are reset to value 0</td>
</tr>
</tbody>
</table>

Table 20 Storing to Flash

**Note:** When a standalone program is downloaded, the program is immediately written to flash memory.
Programming Language Specification

;  
Description:  
Comment notation. In programming, comment must be in its own line. 
Syntax:  
; [Comment Text]  
Examples:  
; ***This is a comment  
JOGX+ ;***Jogs X axis to positive direction  
DELAY= 1000 ;***Wait 1 second  
ABORT ;***Stop immediately all axes including X axis

ABORT  
Description:  
Motion: Immediately stops all axes if in motion without deceleration.  
Syntax:  
ABORT  
Examples:  
JOGX+ ;***Jogs X axis to positive direction  
DELAY= 1000 ;***Wait 1 second  
ABORT ;***Stop immediately all axes including X axis

ABORT[axis]  
Description:  
Motion: Immediately stops individual axis without deceleration.  
Syntax:  
ABORT[axis]  
Examples:  
JOGX+ ;***Jogs X axis to positive direction  
JOGY+ ;***Jogs Y axis to positive direction  
ABORTX ;***Stop the X-axis immediately

ABS  
Description:  
Motion: Changes all move commands to absolute mode.  
Syntax:  
ABS  
Examples:  
ABS ;***Change to absolute mode  
PX=0 ;***Change X position to 0  
X1000 ;***Move X axis to position 1000  
WAITX  
X2000 ;***Move X axis to position 2000  
WAITX  
ABORT ;***Stop immediately all axis including X axis
**ACC**

**Description:**
- **Read:** Get acceleration value
- **Write:** Set acceleration value.

Value is in milliseconds.
Range is from 1 to 10,000

**Syntax:**
- **Read:** [variable] = ACC
- **Write:** ACC = [value]
- **Conditional:** IF ACC=[variable]  
  ENDIF
  IF ACC=[value]  
  ENDIF

**Examples:**
- ACC=300 ;***Sets the acceleration to 300 milliseconds
- V3=500 ;***Sets the variable 3 to 500
- ACC=V3 ;***Sets the acceleration to variable 3 value of 500

**ACC[axis]**

**Description:**
- **Read:** Get individual acceleration value
- **Write:** Set individual acceleration value.

Value is in milliseconds.
Range is from 1 to 10,000

**Syntax:**
- **Read:** [variable] = ACC[axis]
- **Write:** ACC[axis] = [value]
- **Conditional:** IF ACC[axis]=[variable]  
  ENDIF
  IF ACC[axis]=[value]  
  ENDIF

**Examples:**
- ACCX=300 ;***Sets the X acceleration to 300 milliseconds
- V3=500 ;***Sets the variable 3 to 500
- ACCX=V3 ;***Sets the X acceleration to variable 3 value of 500

**AI[1-2]**

**Description:**
- **Read:** Get the analog input value. PMX-2EX-SA has 2 analog inputs. Range is from 0-5000 mV

**Syntax:**
- **Read:** [variable] = AI[1-2]
- **Conditional:** IF AI[1-2]=[variable]  
  ENDIF
  IF AI[1-2]=[value]  
  ENDIF

**Examples:**
- IF AI1 < 500  
  DO=1 ;***If analog input 1 is less than 500, set DO=1  
  ENDIF
**DEC**
Description:
- **Read:** Get deceleration value
- **Write:** Set deceleration value
Value is in milliseconds.
Syntax:
- **Read:** \[variable\] = DEC
- **Write:** DEC = [value]
  
  DEC = \[variable\]
Examples:
- DEC=300 ;***Sets the deceleration to 300 milliseconds
- V3=500 ;***Sets the variable 3 to 500
- DEC=V3 ;***Sets the deceleration to variable 3 value of 500

**DEC[axis]**
Description:
- **Read:** Get individual deceleration value
- **Write:** Set individual deceleration value.
  Value is in milliseconds.
Syntax:
- **Read:** \[variable\] = DEC[axis]
- **Write:** DEC[axis] = [value]
  
  DEC[axis] = \[variable\]
Conditional:  
  IF ACC[axis]=\[variable\]
  ENDIF
  
  IF ACC[axis]=[value]
  ENDIF
Examples:
- DECX=300 ;***Sets the X deceleration to 300 milliseconds
- V3=500 ;***Sets the variable 3 to 500
- DECX=V3 ;***Sets the X deceleration to variable 3 value of 500

**DELAY**
Description:
- Set a delay (1 ms units)
Syntax:
- Delay=[Number](1msunits)
Examples:
- JOGX+ ;***Jogs X axis to positive direction
- DELAY=10000 ;***Wait 10 second
- ABORT ;***Stop with deceleration all axes including X axis
- EX=0 ;***Sets the current X encoder position to 0
- EY=0 ;***Sets the current Y encoder position to 0

**DI**
Description:
- **Read:** Gets the digital input value
- Performax 4EX has 8 digital inputs
Syntax:
Read: \[\text{variable}\] = DI

Conditional:

IF DI = [variable]
ENDIF
IF DI = [value]
ENDIF

Examples:

IF DI = 255
DO = 1 ; **If no digital inputs are triggered, set DO = 1
ENDIF

\textbf{DI(1-8J)}

Description:

\textbf{Read:} Gets the digital input value
Performax 2EX has 8 digital inputs

Syntax:

\textbf{Read:} \[\text{variable}\] = DI[1-8]

Conditional:

IF DI[1-8] = [variable]
ENDIF
IF DI[1-8] = [0 or 1] ENDIF

Examples:

IF DI1 = 1
DO = 1 ; **If digital input 1 is triggered, set DO = 1
ENDIF

\textbf{DO}

Description:

\textbf{Read:} Gets the digital output value
\textbf{Write:} Sets the digital output value
Performax 2EX has 8 digital outputs

Syntax:

\textbf{Read:} \[\text{variable}\] = DO
\textbf{Write:} \[\text{variable}\] = DO = [value]

Conditional:

IF DO = [variable]
ENDIF
IF DO = [value]
ENDIF

Examples:

DO = 7 ; ***Turn first 3 bits on and rest off

\textbf{DO(1-8J)}

Description:

\textbf{Read:} Gets the individual digital output value
\textbf{Write:} Sets the individual digital output value Performax 2EX has 8 digital outputs

Syntax:

\textbf{Read:} \[\text{variable}\] = DO[1-8]
\textbf{Write:} \[\text{variable}\] = DO[1-8] = [0 or 1]
DO[1-8] = [variable]
Conditional: 
  IF DO[1-8]=variable
  ENDF
  IF DO[1-8]=[0 or 1]

ENDIF

Examples:
DO7=1 ;***Turn DO7 on
DO6=1 ;***Turn DO6 on

E[axis]
Description:
Read: Gets the current encoder position
Write: Sets the current encoder position
Syntax:
Read: variable = E[axis]
Write: E[axis] = value
E[axis] = variable
Conditional: 
  IF E[axis]=variable
  ENDF
  IF E[axis]=value
  ENDF

Examples:
JOGX+ ;***Jogs X axis to positive direction
DELAY=1000 ;***Wait 1 second
ABORT ;***Stop with deceleration all axes including X axis
EX=0 ;***Sets the current X encoder position to 0
EY=0 ;***Sets the current Y encoder position to 0

ECLEAR[axis]
Description:
Write: Clears error status.
Syntax:
Write: ECLEAR[axis]

Examples:
ECLEARX ;***Clears error of axis X
ECLEARY ;***Clears error of axis Y

ELSE
Description:
Perform ELSE condition check as a part of IF statement
Syntax:
ELSE

Examples:
IF V1=1
  X1000, ***If V1 is 1, then move to 1000
  WAITX
ELSE
X-1000 ;***If V1 is not 1, then move to -
1000 WAITX
ENDIF

ELSEIF
Description:
Perform ELSEIF condition check as a part of the IF statement

Syntax:
ELSEIF [Argument 1] [Comparison] [Argument 2]
[Argument] can be any of the following:
Numerical value
Pulse or Encoder Position
Digital Output
Digital Input
Enable Output
Motor Status

[Comparison] can be any of the following
= Equal to
> Greater than
< Less than
>= Greater than or equal to
<= Less than or equal to
!= Not Equal to

Examples:
IF V1=1
X1000
WAITX
ELSEIF V1=2
X2000
WAITX
ELSE
X0
WAITX
ENDIF

END
Description:
Indicate end of program.
Program status changes to idle when END is reached.
Note: Subroutine definitions should be written AFTER the END statement

Syntax:
END
Examples:
X0
WAITX
X1000
WAITX
END
**ENDIF**
Description:
Indicates end of IF operation
Syntax:
ENDIF Examples:
IF V1=1
  X1000
  WAITX
ENDIF

**ENDSUB**
Description:
Indicates end of subroutine
When ENDSUB is reached, the program returns to the previously called subroutine.
Note: Subroutine 31 is reserved for error handling
Syntax:
ENDSUB
Examples:
GOSUB 1
END
SUB 1
  X0
  WAITX
  X1000
  WAITX
ENDSUB

**ENDWHILE**
Description:
Indicate end of WHILE loop
Syntax:
ENDWHILE
Examples:
WHILE V1=1 ;***While V1 is 1 continue to loop
  X0
  WAITX
  X1000
  WAITX
ENDWHILE ;***End of while loop so go back to WHILE

**EO**
Description:
Read: Gets the enable output value
Write: Sets the enable output value
Performax 2EX has 2 enable outputs.
Syntax:
Read: [variable] = EO
Write: EO = [value]
  EO = [variable]
Conditional: IF EO= [variable]
ENDIF
IF EO= [value]
ENDIF

Examples:
EO=3 ;***Turn first 2 bits of enable outputs
IF V1=1
EO=V2 ;***Enable output according to variable 2
ENDIF

EO[1-2J
Description:
Read: Gets the individual enable output value 
Write: Sets the individual enable output value 
Performax 2EX has 2 enable outputs.

