OPERATING INSTRUCTIONS

Sizes
17, 23 & 34
The information in this book has been carefully checked and is believed to be accurate; however, no responsibility is assumed for inaccuracies.

Intelligent Motion Systems, Inc., reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Intelligent Motion Systems, Inc., does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights of others. Intelligent Motion Systems and IMS™ are trademarks of Intelligent Motion Systems, Inc.

Intelligent Motion Systems, Inc.’s general policy does not recommend the use of its products in life support or aircraft applications wherein a failure or malfunction of the product may directly threaten life or injury. Per Intelligent Motion Systems, Inc.’s terms and conditions of sales, the user of Intelligent Motion Systems, Inc., products in life support or aircraft applications assumes all risks of such use and indemnifies Intelligent Motion Systems, Inc., against all damages.
**Getting Started - MDrive Motion Control**

**Before you Begin**
This Quick Start will allow you to rapidly set-up and connect your system and “Turn” your MDrive Motion Control utilizing the Immediate Mode of the IMS Terminal User Interface Software. However, it is recommended that you read this entire MDrive Motion Control Manual prior to placing the unit into full operation.

**Tools and Equipment Required**
- MDrive Motion Control Unit
- IMS MD-CC200-000 Communication Cable or equivalent
- IMS Product CD
- An Unregulated Power Supply (See specifications for your exact MDrive Motion Control and required voltage.)
- Basic Tools: Wire Cutters / Strippers / Screwdriver
- Wire for Power Supply (See specifications for your exact MDrive Motion Control.)
- An IBM compatible PC running Windows 9x (95/98) or Windows NT (Windows NT4.0 SP6, Windows 2000 SP1, Windows XP)
- 10 MB hard drive space.
- A free serial communications port.

**Connecting the Power Supply**
Using the recommended wire (see the specifications for your MDrive Motion Control), connect the DC output of the power supply to the red wire on MDrives with flying leads or Pin #7 on units with pluggable connectors.
Connect the power supply ground to the MDrive’s black flying lead or Pin #6 on pluggable units.

**Connecting Communications**
Connect the Host PC to the MDrive Motion Control using the IMS MD-CC200-000 Communication Cable or equivalent.

**Install IMS Terminal**
Insert the IMS CD into the CD Drive of your PC.
The CD will autostart to the IMS CD Main Page. If the CD does not autostart, click “Start > Run” and type “x:\IMS.exe” in the “Open” box and click OK.

**NOTE:** “x” is your CD ROM drive letter.
Place your mouse pointer over the MDrive Icon. The text message “MDrive Integrated Motor & Electronics” will be displayed. This verifies you have selected the correct software.
Click the MDrive Motor Icon. This opens the MDrive Index Page.
Place the mouse pointer over the menu and select IMS Terminal (Win9x) or IMS Terminal (WinNT). The displayed text will again verify your selection. Click your selection and the “Setup” dialog box will be displayed.
Click SETUP in the Setup dialog box and follow the on-screen instructions.
Once IMS Terminal is installed the Communications Settings can be checked and/or set.

**Establishing Communications**
Open the IMS Terminal by clicking Start>Programs>IMS Terminal>IMS Term.
On the Menu Bar click <Edit> <Preferences> to display the “Preferences” Dialog Box.
Click the “Comm Settings” tab at the top of the dialog box. The Comm Settings page will be displayed.
Under “Device” near the bottom of the box verify “MDrive” is selected. All other settings should be left as is.
Click OK.

**Apply Power to the MDrive Motion Control**
Verify all connections are made and apply power to the MDrive Motion Control. Upon Power-Up the following sign-on message should appear in the Terminal window:

“Copyright 2001-2003 by Intelligent Motion Systems, Inc.”

If you can see this sign-on message then you are up and running! If the sign-on message does not appear, try using a software reset. Hold down the “Ctrl” key and press “C” (^C). If the sign-on message still does not appear then there may be a problem with either the connections, hardware or software configuration of the MDrive Motion Control or Host PC.

There are indicators at the bottom of the Terminal Window that show whether you are Connected or Disconnected, the current Baud Rate and the type of device (MDrive) for which the IMS Terminal is configured. These three items may be changed directly from this screen by double clicking on them.
Double Click on “Connected” and the system will disconnect.
Double Click on “Disconnect” and the system will connect.
**Testing the MDrive Motion Control Setup**

NOTE: The character ∧ is used to indicate a space. Do not type this character but be sure to type the space. Click anywhere within the Terminal Window (Right Window) to activate it. The Print instruction <PR> is used to report the values of variables and flags. Type the following: PR∧VM and then press ENTER. The MDrive Motion Control should return a value of 768ØØØ

NOTE: The MDrive Motion Control is not case sensitive. You may type in lower or upper case. Type VM∧36ØØØØand press ENTER. Type PR∧VM and press ENTER. The MDrive Motion Control should return a value of 36ØØØØ

Type FD and press ENTER. (FD = Factory Defaults.) The “Copyright 2001-2003 by Intelligent Motion Systems, Inc.” Message should appear.

CONGRATULATIONS! You are communicating.

**Make the MDrive Motion Control Move**

NOTE: You may want to put a small piece of tape on the motor shaft so you can see it turn.

Type MR∧512ØØØ and press ENTER. (MR=Move Relative.) With the Default settings, the MDrive Motion Control should move one revolution in approximately Ø66 seconds or at a velocity of 15 revolutions per sec.

Type SL∧1ØØ4ØØand press ENTER. (SL = Slew.) With the Default Settings, the MDrive Motion Control should run constantly at a speed of approximately 2 revolutions per second or 12ØØrevolutions per minute.

Type SL∧ØØand press ENTER. The MDrive Motion Control should decelerate to a full stop.

**Motion Sample**

This is a simple motion program that sets the position counter to zero (Ø), moves the MDrive Motion Control 1Ø4ØØmicrosteps in the plus direction and then prints the position in the Terminal Window.

Type the following: (It is not necessary to type the comments.)

Type P=Ø 'Sets position counter to Ø

Press ENTER

Type MR∧1ØØ4ØØ 'Set motion mode to relative, move relative 1Ø4ØØ

Press ENTER

Type PR∧P 'Print position to terminal

Press ENTER

NOTE: The program or motion can be stopped by pressing the Escape Button or by pressing <Ctrl C>.

The Motion Sample above may also be run from a program. If you wish to run from a program go to Section 2.2 in this Manual and then go to the heading “Creating, Downloading and Uploading Programs”. Take yourself through Creating a New Program, Formatting the Program Text and Downloading a Program to the MDrive Motion Control. At the end of the Download instructions you will be able to run the Program. Note that the Program version of the Motion Sample has examples of setting many of the common variables you will be using on a regular basis.

These are basic commands that verify that your MDrive Motion Control is communicating with your PC. More complex commands and movement may require that your I/O and/or Analog Input be interfaced and configured. Please consult the appropriate sections in this manual for details.
Table of Contents

Part 1: General Information And Hardware Information

Section 1.1: Introduction to the MDrive17 Motion Control ................................................................................................................................. 5
  Introduction to the MDrive17 Motion Control ................................................................................................................................. 5
  Feature Summary ........................................................................................................................................................................... 6

Section 1.2: MDrive17 Motion Control Specifications ....................................................................................................................................... 6
  Section Overview ........................................................................................................................................................................... 6
  Rotary Motor Specifications ........................................................................................................................................................................... 6
    Mechanical Specifications - Dimensions in Inches (mm) .................................................................................................................. 6
    MDrive17 Mounting Screws .................................................................................................................................................................. 6
    MDrive Motion Control 1713 Motor Specs and Speed/Torque Curves .......................................................................................... 7
    MDrive Motion Control 1715 Motor Specs and Speed/Torque Curves .......................................................................................... 7
    MDrive Motion Control 1719 Motor Specs and Speed/Torque Curves .......................................................................................... 7
    General Specifications ........................................................................................................................................................................... 8
    Power Supply Requirements ............................................................................................................................................................... 9
    Recommended IMS Power Supplies .................................................................................................................................................. 9
    Thermal Specifications .......................................................................................................................................................................... 9

Section 1.3: Introduction to the MDrive23 Motion Control .......................................................................................................................... 10
  Introduction to the MDrive23 Motion Control ............................................................................................................................... 10
  Feature Summary ........................................................................................................................................................................... 10

Section 1.4: Drive23 Motion Control Specifications ....................................................................................................................................... 11
  Section Overview ........................................................................................................................................................................... 11
  Rotary Motor Specifications ........................................................................................................................................................................... 11
    Mechanical Specifications - Dimensions in Inches (mm) .................................................................................................................. 11
    MDrive Motion Control 2218 Motor Specs and Speed/Torque Curves .......................................................................................... 11
    MDrive Motion Control 2222 Motor Specs and Speed/Torque Curves .......................................................................................... 12
    MDrive Motion Control 2231 Motor Specs and Speed/Torque Curves .......................................................................................... 12
    Linear Motor Specifications ...................................................................................................................................................................... 12
    Mechanical Specifications - Dimensions in Inches (mm) .................................................................................................................. 12
    Linear Actuator MDrive Motion Control 2218 Specs and Speed-Force Curves .................................................................................. 13
    Speed-Force Curve: 24 VDC ............................................................................................................................................................... 13
    Speed-Force Curve: 45 VDC ............................................................................................................................................................... 13
    MDrive Motion Control ACME Screw .................................................................................................................................................. 14
    General Specifications ........................................................................................................................................................................... 14
    Power Supply Requirements ............................................................................................................................................................... 15
    Recommended IMS Power Supplies .................................................................................................................................................. 15
    Thermal Specifications .......................................................................................................................................................................... 15

Section 1.5: Introduction to the MDrive34 Motion Control .......................................................................................................................... 16
  Introduction to the MDrive34 Motion Control ............................................................................................................................... 16
  Feature Summary ........................................................................................................................................................................... 16

Section 1.6: MDrive34 Motion Control Specifications ....................................................................................................................................... 17
  Section Overview ........................................................................................................................................................................... 17
  Rotary Motor Specifications ........................................................................................................................................................................... 17
    Mechanical Specifications - Dimensions in Inches (mm) .................................................................................................................. 17
    MDrive Motion Control 3424 Motor Specs and Speed/Torque Curves .......................................................................................... 18
    MDrive Motion Control 3431 Motor Specs and Speed/Torque Curves .......................................................................................... 18
    MDrive Motion Control 3447 Motor Specs and Speed/Torque Curves .......................................................................................... 18
    General Specifications ........................................................................................................................................................................... 19
    Power Supply Requirements ............................................................................................................................................................... 20
    Recommended IMS Power Supplies .................................................................................................................................................. 20
    Thermal Specifications .......................................................................................................................................................................... 20

Part 2: Connecting, Configuring, And Programming The MDrive Motion Control

Section 2.1: Interfacing the MDrive Motion Control ....................................................................................................................................... 22
  Section Overview ........................................................................................................................................................................... 22
  Layout and Interface Guidelines ............................................................................................................................................................... 22
  Recommended Wiring ........................................................................................................................................................................... 22
  Pin Configuration and Descriptions ............................................................................................................................................................... 22
  Interfacing Power ........................................................................................................................................................................... 23
  Securing MDrive Power and Logic Leads .................................................................................................................................................. 23
  Interfacing RS-485 Communications ............................................................................................................................................................... 24
    Single MDrive ........................................................................................................................................................................... 24
    RS-485 2 Wire Communication (Half Duplex) ........................................................................................................................................ 24
    Multiple MDrive Motion Control System (Party Mode) ........................................................................................................................................ 25
    Data Cable Termination Resistors ............................................................................................................................................................... 25
    MDI Communication Format ............................................................................................................................................................... 26
    MDI Response to Echo Mode ............................................................................................................................................................... 26
    Using Check Sum ........................................................................................................................................................................... 27
    MDrive Motion Control Party Mode Sample Codes ........................................................................................................................................ 28
    MDrive Motion Control Immediate Party Mode Sample Codes ........................................................................................................................................ 29
Part 1: General Information and Hardware Specifications
Section 1.1
Introduction to the MDrive17 Motion Control

Introduction to the MDrive17 Motion Control

The MDrive17 Motion Control offers the system designer a low-cost, intelligent motion controller integrated with a NEMA 17 high torque stepping motor and a +12 to +48 VDC microstepping drive.

The MDrive17 Motion Control adds a versatile array of functions by combining a complete programmable motion controller with our already compact and cost effective standard MDrive17, adding little cost and no increase in size. Standard offerings include four +5V logic to +24V logic programmable I/O points, one 10-bit 0 to 5 volt analog input, 0 to 5 MHz step clock rate, microstep resolution up to 51,200 steps per revolution and a full featured easy-to-program instruction set.

The MDrive17 Motion Control communicates using the RS-485 communications protocol, this allows for point-to-point or multidrop communications using one communications port. Addressing and hardware support up to 62 MDrive nodes in a system. The communications BAUD rate is software selectable and ranges from 4.8 kbps to 115 kbps.

The MDrive17 is also available with an optional closed loop control. The closed loop configuration adds a 512 line (2048 count) internal magnetic rotary encoder with index mark without increasing the length of the unit. Closed loop configuration adds position maintenance, stall detection and find index mark.

Available motor configurations include a single shaft and an external linear actuator only (consult factory for availability). Rotary versions are available in three motor lengths: 13, 15 & 19. Interface connections are accomplished with 12” flying leads.

Feature Summary

- Integrated Microstepping Drive/Motion Controller with Optional Encoder/NEMA 17 High Torque Stepping Motor
- +12 to +48VDC Input Voltage
- Low Cost
- Extremely Compact
- Available Configurations: Single Shaft, External Linear Actuator (Consult Factory for Availability), Integral Encoder
- Three Motor Stack Lengths Available
- Single Power Supply
- Microstep Resolution up to 51,200 Steps Per Revolution
- Open Loop or Optional Closed Loop Control
- Programmable Motor Run and Hold Current Settings
- Four 0V to 24V Inputs that work with +5V Logic (i.e. TTL, CMOS, etc.) or +24V Logic (i.e. PLC’s)
- One Analog 10 Bit, 0 to 5 Volt Analog Input
- 0 to 5 MHz Step Clock Rate, Selectable in 0.59 Hz Increments
- RS-485 Communications Protocol
- Communications BAUD Rate Selectable from 4.8 kbps to 115 kbps
- 62 Software Addresses for Multidrop Communications
- Simple 1 and 2 Character Programming Instructions
- 12” Flying Lead Interface
- Optional Integrated RS-232 to RS-485 Converter/Communications Cable
Section 1.2

MDrive17 Motion Control Specifications

Section Overview

This section contains mechanical, motor and electrical specifications specific to each version of the MDrive17 Motion Control. Shown are:

- Rotary Motor Specifications
- General Specifications
- Power Supply Requirements
- Thermal Specifications

Rotary Motor Specifications

Mechanical Specifications - Dimensions in Inches (mm)

<table>
<thead>
<tr>
<th>Standard Rotary Motor (Lmax)</th>
<th>Inches (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack 1713</td>
<td>2.187 (55.56)</td>
</tr>
<tr>
<td>1715</td>
<td>2.407 (61.15)</td>
</tr>
<tr>
<td>1719</td>
<td>2.786 (70.77)</td>
</tr>
</tbody>
</table>

Figure 1.1: Rotary MDrive17 Motion Control Mechanical Specifications

MDrive17 Mounting Screws

Care must be observed when installing the mounting screws on ALL MDrive17 versions including Linear Actuators. The mounting holes on the flange are not drilled through and have a maximum depth of 0.150” (3.81 mm).

The warning and note below as well as Figure 1.2 illustrate the maximum safe thread length and maximum torque for mounting all versions of the MDrive17.

WARNING! The mounting holes in the MDrive17 mounting flange are not through holes. The maximum length of the screw threads into the motor flange is 0.140” (3.5 mm). (See Below)

MAXIMUM TORQUE! The maximum torque for the M3x0.5 screw is 7.8 lb-in (9 kg-cm) with a thread engagement of 6.5 threads (3.3 mm deep). A lesser thread engagement diminishes the maximum torque.

Figure 1.2: MDrive17 Mounting Screw Depth
**MDrive Motion Control 1713 Motor Specs and Speed/Torque Curves**

<table>
<thead>
<tr>
<th>MDIF1713</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>32 (22.6)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>1.66 (1.17)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.00053 (0.038)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>10.42 (295.4)</td>
</tr>
</tbody>
</table>

*Table 1.1: Rotary MDIF1713 Motor Specifications*

**Figure 1.3: Rotary MDrive Motion Control 1713 Speed/Torque Data**

**MDrive Motion Control 1715 Motor Specs and Speed/Torque Curves**

<table>
<thead>
<tr>
<th>MDIF1715</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>60 (42.4)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>2.08 (1.47)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.00080 (0.057)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>10.42 (295.4)</td>
</tr>
</tbody>
</table>

*Table 1.2: Rotary MDIF1715 Motor Specifications*

**Figure 1.4: Rotary MDrive Motion Control 1715 Speed/Torque Data**

**MDrive Motion Control 1719 Motor Specs and Speed/Torque Curves**

<table>
<thead>
<tr>
<th>MDIF1719</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>74.9 (52.9)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>3.47 (2.45)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.00116 (0.082)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>11.80 (334.5)</td>
</tr>
</tbody>
</table>

*Table 1.3: Rotary MDIF1719 Motor Specifications*

**Figure 1.5: Rotary MDrive Motion Control 1719 Speed/Torque Data**
General Specifications - MDrive17 Motion Control

Input Voltage (+V)
- Range: +12 to +48 VDC

WARNING! The maximum +48 VDC Input Voltage of the MDrive17 includes Motor Back EMF, Power Supply Ripple and High Line.

A characteristic of all motors is back EMF. Back EMF is a source of current that can push the output of a power supply beyond the maximum operating voltage of the driver. As a result, damage to the stepper driver could occur over a period of time. Care should be taken so that the back EMF does not exceed the maximum input voltage rating of the MDrive17.

Analog Input
- Resolution: 10 Bit
- Voltage Range: 0 to +5 Volts
- Current Range: 4 - 20 mA

Programmable I/O
- Number: 4
- Interface Type: Open Collector
- Voltage Range: 0 to +24 VDC
- Logic Threshold:
  - Logic 0: <0.8VDC
  - Logic 1: >2.2VDC
- Output Sink Current: 700 mA
- Protection: Over Temp., Short Circuit, Inductive Clamp

Communication
- Protocol: RS-485, Full/Half Duplex Selectable
- BAUD Rate: 4800, 9600, 19.2k, 38.4k, 115.2k

Motion
- Microstep Resolution – Open Loop Configuration
  - Number of Settings: 14
  - Steps per Revolution: 400, 800, 1000, 1600, 2000, 3200, 5000, 6400, 10000, 12800, 25000, 25600, 50000, 51200
- Microstep Resolution – Closed Loop Configuration (Optional)
  - Steps per Revolution (Fixed): 51200

Encoder (Optional)
- Type: Internal, Magnetic
- Resolution: 512 Lines/2048 counts per Revolution

Counters
- Type: Position(C1), Encoder (C2)
- Resolution: 32 Bit
- Edge Rate (Max): 5 MHz

Velocity
- Range: ±5,000,000 Steps per Second
- Resolution: 1 Step per Second

Acceleration/Deceleration
- Range: 1.5 x 10⁶ Steps per Second²
- Resolution: 90.9 Steps per Second²
Power Supply Requirements

Each MDrive17 Motion Control will require a **maximum power supply current of 2A**. Actual power supply current will depend upon the load and duty cycle.

**WARNING:**
- DO NOT connect or disconnect power leads when power is applied!
- Disconnect the AC power side to power down the DC power supply.
- For battery operated systems, connect a "transient suppressor" across the power switch to prevent arcs and high voltage spikes.

Recommended IMS Power Supplies

For the MDrive17 Motion Control, below are the recommended IMS power supplies.

**IP404 Unregulated Linear Supply**

<table>
<thead>
<tr>
<th>Input Range</th>
<th>120 VAC Versions</th>
<th>240 VAC Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Load Output Voltage*</td>
<td>102-132 VAC</td>
<td>204-264 VAC</td>
</tr>
<tr>
<td>Continuous Output Rating*</td>
<td>32 VDC @ 2 Amps</td>
<td></td>
</tr>
<tr>
<td>Peak Output Rating*</td>
<td>26 VDC @ 4 Amps</td>
<td></td>
</tr>
</tbody>
</table>

**ISP200-4 Unregulated Linear Supply**

<table>
<thead>
<tr>
<th>Input Range</th>
<th>120 VAC Versions</th>
<th>240 VAC Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Load Output Voltage*</td>
<td>41 VDC @ 0 Amps</td>
<td></td>
</tr>
<tr>
<td>Continuous Output Rating*</td>
<td>38 VDC @ 1.5 Amps</td>
<td></td>
</tr>
<tr>
<td>Peak Output Rating*</td>
<td>35 VDC @ 3 Amps</td>
<td></td>
</tr>
</tbody>
</table>

*All measurements were taken at 25°C, 120 VAC, 60 Hz.

**WARNING!** The maximum +48 VDC Input Voltage of the MDrive17 includes Motor Back EMF, Power Supply Ripple and High Line.

Thermal Specifications

Because the MDrive consists of two core components, a drive and a motor, close attention must be paid to the thermal environment where the device is used. The following maximum temperatures apply to the MDrive17:

- **Heatsink Temperature - Max** ................................................................. **85°C**
- **Motor Temperature - Max** ................................................................. **100°C**
Section 1.3

Introduction to the MDrive23 Motion Control

Introduction to the MDrive23 Motion Control

The MDrive23 Motion Control offers the system designer a low-cost, intelligent motion controller integrated with a NEMA 23 high torque stepping motor and a +12 to +48 VDC microstepping drive.

The MDrive23 Motion Control adds a versatile array of functions by combining a complete programmable motion controller with our already compact and cost effective standard MDrive23, adding little cost and no increase in size. Standard offerings include four +5V logic to +24V logic programmable I/O points, one 10-bit 0 to 5 volt analog input, 0 to 5 MHz step clock rate, microstep resolution up to 51,200 steps per revolution and a full featured easy-to-program instruction set.

The MDrive23 Motion Control adds a versatile array of functions by combining a complete programmable motion controller with our already compact and cost effective standard MDrive23, adding little cost and no increase in size. Standard offerings include four +5V logic to +24V logic programmable I/O points, one 10-bit 0 to 5 volt analog input, 0 to 5 MHz step clock rate, microstep resolution up to 51,200 steps per revolution and a full featured easy-to-program instruction set.

The MDrive23 Motion Control communicates using the RS-485 communications protocol, this allows for point-to-point or multi-drop communications using one communications port. Addressing and hardware support up to 62 MDrive nodes in a system. The communications BAUD rate is software selectable and ranges from 4.8 kbps to 115 kbps.

The MDrive23 is also available with an optional closed loop control. The closed loop configuration adds a 512 line (2048 count) internal magnetic rotary encoder with index mark without increasing the length of the unit. Closed loop configuration adds position maintenance, stall detection and find index mark.

Available motor configurations include: single shaft, double shaft with control knob, and long life ACME screw linear actuator. Rotary versions are available in three motor lengths: 18, 22 & 31. Interface connections are accomplished using either a 7 position terminal block or optional 12” flying leads.

