

Product Datasheet - Technical Specifications



More information in our Web-Shop at ► www.meilhaus.com and in our download section.

Your contact

Technical and commercial sales, price information,
quotations, demo/test equipment, consulting:

Tel.: **+49 - 81 41 - 52 71-0**

FAX: **+49 - 81 41 - 52 71-129**

E-Mail: sales@meilhaus.com

Downloads:

www.meilhaus.com/en/infos/download.htm

Meilhaus Electronic GmbH
Am Sonnenlicht 2
82239 Alling/Germany

| | |
|--------|--|
| Tel. | +49 - 81 41 - 52 71-0 |
| Fax | +49 - 81 41 - 52 71-129 |
| E-Mail | sales@meilhaus.com |

Mentioned company and product names may be registered trademarks of the respective companies. Prices in Euro plus VAT. Errors and omissions excepted.
© Meilhaus Electronic.

www.meilhaus.de



**USER'S MANUAL
INTELLIGENT MOTOR CONTROLLERS
UMX FAMILY**

OMS Motion, Inc
15201 NW Greenbrier Pkwy, B-1
BEAVERTON, OR 97006
PHONE 503-629-8081
FAX 503-629-0688
EMAIL: sales@omsmotion.com
<http://www.omsmotion.com/>

COPYRIGHT NOTICE

© 2011 OMS Motion, Inc. ALL RIGHTS RESERVED

This document is copyrighted by OMS Motion, Inc.. You may not reproduce, transmit, transcribe, store in a retrieval system, or translate into any language in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, any part of this publication without the express written permission of OMS Motion, Inc.

TRADEMARKS

IBM, IBM PC, IBM PC/XT, IBM PC/AT, IBM PS/2, and IBM PC DOS are registered trademarks of International Business Machines Corporation. Windows 98, 2000, XP, NT and Windows 7 are registered trademarks of Microsoft Corporation.

DISCLAIMER

OMS Motion, Inc. makes no representations or warranties regarding the contents of this document. We reserve the right to revise this document, or make changes to the specifications of the product described within it at any time without notice and without obligation to notify any person of such revision or change.

3301-3200000
Revision B

| UMX Intelligent Motion Controls | | | | | | | |
|---------------------------------|----------------------|------------|---|------------------|-----|------|----------|
| Model | Interface | Servo Axes | Stepper Axes (Open or Closed – loop) | Axes control I/O | | | User I/O |
| | Virtual COM over USB | | | Limits | Aux | Home | GPIO |
| UMX-25 | Yes | 2 | | 4 | 2 | 2 | 8 |
| UMX-45 | Yes | 4 | | 8 | 4 | 4 | 8 |
| UMX-26 | Yes | | 2 | 4 | 2 | 2 | 8 |
| UMX-46 | Yes | | 4 | 8 | 4 | 4 | 8 |

TABLE OF CONTENTS

| | | |
|------|---|------|
| 1. | GENERAL DESCRIPTION | 1-1 |
| 1.1. | INTRODUCTION | 1-1 |
| 1.2. | SYSTEM OVERVIEW..... | 1-2 |
| 2. | GETTING STARTED | 2-1 |
| 2.1. | PREPARE FOR INSTALLATION | 2-1 |
| 2.2. | CONNECTING THE UMX FOR COMMUNICATION | 2-1 |
| 2.3. | CONNECT AND CHECKOUT THE SERVO SYSTEM | 2-2 |
| 2.4. | TUNE THE SYSTEM | 2-4 |
| 2.5. | SETTING THE USER DEFAULT CONFIGURATION | 2-10 |
| 2.6. | POWER SUPPLY REQUIREMENTS | 2-12 |
| 3. | COMMUNICATION INTERFACE..... | 3-1 |
| 3.1. | INTRODUCTION | 3-1 |
| 4. | CONTROL SIGNAL INTERFACE..... | 4-1 |
| 4.1. | INTRODUCTION | 4-1 |
| 4.2. | LIMIT AND HOME INPUTS | 4-1 |
| 4.3. | CONTROL OUTPUT..... | 4-1 |
| 4.4. | IO68 ADAPTER MODULE..... | 4-4 |
| 4.5. | UIO ADAPTER MODULE | 4-6 |
| 4.6. | ENCODER FEEDBACK | 4-10 |
| 4.7. | ENCODER SELECTION AND COMPATIBILITY | 4-10 |
| 4.8. | HOME PROCEDURES..... | 4-10 |
| 5. | COMMAND STRUCTURE | 5-1 |
| 5.1. | INTRODUCTION | 5-1 |
| 5.2. | COMMAND QUEUES..... | 5-2 |
| 5.3. | COMMAND SUMMARY | 5-4 |
| 6. | HOST SOFTWARE..... | 6-1 |
| 6.1. | INTRODUCTION TO UMX SOFTWARE SUPPORT | 6-1 |
| 6.2. | COMMUNICATION METHODS..... | 6-1 |
| 7. | SERVICE | 7-1 |
| 7.1. | USER SERVICE | 7-1 |
| | A. LIMITED WARRANTY | |
| | B. TECHNICAL INFORMATION / RETURN FOR REPAIR PROCEDURES | |
| | C. SPECIFICATIONS | |
| | D. INDEX | |