Syntax:
Read: [variable] = EO[1-2]
Write: EO[1-2] = [0 or 1]
EO[1-2] = [variable]
Conditional: IF EO= [variable]
ENDIF
IF EO= [value]
ENDIF

Examples:
EO1=1 ;***Turn enable output 1 on
IF V1=1
EO2=V2 ;***Enable output 2 according to variable 2
ENDIF

GOSUB
Description:
Perform go to subroutine operation
Subroutine range is from 1 to 32.
Note: Subroutine definitions should be written AFTER the END statement
Subroutine 31 is reserved for error handling
Syntax:
GOSUB [subroutine number]
[Subroutine Number] range is 1 to 32
Examples:
GOSUB 1
END
SUB 1
X0
WAITX X1000
WAITX ENDSUB

HLHOME[axis][+ or -]
Description:
Command: Perform low speed homing using current high speed, low speed, and acceleration.
Syntax:
HLHOME[Axis][+ or -]
Examples:
HSPD
Description:
Read: Gets high speed. Value is in pulses/second
Write: Sets high speed. Value is in pulses/second.
Range is from 1 to 6,000,000

Syntax:
Read: [variable] = HSPD
Write: HSPD = [value]
       HSPD = [variable]
Conditional: IF HSPD = [variable]
             ENDIF
            IF HSPD = [value]
             ENDIF

Examples:
HSPD=10000 ;***Sets the high speed to 10,000 pulses/sec
V 1=2500 ;***Sets the variable 1 to 2,500
HSPD=V1 ;***Sets the high speed to variable 1 value of 2500

HSPD[axis]
Description:
Read: Gets individual high speed. Value is in pulses/second
Write: Sets individual high speed. Value is in pulses/second.
Range is from 1 to 6,000,000

Syntax:
Read: [variable] = HSPD[axis]
Write: HSPD[axis] = [value]
       HSPD[axis] = [variable]
Conditional: IF HSPD[axis] = [variable]
             ENDIF
            IF HSPD[axis] = [value]
             ENDIF

Examples:
HSPDY=10000 ;***Sets the Y high speed to 10,000 pulses/sec
V 1=2500 ;***Sets the variable 1 to 2,500
HSPDY=V1 ;***Sets the Y high speed to variable 1 value of 250

IF
Description:
Perform IF condition check

Syntax:
IF [Argument 1] [Comparison] [Argument 2]
   [Argument] can be any of the
   following: Numerical value
Pulse or Encoder Position
Digital Output Digital Input
Enable Output Motor Status

[Comparison] can be any of the following
- = Equal to
- > Greater than
- < Less than
- >= Greater than or equal to
- <= Less than or equal to
- != Not Equal to

Examples:
IF V1=1
  X1000
  WAITX
ENDIF

**INC**
Description:
**Command:** Changes all move commands to incremental mode.
**Syntax:**
INC

Examples:
INC ;***Change to increment mode
PX=0 ;***Change X position to 0
X1000 ;***Move X axis to position 1000 (0+1000)
WAITX
X2000 ;***Move X axis to position 3000 (1000+2000)
WAITX
ABORT ;***Stop immediately all axes including X axis

**JOG[axis]**
Description:
**Command:** Perform jogging using current high speed, low speed, and acceleration.
**Syntax:**
JOG [Axis] [+ or -]

Examples:
JOGX+ ;***Jogs X axis in positive direction
JOGY- ;***Jogs Y axis in negative direction

**JOYENA**
Description:
**Write:** Enable joystick feature for axis
**Syntax:**
Write: JOYENA=[0,1]

Examples:
JOYENA=1 ;***Enable joystick feature on X axis only

**JOYHS[axis]**
Description:
Write:  Set high speed setting for joystick control
Syntax:
  Write: JOYHS [axis] = [value]  
          JOYHS [axis] = [variable]
Examples:
  JOYHSX=10000 ;***High speed of X axis is set to 10,000 pps
  JOYHSY=20000 ;***High speed of Y axis is set to 20,000 pps

**JOYDEL[axis]**
Description:  Write: Set maximum delta value of change in speed for joystick control
Syntax:
  Write: JOYDEL[axis] = [value]  
          JOYDEL[axis] = [variable]
Examples:
  JOYDELX= 100 ;***Speed delta of X axis is set to 100 pps
  JOYDELU=200 ;***Speed delta of Y axis is set to 200 pps

**JOYNO[axis]**
Description:  Write: Set negative outer limit for joystick control
Syntax:
  Write: JOYNO[axis] = [value]  
          JOYNO[axis] = [variable]
Examples:
  JOYNOX= -10000 ;*** negative outer limit of x-axis set to -10000
  JOYNIX= -9000 ;*** negative inner limit of x-axis set to -9000
  JOYPIX=9000 ;*** positive inner limit of x-axis set to 9000
  JOYPOX=10000 ;*** positive outer limit of x-axis set to 10000

**JOYN[axis]**
Description:  Write: Set negative inner limit for joystick control
Syntax:
  Write: JOYN[axis] = [value]  
          JOYN[axis] = [variable]
Examples:
  See JOYNO[axis]

**JOYPI[axis]**
Description:  Write: Set positive inner limit for joystick control
Syntax:
  Write: JOYPI[axis] = [value]  
          JOYPI[axis] = [variable]
Examples:
  See JOYNO[axis]
**JOYPO[axis]**

**Description:**
Set positive outer limit for joystick control

**Syntax:**
Write: JOYPO[axis] = [value]
      JOYPO[axis] = [variable]

**Examples:**
See JOYN[axis]

**JOYTOL[axis]**

**Description:**
Set zero tolerance value for joystick control

**Syntax:**
Write: JOYTOL[axis] = [value]
      JOYTOL[axis] = [variable]

**Examples:**
JOYTOLX=10 ;*** zero tolerance value of x-axis set to 10

**LHOME[axis][+ or -]**

**Description:**
Perform limit homing using current high speed, low speed, and acceleration.

**Syntax:**
LHOME[axis][+ or -]

**Examples:**
LHOMEX+ ;***Limit homes X axis in positive direction
WAITX
LHOMEY- ;***Limit homes Z axis in negative direction

**LSPD**

**Description:**
Get low speed. Value is in pulses/second.

**Write:**
Set low speed. Value is in pulses/second. Range is from 1 to 6M

**Syntax:**
Read: [variable] =LSPD
Write: LSPD=[long value]
      LSPD=[variable]

**Conditional:**
IF LSPD=[variable]
ENDIF
IF LSPD= [value]
ENDIF

**Examples:**
LSPD= 1000 ;***Sets the start low speed to 1,000 pulses/sec
V 1=500 ;***Sets the variable 1to 500
LSPD=V1 ;***Sets the start low speed to variable 1 value of 500

**LSPD[axis]**

**Description:**
Get individual low speed. Value is in pulses/second.

**Write:**
Set individual low speed. Value is in pulses/second. Range is from 1 to 6M

**Syntax:**
Read: [variable] =LSPD[axis]
Write:  LSPD[axis]=long value
LSPD[axis]=variable

Conditional:  IF LSPD[axis]=variable
ENDIF
IF LSPD[axis]=value
ENDIF

Examples:
LSPDZ=1000 ;***Sets the Z low speed to 1,000 pulses/sec
V 1=500 ;***Sets the variable 1 to 500
LSPDZ=V1 ;***Sets the Z low speed to variable 1 value of 500

**MST[axis]**
Description:
Command: Get motor status of axis
Syntax:
MST[axis]
Examples:
IF MSTX=0

DIO=6
ELSEIF MSTY=0
DIO=3
ENDIF

**P[axis]**
Description:
Read: Gets the current pulse position
Write: Sets the current pulse position
Syntax:
Read: Variable = P[axis]
Write: P[axis]=value
P[axis] = variable
Conditional:  IF P[axis]=variable
ENDIF
IF P[axis]=value
ENDIF

Examples:
JOGX+ ;***Jogs X axis to positive direction
DELAY= 1000 ;***Wait 1 second
ABORT ;***Stop without deceleration all axes including X axis
PX=0 ;***Sets the current pulse position to 0

**PRG**
Description:
Indicates the start of a program
When END is reached, the program is concluded
Syntax:
PRG [program number]
Examples:
PRG 0 ;***Program 0
X8000
END
PRG 1 ;***Program 1
Y1000
END

PS[axis]
Description: Get the current pulse position of an axis
Syntax:
Read: Variable = PS[axis]
Conditional: IF PS[axis]=variable
ENDIF
IF PS[axis]=value
ENDIF
Examples:
JOGX+ ;***Jogs X axis to positive direction
DELAY= 1000 ;***Wait 1 second
ABORT ;***Stop without deceleration all axes including X axis
V1=PSX ;***Sets variable 1 to pulse X
JOGY+ ;***Jogs Y axis to positive direction
V2=PSY ;***Sets variable 2 to pulse Y

SCV[axis]
Description: Read: Get individual s-curve enable. Value is 0 or 1.
Write: Set individual s-curve enable.
Range is from 0 or 1
Syntax:
Read: [variable]=SCV[axis]
Write: SCV[axis]=[0 or 1]
SCV[axis]=variable
Note: If s-curve is enabled for an axis, on-the-fly speed feature cannot be used for the corresponding axis.
Examples:
SCVX=1 ;***Sets X axis to use s-curve acceleration: on-the-fly speed
; change is NOT allowed for this axis.
SCVY=0 ;***Sets Y axis to use s-curve acceleration: on-the-fly speed
; change is allowed for this axis.

SL[axis]
Description: Write: Set individual StepNLoop enable. Range is from 0 or 1
Syntax:
Write: SL[axis]=[0 or 1]
Examples:
SLX=1 ;***Enables StepNLoop control for the X axis.
SLY=0 ;***Disables StepNLoop control for the Y axis.