Feature Summary

- Integrated Microstepping Drive/Motion Controller with Optional Encoder/NEMA 23 High Torque Stepping Motor
- +12 to +48VDC Input Voltage
- Low Cost
- Extremely Compact
- Available Configurations: Single Shaft*, Linear Actuator, Integral Encoder*, Double Shaft with Knob for Manual Positioning*
- Three Motor Stack Lengths Available*
- Single Power Supply
- Microstep Resolution up to 51,200 Steps Per Revolution
- Open Loop or Optional Closed Loop Control
- Programmable Motor Run and Hold Current Settings
- Four 0V to 24V Inputs that work with +5V Logic (i.e. TTL, CMOS, etc.) or +24V Logic (i.e. PLC’s)
- One Analog 10 Bit, 0 to 5 Volt Analog Input
- 0 to 5 MHz Step Clock Rate, Selectable in 0.59 Hz Increments
- RS-485 Communications Protocol
- Communications BAUD Rate Selectable from 4.8 kbps to 115 kbps
- 62 Software Addresses for Multidrop Communications
- Simple 1 and 2 Character Programming Instructions
- Pluggable Terminal Strip or 12” Flying Lead Interface
- Optional Integrated RS-232 to RS-485 Converter/Communications Cable

*Rotary Motor Only
Section 1.4
MDrive23 Motion Control Specifications

Section Overview
This section contains mechanical, motor and electrical specifications specific to each version of the MDrive23 Motion Control. Shown are:
- Rotary Motor Specifications
- Linear Motor Specifications
- General Specifications
- Power Supply Requirements
- Thermal Specifications

Rotary Motor Specifications

Mechanical Specifications - Dimensions in Inches (mm)

<table>
<thead>
<tr>
<th>Stack</th>
<th>In (mm)</th>
<th>(mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2218</td>
<td>2.632 (66.85)</td>
<td>2.060 (52.33)</td>
</tr>
<tr>
<td>2222</td>
<td>3.000 (76.20)</td>
<td>2.495 (63.39)</td>
</tr>
<tr>
<td>2231</td>
<td>3.960 (100.58)</td>
<td>2.907 (73.84)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stack</th>
<th>In (mm)</th>
<th>(mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2218</td>
<td>3.088 (78.44)</td>
<td>2.500 (63.50)</td>
</tr>
<tr>
<td>2222</td>
<td>3.537 (89.84)</td>
<td>2.950 (74.93)</td>
</tr>
<tr>
<td>2231</td>
<td>4.416 (112.17)</td>
<td>3.357 (85.32)</td>
</tr>
</tbody>
</table>

Figure 1.6: Rotary MDrive23 Motion Control Mechanical Specifications

Table 1.4: Rotary MDI2218 Motor Specifications

<table>
<thead>
<tr>
<th>MDI2218</th>
<th>Holding Torque oz-in (N-cm)</th>
<th>90 (64)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Detent Torque oz-in (N-cm)</td>
<td>3.5 (2.5)</td>
</tr>
<tr>
<td></td>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.0025 (0.18)</td>
</tr>
<tr>
<td></td>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>20.1 (569.8)</td>
</tr>
</tbody>
</table>

Figure 1.7: Rotary MDrive Motion Control 2218 Speed/Torque Data
**MDrive Motion Control 2222 Motor Specs and Speed/Torque Curves**

### Table 1.5: Rotary MDI2222 Motor Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value (Unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>144 (102)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>5.6 (3.92)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.0037 (0.26)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>24.4 (691.7)</td>
</tr>
</tbody>
</table>

![Figure 1.8: Rotary MDrive Motion Control 2222 Speed/Torque Data](image1.png)

**MDrive Motion Control 2231 Motor Specs and Speed/Torque Curves**

### Table 1.6: Rotary MDI2231 Motor Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value (Unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>239 (169)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>9.7 (6.86)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.0065 (0.46)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>38.5 (1091.5)</td>
</tr>
</tbody>
</table>

![Figure 1.9: Rotary MDrive Motion Control 2231 Speed/Torque Data](image2.png)

**Linear Motor Specifications**

**Mechanical Specifications - Dimensions in Inches (mm)**

![Figure 1.10: Linear Actuator MDrive23 Motion Control Mechanical Specifications](image3.png)
Table 1.7: Linear Actuator MDrive23 Motion Control Motor Specifications

<table>
<thead>
<tr>
<th>MDI23 Linear Actuator</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Thrust lbs (kg)</td>
<td>200 (90.7)</td>
</tr>
<tr>
<td>Maximum Screw Deflection</td>
<td>±1°</td>
</tr>
<tr>
<td>Backlash inches (mm)</td>
<td>0.005 (0.127)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>20.4 (578.3)</td>
</tr>
</tbody>
</table>

**Speed-Force Curve: 24 VDC**

Refer to Table 1.8 for screw pitch information

![Figure 1.11: Speed-Force Curve - 24VDC (100% Current)](image)

**Speed-Force Curve: 45 VDC**

Refer to Table 1.8 for screw pitch information

![Figure 1.12: Speed-Force Curve - 45VDC (100% Current)](image)
**WARNING:** The maximum axial load limit for the MDrive23 Linear motor is 200 lbs (90.7 kg). Do not exceed this rating!

**WARNING:** The ACME Screw **MUST NOT** deflect more than ±1 degree perpendicular to the motor face. Additional support for radial loads may be required!

---

### General Specifications - MDrive23 Motion Control

**Input Voltage (+V)**

<table>
<thead>
<tr>
<th>Range</th>
<th>12 to +48 VDC</th>
</tr>
</thead>
</table>

**WARNING!** The maximum +48 VDC Input Voltage of the MDrive23 includes Motor Back EMF, Power Supply Ripple and High Line.

**Analog Input**

- Resolution: 10 Bit
- Voltage Range: 0 to +5 Volts
- Current Range: 4 - 20 mA

**Programmable I/O**

- Number: 4
- Interface Type: Open Collector
- Logic Threshold:
  - Logic 0: <0.8VDC
  - Logic 1: >2.2VDC
- Output Sink Current: 700 mA
- Protection: Over Temp., Short Circuit, Inductive Clamp

**Communication**

- Protocol: RS-485, Full/Half Duplex Selectable
- BAUD Rate: 4800, 9600, 19.2k, 38.4k, 115.2k

**Motion**

**Microstep Resolution – Open Loop Configuration**

- Number of Settings: 14
- Steps per Revolution: 400, 800, 1000, 1600, 2000, 3200, 5000, 6400, 10000, 12800, 25000, 25600, 50000, 51200

**Microstep Resolution – Closed Loop Configuration (Optional)**

- Steps per Revolution (Fixed): 51200

**Encoder (Optional)**

- Type: Internal, Magnetic
- Resolution: 512 Lines/2048 counts per Revolution

**Counters**

- Type: Position(C1), Encoder (C2)
- Resolution: 32 Bit
- Edge Rate (Max): 5 MHz

**Velocity**

- Range: ±5,000,000 Steps per Second
- Resolution: 1 Step per Second

---

**Table 1.8: ACME Screws for the MDrive23 Linear Actuator**

<table>
<thead>
<tr>
<th>Screw</th>
<th>Travel/Full Step - Inches (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>0.002 (0.0508)</td>
</tr>
<tr>
<td>A</td>
<td>0.001 (0.0254)</td>
</tr>
<tr>
<td>B</td>
<td>0.000833 (0.0211582)</td>
</tr>
<tr>
<td>C</td>
<td>0.0005 (0.0127)</td>
</tr>
<tr>
<td>D</td>
<td>0.0004167 (0.0079375)</td>
</tr>
<tr>
<td>E</td>
<td>0.0003125 (0.0079375)</td>
</tr>
</tbody>
</table>

---

A characteristic of all motors is back EMF. Back EMF is a source of current that can push the output of a power supply beyond the maximum operating voltage of the driver. As a result, damage to the stepper driver could occur over a period of time. Care should be taken so that the back EMF does not exceed the maximum input voltage rating of the MDrive23.
**Software**

- Program and Data Storage: Non-Volatile
- User Program Space: 767 Bytes
- User Registers: 4, 32 Bit
- User Program Labels and Variables: 22
- Math, Logic AND Conditional Functions: +, -, x, ÷, <, >, =, <=, >=, & (AND), | (OR), ^ (XOR), ! (NOT)
- Branch Functions: Branch & Call (Conditional)

**Predefined I/O Functions**

- Inputs: Home, Limit +, Limit -, Go, Soft Stop, Pause, Jog +, Jog -
- Outputs: Moving, Fault
- Trip Functions: Input, Position
- Party Mode Node Addresses: 62
- Encoder Functions: Stall Detect, Position Maintenance, Find Index

**Power Supply Requirements**

Each MDrive23 Motion Control will require a maximum power supply current of 2A. Actual power supply current will depend upon the load and duty cycle.

**Recommended IMS Power Supplies**

For the MDrive23 Motion Control, below are the recommended IMS power supplies.

**IP404 Unregulated Linear Supply**

- Input Range:
  - 120 VAC Versions: 102-132 VAC
  - 240 VAC Versions: 204-264 VAC
- Output:
  - No Load Output Voltage*: 43 VDC @ 0 Amps
  - Continuous Output Rating*: 32 VDC @ 2 Amps
  - Peak Output Rating*: 26 VDC @ 4 Amps

**ISP200-4 Unregulated Linear Supply**

- Input Range:
  - 120 VAC Versions: 102-132 VAC
  - 240 VAC Versions: 204-264 VAC
- Output:
  - No Load Output Voltage*: 41 VDC @ 0 Amps
  - Continuous Output Rating*: 38 VDC @ 1.5 Amps
  - Peak Output Rating*: 35 VDC @ 3 Amps

*All measurements were taken at 25°C, 120 VAC, 60 Hz.

**WARNING:**

- DO NOT connect or disconnect power leads when power is applied!
- Disconnect the AC power side to power down the DC power supply.
- For battery operated systems, connect a “transient suppressor” across the power switch to prevent arcs and high voltage spikes.

**Thermal Specifications**

Because the MDrive consists of two core components, a drive and a motor, close attention must be paid to the thermal environment where the device is used. The following maximum temperatures apply to the MDrive23:

- **Heatsink Temperature - Max:** 85°C
- **Motor Temperature - Max:** 100°C
Introduction to the MDrive34 Motion Control

The MDrive34 Motion Control offers the system designer a low-cost, intelligent motion controller integrated with a NEMA 34 high torque stepping motor and a +24 to +75 VDC microstepping drive.

The MDrive34 Motion Control adds a versatile array of functions by combining a complete programmable motion controller with our already compact and cost effective standard MDrive34, adding little cost and no increase in size. Standard offerings include four +5V logic to +24V logic programmable I/O points, one 10-bit 0 to 5 volt analog input, 0 to 5 MHz step clock rate, microstep resolution up to 51,200 steps per revolution and a full featured easy-to-program instruction set.

The MDrive34 Motion Control communicates using the RS-485 communications protocol, this allows for point-to-point or multidrop communications using one communications port. Addressing and hardware support up to 62 MDrive nodes in a system. The communications BAUD rate is software selectable and ranges from 4.8 kbps to 115 kbps.

The MDrive34 is also available with an optional closed loop control. The closed loop configuration adds a 512 line (2048 count) internal magnetic rotary encoder with index mark without increasing the length of the unit. Closed loop configuration adds position maintenance, stall detection and find index mark.

Available motor configurations include a single shaft and a double shaft with control knob. Rotary versions are available in three motor lengths: 24, 31, & 47. Interface connections are accomplished with 12” flying leads.

Feature Summary

- Integrated Microstepping Drive/Motion Controller with Optional Encoder/NEMA 34 High Torque Stepping Motor
- +24 to +75VDC Input Voltage
- Low Cost
- Extremely Compact
- Three Motor Stack Lengths Available
- Single Power Supply
- Microstep Resolution up to 51,200 Steps Per Revolution
- Open Loop or Optional Closed Loop Control
- Programmable Motor Run and Hold Current Settings
- Four 0V to 24V Inputs that work with +5V Logic (i.e. TTL, CMOS, etc.) or +24V Logic (i.e. PLC’s)
- One Analog 10 Bit, 0 to 5 Volt Analog Input
- 0 to 5 MHz Step Clock Rate, Selectable in 0.59 Hz Increments
- RS-485 Communications Protocol
- Communications BAUD Rate Selectable from 4.8 kbps to 115 kbps
- 62 Software Addresses for Multidrop Communications
- Simple 1 and 2 Character Programming Instructions
- 12’ Flying Lead Interface
- Optional Integrated RS-232 to RS-485 Converter/Communications Cable
Section 1.6
MDrive34 Motion Control Specifications

Section Overview
This section contains mechanical, motor and electrical specifications specific to each version of the MDrive34 Motion Control. Shown are:
- Rotary Motor Specifications
- General Specifications
- Power Supply Requirements
- Thermal Specifications

Rotary Motor Specifications
Mechanical Specifications - Dimensions in Inches (mm)

![Figure 1.13: Rotary MDrive34 Motion Control Mechanical Specifications](image)

<table>
<thead>
<tr>
<th>Standard Rotary Motor ($L_{MAX}$)</th>
<th>Stack</th>
<th>In (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3424</td>
<td>3.973 (100.91)</td>
<td></td>
</tr>
<tr>
<td>3431</td>
<td>4.551 (115.60)</td>
<td></td>
</tr>
<tr>
<td>3447</td>
<td>6.073 (154.25)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control Knob ($L_{MAX2}$)</th>
<th>Stack</th>
<th>In (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3424</td>
<td>5.083 (129.10)</td>
<td></td>
</tr>
<tr>
<td>3431</td>
<td>5.661 (143.79)</td>
<td></td>
</tr>
<tr>
<td>3447</td>
<td>7.183 (182.44)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1.13: Rotary MDrive34 Motion Control Mechanical Specifications
Table 1.9: Rotary MDIF3424 Motor Specifications

<table>
<thead>
<tr>
<th></th>
<th>MDIF3424</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>381 (269)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>10.9 (7.7)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.01416 (1.0)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>72.3 (2050)</td>
</tr>
</tbody>
</table>

Figure 1.14: Rotary MDrive Motion Control 3424 Speed/Torque Data

Table 1.10: Rotary MDIF3431 Motor Specifications

<table>
<thead>
<tr>
<th></th>
<th>MDIF3431</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>575 (406)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>14.16 (10.0)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.02266 (1.6)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>72.3 (2050)</td>
</tr>
</tbody>
</table>

Figure 1.15: Rotary MDrive Motion Control 3431 Speed/Torque Data

Table 1.11: Rotary MDIF3447 Motor Specifications

<table>
<thead>
<tr>
<th></th>
<th>MDIF3447</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holding Torque oz-in (N-cm)</td>
<td>1061 (749)</td>
</tr>
<tr>
<td>Detent Torque oz-in (N-cm)</td>
<td>19.83 (14.0)</td>
</tr>
<tr>
<td>Rotor Inertia oz-in-sec² (kg-cm²)</td>
<td>0.04815 (3.4)</td>
</tr>
<tr>
<td>Weight (Motor+Driver) oz (gm)</td>
<td>128.7 (3650)</td>
</tr>
</tbody>
</table>

Figure 1.16: Rotary MDrive Motion Control 3447 Speed/Torque Data
A characteristic of all motors is back EMF. Back EMF is a source of current that can push the output of a power supply beyond the maximum operating voltage of the driver. As a result, damage to the stepper driver could occur over a period of time. Care should be taken so that the back EMF does not exceed the maximum input voltage rating of the MDrive34.

**Analog Input**
- Resolution: 10 Bit
- Voltage Range: 0 to +5 Volts
- Current Range: 4 - 20 mA

**Programmable I/O**
- Number: 4
- Interface Type: Open Collector
- Logic Threshold:
  - Logic 0: <0.8VDC
  - Logic 1: >2.2VDC
- Output Sink Current: 700 mA
- Protection: Over Temp., Short Circuit, Inductive Clamp

**Communication**
- Protocol: RS-485, Full/Half Duplex Selectable
- BAUD Rate: 4800, 9600, 19.2k, 38.4k, 115.2k

**Motion**

**Microstep Resolution – Open Loop Configuration**
- Number of Settings: 14
- Steps per Revolution: 400, 800, 1000, 1600, 2000, 3200, 5000, 6400, 10000, 12800, 25000, 25600, 50000, 51200

**Microstep Resolution – Closed Loop Configuration (Optional)**
- Steps per Revolution (Fixed): 51200

**Encoder (Optional)**
- Type: Internal, Magnetic
- Resolution: 512 Lines/2048 counts per Revolution

**Counters**
- Type: Position(C1), Encoder (C2)
- Resolution: 32 Bit
- Edge Rate (Max): 5 MHz

**Velocity**
- Range: ±5,000,000 Steps per Second
- Resolution: 1 Step per Second

**Acceleration/Deceleration**
- Range: 1.5 x 10^9 Steps per Second²
- Resolution: 90.9 Steps per Second²
Power Supply Requirements

Each MDrive34 Motion Control will require a maximum power supply current of 4A. Actual power supply current will depend upon the load and duty cycle.

Recommended IMS Power Supplies

For the MDrive34 Motion Control, below are the recommended IMS power supplies.

IP804 Unregulated Linear Supply

Input Range
- 120 VAC Versions ................................................................. 102-132 VAC
- 240 VAC Versions ................................................................. 204-264 VAC

Output
- No Load Output Voltage* ................................................... 76 VDC @ 0 Amps
- Continuous Output Rating* .................................................. 68 VDC @ 3 Amps
- Peak Output Rating* ............................................................ 64 VDC @ 6 Amps

ISP300-7 Unregulated Linear Supply

Input Range
- 120 VAC Versions ................................................................. 102-132 VAC
- 240 VAC Versions ................................................................. 204-264 VAC

Output
- No Load Output Voltage* ................................................... 68 VDC @ 0 Amps
- Continuous Output Rating* .................................................. 63 VDC @ 2 Amps
- Peak Output Rating* ............................................................ 59 VDC @ 4 Amps

* All measurements were taken at 25°C, 120 VAC, 60 Hz.

WARNING! The maximum +75 VDC Input Voltage of the MDrive34 includes Motor Back EMF, Power Supply Ripple and High Line.

Thermal Specifications

Because the MDrive consists of two core components, a drive and a motor, close attention must be paid to the thermal environment where the device is used. The following maximum temperatures apply to the MDrive34:

Heatsink Temperature - Max ................................................. 85°C
Motor Temperature - Max .................................................... 100°C
Part 2: Connecting, Configuring and Programming the MDrive Motion Control
Section 2.1

Interfacing the MDrive Motion Control

Section Overview

This section will acquaint the user with connecting and using the MDrive Motion Control.

- Layout and Interface Guidelines
- Pin Configuration and Descriptions
- Interfacing Power
- Interfacing RS-485 Communications
- Interfacing Digital I/O
- Interfacing Analog Input

Layout and Interface Guidelines

Logic level cables must not run parallel to power cables. Power cables will introduce noise into the logic level cables and make your system unreliable.

Logic level cables must be shielded to reduce the chance of EMI induced noise. The shield needs to be grounded at the signal source to earth. The other end of the shield must not be tied to anything, but allowed to float. This allows the shield to act as a drain.

Power supply leads to the driver need to be twisted. If more than one driver is to be connected to the same power supply, run separate power and ground leads from the supply to each driver.

Recommended Wiring

The following wiring/cabling is recommended for use with the MDrive Motion Control:

Power and Ground

- MDI17 and MDI23 .............................................................................................. 20 AWG
- MDI34 .................................................................................................................. 18 AWG

Logic Wiring (I/O, Communications)

- Wire Size .............................................................................................................. 22 AWG

General Practices

The following wire strip length is recommended:

- Wire Strip Length ................................................................................................. 0.250” (6.0 mm)

Pin Configuration and Descriptions

<table>
<thead>
<tr>
<th>Pin</th>
<th>Flying Lead</th>
<th>Wire Size</th>
<th>Function and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>White/Yellow</td>
<td>AWG 22</td>
<td>Open Collector/O Point #1, +5 to +24 VDC</td>
</tr>
<tr>
<td>2</td>
<td>White/Orange</td>
<td>AWG 22</td>
<td>Open Collector/O Point #2, +5 to +24 VDC</td>
</tr>
<tr>
<td>3</td>
<td>White/Violet</td>
<td>AWG 22</td>
<td>Open Collector/O Point #3, +5 to +24 VDC</td>
</tr>
<tr>
<td>4</td>
<td>White/Blue</td>
<td>AWG 22</td>
<td>Open Collector/O Point #4, +5 to +24 VDC</td>
</tr>
<tr>
<td>5</td>
<td>Green</td>
<td>AWG 22</td>
<td>10-8 bit, 0 to +5V Analog Input</td>
</tr>
<tr>
<td>6</td>
<td>Black</td>
<td>AWG 20</td>
<td>Power Ground (Return) MDI17 and MDI23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AWG 18</td>
<td>Power Ground (Return) MDI34</td>
</tr>
<tr>
<td>7</td>
<td>Red</td>
<td>AWG 20</td>
<td>+V: +12 to +48 VDC - MDI17 and MDI23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AWG 18</td>
<td>+V: +24 to +75 VDC, MDI34</td>
</tr>
</tbody>
</table>

Table 2.1: P1 Pin Configuration and Description

NOTE: Wire and insulation type are subject to the user’s application and environment.
Interfacing Power

An advantage of the MDrive Motion Control is that only a single unregulated linear or unregulated switching power supply is required to power the control circuitry and motor power.

A maximum of:
- 2A output .............. MDI17 and MDI23
- 4A output .............. MDI34

is required from the supply for each MDrive. Note that the actual power required will be based upon the load and duty cycle.

Wiring should be accomplished using shielded twisted pair of appropriately gauged wires (see previous page). The shield should be attached to earth at the power supply end and left floating at the MDrive end.

Securing MDrive Power Leads and Logic Leads

Some applications may require that the MDrive move with the axis motion. If this is a requirement of your application, the motor leads (pluggable or flying) must be properly anchored. This will prevent flexing and tugging which can cause damage at critical connection points in the MDrive electronics.

DO NOT bundle the Logic Leads with the Power Leads.
Interfacing RS-485 Communications

The MDrive Motion Control communicates to the host using the RS-485 protocol. Communications may be configured as either half or full duplex using the EM (Echo Mode) Instruction. RS-485 may be used in two ways: either to communicate to a single MDrive Motion Control, or to address up to 62 individually named MDrive nodes in a multidrop system.

Single MDrive

Optionally available for the MDrive Motion Control is a communications cable, IMS P/N MD-CC200-000, which has built-in RS-232 to RS-485 conversion circuitry. This will allow you to connect the serial port of your PC* directly to the MDrive Motion Control.

NOTE: Termination resistors may be required on the Data Cables. (Please see Next Page.)

RS-485 2 Wire Communication

The MDrive Motion Control can be operated in a 2 wire RS-485 communication bus. Before connecting the 2 wire RS-485, download your program and setup instructions using the standard 4 wire RS-485 Communications Cable. If a program is not being used, download and save any setup parameters. To ensure the MDrive responds only to commands specifically meant for it, set the MDrive in Party Mode (Please see Party Mode on Page 25). The Echo Mode command (EM) must be set to the value of 1 (EM=1). This will set the MDrive communication into “half duplex” mode. Connect the MDrive in the 2 wire RS-485 configuration.

The following diagram illustrates how to connect the MDrive 4 wire RS-485 to operate as a 2 wire system.

NOTE: The RS-232 to RS-485 cable, Part # MDCC-200-000 cannot be used in a 2 wire RS-485 system. If you wish to monitor the MDrive serial communication from a PC, an RS-232 to RS-485 2 wire converter must be used. This will allow you to monitor the MDrive.
Multiple MDrive Motion Control System (Party Mode)

In systems with multiple controllers it is necessary to communicate with the control modules using party mode (PY=1) of operation. The MDrive Motion Control nodes in the system are configured in software for this mode of operation by setting the Party Flag (PY) to True (1). It is necessary for all of the nodes in a system to have this configuration selected. When operating in party mode each MDrive Motion Control in the system will need a unique address, or name, to identify it in the system. This is accomplished by using the software command DN, or Device Name. For example, to set the name of an MDrive to “A” you would use the following command: DN=65 or DN="A" (65 is the ASCII decimal equivalent of uppercase A). The factory default name is “!”. The asterisk character “*” is used to issue global commands to every device in the system. NOTE: When using the asterisk “*” in Party Mode, typed entries and commands will not be echoed. See Appendix A for ASCII table.

In setting up your system for party operation, the most practical approach is to observe the following steps:

1. Connect the first MDrive Motion Control to the Host PC configured for Single Mode Operation.
2. Establish communications and download program if required.
3. Using the command DN, name the MDrive Motion Control. This can be any upper or lower case ASCII character or number 0-9. (DN="A"{enter}) (Note: The parentheses before and after the device name are required.)
4. Set the party flag PY=1{enter}.
5. Press CTRL+J to activate the Party Mode.
6. Type the letters AS and press CTRL+J (Save device name and Party Mode).
7. Remove power.
8. Repeat steps 1 through 7 for each additional MDrive in the system.
9. After all MDrives are assigned a Device Name the Multiple MDrive Interface can be configured as shown below.

Data Cable Termination Resistors

Data Cable lengths greater than 15 feet (4.5 meters) are susceptible to signal reflection and/or noise. IMS recommends 120Ω termination resistors at both ends of the Data Cables. An example of resistor placement is shown in Figure 2.6. For systems with Data Cables 15 feet (4.5 meters) or less, the termination resistors are generally not required.
**MDI Communication Format**

The following communication formats used by MDrive Motion Control (MDI) units, began with firmware version 1.043.

- `{}` The contents between the `{}` symbols are transmitted.
- `{0D}` Hex equivalent for a CR (Carriage Return).
- `{0A}` Hex equivalent for a LF (Line Feed).
- `{DN}` Represents the Device Name being sent.
- `{CS}` Check Sum; `{ACK} 06 Hex; `{NAK} 15 Hex

**EM = Echo Mode; PY = PartY Mode; CK = ChecK sum**

The word `{command}` represents the immediate command sent to the MDI. Command Execution Time (CET) is the time the MDI takes to execute a command. This varies from command to command and usually is in the 1-5 millisecond range.

**MDI Response to Echo Mode**

Dependent on how the Echo Mode is set in conjunction with Party Mode and Check Sum, the MDI will respond differently. The following tables illustrate the various responses based on how the EM, PY and CK parameters are set.

**Table 2.3: MDI Response to Echo Mode when Party and Check Sum are Zero (0)**

<table>
<thead>
<tr>
<th>Parameter Setting</th>
<th>Transmission to MDI</th>
<th>MDI Initial Response</th>
<th>MDI Final Response</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM=0 &amp; PY=0 CK=0</td>
<td>(command) (D)</td>
<td>(command) (CR)</td>
<td>CET (0D) (0A) &gt;</td>
<td>The last character sent is the prompt &gt;</td>
</tr>
<tr>
<td>EM=1 &amp; PY=0 CK=0</td>
<td>(command) (0D)</td>
<td>(command) (CR)</td>
<td>CET (0D) (0A)</td>
<td>The last character sent is the LF</td>
</tr>
<tr>
<td>EM=2 &amp; PY=0 CK=0</td>
<td>(command) (0D)</td>
<td>(command) (CR)</td>
<td>No response except to PR and L commands</td>
<td></td>
</tr>
<tr>
<td>EM=3 &amp; PY=0 CK=0</td>
<td>(command) (0D)</td>
<td>(command) (CR)</td>
<td>Queue response. The last character sent is the LF</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2.4: MDI Response to Echo Mode when Party is One (1) and Check Sum is Zero (0)**

<table>
<thead>
<tr>
<th>Parameter Setting</th>
<th>Transmission to MDI</th>
<th>MDI Initial Response</th>
<th>MDI Final Response</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM=0 &amp; PY=1 CK=0</td>
<td>(DN) (command) (0A)</td>
<td>(command) (CR)</td>
<td>CET (0D) (0A)</td>
<td>The last character sent is the prompt &gt;</td>
</tr>
<tr>
<td>EM=1 &amp; PY=1 CK=0</td>
<td>(DN) (command) (0A)</td>
<td>(command) (CR)</td>
<td>CET (0D) (0A)</td>
<td>The last character sent is the LF</td>
</tr>
<tr>
<td>EM=2 &amp; PY=1 CK=0</td>
<td>(DN) (command) (0A)</td>
<td>(command) (CR)</td>
<td>No response except to PR and L commands</td>
<td></td>
</tr>
<tr>
<td>EM=3 &amp; PY=1 CK=0</td>
<td>(DN) (command) (0A)</td>
<td>(command) (CR)</td>
<td>Queue response. The last character sent is the LF</td>
<td></td>
</tr>
</tbody>
</table>
Table 2.5: MDI Response to Echo Mode when Party is Zero (0) and Check Sum is One (1)

Table 2.6: MDI Response to Echo Mode when Party and Check Sum are One (1)

Using Check Sum

For communication using Check Sum, the following 2 commands demonstrate sending and receiving.

