MMC-110

Series

Modular Motion Control System

Reference Manual

micronix USA
PRECISION MOTION SOLUTIONS

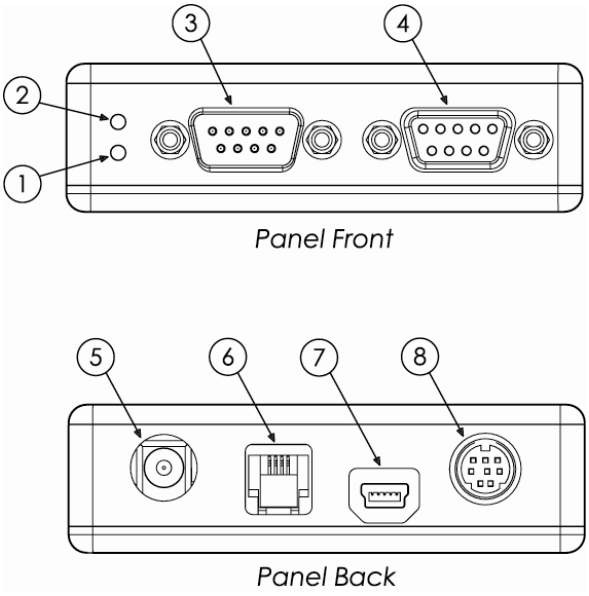
Table of Contents

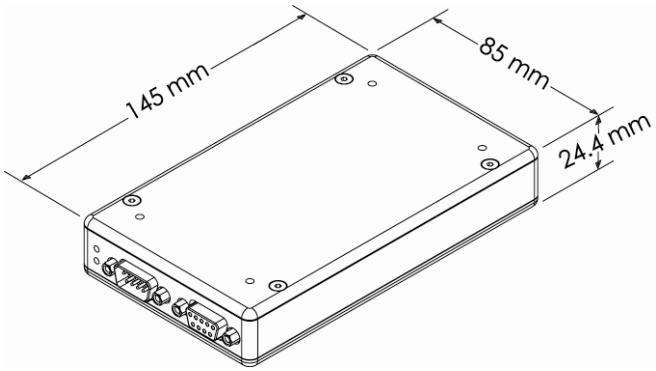
Command Index

1. Introduction

1.1 Product Description

The MMC-110 is a high performance integrated piezo motor controller/driver designed to be used as a standalone single axis unit, or stacked as a compact multi-axis module. The MMC-110 is capable of driving a piezo motor with a resolution to less than 1 nm (motor dependent). The closed loop resolution is dependent on the resolution of the encoder.





1. LED Error Indicator 1
    a. Red – An error has occurred
2. LED Addressing Indicator 2
    a. Red – Stage is Unaddressed
    b. Green – Stage has an address and is ready
3. Encoder Input, Male D-Sub 9 Pin Connector
4. Motor/Axis Output, Female D-Sub 9-Pin Connector
5. Power Supply, +24VDC, Regulated
6. RS232 Intermodular Connector
7. USB Connector
8. I/O Connector

1.2 Features

- Integrated controller/driver for MICRONIX USA stick-slip piezo motors
- Compact, modular design allows for bench-top or standard 2U height rack mounting
- Configurable as a standalone unit or stackable up to 16 axes
- Open loop/closed loop operation
- Open loop resolution of less than 1 nm
- Closed loop resolution dependent on the encoder (typically 2 nm)
- A quad B encoder feedback or 1Vpp analog encoder input
- USB interface (one interface for up to 16 axes)
- Windows GUI, and LabVIEW VI

1.3 Package Contents

If product is damaged or there are missing components, contact MICRONIX USA immediately. Do not discard product packaging in case of return shipment.

Package Contents:

- MMC-110 Controller
- Quick Start Guide
- Supplemental CD
- Power Supply
- USB Cable

2. Quick Start Guide

2.1 Quick Start Guide Overview

The following Quick Start Guide is intended to provide a basic set-up of the MMC-110 in the least amount of time. The following paragraphs will provide a walkthrough of the steps needed to set-up the controller and verify that the system is working correctly.

1. Install Drivers
    a. To ensure correct communication between the module and PC, install the proper drivers onto the communicating computer prior to connecting the MMC-110.
    b. The drivers may be found on the supplemental installation CD or can be downloaded from: http://www.micronixusa.com/motion/support/softwares.cfm

2. Connect Motion Devices
    a. A single MMC-110 controller is capable of driving one piezo motor in either open or closed loop.
    b. Connect the male D-sub 9-pin piezo motor cable to the Motor/Axis Input (as shown in the *Product Description*).

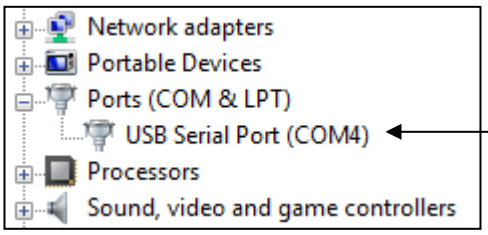
c. If applicable, connect the female D-sub 9-pin closed loop feedback cable to the Encoder Input.

3. Connect Module/Stack to PC
   a. Use the supplied Mini USB to USB cable to connect the MMC-110 controller to the communicating PC. Only one USB cable is required per module/stack.

4. Power Up Controller
   a. Connect the controller to a 24V, regulated power supply with the correct amperage rating.
   b. Each MMC-110 requires 1A. If powering a stack; add up the amperage requirements of the individual controllers to determine the necessary power supply for the stack.

5. Check COM Port
   a. It is necessary to note the COM Port assigned to the MMC-110 when connecting to a PC.
      i. In Windows 8 Open the Device Manager:
         1 Press Ctrl + x
         2 Select device manager
      ii. In Windows Vista/Windows 7 Open the Device Manager:
         1 Windows Logo (in the bottom left corner by default)
         2 Control Panel
         3 Device Manager
      iii. In Window XP Open Device Manager:
         1 Start (in the bottom left corner by default)
         2 Control Panel
         3 System
         4 select the Hardware tab
         5 Click the device manager button

   b. After powering up the controller (Step 4), note the USB Serial Port assigned. See the figure below showing a snapshot of the Device Manager window:



6. Continue to Quick Start MMC-110 Motion Controller Platform
   a. The following section will help you get running with the MMC-110 Motion Controller Platform program.

2.2 Quick Start MMC-110 Motion Controller Platform

The following Quick Start Guide is intended to provide a basic set-up of the MMC-110 MCP program. The following paragraphs will provide a walkthrough of the steps needed to install the program and verify that the system is working correctly.

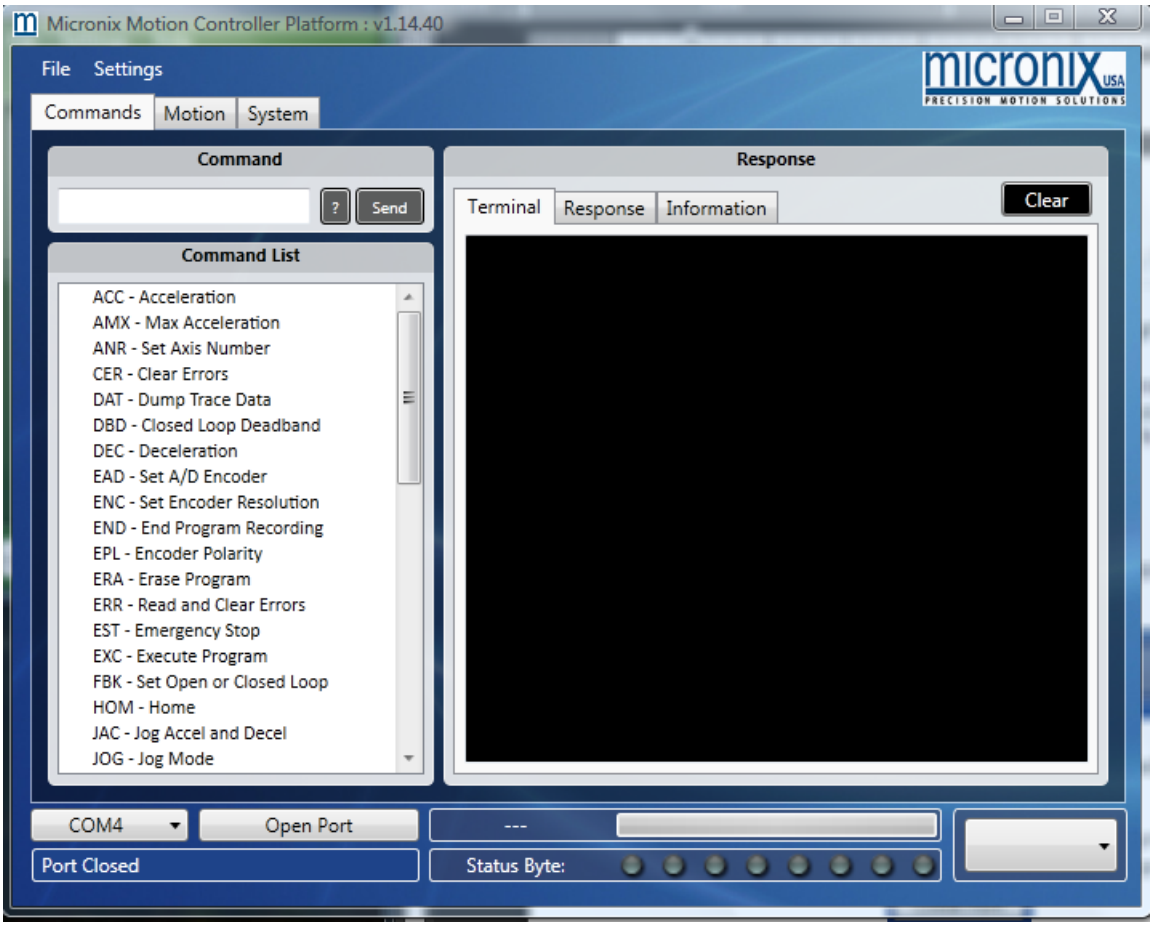
1. Pre-Installation
   a. This guide assumes you have already run through the previous Quick Start guide and that the controller is on and connected to a Com port on your computer. Please verify that this is true.
   b. You will need the .NET Framework 4.0. If you are unsure if you have the .NET Framework 4.0 follow these steps.
      i. Open the start menu (windows icon if using Vista).
      ii. Open the Control Panel
      iii. Open "Add or Remove Programs" ("Programs and Features" if using Vista)
      iv. Scroll through the list and find "Microsoft .NET Framework" If it is 4.0 skip to step-2. Otherwise continue with step c.
   c. To install the .NET Framework 4.0 you will need a connection to the internet.
      i. Navigate to this site: http://www.microsoft.com/downloads/details.aspx?FamilyID=9cfb2d51-5ff4-4491-b0e5-b386f32c0992&displaylang=en
      ii. Download and run the web installer
      iii. At the conclusion of this install you will be asked to restart your computer. Do this now.

2. Install
   a. To install the MMC-110 motion controller platform double click the setup.exe file on the supplied CD and follow the on screen instructions.

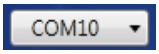
3. Run
   a. The installer placed a start menu short-cut to the MMC-110 MCP program. Make sure that your MMC-110 is connected to your computer, powered on, and connected to a valid COM port as discussed in section 2.1
   b. Open the start menu (or windows icon for vista)
   c. Open the 'all programs' tab
   d. Open the MICRONIX USA folder
   e. Run the MMC-110 MCP program

2.3 Using the MMC-110 Motion Controller Platform

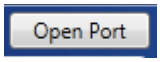
In the Quick Start Guide Overview you connected your MMC-110 to your computer. In the Quick Start MMC-110 Motion Controller Platform you installed and ran the MMC-110 MCP software. This section will describe the capabilities of the MMC-110 MCP program and give you a brief understanding of how to use it.

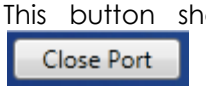


1. Port Control – The picture below depicts the program when the Port has been opened

a. Select the COM port associated with your MMC-110 

as discussed in section 2.1, step 5.

b. Click the Open Port button to connect to the MMC-110 

   i. This button should change giving you the option to close the port 

c. The Port field should change to indicate the Port is Open 

and the terminal should populate with some information about the system and then turn blank. You are now ready to start moving a stage with your MMC-110. For more information about this program see the MMC-110 MCP program guide.