SLS[axis]
Description:

**Command:** Get the StepNLoop status of axis

**Syntax:**

```
SLS [Axis]
V[Value] = SLS [Axis]
```

**Examples:**

```
IF SLSX=0
  DIO=6
ELSEIF SLSY=0
  DIO=3
ENDIF
```

**SR[0,1]**

Description:

**Write:** Set the standalone control for the specified standalone program

**Syntax:**

```
Write: SR[0-1] = [0-3]
```

**Examples:**

```
IF DI1=1 ; If digital input 1 is on
  SR0=0 ; Turn off standalone program 0
ENDIF
```

**SSPD[axis]**

Description:

**Write:** Set on-the-fly speed change for an individual axis.
Range is from 1 to 6,000,000 PPS

**Syntax:**

```
Write: SSPD[axis] = [value]
SSPD[axis] = [variable]
```

**Note:** If s-curve is enabled for an axis, on-the-fly speed feature cannot be used for the corresponding axis.

**Examples:**

```
SCVX=0 ;***Disable s-curve acceleration for X-axis
HSPDX= 1000 ;***X-axis high speed
LSPDX=100 ;***Set X-axis low speed
ACCX= 100 ; **Set X-axis acceleration
JOGX+ ;***Jogs X axis to positive direction
DELAY= 1000 ;***Wait 1 second
SSPDX=3000 ;***Change speed on X-axis on-the-fly to 3000 PPS
```

**SSPDM[axis]**

Description:

**Write:** Set individual on-the-fly speed change mode Range is from 0 to 9

**Syntax:**

```
Write: SSPDM[axis] = [0-9]
SSPDM[axis] = [variable]
```

**Examples:**

```
SCVX=0 ;***Disable s-curve acceleration for X-axis
HSPDX= 1000 ;***X-axis high speed
LSPDX=100 ;***Set X-axis low speed
ACCX= 100 ; **Set X-axis acceleration
```
JOGX+ ;***Jogs X axis to positive direction
DELAY= 1000 ;***Wait 1 second
SSPDMX=1 ;***Set on-the-fly speed change mode to 1
ACCX=20000 ;***Set acceleration to 20 seconds
SSPDX=190000 ;***Change speed on X-axis on-the-fly to 190000 PPS

**STOP**

Description:
Command: Stop all axes if in motion with deceleration. Previous acceleration value is used for deceleration.
Syntax:
STOP
Examples:
JOGX+ ;***Jogs X axis to positive direction
DELAY=1000 ;***Wait 1 second
STOP ;***Stop with deceleration all axes including X axis

**STOP[axis]**

Description:
Stop individual axis if in motion with deceleration. Previous acceleration value is used for deceleration.
Syntax:
STOP[axis]
Examples:
JOGX+ ;***Jogs X axis to positive direction
DELAY=1000 ;***Wait 1 second
JOGY+ ;***Jogs Y axis to positive direction
DELAY=1000 ;***Wait 1 second
STOPX ;***Stop with deceleration X axis only

**STORE**

Description:
Store the device settings and the second half of variables (V32-V63) to flash
Syntax:
STORE
Examples:
V32=100
V33=200
STORE ;***Values of V1 and V2 will now be preserved after power cycle

**SUB**

Description:
Indicates start of subroutine
Syntax:
SUB [subroutine number]
[Subroutine Number] range is 0 to 31
Note: Subroutine 31 is reserved for error handling
Examples:
GOSUB 1
END
SUB 1
  X0
  WAITX
  X1000
  WAITX
ENDSUB

**TOC**
Description:
Sets the communication time-out parameter. Value is in milli-seconds.
Syntax:
TOC=[long value]
Examples:
TOC=10000 ;***Sets time-out parameter to 10 seconds

**V**
Description:
Assign to variable. Performax 2EX has 64 variables [V0-V63]
Syntax:
V[Variable Number] = [Argument]
V[Variable Number] = [Argument 1] [Operation] [Argument2]
*Special case for BIT NOT:*
V[Variable Number] = ~[Argument]
[Argument] can be any of the following:
- Numerical value
- Pulse or Encoder Position
- Digital Output
- Digital Input
- Enable Output
- Motor Status
[Operation] can be any of the following
- Addition
- Subtraction
- Multiplication
- Division
- Modulus
- Bit Shift Right
- Bit Shift Left
- Bit AND
- Bit OR
- Bit NOT
Examples:
V1=12345 ;***Set Variable 1 to 123
V2=V1+1 ;***Set Variable 2 to V1 plus 1
V3=DI ;***Set Variable 3 to digital input value
V5=~EO ;***Sets Variable 5 to bit NOT of enable output value

*Note:* On the STORE command, the second half of generalpurpose variable registers (V32-V63) are stored to flash. Their values will be preserved after power cycle.
**WAIT**

Description:
Tell program to wait until move on the certain axis is finished before executing next line.

Syntax:
```
WAIT[axis]
X[variable]
```

Examples:
```
X10000 ;***Move X Axis to position 10000
WAITX ;***Wait until X Axis move is done
DO=5 ;***Set digital output
Y3000 ;***Move Y Axis to 3000
WAITY ;***Wait until Y Axis move is done
```

**WHILE**

Description:
Perform WHILE loop

Syntax:
```
WHILE [Argument 1] [Comparison] [Argument 2]
[Argument] can be any of the following:
  Numerical value
  Pulse or Encoder Position
  Digital Output
  Digital Input
  Enable Output
  Motor Status
[Comparison] can be any of the following
  =  Equal to
  >  Greater than
  <  Less than
  >= Greater than or equal to
  <= Less than or equal to
  != Not Equal to
```

Examples:
```
WHILE V1=1 ;***While V1 is1 continue to loop
  X0
  WAITX X1000
  WAITX
ENDWHILE
```

**X**

Description:
Command: Perform X axis move to target location
With other Axis moves in the same line, linear interpolation move is done.

Syntax:
```
X[value]
X[variable]
```

Examples:
Y
Description:
  Command: Perform Y axis move to target location
  With other Axis moves in the same line, linear interpolation move is done.
Syntax:
  Y[value]
  Y[variable]
Examples:
  Y10000       ;***MoveYAxis to position 10000
  WAITY
  X2000Y3000   ;***MoveXto2000andYto3000inlinearinterpolationmove
  WAITY
  V10 = 1200   ;***Set variable 10 value to 1200
  XV10         ;***Move X Axis to variable 10 value
  WAITY

ZHOME[axiaI[+ or I
Description:
  Command: Perform Z-homing using current high speed, low speed, and acceleration.
Syntax:
  ZHOME[Axis][+ or -]
Examples:
  ZHOMEX+       ;***Z Homes X axis in positive direction
  ZHOMEZ-       ;***Z Homes Z axis in negative direction

ZOME[axiaJ[+ or J
Description:
  Command: Perform Zoming using current high speed, low speed, and acceleration.
Syntax:
  ZOME[Axis][+ or -]
Examples:
  ZOMEX+        ;***Homes X axis in positive direction
  ZOMEZ-        ;***Homes Z axis in negative direction
Sample Programs

Standalone Example Program 1 – Single Thread
Task: Set the high speed and low speed and move the motor to 1000 and back to 0.

```
HSPD=20000 ;* Set the high speed to 20000 pulses/sec
LSPD=1000 ;* Set the low speed to 1000 pulses/sec
ACC=300 ;* Set the acceleration to 300 msec
EO=1 ;* Enable the motor power
X1000 ;* Moveto1000
WAITX ;* Wait for X-axis move to complete
X0 ;* Move to zero
WAITX ;* Wait for X-axis move to complete
END ;* End of the program
```

Standalone Example Program 2 – Single Thread
Task: Move the motor back and forth indefinitely between position 1000 and 0.

```
HSPD=20000 ;* Set the high speed to 20000 pulses/sec
LSPD=1000 ;* Set the low speed to 1000 pulses/sec
ACC=300 ;* Set the acceleration to 300 msec
EO=1 ;* Enable the motor power
WHILE 1=1 ;* Forever loop
X0 ;* Movetozero
WAITX ;* Wait for X-axis move to complete
X1000 ;* Moveto1000
WAITX ;* Wait for X-axis move to complete
ENDWHILE ;* Go back to WHILE statement
END
```

Standalone Example Program 3 – Single Thread
Task: Move the motor back and forth 10 times between position 1000 and 0.

```
HSPD=20000 ;* Set the high speed to 20000 pulses/sec
LSPD=1000 ;* Set the low speed to 1000 pulses/sec
ACC=300 ;* Set the acceleration to 300 msec
EO=1 ;* Enable the motor power
V1=0 ;* Set variable 1 to value 0
WHILE V1<10 ;* Loop while variable 1 is less than 10
X0 ;* Movetozero
WAITX ;* Wait for X-axis move to complete
X1000 ;* Moveto1000
WAITX ;* Wait for X-axis move to complete
V1=V1+1 ;* Increment variable 1
ENDWHILE ;* Go back to WHILE statement
END
```
Standalone Example Program 4 – Single Thread
Task: Move the motor back and forth between position 1000 and 0 only if the digital input 1 is turned on.

HSPD=20000;* Set the high speed to 20000 pulses/sec
LSPD=1000;* Set the low speed to 1000 pulses/sec
ACC=300;* Set the acceleration to 300 msec
EO=1;* Enable the motor power
WHILE 1=1;* Forever loop
  IF DI1=1;* If digital input 1 is on, execute the statements
    X0;* Move to zero
    WAITX;* Wait for X-axis move to complete
    X1000;* Move to 1000
    WAITX;* Wait for X-axis move to complete
  ENDIF
ENDWHILE
END

Standalone Example Program 5 – Single Thread
Task: Using a subroutine, increment the motor by 1000 whenever the DI1 rising edge is detected.

HSPD=20000;* Set the high speed to 20000 pulses/sec
LSPD=1000;* Set the low speed to 1000 pulses/sec
ACC=300;* Set the acceleration to 300 msec
EO=1;* Enable the motor power
V1=0;* Set variable 1 to zero
WHILE 1=1;* Forever loop
  IF DI1=1;* If digital input 1 is on, execute the statements
    GOSUB 1;* Move to zero
  ENDIF
ENDWHILE
END

SUB 1
  XV1;* Move to V1 target position
  WAITX;* Wait for X-axis move to complete
  V1=V1+1000;* Increment V1 by 1000
  WHILE DI1=1;* Wait until the DI1 is turned off so that
    ENDWHILE;* 1000 increment is not continuously done
ENDSUB

Standalone Example Program 6 – Single Thread
Task: If digital input 1 is on, move to position 1000. If digital input 2 is on, move to position 2000. If digital input 3 is on, move to 3000. If digital input 5 is on, home the motor in negative direction. Use digital output 1 to indicate that the motor is moving or not moving.

HSPD=20000;* Set the high speed to 20000 pulses/sec
LSPD=1000;* Set the low speed to 1000 pulses/sec
ACC=300;* Set the acceleration to 300 msec
EO=1;* Enable the motor power
WHILE 1=1;* Forever loop
IF DI1=1 ;*If digital input 1 is on

    X1000 ;* Move to 1000
    WAITX ;* Wait for X-axis move to complete
ELSEIF DI2=1 ;* If digital input 2 is on
    X2000 ;* Move to 2000
    WAITX ;* Wait for X-axis move to complete
ELSEIF DI3=1 ;* If digital input 3 is on
    X3000 ;* Move to 3000
    WAITX ;* Wait for X-axis move to complete
ELSEIF DI5=1 ;* If digital input 5 is on
    HOMEX- ;* Home the motor in negative direction
    WAITX ;* Wait for X-axis move to complete
ENDIF

V1=MSTX ;* Store the motor status to variable 1
V2=V1&7 ;* Get first 3 bits
IF V2!=0
DO1=1
ELSE
DO1=0
ENDIF
ENDWHILE ;* Go back to WHILE statement

Standalone Example Program 7 – Multi Thread
Task: Program 0 will continuously move the motor between positions 0 and 1000. Simultaneously, program 1 will control the status of program 0 using digital inputs.