This page is intentionally left blank.

1. GENERAL DESCRIPTION

1.1. INTRODUCTION

The UMX family of motion controllers is a four axes motion controller that communicates through a USB 2.0 interface. The UMX is built on the successful PC78 architecture and shares the same command set and syntax and is compatible with other OMS motion controllers. (See list of [UMX models](#).)

The UMX family of controllers has models that can manage up to four (4) stepper or four (4) servo axes with encoder feedback. For open loop stepper operation, the encoders are not connected. It can manage coordinated or independent motion of each or all of the axes simultaneously. With high level functionality, such as circular and linear interpolation, multi-tasking, custom profiling, etc., the UMX can satisfy most any motion control application. The UMX utilizes a 32-bit microprocessor and patented, proprietary technology to control the trajectory profile, acceleration, velocity, deceleration, and direction of selected axes. In response to commands from the host computer, the UMX controller will calculate the optimum velocity profile to reach the desired destination in the minimum time while conforming to the programmed acceleration and velocity parameters.

The stepper control of the UMX produces a 50% duty cycle square wave step pulse at velocities of 0 to 1,044,000 pulses per second and an acceleration of 0 to 8,000,000 pulses per second, per second. The encoder feedback control can be used as feedback or position maintenance for the stepper axes or as strictly a position feedback of any axis. The encoder input supports either differential or single-ended quadrature TTL signals at a rate of up to 12MHz and counts at a 4 times resolution. This means a 1000 line encoder will produce 4000 counts per revolution in the UMX controller.

The UMX is commanded using virtually any programming language to the pass simple ASCII command strings to the UMX through the USB interface. For a typical motion requirement of 1,000,000 counts at 400,000 counts/second and an acceleration of 500,000 counts/second/second the following string would be sent from the host computer to the UMX:

[VL](#)400000;

[AC](#)500000;

[MR](#)1000000;

[GO](#)

For additional command programming examples see Chapter 5.

1.2. SYSTEM OVERVIEW

The UMX circuit board is a compact 3.536" x 3.278". As a stand-alone controller it communicates as a slave device through a USB interface. The UMX utilizes Flash Memory where programs (command sequences) can be stored permanently as macros.

The UMX utilizes the Motorola 68332 32-bit micro controller and FPGA technology for extensive logic integration and flexibility. The firmware, which resides in Flash Memory, can be upgraded through USB interface, without having to remove the controller from the system, if you have a 32-bit operating system such as Windows 2000 or XP. All signals to and from the UMX are buffered through TTL devices and are found on the shielded SCSI-3 type connector (J5). The IO68-M breakout can be used to route the signals of the UMX to individual screw terminal blocks, the UIO breakout can be used to route signals of the UMX to it's 15-pin connectors. Both the UMX and IO68-M utilize a resettable fuse on the +5V on the SCSI-3 connectors for protection.

As a stand-alone controller the UMX obtains its power from connector J3 and the USB communication port connects to J4. At the computer operating system level the UMX looks like a serial port device though the device driver directs the communications to the USB port. The UMX supports baud rates of 300 to 38.4K baud. With the ability to store macros into Flash Memory, the UMX could essentially be programmed once and then be embedded into a machine where it could run independently.

2. GETTING STARTED

2.1. PREPARE FOR INSTALLATION

There are two driver installation modes: (Decide before connecting the UMX to the host system!)