2. More information – more information about the MMC-110 MCP can be found in the MMC-110 MCP program guide.

3. Technical Information

3.1 MMC-110 Specifications

Parameter	Description
Axes	1 (stackable up to 16 axes)
Motor Type	Stick-slip piezo motors
Interface	RS-232 and USB 2.0
Commands	ASCII Commands
Trajectory Mode	Trapezoidal velocity profile
Servo Clock	20 kHz
Trajectory Update	2 MHz
Power Supply	Regulated 24V DC (1A inrush, 0.5A max operation per axis)
Enclosure Dimensions	145 x 85 x 25
Software Interface	MCP-GUI, LabVIEW VI's

*A single power supply may be used per stack. Each module/axis requires 1A, therefore add up Individual module amperages to determine the power supply amperage requirement.

3.2 Serial Port Setup

If the MMC-110 is not automatically recognized by your computer, you will have to first install the FTDI interface drivers before communicating with the controller. The drivers are supplied on the supplemental CD under the folder *MMC-110 Drivers* or can be downloaded from:

http://www.ftdichip.com/Drivers/VCP.htm

Below are the virtual RS-232 configuration settings necessary for correct communication setup:

Software Parameter	Setting
Data Bits	8
Stop Bits	1
Parity	No
Handshake	No
Baud rate	38400

3.3 RJ11 RS232 Bus

This connector can be used to communicate with the MMC-110 in the place of the USB connection. For more information on the RS-232 Intermodular RJ11 connector, see the appendix 6.4.

4. Operation

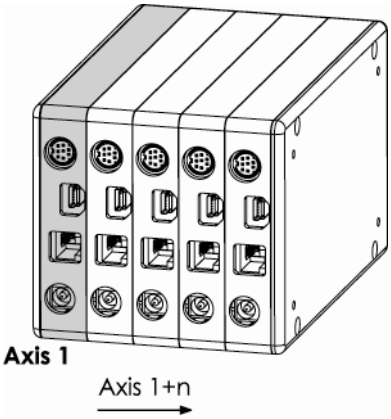
4.1 Axis Addressing

Auto Addressing is the default method of assigning axis numbers on start up. Controllers are automatically assigned axis numbers on every power up, starting with axis 1 and increasing consecutively until reaching axis 16.

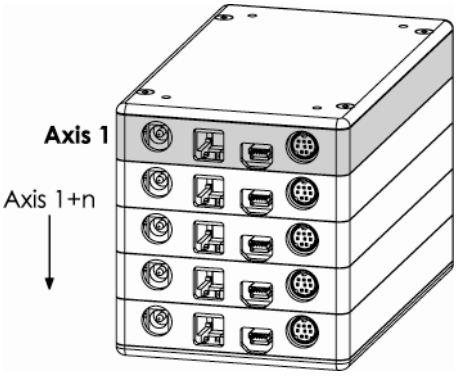
Manual axis numbers may be assigned to a unique controller using the ANR Command. This overrides Auto Addressing, as the controller stores the axis number until reassigned or reset back to Auto Addressing. In the case of having a mix of manually assigned and auto addressed controllers, the Auto Addressed axis numbers increase consecutively after each manually assigned axis in the stack. For example; in a stack of 5 controllers with the third controller manually assigned to axis 10, the axis numbers will read: 1, 2, 10, 11, 12

If two controllers are accidentally assigned the same axis number, use the global command "0ANR0" to reset all controllers back to Auto Addressing.

The figures shown below illustrate axis numbers for a 5 module stack with Auto Addressing assigned. Axis 1 is noted and shown in grey.



Horizontal stack (rear view)
With power inputs along bottom, Axis 1 is on the far left.



Vertical stack (rear view)
With power inputs along left hand side, Axis 1 is on the very top.

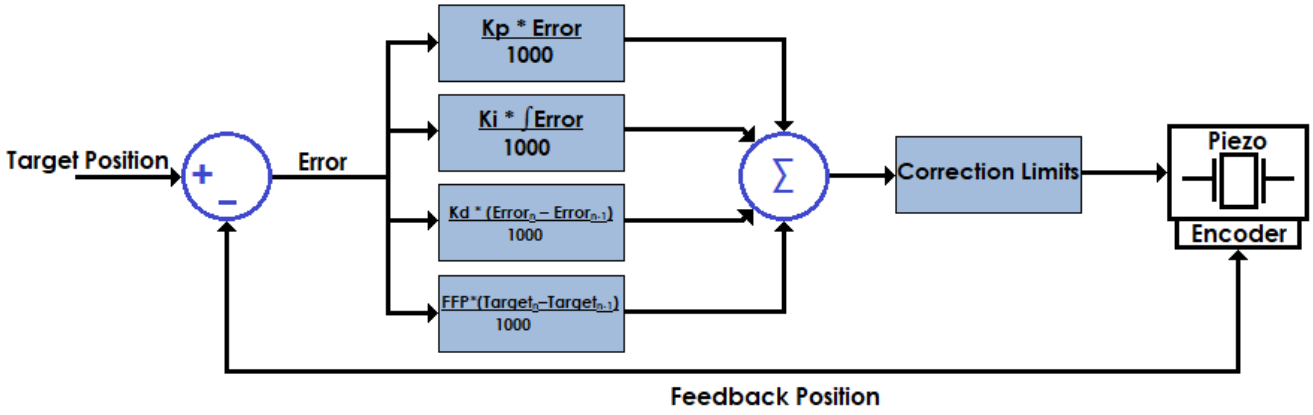
4.2 Feedback Control

The MMC-110 has three different movement modes of operation. When executing a move command, the controller will drive a stage differently when set to different modes. The FBK command is used to switch between these modes.

The first mode (nFBK0) is a traditional Open Loop. It follows a standard trapezoidal velocity characteristic. It bases the transition between acceleration, constant velocity and deceleration on the resolution settings (nREZx) or the distance it travels in one pulse. This is entirely open loop and does not guarantee a set trajectory or end point.

The second mode (nFBK2) is a version of closed loop; taking position data from an attached encoder and using it to ensure that it stops at the desired position. In this mode the controller runs in open loop mode until it reaches the deceleration point. At this point it constantly reads from encoder and corrects its position to arrive at the correct position. This guarantees position within the specified deadband (DBN Command). However, this mode does not operate in closed loop during motion.

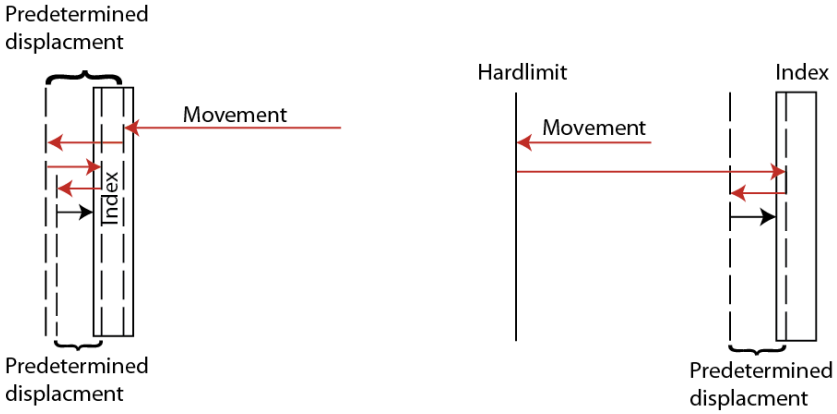
The third mode (nFBK3) is closed loop. The controller will constantly try to achieve an ideal trapezoidal velocity characteristic. Like the previous mode it too uses position data from an attached encoder and guarantees final position within the specified deadband. The closed loop algorithm in the third feedback mode will utilize a PID control loop (PID command) along with a feedforward parameter (FFP command). Below is the servo loop block diagram.



In each servo cycle the calculated target position and the encoder feedback position will be used to determine the error. The summation of the proportional, integral, derivative, and feed forward terms will determine the error adjustment for the given servo cycle. The proportional (Kp), integral (Ki), and derivative (Kd) gains can be adjusted through the PID setting (PID command). The feed forward parameter can be set by the feed forward setting (FFP).

4.3 HOM, MLN, and MLP

The HOM, MLN and MLP commands all require the attached stage to have an encoder. The HOM command will move negative direction by default. This can be changed using the HCG command. If the stage is above the index, it will move until it reaches the index then move a predetermined distance out of the index in the negative direction. The stage will then travel in the positive direction at a slower speed stopping at the edge of the index. If the stage is below the index it will move until it reaches a hard limit or the maximum travel. It then reverses direction and proceeds until it reaches the index. It will then travel a predetermined distance out of the index in the negative direction and finally travel toward the index at a slower velocity finally resting on the edge of the index. The HOM command will always home to the negative side of the limit.



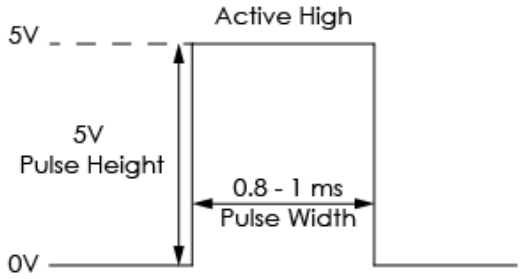
4.4 I/O Commands

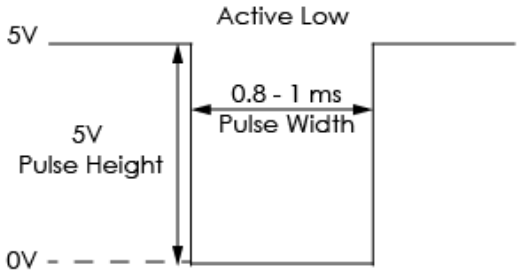
The MMC-110 has an I/O connector that can be used to output pulse signals when the stage has completed certain tasks. The I/O connector uses an 8-Pin Din connector, and its pinout is as described in section 6.3 of the Appendix.

Each IO pin on the MMC-110 is assigned as either an input or output. IO pins 1, 4, and 5 are outputs and IO pins 2, 3, and 6 are inputs. The available IO functions will only operate on IO pins with the required direction. For example, function 1 of IOF (pg. 5-43) is available for defined inputs (IO pins 2, 3, and 6) and functions 2 to 6 are available for defined outputs (IO pins 1, 4, and 5).

The IO Function (IOF) command (pg. 5-43) assigns functions to the corresponding IO pins. The IO function will either to output a signal via the IO pin when certain conditions are met or using an incoming signal from an IO pin to perform an operation.

The IO Polarity (IOP) command (pg. 5-44) is used to change the polarity of the pulse output of an IO pin. The IO polarity sets active high or active low pulse output for the following commands (CVP, PIP, and PTP).





The IO Status (IOS) command (pg. 5-45) can be used to manually read and write IO pins. Note that the IO pin can only be set high or low on defined outputs (OP pins 1, 4, and 5). The IO polarity setting (IOP) will also affect the read and write operation of the IO status.

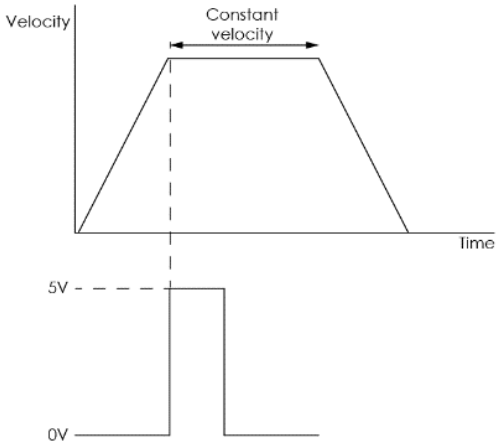
CVP – Pulse at Constant Velocity

The CVP(Pulse at Constant Velocity) command sends a pulse trigger once the stage reaches constant velocity. All IO pins assign to the IO function (IOF) #4 will output a pulse at constant velocity. The CVP command requires an enable (1)/disable (0) parameter.

The CVP operation is only available on IO pins 1, 4, and 5.