PRG 0 ;* Start of Program 0
HSPD=20000 ;* Set high speed to 20000pps
LSPD=500 ;* Set low speed to 500pps
ACC=500 ;* Set acceleration to 500ms
WHILE 1=1 ;* Forever loop
    X0 ;* Move to position 0
    WAITX ;* Wait for the move to complete
    X1000 ;* Move to position 1000
    WAITX ;* Wait for the move to complete
ENDWHILE ;* Go back to WHILE statement
END ;* End Program 0

PRG 1 ;* Start of Program 1
WHILE 1=1 ;* Forever loop
    IF DI1=1 ;* If digital input 1 is triggered
        ABORTX ;* Stop movement
        SR0=0 ;* Stop Program 1
    ELSE
        SR0=1 ;* If digital input 1 is not triggered
        IF SR0=1
            RUN Program 1
        ELSE
            Endif statements
        ENDIF
ENDWHILE ;* Go back to WHILE statement
END ;* End Program 1
### Standalone Example Program 8 – Multi Thread

Task: Program 0 will continuously move the motor between positions 0 and 1000. Simultaneously, program 1 will monitor the communication time-out parameter and triggers digital output 1 if a time-out occurs. Program 1 will also stop all motion, disable program 0 and then re-enable it after a delay of 3 seconds when the error occurs.

#### Program 0

```
PRG 0
    ;* Start of Program 0
    HSPD=1000 ;* Set high speed to 1000 pps
    LSPD=500  ;* Set low speed to 500pps
    ACC=500   ;* Set acceleration to 500ms
    TOC=5000  ;* Set time-out counter alarm to 5 seconds
    EO=1      ;* Enable motor
    WHILE 1=1 ;* Forever loop
        X0    ;* Move to position 0
        WAITX ;* Wait for the move to complete
        X1000 ;* Move to position 1000
        WAITX ;* Wait for the move to complete
    ENDWHILE ;* Go back to WHILE statement
    END      ;* End Program 0
```

#### Program 1

```
PRG 1
    ;* Start of Program 1
    WHILE 1=1 ;* Forever loop
        V1=MSTX&2048 ;* Get bit time-out counter alarm variable
        IF V1 = 2048 ;* If time-out counter alarm is on
            SR0=0 ;* Stop program 0
            ABORTX ;* Abort the motor
            DO=0 ;* Set DO=0
            DELAY=3000 ;* Delay 3 seconds
            SR0=1 ;* Turn program 0 back on
            DO=1 ;* Set DO=1
        ENDIF
    ENDWHILE ;* Go back to WHILE statement
    END      ;* End Program 1
```
Where to Go Next

Continue with the next chapter, *Chapter 8 - Interactive Commands & USB Communications*. Previous sections described how to use the PMX-2EX-SA software to communicate to the DMM-0200. Section 8 describes the basic ASCII commands and USB communications when using some other software such as Matlab or Labview.
Chapter 8
Interactive Commands & USB Communications

About This Chapter

Introduction

This chapter describes the USB communications and ASCII commands in detail to communicate to the DMM-0200. Previous sections explain the communication with the PMX-2EX-SA software. The user does not necessarily need to use the PMX-2EX-SA software to communicate to the DMM-0400. Visual BASIC, Visual C++, Labview, Matlab or other software can communicate to the DMM-200 through the USB connection. The command set described in this section are interactive commands that are sent through the USB port.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB Communication Description</td>
<td>89</td>
</tr>
<tr>
<td>Motion Control Overview and Interactive Command Description</td>
<td>91</td>
</tr>
<tr>
<td>ASCII Programming Language</td>
<td>103</td>
</tr>
<tr>
<td>Error Codes</td>
<td>108</td>
</tr>
<tr>
<td>Where to Go Next</td>
<td>109</td>
</tr>
</tbody>
</table>
USB Communication Description

PMX-2EX-SA USB communication is USB 2.0 compliant.

Communication between the PC and PMX-2EX-SA is done using Windows compatible DLL API function calls as shown below. Windows programming language such as Visual BASIC, Visual C++, LAB View, or any other programming language that can use DLL can be used to communicate with the Performax module.

Typical communication transaction time between PC and PMX-4EX-SA for sending a command from a PC and getting a reply from PMX-2EX-SA using the fnPerformaxComSendRecv API function is in single digit milliseconds. This value will vary with CPU speed of PC and the type of command.

USB Communication API Functions

For USB communication, following DLL API functions are provided.

BOOL fnPerformaxComGetNumDevices(OUT LPDWORD lpNumDevices);
- This function is used to get total number of all types of Performax and Performax USB modules connected to the PC.

BOOL fnPerformaxComGetProductString(IN DWORD dwNumDevices,
                                       OUT LPVOID lpDeviceString,
                                       IN DWORD dwOptions);
- This function is used to get the Performax or Performax product string. This function is used to find out Performax USB module product string and its associated index number. Index number starts from 0.

BOOL fnPerformaxComOpen(IN DWORD dwDeviceNum,
                         OUT HANDLE* pHandle);
- This function is used to open communication with the Performax USB module and to get communication handle. dwDeviceNum starts from 0.

BOOL fnPerformaxComClose(IN HANDLE pHandle);
- This function is used to close communication with the Performax USB module.

BOOL fnPerformaxComSetTimeouts(IN DWORD dwReadTimeout,
                                DWORD dwWriteTimeout);
- This function is used to set the communication read and write timeout. Values are in milliseconds. This must be set for the communication to work. Typical value of 1000 msec is recommended.

BOOL fnPerformaxComSendRecv(IN HANDLE pHandle,
                             IN LPVOID wBuffer,
                             IN DWORD dwNumBytesToWrite,
                             IN DWORD dwNumBytesToRead,
                             OUT LPVOID rBuffer);
- This function is used to send command and get reply. Number of bytes to read and write must be 64 characters.

BOOL fnPerformaxComFlush(IN HANDLE pHandle)
- Flushes the communication buffer on the PC as well as the USB controller. It is recommended to perform this operation right after the communication handle is opened.
**USB Communication Issues**

A common problem that users may have with USB communication is that after sending a command from the PC to the device, the response is not received by the PC until another command is sent. In this case, the data buffers between the PC and the USB device are out of sync. Below are some suggestions to help alleviate this issue.

1) **Buffer Flushing**: If USB communication begins from an unstable state (i.e. your application has closed unexpectedly, it is recommended to first flush the USB buffers of the PC and the USB device. See the following function prototype below:

   ```
   BOOL fnPerformaxComFlush(IN HANDLE pHandle)
   ```

   **Note**: fnPerformaxComFlush is only available in the most recent PerformaxCom.dll which is not registered by the standard USB driver installer. A sample of how to use this function along with this newest DLL is available for download from the following website:


2) **USB Cable**: Another source of USB communication issues may come from the USB cable. Confirm that the USB cable being used has a noise suppression choke. See photo below:

   ![Figure 32 USB Cable](image)

   Figure 32 USB Cable
Motion Control Overview and Interactive Command Description

All the commands described in this section are interactive commands and are not analogous to stand-alone commands. Refer to the “Standalone Language Specification” section for details regarding stand-alone commands.

**Motion Profile**

By default, the PMX-2EX-SA uses trapezoidal velocity profile. See Figure 33.

![Figure 33 Trapezoidal Velocity Profile](image)

S-curve velocity profile can also be achieved by using the SCV[\texttt{axis}] command. See Figure 34.

![Figure 34 S-Curve Velocity Profile](image)

High speed and low speed are in pps (pulses/second). Use HS[\texttt{axis}] and LS[\texttt{axis}] to set/get individual high speed and low speed settings. To set/get the global high speed and low speed values use the HS and LS commands.

Acceleration and deceleration time are in milliseconds and are symmetrical. Use the ACC[\texttt{axis}] command to set/get individual acceleration/deceleration values. To set/get the global acceleration value, use the ACC command.

**Note:** By default, moves by a single axis use global speed settings, unless individual high speed, low speed and acceleration values for that axis are non-zero.
Example: To set the high-speed of the X-axis to 1500 pulses/second, and the Y-axis to 2000 pulses/second, issue the following speed setting commands:

```
HSX=1500  ' set high speed for x-axis only
HSY=2000  ' set high speed for y-axis only
LSX=300   ' other parameters for the axis MUST be set as well for
LSY=300   ' the controller to use the individual speed settings instead
ACCX=100  ' of the global speed settings
ACCY=100
```

It is possible to have unique acceleration and deceleration times. In order to decelerate using the value set in the DEC[axis] or DEC parameter, set EDEC to 1.

The minimum and maximum acceleration values depend on the high speed and low speed settings. Refer to Table A.0 and Figure A.0 in Appendix A for details.

**Pulse Speed**

Current pulse rate can be read using the PSX/PSY command. For units, see Table 21

<table>
<thead>
<tr>
<th>Operation Mode</th>
<th>Speed Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>StepNLoop disabled</td>
<td>Pulse / sec</td>
</tr>
<tr>
<td>ALL interpolated moves</td>
<td>Pulse / sec</td>
</tr>
<tr>
<td>StepNLoop enabled and non-interpolated move</td>
<td>Encoder counts/ sec</td>
</tr>
</tbody>
</table>

Table 21 Pulse Speed

**On-The-Fly Speed Change**

On-the-fly speed change can be achieved with the SSPD[axis] command. In order to use the SSPD[axis] command, s-curve velocity profile must be disabled.

**SSPD Mode**

The correct speed window must be selected in order to use the SSPD command. To select a speed window, use the SSPDM[axis] command. Refer to Appendix A for details.

During on-the-fly speed change operation, you must keep the initial and destination speeds within the speed window.

For non on-the-fly speed change moves, set SSPDM[axis] to 0.
**Motor Status**

Motor status can be read anytime using MSTX/MSTY command. Value of the motor status is replied as an integer with following bit assignment:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Motor in acceleration</td>
</tr>
<tr>
<td>1</td>
<td>Motor in deceleration</td>
</tr>
<tr>
<td>2</td>
<td>Motor running at constant speed</td>
</tr>
<tr>
<td>3</td>
<td>Alarm input status</td>
</tr>
<tr>
<td>4</td>
<td>Plus limit input switch status</td>
</tr>
<tr>
<td>5</td>
<td>Minus limit input switch status</td>
</tr>
<tr>
<td>6</td>
<td>Home input switch status</td>
</tr>
<tr>
<td>7</td>
<td>Plus limit error. This bit is latched when plus limit is hit during motion. This error must be cleared using the CLR/CLRX/CLRY command before issuing any subsequent move commands.</td>
</tr>
<tr>
<td>8</td>
<td>Minus limit error. This bit is latched when minus limit is hit during motion. This error must be cleared using the CLR/CLRX/CLRY command before issuing any subsequent move commands.</td>
</tr>
<tr>
<td>9</td>
<td>Z Index Channel status</td>
</tr>
<tr>
<td>10</td>
<td>Joystick Control On status</td>
</tr>
<tr>
<td>11</td>
<td>TOC time-out status</td>
</tr>
</tbody>
</table>

**Table 22 Motor Status**

**Individual/Linear Interpolation Moves**

For individual axis control use X and Y command followed by the target position value. **Individual/Linear Move Examples:**

[X1000]: MoveX-axistoposition1000.