Option 1: UMX controller mapped to COM port (default USB behavior)

A UMX controller appears at the same COM port no matter which USB port the controller is connected. A UMX with a different serial number will require another driver installation and mapping to a different COM port.

On a virgin system no special preparation is needed, otherwise run AttachUsbToComPort.exe before connecting the UMX controller to the System.

Option 2: USB port mapped to COM port (recommended)

Any UMX controller plugged into a specific USB port is mapped to the same COM port each time. A UMX controller connected to a different USB port requires a reinstallation of the driver and mapping to a different COM port.

You need to run AttachUsbToComPort.exe before connecting a UMX controller to the System.

2.2. CONNECTING THE UMX FOR COMMUNICATION

The first requirement for communication through the USB interface is to ensure that the UMX is securely and safely mounted where damage is unlikely. This includes the exposure to possible static discharge, moisture, debris, etc. Special mounting efforts may be required to protect the extended pins on the bottom of the UMX.

CAUTION:

The UMX is a static sensitive device and standard ESD (Electro Static Discharge) techniques are required when handling and installing the UMX.

Connect a +5VDC, 1 Amp power source to the power connector at J3. (See [Table 2-1](#) for the connector pin-out) NOTE: +/- 12VDC is required only for servo operation.

Connect the USB cable to the Host Computer.

The Hardware setup wizard will appear. Choose the Advanced option and direct the wizard to the UMXDriver folder.

The wizard will display a warning that about an uncertified device description. It is safe to ignore it and continue.

Follow the same choices in the second wizard that maps the COM port.

For more detailed instruction read the UMXDriverInstallation.pdf in the Driver package.

TABLE 2-1 CONNECTOR J3

| | PIN | PIN | |
|---------|-----|-----|----------------|
| +5V DC | 5 | 6 | Digital Ground |
| -12V DC | 3 | 4 | No Connect |
| +12V DC | 1 | 2 | Analog Ground |

Reference the support software disk to find a serial communications utility that will work with the UMX.

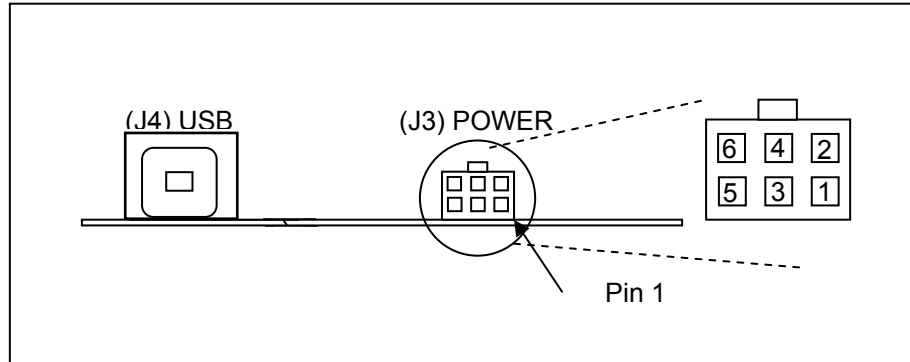


FIGURE 2-1 CONNECTOR CONFIGURATION

2.3. CONNECT AND CHECKOUT THE SERVO SYSTEM

Servo systems tend not to respond gracefully to connection errors. You can reduce the chance of making connection errors by following a step-by-step procedure:

Caution

The servo motor may jump or spin at a very high velocity during connection and configuration. The motor should be restrained via fastening to the physical system or by some other means before beginning this procedure. Keep hands and clothing clear of the motor and any mechanical assemblies while performing this procedure.