An example of setting up an IO pin to output at constant velocity is as follows:

```
1IOP1,1      //IO1 pulse trigger as active high
1IOF1,4      //IO1 pulse trigger with CVP
1CVP1        //CVP is enabled.
```

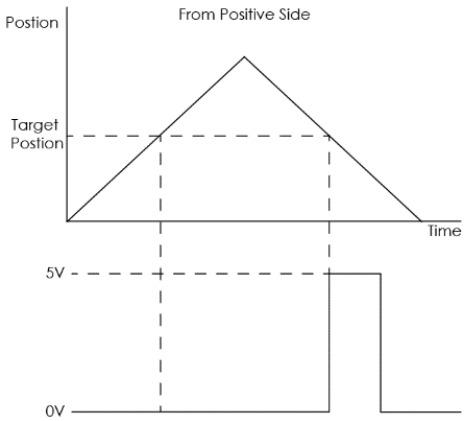


PTP – Pulse at Target Position

The PTP operation will send an output pulse to assigned IO pins once the MMC-110 has reached a specified target position. Note that the PTP function will require an encoder feedback position for the target position. All IO pins assigned to the IO Function (IOF) #5 will output a pulse.

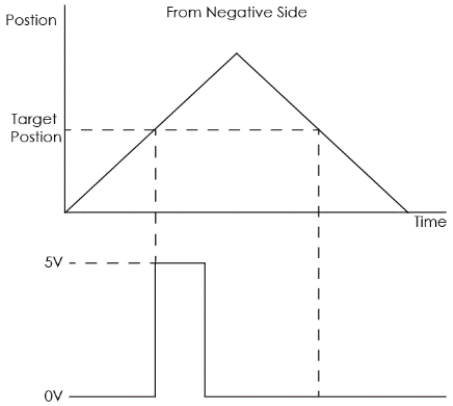
The PTP command will accept two parameters. The first parameter defines the target position and the second parameter determines the required direction of motion to trigger an output pulse. Depending on the direction chosen for the PTP command, the pulse trigger may activate when moving negative (0), positive (1), or from either direction (2).

The PTP function is only available on IO pins 1, 4, and 5.



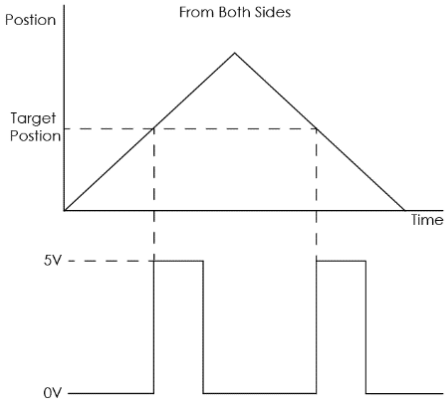
Example 1: Output pulse to IO #4 at target position 3.0mm in the negative direction

```
1IOF4,5        //Assign PTP to IO4
1PTP4,0        //pulse trigger (-) motion
```



Example 2: Output pulse to IO #5 at target position 3.0mm in the positive direction

```
2IOF5,5        //Assign PTP to IO5
2PTP3,1        //pulse trigger (+) motion
```



Example 3: Output pulse to IO #1 at target position 3.0mm in either direction.

```
3IOF1,5        //Assign PTP to IO1
3PTP3,2        //pulse trigger from either direction
```

PIP – Pulse in Regular Intervals

The pulse at interval position (PIP command) function will send pulses at a defined position interval. Note that the PIP function will require an encoder feedback position for the position interval. All IO pins assigned to the IO function (IOF) #6 will output a pulse during the defined position intervals.

The syntax for PIP is as follows: PIP x, i, y where "x" is the starting position, "i" is the interval for each pulse trigger, and "y" is the ending position.

The interval, "i", can be either positive or negative interval depending on the difference between the ending position and starting position.

If y – x > 0 , then interval, i, has to be positive.
If y – x < 0 , then interval, i, has to be negative.

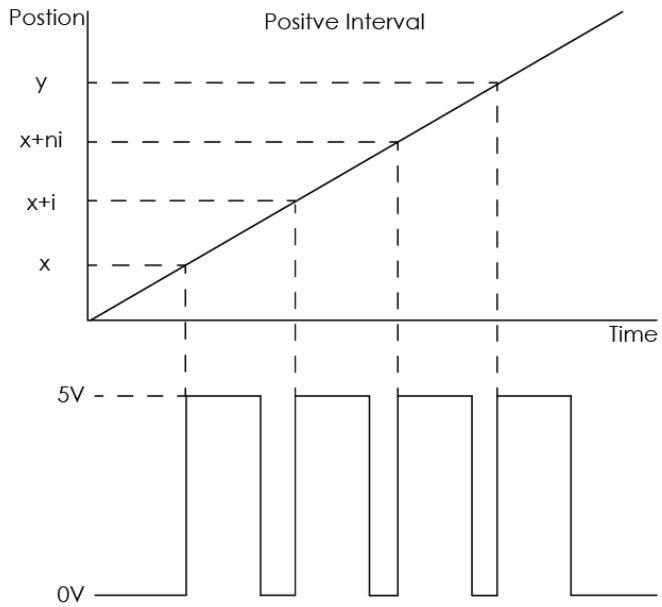
The total number of pulses expected is calculated as

Total pulses = int[(y-x) / i] + 1 if y-x ≥ i
int[(y-x) / i] + 2 if y-x < i

The PTP operation is only available on IO pins 1, 4, and 5.

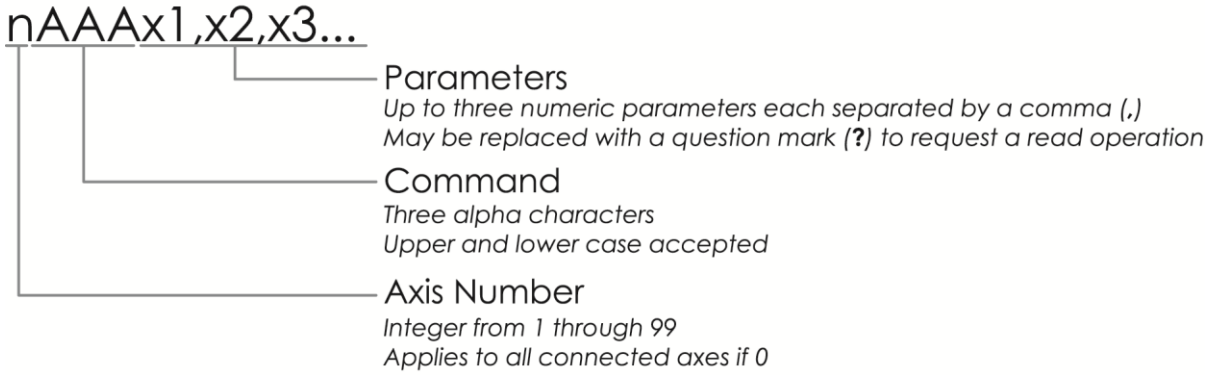
An example of setting up an IO pin to output pulses with PIP is as follows:

```
1IOP4,1      //IO4 pulse trigger as active high
1IOF4,6      //IO4 pulse trigger with PIP
1PIP1,1,3    //Pulse triggers at 1mm, 2mm, and //3mm.
```



5. Commands

5.1 Command Line Syntax



There are three components to every command. The first is the "Axis Number" which designates which controller, or axis, will receive the command. If the "Axis Number" is 0, then the command will be sent globally to all connected controllers. It is possible to connect up to 16 controllers; therefore the "Axis Number" will be an integer value from 0 through 16.

The second component is the "Command", which is always comprised of three letters. Each command is outlined, along with its corresponding parameters, in the *Command Description* section 5.9 of this manual.

The third and final component is the "Parameter". This portion is command dependent, meaning that the parameter value will change depending on the specific requirements of the "Command". Where applicable, a question mark (?) may be substituted to initiate a read operation which will return information regarding the particular command. There may be up to three separate parameters for a particular command, each parameter value is separated by a comma (,).

All white space (blank spaces) are ignored in the command format. The following are examples of equivalent commands:

```
1VEL2
1 VEL 2
```

When communicating with the controller, it is necessary to note the terminating characters involved in transmitting and receiving data. To send data to the controller, enter the desired commands in the command line followed by the new line and carriage return terminating characters [\n\r], or just the carriage return terminating character [\r]. When receiving, each line of data will be followed by the new line terminating character [\n] and the final line will end in the new line and carriage return terminating characters [\n\r]. The ASCII value for new line [\n] is 0X0A and for carriage return [\r] is 0X0D. The following is an example of data transmission:

```
1VEL0.005 \n\r      | Axis 1, Set velocity to .005 mm/s [degrees/s] [New line, Carriage Return]
```

5.2 Command Line Format

Commands are first executed in the order that they are input, then line by line. This means that two commands on the same line are executed significantly closer to each other than if they were on two separate lines. Each command is separated by a semicolon (;) and every command line ends in a terminator (EX: carriage return). The following is an example of a command line entry:

```
1MVR16;3MVR12        | Axis 1, Move 16 mm [16 degrees]; Axis 3, Move 12 mm [12 degrees]
```

Using multiple commands on the same command line allows for closer synchronization of different commands to different axes.

5.3 Global Commands

Some commands have the option of being executed globally. This means that you can send the same command to all available axes. To do this, replace the axis number of a global command with a '0'. For example; 0ACC 50 will set the acceleration of all available axes to 50 mm/s^2 [degrees/s^2].

5.4 Multiple Parameters

When dealing with a command that has multiple parameters, it is possible to change a single parameter by omitting numbers for the parameters that will remain unchanged. For example; `4DBD,0.3` will only change the second parameter to a new value, "0.3".

5.5 Synchronous Move

It is possible to execute multiple motions at the same time by setting up and executing a synchronous move. To set up a synchronous move, use the MSA and MSR commands. These commands can be written on the same command line (up to 8 allowed) or on separate lines followed by a line terminator. To execute the move, use the RUN command on the proceeding command line followed by a line terminator. For example;

```
1MSA4;2MSA4;3MSA4        | Axis 1, Move 4mm;  Axis 2, Move 4mm;  Axis 3 Move 4mm
0RUN                     | Run Synchronous Move
```

Or

```
1MSA4                    | Axis 1, Move 4mm
2MSA2                    | Axis 2, Move 2mm
3MSA3                    | Axis 3 Move 3mm
0RUN                     | Run Synchronous Move
```

5.6 Internal Programming

An internally stored program may be used to save time when repeatedly using a sequence of commands. Each controller or axis must be programmed individually; however, multiple controllers may execute the same program at the same time.

Existing program numbers cannot be overridden unless previously erased using the ERA command. nPGM? will return a binary representation of which program slots are already programmed. nLST1 will return a list of the commands that are written in program 1.

To record a program sequence, enter the PGM command on a unique line followed by a line terminator. End a program sequence by entering the END command on a unique line followed by a line terminator. When you want to execute this program, use the EXC command. See the *Summary of Commands* page for a list of program compatible commands and more information about the PGM, END and EXC commands.

Commands intended to be stored in a program must be preceded with the '%' character, or else they will be executed immediately.

Program Examples:
1pgm1
%1mvr2
%1vel?
1mvr2 //This will happen immediately and will not be part of the program.
%1end

1pgm2
%1mvr2
%1vel? //Program output is routed using the PRT setting. Options listed in attachment
%1end

A command stored into a program with a hard-coded axis (ex. '1mvr5'), will no longer execute after a change in the axis number (ANR command). Programs can contain a special 'me' character ('*'), this allows programs to function properly if the address for a controller is changed.