[Y1000]: MoveY-axistoposition1000.

For linear interpolation axis control use the [XTarget]:[YTarget] to perform coordinated movement to the specified target positions. **Linear Interpolation Move Examples:**

[X1000:10000]: MoveX-axistoposition1000,Y-axistoposition1000usinglinearinterpolation.

[X10000:-10000]: MoveX-axistoposition10000,Y-axistoposition-10000usinglinearinterpolation.

Individual/Linear Interpolation moves can be performed in two modes: incremental mode. To set modes, use the INC and ABS commands respectively. **Move Mode Examples:**

[X1000]–INC mode: The motor will move by 1000 from the current position.

[X1000]–ABS mode: The motor will move to the absolute position 1000.
On-The-Fly Target Position Change

On-the-fly target position change can be achieved using the \texttt{T[axis][value]} command. While the motor is moving, \texttt{T[axis][value]} will change the final destination of the motor. If the motor has already passed the new target position, it will reverse direction once the target position change command is issued.

\textbf{Note:} If a \texttt{T} command is sent while the controller is not performing a target move, the command is not processed. Instead, an error response is returned.

Homing

Home search sequence involves moving the motor towards the home or limit switches and then stopping when the relevant input is detected. The PMX-2EX-SA has five different homing routines.

\textbf{Home Input Only (High Speed Only)}

Use the \texttt{H[axis]+/H[axis]-} command. (use the H+/H- command for both axes). Figure 35 shows the homing routine.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure35.png}
\caption{Homing - Home Input Only (High Speed)}
\end{figure}

\begin{enumerate}
\item Starts the motor from low speed and accelerates to high speed.
\item As soon as the home input is triggered, the position counter is reset to zero and the motor begins to decelerate to low speed. As the motor decelerates, the position counter keeps counting with reference to the zero position.
\item Once low speed is reached, the motor stops. The position is non-zero.
\end{enumerate}

\textbf{Home Input Only (High Speed and Low Speed)}

Use the \texttt{HL[axis]+/HL[axis]-} command (use the HL+/HL- for both axes). Figure 36 shows the homing routine.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure36.png}
\caption{Homing - Home Input Only (High Speed and Low Speed)}
\end{figure}
A. Starts the motor from low speed and accelerates to high speed.
B. As soon as the home input is triggered, the position counter is reset to zero and the motor decelerates to low speed.
C. Once low speed is reached, the motor reverses direction to search for the home switch.
D. Once the home switch is reached, it will continue past the home switch by the amount defined by the home correction amount (HCA) at high speed.
E. The motor is now past the home input by the amount defined by the home correction amount (HCA). The motor now moves back towards the home switch at low speed.
F. The home input is triggered again, the position counter is reset to zero and the motor stops immediately.

**Limit Only**

Use the L[axis]+/L[axis]- command (use the L+/L- command for both axes). Figure 6.4 shows the homing routine. Figure 37 shows the homing routine.

---

**Figure 37 Homing - Limit Only**

A. Issuing a limit home command starts the motor from low speed and accelerates to high speed.
B. The corresponding limit is triggered and the motor stops immediately.
C. The motor reverses direction by the amount defined by the limit correction amount (LCA) at high speed.
D. The zero position is reached.

---

**Home and Z-index**

Use the HZ[axis]+/HZ[axis]- command (use the HZ+/HZ- command for both axes). Figure 38 shows the homing routine.
A. Issuing the command starts the motor from low speed and accelerates to high speed.
B. As soon as the home input is triggered, the motor decelerates to low speed.
C. Once low speed is reached, the motor begins to search for the z-index pulse.
D. Once the z-index pulse is found, the motor stops and the position is set to zero.

**Z-index only**
Use the Z{axis}+/Z{axis}- command (use the Z+/Z- command for both axes). Figure 39 shows the homing routine.

A. Issuing the command starts the motor at low speed.
B. Once the z-index pulse is found, the motor stops and the position is set to zero.

**Jogging**
Jogging is available for continuous speed operation. Use J{axis}+/J{axis}- command. To have both motors jog synchronously use the J+/J- command.

**Stopping Motor**
When the motor is moving, the ABORT{axis} command will immediately stop an individual axis. Use the ABORT command to immediately stop ALL axes.

To employ deceleration on a stop, use the STOP{axis} to stop an individual axis. Use the STOP command to stop ALL axes.

**Note:** If an interpolation operation is in process when a STOP{axis} or ABORT{axis} command is entered, all axes involved in the interpolation operation will stop.
**Motor Position**

Motor positions can be read using the PX/PY command which returns the pulse position of the specified axis.
Encoder positions can be read using EX/EY command which returns the encoder position of the specified axis.

To manually set/get the pulse position of an individual axis, use the PX/PY command.
Note that setting the pulse position is not allowed if StepNLoop is enabled.

To manually set/get the encoder position of an individual axis, use the EX/EY command.

**Polarity**

The polarity settings of the PMX-2EX-SA can also be read or set at any time using the POLX/POLY commands. The following is the bit representation of the polarity:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Pulse</td>
</tr>
<tr>
<td>1</td>
<td>Direction</td>
</tr>
<tr>
<td>2</td>
<td>Not Used</td>
</tr>
<tr>
<td>3</td>
<td>Not Used</td>
</tr>
<tr>
<td>4</td>
<td>Not Used</td>
</tr>
<tr>
<td>5</td>
<td>Home</td>
</tr>
<tr>
<td>6</td>
<td>+/- Limit</td>
</tr>
<tr>
<td>7</td>
<td>Z-Index</td>
</tr>
<tr>
<td>8</td>
<td>Encoder decoding</td>
</tr>
<tr>
<td>9</td>
<td>00</td>
</tr>
<tr>
<td>10</td>
<td>01</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>Digital Input</td>
</tr>
<tr>
<td>13</td>
<td>Digital Output</td>
</tr>
<tr>
<td>14</td>
<td>Enable Output</td>
</tr>
<tr>
<td>15</td>
<td>Jump to Line 0 on</td>
</tr>
</tbody>
</table>

Table 23 Polarity

†Used for error handling within standalone operation. If this bit is on, the line that is executed after SUB31 is called will be line 0. Otherwise, it will be the line that caused the error.

**Limits**

If positive limit switch is triggered while moving in positive direction, the motor will immediately stop and the motor status bit for positive limit error is set. The same is for the negative limit while moving in the negative direction. To read the limit switch input status, use the MSTX/MSTY command.

Once the limit or alarm error is set, use the CLR[axis] command to clear the error.

The limit error states can be ignored by setting IERR=1. In this case, the motor will still stop when the appropriate switch is triggered; however, it will not enter an error state.
Digital Inputs/Outputs and Enable Outputs

PMX-2EX-SA module comes with 8 digital inputs and 8 digital outputs and 4 enable outputs.

Inputs
Read digital input status using the DI command.

Digital input values can also be referenced one bit at a time by the DI[1-8] commands. Note that the indexes are 1-based for the bit references (i.e. DI1 refers to bit 0, not bit 1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Bit-Wise Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Digital Input 1</td>
<td>DI1</td>
</tr>
<tr>
<td>1</td>
<td>Digital Input 2</td>
<td>DI2</td>
</tr>
<tr>
<td>2</td>
<td>Digital Input 3</td>
<td>DI3</td>
</tr>
<tr>
<td>3</td>
<td>Digital Input 4</td>
<td>DI4</td>
</tr>
<tr>
<td>4</td>
<td>Digital Input 5</td>
<td>DI5</td>
</tr>
<tr>
<td>5</td>
<td>Digital Input 6</td>
<td>DI6</td>
</tr>
<tr>
<td>6</td>
<td>Digital Input 7</td>
<td>DI7</td>
</tr>
<tr>
<td>7</td>
<td>Digital Input 8</td>
<td>DI8</td>
</tr>
</tbody>
</table>

Table 24 Digital Inputs

Digital Outputs
The digital output status can be controlled using the DO command. DO value must be within the range of 0-255.

Digital output values can also be referenced one bit at a time by the DO[1-8J commands. Note that the indexes are 1-based for the bit references (i.e. DO1 refers to bit 0, not bit 1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Bit-Wise Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Digital Output 1</td>
<td>DO1</td>
</tr>
<tr>
<td>1</td>
<td>Digital Output 2</td>
<td>DO2</td>
</tr>
<tr>
<td>2</td>
<td>Digital Output 3</td>
<td>DO3</td>
</tr>
<tr>
<td>3</td>
<td>Digital Output 4</td>
<td>DO4</td>
</tr>
<tr>
<td>4</td>
<td>Digital Output 5</td>
<td>DO5</td>
</tr>
<tr>
<td>5</td>
<td>Digital Output 6</td>
<td>DO6</td>
</tr>
<tr>
<td>6</td>
<td>Digital Output 7</td>
<td>DO7</td>
</tr>
<tr>
<td>7</td>
<td>Digital Output 8</td>
<td>DO8</td>
</tr>
</tbody>
</table>

Table 25 Digital Outputs

The initial state of the digital outputs can be defined by setting the DOBOOT register to the desired initial digital output value. The value is stored to flash memory once the STORE command is issued.
Enable Outputs
The enable output status can be controlled using the EO command. EO value must be within the range of 0-3.

Enable output values can also be referenced one bit at a time by the EO[1-2] commands. Note that the indexes are 1-based for the bit references (i.e. EO1 refers to bit 0, not bit 1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Bit-Wise Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Enable Output 1 [X-axis]</td>
<td>EO1</td>
</tr>
<tr>
<td>1</td>
<td>Enable Output 2 [Y-axis]</td>
<td>EO2</td>
</tr>
</tbody>
</table>

Table 26 Enable Outputs

The initial state of the enable outputs can be defined by setting the EOBOOT register to the desired initial enable output value. The value is stored to flash memory once the STORE command is issued.

Analog Inputs
2 x 10-bit analog inputs are available on PMX-2EX-SA. Use AI[1-2] command to read the analog input value. Range is from 0-5000 mV.