Sending Command

1. Check Sum set to ZERO before first character is sent.
2. All characters (ASCII values) are added to Check Sum, including the Device Name DN (if PY=1), to the end of the command, but not including terminator.
3. Check Sum is 2’s complement OR’d with Hex 80 (prevents Check Sum from being seen as Command Terminator).

Example command:

```
MR (space) 1
77 82 32 49 Decimal value
4D 52 20 31 Hex
77+82+32+49=240 Add decimal values together
1111 0000 240 Change 240 decimal to binary
0000 1111 1’s complement
0001 0000 Add 1 [2’s complement]
1000 0000 OR result with 128
1001 000 144 Result Check Sum value
```

Once the result is reached, add the check Sum value of 144 to your string by typing: MR 1(Alt Key + 0144) (use the symbol of 144 in your string by holding down the alt key and typing 0144).
Receiving Command

1. Check Sum set to ZERO.
2. All characters are added to Check Sum.
3. When receiving a Command Terminator, the lower 7 bits of the Check Sum should be = to ZERO.
   a) If not ZERO, then command is ignored and NAK echoed.
   b) If ZERO, then ACK is sent instead of CR/LF pair.
4. Responses to PR commands will be Check Summed as above, but the receiving device should NOT respond with
   ACK or NAK.

MDrive Motion Control Party Mode Sample Codes

1. Download this segment of code into the first MDrive Motion Control. After downloading the program to the unit,
   follow the Set Up instructions described earlier. Be sure to set your first unit with the unique address of A (device
   name is case sensitive).

   RC=25 'Run current
   HC=5 'Hold current
   MS=256 'Microstep selection
   A=250000 'Acceleration
   D=250000 'Deceleration
   PG 1 'Enter program mode
   S1=0,0 'Setup I/O 1 as an input low true
   LB SU 'Start program upon power up
   LB AA 'Label program AA
   MR 104400 'Move relative 104400 counts
   H 'Hold program execution to complete the move
   LB DD 'Label program DD
   BR DD,I1=0 'Branch to DD if I1=0
   PR "Bex 1" 'Print device name B to execute program at address 1
   H 2000 'Hold program execution 2000 milliseconds
   PR "Cex 1" 'Print device name C to execute program at address 1
   H 2000 'Hold program execution 2000 milliseconds
   BR AA 'Branch to label AA
   PG 'Exit program, return to immediate mode

2. Download this segment of code into your second MDrive Motion Control. After downloading
   the program to the unit, follow the previous party mode instructions. Be sure to set your
   second unit with the unique address of B (device name is case sensitive).

   RC=25 'Run current
   HC=5 'Hold current
   MS=256 'Microstep selection
   A=250000 'Acceleration
   D=250000 'Deceleration
   PG 1 'Enter program mode
   LB BB 'Label program BB
   MR 208000 'Move relative 208000 counts
   H 'Hold program execution to complete the move
   PG 'Exit program, return to immediate mode
3. Download this segment of code into your third MDrive Motion Control. After downloading the program to the unit, follow the previous party mode instructions. Be sure to set your third unit with the unique address of C (device name is case sensitive).

RC=25  'Run current
HC=5   'Hold current
MS=256 'Microstep selection
A=250000 'Acceleration
D=250000 'Deceleration
PG 1  'Enter program mode
LB CC 'Label program CC
MR 300000 'Move relative 300000 counts
H    'Hold program execution to complete the move
PG    'Exit program, return to immediate mode

MDrive Motion Control Immediate Party Mode Sample Codes

Once Party Mode has been defined and set up as previously described under the heading “Multiple MDrive Motion Control System (Party Mode)”, you may enter commands in the Immediate Mode in the IMS Terminal Window. Some examples follow.

Move MDrive A, B or C 10000 Steps
Assuming there are three MDIves set up in Party Mode as shown in the Sample Codes above.

NOTE: When instructed to type CtrlJ, that is the \[Esc\] key + the \[Enter\] key. It will not display in the Terminal Window so be certain you press the correct keys. CtrlJ activates the Party Mode.

The \[ Esc \] symbol represents a space. Be certain to type a space where the \[ Esc \] symbol indicates.

To move MDrive Unit “A”,Press CtrlJ and then type: AMR\[Esc]10000 and press CtrlJ. MDrive Unit “A” will move 10000 steps.

NOTE: Once you have activated Party Mode with the first CtrlJ you do not have to type it before each successive command. However, every command must be followed with a CtrlJ.

To print the position type: APR\[Esc]P and press CtrlJ. The position of MDrive Unit “A” will be printed.

To move MDrive Unit “B” type: BM\[Esc]10000 and press CtrlJ. MDrive Unit “B” will move 10000 steps.

To move all three MDrives at the same time type: *M\[Esc]10000 and press CtrlJ. All MDrives will move 10000 steps.

NOTE: The asterisk (*) is a global command which addresses all units. Since three units can not answer together, the asterisk (*) as well as other global commands will not be displayed in the Terminal Window.

To change a Variable in the “C” unit type: C<variable name><number> and press CtrlJ. The variable will be changed.

To verify the change type: CPR<variable name> and press CtrlJ. The new value will be displayed.

All Commands and Variables may be programmed in this manner.

To take an MDrive out of Party Mode type: <device name> PY=0 and press CtrlJ. That unit will be taken out of Party Mode.

To take all units out of Party Mode type: *PY=0 and press CtrlJ. All units will be taken out of Party Mode.
Interfacing the Digital I/O

The MDrive Motion Control comes standard with a set of four (4) open collector +5 to +24VDC I/O point which may be programmed individually as either general purpose or dedicated inputs or outputs, or collectively as a group.

The digital I/O may be defined as either active HIGH or active LOW. When the I/O is configured as active HIGH, the level is +5 to +24 VDC and the state will be read/set as a “1”. If the level is 0 VDC then the state will be read/set as “0”. Inversely, if configured as active LOW, then the state of the I/O will be read/set as a “1” when the level is LOW, and a “0” when the level is HIGH. The active HIGH/LOW state is configured by the third parameter of the I/O Setup (S1-4) variable, which is explained further on. The goal of this I/O configuration scheme is to maximize compatibility between the MDrive Motion Control and standard sensors and switches.

The MDrive Motion Control’s I/O scheme is a powerful tool for machine and process control.

Uses of the Digital I/O

The I/O may be utilized to receive input from external devices such as sensors, switches or PLC outputs. When configured as outputs, devices such as relays, solenoids, LED’s and PLC inputs may be controlled from the MDrive Motion Control.

Each I/O point may be individually programmed to any one of 9 dedicated input functions, 3 dedicated output functions, or as general purpose inputs or outputs. The I/O may be addressed individually, or as a group. The active state of the line or group may also be set. All of these possible functions are accomplished with of the I/O Setup Variable (S1-4).

Interfacing Inputs

The MDrive Motion Control inputs may be interfaced to a variety of sinking devices. A single input may be programmed to be a general purpose user input, or to one of nine dedicated input functions. These then may be programmed to have an active state of either HIGH or LOW.

Additionally the inputs may read as a group using the “IN” keyword. This will display as a decimal between 0 and 15 representing the 4 bit binary number. Used thus Input 1 is the Least Significant Bit (LSB) and Input 4 will be the Most Significant Bit (MSB).

Interfacing a Single Input Examples

| Input Functions |
|-----------------|-----------------|
| S1-S4 | Function | Active |
| 0   | General Purpose | 0/1 |
| 1   | Home | 0/1 |
| 2   | Limit + | 0/1 |
| 3   | Limit - | 0/1 |
| 4   | G0 | 0/1 |
| 5   | Soft Stop | 0/1 |
| 6   | Pause | 0/1 |
| 7   | Jog + | 0/1 |
| 8   | Jog - | 0/1 |

*Table 2.7: Input Functions*

![Figure 2.7: Input Interfaced to a Switch](image)

![Figure 2.8: Input Interfaced to a PLC](image)
Interfacing Inputs as a Group Example

Sample Software Configuration

'Set inputs to user inputs active low,
S1=0,0
S2=0,0
S3=0,0
S4=0,0

PR IN 'Read BCD State of Input Group

Figure 2.9: TTL Interface to Input Group

<table>
<thead>
<tr>
<th>DEC</th>
<th>IO4</th>
<th>IO3</th>
<th>IO2</th>
<th>IO1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2.8: I/O Group Truth Table
**Interfacing Outputs**

The MDrive Motion Control Outputs may be configured as either general purpose or set to one of two dedicated functions, Fault or Moving. These outputs will sink up to 700 mA max and may be connected to +5 to +24VDC. Note that a current limiting resistor may be required to limit the current to 700 mA.

As with the inputs the MDrive Motion Control Outputs may be used singularly or collectively as a group.

**Interfacing a Single Output Examples**

![Diagram](image1)

**Figure 2.10: Output Interfaced to an LED**

![Diagram](image2)

**Figure 2.11: Output Interfaced to a Relay**

**Interfacing Outputs as a Group Example**

To write to the outputs as a group the OT instruction is used. This will give you a binary output of 0000 to 1111 from a decimal entry of 0-15. Output 1 will be the Least Significant Bit (LSB), Output 4 will be the Most Significant Bit (MSB). See Table 2.4 for Truth Table.

![Diagram](image3)

**Figure 2.12: Outputs Interfaced to LED’s as a Group**

<table>
<thead>
<tr>
<th>Output Functions</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>S&lt;point&gt;=</td>
<td></td>
</tr>
<tr>
<td>16 General Purpose</td>
<td>0/1</td>
</tr>
<tr>
<td>17 Moving</td>
<td>0/1</td>
</tr>
<tr>
<td>18 Fault</td>
<td>0/1</td>
</tr>
<tr>
<td>19 Stall</td>
<td>0/1</td>
</tr>
<tr>
<td>20 Velocity Changing</td>
<td>0/1</td>
</tr>
</tbody>
</table>

**Table 2.9: Output Functions**
Interfacing the Analog Input

The analog input of the MDrive Motion Control is configured from the factory as a 0 to 5V, 10 bit resolution input ($S5 = 9$). This offers the user the ability to receive input from temperature, pressure or other forms of sensors, and then control events based upon the input.

The value of this input will be read using the I5 instruction, which has a range of 0 to 1024, where $0 = 0$ volts and $1024 = 5.0$ volts. You may then use the program branch (BR) or subroutine call (CL) instructions to control events within the system.

The MDrive Motion Control may also be configured for a 4 - 20 mA Analog Input ($S5 = 10$).

Sample Usage

```
******Main Program*************
S5=9                      'set analog input to read variable voltage (0 to +5VDC)
PG 100                  'start prog. at address 100
LB A1                   'label program A1
CL A2, I5<500          'Call Sub A2, If I5 is less than 500
CL A3, I5>524          'Call Sub A3, If I5 is greater than 524
BR A1                  'loop to A1
E                       'End
PG                      'Exit program

******Subroutines************
LB A2                   'label subroutine A2
MA 2000              'Move Absolute 2000 steps
H                     'Hold program execution until motion ceases
RT                    'return from subroutine

LB A3                   'label subroutine A3
MA -2000             'Move Absolute -2000 steps
H                     'Hold program execution until motion ceases
RT                    'return from subroutine
```

Figure 2.13: Analog Input Interface (0 to +5 VDC / 4 - 20 mA)
Section Overview

This section will acquaint the user with basics of MDrive Motion Control Programming

- Installing IMS Terminal Software
- Upgrading the MDrive Firmware
- The MDrive Program

Installing and Using IMS Terminal

System Requirements

- IBM Compatible PC.
- Windows 9x (95/98) or Windows NT (Windows NT4.0 SP6, Windows 2000 SP1, Windows XP)
- 10 MB hard drive space.
- A free serial communications port.

Installation

The IMS Terminal software is a programming/communications interface. This program was created by IMS to simplify programming and upgrading the MDrive Motion Control. The IMS Terminal is also necessary to upgrade the firmware in your MDrive Motion Control. These updates will be posted to the IMS web site at www.imshome.com as they are made available.

To install the IMS Terminal to your hard drive, insert the IMS Product CD into your CD-ROM Drive. The CD should autostart to the IMS Main Index Page. If the CD does not autostart, click “Start > Run” and type “x:\IMS.exe” in the “Open” box and click OK.

NOTE: “x” is your CD ROM drive letter.

1) The IMS Main Index Page will be displayed.

2) Place your mouse pointer over the MDrive Icon. The text message “MDrive Integrated Motor & Electronics” will be displayed. This verifies you have selected the correct software.

3) Click the MDrive Motor Icon. This opens the MDrive Index Page.
4) Place the mouse pointer over the menu and select IMS Terminal (Win9x) or IMS Terminal (WinNT). The displayed text will again verify your selection. Click your selection and the “Setup” dialog box will be displayed.

5) Click SETUP in the Setup dialog box and follow the on-screen instructions.

Once IMS Terminal is installed the Communications Settings can be checked and/or set.

**Configuring Communications Settings**

The communications settings are configured by means of the “Preferences Dialog Box”. The preferences dialog gives the user the ability to set the format for text size, font and color, as well as general communications settings. The optimum communications settings for the MDrive Motion Control are set by default. After the IMS Terminal Software is installed you may start it and perform the configuration.

1) Open the IMS Terminal by clicking Start>Programs>IMS Terminal>IMS Term.

   The following screen will be displayed.

The left window is the Program Editing Window. The right window is the IMS Terminal Window. Resident programs and immediate commands can be executed, stopped and tracked from the Terminal Window.
2) You must select or verify the Communications Port that you will be using with your MDrive.

a) On the Menu Bar click <Edit> <Preferences> or click the Preferences Button on the main Tool Bar to display the "Preferences" Dialog Box.

![Preferences Dialog Box](image)

Figure 2.18: Preferences Dialog Box

b) You may also display the Preferences Dialog Box by right clicking in the Terminal Window. A small dialog box will be displayed.

![Preferences Selection Dialog Box in the Terminal Window](image)

Figure 2.19: Preferences Selection Dialog Box in the Terminal Window

c) Click “Preferences” in the small dialog box to display the Preferences Dialog Box. The Preferences Dialog Box allows you to select window colors and fonts for the Text Editing Window and Terminal Window as well as Communications Setup.

![The Comm Settings Dialog Box](image)

Figure 2.20: The Comm Settings Dialog Box

d) Click the “Comm Settings” tab at the top of the dialog box. The Comm settings page will be displayed.

e) Under “Device” near the bottom of the box verify “MDrive” is selected. The BAUD rate is already set to the MDrive default. Do not change this setting until you have established communications with the MDrive Motion Control. If you change the BAUD rate setting for the MDrive Motion Control, power will have to be cycled for the change to take effect. Ensure that the IMS Terminal preferences are adjusted for the new BAUD settings.

f) Verify the Comm Port you are using.

g) The “Window Size” settings are strictly optional. You may set these to whatever size is comfortable to you.

h) Click “APPLY” and “OK”. The settings will be saved automatically.
3) Verify all connections are made and apply power to the MDrive. The following sign-on message should appear in the Terminal window: “Copyright 2001-2003 by Intelligent Motion Systems, Inc.”

If you can see this sign-on message then you are up and running! If the sign-on message does not appear, try using a software reset. Hold down the “Ctrl” key and press “C” (^C). If the sign-on message still does not appear then there may be a problem with either the connections, hardware or software configuration of the MDrive Motion Control or Host PC.

There are also indicators at the bottom of the page (See Figure 2.21) that show whether you are Connected or Disconnected, the current Baud Rate and the type of device (MDrive) for which the IMS Terminal is configured. These three items may be changed directly from this screen by double clicking on them.

Double Click on “Connected” and the system will disconnect. Double Click on “Disconnect” and the system will connect.

Double Click on the Baud Rate and the preferences page will open so you can change it.

Double Click on the “MDrive” and the preferences page will open with the option to change the drive. (For this application you would not change the MDrive.)

Using the IMS Terminal Software

The IMS Terminal software is an easy to setup and use interface for MDrive Motion Control programming. It is also required to upgrade the firmware in the MDrive Motion Control.

Many of the commands you will be using work in both the Program Editor Window and the Terminal Window. You must have the proper window selected before activating the command.

IMS Terminal Tool Bar

The IMS Terminal Tool Bar is configured with all the necessary functions to operate IMS Terminal.
Creating, Downloading and Uploading Programs

Existing programs may be edited in the Program Editor Window from a file on a disk, a file on the hard drive or a file uploaded from the MDrive. You may also create a new program in the Program Editor Window.

NOTE: Your system must be connected and running and perform these steps as they are outlined.

Creating a New Program
Before you create a program you must have a new Program Editor Window open. Follow these steps:

1) Click on the Drop-Down Menu “View”. The following dialog box will be displayed:

![Figure 2.23: Drop-Down Menu for New Edit Window](image)

2) Click on “New Edit Window”. The following dialog box will be displayed:

![Figure 2.24: Naming the New Program/Program Editor Window](image)

3) You must assign a file name in order to open the new window. If there is no file name the “OK” button will not be highlighted. Name this file `<motion sample.mxt>`. The `<mxt>` extension designates programs for the MDrive.

4) Click “OK” and the new Program Editor Window will be displayed.

![Figure 2.25: New Program Editor Window Named “motion sample.mxt”](image)

Naming the program with the `<mxt>` extension automatically formats the text color and makes most of the characters appear in upper case. When you type a program the text will be color coded. In complex programs it may be difficult to read the text easily. By formatting indents, the overall appearance and readability will be greatly improved.
**Formatting the Program Text**

To format the text for indents you need to call up the “Preferences” dialog box. Click the “Program Editor Format” tab at the top of the box. The screen shown below will be displayed. In the “Edit Features” block (See 1 below) click on the small box to the left of “Auto Indent” and verify there is a check mark (✓) in the box. This will enable Automatic Indents. Once you indent your text with the “Tab” key all subsequent lines will adopt the same indent. Simply backspace to return to the left margin. There is also an “Enable Tabs” option. If this box is checked, tabs will be inserted into your text. If the “Tabs” option is disabled, character spaces will be inserted. For this example the “Enable Tabs” will be turned off. In the “Set” block (See 2 below) you may also set the tab spacing. The default is 2 characters. When completed, click “Apply” and then click “OK”.

![Figure 2.26: Program Editor Preferences](image)

Now you can indent your text. Individual preference will govern how you set up your indents. The format illustrated below is most commonly used. All of the set Variables and Program Modes are left aligned. All the Labels are indented 2 characters or 1 tab. The remaining commands are indented 4 characters or 2 tabs. Indent your text by pressing the “Tab” key.

A program can now be typed into the new Program Editor Window. For this example we will use one of the sample programs found in Appendix C. The program is the first one. It is titled **Motion Sample**.

Type the program in as it is shown in Appendix C. You can type upper or lower case. Be sure to put all spaces in as they are indicated by the ∧ symbol. It is not necessary to put in the comments but they are allowed in the program provided they begin with an apostrophe (’). NOTE: The total number of Characters and spaces must be limited to 64 per line.

As you type, the text will be automatically formatted and color coded for the MDrive. When you edit or type new commands they will appear black and will then be automatically changed to the proper color and case when you press “Enter”. If you type in all lower case characters, upon pressing “Enter” part or all of the text will be changed to upper case characters. This is an indicator that the syntax was correct and accepted by the IMS Terminal. If the entire command line is changed to red with no uppercase characters it is a bad command. Add tabs where they are desired. When complete, your program should resemble the example below. Be sure to SAVE YOUR PROGRAM by clicking “File>Save”.

NOTE: The indicator lines and labels are not part of the program. They have been added for illustration purposes only.

![Figure 2.27: Example of Indented Text](image)

All Commands and Registers are indented 4 characters.

All Labels are indented 2 characters.

Variables and Program Modes are left aligned.
In the illustration below the default color coding is Dark Blue, Light Blue, Red, Green, Olive and Brown. Their designations are:

- **Dark Blue** = Key Words
  - One Upper/One Lower Case = IMS Variables or Flags
  - All Upper Case = IMS Commands
- **Light Blue** = Numerical Signs
- **Red** = User Defined Data
- **Green** = Remarks
- **Olive** = Numerical Values
- **Brown** = Text Strings in Quotes (Not Shown)

The colors may be changed to suit the user's preference. To change the colors call up the “Preferences” page. Click on the “Program Editor Format” tab at the top of the page. In the “Edit Colors” block you can set up your preferential colors for the different parts of your program. These changes will become the defaults after clicking “Apply” and re-saving your program.
**Downloading a Program to the MDrive Motion Control**

**NOTE:** Before downloading any programs type FD and press ENTER to set the MDrive to the Factory Defaults.

There are two basic sources from which you can download programs to the MDrive Motion Control:

1. Directly from the Program Editor Window of the IMS Terminal.
2. From a file folder located on a hard drive or removable disk.

There are also two ways to enable the download dialog box.

1. Click the menu item “Transfer > Download”. The Download Dialog Box will open.
2. Click the Download Button on the Main Tool Bar. The Download Dialog Box will open.

Select the “Source Type > Edit Window” option, and click download. The program will transfer to the MDrive Motion Control.

If a Program has been previously created and stored, it may be downloaded to the MDrive Motion Control from the text file by selecting “Source Type > File” on the dialog box and typing in a drive location/file name in the “File Name” box on the dialog, or by browsing to the file location. Ensure the programs have been saved with the `<mxt>` extension for MDrive.

3. Once the program is downloaded, type S and press ENTER to Save the program. (Always Save your Programs!)
4. Now type EX 100 and press ENTER. (EX=Execute and 100 is the Program Number.) Your MDrive should move 200000 steps and then print the position.

**NOTE:** The program can be stopped by pressing the Escape Button or by pressing <Ctrl C>.

**NOTE:** The program is not downloaded to the Terminal Window. It is downloaded directly to the MDrive. What is shown in the Terminal Window is an echo of the downloaded program.

**NOTE:** Because the program is downloaded directly to the MDrive, the unit must be powered up and the sign-on message must be displayed (communicating).

**NOTE:** When the program is downloaded, the color of all characters will be changed to black and line numbers will be added.

**NOTE:** After the program is downloaded it must be saved. Type an <s> next to the cursor and press Enter to save the program.
Setting the Programmable Function Keys

The IMS Terminal has the capability of programming up to 10 Function Keys, a feature typically found in more advanced terminal programs. The Function Keys can be set to provide quick access to commonly used MDrive Immediate mode commands, execute programs, or even hold entire MDrive programs up to 2048 characters.

To access the function key setup dialog box, right-click the function key area (See Figure 2.35) at the bottom of the Terminal Window. The window below will be displayed. To setup the function keys:

1) In this example the “Save” command is used. Enter “Save” in the Captions text field, this will be displayed on the function button.

2) Enter the text string in the Contents field consisting of MDrive Motion Control commands and ASCII control codes. For the “save” command the letter “s” is entered. Each command must be terminated with a Carriage Return (^M) and a pause time. Typically 50 msec (^m) is sufficient.

A fly-out dialog can be brought up by clicking the arrow on the right of the function key “Contents” field. This enables the programmer to embed common ASCII control codes in the function key text string.

3) Click “Done” to set the function.

Figure 2.33: Function Key (s) Configuration Page

NOTE: When uploading Program Files from the MDrive they will be slightly changed from the original. The MDrive will upload the Program only with the data within the Program. That is, the data between the two Program Modes (PG). Data such as Variables entered outside the PG Modes will not be uploaded. The uploaded program will also have a header ‘[PROGRAMS]’ and a footer ‘[END]. These will not affect your program as they are remarked with the apostrophe (‘) or they can be removed during editing.

You may Upload the Program Variables by clicking “Variables” in the Upload Dialog Box. However, this will upload all of the current Variables, not just those associated with the Program.

Uploading a Program From the MDrive Motion Control

NOTE: Be certain the program is stopped by pressing the Escape Button [Esc] or by pressing <Ctrl C> [C].

There are two ways to upload programs from the MDrive Motion Control:

1) Directly to the Program Editor Window of the IMS Terminal.
2) To a file folder located on a hard drive or removable disk.

There are also two ways to enable the upload dialog box.

1) Click the menu item “Transfer > Upload”. The Upload Dialog Box will open.
2) Click the Upload Button [ ] on the Main Tool Bar. The Upload Dialog Box will open. The Upload Dialog box is similar in appearance to the Download Dialog box.

With the Upload Dialog Box open, select the “Destination Type > Edit Window” option, click “Upload”. The program will transfer from the MDrive Motion Control.