1pgm2
*%*mvr2*
*%*vel?*
*%*end*

5.7 Summary of Commands

Command	Description	During Motion		Real-time		Program		Global		Page
		Set	Read	Set	Read	Set	Read	Set	Read	
ACC	Acceleration		✓	✓	✓	✓	✓	✓	✓	20
AMX	Maximum Allowable Acceleration		✓		✓		✓		✓	21
ANR	Set Axis Number		✓	✓	✓		✓	✓*	✓	22
CER	Clear Errors			✓				✓		23
CVL	Correction Velocity		✓	✓	✓			✓		24
CVP	Pulse at Constant Velocity		✓	✓	✓			✓		25
DAT	Dump Trace Data				✓					26
DBD	Closed Loop Deadband		✓	✓	✓		✓	✓	✓	27
DEC	Deceleration		✓	✓	✓	✓	✓	✓	✓	28
DEF	Restore Factory Defaults			✓				✓		29
EAD	Set Analog or Digital Encoder		✓	✓	✓		✓	✓	✓	30
ENC	Select Encoder Resolution		✓	✓	✓		✓	✓	✓	31
END	End Program Recording			✓		✓				32
EPL	Encoder Polarity		✓	✓	✓		✓	✓	✓	33
ERA	Erase Program	✓		✓				✓		34
ERR	Read and Clear Errors		✓		✓		✓		✓	35
EST	Emergency Stop	✓		✓		✓		✓		36
EXC	Execute Program			✓				✓		37
FBK	Set Open or Closed Loop Mode		✓	✓	✓		✓		✓	38
FFP	Feed Forward Parameter		✓	✓	✓					39
HCG	Home Configuration		✓	✓	✓	✓	✓	✓	✓	40
HOM	Home		✓	✓	✓	✓	✓	✓	✓	41
HST	Hard Stop Detection	✓		✓	✓	✓	✓	✓		42
INP	In Position		✓	✓	✓			✓		43
IOF	IO Function			✓					✓	44
IOP	IO Polarity		✓	✓	✓					45
IOS	IO Status		✓	✓	✓					46
IWL	Integrator Windup Limit		✓	✓	✓					47
JAC	Jog Acceleration and Deceleration		✓	✓	✓		✓	✓	✓	48
JOG	Jog Mode	✓		✓		✓		✓		49
LCG	Limit Configuration		✓	✓	✓		✓	✓	✓	50
LDP	Load Parameters			✓				✓		51
LDR	Limit Switch Direction		✓	✓	✓	✓	✓	✓	✓	52
LPL	Limit Switch Polarity		✓	✓	✓		✓		✓	53
LST	Program List		✓	✓	✓		✓		✓	54
MLN	Move to Negative Limit			✓		✓		✓		55
MLP	Move to Positive Limit			✓		✓		✓		56

Continued…

Command	Description	During Motion		Real-time		Program		Global		Page
		Set	Read	Set	Read	Set	Read	Set	Read	
MOT	Toggle Motor On/Off		✓	✓	✓	✓	✓	✓	✓	57
MPL	Motor Polarity		✓	✓	✓	✓	✓	✓	✓	58
MSA	Synchronous Move – Absolute	✓		✓				✓		59
MSR	Synchronous Move – Relative	✓		✓				✓		60
MVA	Move Absolute	✓		✓		✓		✓		61
MVR	Move Relative	✓		✓		✓		✓		62
PGL	Loop Program		✓	✓	✓	✓	✓	✓	✓	63
PGM	Begin Program Recording		✓	✓	✓		✓		✓	64
PGS	Run Program At Start-Up		✓	✓	✓		✓	✓	✓	65
PID	Set Feedback Constants		✓	✓	✓		✓	✓	✓	66
PIP	Pulse at Interval Position		✓	✓	✓					67
POS	Position		✓		✓		✓		✓	68
PTP	Pulse at Target Position		✓	✓	✓					69
REZ	Set Resolution		✓	✓	✓		✓	✓	✓	70
RUN	Start Synchronous Move			✓				✓		71
SAV	Save Axis Settings			✓				✓		72
STA	Status Byte		✓		✓		✓		✓	73
STP	Stop Motion	✓		✓		✓		✓		74
SVP	Save Startup Position		✓	✓	✓	✓	✓	✓	✓	75
SYN	Sync					✓		✓		76
TLN	Negative Soft Limit Position		✓	✓	✓	✓	✓	✓	✓	77
TLP	Positive Soft Limit Position		✓	✓	✓	✓	✓	✓	✓	78
TRA	Perform Trace		✓	✓	✓	✓		✓		79
VEL	Velocity	✓	✓	✓	✓	✓	✓	✓	✓	80
VER	Firmware Version		✓		✓		✓		✓	81
VMX	Max. Allowable Velocity		✓		✓		✓		✓	82
VRT	Encoder Velocity		✓		✓		✓		✓	83
WST	Wait For Stop					✓				84
WSY	Wait For Sync					✓		✓		85
WTM	Wait For Time Period					✓				86
ZRO	Zero Position			✓		✓		✓		87

5.8 Command Descriptions

ACC

Acceleration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired acceleration for the specified axis, distinct from the deceleration [DEC]. The acceleration value must be less than the maximum acceleration [AMX] for the command to be accepted.
Returns:	A read operation returns the acceleration value in mm/s^2 for the specified axis.
Syntax:	nACCx – Standard syntax nACC? – Read acceleration value 0ACCx –All axes set acceleration value Error [#]: ACC? – Read operation with missing axis number [27] nACC – Missing acceleration parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Acceleration ? – Read acceleration value
Parameter Range:	n – 0 to 99 x – 000.000001 to AMX
Related Commands:	DEC, VEL, JAC, AMX
Example:	`3ACC100` | Axis 3, Set acceleration to 100 [degrees/s^2] – `4ACC?` | Axis 4, Read acceleration value

AMX

Maximum Allowable Acceleration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓	✓	✓

Command Description:	This command is used to set the maximum allowable acceleration for the specified axis.
Returns:	A read operation returns the maximum allowable acceleration value in mm/s^2 for the specified axis.
Syntax:	nAMXx – Standard syntax nAMX? – Read maximum allowable acceleration value 0AMXx – All axes set maximum allowable acceleration value Error [#]: AMX? – Read operation with missing axis number [27] nAMX – Missing maximum acceleration parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Maximum acceleration ? – Read maximum allowable acceleration value
Parameter Range:	n – 0 to 99 x – 000.001 to 2000.000 mm/s^2 [degrees/s^2]
Related Commands:	DEC, VEL, JAC, VMX, ACC
Example:	`2AMX1.500` | Axis 2, Set max acceleration to 1.500 mm/s^2 [degrees/s^2] – `6AMX?` | Axis 6, Read max acceleration value

ANR

Set Axis Number

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓*	✓

Command Description:	This command is used to override Auto Addressing by manually assigning an axis number to a controller. Auto Addressing is the default method of assigning axis numbers on power up and may be reassigned to an axis by substituting a "0" for the parameter value. Simultaneous axis swapping is possible by using multiple ANR commands on the same command line. *This command can be called globally by specifying a '0' for the axis number; however it will only work if the new axis number parameter is set to '0' for auto-addressing.
Returns:	A read operation returns the following axis number values for the specified axis: 0 – Auto Addressing assigned (default) 1-99 – Manually assigned, axis number displayed
Syntax:	nANRx – Standard syntax nANR? – Read axis number value Error [#]: ANR? – Read operation with missing axis number [27] nANR – Missing new axis number parameter [28] ANRx – Missing axis number [30]
Parameter Description:	n[int] – Axis number x[int] – New axis number, 0 for Auto Addressing ? – Read axis number value
Parameter Range:	n – 0 to 99 x – 0 to 99
Related Commands:	None
Example:	`5ANR1;1ANR5` | Simultaneous axis swapping: Axis 5, Set to axis 1; Axis 1, Set to axis 5 – `4ANR0` | Axis 4 Set to Auto Addressing. However it will remain axis 4 until the MMC-110 is reset

CER

Clear Errors

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓		✓		✓		✓	

Command Description:	This command is used to clear all error messages without reading them.
Returns:	A read operation cannot be used with this command.
Syntax:	nCER – Standard syntax 0CER – All axes clear error messages
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	ERR
Example:	`1CER` | Axis 1, clear error messages – `0CER` | All axes, clear error messages

CVL

Correction Velocity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓			✓	

Command Description:	This command is used to change the correction velocity. This term is only used the FBK modes 2 and 3.
Returns:	The value for the correction velocity
Syntax:	nCVL – Standard syntax Error [#]: CVLx – Missing axis number [30]
Parameter Description:	n[int] – Axis number x[float] – Correction velocity ? – Read correction velocity
Parameter Range:	n – 1 to 99 x – 000.001 to VMX
Related Commands:	CST, FBK
Example:	`2CVL1` | Axis 2, Set correction velocity to 1]

CVP

Pulse at Constant Velocity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓			✓	

Command Description:	This command is used in conjunction with IOF and IOP to command the controller to emit a pulse once the constant velocity state of motion is reached. CVP operation is available on IO pins 1, 4, and 5.
Returns:	A read operation returns the values assigned to the enable value.
Syntax:	nCVPx – Standard syntax 0 CVPx – All axes execute enable value nCVP? – Read enable value Error [#]: 1CVP2 Invalid Input Parameter (enable must be a "1" or a "0").
Parameter Description:	n[int] – Axis number x[int] – Enable value
Parameter Range:	n – 0 to 99 x – 0 CVP Disabled 1 CVP Enabled
Related Commands:	IOF, IOP
Example:	`4IOP1,1` | Axis 4, Set IO Pin 1's pulse as active high `4IOF1,4` | Axis 4, Set IO Pin 1 for CVP function `4CVP1` | Axis 4, Pulse trigger at IO pins with CVP when constant velocity is reached.

DAT

Dump Trace Data

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
			✓				

Command Description:	This command is used to read trace data from a specified axis initially recorded by the trace command [TRA]. The retrieved trace data set is dumped from the controller, consequently allowing the data to be retrieved only once.
Returns:	A read operation returns the trace data values for the specified axis in the following format: [Theoretical Position (.5nm)],[Actual Position(.5nm)], [DAC Value], [Not Used]
Syntax:	nDAT? – Read trace data values Error [#]: DAT? – Read operation with missing axis number [27] nDAT – Missing read operation parameter [28]
Parameter Description:	n[int] – Axis number ? – Read trace data values
Parameter Range:	n – 1 to 99
Related Commands:	TRA
Example:	`11DAT?` | Axis 11, Read trace data values

DBD — Closed Loop Deadband

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the acceptable deadband and deadband timeout values. Deadband refers to the number of encoder counts (±) from the target that is considered acceptable. If the parameter (x1) is set to "0", the controller will continuously oscillate around the target. Deadband timeout refers to the amount of time that the controller will try to move into the deadband area. If the parameter (x2) is set to "0", the controller will seek continuously.
Returns:	A read operation returns the deadband and deadband timeout values for the specified axis.
Syntax:	nDBDx1,x2 – Standard syntax nDBD? – Read deadband and deadband timeout values 0DBDx1,x2 – All axes set deadband and deadband timeout values Error [#]: DBD? – Read operation with missing axis number [27] nDBD – Missing deadband and deadband timeout parameter values [28]
Parameter Description:	n[int] – Axis number x1[int] – Deadband x2[float] – Deadband timeout ? – Read deadband and deadband timeout values
Parameter Range:	n – 0 to 99 x1 – Encoder dependent, 0 for continuous, Encoder Counts x2 – Encoder dependent, 0 for infinite, Seconds (default 0)
Related Commands:	ENC, EPL
Example:	`1DBD10,1` | Axis 1, Set deadband to 10 encoder counts & deadband timeout to 1 second – `4DBD5,0` | Axis 4, Set deadband to 5 encoder counts & deadband timeout to infinite

DEC

Deceleration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired deceleration for the specified axis, distinct from the acceleration [ACC]. The deceleration value must be less than the maximum acceleration value [AMX] for the command to be accepted.
Returns:	A read operation returns the deceleration value in mm/s^2 for the specified axis.
Syntax:	nDECx – Standard syntax nDEC? – Read deceleration value 0DECn – All axes set deceleration value Error [#]: DEC? – Read operation with missing axis number [27] nDEC – Missing deceleration parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Deceleration ? – Read deceleration value
Parameter Range:	n – 0 to 99 x – 000.000001 to AMX
Related Commands:	ACC, AMX, VEL
Example:	`2DEC1.25` | Axis 2, Set deceleration to 1.25 mm/s^2 [degrees/s^2] – `7DEC?` | Axis 7, Read deceleration value

DEF

Restore Factory Defaults

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓					

Command Description:	This command restores the factory default parameters.
Returns:	A read operation is not available with this command.
Syntax:	nDEF – Standard syntax Error [#]: DEF – Missing axis number [30]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	SAV
Example:	`1DEF` | Axis 1, Set default parameters]

EAD

Set Analog or Digital Encoder

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to specify whether the encoder signal for a specified axis is analog or digital.
Returns:	A read operation returns the following encoder mode values for the specified axis: 0 – Digital 1 – Analog
Syntax:	nEADx – Standard syntax nEAD? – Read encoder mode value 0EADx – All axes set encoder value Error [#]: xEAD – Missing encoder mode parameter [28] EAD? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number x[int] – Encoder mode ? – Read encoder mode value
Parameter Range:	n – 0 to 99 x – 0 for digital, 1 for analog
Related Commands:	ENC
Example:	`9EAD0` | Axis 9, Set encoder to digital input