Joystick Control
Joystick control is available on PMX-2EX-SA. When this mode is enabled, the pulse speed and direction output can be controlled by corresponding analog input. See the axis to analog input relationship in the table below:

<table>
<thead>
<tr>
<th>Axis</th>
<th>Analog Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>AI1</td>
</tr>
<tr>
<td>Y</td>
<td>AI2</td>
</tr>
</tbody>
</table>

Table 27 Joystick Control
Maximum joystick speed is set using the JV1 and JV2 variables.
Maximum speed change (delta) is set using the JV3 and JV4 variables.
Tolerance of the zero joystick position, use JV5 and JV6 variables.

Joystick control also has soft limit controls. Limits are broken into: negative outer limit, negative inner limit, positive inner limit and positive outer limit.

When moving in positive direction, as soon as the positive inner limit is crossed, the speed is reduced. If the position reaches the positive outer limit, the joystick speed is set to zero. Same goes for the negative direction and negative limits.

The behavior of the limits of the joystick control is explained by the following:
Summary of Joystick Control Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JV1</td>
<td>X-axis maximum joystick speed at 5000 mV and 0 mV</td>
</tr>
<tr>
<td>JV2</td>
<td>Y-axis maximum joystick speed at 5000 mV and 0 mV</td>
</tr>
<tr>
<td>JV3</td>
<td>X-axis maximum speed change</td>
</tr>
<tr>
<td>JV4</td>
<td>Y-axis maximum speed change</td>
</tr>
<tr>
<td>JV5</td>
<td>X-axis zero tolerance range for analog input</td>
</tr>
<tr>
<td>JV6</td>
<td>Y-axis zero tolerance range for analog input</td>
</tr>
<tr>
<td>JL1</td>
<td>X-axis negative outer limit</td>
</tr>
<tr>
<td>JL2</td>
<td>X-axis negative inner limit</td>
</tr>
<tr>
<td>JL3</td>
<td>X-axis positive inner limit</td>
</tr>
<tr>
<td>JL4</td>
<td>X-axis positive outer limit</td>
</tr>
<tr>
<td>JL5</td>
<td>Y-axis negative outer limit</td>
</tr>
<tr>
<td>JL6</td>
<td>Y-axis negative inner limit</td>
</tr>
<tr>
<td>JL7</td>
<td>Y-axis positive inner limit</td>
</tr>
<tr>
<td>JL8</td>
<td>Y-axis positive outer limit</td>
</tr>
</tbody>
</table>

Table 28 Summary of Joystick Control Parameters

To enable/disable joystick control for an axis, use the JE command. Joystick enable parameter is a 2 bit value. For example, joystick enable value of 3 means joystick feature is enabled on both axes.

**Note:** If joystick control is enabled, StepNLoop is automatically disabled.

**StepNLoop Closed Loop Control**

PMX-2EX-SA features a closed-loop position verification algorithm called StepNLoop (SNL). The algorithm requires the use of an incremental encoder.

SNL performs the following operations:

1) **Position Verification:** At the end of any targeted move, SNL will perform a correction if the current error is greater than the tolerance value.

2) **Delta Monitoring:** The delta value is the difference between the actual and the target position. When delta exceeds the error range value, the motor is stopped and the SNL Status goes into an error state. Delta monitoring is performed during moves – including homing and jogging. To read the delta value, use the DX command.

See Control for a list of the SNL control parameters.

<table>
<thead>
<tr>
<th>SNL Parameter</th>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>StepNLoop Ratio</td>
<td>†Ratio between motor pulses and encoder counts. This ratio will depend on</td>
<td>SLR[axis]</td>
</tr>
<tr>
<td></td>
<td>the motor type, micro-stepping, encoder resolution and decoding multiplier.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value must be in the range [0.001, 999.999].</td>
<td></td>
</tr>
<tr>
<td>Tolerance</td>
<td>Maximum error between target and actual position that is considered “In</td>
<td>SLT[axis]</td>
</tr>
<tr>
<td></td>
<td>Position”. In this case, no correction is performed. Units are in encoder</td>
<td></td>
</tr>
<tr>
<td></td>
<td>counts.</td>
<td></td>
</tr>
</tbody>
</table>
Error Range | Maximum error between target and actual position that is not considered a serious error. If the error exceeds this value, the motor will stop immediately and go into an error state. | SLE[axis]
Correction Attempt | Maximum number of correction tries that the controller will attempt before stopping and going into an error state. | SLA [axis]

Table 29 StepNLoop Closed Loop Control

†A convenient way to find the StepNLoop ratio is to set EX=0, PX=0 and move the motor + 1000 pulses. The ratio can be calculated by dividing 1000 by the resulting EX value. Note that the value must be positive. If it is not, then the direction polarity must be adjusted. This test can be performed on all axes that require StepNLoop.

To enable/disable the SNL feature use the SL[axis] command. To read the SNL status, use SLS[axis] command to read the status.

See Table 30 StepNLoop Return Values for a list of the SLS[axis] return values.

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Idle</td>
</tr>
<tr>
<td>1</td>
<td>Moving</td>
</tr>
<tr>
<td>2</td>
<td>Correcting</td>
</tr>
<tr>
<td>3</td>
<td>Stopping</td>
</tr>
<tr>
<td>4</td>
<td>Aborting</td>
</tr>
<tr>
<td>5</td>
<td>Jogging</td>
</tr>
<tr>
<td>6</td>
<td>Homing</td>
</tr>
<tr>
<td>7</td>
<td>Z-Homing</td>
</tr>
<tr>
<td>8</td>
<td>Correction range error. To clear this error, use CLRS or CLR command.</td>
</tr>
<tr>
<td>9</td>
<td>Correction attempt error. To clear this error, use CLRS or CLR command.</td>
</tr>
<tr>
<td>10</td>
<td>Stall Error. DX value has exceeded the correction range value. To clear this error, use CLRS or CLR command.</td>
</tr>
<tr>
<td>11</td>
<td>Limit Error</td>
</tr>
<tr>
<td>12</td>
<td>N/A (i.e. SNL is not enabled)</td>
</tr>
<tr>
<td>13</td>
<td>Limit homing</td>
</tr>
</tbody>
</table>

Table 30 StepNLoop Return Values
See Table 31 for SNL behavior within different scenarios.

<table>
<thead>
<tr>
<th>Condition</th>
<th>SNL behavior (motor is moving)</th>
<th>SNL behavior (motor is idle)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 &lt;= SLT</td>
<td>Continue to monitor the DX[axis]</td>
<td>In Position. No correction is performed.</td>
</tr>
<tr>
<td>5 &gt; SLT AND 5 &lt; SLE</td>
<td>Continue to monitor the DX[axis]</td>
<td>Out of Position. A correction is performed.</td>
</tr>
<tr>
<td>Correction Attempt&gt; SLA</td>
<td>NA</td>
<td>Max Attempt Error. Motor stops and signals and error.</td>
</tr>
</tbody>
</table>

Table 31 StepNLoop Conditions

Key
[5]: Error between the target position and actual position
SLT: Tolerance range
SLE: Error range
SLA: Max correction attempt

Notes:
Once SNL is enabled, position move commands are in term of encoder position. For example, X1000 means to move the motor to encoder 1000 position. This applies to individual as well as interpolated moves.

Once SNL is enabled, the speed is in encoder speed. For example HSPD= 1000 when SNL is enabled means that the target high speed is 1000 encoder counts per second. This only applies to individual axis moves.

Linear Interpolation w/ StepNLoop: If StepNLoop is used during a linear interpolation move, StepNLoop must be enabled for all axes being moved. Also note that unlike the individual axis moves, the speed during a linear interpolation is calculated as pulse/sec, NOT encoder counts/sec.

Device Number

PMX-2EX-SA module provides the user with the ability to modify the unique device number. In order to make these changes, first store the desired number using the DN command. Note that this value must be within the range [2EX00,2EX99].

To write the values to the device’s flash memory, use the STORE command. After a complete power cycle, the new device number will be written to memory. Note that before a power cycle is completed, the settings will not take affect.

By default: Device name is set to: 2EX00
Invalid command is returned with ?(Error Message). Always check for proper reply when command is sent. Like the commands, all responses are in ASCII form.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>Immediately stops all the motors if in motion. Abort turns off the buffered move.</td>
<td>OK</td>
</tr>
<tr>
<td>ABORTX</td>
<td>Immediately stops individual motor if in motion. Abort turns off the buffered move.</td>
<td>OK</td>
</tr>
<tr>
<td>ABORTY</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>ABS</td>
<td>Turns on absolute move mode</td>
<td>OK</td>
</tr>
<tr>
<td>ACC</td>
<td>Returns current global acceleration value in milliseconds.</td>
<td>32-bit number</td>
</tr>
<tr>
<td>ACC={value}</td>
<td>Sets global acceleration value in milliseconds.</td>
<td>OK</td>
</tr>
<tr>
<td>ACC X</td>
<td>Returns current individual acceleration value in milliseconds.</td>
<td>32-bit number</td>
</tr>
<tr>
<td>ACCX={value}</td>
<td>ACCY={value}</td>
<td>Sets individual acceleration value in milliseconds.</td>
</tr>
<tr>
<td>AI[1-2]</td>
<td>Returns Analog Input in millivolt.</td>
<td>[0-5000]</td>
</tr>
<tr>
<td>CLRX</td>
<td>Clears motor limit or alarm status bit. Also clears a StepNLoop errors</td>
<td>OK</td>
</tr>
<tr>
<td>CLRY</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>DEC</td>
<td>Returns the current global deceleration value in milliseconds</td>
<td>32-bit number</td>
</tr>
<tr>
<td>DEC={value}</td>
<td>Sets the global deceleration value in milliseconds</td>
<td>OK</td>
</tr>
<tr>
<td>DECX</td>
<td>Returns the current individual deceleration value in milliseconds</td>
<td>32-bit number</td>
</tr>
<tr>
<td>DECY</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>DECX={value}</td>
<td>DECY={value}</td>
<td>Sets the individual deceleration value in milliseconds</td>
</tr>
<tr>
<td>DI</td>
<td>Returns 8 bits of general purpose digital input.</td>
<td>[0-255]</td>
</tr>
<tr>
<td>DI[1-8]</td>
<td>Returns bit status of general purpose digital input.</td>
<td>[0,1]</td>
</tr>
<tr>
<td>DO</td>
<td>Returns 8 bits of general purpose digital output value.</td>
<td>[0-255]</td>
</tr>
<tr>
<td>DO={value}</td>
<td>Sets 8 bits of general purpose digital output.</td>
<td>OK</td>
</tr>
<tr>
<td>DO[1-8]</td>
<td>Returns bit of general purpose digital output value.</td>
<td>[0,1]</td>
</tr>
<tr>
<td>DO[1-8]={value}</td>
<td>Sets bit of general purpose digital output.</td>
<td>OK</td>
</tr>
<tr>
<td>DOBOOT</td>
<td>Get DO boot-up state</td>
<td>See Table 25</td>
</tr>
<tr>
<td>DOBOOT={value}</td>
<td>Set DO boot-up state</td>
<td>OK</td>
</tr>
<tr>
<td>DN</td>
<td>Return device number</td>
<td>2EXXX</td>
</tr>
<tr>
<td>DN={value}</td>
<td>Set device number</td>
<td>OK</td>
</tr>
<tr>
<td>DXX</td>
<td>Get StepNLoop delta value of axis</td>
<td>32-bit number</td>
</tr>
<tr>
<td>DXY</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>EDEC</td>
<td>Returns the enable deceleration status</td>
<td>[0,1]</td>
</tr>
<tr>
<td>EDEC={0 or 1}</td>
<td>Sets the enabled deceleration status</td>
<td>OK</td>
</tr>
<tr>
<td>EO</td>
<td>Returns 2 bits of enable output value.</td>
<td>[0-3]</td>
</tr>
<tr>
<td>EO={value}</td>
<td>Sets 2 bits of enable outputs.</td>
<td>OK</td>
</tr>
<tr>
<td>EO[1-2]</td>
<td>Returns bit of enable output value.</td>
<td>[0,1]</td>
</tr>
<tr>
<td>EO[1-2]={value}</td>
<td>Set bit of enable outputs.</td>
<td>OK</td>
</tr>
<tr>
<td>EOBOOT</td>
<td>Get EO boot-up state</td>
<td>See Table 26</td>
</tr>
<tr>
<td>EOBOOT={value}</td>
<td>Set EO boot-up state</td>
<td>OK</td>
</tr>
<tr>
<td>EX</td>
<td>Returns Current Encoder Position</td>
<td>28 bit signed position</td>
</tr>
<tr>
<td>EY</td>
<td>Set encoder value of axis</td>
<td>OK</td>
</tr>
</tbody>
</table>