Programs may also be uploaded from the MDrive Motion Control directly to a text file by selecting “Destination Type > ‘File” as the Destination and typing in a drive location:name in the “File Name” box on the dialog box.

NOTE: When uploading Program Files from the MDrive they will be slightly changed from the original. The MDrive will upload the Program only with the data within the Program. That is, the data between the two Program Modes (PG). Data such as Variables entered outside the PG Modes will not be uploaded. The uploaded program will also have a header ‘[PROGRAMS]’ and a footer ‘[END]. These will not affect your program as they are remarked with the apostrophe (‘) or they can be removed during editing.
To activate the Function, Click the F1 Function Key or press the 
F1 key on your keyboard.

Note: Holding the mouse pointer over the function key will display a small identification box which shows the Function Key number and the data it contains. The Function Keys are numbered left-to-right: F1..F5 and F6..F10.

To activate the Function, Click the F1 Function Key or press the 
F1 key on your keyboard.

Note: Holding the mouse pointer over the function key will display a small identification box which shows the Function Key number and the data it contains. The Function Keys are numbered left-to-right: F1..F5 and F6..F10.

Program Troubleshooting

The IMS Terminal offers several tools to help you troubleshoot and analyze programs. They are:

- Execute in Single Step Mode
- Execute in Trace Mode
- The Scroll Back Function
- The Capture Function

Single Step Mode

The Single Step Mode allows the user to execute a program in the Immediate Mode one line at a time. This will help the user to define problem areas by process of elimination. To use Single Step Mode, do the following:

1) Have the system and the program ready to run.
2) To run in Single Step Mode add a comma and the number two (2) to the execute command.
   Example: The Program Label is <aa>. Type EX aa, 2. The program will run one line at a time.
3) Each line will be executed and listed in the Terminal Window and the Program will stop.
4) To execute and list the next line, press the Space Bar.
5) Press the Space Bar for each successive line until the program has completed.

While the program is executing, it will stop after each line is listed. At this time you may enter immediate commands such as velocity variables or actual moves as tests within the program. After entering immediate commands you may continue running in Single Step Mode by pressing the Space Bar again.

If you decide to cancel the Single Step Mode press the “Enter” key and the program will run in normal mode and finish or press Escape (Esc) to abort the program.
Trace Mode

The Trace Mode allows the user to run a program and list each line as it is executed. Running Trace Mode in conjunction with the Scroll Back Function or the Capture Function will enhance your program troubleshooting tasks. To run Trace Mode:

1) Have the system and the program ready to run.
2) To run in Trace Mode add a comma and the number one (1) to the execute command.
   Example: The Program Label is <aa>. Type `EX aa,1`. The program will run in Trace Mode and each line will be executed and listed in the Terminal Window.
3) Each line can now be analyzed.

On very large programs all of the lines may not be displayed if the "Scroll Back Buffer" value is set too low. The Scroll Back Buffer can be set to a higher value allowing you to Scroll Back farther in the program.

The Scroll Back Buffer

The “Scroll Back Buffer” function for the IMS Terminal Window can be set to different line values. It allows you to scroll back in the program that has already been displayed in the Terminal Window. It can be very useful when troubleshooting a long program.

To set the Scroll Back Buffer:

1) Open the Preferences Page for the IMS Terminal Window.
2) Click on the “Comm Settings” tab at the top of the page. The following screen will be displayed.
3) In the highlighted area in the Figure below you will see a dialog box for “Scroll Back”.
4) To the left of the current value there is a small arrow to drop down the list. The list covers up to 2000 lines. You can select a value up to 2000 lines from the list.
5) If you wish to set the value higher, DO NOT open the drop down list. Simply click on the displayed value to highlight it and type in the new value up to a maximum of 32,000 lines.

NOTE: The Scroll Back Buffer utilizes RAM to store the data. The greater you set the Scroll Back Buffer capacity the greater the amount of RAM used.

When running Trace Mode, all lines executed can be analyzed.

On very large programs all of the lines may not be displayed if the "Scroll Back Buffer" value is set too low. The Scroll Back Buffer can be set to a higher value allowing you to Scroll Back farther in the program.

The Capture Function

The Capture Function allows you to capture Terminal Communications into a text file for the purpose of troubleshooting. You may have a program that fails after running a number of times. It may be from an accumulation of position errors or other factors. By enabling the Capture Function you can store an entire text file of the received communications to your hard drive for analysis.
Enable the Capture Function

The Capture function may be enabled through the drop-down menu under “Transfer”. When you click on “Capture” a dialog box will be displayed.

![Image showing the Capture Dialog Box](image)

Figure 2.37: The Capture Dialog Box

Give the file you will be capturing a name and be certain to save it as a `.txt` file and click “Save”.

NOTE: The Capture Function may also be enabled through the Fly-Out menu on the Function Key configuration page by inserting it into the command string in the “Contents” line. However, the Capture Function can not be programmed with the Repeat command.

Upon clicking Save, the faded (disabled) Capture title below the Function Keys will change to “Capture ON” and to black letters.

![Image showing the Capture ON Indicator](image)

Figure 2.39: Capture ON Indicator

You are now ready to run the program. The program in this example will cycle five (5) times. The data will scroll up the Terminal Window while a copy of the data is captured into the text file simultaneously.

Once the program stops, return to the “Transfer” Drop-Down menu and click on “Stop Capture”. The data that is currently in the Terminal Window is now also saved as the prenamed text file in the IMS Folder.

![Image showing the Stop Capture Command](image)

Figure 2.40: Stop Capture Command in Transfer Drop-Down Menu
Upgrading the MDrive Motion Control Firmware

Before Upgrading

First download the version of firmware you wish to use for the upgrade. (www.imshome.com)

An isolated communications system free of electrical noise and interference is essential for trouble free communication.

During upgrades, the communication baud rate is switched from 9600 to 19,200 and is more susceptible to electrical noise. Your communications cable should be kept to a minimum length of 6 feet.

When using a laptop PC it is recommended that you power the RS-232 to RS-485 cable with an external +5 VDC power supply. This will fortify communications.

The MDrive will remain in the Upgrade Mode until the Upgrade is completed. Cycling power will not clear the Upgrade Mode.

Procedure

1) Open “IMS Terminal”. The following screen should be displayed. The left panel is the Program Edit Window and the right panel is the Terminal Window. The Firmware Upgrade will superimpose several dialog boxes and instructions over these two windows.

2) Check to see that the terminal window is set for MDrive communication.
   • Right click in the Terminal Window.
   • Click “Preferences” near the bottom of the pop-up menu.
   • A “Preferences” dialog box will be displayed.
   • Click on the “Comm Settings” tab at the top of the box. The following page will be displayed.
   • Confirm that MDrive is selected in the “Devices” block.

3) Power up the MDrive Motion Controller.
   • The sign on message will appear.
     “Copyright 2001-2003 by Intelligent Motion Systems, Inc.”

4) Check and/or reestablish communications if the sign on message does not appear.

5) Type UG 2956102 in the Terminal Window and then press <enter>. Include the space between the G and the 2.
   • NOTE: The only way to get out of the upgrade mode is to complete the upgrade. Cycling power will not help.
   • The MDrive will return a random symbol character (ô or ö) when it is in the upgrade mode.

6) Click the “Upgrade” menu item on the IMS Terminal menu bar.

7) Message appears: “During upgrade, the baud rate is changed to 19,200.”
   • Click “OK”

8) Message: “Welcome to the MDrive Motion Controller upgrader. Press next to continue.
   • You do not need to enter data in the windows. This will fill in automatically as you progress.
   • Click “Next”
9) The Windows Explorer page “Select MDrive upgrade file” opens.
   • Browse and select the desired version of the upgrade file.
   • Click “Open” or double click the file.

10) Message appears: Step 2 Select upgrade file.
    • The Upgrade Version will now appear in the Upgrade Version window.
    • Click “Next”

11) Message appears: Step 3 Reminder Press cancel if you need to setup Comm port.
    • The Comm port has been setup previously. This is just a reminder.
    • Click “Next”

12) Message appears: Step 4 Connect RS-422 cable to the MDrive Controller.
    • The RS-422 has been connected previously. This is just a reminder.
    • Click “Next”

13) Message appears: Step 5 If MDrive Controller is not in the Upgrade mode, press cancel then type ‘UG 2956102’ in the terminal window.
    • The MDrive Controller was placed in the Upgrade mode previously. **DO NOT ENTER CODE AGAIN.**
    • Click “Next”

14) Message: Step 6 Power up or cycle power to MDrive Controller.
    • The unit has been previously powered up. **Do not cycle power.**
    • Click “Next”
15) Message: Step 7 Establishing Comm with MDrive Controller.
   • Wait for step 8 to appear.

16) Message: Step 8 Press upgrade button to start.
   • Click the upgrade button.

17) Message: Step 9 Press ABORT to abort upgrade.
   • DO NOT abort the upgrade. The MDrive will remain in the upgrade mode and the upgrade will have to be completed.
   • Monitor the progress in the “Upgrading…%” window.
   • Step 10 will appear when DONE

18) Message: Step 10 Resetting MDrive Controller. Then Press DONE.
   • Click “DONE”
   • Upgrade window will close.

19) Press “Control + C” <Ctrl + C> in the Terminal Window to reset the MDrive Controller and exit the upgrade mode.
   • The sign on message will appear. “Copyright 2001-2003 by Intelligent Motion Systems, Inc.”
   • The > cursor will appear.

20) The MDrive Motion Controller firmware has been upgraded.
21) Optional confirmation of the upgrade: Type “PR VR” in the terminal window and press <enter>.
   • The new firmware version is displayed.

NOTE:
The IMS Terminal automatically shifts to a 19,200 Baud Rate upon clicking the “Upgrade” command.

NOTE:
In the event of loss of power or disconnection of the RS-232 cable, the unit will maintain the “Upgrade” mode on Power Up. The Upgrade must be completed. **DO NOT** retype “UG 2956102”!

Begin with Step 6 of the above procedure and continue the Upgrade.
MDrive Motion Control Programming

The MDrive programming language consists of simple 1-2 character mnemonics.

Operational Modes

There are two operational modes for the MDrive. Immediate and Program:

1] Immediate: Commands are issued and executed directly to the MDrive Motion Control by user input into the terminal window.

2] Program: Commands and processes are run from within an MDrive program. This mode is also used for program input.

Basic Components of MDrive Motion Control Software

Instructions

An instruction results in an action. There are three types of Instructions:

Motion

Motion instructions are those that result in the movement of a motor. The syntax of these commands are as such: first type the command followed by a space, and then the velocity or position data. For example, MA 2000 will move the motor to position 2000.

I/O

An I/O instruction results in the change of parameters or the state of an Input or Output. The syntax of these commands are as such: first type the command followed by a space, then the I/O #, then an equal sign, then the data. Example: PR I1 will read the state of input 1, O2=0 will set output 2 to 0.

Program

A program instruction allows program manipulation. The syntax of these vary due to the nature of the command. Some command examples would be: PG 100, which toggles the system into program mode starting at address 100; BR LP, I1=1, which will Branch to a program labeled LP if Input 1 is true.

System

A system instruction is an instruction that can only be used in immediate mode to perform a system operation such as program execution (EX) or listing the contents of program memory (L). For example: EX 100 will execute a program located at line 100 of program memory space, or EX K1 will execute a program labeled K1.

Variables

Variables are labeled data that allow the user to define or manipulate data. These can also be used with the built-in math functions to manipulate data. There are two classes of variables: factory defined and user defined. There are 22 user program labels and variables available. The syntax for each variable may differ.

Factory Defined Variables

These variables are predefined at the factory. They cannot be deleted. When an FD (Factory Default) instruction is given, these variables will be reset to their factory default values. There are two types of factory defined variables:

- Read/Writable: These factory defined variables can have their value altered by the user to effect events inside or outside of a program. For example A (Acceleration Variable) can be used to set the Acceleration, or P (Position Variable) can be used to set a position reference point.

- Read Only: These factory defined variables cannot be manipulated by the user, but contain data that can be viewed or used to effect events inside a program. For example V (Velocity Variable) registers the current velocity of the motor in steps per second.

User Defined Variables

The VA instruction allows the user to assign a 2 bit character name to a user defined (32 bit value) variable.

The restrictions for this command are:

1] A variable cannot be named after an MDrive Motion Control Instruction, Variable or Flag.

2] The first character must be alpha, the second character may be alpha-numeric.

3] A variable is limited to two characters.

With these the user can define a variable to store and retrieve data and perform math functions. When the FD (Factory Defaults) instruction is given, these variables will be deleted! There are two types of user defined variables:

- Global Variables: Global variables are variables that are defined outside of a program. The benefit to using a global variable is that no user program memory is required. For example the user can define a variable called SP for speed by entering VA SP into the terminal. The user can then set that variable to equal the value of the read only variable V (velocity) by entering SP = V into the terminal.

- Local Variables: This type of user defined variable is defined within a program and can only effect events within that program. It is stored in RAM. Examples of this type of variable will be given later in the section. It is worthy of note that a local variable is not static, but is erased and declared again each time a program is executed.
Flags

Flags show the status of an event or condition. A flag will only have one of two possible states: either 1 or 0. Unlike variables, there are only factory defined flags.

Factory Defined Flags

Factory defined flags are predefined at the factory and cannot be deleted. When a FD (Factory Defaults) instruction is given, these flags will be returned to their factory default state. There are two types of factory defined flags:

- Read/Writable: This type of flag is user alterable. They are typically used to set a condition or mode of operation for the MDrive Motion Control. For example EE = 1 would enable encoder operation, or EE = 0 would disable the encoder functions.
- Read Only: Read Only flags cannot be modified by the user. They only give an indication of an event or condition. Typically this type of flag would be used in a program in conjunction with the BR (Branch Instruction) to generate an if/then event based upon a condition. For example the following line of code in a program BR SP, MV = 0 would cause a program to branch to a subroutine named “SP” when the MV, the read only moving flag, is false.

Keywords

Keywords are used in conjunction with the PR and IP instructions to indicate or control variables and flags. For instance, PR UV would print the state of all the user-defined variables to the screen. IP would restore all the factory variables from the EEPROM.

Most Commonly Used Variables and Commands

Variables

MS

MS (Microsteps Select) defines the resolution of the stepping motor.

- An MDrive rotates 1.8° per step or 200 steps per revolution.
- The MS selection divides the number of MDrive steps to yield a finer resolution.
- An MS value of 256 x 200 would yield 51200 microsteps per revolution. (Each Motor step will be divided into 256 Microsteps.)
- The MS default is 256.
- To read the MS value, type PR MS and press enter
- To write the MS value, type MS=<number> and press enter
- As we continue you will see that all motion variables use this value.

P

P indicates the Position in either steps or encoder counts depending upon the enable/disable state of encoder functions.

- P takes its reading from C1 (Counter 1) when encoder functions are disabled. The reading is taken from C2 (Counter 2) when encoder functions are enabled.
- To read the position, type PR P or PR C1/C2 then press enter
- To zero the position, type P=0 then press enter

VI

Initial Velocity in steps per second. (Step size is a function of the value of MS).

- To read the initial velocity, type PR VI then press enter
- To write to the Initial velocity, type VI=<number> then press enter
- The VI default is 1000

VM

Maximum or final Velocity in steps per second. (Step size is a function of the value of MS).

- To read the final velocity, type PR VM then press enter
- To write to the final velocity, type VM=<number> then press enter
- The default VM Value is 768000
A
Acceleration in steps per second\(^2\). (Steps per second, per second.)
- The velocity of the motor will increase by the value of the Acceleration Rate every second until it reaches the programmed velocity in SL mode or it reaches VM.
- To read the acceleration, type PR A then press enter
- To write to the acceleration, type A=<number> then press enter
- The Acceleration Default value is 1000000

D
Deceleration in steps per second\(^2\). (Steps per second, per second.)
- The velocity of the motor will decrease by the value of the Deceleration Rate every second until it reaches the programmed velocity in SL mode or it reaches VI.
- To read the deceleration, type PR D then press enter
- To write to the deceleration, type D=<number> then press enter
- The Deceleration Default value is 1000000

Math Functions
Another powerful feature of the MDrive Motion Control is its ability to perform common math functions and to use these to manipulate data.

<table>
<thead>
<tr>
<th>Math Function</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition</td>
<td>( K2 = P + R2 )</td>
</tr>
<tr>
<td>Subtraction</td>
<td>( K3 = R1 - P )</td>
</tr>
<tr>
<td>Multiplication</td>
<td>( A = A \times 2 )</td>
</tr>
<tr>
<td>Division</td>
<td>( A = A / 2 )</td>
</tr>
</tbody>
</table>

\(^1\)User-defined variable used as an example.

Motion Commands
Motion Commands are those that cause the MDrive to move or affect the movement of the MDrive. There are a few factors that must be considered when programming motion commands. Linear distances, number of revolutions, degrees of rotation and timed moves can be calculated and programmed from these factors.

MA
Move to an Absolute position relative to a defined zero position.
For example, type the following commands followed by pressing enter:

\[ \begin{align*}
P &= 0 & \text{'set the current position to 0 (zero)'} \\
MA & 20000 & \text{'move 20000 steps from 0 in the plus direction'} \\
PR P & \text{'the terminal screen will read 20000'} \\
MA & 3000 & \text{'move to 3000 steps from 0 in the plus direction'} \\
PR P & \text{'the terminal screen will read 3000'}
\end{align*} \]

Absolute moves are always relative to 0 (zero).
You may program moves in the minus direction by typing the minus sign (-) before the value.

MR
Move the number of steps programmed relative to current position.
For example, type the following commands followed by pressing enter:

\[ \begin{align*}
P &= 0 & \text{'set the current position to 0 (zero)'} \\
MR & 20000 & \text{'move 20000 steps from the current position in the plus direction'} \\
PR P & \text{'the terminal screen will read 20000'} \\
MR & 3000 & \text{'move 3000 steps from the current position in the plus direction'} \\
PR P & \text{'notice the position read is 23000 and not 3000'}
\end{align*} \]

Relative moves are cumulative and are either added to or subtracted from the current position.
You may program moves in the minus direction by typing the minus sign (-) before the value.
SL
Move at a constant velocity.

    SL 200000 'the motor moves at a constant velocity 200000 steps per second

The Slew Command overrides the VM (Maximum Velocity) parameter.
The value of the Slew Command may be changed “on the fly”.
You may program moves in the minus direction by typing the minus sign (-) before the value.

H
An H (Hold Command) should typically follow any MA or MR commands in a program so that program execution is suspended until the motion is complete.
Below is a usage example.

    PG 100 'enter program mode at address 100
    LB M1 'label program M1
    MR 20000 'set mode to relative, move relative 20000 steps
    H 'hold until motion completes
    MR -20000 'move relative -20000 steps
    H 'hold until motion completes
    E 'end program
    PG 'exit program mode

A delay time value (1 to 65000 milliseconds) may be programed with the Hold Command.
(Note: There are circumstances where you may not want to hold up program execution.)

All motion is programmed either Microsteps Per Second or (when the Encoder is enabled) Encoder Counts (Pulses) Per Second.
All Motion is directly affected by the Motion Command and the Program Variables.

Factors

Motor Steps:
All IMS MDrives are 200 step motors. They rotate at 1.8° per pulse. 200 steps would equal 1 revolution.

Microsteps: (MS)
Microsteps divide the 200 Motor Steps into smaller steps to improve smoothness and resolution of the MDrive. Using the default setting of 256 for MS, the 200 motor steps are increased to 51200 Microsteps. One motor revolution requires 51200 Microsteps with the MS set at 256. If you were to set the MS to 128, one revolution of the MDrive would now require 25600 Microsteps.

Move Command:
The Move Absolute (MA) and the Move Relative (MR) Commands are programmed in Microsteps Per Second or if the Encoder is enabled, Encoder Pulses Per Second. If the MS was set at 256 and you were to program a move of 51200 Microsteps per second, the MDrive would turn one full revolution in 1 second (1 RPS). This also equates to Revolutions Per Minute (60 RPM). If the MS was set to 128, one full revolution of the motor would be 25600 Microsteps (128 x 200). If you programmed a move of 51200 Microsteps per second, the MDrive would turn 2 full revolutions in 1 second or 120 revolutions per minute.

If the Encoder is enabled the Move Commands use different values. The Encoder has 512 lines and yields 2048 pulses or counts per revolution. Therefore, the MR and MA Command values are programmed in Encoder counts. One full revolution would be programmed as MR or MA 2048.

When the Encoder is enabled, the MS value is defaulted to 256. It cannot be changed.
Knowing these factors you can program a multitude of different movements, speeds, and time intervals.
**Linear Movement:**
You have a rack and pinion or a ball screw to move a linear axis. The rack and pinion or ball screw moves the linear axis 0.1 inches for each revolution. You need to move 7.5 inches.

7.5 inches divided by 0.1 inches = 75 MDrive revolutions.

Assuming an MS of 256 (51200 Microsteps) is programmed, 51200 Microsteps x 75 revolutions requires a move of 3840000 microsteps.

Knowing the values of the Variables as well as the required move, you can calculate the actual time it takes to move the axis the required distance. This is done with a Trapezoidal Profile as shown below.

**Calculating Axis Speed (Velocity)**

There are several steps required to determine the actual axis speed. They are all based on the Trapezoidal Profile above.

**Known Values and Parameters:**

- VM = 768000 Steps/Sec.
- VI = 1000 Steps/Sec.
- A = 1000000 Steps/Sec^2.
- D = 1000000 Steps/Sec^2.
- MA/MR = 3840000 Microsteps

Determine the Acceleration (A) and Deceleration (D) times (t₁ and t₃). Since the Deceleration (D) value is also 1000000 Steps/Sec, the Deceleration time (t₃) will be the same as the Acceleration time (t₁).

\[(t₁ and t₃) = \frac{VM-VI}{A} or \frac{768000 - 1000}{1000000} = 0.767 \text{ Seconds}\]

Determine the distance (Steps) traveled in t₁ or t₃.

Distance = \(\frac{1}{2} (t₁ \times VM)\) or \(\frac{0.767 \times 768000}{2} = 294528 \text{ Steps}\).

Determine the t₂ time.

The t₂ time is calculated by dividing the remainder of MA/MR by VM.

The remainder of MA/MR = MA/MR - (t₁ steps + t₃ steps) or 3840000 - 589056 = 3250944.

\[t₂ = \frac{3250944}{768000} = 4.233 \text{ Seconds}\]

Determine the total time. (t₁ + t₂ + t₃) or (0.767 + 4.233 + 0.767) = 5.767 Seconds

The linear axis took 5.767 seconds to move 7.5 inches or an average speed of 78 inches/minute.

Note that the average speed includes the Acceleration and Deceleration. The maximum axis speed attained is approximately 90 inches/minute.

\[\frac{768000}{51200} \times 0.1 \times 60 = 90 \text{ IPM}\]
**Rotary Movement:**

Again, assume the MS is set to 256. You are using the MDrive to drive a shaft with a timing belt and pulley arrangement. The MDrive pulley is 1” in diameter and the shaft pulley is 2.5” in diameter. You must turn the shaft 270°. First of all, the shaft will rotate 1 full revolution for every 2.5 revolutions of the MDrive. 270° is 0.75 of a revolution. $0.75 \times 2.5 = 1.875$ MDrive revolutions to turn the shaft 270°. If 51200 Microsteps is 1 MDrive revolution then the MDrive must be programmed to move 96000 Microsteps ($51200 \times 1.875$).

You may also do many of the calculations in reverse to calculate MDrive moves to meet a required move of your device. A linear or rotational move as well as speed may be translated into an MDrive command.

In the example above, the belt driven Rotary Table must be turned 110° at 3 RPM. How should the MDrive be set up?

Bear in mind that all the numbers are approximate due to rounding.

Mechanical ratio between the MDrive and the rotary table is 2.666:1. That is, the MDrive must rotate 2.666 revolutions for the table to rotate 1 revolution and the table will rotate 2.666 times slower than the MDrive.

In order to move the table 110° the MDrive must move 293.3°.

$110 \times 2.666 = 293.3°$

If 51200 steps = 1 revolution then 1° = 142.222 steps.

$\frac{51200}{360} = 142.222$ steps

The MDrive must be programmed to move 41713 steps to rotate 293.3°.

$142.222$ steps $\times 293.3° = 41713$ steps

In order to rotate the table at 3 RPM the MDrive must turn at 8 RPM.

$3 \text{ RPM} \times 2.666 = 8 \text{ RPM}$

If you were to set VM at 51200 and MS set at 256 the MDrive will rotate 1 full revolution (51200 steps) in 1 second or 1 RPS. In order to rotate at 8 RPM, the MDrive must rotate at 0.13333 RPS.

$\frac{8}{60} = 0.13333 \text{ RPS}$

In order to rotate at 0.13333 RPS the VM must be set at 6827 steps/sec.

$51200 \times 0.13333 = \text{VM 6827}$
Programming with the Optional Encoder Enabled

An optional 512 line encoder is available. When the Encoder is enabled (EE=1) the programming also changes. All motion must now be programmed by the encoder pulses. The Encoder operates in the “Quadrature” format. That is, there are four Encoder pulses for each Encoder line or 2048 pulses per revolution \((512 \times 4 = 2048)\). (See Figure below.) If you were to program motion using the MR (Move Relative) or the MA (Move Absolute) commands the motor would rotate a distance equal to the pulses.

Example:
A programmed move of 7168 pulses would result in the motor rotating 3.5 revolutions at a velocity controlled by VM.
\[
\frac{7168}{2048} = 3.5 \text{ revolutions}
\]

If you were to program motion using the SL (Slew) command the motor would rotate at a “pulses per second” rate based on the programmed value.