ENC

Set Encoder Resolution

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to set the desired encoder resolution for the specified axis. When a digital encoder is connected, encoder resolution is determined by the encoder itself and the ENC setting will need to reflect this value. Analog encoder resolution is set by the controller.
Returns:	A read operation returns the encoder resolution value for the specified axis.
Syntax:	nENCx – Standard syntax nENC? – Read encoder resolution value 0ENCx – All axes execute encoder resolution value Error [#]: ENC? – Read operation with missing axis number [27] nENC – Missing encoder resolution parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Encoder resolution ? – Read encoder resolution value
Parameter Range:	n – 0 to 99 x – 0.0000005 to 999.999999 µm/count (milli-degrees/count)
Related Commands:	EAD
Example:	`2ENC10` | Axis 2, Set encoder resolution to 10 microns/count (10 milli-degrees/count)

END

End Program Recording

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓			

Command Description:	This command is used to exit out of program recording mode, which is initiated by the PGM command. The END command must be placed separately on the last line of the program sequence and preceded by the % sign to be considered a part of the current program. The resulting program is saved upon exit for later use.
Returns:	A read operation is not available with this command.
Syntax:	%nEND – Standard syntax Error [#]: END – Missing axis number [30] 1END – Missing "%", will not be stored in program.
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	REC, EXC, PGM
Example:	*1PGM* | Axis 1, Begin program recording *1VEL1;1ACC.5* | Axis 1, Set velocity value to 1 mm/s; Axis 1, Set acceleration value to 0.5 mm/s^2 [degrees/s^2] 1END | Axis 1, End program recording

EPL

Encoder Polarity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to switch the encoder signal polarity for the specified axis. If the controller doesn't seem to be recording encoder position correctly, the polarity of the encoder signals could be reversed. Use this command to switch from the default setting (normal operation, n=0).
Returns:	A read operation returns the following encoder polarity values for the specified axis: 0 – Normal operation 1 – Reverse operation
Syntax:	nEPLx – Standard syntax nEPL? – Read encoder polarity value 0EPLx – All axes execute encoder polarity value Error [#]: EPL? – Read operation with missing axis number [27] nEPL – Missing encoder polarity parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Encoder polarity ? – Read encoder polarity value
Parameter Range:	n – 0 to 99 x – 0 for normal operation, 1 for reverse operation
Related Commands:	DBD
Example:	`13EPL0` | Axis 13, Set encoder polarity to normal operation – `6EPL1` | Axis 6, Set encoder polarity to reverse operation

ERA

Erase Program

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓			✓		✓

Command Description:	This command is used to erase a specified program from an axis. Before recording a program, use the LST command to see what program numbers are available for that axis. There are 16 program numbers available allowing up to 16 programs to be stored. An existing program cannot be overwritten and must be erased first. Therefore, use this command to erase the specified program and make space for a new one.
Returns:	A read operation is not available with this command.
Syntax:	nERAx – Standard syntax Error [#]: ERAx – Missing axis number [30] nERA – Missing program number parameter [28]
Parameter Description:	n[int] – Axis number x[int] – Program number to be erased
Parameter Range:	n – 1 to 99 x – 1 to 16
Related Commands:	LST
Example:	`5ERA4` | Axis 8, Erase program 4

ERR

Read and Clear Errors

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓				✓

Command Description:	This command is used to read and clear any pending error messages.
Returns:	A read operation returns a list of error messages for the specified axis in the following format. "AAA" signifies the specific command name that the error corresponds to. Error Number – Description [AAA]
Syntax:	nERR? – Standard syntax Error [#]: ERR? – Read operation with missing axis number [123]
Parameter Description:	n[int] – Axis number ? – Read error messages
Parameter Range:	n – 1 to 99
Related Commands:	None
Example:	`3ERR?` | Axis 3, Read error messages

EST

Emergency Stop

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓		✓		✓		✓	

Command Description:	This command is used to stop a specific axis or all connected axes simultaneously in case of an emergency. The controller executes the largest possible deceleration.
Returns:	A read operation is not available with this command.
Syntax:	nEST – Standard syntax 0EST – All axes execute emergency stop
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	STP
Example:	`8EST` | Axis 8, Emergency stop – `0EST` | All axes, Emergency stop

EXC

Execute Program

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓				✓	

Command Description:	This command is used to execute a specified program for one or multiple axes. If executing a program globally, all connected axes should have individual programs stored under the specified program number prior to execution.
Returns:	A read operation is not available with this command.
Syntax:	nEXCx – Standard syntax 0EXCx – All axes execute program Error [#]: nEXC – Missing program number parameter [123]
Parameter Description:	n[int] – Axis number x[float] – Program number to be executed
Parameter Range:	n – 0 to 99 x – 1 to 32
Related Commands:	PGM
Example:	`4EXC5` | Axis 4, Execute program 5 – `0EXC2` | All axes, Execute program 2

FBK

Set Open or Closed Loop Mode

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓		✓

Command Description:	This command is used to select the feedback mode of the controller. See section 4.2 for more details
Returns:	A read operation returns the following loop mode values for the specified axis: 0 – Open Loop [default] 2 – Clean Open Loop Movement, Closed Loop deceleration 3 – Closed Loop
Syntax:	nFBKx – Standard syntax nFBK? – Read encoder mode value Error [#]: FBKx – Missing axis number [30] FBK? – Read operation with missing axis number [27] nFBK – Missing closed/open loop parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Open/closed loop mode ? – Read encoder mode value
Parameter Range:	n – 1 to 99 x – 0 for open loop mode, 2 for open loop with closed loop deceleration, 3 closed loop
Related Commands:	ENC, EAD, EPL, DBD
Example:	`2FBK3` | Axis 2, Set closed loop mode

FFP

Feed Forward Parameter

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓				

Command Description:	This command is used to set the feedforward parameter of the controller. This is used when the controller is operating in feedback mode 2 or 3.
Returns:	A read operation returned the current feed forward parameter
Syntax:	nFFPx – Standard syntax nFFP? – Read encoder mode value Error [#]: FFPx – Missing axis number [30] FFP? – Read operation with missing axis number [27] nFFP – Missing closed/open loop parameter [28]
Parameter Description:	n[int] – Axis number x[int] – Feed forward parameter ? – Read encoder mode value
Parameter Range:	n – 1 to 99 x – 0 to 65,536
Related Commands:	PID, FBK
Example:	`2FFP1000` | Axis 2, Set feed forward parameter

HCG Home Configuration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to select the direction of motion when the Home [HOM] command is initialized.
Returns:	A read operation returns the current direction setting: 0 – Home starts in the direction of the negative limit 1 – Home starts in the direction of the positive limit
Syntax:	nHCGx – Standard syntax 0HCGx – All axes set direction nHCG? – Read direction setting Error [#]: HCG? – Read operation with missing axis number [27] nHCG – Missing direction setting [28]
Parameter Description:	n[int] – Axis number x [int] – Set direction of motion.
Parameter Range:	n – 0 to 99 x – 0 for setting motion in the direction of the negative limit 1 for setting motion in the direction of the positive limit
Related Commands:	HOM
Example:	`3HCG0` | Axis 3, Set initial direction of Home command towards the negative limit – `0HCG1` | All Axes, Set initial direction of Home command towards the positive limit

HOM Home

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to find the home (zero) position for a specified axis. An error will occur if there is no encoder signal at the time of execution. Home is configured using the HCG command. This command will jog the stage till it reaches the limit configured by the HCG command. It will then acquire the zero position by looking for the index. This command blocks all communication over the serial port during motion. The controller will buffer all commands sent during this period and execute them once the command has found the index. Caution: if you write too many commands while this command is executing you run the risk of overloading the receive buffer.
Returns:	A read parameter returns the following calibration values for the specified axis: 0 – Not calibrated to home position 1 – Calibrated to home position
Syntax:	nHOM – Standard syntax nHOM? – Returns 1 if homed since last startup otherwise returns 0 0HOM – All axes execute home position Error [#]: HOM? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	HCG
Example:	`1HOM` | Axis 1, Move to home position

HST

Hard Stop Detection

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓		✓	✓	✓	✓	✓	

Command Description:	This command is used to enable hard stop detection, which stops motion when the stage reaches a hard stop.
Returns:	A read operation returns the following hard stop detection off/on values for the specified axis: 0 – Hard Stop Detection is off 1 – Hard Stop Detection is on
Syntax:	nHSTx – Standard syntax nHST? – Read Hard Stop Detection off/on value 0HSTx – All axes set Hard Stop Detection value Error [#]: HST? – Read operation with missing axis number [27] xHST – Missing motor off/on parameter [28]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	STP
Example:	`8HST1` | Axis 8, Hard Stop Detection on – `0HST0` | All axes, Hard Stop Detection off

INP

In Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓			✓	

Command Description:	This command is used to set the in-position range and time requirements. If both the in-position range and time are met by the controller, it is "in position" and bit 3 of the STA register will be set. The in-position range refers to the maximum number of encoder counts (±) from the target that is considered acceptable. The in-position time refers to the number of seconds the stage must be within the in-position range. The in-position feature is applicable in feedback modes FBK2 and FBK3. If either parameter of in-position is set to zero, the deadband setting (DBD) will be used to determine bit 3 of the STA register.
Returns:	A read operation returns the values assigned to the In-position range and time parameters
Syntax:	nINPx1,x2 – Standard syntax 0INPx1,x2 – All axes execute acceleration value nINP? – Read in-position range and time values Error [#]: INP? – Read operation with missing axis number [27] nINP – Missing in-position range and timeout parameter values [28]
Parameter Description:	n[int] – Axis number x1[int] – In-position Range x2[int] – In-position time ? - Read in-position range and timeout values
Parameter Range:	n – 0 to 99 x1 – Encoder counts, 0 to disable in-position feature x2 – Seconds, 0 to disable in-position feature
Related Commands:	ENC, DBD, FBK, STA
Example:	`4INP5,1.5` | Axis 4, set in-position range to ±5 encoder counts. Controller must stay in-position for 1.5 seconds to be in-position

IOF

Set IO Function

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓	✓	✓	✓	✓	✓

Command Description:	This command is used to select the function of an IO pin.
Returns:	A read operation is not available with this command.
Syntax:	nIOFx1,x2 – Standard syntax nIOF? – Read encoder mode value Error [#]: IOFx1,x2 – Missing axis number [30] IOF? – Read operation with missing axis number [27] nIOF – Missing closed/open loop parameter [28]
Parameter Description:	n [int] – Axis number x1[int] – IO Pin x2 [int] – IO Function ? – Read encoder mode value
Parameter Range:	n – 1 to 99 x1 – 1 – IO1 2 – IO2 3 – IO3 4 – IO4 5 – IO5 6 – IO6 x2 – 0 – No function 1 – Trace data acquisition on trigger (TRA) 2 – Output pulse trigger when in position 3 – Output level when motion ends 4 – Output pulse trigger at constant velocity (CVP) 5 – Output pulse when position is reached (PTP) 6 – Output pulse at regular intervals (PIP)
Related Commands:	IOF, TRA, CVP, PTP, PIP, IOS
Example:	`2IOF2,1` | Axis 2, Set IO2 to data logging trigger

IOP

Set IO Polarity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓				

Command Description:	This command is used to select the polarity of an IO pin.
Returns:	A read operation returns the polarity values assigned to the IO Pins.
Syntax:	nIOPx1,x2 – Standard syntax nIOP? – Read polarity value Error [#]: IOPx1,x2 – Missing axis number [30] IOP? – Read operation with missing axis number [27]
Parameter Description:	n [int] – Axis number x1 [int] – IO Pin x2 [int] – IO Polarity ? – Read all IO pins polarity value
Parameter Range:	n – 1 to 99 x1 – 1 – IO1 2 – IO2 3 – IO3 4 – IO4 5 – IO5 6 – IO6 x2 – 0 – Active Low Pulses (5 to 0V pulses) 1 – Active High Pulses (0 to 5V pulses)
Related Commands:	IOF, TRA, CVP, PTP, PIP, IOS
Example:	`2IOP2,1` | Axis 2, Set IO2 to Active High