2.3.1.CONNECT AND CONFIGURE THE MOTOR\AMPLIFIER

1. Connect and configure your amplifier per the manufacturer's instructions (for "Torque" or "Open-Loop" mode).
2. With the motor and amplifier power turned off, connect the UMX to the amplifier. Do not connect the encoder yet.
3. Balance your motor:
 - a. Using a voltage meter, verify that the command signal from the UMX is within +/-500mV. If it is not, send the command "[DZ0](#)"; to the UMX and recheck the voltage. If the voltage is still too high, contact OMS Motion, Inc. Technical Support department for guidance.
 - b. Turn on power to the amplifier and then to the motor.
 - c. Adjust the balance setting of your amplifier (if equipped) until the motor stops moving.
 - d. If the motor continues to revolve or your amplifier has no balance adjustment:
 - i. Send the command "[DZ100](#);" to the UMX.
 - ii. If the motor spins faster, reduce the command parameter and resend the command, e.g. "[DZ50](#);"
 - iii. If the motor spins slower but does not stop, increase the command parameter and resend the command, e.g. "[DZ150](#);"
 - iv. Continue adjusting and resending the [DZ](#) command until the motor comes to rest. Write down the final [DZ](#) value for later reference as your "zero" setting.
4. Maximize your system's usage of the UMX's DAC:
 - a. Connect the servo encoder to the UMX.
 - b. Set the signal/command gain of your amplifier to its minimum setting.
 - c. Send the "[DZ3277](#)"; command to the UMX and observe the velocity of the motor. The output of UMX will be near 1VDC.
 - d. If the motor does not move at all, your amplifier does not work well at a low velocity. In this case, adjust the signal/command gain of the amplifier to approximately 20% of maximum or until the motor begins to move.
 - e. Using a frequency meter, measure the pulse rate of Phase A of the encoder. The frequency measured is ¼ of the actual pulse rate.
 - f. Adjust the signal/command gain of the amplifier until the pulse rate of Phase A (4 is approximately 10% of your desired peak operational velocity. If the pulse rate is already greater than 10% of peak, your amplifier is not designed for low velocity motion and you will likely have some difficulty tuning your motors.
 - g. Send the "[DZ-3277](#);" command to the UMX and recheck the velocity. You may need to readjust your amplifier. If so, do not reduce the signal/command gain – only increase the setting as needed. Increasing the gain will not impair the forward peak velocity but reduction will.
 - h. Send the [DZ](#) command with the "zero" value noted at the end of step 3d (iv) to the UMX. Send the same value using the [KO](#) command, e.g. "[KO-175](#);"
5. Verify the direction of your servo encoder:
 - a. Send the "[LP0:DZ2000](#);" command to the UMX.

- b. Send the [“RE”](#) command to the UMX and observe the response.
 - c. If the response is positive, no further action need be taken; go to step 6.
 - d. If the response is negative, your encoder must be reversed.
 - i. If your encoder produces a differential signal, swap Phase B with Phase B-not and repeat from step (a.) above.
 - ii. If your encoder produces a single-ended (or TTL) signal, swap Phase A with Phase B and repeat from step (a.) above.
 - e. If the [RE](#) response is still negative, contact OMS Technical Support for assistance.
6. Repeat from step 1 for the other servo axes.
 7. Remember to set [DZ](#) and [KO](#) for each axis at every power-up unless you store the values in flash (see Section 5 COMMAND STRUCTURE)

NOTE: Most encoder problems are caused by lack of power or incorrect connections. If the encoder position changes by only 1 count, this is an indication that one of the phases is not connected.

Caution

Do not proceed until you perform all the steps in this procedure, ensure that the outputs of the UMX are as described, and ensure that the encoder is operating correctly

2.4. TUNE THE SYSTEM

2.4.1.INTRODUCTION

Tuning a servo system is the process of balancing the conflicting requirements to achieve optimum performance of a real world system.

The first of these requirements is that of accuracy. In a closed loop system, an error signal is derived, then amplified, then supplied to the motor to correct any error. Clearly, if a system is to compensate for infinitely small errors, the gain of the amplifier needs to be infinite. Real world amplifiers do not possess infinite gain; therefore, there is some minimal error which cannot be corrected. In order to have the greatest possible accuracy, the gain needs to be as high as possible. Unfortunately, other real world considerations limit the maximum gain of the system.

The second of the requirements is that of stability. The system must not be unstable, e.g. oscillate. The degree to which a system is stable affects its performance. The effects can be seen when looking at the system's response to a step change at the input. The step response falls into one of three categories: under damped, critically damped, over damped. Over damped systems are slow to reach their final value. Critically damped systems reach final value quickly, without overshoot. Under damped systems reach final value quickly, but have various degrees of "ringing" that decay to zero.

The third requirement is that of bandwidth. The system should respond to the highest input frequency possible. The motor/load combination is the predominant feature of the open loop bandwidth. In the closed loop situation, the amplifier attempts to compensate for the limited

response characteristics of the motor load. Increasing gain extends the closed loop bandwidth at the expense of stability.