Example:
An SL (Slew) rate of 7168 pulses was programmed. The motor will rotate at 7168 pulses/sec., 3.5 RPS, or 210 RPM.
\[
\frac{7168}{2048} = 3.5 \text{ RPS} \times 60 = 210 \text{ RPM}
\]

When the Encoder is enabled, the parameters are also changed to be compatible with the 2048 pulses.
The Encoder Enabled defaults are:

- VM 30720 Pulses/Sec.
- VI 40 Pulses/Sec.
- A 40000 Pulses/Sec
- D 40000 Pulses/Sec

To enable the Encoder the program syntax is \(<EE=n>\) where \(n\) is a zero (0) or a one (1). The default is zero (0) which is Encoder disabled. To enable the Encoder, program EE=1.

Any motion will now be programmed in Encoder pulses. You can calculate the distance or velocity you need in a similar manner as done previously only with different factors.

![Figure 2.43: Quadrature Encoder Pulses](image-url)
Several Variables work in conjunction with Encoder Enable (EE). They are:

- **DB**: Encoder Deadband
- **SF**: The Stall Factor Variable
- **SM**: The Stall Detection Mode
- **ST**: Stall Flag
- **PM**: Position Maintenance

**EE - Encoder Enabled**

When the Encoder is enabled, all motion is “closed loop”. That is, motion counts are delivered from the MDrive Electronics to the motor which turns the encoder. The encoder sends counts back to the drive to complete the motion. If you programmed the MDrive to move 2048 counts, the driver would attempt to output 2048 counts provided the Stall Factor (SF) value or other fault is not encountered. If no faults were encountered, the driver would output the full 2048 counts. Depending on which variables were set, the driver would then wait until the position (plus or minus the Encoder Deadband) was read and confirmed.

**DB - Encoder Deadband**

The Encoder Deadband is a Variable that is set in Encoder Counts. Motion will be deemed complete when the Encoder Counts are within the Deadband variable. With DB=5 the motion of 2048 counts would be complete between 2043 and 2053 counts.

**SF - Stall Factor**

The Stall Factor is a Variable which is entered in Encoder Counts. The Stall Factor might be compared to the “following error” or “lag error” of a servo drive. The Stall Factor is triggered by the number of counts output from the MDrive Electronics to the motor as compared to the number of counts returned by the encoder. The comparison should always be within the value of the Stall Factor, otherwise a fault will occur and the Stall Flag (ST) will be set. If Stall Detection Mode (SM=0) is active, the motion will be stopped.

Example:

A Stall Factor of 30 counts (SF=30) is programmed. A motion command of 2048 counts is programmed. The MDrive reaches a mechanical bind at 2000 counts. The driver will keep outputting motion counts until 2030 counts (present position plus the SF value) and then the Stall Flag (ST) will be set. The MDrive will be stopped if the Stall Detection Mode (SM=0) is active.

**SM - Stall Detection Mode**

The Stall Detection Mode can be programmed to stop the MDrive (SM=0) or to allow the MDrive to continue (SM=1) when the Stall Factor (SF) is reached. Whether SM is active or not, the Stall Flag will always be set when the SF is encountered.

**ST - Stall Flag**

The Stall Flag will be set any time the SF is reached regardless of the state of the Stall Detection Mode (SM). If the Stall Flag is set, the user must reset it to zero (0).

**PM - Position Maintenance**

Position Maintenance (PM) is active only after the motion has completed. Position Maintenance is used to maintain position when there might be an external force on the drive. If Position Maintenance is enabled (PM=1) and the Stall Detection Mode is enabled (SM=0), the MDrive will be driven back to its final position if it was forced out of position provided the Stall Factor (SF) was not reached.

If Position Maintenance is enabled (PM=1) and the Stall Detection Mode is disabled (SM=1), the MDrive will be driven back to its final position if it was forced out of position regardless of whether the Stall Factor (SF) was reached or not.

There are three other variables, although not directly connected to EE, that do affect the overall operation when in Encoder Mode. They are:

- **HC**: Motor Hold Current
- **HT**: Motor Hold Current Delay Time
- **MT**: Motor Settling Delay Time

**HC - Motor Hold Current**

When motion is complete, the MDrive Electronics will switch from Motor Run Current (RC) to Motor Hold Current (HC). The Hold Current is set at a lower percentage than the Run Current (RC). However, the Hold Current must be sufficient to overcome an outside force such as an MDrive driving a vertical slide which maintains a load on the MDrive at all times. Actual Hold Current values will vary depending on the application and the load on the MDrive when it is at rest.

**HT - Motor Hold Current Delay Time**

The Motor Hold Current Delay Time (HT) is a variable that delays the change from Run Current (RC) to Hold Current (HC) at the end of a move. The end of the move is triggered by the MDrive Electronics when it has completed outputting the correct number of counts. Depending on the application, including velocity, deceleration, load and inertia, the MDrive may lag behind a few counts. The HT will allow the MDrive to finish its move before applying the lower HC.
MT - Motor Settling Delay Time

A stepping motor may ring or oscillate in minuscule amounts at the completion of a move until it satisfies the target position. The amount of this “ringing” is dependent on the application including velocity, deceleration, inertia, friction and load. The Motor Settling Delay Time (MT) allows the motor to stop “ringing” before checking the position count. If the MDrive Electronics tried to check the position count during this ringing, it would assume a position error and try to correct an already moving MDrive and possibly cause ringing of a larger magnitude and longevity. Typically, the MT is set between 50 and 100 milliseconds. It is recommended that there is always a Motor Settling Time programmed any time you are in EE=1 mode.

Note: If MT has no value, the motor may hunt and never satisfy the Position Check.

Figure 2.44: EE=1 Flowchart
**I/O Commands**

*S <1-4>*
This command configures the Type and Active state of I/O points 1-4.
Using the PR command to read IO parameters

*Read IO1 Setup – “PR S1”*
*Read IO2 Setup – “PR S2”*

Setting the I/O parameters

*Set IO 3 parameters – “S3=0,1”* Sets IO3 as a General Purpose Input, Active High

For example: To set IO4 as a Jog+ Input/Active Low
S4 =7,0

*I <1-4>*
Used to read the state of an individual input.

*PR I1 will read the state of input 1 and display it to the terminal window.*

*BR K5, I2=0 will branch to the program address labeled K5 when Input 2 is LOW*

*IN*
Used to read the decimal equivalent of the 4 bit binary number represented by all 4 inputs collectively. Note the Input 4 is the Most Significant Bit.

*PR IN will print the decimal value of the inputs.*

*O <1-4>*
Used to set the state of an output.

*O2=1 will set Output 2 TRUE*

**System Instructions**

The following System Instructions will be used frequently.

*CP*
The CP Instruction is used to clear Program memory space.

*FD*
The FD Instruction is used to return the MDrive Motion Control to its factory default state.

*<esc>*
The ESCAPE key will stop the user program and stop the motor with no decel rate.

*<control C>*
CONTROL C will reboot the unit. This includes reloading of the programs stored in nonvolatile memory into RAM and executing any programs residing at label SU (Start Up).

**Program Instructions**

*PG*
This instruction toggles the MDrive Motion Control into or out of program mode.

Switch to program mode at address 200  
PG 200

xxxxx

Program starting at address 200  
xxxxx
xxxxx

Switch out of program mode  
PG
LB
The MDrive Motion Control also offers the user the convenience of naming programs, subroutines and processes to ease in branch-
ing from one part of a program to another, or calling a subroutine.
These labels, once set, will act as pointers to locations in program memory space.
The LB, or Label Instruction, allows the user to assign a 2 character name to a program or branch process within a program or
subroutine.
The restrictions for this command are:
1] A label cannot be named after a MDrive Motion Control Instruction, Variable or Flag.
2] The first character must be alpha, the second character may be alpha-numeric.
3] A label is limited to to characters.
4] A program labeled SU will run on power-up

Please Note: Any program labeled “SU” will execute on power-up.

Switch to program mode at address 200  PG 200
Label command will name the program   LB K1
Program named by LB command             xxxx
Switch out of program mode               PG

BR
Used to branch conditionally or unconditionally to a routine.

Switch to program mode at address 200  PG 200
Label command will name the program   LB K1
Program named by LB command             xxxx
Unconditional branch to Program Label K1 BR K1
Switch out of program mode               PG

E
Designates the end of a program.

Switches to program mode at address 200  PG 200
Label command will name the program   LB K1
Program named by LB command             xxxx
Unconditional branch to Program Label K1 BR K1
Designates the end of the program        E
Switches out of program mode              PG
**H**

Delays program execution in milliseconds.

Switches to program mode at address 200  PG 200
Label command will name the program  LB K1
Program named by LB command  xxxxx

Delay 2 seconds between re-execution of program  H 2000
Unconditional branch to K1  BR K1
Designates the end of the program  E
Switches out of program mode  P

**PRINT**

Outputs specified text and parameter values to a terminal or terminal software on a Host PC.

Switches to program mode at address 200  PG 200
Label command will name the program  LB K1
Program named by LB command  xxxxx

Prints text in quotes and then POS  PR “Position = ” P
Delay 2 seconds between re-execution of program  H 2000
Unconditional branch to K1  BR K1
Designates the end of the program  E
Switches out of program mode  PG

**VAR**

Command used to define a variable with 8 alphanumeric characters.

Switches to program mode at address 200  PG 200
Define a variable named CT for Count  VR CT
Label command will name the program  LB K1
Increment CT  IC CT
Program named by LB command  xxxxx

Prints text in quotes and then POS  PR
Delay 2 seconds between re-execution of program  H 2000
Branch to K1 while CT <10  BR K1, CT <10
Designates the end of the program  E
Switches out of program mode  PG
## Setup Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>BD</td>
<td>Communications BAUD Rate</td>
<td>BAUD</td>
<td>48, 96, 19, 38, 11</td>
<td>BD=&lt;baud&gt;</td>
</tr>
<tr>
<td>CK</td>
<td>Check Sum Enable</td>
<td>-</td>
<td>-</td>
<td>CK=&lt;1/0&gt;</td>
</tr>
<tr>
<td>DE</td>
<td>Enable/Disable Drive</td>
<td>-</td>
<td>1/0</td>
<td>DE=&lt;1/0&gt;</td>
</tr>
<tr>
<td>DN</td>
<td>Device Name</td>
<td>Character</td>
<td>a-z, A-Z, 0-9</td>
<td>DN=&lt;char&gt;</td>
</tr>
<tr>
<td>EM</td>
<td>Echo Mode 0 (def)=Full Duplex, 1=Half Duplex</td>
<td>Mode</td>
<td>&lt;0..3&gt;</td>
<td>EM=&lt;mode&gt;</td>
</tr>
<tr>
<td>IP</td>
<td>Initial Parameters from EEPROM</td>
<td>-</td>
<td>-</td>
<td>IP</td>
</tr>
<tr>
<td>PY</td>
<td>Enable/Disable Party Mode</td>
<td>Mode</td>
<td>1/0</td>
<td>PY=&lt;mode&gt;</td>
</tr>
<tr>
<td>UG</td>
<td>Upgrade Firmware Code</td>
<td>Code</td>
<td>2956102</td>
<td>IMS Term. Upgrader</td>
</tr>
</tbody>
</table>

## Miscellaneous Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>All Parameters, Used with PR (Print)</td>
<td>-</td>
<td>-</td>
<td>PR AL</td>
</tr>
<tr>
<td>BY</td>
<td>BSY Flag 1=Prog. Running</td>
<td>-</td>
<td>0/1</td>
<td>PR BY</td>
</tr>
<tr>
<td>FD</td>
<td>Return to Factory Defaults</td>
<td>-</td>
<td>-</td>
<td>FD</td>
</tr>
<tr>
<td>PR</td>
<td>Print Selected Data and/or Text</td>
<td>-</td>
<td>-</td>
<td>PR &lt;data/text string&gt;</td>
</tr>
<tr>
<td>R1</td>
<td>User Register 1</td>
<td>Number</td>
<td>Signed 32 bit</td>
<td>R1=&lt;number&gt;</td>
</tr>
<tr>
<td>R2</td>
<td>User Register 2</td>
<td>Number</td>
<td>Signed 32 bit</td>
<td>R2=&lt;number&gt;</td>
</tr>
<tr>
<td>R3</td>
<td>User Register 3</td>
<td>Number</td>
<td>Signed 32 bit</td>
<td>R3=&lt;number&gt;</td>
</tr>
<tr>
<td>R4</td>
<td>User Register 4</td>
<td>Number</td>
<td>Signed 32 bit</td>
<td>R4=&lt;number&gt;</td>
</tr>
<tr>
<td>VR</td>
<td>Firmware Version</td>
<td>Number</td>
<td>-</td>
<td>PR VR</td>
</tr>
<tr>
<td>UV</td>
<td>Read User Variables</td>
<td>-</td>
<td>-</td>
<td>PR UV</td>
</tr>
</tbody>
</table>

## Motion Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-)</td>
<td>Do Previously Set Mode to/at This Value</td>
<td>per mode</td>
<td>per mode</td>
<td>&lt;number&gt;</td>
</tr>
<tr>
<td>A</td>
<td>Set Acceleration</td>
<td>Steps/Sec(^2)</td>
<td>1000000000</td>
<td>A=&lt;accel&gt;</td>
</tr>
<tr>
<td>D</td>
<td>Set Deceleration</td>
<td>Steps/Sec(^2)</td>
<td>1000000000</td>
<td>D=&lt;decel&gt;</td>
</tr>
<tr>
<td>HC</td>
<td>Set Hold Current</td>
<td>% (Percent)</td>
<td>0 to 100</td>
<td>HC=&lt;percent&gt;</td>
</tr>
<tr>
<td>HT</td>
<td>Set Hold Current Delay Time</td>
<td>milliseconds</td>
<td>0-65000</td>
<td>HT=&lt;msec&gt;</td>
</tr>
<tr>
<td>LM</td>
<td>Limit Stop Mode</td>
<td>-</td>
<td>1-6</td>
<td>LM=&lt;number&gt;</td>
</tr>
<tr>
<td>MA</td>
<td>Set Mode and Move to Abs. Position</td>
<td>±Position</td>
<td>Signed 32 bit</td>
<td>MA &lt;±pos&gt;</td>
</tr>
<tr>
<td>MD</td>
<td>Motion Mode Setting</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MR</td>
<td>Set Mode and Move to Relative Position</td>
<td>±Distance</td>
<td>Signed 32 bit</td>
<td>MR &lt;±dist&gt;</td>
</tr>
<tr>
<td>MS</td>
<td>Set Microstep Resolution</td>
<td>Microsteps/step</td>
<td>MSEL Table</td>
<td>MS=&lt;param&gt;</td>
</tr>
<tr>
<td>MT</td>
<td>Motor Settling Delay Time</td>
<td>milliseconds</td>
<td>0-65000</td>
<td>MT=&lt;msec&gt;</td>
</tr>
<tr>
<td>MV</td>
<td>Moving Flag</td>
<td>-</td>
<td>-</td>
<td>PR MV</td>
</tr>
<tr>
<td>RC</td>
<td>Set Run Current</td>
<td>% (Percent)</td>
<td>1 to 100</td>
<td>RC=&lt;percent&gt;</td>
</tr>
<tr>
<td>SL</td>
<td>Set Mode and Slew Axis</td>
<td>Steps/sec</td>
<td>±5000000</td>
<td>SL &lt;velocity&gt;</td>
</tr>
<tr>
<td>V</td>
<td>Read Current Velocity</td>
<td>Steps/sec</td>
<td>±5000000</td>
<td>PR V</td>
</tr>
<tr>
<td>VI</td>
<td>Set Initial Velocity</td>
<td>Steps/sec</td>
<td>1-5000000</td>
<td>VI=&lt;velocity&gt;</td>
</tr>
<tr>
<td>VM</td>
<td>Set Maximum Velocity</td>
<td>Steps/sec</td>
<td>1-5000000</td>
<td>VM=&lt;velocity&gt;</td>
</tr>
</tbody>
</table>
### I/O Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Set Input 1 Digital Filtering</td>
<td>Milliseconds</td>
<td>0-255</td>
<td>D1=&lt;time&gt;</td>
</tr>
<tr>
<td>D2</td>
<td>Set Input 2 Digital Filtering</td>
<td>Milliseconds</td>
<td>0-255</td>
<td>D2=&lt;time&gt;</td>
</tr>
<tr>
<td>D3</td>
<td>Set Input 3 Digital Filtering</td>
<td>Milliseconds</td>
<td>0-255</td>
<td>D3=&lt;time&gt;</td>
</tr>
<tr>
<td>D4</td>
<td>Set Input 4 Digital Filtering</td>
<td>Milliseconds</td>
<td>0-255</td>
<td>D4=&lt;time&gt;</td>
</tr>
<tr>
<td>D5</td>
<td>Set Input 5 Digital Filtering</td>
<td>Milliseconds</td>
<td>0-255</td>
<td>D5=&lt;time&gt;</td>
</tr>
<tr>
<td>I1</td>
<td>Read Input 1</td>
<td>-</td>
<td>0/1</td>
<td>PR I1, BR I1,&lt;cond&gt;</td>
</tr>
<tr>
<td>I2</td>
<td>Read Input 2</td>
<td>-</td>
<td>0/1</td>
<td>PR I2, BR I2,&lt;cond&gt;</td>
</tr>
<tr>
<td>I3</td>
<td>Read Input 3</td>
<td>-</td>
<td>0/1</td>
<td>PR I3, BR I3,&lt;cond&gt;</td>
</tr>
<tr>
<td>I4</td>
<td>Read Input 4</td>
<td>-</td>
<td>0/1</td>
<td>PR I4, BR I4,&lt;cond&gt;</td>
</tr>
<tr>
<td>I5</td>
<td>Read Input 5 (Analog)</td>
<td>-</td>
<td>0-1024</td>
<td>PR I5, BR I5,&lt;cond&gt;</td>
</tr>
<tr>
<td>I6</td>
<td>Read Encoder Index Mark Low true</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IN</td>
<td>Read Inputs 1-4 as One Value</td>
<td>data</td>
<td>0-15</td>
<td>PR IN</td>
</tr>
<tr>
<td>O1</td>
<td>Set Output 1 to Logic State</td>
<td>-</td>
<td>0/1</td>
<td>O1=&lt;1/0&gt;</td>
</tr>
<tr>
<td>O2</td>
<td>Set Output 2 to Logic State</td>
<td>-</td>
<td>0/1</td>
<td>O2=&lt;1/0&gt;</td>
</tr>
<tr>
<td>O3</td>
<td>Set Output 3 to Logic State</td>
<td>-</td>
<td>0/1</td>
<td>O3=&lt;1/0&gt;</td>
</tr>
<tr>
<td>O4</td>
<td>Set Output 4 to Logic State</td>
<td>-</td>
<td>0/1</td>
<td>O4=&lt;1/0&gt;</td>
</tr>
<tr>
<td>OT</td>
<td>Write Data to Outputs 1-4 as One Value</td>
<td>data</td>
<td>0-15</td>
<td>OT=&lt;data&gt;</td>
</tr>
<tr>
<td>S1</td>
<td>Setup IO Point 1</td>
<td>Type, Active</td>
<td>Type Table, 0/1</td>
<td>S1=&lt;type&gt;,&lt;active&gt;</td>
</tr>
<tr>
<td>S2</td>
<td>Setup IO Point 2</td>
<td>Type, Active</td>
<td>Type Table, 0/1</td>
<td>S2=&lt;type&gt;,&lt;active&gt;</td>
</tr>
<tr>
<td>S3</td>
<td>Setup IO Point 3</td>
<td>Type, Active</td>
<td>Type Table, 0/1</td>
<td>S3=&lt;type&gt;,&lt;active&gt;</td>
</tr>
<tr>
<td>S4</td>
<td>Setup IO Point 4</td>
<td>Type, Active</td>
<td>Type Table, 0/1</td>
<td>S4=&lt;type&gt;,&lt;active&gt;</td>
</tr>
<tr>
<td>TI</td>
<td>Trip on Input</td>
<td></td>
<td></td>
<td>TI &lt;input&gt;,&lt;addr&gt;</td>
</tr>
<tr>
<td>TE</td>
<td>Trip Enable</td>
<td>See Table</td>
<td>&lt;1-4&gt;</td>
<td>TE=&lt;num&gt;</td>
</tr>
</tbody>
</table>

### Program Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>Branch (Conditional/Unconditional)</td>
<td>-</td>
<td></td>
<td>BR &lt;addr&gt;, &lt;cond&gt;</td>
</tr>
<tr>
<td>CL</td>
<td>Call Subroutine (Conditional/Unconditional)</td>
<td>-</td>
<td></td>
<td>CL &lt;addr&gt;, &lt;cond&gt;</td>
</tr>
<tr>
<td>CP</td>
<td>Clear Program</td>
<td>Address</td>
<td>1-767</td>
<td>CP &lt;addr&gt;</td>
</tr>
<tr>
<td>DC</td>
<td>Decrement Variable</td>
<td>-</td>
<td></td>
<td>DC &lt;var/ureg&gt;</td>
</tr>
<tr>
<td>E</td>
<td>End Program Execution</td>
<td>-</td>
<td></td>
<td>E</td>
</tr>
<tr>
<td>EX</td>
<td>Execute Program at Address Using Selected Trace Mode</td>
<td>Address</td>
<td>1-767</td>
<td>EX &lt;addr&gt;, &lt;mode&gt;</td>
</tr>
<tr>
<td>H</td>
<td>Hold Prog. Execution Blank/0=Motion stops</td>
<td>milliseconds</td>
<td>Blank(0)/1-65000</td>
<td>H=&lt;msec&gt;</td>
</tr>
<tr>
<td>IC</td>
<td>Increment Variable</td>
<td>-</td>
<td></td>
<td>IC &lt;var&gt;</td>
</tr>
<tr>
<td>L</td>
<td>List Program</td>
<td>Address</td>
<td>1-767</td>
<td>L &lt;addr&gt;</td>
</tr>
<tr>
<td>LB</td>
<td>Create a Program Address Label Name</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OE</td>
<td>On Error Handler 0=Disabled</td>
<td>Address</td>
<td>0/1-767</td>
<td>OE &lt;addr&gt;</td>
</tr>
<tr>
<td>PG</td>
<td>Start Program Entry at Specified Address</td>
<td>-</td>
<td>Blank/1-767</td>
<td>PG &lt;addr&gt;</td>
</tr>
<tr>
<td>RT</td>
<td>Return from Subroutine</td>
<td>-</td>
<td></td>
<td>RT</td>
</tr>
<tr>
<td>S</td>
<td>Save to EEPROM</td>
<td>-</td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>VA</td>
<td>Create A User Variable Name</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UV</td>
<td>Read User Variables</td>
<td>-</td>
<td></td>
<td>PR UV</td>
</tr>
</tbody>
</table>
### Position Related Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Set Counter 1</td>
<td>Motor Counts</td>
<td>Signed 32 bit</td>
<td>C1=&lt;counts&gt;</td>
</tr>
<tr>
<td>HM</td>
<td>Home to Home Switch</td>
<td>Type</td>
<td>1-4</td>
<td>HM &lt;type&gt;</td>
</tr>
<tr>
<td>P</td>
<td>Set/Read Position</td>
<td>Motor/Encoder Counts</td>
<td>Signed 32 bit</td>
<td>P=&lt;counts&gt;</td>
</tr>
<tr>
<td>PC</td>
<td>Read Captured Position at Trip</td>
<td>Motor/Encoder Counts</td>
<td>Signed 32 bit</td>
<td>PR PC</td>
</tr>
<tr>
<td>TP</td>
<td>Trip on Position</td>
<td>Position</td>
<td>-</td>
<td>TP &lt;pos&gt;, &lt;addr&gt;</td>
</tr>
<tr>
<td>TE</td>
<td>Trip Enable</td>
<td>See Table</td>
<td>&lt;0-3&gt;</td>
<td>TE=&lt;num&gt;</td>
</tr>
</tbody>
</table>

### Encoder Related Instructions, Variables and Flags

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Function</th>
<th>Unit</th>
<th>Range</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>Set Counter 2</td>
<td>Encoder Counts</td>
<td>Signed 32 bit</td>
<td>C2=&lt;counts&gt;</td>
</tr>
<tr>
<td>DB</td>
<td>Set Encoder Deadband</td>
<td>Encoder Counts</td>
<td>0-65000</td>
<td>DB=&lt;counts&gt;</td>
</tr>
<tr>
<td>EE</td>
<td>Enable/Disable Encoder Functions</td>
<td>-</td>
<td>1/0</td>
<td>EE=&lt;1/0&gt;</td>
</tr>
<tr>
<td>HI</td>
<td>Home to Encoder Index</td>
<td>Type</td>
<td>1-4</td>
<td>HI=&lt;type&gt;</td>
</tr>
<tr>
<td>I6</td>
<td>Read Encoder Index Mark</td>
<td>-</td>
<td>I6</td>
<td></td>
</tr>
<tr>
<td>SF</td>
<td>Set Stall Factor</td>
<td>Encoder Counts</td>
<td>0-65000</td>
<td>SF=&lt;counts&gt;</td>
</tr>
<tr>
<td>SM</td>
<td>Set Stall Mode</td>
<td>0=Stop Motor/1=Don’t Stop</td>
<td>1/0</td>
<td>SM=&lt;mode&gt;</td>
</tr>
<tr>
<td>ST</td>
<td>Stall Flag</td>
<td>-</td>
<td>0/1</td>
<td>PR ST</td>
</tr>
</tbody>
</table>

### Mathematical Functions

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add Two Variables and/or Flags</td>
</tr>
<tr>
<td>-</td>
<td>Subtract Two Variables and/or Flags</td>
</tr>
<tr>
<td>*</td>
<td>Multiply Two Variables and/or Flags</td>
</tr>
<tr>
<td>/</td>
<td>Divide Two Variables and/or Flags</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Not Equal</td>
</tr>
<tr>
<td>=</td>
<td>Equal</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less Than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less Than and/or Equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater Than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater Than and/or Equal</td>
</tr>
<tr>
<td>&amp;</td>
<td>AND (Bitwise)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>^</td>
<td>XOR (Bitwise)</td>
</tr>
<tr>
<td>!</td>
<td>NOT (Bitwise)</td>
</tr>
</tbody>
</table>
**Section 2.4**

**MDrive Motion Control Command Set**

**MNEMONIC**  
A  
**FUNCTION**  
Acceleration  
**TYPE**  
Motion Variable  
**USAGE**  
P/I R/W  

**DESCRIPTION**  
The A Variable sets the peak acceleration that will be reached by the MDrive in steps per second\(^2\). That is, steps per second, per second. If the A was set at 76800 microsteps per second\(^2\) the motor would accelerate at a rate of 76800 microsteps per second, every second. If the maximum velocity was set at 768000 microsteps per second it would take 10 seconds to reach maximum speed.