IOS

Set IO Status

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓				

Command Description:	This command is used to select the output or read input of an IO pin. *This command supersedes all IO commands (IOF, IOP, CVP, PTP, and PIP)
Returns:	A read operation returns the output values assigned to the IO Pins.
Syntax:	nIOSx1,x2 – Standard syntax nIOS? – Read input/output value Error [#]: IOSx1,x2 – Missing axis number [30] IOS? – Read operation with missing axis number [27]
Parameter Description:	n [int] – Axis number x1 [int] – IO Pin x2 [int] – Output ? – Read input/output value
Parameter Range:	n – 1 to 99 x1 – 1 – IO1 2 – IO2 3 – IO3 4 – IO4 x2 – 0 – Output Off 1 – Output On
Related Commands:	IOF, IOP
Example:	`2IOS2,1` | Axis 2, Set IO2 to output

IWL

Integrator Windup Limit

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓				

Command Description:	This command sets the integrator windup limit during feedback mode 3. This setting will effectively limit the max allowable error for the integrator term of the PID feedback loop.
Returns:	A read operation will return the current integrator windup limit.
Syntax:	nIWLx1 – Standard syntax nIWL? – Read output value Error [#]: IWLx1 – Missing axis number [30] IWL? – Read operation with missing axis number [27]
Parameter Description:	n [int] – Axis number x1 [int] – Integrator Windup Value ? – Read output value
Parameter Range:	n – 1 to 99 x1 – 0 to 32-bit positive integer
Related Commands:	FBK, PID
Example:	`2IWL1000` | Axis 2, Set integrator windup value

JAC

Jog Acceleration and Deceleration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired value for the jog acceleration and deceleration for a specified axis. The controller will not allow for JAC values that are greater than AMX.
Returns:	A read operation returns the jog acceleration and deceleration value in mm/s^2 for the specified axis.
Syntax:	nJACx – Standard syntax 0JACx – All axes execute acceleration value nJAC? – Read acceleration value Error [#]: JAC? – Read operation with missing axis number [27] nJAC – Missing acceleration parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Acceleration ? – Read acceleration value
Parameter Range:	n – 0 to 99 x – .001 to AMX
Related Commands:	ACC, DEC, AMX
Example:	`4JAC0.1` | Axis 4, Set jog acceleration & deceleration to 0.1 mm/s^2 [degrees/s^2]

JOG

Jog Mode

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓		✓	✓	✓	✓	✓	✓

Command Description:	This command is used to jog a specific axis, or move continuously in a direction with no target position. The jog velocity is a percentage of the maximum velocity and may be changed on-the-fly by sending another JOG command during motion.
Returns:	A read operation is not available with this command.
Syntax:	nJOGx – Standard syntax Error [#]: JOGx – Missing axis number [30] nJOG – Missing velocity parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Velocity
Parameter Range:	n – 1 to 99 x – 0.001 to 100.000 % (of maximum velocity)
Related Commands:	JAC
Example:	`4JOG10` | Axis 4, Jog at 10% maximum velocity

LCG

Limit Configuration

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command selects whether the limit switch inputs on the motor connector are ignored, otherwise it will stop motion.
Returns:	A read operation is not available with this command.
Syntax:	nLCGx – Standard syntax Error(s): LCGx – Missing axis number [30] nLCG – Missing program number parameter [28]
Parameter Description:	n[int] – Axis number x[int] – 0 – ignore [default] 1– active
Parameter Range:	n – 1 to 99 x – 0 – ignore [default] 1 – Soft Limits Only 2 – Limit Switches Only 3 – Limit Switches and Soft Limits enabled
Related Commands:	LPL
Example:	`1LCG1` | Axis 1, set limit switches active

LDP

Load Parameters

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓				✓	

Command Description:	This command is used to restore the set of most recently saved settings for the specified axis. This allows the user to return to a user set predefined state.
Returns:	A read operation cannot be used with this command.
Syntax:	nLDP – Standard syntax 0LDP – All axes restore saved settings
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	None
Example:	`1LDP` | Axis 1, restore saved settings

LDR

Positive/ Negative Limit Location

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	Determines orientation of Positive limit, and negative limit.
Returns:	A read operation returns the following limit direction values for the specified axis: 0 – Normal orientation 1 – Reverse orientation
Syntax:	nLDRx – Standard syntax nLDR? – Read velocity value 0LDRx – Missing axis number, all axes set limit direction Error [#]: LDR? – Read operation with missing axis number [27] nLDR – Missing limit parameter [28]
Parameter Description:	n[int] – Axis number x[int] – limit direction value ? – Read limit direction value
Parameter Range:	n – 0 to 99 x – 0 or 1
Related Commands:	
Example:	`1LDR1` | Axis 1, set to reverse orientation – `5LDR?` | Axis 5, Read limit switch orientation

LPL

Limit Switch Polarity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command sets whether the limit switch inputs are active high[1] or low[0]
Returns:	A read operation returns the program table for the specified axis.
Syntax:	nLPLx – Standard syntax Error(s): LPLx – Missing axis number [30] nLPL – Missing program number parameter [28]
Parameter Description:	n[int] – Axis number x – 0 –Active Low – 1 – Active High
Parameter Range:	n – 1 to 99 x – 0 – active low [default] 1– active high
Related Commands:	LCG
Example:	`6LPL1` | Axis 5, limit switches set to active high

LST

Program List

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
			✓				

Command Description:	This command is used to display the stored commands that make up a given internal program.
Returns:	A list of the commands in order as executed when the given program is run.
Syntax:	nLSTx – Standard syntax Error [#]: 1LST? – Read Not Available For This Command [38]
Parameter Description:	n[int] – Axis number x[int] – Program# to be read
Parameter Range:	n – 1 to 99 x – 1 to 16
Related Commands:	None
Example:	`6LST1` | Axis 6, return program 1 list of commands

MLN

Move to Negative Limit

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command initiates a move to the negative limit position. Upon reaching the negative hard limit the controller will then move the stage back from the hard limit and stop. An error will occur if there is no encoder signal at the time of execution.
Returns:	A read operation is not available with this command.
Syntax:	nMLN – Standard syntax 0MLN – All axes execute move to negative limit position Error [#]: MLN – Missing axis number [30]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	MLP
Example:	`8MLN` | Axis 8, Move to negative limit position – `0MLN` | All Axes, Move to negative limit position

MLP

Move to Positive Limit

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command initiates a move to the positive limit position. Upon reaching the positive hard limit the controller will then move the stage back from the hard limit and stop. An error will occur if there is no encoder signal at the time of execution.
Returns:	A read operation is not available with this command.
Syntax:	nMLP – Standard syntax 0MLP – All axes execute move to positive limit position Error [#]: MLP – Missing axis number [30]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	MLN
Example:	`1MLP` | Axis 1, Move to positive limit position – `0MLP` | All Axes, Move to positive limit position

MOT

Toggle Motor Off/On

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to turn the motor current flow "Off" or "On" for a specified axis. Turning the motor current off will cause the piezo to relax and the stage will shift slightly.
Returns:	A read operation returns the following motor current off/on values for the specified axis: 0 – Motor current is off 1 – Motor current is on
Syntax:	nMOTx – Standard syntax nMOT? – Read motor current off/on value 0MOTx – All axes set motor value Error [#]: MOT? – Read operation with missing axis number [27] xMOT – Missing motor off/on parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Motor current off/on ? – Read motor current off/on value
Parameter Range:	n – 0 to 99 x – 0 for motor current off 1 for motor current on
Related Commands:	None
Example:	`1MOT0` | Axis1, Set motor current to off

MPL

Toggle Motor Polarity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command set the motor polarity for the specified axis. If the theoretical positive direction is away from the motor, changing this setting will make the theoretical positive direction towards to motor.
Returns:	A read operation returns the current motor polarity setting for the specified axis.
Syntax:	nMPLx – Standard syntax nMPL? – Read motor current off/on value 0MPLx – All axes set motor value Error [#]: MPL? – Read operation with missing axis number [27] nMPL – Missing motor off/on parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Motor Polarity setting ? – Read motor current off/on value
Parameter Range:	n – 0 to 99 x – 0 Normal 1 Reverse
Related Commands:	MVR
Example:	`1MPL0` | Axis1, To normal Polarity

MSA

Synchronous Move - Absolute

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to set up a synchronous move using the absolute position of the axes involved. This command is most useful when coordinating motion to an absolute position between 2 or more axes and requires a RUN command on a separate line to execute the synchronous move. It is recommended to run multiple MSA commands on the same command line, as they are executed closer together than on separate lines. An error will occur if the commanded position is outside of the soft limits.
Returns:	A read operation is not available with this command.
Syntax:	nMSAx – Standard syntax 0MSAx – All axes execute synchronous move Error [#]: nMSA – Missing absolute position parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Absolute position
Parameter Range:	n – 0 to 99 x – 0.000000 to 999.999999 mm (degrees)
Related Commands:	RUN, MSR
Example:	`1MSA10;2MSA10` | Axis 1, Move to absolute position: 10 mm[degrees]; Axis 2, Move to absolute position: 10 mm [degrees] *`0RUN`* | All axes, Execute synchronous move - `0MSA5` | All axes, Move to absolute position: 5 mm [degrees] *`0RUN`* | All axes, Execute synchronous move

MSR

Synchronous Move – Relative

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to set up a relative move using the relative position of the axes involved. This command is most useful when coordinating relative positions between 2 or more axes and requires a RUN command on a separate line to execute the synchronous move. It is recommended to use multiple MSR commands on the same command line, as they are executed closer together than on separate lines. An error will occur if the commanded increment will cause the stage to travel outside of the set soft limits.
Returns:	A read operation is not available with this command.
Syntax:	nMSRx – Standard syntax 0MSAx – All axes execute synchronous move Error [#]: nMSA – Missing relative position parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Relative position
Parameter Range:	n – 0 to 99 x – 0.000000 to 999.999999 mm (degrees)
Related Commands:	RUN, MSA
Example:	`4MSR.1;5MSR.5` | Axis 4, Move 0.1 mm [degrees]; Axis 5, Move 0.5 mm [degrees] `0RUN` | Execute synchronous move – `0MSR0.01` | All axes, Move 0.01 mm [degrees] `0RUN` | All axes, execute synchronous move

MVA

Move Absolute

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to initiate an instantaneous move to an absolute position for a specified axis. An error will occur if the commanded position is outside of the soft limits.
Returns:	A read operation is not available with this command.
Syntax:	nMVAx – Standard syntax 0MVAx – All axes execute instantaneous move Error(s): nMVA – Missing absolute position parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Absolute position
Parameter Range:	n – 0 to 99 x – 0.000000 to ± 999.999999 mm (degrees)
Related Commands:	MVR, WFS
Example:	`4MVA14.5` | Axis 4, Move to absolute position: 14.5 mm [degrees] – `0MVA5.5` | All axes, Move to absolute position: 5.5 mm [degrees]

MVR

Move Relative

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to initiate an instantaneous move to a relative position for a specified axis. An error will occur if the commanded increment will cause the stage to travel outside of the set soft limits.
Returns:	A read operation is not available with this command.
Syntax:	nMVRx – Standard syntax 0MVRx – All axes execute command. Error(s): nMVR – Missing relative position parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Relative position
Parameter Range:	n – 0 to 99 x – 0.000000 to ± 999.999999 mm [degrees]
Related Commands:	MVR, WFS
Example:	`6MVR10` | Axis 6, Move 10 mm [degrees] – `0MVR.89` | All axes, Move 0.89 mm [degrees]

PGL

Loop Program

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to change the program loop setting. If the program loop flag is set, any program that is executed will run in a continuous loop. It can be combined with the PGS command to run a program continuously on startup. A looping program can be stopped at any time by sending a STP command to the controller
Returns:	A read operation returns the program loop setting for the specified axis.
Syntax:	nPGLx1,x2 – Standard syntax nPGLx1? – Read Error(s): PGLx1,x2 – Missing axis number [30] nPGL – Missing program number parameter [28]
Parameter Description:	n[int] – Axis number x1[int] – Program to be looped x2[int] – Number of repetitions
Parameter Range:	n – 1 to 99 x1 – 1 to 32 x2 – 0 for infinite repetition – 1 to 99
Related Commands:	PGS, STP
Example:	1PGL1,0 | Axis 1, Program 1 will run continuously 8PGL5,3 | Axis 8, Program 5 will run 3 times 2PGL3? | Axis 2, read number of loops of program 3