2.4.2.MANUAL TUNING

In most all motion control applications the optimum tuning of the servo system is achieved through a manual tuning process. Auto-tuning algorithms typically can only get the system parameters close and require manual steps to fine tune the parameters. An empirical trial and error approach will be discussed first.

NOTE: You may want to use the OMS software to help during this process. It is capable of capturing the actual data and plotting it in reference to an ideal motion profile.

There are some system parameters that need to be defined to before attempting to tune a motor. The encoder resolution, counts per revolution, is one element to be determined. Another is the systems maximum velocity. Note that a motor should never exceed 90% of the motor's top rpm. If the system requirement is for a velocity higher than 90% of the motors top rpm, then another motor with higher rpm capability is to be used.

The system's maximum acceleration is determined a couple of different ways. The best method is to determine the system time constant, which includes "hitting" or "bumping" the motor under system load and measure the time from 0 rpm to maximum rpm and divide this value by 5. The maximum acceleration is either 2.5 times this value, or is based on the system requirements for handling the load as defined in the operating specifications of the system. This value is always lower than the calculated value and if this acceleration value is not high enough then a different motor/amplifier with more power or band-width should be utilized.

The UMX can control either current mode or voltage mode amplifiers. The servo update rate of the UMX is 488μs, for four axes. High following errors can be compensated for using the feedforward coefficients explained later in this section. There are some general formulas that have been developed to determine acceptable following error for both current and velocity mode systems:

Current mode following error for $K_P = (3^\circ/360^\circ) \times (\text{counts per revolution})$

Voltage mode following error for $K_P = (90^\circ/360^\circ) \times (\text{counts per revolution})$

It is obvious that the voltage mode allows for much greater following errors than the current mode. This value is the following error when the motor is at peak velocity and will be used when determining the proportional gain (K_P).

The following error for the integral term (K_I) or long-term gain value will follow the following guidelines:

Current Mode following error for $K_I = 0$ counts

Voltage Mode following error for $K_I = 80^\circ$ of 360° (expressed in motor counts)

While still in open-loop mode, hold off (H_F use the D_Z command to zero the motor. This variable is used to provide a constant output that will compensate for any torque offset from the load. So, when the system should be stationary, the necessary voltage will be sent to the amplifier to cause the motor to maintain position. With the correct D_Z value, the motor should successfully maintain a zero position.

[KO](#) is the offset coefficient used while in closed-loop mode, hold on ([HN](#)). [KO](#) is essentially the same as [DZ](#), but used for closed-loop operation. Once you have determined the correct value for [DZ](#), this same value should be used for the [KO](#) variable before beginning to tune the PID filter

The values for [DZ](#) and [KO](#) range from -32767 to 32767.

Set the known values for velocity, acceleration and the move distance for a trapezoidal profile with at least a 20% flat spot at peak velocity. Formula:

Profile distance = $((\text{peak velocity})^2 / (2 \times \text{acceleration})) \times 2.4$

Example: $((50,000)^2 / (2 \times 500,000)) \times 2.4 = 6,000$

Execute the move by sending the move commands to the UMX.

Example: [MR](#)6000;
 [GO](#);

Adjust the [KP](#) term while repeating step 2 until the following error at the flat spot of the profile is acceptable. If the motor becomes unstable prior to obtaining the optimum [KP](#) term than increase the [KD](#) term until the motor stabilizes.

Example: [LP](#)0;
 [KP](#)3;
 [HN](#);
 [MR](#)6000;
 [GO](#);
 [LP](#)0;
 [KP](#)10;
 [HN](#);
 [MR](#)6000;
 [GO](#);
 [LP](#)0;
 [KP](#)25;
 [HN](#);
 [MR](#)6000;
 [GO](#);
 [LP](#)0;
 [KD](#)100;
 [HN](#);
 [LP](#)0;
 [KP](#)35;
 [HN](#);
 [MR](#)6000;
 [GO](#);
 [LP](#)0;
 [KD](#)125;
 [HN](#);

The values in the above example are totally arbitrary and may vary drastically with different systems. The [LP](#)0; command is used to set the position error to 0.

The values for [KP](#) range from 0 to 32767.

Once the [KP](#) term has been obtained, and then continues executing the motion while rising the [KI](#) term until the long-term following error is acceptable. This error can be measured at the two

knees of the motion profile. By increasing the [KI](#) term, the response time of your system will increase. The motion profile should have a steeper slope as [KI](#) increases.