**USE**  
A=<accl>  
**UNITS**  
Steps/sec\(^2\)  
**RANGE**  
0 to 1525878997  
**DEFAULT**  
1000000

**EXAMPLE:**

- A=20000  
  ‘set acceleration to 20000 steps/sec\(^2\)
- A=D  
  ‘set acceleration equal to deceleration

**RELATED COMMANDS:** D

**MNEMONIC**  
AL  
**FUNCTION**  
Retrieve All Parameters  
**TYPE**  
Variable  
**USAGE**  
I R

**DESCRIPTION**  
The AL variable is used with the PR (PRINT) instruction to print the value/state of all variables and flags to the terminal program.

**USE**  
PR AL

**RELATED COMMANDS:** PR

**MNEMONIC**  
BD  
**FUNCTION**  
BAUD Rate  
**TYPE**  
Setup Variable  
**USAGE**  
P/I R/W

**DESCRIPTION**  
This variable sets the baud rate for serial communications with the MDrive. It sets the rate for the RS-485 interface. The baud rate is set by indicating the first two digits of the desired rate as shown in the range section below.

In order for the new BAUD rate to take effect, the user must issue the S (SAVE) instruction and then reset the MDrive. When the MDrive is reset, it will communicate at the new BAUD rate.

48 = 4800 bps, 96 = 9600 bps, 19 = 19200 bps, 38 = 38000 bps, 11 = 115200 bps

**NOTE:** If you change the Baud Rate in the MDrive it must be matched in IMS Terminal.

**USE**  
BD=<baud>  
**UNITS**  
bits per second  
**RANGE**  
48, 96, 19, 38, 11  
**DEFAULT**  
9600 bps

**EXAMPLE:**

- BD=96  
  ‘set communications BAUD rate to 9600 bps

**RELATED COMMANDS:** CK
The branch instruction can be used to perform a conditional or unconditional branch to a routine in an MDrive program. It can also be used to perform loops and IF THEN logic within a program.

There are two parameters to a branch instruction. These are used to perform two types of branches:

### Conditional Branch
This type of branch first specifies an address or process label where program execution should continue if the second parameter, the condition, is true. The condition parameter may include flags as well as logical functions that are to be evaluated.

### Unconditional Branch
In this type of branch the second parameter is not specified, then the execution will continue at the address specified by the first parameter.

**USE**

```
BR <addr/label, cond>
```

**EXAMPLE:**

- BR 256, I2 'Branch to program line 256 if Input 2 is TRUE
- BR 120 'Unconditional Branch to program line 120
- BR JC, I1=1 'Branch to process labeled JC if input 1 is True

**RELATED COMMANDS:** —

---

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BY</td>
<td>Busy Flag (Read Only)</td>
<td>Busy Flag</td>
<td>P/I R</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
This read only status flag will indicate whether or not a Program is executing.

**USE**

```
PR BY
```

**EXAMPLE:**

```
PR BY
```

' read the state of the busy flag

**RELATED COMMANDS:** PR

---

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Counter 1 (Motor Counts)</td>
<td>Motion Variable</td>
<td>P/I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
This variable contains the raw count representation of the clock pulses sent to the MDrive. Counter 1 may be preset if necessary.

**USAGE**

```
C1=<counts>
```

**EXAMPLE:**

```
C1=20000
```

' Set Counter 1 to 20000 motor counts

```
PR C1
```

' Print the value of C1 to the terminal screen

**RELATED COMMANDS:** C2, P
**MNEMONIC FUNCTION TYPE USAGE**

**CL Call Subroutine Program Instruction P**

**DESCRIPTION**
This function can be used to invoke a subroutine within a program. This allows the user to segment code and call a subroutine from a number of places rather than repeating code within a program.

There are two parameters to the CL instruction. The first specifies the program address or label of the subroutine to be invoked if the second parameter, the condition, is true. If the second parameter is not specified, the subroutine specified by the first parameter is always invoked. The condition parameter can include flags as well as logical functions that are to be evaluated.

The subroutine should end with a RT (RET) instruction. The RT instruction will cause program execution to return to the line following the CL instruction.

**USE**

| CL <addr/label, cond> |

**EXAMPLE:**

- CL 256, I1=1 'Call subroutine at program line 256 if Input 1 is TRUE
- CL JK 'Call subroutine labeled JK

**RELATED COMMANDS:** RT
### MNEMONIC FUNCTION TYPE USAGE

#### D Deceleration Motion Variable

**TYPE**: Motion Variable

**USAGE**: P/I R/W

**DESCRIPTION**

The D variable sets the peak deceleration of the MDrive in steps per second\(^2\). That is, steps per second, per second. If the D was set at 76800 microsteps per second\(^2\) the motor would decelerate at a rate of 76800 microsteps per second, every second. If the MDrive was running at a maximum velocity of 768000 microsteps per second it would take 10 seconds to decelerate.

**USE**

\[
D = \langle \text{decl} \rangle \\
\text{Units: Steps/sec}^2 \\
\text{Range: 0 to 1525878997} \\
\text{Default: 1000000}
\]

**EXAMPLE**:

- \(D = 20000\) 'set acceleration to 20000 step/sec\(^2\)
- \(D = A\) 'set deceleration equal to acceleration

**RELATED COMMANDS**: A,

#### D1-D5 Digital Input Filtering

**TYPE**: I/O Variable

**USAGE**: P/I R/W

**DESCRIPTION**

This variable will set the digital filtering to be applied to the selected input 1 - 5. The input must be stable for “time” amount of milliseconds before a change in state is available.

**USE**

\[
D <1-5> = \langle \text{time} \rangle \\
\text{Units: Milliseconds} \\
\text{Range: 0 to 255}
\]

**EXAMPLE**:

- \(D1 = 0\) 'no debounce
- \(D4 = 150\) '150 mSec of filtering

**RELATED COMMANDS**: I1-I5

---

**USAGE**: P = Program  I = Immediate  R = Read  W = Write

---
**MNEMONIC**

<table>
<thead>
<tr>
<th>DB</th>
<th>Encoder Deadband</th>
<th>Setup Variable</th>
<th>USAGE</th>
</tr>
</thead>
</table>

**DESCRIPTION**

This variable defines the plus (+) and minus (-) length of the encoder deadband in encoder counts. When the encoder is enabled, a move is not completed until motion stops within DB.

**USE**

<table>
<thead>
<tr>
<th>UNITS</th>
<th>RANGE</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB=&lt;counts&gt;</td>
<td>0 to 65000</td>
<td>1</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
DB=5  # Set Encoder Deadband to ± 5 encoder counts
```

**RELATED COMMANDS:** EE, C2, SF, SM, ST, PM

**MNEMONIC**

| DC  | Decrement Variable | Program Instruction | P/I |

**DESCRIPTION**

The DC instruction will decrement the specified variable by one.

**USE**

```
DC <var>
```

**EXAMPLE:**

```
DC R1  # Decrement User Register 1
```

**RELATED COMMANDS:** IC

**MNEMONIC**

| DE  | Drive Enable Flag | Setup Flag | P/I | R/W |

**DESCRIPTION**

The DE flag enables or disables the drive portion of the MDrive Motion Control.

**USE**

```
DE= <0/1>  # 1 (Enabled)
```

**EXAMPLE:**

```
DE=0  # Disable drive
DE=1  # Enable drive
```

**RELATED COMMANDS:** —

**MNEMONIC**

| DN  | Device Name | Setup Variable | P/I | R/W |

**DESCRIPTION**

The DN Variable stores the device name to be used when the MDrive is to be addressed in party mode operation. The name is only used when party mode communications is being used (PY = 1). All Mdrive system nodes will respond if the name in a command is given as “*”. When the name is changed it must be saved into the nonvolatile memory if it is to be used in later sessions without being changed again.

See Appendix A: ASCII table for decimal codes.

**USE**

```
DN=<“char”>  # ASCII Characters
```

**EXAMPLE:**

```
DN=“A” or 65  # Set the device name to the character A
```

**RELATED COMMANDS:** PY, S
### Encoder Enable Flag (EE)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>End Program Execution</td>
<td>Program Instruction</td>
<td>P</td>
</tr>
</tbody>
</table>

#### DESCRIPTION
The EE flag enables or disables the optional encoder mode of the MDrive Motion Control. When in Encoder Mode, all programming is done by Encoder Pulses. The 512 line Encoder generates pulses in a Quadrature format which results in 2048 pulses per revolution. (See Programming With the Optional Encoder Enabled in Section 2.2.)

#### USE

| EE= <0/1> | 0 (Disabled) |

#### EXAMPLE:

- `EE=0` 'Disable encoder mode
- `EE=1` 'Enable encoder mode

#### RELATED COMMANDS: DB, C2, SF, SM, ST, PM

### Error Flag (EF)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Encoder Enable Flag</td>
<td>Setup Flag</td>
<td>P/I R/W</td>
</tr>
</tbody>
</table>

#### DESCRIPTION
The Error flag will indicate whether or not an error condition exists. It is automatically cleared when a new program is executed. The only way to manually clear the EF flag is to read the value of the ER variable or set ER=0. There is an instruction, OE, which allows the user to specify the execution of a subroutine in the program memory when an error occurs. The subroutine might contain instructions to read the ER variable which would clear the EF flag.

#### USE

<table>
<thead>
<tr>
<th>PR EF</th>
<th>0 = No Error Exists</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 = Error Condition Exists</td>
</tr>
</tbody>
</table>

#### EXAMPLE:

- `PR EF` 'read the state of the error flag

#### RELATED COMMANDS: ER, OE

---

**USAGE:**  
- **P** = Program  
- **I** = Immediate  
- **R** = Read  
- **W** = Write
### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM</td>
<td>Echo Mode Flag</td>
<td>Setup Flag</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

The Echo Mode Flag will set the full/half duplex configuration of the RS-485 channel. 0=Full Duplex (default), 1=Half Duplex.

**USE**

<table>
<thead>
<tr>
<th>USE</th>
<th>DEFAULT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EM=&lt;0-3&gt;</td>
<td>0 (Full Duplex)</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- EM=0: 'Echo all information back over communications line. CR/LF Indicates Command Accepted (Full Duplex).
- EM=1: 'Don’t echo the information, only send back prompt. CR/LF Indicates Command Accepted (Half Duplex).
- EM=2: 'Does not send back prompt, only responds to PRINT and LIST commands.
- EM=3: 'Saves Echo in Print Queue then executes Command. Prints after execution.

**RELATED COMMANDS:** BD

### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER</td>
<td>Error Number Variable</td>
<td>Status Variable</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

The ER variable indicates the program error code for the most recent error that has occurred in the MDrive Motion Control. The ER variable must be read or set to zero to clear the EF flag.

See Appendix A of this document for a complete listing of MDrive Motion Control Error Codes.

**USE**

<table>
<thead>
<tr>
<th>USE</th>
<th>RESPONSE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PR ER</td>
<td>&lt;numerical error code&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- PR ER 'read the error number

**RELATED COMMANDS:** EF, OE

### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EX</td>
<td>Execute Program</td>
<td>Program Instruction</td>
<td>I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

Execute program at a specified address or label using a selected trace mode. Used in immediate mode.

There are three modes of program execution:

- **Mode 0** Normal execution, is specified by a mode of 0 (or simply leaving the mode blank).
- **Mode 1** Trace mode is specified by a mode of 1. This means that the program executes continuously until the program END is encountered, but the instructions are “traced” to the communications port so the user can see what instructions have been executed.
- **Mode 2** Single step mode is specified by a mode of 2. In this mode, the user can step through the program using the space bar to execute the next line of the program. The program can be resumed at normal speed in this mode by pressing the enter key.

**USE**

<table>
<thead>
<tr>
<th>USE</th>
<th>MODES</th>
<th></th>
</tr>
</thead>
</table>
| EX <addr/label>,<mode> | <mode> = 0: run program normally
|                   | <mode> = 1: run program in trace mode
|                   | <mode> = 2: run program in single-step mode |

**EXAMPLE:**

- EX 127 'execute program at line 127 normally
- EX 127,1 'execute program at line 127 in trace mode

**RELATED COMMANDS:** PG, E
### MNEMONIC FUNCTION TYPE USAGE

#### H Hold Program Execution

**TYPE**: Program Instruction  
**USAGE**: P

**DESCRIPTION**
The hold instruction is used in a program to suspend program execution. If no parameter is specified the execution of the program will be suspended while motion is in progress. This will typically be used following a MA, MR, HI or HM instruction.

A time in milliseconds may be placed as a parameter to the hold instruction, this will suspend program execution for the specified number of milliseconds.

**USE**

- **H <time>**  
  - Blank or 0 - hold while moving, 1 - 65000 Milliseconds.

**EXAMPLE:**

- **example 1**
  
  - MA 20000  
  - ’move absolute 20000 motor units
  - H  
  - ’hold program execution until motion completes
  - MA -20000  
  - ’move absolute -20000 motor units
  - H  
  - ’hold program execution until motion completes

- **example 2**
  
  - O2=1  
  - ’set output 2 HIGH
  - H 1000  
  - ’hold 1 second (1000 Milliseconds)
  - O2=0  
  - ’set output 2 LOW

**RELATED COMMANDS:** PG, E

### MNEMONIC FUNCTION TYPE USAGE

#### HC Hold Current

**TYPE**: Setup Variable  
**USAGE**: P/I R/W

**DESCRIPTION**
This variable defines the motor holding current in percent.

**USE**

- **HC=<PERCENT>**
  
  - UNITS: PERCENT
  - RANGE: 0 TO 100
  - DEFAULT: 5

**EXAMPLE:**

- **HC=5**  
  - ’Set motor holding current to 5%

**RELATED COMMANDS:** HT, RC
### MNEMONIC FUNCTION TYPE USAGE

#### HM Home to Home Switch Motion Instruction P/I

**DESCRIPTION**

This instruction will find the selected I/O switch assigned to “Home”.

1) **Speed (S):** Specifies the direction and speed that the axis will move until the switch is activated (VM).
2) **Creep (C):** Specifies the direction and speed that the axis will move off the switch until it becomes inactive again (VI).

When HM is executed, the axis moves in the direction specified by the (S) at VM until it reaches the index mark. It then creeps off of the index in the direction specified by the sign of (C) at VI. Motion is stopped as soon as the switch becomes deactivated.

The diagram on the following page illustrates the different scenarios possible during the Homing (HM) sequence. The diagrams represent the four HM commands. Below are the four combinations of the HM command.

**USE TYPES**

HM=<type>  
1: S- C+,  2: S- C-,  3: S+ C-,  4: S+ C+

**EXAMPLE:**

HM=3  ‘Find home switch at VM in the plus direction, Creep off at VI in the minus direction

**RELATED COMMANDS:** VM, VI, S1-S4, HI

---

### MNEMONIC FUNCTION TYPE USAGE

#### HI Home to Index Mark Motion Instruction P/I

**DESCRIPTION**

This instruction will find the the encoder index mark. There are four combinations for this command. (See Use below.)

1) **Speed (S):** Specifies the direction and speed that the axis will move until the index mark is found (VM).
2) **Creep (C):** Specifies the direction and speed that the axis will move off the switch until it becomes inactive again (VI).

When HI is executed, the axis moves in the direction specified by the (S) at VM until it reaches the index mark. It then creeps off of the index in the direction specified by the sign of (C) at VI. Motion is stopped as soon as the index changes state.

**USE TYPES**

HI=<type>  
1: S- C+,  2: S- C-,  3: S+ C-,  4: S+ C+

**EXAMPLE:**

HI=2  ‘Find index mark at VM in the minus direction, Creep off at VI in the minus direction

**RELATED COMMANDS:** VM, VI, EE, I6, HM

---

**USAGE:**  
P = Program  
I = Immediate  
R = Read  
W = Write
Home Switch True False
Home Switch Range

H1: Slew – and Creep +

Deceleration Rate too long. Home Switch is overshot. Motion must reverse to find Home Switch and stop.

H2: Slew – and Creep –

Deceleration Rate too long. Home Switch is overshot. Motion must reverse to find Home Switch and stop.

H3: Slew + and Creep –

Deceleration Rate is short enough so that motion can stop within the range of the Home Switch.

H4: Slew + and Creep +

Deceleration Rate is short enough so that motion can stop within the range of the Home Switch.
### MNEMONIC FUNCTION TYPE USAGE

#### HT
**Hold Current Delay Time**
**Setup Variable**
**P/I**

**DESCRIPTION**
The HT variable sets the delay time in milliseconds between the cessation of motion and when the MDrive Motion Control shifts to the holding current level specified by the HC (Motor Holding Current) variable. The delay time is also effected by the MT (Motor Settling Delay Time) variable in that the total time from motion ceasing to current change is represented by the sum of MT + HT.

**USE**

<table>
<thead>
<tr>
<th>USE</th>
<th>UNITS</th>
<th>RANGE</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>HT=&lt;time&gt;</td>
<td>milliseconds</td>
<td>0 to 65000</td>
<td>500</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

HT=1500  ‘Set hold current delay time to 1.5 seconds

**RELATED COMMANDS:** HC, MT, RC

#### I1 - I4
**Read Input**
**I/O Variable**
**P/I**

**DESCRIPTION**
This variable will read the state of the specified input 1 - 4. Can be used with PR (Print), BR (Branch) and CL (Call Subroutine) instructions. Can also be used with R1-R4 and User Variables.

The value of the bit state will be dependant on active (low/high) state of the input, specified by the S<1-4> variable.

**USE**

<table>
<thead>
<tr>
<th>USE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR I&lt;1-4&gt;</td>
</tr>
<tr>
<td>BR &lt;addr&gt;, I&lt;1-4&gt;=&lt;1/0&gt;</td>
</tr>
<tr>
<td>CL &lt;addr&gt;, I&lt;1-4&gt;=&lt;1/0&gt;</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

PR I2  ‘Print the state of Input 2 to the Terminal Screen
BR 128, I3=1  ‘Conditional branch to program line 128 if Input 3 = 1
CL 432, I4=0  ‘Call subroutine at line 432 if Input 4 = 0

**RELATED COMMANDS:** IN, O1-O4, PR, S1-S4
### I5 Read Analog Input

**DESCRIPTION**
This variable will read the value of the voltage seen on the Analog Input. Can be used with PR (Print), BR (Branch) and Cl (Call Subroutine) instructions. The value read will be between 0 and 1028.

**USE**

- PR I5
- BR <addr/label>, I5=<0 - 1028>
- CL <addr/label>, I5=<0 - 1028>

**EXAMPLE:**
- PR I5  "Print the value of the Analog Input to the Terminal Screen"
- BR K1, I5=512  "Branch to Program labeled K1 if Analog Input = 512"
- CL 432, I5=0  "Call subroutine at line 432 if Analog Input = 0"

**RELATED COMMANDS:** BR, CL, PR

### I6 Read Encoder Index Mark

**DESCRIPTION**
This variable will read the on/off state of the Encoder Index Mark. Can be used with PR (Print), BR (Branch) and Cl (Call Subroutine) instructions. The value read will be 0 (off mark) or 1 (on mark).

**USE**

- PR I6
- BR <addr/label>, I6=<0/1>
- CL <addr/label>, I6=<0/1>

**EXAMPLE:**
- PR I6  "Print the on/off state of the encoder index mark"
- BR K1, I6  "Branch to Program labeled K1 if encoder index mark is TRUE"
- CL 432, I6=0  "Call subroutine at line 432 if I6=0"

**RELATED COMMANDS:** BR, CL, PR
MNEMONIC FUNCTION TYPE USAGE

**IN** Read Inputs 1-4 As 1 Value I/O Variable P/I R

**DESCRIPTION**
This keyword will read the binary state (unprocessed by S1-4) of inputs 1-4 and print them as a decimal value. When used thus, Input 1 is the Least Significant Bit (LSb) and Input 4 is the Most Significant Bit (MSb). It may be used in conjunction with the R1-R4 (User Registers), PR (Print), BR (Branch) and Cl (Call Subroutine) instructions. The value is a function of the actual state of the IO where 1 = +V and 0 = Ground. (Not a function of the active state defined in S1 to S4 variables).

**USE**
PR IN
BR IN=<0-15>
CL IN=<0-15>

**EXAMPLE:**
PR IN 'Print the state of Inputs 1-4 to the Terminal Screen
BR 128, IN=8 'Conditional branch to program line 128 if the binary state of Inputs 1-4 = 8 (1000)
CL 432, IN=13 'Call subroutine at line 432 if the binary state of Inputs 1-4 = 13 (1101)

**RELATED COMMANDS:** I1-5, O1-05, PR, S1-S4

---

**IC**
**DESCRIPTION**
The IC instruction will increment the specified variable by one.

**USE**
IC <var>

**EXAMPLE:**
IC R4 'Increment User Register 4

**RELATED COMMANDS:** IC

---

**IP** Initialize Parameters Instruction P/I

**DESCRIPTION**
The IP instruction will return all of the MDrive Motion Control parameters to their stored values.

**USE**
IP

**EXAMPLE:**
IP

**RELATED COMMANDS:** CP, S
### MNEMONIC FUNCTION TYPE USAGE

**JE**

**Description:**
The will enable Jog Mode if IO are set for Jog Plus and/or Jog Minus. States are 0=Disabled, 1=Enabled.

**Use:**

```
JE= <0/1>
```

**Default:**

0 (Disabled)

**Example:**

JE=0  "Disable Jog Mode"

JE=1  "Enable Jog Mode"

**Related Commands:** I5

---

### MNEMONIC FUNCTION TYPE USAGE

**L**

**Description:**
The L instruction will print the contents of program space beginning at the specified address to the end. If no address is specified it will list beginning at line 1.

**Use:**

```
L
L <addr/label>
```

**Example:**

L 128  "display contents of program space beginning at line 128"

**Related Commands:** —

---

### MNEMONIC FUNCTION TYPE USAGE

**LB**

**Description:**
The LB, or Label Instruction, allows the user to assign a 2 character name to a program, branch process within a program or subroutine.

The restrictions for this command are:

1] A label cannot be named after a MDrive Motion Control Instruction, Variable or Flag.
2] The first character must be alpha, the second character may be alpha-numeric.
3] A label is limited to two characters.
4] A program labeled SU will run on power-up

**Use:**