PGM

Begin Program Recording

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓		✓

Command Description:	This command is used to enter program recording mode for a specified axis. The program being recorded must use a unique program number or else the program will be ignored. Us the LST command to check program number availability and use the ERA command to erase any previously recorded programs. Each program has a size limit of 4Kb.
Returns:	A read operation returns the program table for the specified axis.
Syntax:	nPGMx – Standard syntax nPGM? – Read a binary representation of written program numbers If programs 1 and 2 are written it will return 3 If programs 1 and 4 are written it will return 9 If only program 1 is written it will return 1 If only program 3 is written it will return 4 Error(s): PGMx – Missing axis number [30] nPGM – Missing program number parameter [28]
Parameter Description:	n[int] – Axis number x[int] – Program number to be recorded
Parameter Range:	n – 1 to 99 x – 1 to 32
Related Commands:	END, EXC, LST, ERA
Example:	`1PGM3` | Axis 1, Begin recording program. Save program as program 3

PGS

Run Program At Start-Up

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓	✓		✓	✓	✓

Command Description:	This command is used to set a program to run immediately on start-up. Only one program per axis can run on start up. NOTE: The PGS value must be saved using the SAV command prior to power down for a program to run on startup.
Returns:	A read operation returns a value for the specified axis in the format below: 0 – No program set to run 1-32 – Program set to run on start-up
Syntax:	nPGSx – Standard syntax 0PGSx – Missing axis number, all axes set program to run on start-up nPGS? – Read program(s) set to run on start-up Error [#]: PGS? – Read operation with missing axis number [27] nPGS – Missing program set to run on start-up parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Program set to run on start-up ? – Read encoder mode value
Parameter Range:	n – 0 to 99 x – 0 - No program 1 to 32 - Specific program set to run on start-up
Related Commands:	LST, PGM
Example:	`6PGS5` | Axis 6, set program 5 to run on start-up – `0PGS16` | All axes, set program 16 to run on start-up – `3PGS?` | Axis 3, Read program to run on start-up – `3PGS0` | Axis 3, Set no program to run on start-up

PID

Set Feedback Constants

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓		✓

Command Description:
This command is used to set the encoder feedback constants for a specified controller.

Returns:
A read operation returns the encoder feedback constant values for the specified axis.

Syntax:

nPIDx – Standard syntax
nPID? – Read encoder feedback constant values

Error(s):
PIDx – Missing axis number [30]
PID? – Read operation with missing axis number [27]
nPID – Missing encoder feedback constant parameters [28]

Parameter Description:

n[int] – Axis number
x1[float] – K_p (proportional constant)
? – Read encoder feedback constants and values

Parameter Range:

n – 1 to 99
x1 – 0.000 to 2.000
x2 – 0.000 to 2.000
x3 – 0.000 to 2.000

Related Commands: FBK, ENC, POS, FFP, IWL

Example:

```
5PID.02        | Axis 5, Set encoder feedback constant to 0.02.

3PID?          | Axis 3, Read PID settings
```

PIP

Pulse at Regular Intervals

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓			✓	

Command Description:

This command is used in conjunction with IOF and IOP to command the controller to emit a pulse at specified regular intervals when given a start point, an interval distance, and an end point.

PIP operation is available on IO pons 1, 4 and 5.

PIP operation is a one-time use and will end after finishing all pulse triggers.

Returns:

A read operation returns the values assigned to the start point, interval distance, and end point.

Syntax:

nPIPx1,x2,x3 – Standard syntax
0 PIPx1,x2,x3 – All axes execute position and interval value
nPIP? – Read interval pulse values

Error [#]:
1PIP1,-0.3,2 Invalid Input Parameter (interval does not have the same sign as end position minus start position).

Parameter Description:

n[int] – Axis number
x1[float] – Beginning position for PIP command
x2[float] – Desired pulse interval
x3[float] – Ending position for PIP command

Parameter Range:

n – 0 to 99
x1– -400.000 to 400.000 mm
x2 – -400.000 to 400.000 mm
x3 – -400.000 to 400.000 mm

Related Commands: IOF, IOP

Example:

```
4IOP1,0          | Axis 4, Set IO Pin 1's pulse as active low
4IOF1,6          | Axis 4, Set IO Pin 1 for PIP function
4PIP1,0.2,5      | Axis 4, Pulse triggers at IO pins with PIP every
                   0.2mm from 1mm to 5mm.

2IOP4,1          | Axis 2, Set IO Pin 4's pulse as active high
2IOF4,6          | Axis 2, Set IO Pin 4 for PIP function
2PIP-2,-0.5,-6   | Axis 2, Pulse triggers at IO pin with PIP every
                   0.5mm from -2mm to -6mm.
```

POS

Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓		✓

Command Description:	This command is used to read the position information from the specified axis controller
Returns:	A read operation returns the position values in mm for the specified axis in the following format: [Theoretical position in mm, Encoder position in mm] [Theoretical position in degrees, Encoder position in degrees]
Syntax:	nPOS? – Standard syntax Error(s): POS? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number ? – Read position values
Parameter Range:	n – 1 to 99
Related Commands:	MVR
Example:	`4POS?` | Axis 4, Read position values

PTP

Pulse at Target Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓			✓	

Command Description:	This command is used in conjunction with IOF and IOP to command the controller to emit a pulse at a specified position, from a specified direction (or from both directions). PTP operation is available on IO pins 1, 4 or 5.
Returns:	A read operation returns the values assigned to the trigger position and trigger direction.
Syntax:	nPTPx1,x2 – Standard syntax 0 PTPx1,x2 – All axes execute position and pulse value nPTP? – Read interval pulse values Error [#]: 1PTP1,3 Invalid Input Parameter ("3" is not an accepted value for "trigger direction").
Parameter Description:	n[int] – Axis number x1[float] – Trigger Position x2[int] – Trigger direction
Parameter Range:	n – 0 to 99 x1– -400.000 to 400.000 mm x2 – 0 triggers when moving negative 1 triggers when moving positive 2 triggers from both sides
Related Commands:	IOF, IOP
Example:	`4IOP4,1` | Axis 4, Set IO Pin 4's pulse as active high `4IOF4,5` | Axis 4, Set IO Pin 4 for PIP function `4PTP5.73,2` | Axis 4, Pulse triggers at IO pins with PTP upon reaching 5.73mm from either direction.

REZ

Set Resolution

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓		✓	✓	✓

Command Description:	This command is used to set the DAC (digital to analog converter) steps per micron resolution for the specified axis.
Returns:	A read operation returns the resolution value in steps per micron for the specified axis.
Syntax:	nREZx – Standard syntax nREZ? – Read steps per micron resolution value Error(s): REZ? – Read operation with missing axis number [27] REZx – Missing axis number [30] nREZ – Missing steps per micron resolution parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Steps per micron resolution (steps/mili-degrees) (default is 8,000) ? – Read steps per micron resolution value (steps/milli-degrees)
Parameter Range:	n – 1 to 99 x – 0 to 99999 DAC steps per micron (steps/milli-degrees) //This allows for 2K AMX, should this be capped?
Related Commands:	None
Example:	`9REZ25` | Axis 9, Set resolution to 25 steps/micron [steps/milli-degrees] – `3REZ?` | Axis 3, Read steps/micron [steps/degrees] resolution value

RUN

Start Synchronous Move

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to start a global synchronous move previously set up by using the MSA or MSR commands.
Returns:	A read operation cannot be used with this command.
Syntax:	RUN – Standard syntax
Parameter Description:	-
Parameter Range:	-
Related Commands:	MSA, MSR
Example:	`3MSR5;4MSR5` | Axis 3, setup 5 mm[degrees] move; Axis 4, setup 5 mm [degrees] move `0RUN` | All axes, Execute synchronous moves

SAV

Save Axis Settings

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓				✓	

Command Description:	This command is used to save all settings for the specified axis. This allows an axis to be configured on power up.
Returns:	A read operation cannot be used with this command.
Syntax:	nSAV – Standard syntax 0SAV – All axes save settings
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	None
Example:	`16SAV` | Axis 16, save settings

STA

Status Byte

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓		✓

Command Description: This command is used to check the status register for a specified axis.

Returns:

A read operation will return an integer from 0 to 255 describing the status of the axis. The byte must be decoded in binary to determine the value of each bit.

Bit	7	6	5	4	3	2	1	0
Name	ERR	ACC	CNST	DEC	STP	PGM	PLS	NLS

Bit 7: 1 – One or more errors have occurred. Use ERR? Or CER to clear.
0 – No Errors have occurred.

Bit 6: 1 – Currently in Acceleration phase of motion.
0 – Not in Acceleration phase of motion.

Bit 5: 1 – Currently in Constant Velocity phase of motion.
0 – Not in Constant Velocity phase of motion.

Bit 4: 1 – Currently in Deceleration phase of motion.
0 – Not in Deceleration phase of motion.

Bit 3: 1 – Stage has stopped. (In Closed Loop Stage, is in the deadband) OR in position
0 – Stage is moving. (In Closed Loop, Stage is out of deadband) OR not in position

Bit 2: 1 – A Program is currently running
0 – No program is running

Bit 1: 1 – Positive Switch is Activated
0 – Positive Switch is not Activated

Bit 0: 1 – Negative Switch is Activated
0 – Negative Switch is not Activated

Syntax: nSTA? – Standard syntax
Error(s):
STA? – Read operation with missing axis number [27]
nSTA – Missing read operation parameter [28]

Parameter Description:
n[int] – Axis number
? – Read status register

Parameter Range: n – 1 to 99

Related Commands: INP

Example:
```
6STA?
```
| Axis 6, Read status register

STP

Stop Motion

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓		✓				✓	

Command Description:	This command is used to stop motion for a specified axis.
Returns:	A read operation cannot be used with this command.
Syntax:	nSTP – Standard syntax 0STP – All axes execute stop
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	EST, DEC
Example:	`8STP` | Axis 8, execute stop

SVP

Save Startup Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the startup position. Default is 0. This setting does not require the SAV command to save it into memory. It also does not change with a DEF command. To reset the Startup position to the default, send nSVP0.
Returns:	A read operation returns the Startup position setting for the specified axis.
Syntax:	nSVP – Standard syntax 0SVP – Missing axis number, command accepted as standard syntax
Parameter Description:	n[int] – Axis number x[float] – Startup Position mm ? – Read Startup Position
Parameter Range:	n – 0 to 99 x – TLN (-999.999999mm) to TLP(999.999999mm)
Related Commands:	None
Example:	`4SVP` | Set current position to Startup position `2SVP2.3` | Set startup position to 2.3mm

SYN

Sync

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
				✓		✓	

Command Description:	This command is used in a program together with the wait for sync [WSY] command in order to synchronize motion between multiple axes. Command must be preceded by a % to be considered part of the current program.
Returns:	A read operation cannot be used with this command.
Syntax:	%nSYN – Standard syntax 0SYN – Missing axis number, command accepted as standard syntax
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 0 to 99
Related Commands:	WSY
Example:	`4SYN` | Send sync to axis 4

TLN

Negative Soft Limit Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired negative soft limit position, using absolute position, for the specified axis. The negative soft limit position value must be less than the positive soft limit position value [TLP] for the command to be accepted.
Returns:	A read operation returns the negative soft limit position value.
Syntax:	nTLNx – Standard syntax nTLN? – Read negative soft limit position value 0TLNx – All axes set limit position value nTLN – Set current position to negative limit Error(s): TLN? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number x[float] – Negative soft limit position ? – Read negative soft limit position
Parameter Range:	n – 0 to 99 x – -999.999999 to TLP(999.999999) mm [degrees]
Related Commands:	TLP
Example:	`2TLN0.005` | Axis 2, Set negative soft limit position to 0.005 mm [degrees] – `6TLN?` | Axis 6, Read negative soft limit position value

TLP

Positive Soft limit Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired positive soft limit position, using absolute position, for the specified axis. The positive soft limit position value must be greater than the negative soft limit position value [TLN] for the command to be accepted.
Returns:	A read operation returns the positive soft limit position value for the specified axis.
Syntax:	nTLPx – Standard syntax nTLP? – Read positive soft limit position value 0TLPx – All axes set limit position value nTLN – Set current position to negative limit Error(s): TLP? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number x[float] – Positive soft limit position ? – Read positive soft limit position
Parameter Range:	n – 0 to 99 x – TLN(-999.999999mm)to + 999.999999 mm [degrees]
Related Commands:	TLN
Example:	`4TLP10.005` | Axis 2, Set positive soft limit position to 10.005 mm [degrees] – `9TLP?` | Axis 9, Read positive soft limit position value