Table 32 ASCII Commands
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS[SubNumber]</td>
<td>Call a defined subroutine</td>
<td>OK</td>
</tr>
<tr>
<td>HCA</td>
<td>Returns the global home correction amount</td>
<td>28-bit number</td>
</tr>
<tr>
<td>HCA=[Value]</td>
<td>Sets the global home correction amount.</td>
<td>OK</td>
</tr>
<tr>
<td>HCA X HCAY</td>
<td>Returns the home correction amount for the specified axis.</td>
<td>28-bit number</td>
</tr>
<tr>
<td>HCA X HCAY=[Value]</td>
<td>Sets the home correction amount for the specified axis.</td>
<td>OK</td>
</tr>
<tr>
<td>HSPD</td>
<td>Returns the global high speed setting.</td>
<td>High Speed</td>
</tr>
<tr>
<td>HSPD=[Value]</td>
<td>Set the global high speed setting</td>
<td>OK</td>
</tr>
<tr>
<td>HSPD X HSPDY</td>
<td>Returns high speed setting for the X-axis and Y-axis</td>
<td>High Speed</td>
</tr>
<tr>
<td>HSPD X HSPDY=[Value]</td>
<td>Sets high speed setting for the X-axis and Y-axis</td>
<td>OK</td>
</tr>
<tr>
<td>H+</td>
<td>Homes both X and Y axis at high speed in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>H-</td>
<td>Homes both X and Y axis at high speed in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>HX+ HY+</td>
<td>Homes X/Y axis at high speed in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>HX- HY-</td>
<td>Homes X/Y axis at high speed in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>HL+</td>
<td>Homes both X and Y axis at high and low speed in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>HL-</td>
<td>Homes both X and Y axis at high and low speed in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>HLX+ HLY+</td>
<td>Homes X/Y axis at high and low speed in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>HLX- HLY-</td>
<td>Homes X/Y axis at high and low speed in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>[X Target]: [Y Target]</td>
<td>Perform linear interpolated motion</td>
<td>OK</td>
</tr>
<tr>
<td>ID</td>
<td>Returns Controller ID</td>
<td>Performax-2EX-</td>
</tr>
<tr>
<td>IERR</td>
<td>Get the ignore limit error status</td>
<td>[0-1]</td>
</tr>
<tr>
<td>IERR=[0 or 1]</td>
<td>Set the ignore limit error status</td>
<td>OK</td>
</tr>
<tr>
<td>INC</td>
<td>Enable incremental move mode</td>
<td>OK</td>
</tr>
<tr>
<td>JF</td>
<td>Turns off Joystick Control</td>
<td>OK</td>
</tr>
<tr>
<td>JL[1-8]</td>
<td>Return Joystick Control Limits</td>
<td>See Table 28</td>
</tr>
<tr>
<td>JL[1-8]=[value]</td>
<td>Sets Joystick Control Limits.</td>
<td>OK</td>
</tr>
<tr>
<td>See Table 28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JO</td>
<td>Turns on Joystick Control</td>
<td>OK</td>
</tr>
<tr>
<td>JS</td>
<td>Get the Joystick status</td>
<td>[0,1]</td>
</tr>
<tr>
<td>JV[1 to 6]</td>
<td>Returns Joy Stick Control Parameters</td>
<td>See Table 28</td>
</tr>
<tr>
<td>JV[1 to 6]=[Value]</td>
<td>Sets Joystick Control Parameters. See Table 6.7</td>
<td>OK</td>
</tr>
<tr>
<td>J+</td>
<td>Jogs both X/Y Motor Positive</td>
<td>OK</td>
</tr>
<tr>
<td>J-</td>
<td>Jogs both X/Y Motor Negative</td>
<td>OK</td>
</tr>
<tr>
<td>JX+ JY+</td>
<td>Jogs Motor Positive</td>
<td>OK</td>
</tr>
<tr>
<td>JX- JY-</td>
<td>Jogs Motor Negative</td>
<td>OK</td>
</tr>
<tr>
<td>L+</td>
<td>Homes both X and Y axis to the positive limit input in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>L-</td>
<td>Home both X and Y axis to the negative limit input in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>LX+ LY+</td>
<td>Homes X/Y axis to the positive limit input in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>LX- LY-</td>
<td>Homes X/Y axis to the negative limit input in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>LCA</td>
<td>Returns the global limit correction amount</td>
<td>28-bit number</td>
</tr>
<tr>
<td>LCA=[Value]</td>
<td>Sets the global limit correction amount</td>
<td>OK</td>
</tr>
<tr>
<td>LCAX LCAY</td>
<td>Returns the specified limit correction amount</td>
<td>28-bit number</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Return</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>LCAX=[value]</td>
<td>Sets the specified limit correction amount</td>
<td>OK</td>
</tr>
<tr>
<td>LCAY=[value]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSPD</td>
<td>Returns the global low speed setting</td>
<td></td>
</tr>
<tr>
<td>LSPD=[Value]</td>
<td>Sets the global low speed setting.</td>
<td></td>
</tr>
<tr>
<td>LSPDX</td>
<td>Returns low speed setting for the X-axis</td>
<td></td>
</tr>
<tr>
<td>LSPDY</td>
<td>and Y-axis</td>
<td></td>
</tr>
<tr>
<td>LSPDX=[Value]</td>
<td>Sets low speed setting for the X-axis</td>
<td></td>
</tr>
<tr>
<td>LSPDY=[Value]</td>
<td>and Y-axis</td>
<td></td>
</tr>
<tr>
<td>MM</td>
<td>Returns the move mode that the controller is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>currently in.</td>
<td></td>
</tr>
<tr>
<td>MSTX</td>
<td>Returns motor status</td>
<td></td>
</tr>
<tr>
<td>MSTY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLX</td>
<td>Returns polarity</td>
<td></td>
</tr>
<tr>
<td>POLY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLX=[Value]</td>
<td>Sets polarity.</td>
<td></td>
</tr>
<tr>
<td>POLY=[Value]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSX</td>
<td>Returns current pulse speed</td>
<td></td>
</tr>
<tr>
<td>PSY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PX</td>
<td>Returns current pulse position</td>
<td></td>
</tr>
<tr>
<td>PY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PX=[Value]</td>
<td>Sets current pulse position</td>
<td></td>
</tr>
<tr>
<td>PY=[Value]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RZ</td>
<td>Returns the return to zero enable status</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Used during homing operations</td>
<td>[0,1]</td>
</tr>
<tr>
<td>RZ=[0,1]</td>
<td>Sets the return to zero enable status</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Used during homing operations</td>
<td>OK</td>
</tr>
<tr>
<td>SASTAT[0,1]</td>
<td>Get standalone program status</td>
<td>[0-4]</td>
</tr>
<tr>
<td></td>
<td>0 – Stopped</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 – Running</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 – Paused</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 – In Error</td>
<td></td>
</tr>
<tr>
<td>SA[LineNumber]</td>
<td>Get standalone line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LineNumber: [0,1275]</td>
<td></td>
</tr>
<tr>
<td>SA[LineNumber]=[Value]</td>
<td>Set standalone line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LineNumber: [0,1275]</td>
<td>OK</td>
</tr>
<tr>
<td>SCVX</td>
<td>Get s-curve on/off status</td>
<td></td>
</tr>
<tr>
<td>SCVY</td>
<td></td>
<td>[0,1]</td>
</tr>
<tr>
<td>SCVX=[0 or 1]</td>
<td>Set s-curve on/off status</td>
<td></td>
</tr>
<tr>
<td>SCVY=[0 or 1]</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>SLAX</td>
<td>Returns maximum number of StepNLoop control</td>
<td></td>
</tr>
<tr>
<td>SLAY</td>
<td>attempt</td>
<td>32-bit number</td>
</tr>
<tr>
<td>SLAX=[value]</td>
<td>Sets maximum number of StepNLoop control</td>
<td></td>
</tr>
<tr>
<td>SLAY=[value]</td>
<td>attempt</td>
<td>OK</td>
</tr>
<tr>
<td>SLEX</td>
<td>Returns StepNLoop correction value</td>
<td></td>
</tr>
<tr>
<td>SLEY</td>
<td></td>
<td>32-bit number</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Return</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>SLEX=[value] SLEY=[value]</td>
<td>Sets StepNLoop correction value.</td>
<td>OK</td>
</tr>
<tr>
<td>SLRX SLRY</td>
<td>Returns StepNLoop ratio value</td>
<td>32-bit number</td>
</tr>
<tr>
<td>SRLX SRLY</td>
<td>Returns current status of StepNLoop control</td>
<td>See Table 30</td>
</tr>
<tr>
<td>SLTX=[value]</td>
<td>Sets StepNLoop tolerance value.</td>
<td>OK</td>
</tr>
<tr>
<td>SLX=[0 or 1] SLY=[0 or 1]</td>
<td>Enable or disable StepNLoop Control</td>
<td>OK</td>
</tr>
<tr>
<td>SLX SLY</td>
<td>Returns StepNLoop enable status</td>
<td>[0,1]</td>
</tr>
<tr>
<td>SLOAD</td>
<td>Returns RunOnBoot parameter</td>
<td>See Table 19</td>
</tr>
<tr>
<td>SLOAD=[0-3]</td>
<td>Set RunOnBoot parameter</td>
<td>Table 19</td>
</tr>
<tr>
<td>SPC[0,1]</td>
<td>Get program counter for standalone program</td>
<td>Standalone [0-1275]</td>
</tr>
<tr>
<td>SR[0,1]=[Value]</td>
<td>Control standalone program; 0 – Stop standalone program</td>
<td>OK</td>
</tr>
<tr>
<td></td>
<td>1 – Run standalone program</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 – Pause standalone program</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 – Continue standalone program</td>
<td></td>
</tr>
<tr>
<td>SSPDX=[Value] SSPDY=[Value]</td>
<td>Set speed of X/Y axis on-the-fly to [Value]</td>
<td>OK</td>
</tr>
<tr>
<td>SSPDMX SSPDXY</td>
<td>Get on-the-fly speed change mode for each axis</td>
<td>[0-9]</td>
</tr>
<tr>
<td>SSPDMX=[value] SSPDXY=[value]</td>
<td>Set on-the-fly speed change mode for each axis.</td>
<td>OK</td>
</tr>
<tr>
<td>STOP</td>
<td>Stops both X/Y motor with deceleration</td>
<td>OK</td>
</tr>
<tr>
<td>STOP X</td>
<td>Stops the motor with Deceleration</td>
<td>OK</td>
</tr>
<tr>
<td>STORE</td>
<td>Store device settings to Flash. See Table 6.12</td>
<td>OK</td>
</tr>
<tr>
<td>TX[value] TY[value]</td>
<td>Perform on-the-fly target position change for the specified axis</td>
<td>OK</td>
</tr>
<tr>
<td>TOC</td>
<td>Returns the time counter (ms)</td>
<td>32-bit number</td>
</tr>
<tr>
<td>TOC=[value]</td>
<td>Sets the time-out counter (ms)</td>
<td>OK</td>
</tr>
<tr>
<td>V[0-63]</td>
<td>Returns value of indicated general purpose variable register</td>
<td>32-bit number</td>
</tr>
<tr>
<td>V[0-63]=[Value]</td>
<td>Sets value to general purpose variable register</td>
<td>OK</td>
</tr>
<tr>
<td>VER</td>
<td>Returns Version</td>
<td>VXXX</td>
</tr>
<tr>
<td>X[Value]</td>
<td>Individual move command. If in ABS mode, move to position value. If in INC mode, increase position by value.</td>
<td>OK</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Return</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>ZH-</td>
<td>Homes both X and Y axis using the home and Z-index input in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>ZHX+</td>
<td>Homes X/Y axis using the home and Z-index input in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>ZHY+</td>
<td>Homes X/Y axis using the home and Z-index input in the positive direction</td>
<td>OK</td>
</tr>
<tr>
<td>ZHX-</td>
<td>Homes X/Y axis using the home and Z-index input in the negative direction</td>
<td>OK</td>
</tr>
<tr>
<td>ZHY-</td>
<td>Homes X/Y axis using the home and Z-index input in the negative direction</td>
<td>OK</td>
</tr>
</tbody>
</table>