```
LB <char><char>
```

**Example:**

PG 100  "start program at address 100"

LB J0  "Label program J0"

**Related Commands:** BR, CL, EX, TI, TP, L, CP

---

**Usage:**

- **P** = Program
- **I** = Immediate
- **R** = Read
- **W** = Write
### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>LK</td>
<td>Lock User Program</td>
<td>Setup Flag</td>
<td>I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

This flag allows the user to lock the program from being listed or modified. It can only be reset by clearing the entire program space: CP (no address). If CP addr, L (addr) or PG addr are entered, then error 44 (Program Locked) will be set and nothing else will happen.

To clear LK, don’t save (S) then do a Ctrl-C or Cycle Power and the LK will be reset to previous unlocked state. (Program is automatically stored in NVM as it is entered.) Or you may clear program (CP). This will clear the program and reset LK to 0 in NVM as well as in local ram.

**USE**

| LK = <0/1> |

**DEFAULT**

| 0 (Disabled) |

**RELATED COMMANDS:** CP, L

---

### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM</td>
<td>Limit Stop Mode</td>
<td>Motion Variable</td>
<td>R/I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

The LM variable specifies the Limit Stop Mode for the MDrive. There are six LM modes. They are as follows.

- **LM=1** Normal Limit function with a decel ramp.
  
  The I/O must be set for Limits. If the limit switch in the direction of travel is reached, the motion will decel to a stop. That is, the plus limit works only in the plus direction of travel and the minus limit works only in the minus direction of travel.
  
  In the illustration below, the Limit is activated at a given position but because of the deceleration rate the motion continues for the duration of the deceleration time. This position may be beyond the trip point of the limit and a subsequent move in the same direction will not stop. A crash may be imminent.
  
  If the limit is activated and maintained the software will allow motion only in the opposite direction.
  
  If Homing (HM) is active and a limit is reached, the motion will decel to a stop and then reverse direction and seek the Homing Switch. If the Homing Switch is not activated on the reverse and the opposite limit is reached all motion will stop with a decel ramp. (See HM)
  
  It is possible for the Home Switch to be overshot because of long decel time. Whenever the Homing sequence is activated, a short decel time should be used.

- **LM=2** A Limit stops all motion with a decel ramp but no Homing.

- **LM=3** A Limit will stop all motion with a decel ramp and stop program execution.

- **LM=4** Functions as LM=1 but with no deceleration ramp.

- **LM=5** Functions as LM=2 but with no deceleration ramp.

- **LM=6** Functions as LM=3 but with no deceleration ramp.

**USE**

| LM = <1-6> |

**DEFAULT**

| 1 |

**RELATED COMMANDS:** HI, HM, JE, MA, MR, SL
### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MA</strong></td>
<td>Move To Absolute Position</td>
<td>Motion Instruction</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
Set mode for absolute move and move to an absolute position relative to (0) zero. MD (Current Mode) will be set to MA. If flag is true, then DN will be sent out when move is complete.

**USE**
MA ±pos, <flag>  

**EXAMPLE:**
- MA 51200  
  'move motor to absolute position 51200 in positive direction
- MA -51200  
  'move motor to absolute position 51200 in negative direction

**RELATED COMMANDS:** MD, MR, MS, P

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MD</strong></td>
<td>Motion Mode</td>
<td>Motion Variable</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
Indicates what the last motion command was, so that when just a number is entered, then it will read MD to define the new motion.

**USE**
MD

**EXAMPLE:**
- MA 200000  
  'move absolute 200000 steps, set current mode to MA
- -200000  
  'move absolute -200000 steps
- MR 1000000  
  'move relative 1000000 steps, set current mode to MR
- -1000000  
  'move relative -1000000 steps
- SL 20000  
  'slew 20000 steps/sec. set current mode to SL
- -10000  
  'slew 10000 steps/sec in minus direction
- PR MD  
  return current mode setting

**RELATED COMMANDS:** MD, MR, MS, P, PR, SL

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MR</strong></td>
<td>Move To Relative Position</td>
<td>Motion Instruction</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
Set mode for relative move and move a relative distance. MD (Current Mode) will be set to MR If flag is true, then DN will be sent out when move is complete.

**USE**
MR ±distance, <flag>  

**EXAMPLE:**
- MR 200000  
  'move motor 200000 motor counts positive direction
- MR -50000  
  'move motor 50000 motor counts in a negative direction

**RELATED COMMANDS:** MD, MA, MS, P
### Microstep Resolution (MS)

<table>
<thead>
<tr>
<th>MS (Microsteps/Step)</th>
<th>Steps/Rev</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>400</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
</tr>
<tr>
<td>8</td>
<td>1,600</td>
</tr>
<tr>
<td>16</td>
<td>3,200</td>
</tr>
<tr>
<td>32</td>
<td>6,400</td>
</tr>
<tr>
<td>64</td>
<td>12,800</td>
</tr>
<tr>
<td>128</td>
<td>25,600</td>
</tr>
<tr>
<td>256</td>
<td>51,200</td>
</tr>
</tbody>
</table>

**Table 2.10: Microstep Resolution Settings**

**Decimals Microstep Resolution Settings**

<table>
<thead>
<tr>
<th>MS (Microsteps/Step)</th>
<th>Steps/Rev</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1,000</td>
</tr>
<tr>
<td>10</td>
<td>2,000</td>
</tr>
<tr>
<td>25</td>
<td>5,000</td>
</tr>
<tr>
<td>50</td>
<td>10,000</td>
</tr>
<tr>
<td>125</td>
<td>25,000</td>
</tr>
<tr>
<td>250</td>
<td>50,000</td>
</tr>
</tbody>
</table>

### Motor Settling Delay Time (MT)

<table>
<thead>
<tr>
<th>USE</th>
<th>UNITS</th>
<th>RANGE</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT</td>
<td>time</td>
<td>0 to 65000</td>
<td>0</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- MT=50  'Set motor settling delay to 50 milliseconds

**RELATED COMMANDS:** HC, HT, RC

---

**DESCRIPTION**

The MS variable controls the microstep resolution of the MDrive Motion Control. There are 14 parameters that can be used with this variable, 8 binary and 6 decimal. The table below illustrates the parameter settings and their associated resolutions for the 1.8° stepping motor used with the MDrive Motion Control.

The MS parameters given in the table below are the only valid parameters that will be accepted by the MDrive.

**USAGE**

- MS=<parameter>

**DEFAULT**

- 256

**EXAMPLE:**

- MS=4  'Set Microstep Resolution to 4 Microsteps/Step (800 Steps/Rev)
- MS=50 'Set Microstep Resolution to 50 Microsteps/Step (10000 Steps/Rev)
- PR MS  'Print the MS setting to the terminal
### ON ERROR Function

**Usage:**

```
OE <address>
```

**Example:**

```
OE E1 'On Error go to E1
LB E1 'label subroutine E1
O3=1 'Set Output 3 to Logic 1
RT 'Return from subroutine
E 'End program
PG 'Return to immediate mode
```

**Related Commands:** EF, ER

### Set Output Logic State

**Mnemonic:** O1 - O4

**Function:** Set Output Logic State

**Type:** I/O Variable

**Usage:**

```
O<1-4>=<0/1>
```

**Example:**

```
O4=1 'Set Output 4 to 1
```

**Related Commands:** OT, I1-I4, PR, S1-S4

### Moving Flag

**Mnemonic:** MV

**Function:** Moving Flag

**Type:** Moving Flag

**Usage:**

```
P/I R
```

**Description:** Output is true when motor is moving.

**Use:**

```
PR MV
```

**Example:**

```
PR MV^M
```

**Related Commands:**
<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OT</td>
<td>Set Outputs 1-4 As 1 Value</td>
<td>I/O Variable</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

The OT variable allows the user to set Outputs 1-4 (unprocessed by S<1-4>) as one 4 bit binary value. The value is entered in decimal, with a range of 0-15 in binary where Output 1 will be the LSb and Output 4 will be the MSb.

Example: OT=12
- Output 4 = 1
- Output 3 = 1
- Output 2 = 0
- Output 1 = 0

**USE**

<table>
<thead>
<tr>
<th>UNITS</th>
<th>RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>P &lt;position&gt;</td>
<td>Steps</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- PR OT

**RELATED COMMANDS:** I1-I4, S1-S4

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>Position Counter</td>
<td>Motion Variable</td>
<td>P/I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

This instruction is used to set or print the value of the MDrive Motion Control position counter. The position will read in Motor Counts from C1 (Counter 1) by default, if encoder functions are enabled, the position counter will read in Encoder Counts from C2 (Counter 2).

The main difference in the relationship of the two counters is that where C1 is variable, the value of each count in terms of distance moved is based upon the MS, or microstep resolution setting, C2 will always be 2048 counts per motor revolution, regardless of the microstep resolution setting.

Modifying P in essence changes the frame of reference for the axis. P will probably be set once during system set up to reference or “home” the system.

**USE**

<table>
<thead>
<tr>
<th>UNITS</th>
<th>RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>P &lt;position&gt;</td>
<td>Steps</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- P=0 ‘Clear position counter, set to 0
- PR P ‘Print the state of the position counter

**RELATED COMMANDS:** C1, C2

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>Position Capture At Trip</td>
<td>Program Instruction</td>
<td>I</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

Captures motor or encoder position at activation.

**USE**

<table>
<thead>
<tr>
<th>UNITS</th>
<th>RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>Motor/Encoder Counts</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

- PR PC^M ‘Display captured position

**RELATED COMMANDS:**

**USAGE:**

- P = Program
- I = Immediate
- R = Read
- W = Write
**MNEMONIC FUNCTION TYPE USAGE**

**PM Postion Maintenance Enable Setup Flag P/I R/W**

**DESCRIPTION**
This flag will enable the position maintenance functions of an MDrive Motion Control with encoder. The position maintenance velocity will be at the setting for VI (Initial Velocity).

If SM = 0 and PM = 1, Position Maintenance will take place provided the position does not exceed the Stall Factor (SF).

If SM = 1 and PM = 1, Position Maintenance will take place even if the Stall Factor (SF) is exceeded, unless VI is set too high causing the motor to stall.

**USE**
PM = <0/1> 0 (Disabled)

**EXAMPLE:**
PM=0 'Position Maintenance Disabled (Default)
PM=1 'Position Maintenance Enabled

**RELATED COMMANDS:** VI, EE, SM, DB, C2, SF

**USE:**
PG <address>

**EXAMPLE:**
PG 100 'Enter program mode, start program at address 100

**********PROGRAM**********
E 'End program
PG 'Exit program, return to immediate mode

**RELATED COMMANDS:** E,

---

**MNEMONIC FUNCTION TYPE USAGE**

**PR Print Selected Data/Text Instruction P/I**

**DESCRIPTION**
This instruction is used to output text and parameter value(s) to the host PC. Text should be enclosed in quotation marks while parameters (variables and flags) should not. Text strings and parameters which are to be output by the same PR instruction should be separated by commas. The information being output is followed by a carriage return and line feed unless a semicolon (;) is included at the end of the PR instruction to indicate that the cursor should remain on the same line.

It is important to note that the receive buffer for the MDrive Motion Control is 64 characters, this includes the PR instruction itself, any spaces, text characters, etc. If the buffer length is exceeded ASCII code “0xFF” will echo to the terminal screen.

**USE**
PR <data/text>

**EXAMPLE:**
PR "Position =", P'print axis position, 18 characters used
the terminal will display: Position = 1234567

**RELATED COMMANDS:** —

**USAGE:** P = Program I = Immediate R = Read W = Write
### MNEMONIC FUNCTION TYPE USAGE

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PS</td>
<td>Pause Program Instruction</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>PY</td>
<td>Party Mode Enable Flag</td>
<td>P/I</td>
<td>R/W</td>
</tr>
</tbody>
</table>

#### DESCRIPTION

**PS**

This instruction is used to pause an executing program and invoke normal deceleration of any motion being executed to Zero. Immediate mode instructions are allowed while a program is in a paused state. To resume the program the RS instruction is used.

**PY**

The party flag must be set to 1 if the MDrive Motion Control is being used in a multidrop system. When Party Mode is enabled, each MDrive in the system must be addressed by the host computer by using the device name specified by the DN instruction. This name will precede any command given to a specified unit in the system and be terminated with a Control J (CTRL + J). One CTRL + J must be issued after power up or entering the Party Mode to activate the Party Mode. By default the DN assigned at the factory is the exclamation character (!).

The global name is the asterisk character (*). Commands preceded by this character will be recognized by every MDrive in the system.

After the Party Mode is enabled, send CTRL + J (^J) to activate it. Type commands with Device Name (DN) and use CTRL + J as the Terminator.

#### USE

**PS**

**EXAMPLES:**

- PY=0 'Party Mode Disabled (Default)
- PY=1 'Party Mode Enabled

**RELATED COMMANDS:** RS, S1-S4

### RELATED COMMANDS: DN
### MNEMONIC FUNCTION TYPE USAGE

#### RC Run Current Setup Variable P/I    R/W

**DESCRIPTION**
This variable defines the motor run current in percent.

**USE**
- **UNITS**: Percent
- **RANGE**: 1 to 100
- **DEFAULT**: 25

**EXAMPLE:**
RC=75  // Set motor run current to 75%

**RELATED COMMAND:** HC

---

#### R1 - R4 User Registers User Variable P/I    R/W

**DESCRIPTION**
The MDrive Motion Control has four 32 bit user registers to contain numerical data. These registers may contain up to 11 digits including the sign and may be used to store and retrieve data to set variables, perform math functions, store and retrieve moves and set conditions for branches and subroutine calls.

**USE**
- **R<x>=<data>**

**RANGE**
-2147483647 to 2147483647

**DEFAULT**
0

**EXAMPLES:**
- R1=50000  // Set Register 1 to 50000
- MS=256  // Set resolution to 256 microsteps/step
- P=0  // Set position counter to 0
- R1=51200/1  // 51200 steps = 1 rev

**RELATED COMMANDS:** —
### MNEMONIC FUNCTION TYPE USAGE

#### S Save to EEProm Instruction

**DESCRIPTION**
Saves all variables and flags currently in working memory (RAM) to nonvolatile memory (NVM). The previous values in NVM are completely overwritten with the new values.

**USE**
S

**EXAMPLE:**
S

**RELATED COMMANDS:** —

---

#### RT Return From Subroutine Instruction

**DESCRIPTION**
This instruction defines the end of a subroutine. This instruction is required and will be the final instruction in the subroutine executed by the CL instruction. When used, it will return to the program address immediately following the CL instruction which executed the subroutine.

**USE**
RT

**EXAMPLE:**