TRA

Perform Trace

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓	✓	✓	✓		✓	

Command Description:	This command is used to execute a trace of the specified axis.
Returns:	A read operation returns the position samples taken for the specified axis.
Syntax:	nTRAx1,x2,x3 – Standard syntax nTRA? – Read position values 0TLPx1,x2,x3 – All axes execute trace Error(s): TRA? – Read operation with missing axis number [27] nTRA – Missing parameters [28]
Parameter Description:	n[int] – Axis number x1[int] – Number of samples taken (default is 1000) x2[int] – 10kHz /Sampling frequency (default is 1) x3[float] – Trace starting position (default is immediate) ? – Read position
Parameter Range:	n – 0 to 99 x1 – 1 to 9000 x2 – 1 to 1000 Servo clocks per cycle x3 – 000.000001 to 999.999999 mm [degrees]
Related Commands:	DAT
Example:	`5TRA5,10,1` | Axis 5, execute trace with 5 samples at a sampling frequency of 1kHz starting at a position of 1 mm [degrees] `3TRA2000,,` | Axis 3, execute trace with 2000 samples at a sampling frequency of 10kHz starting at the current position

VEL

Velocity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
✓	✓	✓	✓	✓	✓	✓	✓

Command Description:	This command is used to set the desired velocity for the specified axis. The velocity may be changed on-the-fly by sending another VEL command during motion. The velocity value should be lower than the maximum allowable velocity [VMX] for the command to be accepted.
Returns:	A read operation returns the velocity value in mm/s for the specified axis.
Syntax:	nVELx – Standard syntax nVEL? – Read velocity value 0VELx – Missing axis number, all axes set velocity Error [#]: VEL? – Read operation with missing axis number [27] nVEL – Missing velocity parameter [28]
Parameter Description:	n[int] – Axis number x[float] – Velocity value ? – Read velocity value
Parameter Range:	n – 0 to 99 x – 000.000001 to VMX
Related Commands:	VMX, REZ
Example:	`1VEL.25` | Axis 1, Set velocity to 0.25mm/s [degrees/s] `5VEL?` | Axis 5, Read velocity value

VER

Firmware Version

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓		✓

Command Description:	This command is used to check the firmware version for the specified axis.
Returns:	A read operation returns the firmware version for the specified axis.
Syntax:	nVER? – Standard syntax Error(s): VER? – Read operation with missing axis number [27] nVER – Missing read operation parameter [28]
Parameter Description:	n[int] – Axis number ? – Read firmware version
Parameter Range:	n – 1 to 99
Related Commands:	None
Example:	`11VER?` | Axis 11, Read firmware version

VMX

Maximum Allowable Velocity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓		✓

Command Description:	This command is used to read the maximum allowable velocity for a specific axis. This value is calculated based on the steps per micron parameter in the REZ command.
Returns:	A read operation returns the maximum allowable velocity value in mm/s for the specified axis.
Syntax:	nVMX? – Read maximum allowable velocity value Error [#]: VMX? – Read operation with missing axis number [27] nVMX – Missing read operation parameter [123]
Parameter Description:	n[int] – Axis number ? – Read maximum allowable velocity value
Parameter Range:	n – 1 to 2000
Related Commands:	REZ, VEL
Example:	`4VMX?` | Axis 4, Read maximum allowable velocity value

VRT

Encoder Velocity

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
	✓		✓		✓		✓

Command Description:	This command returns the actual velocity calculated from the encoder.
Returns:	A read operation returns the encoder velocity in mm/s.
Syntax:	nVRT? – Standard syntax Error [#]: VRT? – Read operation with missing axis number [27]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	POS
Example:	`5VRT?` | Axis 5, Read encoder velocity

WST

Wait For Stop

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
				✓			

Command Description:	This command is used in a program to wait until motion is completed to begin executing the next command. Command must be preceded by a % to be considered part of the current program.
Returns:	A read operation cannot be used with this command.
Syntax:	%nWST – Standard syntax WST – Missing axis number, command accepted as standard syntax
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	PGM
Example:	`%7WST` | Axis 7, Wait for motion to stop before executing next command

WSY

Wait For Sync

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
				✓		✓	

Command Description:	This command is used in a program together with the sync [SYN] command in order to synchronize motion between multiple axes. Command must be preceded by a % to be considered part of the current program.
Returns:	A read operation cannot be used with this command.
Syntax:	%nWSY – Standard syntax WSY – Missing axis number, command accepted as standard syntax
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	SYN
Example:	`%1WSY` | Axis 1, Wait until sync command is received before executing next command

WTM

Wait For Time Period

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
				✓			

Command Description:	This command is used in a program to wait for a specified period of time before executing the next command. Command must be preceded by a % to be considered part of the current program.
Returns:	A read operation cannot be used with this command.
Syntax:	%nWTMx – Standard syntax WSTx – Missing axis number, command accepted as standard syntax
Parameter Description:	n[int] – Axis number x[int] – Time
Parameter Range:	n – 1 to 99 x – 0 to 999999 milliseconds
Related Commands:	PGM
Example:	`%2WTM42` | Axis 2, Wait for 42 milliseconds before executing next command

ZRO

Zero Position

During Motion		Real-time		Program		Global	
Set	Read	Set	Read	Set	Read	Set	Read
		✓		✓		✓	

Command Description:	This command is used to set the absolute zero position for the specified axis.
Returns:	A read operation cannot be used with this command.
Syntax:	nZRO – Standard syntax Error [#]: ZRO – Missing axis number [123]
Parameter Description:	n[int] – Axis number
Parameter Range:	n – 1 to 99
Related Commands:	None
Example:	`1ZRO` | Axis 1, set current position as absolute zero

5.9 Error Messages

Error Number	Name	Description
10	Receive Buffer Overrun	The Receive Buffer has reached or exceeded maximum capacity.
11	Motor Disabled	The command that triggered this error was trying to move the servo while it was disabled.
12	No Encoder Detected	The command that triggered this error was trying to access encoder data when no encoder was attached.
13	Index Not Found	The controller moved across the full range of motion and did not find an index.
14	Home Requires Encoder	The HOM command requires an encoder signal.
15	Move Limit Requires Encoder	The MLN and MLP commands require an encoder signal.
20	Command is Read Only	The command that triggered this error only supports read operations. The command must be followed by a question mark to be accepted. Ex: XXX?
21	One Read Operation Per Line	Multiple read operations on the same command line. Only one read operation is allowed per line, even if addressed to separate axes.
22	Too Many Commands On Line	The maximum number of allowed commands per command line has been exceeded. No more than 8 commands are allowed on a single command line.
23	Line Character Limit Exceeded	The maximum number of characters per command line has been exceeded. Each line has an 80 character limit.
24	Missing Axis Number	The controller could not find an axis number or the beginning of an instruction. Check the beginning of the command for erroneous characters.
25	Malformed Command	The controller could not find a 3-letter instruction in the input. Check to ensure that each instruction in the line has exactly 3 letters referring to a command.

26	Invalid Command	The 3-letter instruction entered is not a valid command. Ensure that the 3-letter instruction is a recognizable command.
27	Global Read Operation Request	A read request for a command was entered without an axis number. A read request cannot be used in a global context.
28	Invalid Parameter Type	1. The parameter entered does not correspond to the type of number that the instruction requires. For example, the command may expect an integer value, therefore sending a floating point value will trigger this error. 2. The allowable precision for a parameter has been exceeded. For example, velocity can be specified with a precision of 0.001 mm/sec. If a more precise velocity value of 0.0001 mm/sec is entered, this error will be triggered. Refer to the command pages for the type of parameter that each command expects.
29	Invalid Character in Parameter	There is an alpha character in a parameter that should be a numeric character.
30	Command Cannot Be Used In Global Context	The command entered must be addressed to a specific axis number. Not all commands can be used in a global context. Check the specific command page or the table of commands for more info.
31	Parameter Out Of Bounds	The parameter is out of bounds. The current state of the controller will not allow this parameter to be used. Check the command page for more information.
32	Incorrect Jog Velocity Request	The jog velocity can only be changed during motion by using a new JOG command. If the VEL command is used to change the velocity, this error will be triggered. The VEL command can only be used to change velocity during motion initiated by the move commands [MVR, MVA, MSR, MSA].
33	Not In Jog Mode	Sending a JOG command during motion initiated by a move command will trigger this error. To initiate Jog Mode, the controller should be at stand-still. To change velocity during a move, use the VEL command.
34	Trace Already In Progress	This error is triggered when a new trace command is received after a trace is already in progress. Trace settings may be modified only if the trace hasn't started recording data. Otherwise, wait until the trace has finished before modifying the trace settings.
35	Trace Did Not Complete	An error occurred while recording trace data. Try the operation again.

36	Command Cannot Be Executed During Motion	Only certain commands can be executed when motion is in progress. Check the command pages for information on individual commands.
37	Move Outside Soft Limits	If a requested move will take the controller outside of the preset travel limits, then the command will not be executed.
38	Read Not Available For This Command	This error is triggered by a read request from a command that does not support a read operation.
39	Program Number Out of Range	The number entered for the program number was either less than 1 or greater than 16.
40	Program Size Limit Exceeded	The program has exceeded the character limit of 4 Kb.
41	Program failed to Record	Error in recording program. Erase program and try operation again.
42	End Command Must Be on its Own Line	The End command used to end a program must be on a separate line from all other instructions.
43	Failed to Read Program	An error occurred while trying to read a program. Try the Operation again.
44	Command Only Valid Within Program	The command that triggered this error is only suitable for use within a program.
45	Program Already Exists	A program already exists for the indicated program parameter. The program must be erased with the ERA command before being written again.
46	Program Doesn't Exist	The indicated program does not exist. This error can occur when you try to execute a program number that has not had a program assigned to it.
47	Read Operations Not Allowed Inside Program	Read Operations are not permitted in programs.
48	Command Not Allowed While Program in Progress	The command that triggered this error was given while a program was executing.
50	Limit Activated	Motion in the direction of the activated limit switch is disallowed if limit switches are enabled.

51	End of Travel Limit	The requested move will take the controller outside of its valid travel range, therefore the move is disallowed.
52	Home In Progress	A Home or a Move To Limit Procedure is in progress. Motion commands are disallowed during this time. A STP or EST command can be used to terminate the Home, and then a motion command can be sent.
53	IO Function Already In Use	The I/O Function in question is already assigned to another I/O pin. Some Functions can only be assigned to one pin at a time. See the documentation for each function for more details.
55	Limits Are Not Configured Properly	Both Limit Switches are active, so motion is disallowed in both directions. Most likely the LPL (Limit Polarity command) setting should be switched.
80	Command Not Available in this Version	The command entered is not supported in this version of the firmware.
81	Analog Encoder Not Available In this Version	The current version of firmware installed does not support Analog Encoders.

6. Appendix

6.1 Encoder Input Pin-out

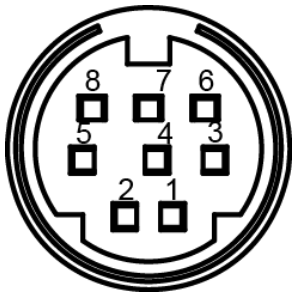
Pin	Description
1	A+/Cos+
2	B+/Sin+
3	Index +
4	Ground
5	+5V
6	A-/Cos-
7	B-/Sin-
8	Index -
9	Not In Use

6.2 Motor Input Pin-out

Pin	Description
1	Phase 1
2	Phase 2
3	N/C
4	N/C
5	Ground
6	Positive Limit
7	Negative Limit
8	Not In Use
9	GND

6.3 8-Pin Din IO connector

Pin8 - +5V
Pin7 – IO1 (output only)
Pin6 – IO2
Pin5 – IO3
Pin4 – GND
Pin3 – IO4
Pin2 – IO5
Pin1 – IO6



6.4 RS-232 RJ-11 Pin-out

Pin	Description
1	NC
2	GND
3	RX
4	TX
5	ID-Out (Not Used)
6	ID-In (Not Used)

6.5 Firmware

The firmware on the MMC-110 can be updated without returning the controller to manufacturer. For questions regarding the current firmware release and updating the firmware of your MMC-110 Please contact Micronix USA support. Please be sure to check the firmware version by using the VER command.