### Error Codes

If an ASCII command cannot be processed by the PMX-2EX-SA, the controller will reply with an error code. See below for possible error responses:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?[Command]</td>
<td>The ASCII command is not understood by the PMX-2EX-SA</td>
</tr>
<tr>
<td>?ABS/INC is not in operation</td>
<td>T[] command is invalid because a target position move is not in operation</td>
</tr>
<tr>
<td>?Clear SNL Error</td>
<td>A move command has been issued while the axis is in StepNLoop error</td>
</tr>
<tr>
<td>?CommandOn</td>
<td>On-the-fly speed change attempted during an interpolated move</td>
</tr>
<tr>
<td>?Index out of Range</td>
<td>The index for the command sent to the controller is not valid.</td>
</tr>
<tr>
<td>?Invalid Answer</td>
<td>Invalid parameter input</td>
</tr>
<tr>
<td>?Low speed out of range</td>
<td>Low speed parameter is out of range</td>
</tr>
<tr>
<td>?Moving</td>
<td>A move or position change command is sent while the PMX-2EX-SA is outputting pulses.</td>
</tr>
<tr>
<td>?S-curve on</td>
<td>Cannot perform SSPD move because s-curve is enabled</td>
</tr>
<tr>
<td>?Speed out of range</td>
<td>SSPD move parameter is out of the range of the SSPDM speed window.</td>
</tr>
<tr>
<td>?SSPD Mode not Initialized</td>
<td>An attempt to perform an on-the-fly speed change without setting the SSPDM register has been made.</td>
</tr>
<tr>
<td>?Sub not Initialized</td>
<td>Call to a subroutine using the GS command is not valid because the specified subroutine has not been defined.</td>
</tr>
</tbody>
</table>

**Table 33 Error Codes**
Where to Go Next

The user should now have all the information required to run the DMM-0200. If any further assistance is required, please contact the service department at Dover Motion.
Appendix

About This Chapter

Introduction

This appendix contains supporting information for the DMM-0200.

Topics

This chapter covers the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed Settings</td>
<td>111</td>
</tr>
<tr>
<td>Acceleration/Deceleration Range</td>
<td>111</td>
</tr>
<tr>
<td>Acceleration/Deceleration Range – Positional Move</td>
<td>112</td>
</tr>
<tr>
<td>Review/Revision History</td>
<td>113</td>
</tr>
</tbody>
</table>
Speed Settings

Table 34 Speed Settings

<table>
<thead>
<tr>
<th>HSPD value [PPS] †</th>
<th>Speed Window [SSPDM]</th>
<th>Min. LSPD value</th>
<th>Min. ACC [ms]</th>
<th>δ</th>
<th>Max ACC setting [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 16K</td>
<td>0.1</td>
<td>1</td>
<td>1</td>
<td>300</td>
<td>((HSPD – LSPD) / δ) × 1000</td>
</tr>
<tr>
<td>16k - 32K</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>775</td>
<td>1,900</td>
</tr>
<tr>
<td>32K - 80K</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>1,900</td>
<td>3,700</td>
</tr>
<tr>
<td>80K - 160K</td>
<td>4</td>
<td>10</td>
<td>1</td>
<td>7,300</td>
<td>18,000</td>
</tr>
<tr>
<td>160K - 325K</td>
<td>5</td>
<td>20</td>
<td>1</td>
<td>18,000</td>
<td>38,400</td>
</tr>
<tr>
<td>325K - 815K</td>
<td>6</td>
<td>50</td>
<td>1</td>
<td>38,400</td>
<td>68,000</td>
</tr>
<tr>
<td>815K - 1.6M</td>
<td>7</td>
<td>100</td>
<td>1</td>
<td>135,000</td>
<td>23,203 ms (23.3 sec)</td>
</tr>
<tr>
<td>1.6M - 3.2M</td>
<td>8</td>
<td>200</td>
<td>1</td>
<td>23,203</td>
<td>38,400</td>
</tr>
<tr>
<td>3.2M - 6M</td>
<td>9</td>
<td>400</td>
<td>1</td>
<td>38,400</td>
<td>68,000</td>
</tr>
</tbody>
</table>

†If StepNLoop is enabled, the [HSPD range] values needs to be transposed from PPS (pulse/sec) to EPS (encoder counts/sec) using the following formula:

\[ EPS = \frac{PPS}{\text{Step-N-Loop Ratio}} \]

Acceleration/Deceleration Range

The allowable acceleration/deceleration values depend on the LS and HS settings.

The minimum acceleration/deceleration setting for a given high speed and low speed is shown below.

\[ \text{Max ACC} = \left(\frac{\text{HS} - \text{LS}}{\delta}\right) \times 1000 \text{ [ms]} \]

Note: The ACC parameter will be automatically adjusted if the value exceeds the allowable range.

Examples:

a) If HSPD = 20,000 pps, LSPD = 10,000 pps:
   a. Min acceleration allowable: \(1\text{ ms}\)
   b. Max acceleration allowable:
      \[
      \left(\frac{20,000 - 10,000}{775}\right) \times 1,000 \text{ ms} = 12,903 \text{ ms (12.9 sec)}
      \]

b) If HSPD = 900,000 pps, LSPD = 9,000 pps:
   a. Min acceleration allowable: \(1\text{ ms}\)
   b. Max acceleration allowable:
      \[
      \left(\frac{900,000 - 9,000}{38,400}\right) \times 1000 \text{ ms} = 23,203 \text{ ms (23.3 sec)}
      \]
Acceleration/Deceleration Range – Positional Move

When dealing with positional moves, the controller automatically calculates the appropriate acceleration and deceleration based on the following rules.

1) **ACC vs. DEC 1:** If the theoretical position where the controller begins deceleration is less than $L/2$, the acceleration value is used for both ramp up and ramp down. This is regardless of the EDEC setting.

2) **ACC vs. DEC 2:** If the theoretical position where the controller begins constant speed is greater than $L/2$, the acceleration value is used for both ramp up and ramp down. This is regardless of the EDEC setting.

3) **Triangle Profile:** If either (1) or (2) occur, the velocity profile becomes triangle. Maximum speed is reached at $L/2$.
## Review/Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Summary</th>
<th>ECO Number</th>
<th>Writer/ Reviser</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>09/14/12</td>
<td>Initial Release</td>
<td></td>
<td>Phil Li</td>
</tr>
<tr>
<td>B</td>
<td>3/11/13</td>
<td>Removed servo references</td>
<td>S04_101116</td>
<td>Phil Li</td>
</tr>
<tr>
<td>C</td>
<td>06/18/13</td>
<td>Added Switch silkscreen update (PG23). Changed table 14 from 0 and 1 to up/down. (PG29).</td>
<td>S04_101185</td>
<td>Robert Winslow</td>
</tr>
<tr>
<td>D</td>
<td>08/24/15</td>
<td>Change pinout for J25 (PG27)</td>
<td>S04_102269</td>
<td>Brandon Tveito</td>
</tr>
</tbody>
</table>