```
****Program****
PG100 'enter program mode at address 100
100 MR 51200 'move relative 51200
105 H 'suspend prog. execution until motion completes
109 CL 238 'Call subroutine at address 238
238 O1=1 'set output 1 to 1
241 RT 'return from subroutine
```

**RELATED COMMANDS:** CL

---

#### RS Resume Program Instruction

**DESCRIPTION**
This instruction is used to resume a program that has been paused using the PS instruction. Motion will resume using the normal acceleration profiles.

**USE**
RS

**EXAMPLE:**
RS

**RELATED COMMANDS:** PS, S1-S4
### DESCRIPTION

This instruction is used to setup the I/O type and active states for I/O points 1 - 4. Each of MDrive Motion Control I/O points 1-4 may be programmed as either general purpose inputs and outputs, or to one of nine dedicated input functions or one of two dedicated output functions.

When programmed as inputs, these points will be sinking and may be programmed such that they are active when pulled to ground, or active when left floating. By default each point is configured as a general purpose input, active when LOW.

There are two parameters attached to this instruction: the type specifies the function of the I/O point. The second parameter sets the active state, which defines the point as LOW or HIGH TRUE.

### I/O FUNCTION

<table>
<thead>
<tr>
<th>I/O FUNCTION</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>ACTIVE STATE PARAMETER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INPUTS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Purpose Input</td>
<td>Multiple Usage</td>
<td>0</td>
<td>LOW = TRUE 0</td>
</tr>
<tr>
<td>Home Input</td>
<td>For “Homing” Sequence (See HM)</td>
<td>1</td>
<td>HIGH = TRUE 1</td>
</tr>
<tr>
<td>Limit + Input</td>
<td>Motion stops with DECEL, sets Motion Error 83, program continues, moves in plus direction ignored. (See OE)</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Limit - Input</td>
<td>Motion stops with DECEL, sets Motion Error 84, program continues, moves in minus direction ignored. (See OE)</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>GO Input</td>
<td>Initiate Program Start. (Always at address 1)</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Soft Stop Input</td>
<td>Program and Motion Stop with DECEL</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Pause Input</td>
<td>Pause/Resume Program and Motion</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Jog + Input</td>
<td>Activate Plus Jog Input</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Jog - Input</td>
<td>Activate Minus Jog Input</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td><strong>OUTPUTS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Purpose Output</td>
<td>Multiple Usage</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Moving Output</td>
<td>Output set if Motor is Moving (See MV)</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Fault Output</td>
<td>Output set if Fault detected (See EF)</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Stall</td>
<td>Output set if Stall detected (See ST)</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>VCHG</td>
<td>Output set if Velocity Changing (See VC)</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

### USE

S<1-4>=<type>,<active>  
<type>=0, <active>=0

### EXAMPLE:

S1=2,0  
'Set i/o point 1 to a limit + function, active when LOW'

S4=17,1  
'Set i/o point 4 as moving output, active when HIGH'

### RELATED COMMANDS:

I1-4, IN, O1-4, OT, D1-D4, PS, RS, EF, ST, VC
### MNEMONIC FUNCTION TYPE USAGE

**SF Stall Factor Variable Encoder Variable P/I    R/W**

**DESCRIPTION**

If the encoder is enabled (EE = 1) and the encoder differs from the motor by more than the specified factor, a STALL is indicated. If SM is set to 0, then the motor will be stopped when a STALL is detected.

**USE**

- **SF=counts**
- **Encoder counts**

**RANGE**

- 0 to 65000

**DEFAULT**

- 10

**EXAMPLE:**

SF=20 ‘Set the stall factor to twenty counts. If the motor falls behind by more than 20 encoder counts a stall is detected.

**RELATED COMMANDS:** EE, SM, ST

### MNEMONIC FUNCTION TYPE USAGE

**SL Slew Axis Instruction Motion Instruction P/I**

**DESCRIPTION**

The SL instruction will slew the axis at the specified velocity in counts per second. The axis will accelerate at the rate specified by the A (Acceleration) variable.

Note that the maximum slew velocity is independent of the maximum velocity specified by the VM variable. If a slew is commanded at a velocity greater than the setting of VM, the axis will accelerate to that velocity regardless of the setting of VM.

**USE**

- **SL <±velocity>**
- ± Counts per sec

**RANGE**

- ±5000000

**EXAMPLE:**

SL 20000 ‘slew the axis at 20000 counts/sec

**RELATED COMMANDS:** A, D, MS, MR

---

**MNEMONIC FUNCTION TYPE USAGE**

**S5 Set/Print I/O Point 5 I/O Instruction P/I    R/W**

**DESCRIPTION**

This I/O point differs from I/O points 1-4 in that it is factory configured as a 0 - 5 V Analog Input with 10 bit A/D resolution.

**I/O FUNCTION**

- 0-5V Analog Input
- 4-20 mA Analog Input

**RELATED COMMANDS:** I5, JE
### Stall Detection Mode Variable (SM)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM</td>
<td>Stall Detection Mode Variable</td>
<td>Encoder Variable</td>
<td>P/I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
The SM variable specifies the action which will be taken by the MDrive Motion Control when a stall is detected. When set to 0 (default) the motion will be stopped upon a stall detection. When SM=1, the motor will continue to move. In either case ST (Stall Flag) will be set.

**USE**

<table>
<thead>
<tr>
<th>SM = 0/1</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM=0</td>
<td>0 (Stop Motor)</td>
</tr>
<tr>
<td>SM=1</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**
- SM=0: ‘stop motor when a stall is detected
- SM=1: ‘do not stop motor upon a stall

**RELATED COMMANDS:** EE, SF, ST, PM

### Stall Flag (ST)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST</td>
<td>Stall Flag</td>
<td>Encoder Flag</td>
<td>P/I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
The ST flag will be set to 1 when a stall is detected. It is the responsibility of the user to reset it to zero (0).

**USE**

<table>
<thead>
<tr>
<th>PR ST</th>
<th>BR &lt;addr&gt;, ST=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CL &lt;addr&gt;, ST=1</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE RESPONSE:**
- ST=0: ‘motor not stalled
- ST=1: ‘motor stalled

**RELATED COMMANDS:** EE, SF, ST, OE

### Trip Enable Flag (TE)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>FUNCTION</th>
<th>TYPE</th>
<th>USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TE</td>
<td>Trip Enable Flag</td>
<td>Setup Flag</td>
<td>P/I R/W</td>
</tr>
</tbody>
</table>

**DESCRIPTION**
This flag will enable or disable specified trip functions.

- TE=0: TI Disabled
- TE=1: TI Enabled
- TE=2: TI Disabled
- TE=3: TI Enabled

**USE**

<table>
<thead>
<tr>
<th>TE=1-4</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>TE=1</td>
<td>0 (Trips Disabled)</td>
</tr>
</tbody>
</table>

**EXAMPLE:**
- TE=1: ‘Enable trip on input functions

**RELATED COMMANDS:** I1-I4, P, S1-S4, TI, TP

**USAGE:**
P = Program    I = Immediate    R = Read    W = Write
### TP Trip on Position Variable P/I R/W

**DESCRIPTION**

Sets up a position event (trip) for the specified position. There are two parameters for the TP variable. The first specifies the position which will cause the event. The second specifies the subroutine that should be executed when the position is detected.

The TE (Trip Enable which Enables/Disables TP) is reset when a Trip occurs. TE must be re-enabled in the main program prior to the next Trip if it is to be repeated.

**USE**

\[ TP = \langle \text{position} \rangle, \langle \text{addr/label} \rangle \]

**EXAMPLE:**

\[ TP = 200000, 300 \] 'execute subroutine at address 300 when at position 200000

**RELATED COMMANDS:** P, TI, PC

### TI Trip on Input Variable P/I R/W

**DESCRIPTION**

Sets up an input event (Trip) for the specified input. There are two parameters for the TI variable. The first specifies which input line to monitor. The second specifies the subroutine that should be executed when the input goes to true.

The TE (Trip Enable which Enables/Disables TI) is reset when a Trip occurs. TE must be re-enabled in the main program prior to the next Trip if it is to be repeated.

**USE**

\[ TI = \langle \text{input} \rangle, \langle \text{addr/label} \rangle \]

**EXAMPLE:**

\[ TI = 2, K1 \] 'execute subroutine labeled K1 when input 2 is active.

**RELATED COMMANDS:** I1-4, S1-4, TP

### UG Upgrade Firmware Instruction I

**DESCRIPTION**

Upgrade Firmware Instruction. Upgrade code is 2956102. This will put the MDrive in Upgrade Mode. Once set, the firmware Upgrade MUST be completed.

**USE**

\[ UG = 2956102 \]

**RELATED COMMANDS:** —

### UV Read User Variables Variable P/I R

**DESCRIPTION**

Read User Variables is used with the PR (Print) Instruction to read the value of all user variables.

**USE**

\[ PR \ UV \]

**RELATED COMMANDS:** , PR, VA

---

**USAGE:** P = Program  I = Immediate  R = Read  W = Write
MNEMONIC FUNCTION TYPE USAGE

V  Read Only Velocity Variable  Motion Variable  P/I   R

DESCRIPTION
The velocity variable is used in conjunction with the PR (print) instruction to read the current velocity of the axis in counts per second. This variable can also be used with the BR and CL instructions to set a condition based upon a velocity. This variable can also be used in conjunction with the user registers to compute another velocity.

USE
PR V
BR <addr>, V=<counts/sec>
CL <addr>, V=<counts/sec>

RELATED COMMANDS: VI, VM

USE UNITS
Counts per Second

MNEMONIC FUNCTION TYPE USAGE

VA  Create User Variable Name  Instruction  P/I   R/W

DESCRIPTION
The VA instruction allows the user to assign a 2 character name to a user defined variable.
The restrictions for this command are:
1] A variable cannot be named after a MDrive Motion Control Instruction, Variable or Flag.
2] The first character must be alpha, the second character may be alpha-numeric.
3] A variable is limited to two characters.

USE
VA <char><char>=<value>

EXAMPLE:
VA P2  'create user var P2
P2=20000  'set P2 to 20000

RELATED COMMANDS: UV

MNEMONIC FUNCTION TYPE USAGE

VC  Velocity Changing  Motion Flag  P/I   R

DESCRIPTION
Indicates that the Velocity is changing.

USE
BR <addr>, VC

RELATED COMMANDS:
### VI: Initial Velocity Variable

**Type:** Motion Variable  
**Usage:** P/I, R/W

**Description:**
Initial velocity for all motion commands. The factory default value is 1000 clock pulses per second.

The initial velocity for a stepper should be set to avoid the low speed resonance frequency and must be set lower than the pull in torque of the motor. It must also be set to a value lower than VM (Max. Velocity).

VI must be less than VM.

**Use/Units/RANGE/Default**

<table>
<thead>
<tr>
<th>Use</th>
<th>Units</th>
<th>Range</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VI=&lt;velocity&gt;</td>
<td>Counts per sec</td>
<td>1 to 5000000</td>
<td>1000</td>
</tr>
</tbody>
</table>

**Example:**

VI=2000 ‘set initial velocity to 2000 counts per second

**Related Commands:** VM

### VM: Maximum Velocity Variable

**Type:** Motion Variable  
**Usage:** P/I, R/W

**Description:**
The VM variable specifies the maximum velocity in counts per second that the axis will reach during a move command. VM must be greater than VI.

**Use/Units/RANGE/Default**

<table>
<thead>
<tr>
<th>Use</th>
<th>Units</th>
<th>Range</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM=&lt;velocity&gt;</td>
<td>Counts per sec</td>
<td>1 to 5000000</td>
<td>768000</td>
</tr>
</tbody>
</table>

**Example:**

VM=51200 ‘set max velocity to 51200 counts per second

**Related Commands:** VM

### VR: Read Only Firmware Version

**Type:** Factory Variable  
**Usage:** P/I, R

**Description:**
This variable is used in conjunction with the PR instruction to read the version of the firmware installed at the factory. If the Version number is followed by an E, the Mdrive is an Encoder Version. An I will indicate and Index version. Blank will indicate no options.

**Use**

PR VR

**Related Commands:** —
# Appendix A

## ASCII TABLE

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>NUL</td>
<td>32</td>
<td>20</td>
<td></td>
<td>64</td>
<td>40</td>
<td>@</td>
<td>96</td>
<td>60</td>
<td>`</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>SOH</td>
<td>33</td>
<td>21</td>
<td>!</td>
<td>65</td>
<td>41</td>
<td>A</td>
<td>97</td>
<td>61</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>STX</td>
<td>34</td>
<td>22</td>
<td>&quot;</td>
<td>66</td>
<td>42</td>
<td>B</td>
<td>98</td>
<td>62</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>ETX</td>
<td>35</td>
<td>23</td>
<td>#</td>
<td>67</td>
<td>43</td>
<td>C</td>
<td>99</td>
<td>63</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>EOT</td>
<td>36</td>
<td>24</td>
<td>$</td>
<td>68</td>
<td>44</td>
<td>D</td>
<td>100</td>
<td>64</td>
<td>d</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>ENQ</td>
<td>37</td>
<td>25</td>
<td>%</td>
<td>69</td>
<td>45</td>
<td>E</td>
<td>101</td>
<td>65</td>
<td>e</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>ACK</td>
<td>38</td>
<td>26</td>
<td>&amp;</td>
<td>70</td>
<td>46</td>
<td>F</td>
<td>102</td>
<td>66</td>
<td>f</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>BEL</td>
<td>39</td>
<td>27</td>
<td>'</td>
<td>71</td>
<td>47</td>
<td>G</td>
<td>103</td>
<td>67</td>
<td>g</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>BS</td>
<td>40</td>
<td>28</td>
<td>(</td>
<td>72</td>
<td>48</td>
<td>H</td>
<td>104</td>
<td>68</td>
<td>h</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>TAB</td>
<td>41</td>
<td>29</td>
<td>)</td>
<td>73</td>
<td>49</td>
<td>I</td>
<td>105</td>
<td>69</td>
<td>i</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>LF</td>
<td>42</td>
<td>2A</td>
<td>*</td>
<td>74</td>
<td>4A</td>
<td>J</td>
<td>106</td>
<td>6A</td>
<td>j</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>VT</td>
<td>43</td>
<td>2B</td>
<td>+</td>
<td>75</td>
<td>4B</td>
<td>K</td>
<td>107</td>
<td>6B</td>
<td>k</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>FF</td>
<td>44</td>
<td>2C</td>
<td>'</td>
<td>76</td>
<td>4C</td>
<td>L</td>
<td>108</td>
<td>6C</td>
<td>l</td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>CR</td>
<td>45</td>
<td>2D</td>
<td>-</td>
<td>77</td>
<td>4D</td>
<td>M</td>
<td>109</td>
<td>6D</td>
<td>m</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>SO</td>
<td>46</td>
<td>2E</td>
<td>.</td>
<td>78</td>
<td>4E</td>
<td>N</td>
<td>110</td>
<td>6E</td>
<td>n</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>SI</td>
<td>47</td>
<td>2F</td>
<td>/</td>
<td>79</td>
<td>4F</td>
<td>O</td>
<td>111</td>
<td>6F</td>
<td>o</td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>DLE</td>
<td>48</td>
<td>30</td>
<td>0</td>
<td>80</td>
<td>50</td>
<td>P</td>
<td>112</td>
<td>70</td>
<td>p</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>DC1</td>
<td>49</td>
<td>31</td>
<td>1</td>
<td>81</td>
<td>51</td>
<td>Q</td>
<td>113</td>
<td>71</td>
<td>q</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>DC2</td>
<td>50</td>
<td>32</td>
<td>2</td>
<td>82</td>
<td>52</td>
<td>R</td>
<td>114</td>
<td>72</td>
<td>r</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>DC3</td>
<td>51</td>
<td>33</td>
<td>3</td>
<td>83</td>
<td>53</td>
<td>S</td>
<td>115</td>
<td>73</td>
<td>s</td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>DC4</td>
<td>52</td>
<td>34</td>
<td>4</td>
<td>84</td>
<td>54</td>
<td>T</td>
<td>116</td>
<td>74</td>
<td>t</td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>NAK</td>
<td>53</td>
<td>35</td>
<td>5</td>
<td>85</td>
<td>55</td>
<td>U</td>
<td>117</td>
<td>75</td>
<td>u</td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>SYN</td>
<td>54</td>
<td>36</td>
<td>6</td>
<td>86</td>
<td>56</td>
<td>V</td>
<td>118</td>
<td>76</td>
<td>v</td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>ETB</td>
<td>55</td>
<td>37</td>
<td>7</td>
<td>87</td>
<td>57</td>
<td>W</td>
<td>119</td>
<td>77</td>
<td>w</td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>CAN</td>
<td>56</td>
<td>38</td>
<td>8</td>
<td>88</td>
<td>58</td>
<td>X</td>
<td>120</td>
<td>78</td>
<td>x</td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>EM</td>
<td>57</td>
<td>39</td>
<td>9</td>
<td>89</td>
<td>59</td>
<td>Y</td>
<td>121</td>
<td>79</td>
<td>y</td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>SUB</td>
<td>58</td>
<td>3A</td>
<td>:</td>
<td>90</td>
<td>5A</td>
<td>Z</td>
<td>122</td>
<td>7A</td>
<td>z</td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>ESC</td>
<td>59</td>
<td>3B</td>
<td>;</td>
<td>91</td>
<td>5B</td>
<td>[</td>
<td>123</td>
<td>7B</td>
<td>{</td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>FS</td>
<td>60</td>
<td>3C</td>
<td>&lt;</td>
<td>92</td>
<td>5C</td>
<td>\</td>
<td>124</td>
<td>7C</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>GS</td>
<td>61</td>
<td>3D</td>
<td>=</td>
<td>93</td>
<td>5D</td>
<td>]</td>
<td>125</td>
<td>7D</td>
<td>}</td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>RS</td>
<td>62</td>
<td>3E</td>
<td>&gt;</td>
<td>94</td>
<td>5E</td>
<td>^</td>
<td>126</td>
<td>7E</td>
<td>~</td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>US</td>
<td>63</td>
<td>3F</td>
<td>?</td>
<td>95</td>
<td>5F</td>
<td>_</td>
<td>127</td>
<td>7F</td>
<td>DEL</td>
</tr>
</tbody>
</table>
## Appendix B

### Error Codes

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No Error</td>
</tr>
<tr>
<td><strong>I/O Errors</strong></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>I/O1 Fault</td>
</tr>
<tr>
<td>2</td>
<td>I/O2 Fault</td>
</tr>
<tr>
<td>3</td>
<td>I/O3 Fault</td>
</tr>
<tr>
<td>4</td>
<td>I/O4 Fault</td>
</tr>
<tr>
<td>5</td>
<td>I/O5 Fault</td>
</tr>
<tr>
<td>6</td>
<td>An I/O is already set to this type.</td>
</tr>
<tr>
<td>7</td>
<td>Tried to set an Input or defined I/O.</td>
</tr>
<tr>
<td>8</td>
<td>Tried to set an I/O to an incorrect I/O type.</td>
</tr>
<tr>
<td>9</td>
<td>Tried to write to I/O set as input or is “TYPED”.</td>
</tr>
<tr>
<td>10</td>
<td>Illegal I/O number.</td>
</tr>
<tr>
<td><strong>Data Errors</strong></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Tried to set unknown variable or flag.</td>
</tr>
<tr>
<td>21</td>
<td>Tried to set an incorrect value.</td>
</tr>
<tr>
<td>22</td>
<td>VI set greater than or equal to VM.</td>
</tr>
<tr>
<td>23</td>
<td>VM is set less than or equal to VI.</td>
</tr>
<tr>
<td>24</td>
<td>Illegal data entered.</td>
</tr>
<tr>
<td>25</td>
<td>Variable or flag is read only.</td>
</tr>
<tr>
<td>26</td>
<td>Variable or flag is not allowed to be incremented or decremented.</td>
</tr>
<tr>
<td>27</td>
<td>Trip not defined.</td>
</tr>
<tr>
<td>28</td>
<td>Trying to redefine a program label or variable.</td>
</tr>
<tr>
<td>29</td>
<td>Trying to redefine an embedded command or variable.</td>
</tr>
<tr>
<td>30</td>
<td>Unknown label or user variable.</td>
</tr>
<tr>
<td>31</td>
<td>Program label or user variable table is full.</td>
</tr>
<tr>
<td>32</td>
<td>Trying to set a label (LB).</td>
</tr>
<tr>
<td><strong>Program Errors</strong></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>Program not running.</td>
</tr>
<tr>
<td>41</td>
<td>Program running.</td>
</tr>
<tr>
<td>42</td>
<td>Illegal program address.</td>
</tr>
<tr>
<td>43</td>
<td>Tried to overflow program stack.</td>
</tr>
<tr>
<td>44</td>
<td>Program locked.</td>
</tr>
<tr>
<td><strong>Communications Errors</strong></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>Tried to enter unknown command.</td>
</tr>
<tr>
<td>61</td>
<td>Trying to set illegal BAUD rate.</td>
</tr>
<tr>
<td><strong>Motion Errors</strong></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>HOME switch not defined.</td>
</tr>
<tr>
<td>81</td>
<td>HOME type not defined.</td>
</tr>
<tr>
<td>82</td>
<td>Went to both LIMITS and did not find home</td>
</tr>
<tr>
<td>83</td>
<td>Reached plus LIMIT switch.</td>
</tr>
<tr>
<td>84</td>
<td>Reached minus LIMIT switch.</td>
</tr>
<tr>
<td>85</td>
<td>MA or MR not allowed while in motion.</td>
</tr>
<tr>
<td>86</td>
<td>Stall detected.</td>
</tr>
</tbody>
</table>
Appendix C

MDrive Motion Control (MDI) Program Samples

For additional MDI Program Samples, go to www.imshome.com/app_note_MDI.html

Some of these sample programs require digital and analog inputs which can be configured in several different ways. In some cases hardware such as a small mechanical switch, potentiometer or joystick may be needed. For more information please see:

Interfacing the Digital I/O
Interfacing the Analog Input

NOTE: The character ∧ is used to indicate a space. Do not type this character but be sure to type the space.

Motion Sample

This is a simple motion program that sets the position counter to zero (0), moves the MDI 200000 microsteps in the plus direction and then prints the position in the Terminal Window.

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>VI∧750</td>
<td>'set initial velocity to 750 steps/sec</td>
</tr>
<tr>
<td>VM∧500000</td>
<td>'set max velocity to 500000 steps/sec</td>
</tr>
<tr>
<td>A∧750000</td>
<td>'set acceleration rate to 750000 steps/sec/sec</td>
</tr>
<tr>
<td>D=A</td>
<td>'set deceleration rate equal to A</td>
</tr>
<tr>
<td>HC∧10</td>
<td>'set motor hold current to 10%</td>
</tr>
<tr>
<td>RC∧35</td>
<td>'set motor run current to 35%</td>
</tr>
<tr>
<td>PG∧100</td>
<td>'enter program mode at address 100</td>
</tr>
<tr>
<td>LB m1</td>
<td>'label program m1</td>
</tr>
<tr>
<td>P=0</td>
<td>'set position counter to 0</td>
</tr>
<tr>
<td>MR∧200000</td>
<td>'set motion to relative, move 200000</td>
</tr>
<tr>
<td>H</td>
<td>'hold until motion is complete</td>
</tr>
<tr>
<td>PR∧P</td>
<td>'print position to terminal</td>
</tr>
<tr>
<td>E</td>
<td>'end program</td>
</tr>
<tr>
<td>PG</td>
<td>'exit program mode</td>
</tr>
</tbody>
</table>

IF THEN Sample

This program moves the MDI in the plus direction IF Input One (1) is low. IF Input 1 is high, THEN the MDI will reverse direction.

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG∧100</td>
<td>'Enter program mode at address 100</td>
</tr>
<tr>
<td>LB P1</td>
<td>'Label program P1</td>
</tr>
<tr>
<td>S1=0,1</td>
<td>'Sets I/O 1 as a general purpose input, active high</td>
</tr>
<tr>
<td>LB∧M1</td>
<td>'Label to begin motion profile 1</td>
</tr>
<tr>
<td>MR∧200000</td>
<td>'Move relative 200000 steps</td>
</tr>
<tr>
<td>H</td>
<td>'Hold until motion is complete</td>
</tr>
<tr>
<td>BR∧M1∧11=1</td>
<td>'If input is true then branch to label M1</td>
</tr>
<tr>
<td>LB∧M2</td>
<td>'Label to begin motion profile 2</td>
</tr>
<tr>
<td>MR∧100000</td>
<td>'Move relative 100000 steps</td>
</tr>
<tr>
<td>H</td>
<td>'Hold until motion is complete</td>
</tr>
<tr>
<td>BR∧M1∧11=1</td>
<td>'If input is true then branch to label M1</td>
</tr>
<tr>
<td>BR∧M2</td>
<td>'Unconditional branch to label M2</td>
</tr>
<tr>
<td>E</td>
<td>'End program</td>
</tr>
<tr>
<td>PG</td>
<td>'Exit program mode</td>
</tr>
</tbody>
</table>
Appendix C

MDrive Motion Control (MDI) Program Samples

Use of I/O Sample

This is another simple program that runs the MDI until Input One (1) is turned on. When Input One (1) is turned on, the MDI will stop.

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG∧100</td>
<td>'Enter program mode at address 100</td>
</tr>
<tr>
<td>LB∧P1</td>
<td>'Label program ZZ</td>
</tr>
<tr>
<td>A=1000000</td>
<td>'Sets acceleration to 1000000 steps/second square</td>
</tr>
<tr>
<td>S1=0,1</td>
<td>'Sets I/O 1 as a general purpose input, active high</td>
</tr>
<tr>
<td>S5=9</td>
<td>'Sets I/O 5 as a 0-5V analog input</td>
</tr>
<tr>
<td>LB∧WT</td>
<td>'Label WT for setting up wait loop</td>
</tr>
<tr>
<td>BR∧M1,I1=1</td>
<td>'If input is active then branch to label M1</td>
</tr>
<tr>
<td>SL=0</td>
<td>'Sets slew speed to 0</td>
</tr>
<tr>
<td>BR∧WT,I1=0</td>
<td>'If input is inactive then branch to label WT</td>
</tr>
<tr>
<td>LB∧M1</td>
<td>'Label for motion profile 1</td>
</tr>
<tr>
<td>SL∧15*10000</td>
<td>'Slew the axis at the rate of 0-1024 times 100000</td>
</tr>
<tr>
<td>BR∧WT</td>
<td>'Unconditional branch to wait loop to scan input</td>
</tr>
<tr>
<td>E</td>
<td>'End program</td>
</tr>
</tbody>
</table>

PG

Integer Math Only (No Decimals)

This program illustrates the math capabilities of the IMS Terminal. No MDI motion occurs.

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>VA∧V1=10</td>
<td>'declare gloabal variable V1</td>
</tr>
<tr>
<td>VA∧V2=3</td>
<td>'declare gloabal variable V2</td>
</tr>
<tr>
<td>VA∧V3</td>
<td>'declare gloabal variable V3</td>
</tr>
<tr>
<td>PG∧1</td>
<td>'enter progranm mode at address 1</td>
</tr>
<tr>
<td>LB∧aa</td>
<td>'label aa</td>
</tr>
<tr>
<td>V3=V1+V2</td>
<td>'add values in V1 and V2 put result into V3</td>
</tr>
<tr>
<td>PR∧“V3=’V3”</td>
<td>'print the value in V3 to the terminal</td>
</tr>
<tr>
<td>H=1000</td>
<td>'hold program execution for 1000 milliseconds</td>
</tr>
<tr>
<td>V3=V1*V2</td>
<td>'multiply value in V1 and V2 put the result into V3</td>
</tr>
<tr>
<td>PR∧“V3=’V3”</td>
<td>'print the value of V3 to the terminal</td>
</tr>
<tr>
<td>E</td>
<td>'end program</td>
</tr>
<tr>
<td>PG</td>
<td>'exit program mode</td>
</tr>
</tbody>
</table>

******************************************************************************************

'program action

'V3=V1+V2 produces an answer of 13
'V3=V1*V2 produces an answer of 30
'V3=V1/V2 produces an answer of 3
'try typing different values of V1 and V2 and “ex aa”
'V1=35<enter>
'V2=4<enter>
'EX aa<enter>
'results
'V3=V1+V2=39
'V3=V1*V2=140
'V3=V1/V2=8
## Appendix C

### MDrive Motion Control (MDI) Program Samples

#### Limit Switch Test Program

This program demonstrates the use of limit switches. The MDI will rotate back and forth in both directions with a 250 mS HOLD between each reversal. If Input One (1) is high, the MDI will stop rotating in the plus direction and Error 83 will be printed in the Terminal Window. It will continue to rotate in the minus direction with the HOLD. If Input Two (2) is HIGH, the MDI will stop rotating in the minus direction and Error 84 will be printed to the Terminal Window. It will continue to rotate in the plus direction with the HOLD. Only the direction of travel monitored by the limit switch will be stopped.

Ref: Error Code 83 = Reached the Plus Limit Switch
Error Code 84 = Reached the Minus Limit Switch

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1=2.0</td>
<td>‘set input 1 as positive limit, low true</td>
</tr>
<tr>
<td>S2=3.0</td>
<td>‘set input 2 as negative limit, low true</td>
</tr>
<tr>
<td>PG^1</td>
<td>‘enter prog mode at address 1</td>
</tr>
<tr>
<td>LB^aa</td>
<td>‘label aa</td>
</tr>
<tr>
<td>VM=51200</td>
<td>‘set max velocity to 51200 steps/sec</td>
</tr>
<tr>
<td>MR^102400</td>
<td>‘move positive 102400 steps</td>
</tr>
<tr>
<td>H</td>
<td>‘hold prog exec until motion complete</td>
</tr>
<tr>
<td>H^250</td>
<td>‘hold prog exec for 250 milliseconds</td>
</tr>
<tr>
<td>PR^Er</td>
<td>‘print error number to terminal window</td>
</tr>
<tr>
<td>MR^-102400</td>
<td>‘move negative 102400 steps</td>
</tr>
<tr>
<td>H</td>
<td>‘hold prog exec until motion complete</td>
</tr>
<tr>
<td>H^250</td>
<td>‘hold prog exec for 250 milliseconds</td>
</tr>
<tr>
<td>PR^Er</td>
<td>‘print error number to terminal window</td>
</tr>
<tr>
<td>BR^aa</td>
<td>‘branch to label aa</td>
</tr>
<tr>
<td>E</td>
<td>‘end prog</td>
</tr>
<tr>
<td>PG</td>
<td>‘exit prog mode</td>
</tr>
<tr>
<td></td>
<td>‘hit “Esc” key to stop program</td>
</tr>
</tbody>
</table>
This program allows the user to perform speed control using the analog input on the MDI. This program will function with a potentiometer or a joystick.

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S5=9</td>
<td>'sets analog input to accept 0-5Vdc.</td>
</tr>
<tr>
<td>S1=0.1</td>
<td>'sets I/O point 1 to general purpose output</td>
</tr>
<tr>
<td>A=2000000</td>
<td>'acceleration set to 2000000 microsteps/sec/sec</td>
</tr>
<tr>
<td>D=2000000</td>
<td>'decelleration set to 2000000 microsteps/sec/sec</td>
</tr>
<tr>
<td>R4=20</td>
<td>'deadband value</td>
</tr>
</tbody>
</table>

PG\∧1           | 'initiate program at address 1                   |
LB\∧M1         | 'startup label. Program executes on power up      |
LB\∧ZZ         | 'label called ZZ                                  |
R1=I5          | 'register 1 set to analog input value             |
CL\∧ab         | 'computes new velocity by calling subroutine at label ab. |
SL\∧R3         | 'slew at the value of register 3                  |
H\∧10          | 'wait 10 milliseconds                             |
BR\∧ZZ         | 'branch to the label called ZZ                    |
E              | 'end of ZZ routine                               |

LB ab          | 'routine to determine direction and remove deadband|
R1=R1-508      | 'offset from joystick center                      |
R2=1           | 'positive direction                               |
BR\∧a1,R1>=0   | 'get dir of r1                                    |
R2=-1          | 'negative direction                               |
R1=R1*R2       | 'get absolute value of r1                         |
LB a1          |                                                 |
BR A2,R1<R4    | 'go to A2 routine if R1 value is less than deadband|
R1=R1*1000    | 'scale multiplier                                |
R3=R1*R2       | 'resultant R3 value for slew speed                |
RT             | 'return to command below call in ZZ routine.       |
LB A2          |                                                 |
R3=0           | 'sets slew velocity to zero                       |
RT             |                                                 |
E              | 'end                                             |
PG             | 'exit program space                              |
Appendix D

Recommended Cable Configurations for MDrive

Cable length, wire gauge and power conditioning devices play a major role in the performance of your MDrive.

NOTE: The length of the DC power supply cable to an MDrive should not exceed 50 feet.

Example A demonstrates the recommended cable configuration for DC power supply cabling under 50 feet long. If cabling of 50 feet or longer is required, the additional length may be gained by adding an AC power supply cable (see Examples B & C).

Correct AWG wire size is determined by the current requirement plus cable length. Please see the MDrive Supply Cable AWG Table at the end of this Appendix.

Example A – Cabling Under 50 Feet, DC Power

Example B – Cabling 50 Feet or Greater, AC Power to Full Wave Bridge
Example C – Cabling 50 Feet or Greater, AC Power to Power Supply

NOTE: These recommendations will provide optimal protection against EMI and RFI. The actual cable type, wire gauge, shield type and filtering devices used are dependent on the customer’s application and system.

### MDrive Supply Cable AWG Table

<table>
<thead>
<tr>
<th>Current (Peak)</th>
<th>Length (Feet)</th>
<th>10</th>
<th>25</th>
<th>50*</th>
<th>75*</th>
<th>100*</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Ampere</td>
<td>Minimum AWG</td>
<td>20</td>
<td>20</td>
<td>18</td>
<td>18</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current (Peak)</th>
<th>Length (Feet)</th>
<th>10</th>
<th>25</th>
<th>50*</th>
<th>75*</th>
<th>100*</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Amperes</td>
<td>Minimum AWG</td>
<td>20</td>
<td>18</td>
<td>16</td>
<td>14</td>
<td>14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current (Peak)</th>
<th>Length (Feet)</th>
<th>10</th>
<th>25</th>
<th>50*</th>
<th>75*</th>
<th>100*</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Amperes</td>
<td>Minimum AWG</td>
<td>18</td>
<td>16</td>
<td>14</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current (Peak)</th>
<th>Length (Feet)</th>
<th>10</th>
<th>25</th>
<th>50*</th>
<th>75*</th>
<th>100*</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Amperes</td>
<td>Minimum AWG</td>
<td>18</td>
<td>16</td>
<td>14</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

* Use the alternative methods illustrated in Examples A and B when the cable length is ≥ 50 feet. Also, use the same current rating when the alternate AC power is used.

MDrive Wire Size

NOTE: Always use Shielded/Twisted Pairs for the MDrive DC Supply Cable and the AC Supply Cable.
TWENTY-FOUR (24) MONTH LIMITED WARRANTY

Intelligent Motion Systems, Inc. (“IMS”), warrants only to the purchaser of the Product from IMS (the “Customer”) that the product purchased from IMS (the “Product”) will be free from defects in materials and workmanship under the normal use and service for which the Product was designed for a period of 24 months from the date of purchase of the Product by the Customer. Customer’s exclusive remedy under this Limited Warranty shall be the repair or replacement, at Company’s sole option, of the Product, or any part of the Product, determined by IMS to be defective. In order to exercise its warranty rights, Customer must notify Company in accordance with the instructions described under the heading “Obtaining Warranty Service.”

NOTE: MDrive Motion Control electronics are not removable from the motor in the field. The entire unit must be returned to the factory for repair.

This Limited Warranty does not extend to any Product damaged by reason of alteration, accident, abuse, neglect or misuse or improper or inadequate handling; improper or inadequate wiring utilized or installed in connection with the Product; installation, operation or use of the Product not made in strict accordance with the specifications and written instructions provided by IMS; use of the Product for any purpose other than those for which it was designed; ordinary wear and tear; disasters or Acts of God; unauthorized attachments, alterations or modifications to the Product; the misuse or failure of any item or equipment connected to the Product not supplied by IMS; improper maintenance or repair of the Product; or any other reason or event not caused by IMS.

IMS HEREBY DISCLAIMS ALL OTHER WARRANTIES, WHETHER WRITTEN OR ORAL, EXPRESS OR IMPLIED BY LAW OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. CUSTOMER’S SOLE REMEDY FOR ANY DEFECTIVE PRODUCT WILL BE AS STATED ABOVE, AND IN NO EVENT WILL THE IMS BE LIABLE FOR INCIDENTAL, CONSEQUENTIAL, SPECIAL OR INDIRECT DAMAGES IN CONNECTION WITH THE PRODUCT.

This Limited Warranty shall be void if the Customer fails to comply with all of the terms set forth in this Limited Warranty. This Limited Warranty is the sole warranty offered by IMS with respect to the Product. IMS does not assume any other liability in connection with the sale of the Product. No representative of IMS is authorized to extend this Limited Warranty or to change it in any manner whatsoever. No warranty applies to any party other than the original Customer.

IMS and its directors, officers, employees, subsidiaries and affiliates shall not be liable for any damages arising from any loss of equipment, loss or distortion of data, loss of time, loss or destruction of software or other property, loss of production or profits, overhead costs, claims of third parties, labor or materials, penalties or liquidated damages or punitive damages, whatsoever, whether based upon breach of warranty, breach of contract, negligence, strict liability or any other legal theory, or other losses or expenses incurred by the Customer or any third party.

OBTAINING WARRANTY SERVICE

Warranty service may be obtained by a distributor, if the Product was purchased from IMS by a distributor, or by the Customer directly from IMS, if the Product was purchased directly from IMS. Prior to returning the Product for service, a Returned Material Authorization (RMA) number must be obtained. Complete the form at http://www.imshome.com/rma.html after which an RMA Authorization Form with RMA number will then be faxed to you. Any questions, contact IMS Customer Service (860) 295-6102.

Include a copy of the RMA Authorization Form, contact name and address, and any additional notes regarding the Product failure with shipment. Return Product in its original packaging, or packaged so it is protected against electrostatic discharge or physical damage in transit. The RMA number MUST appear on the box or packing slip. Send Product to: Intelligent Motion Systems, Inc., 370 N. Main Street, Marlborough, CT 06447.

Customer shall prepay shipping charges for Products returned to IMS for warranty service and IMS shall pay for return of Products to Customer by ground transportation. However, Customer shall pay all shipping charges, duties and taxes for Products returned to IMS from outside the United States.