However, as [KI](#) increases the system can also become unstable. If the instability becomes unacceptable increase the [KD](#) parameter. This will increase the dampening on the system's motion profile (therefore reducing oscillation, or "ringing"). Continue adjusting the [KI](#) and [KD](#) terms until the proper response time is obtained.

The values for [KI](#) range from 0 to 32767.

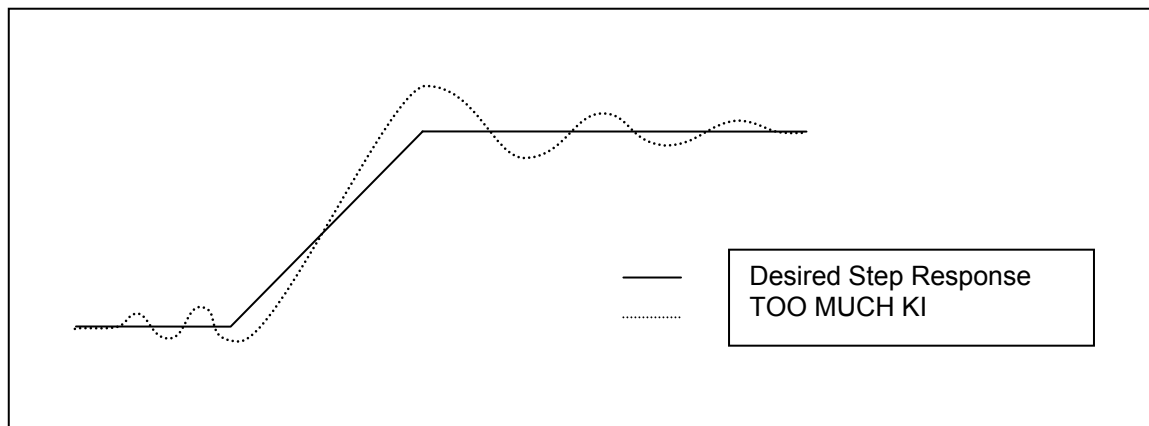


FIGURE 2-2

If you are getting too much "ringing" in the motion profile, then increase [KD](#) to help dampen the system's response. If, instead, the system is over-damped and is reaching the final velocity too slowly, then reduce the [KD](#) parameter. Optimally, the system's motion profile should show the motor reaching the desired velocity as quickly as possible without overshoot and oscillation ("ringing").

The values for [KD](#) range from 0 to 32767.

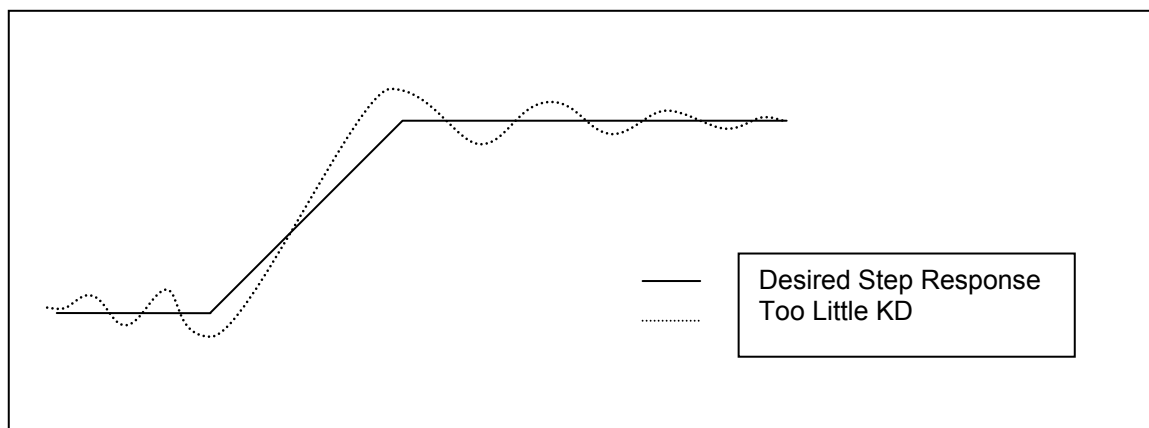


FIGURE 2-3

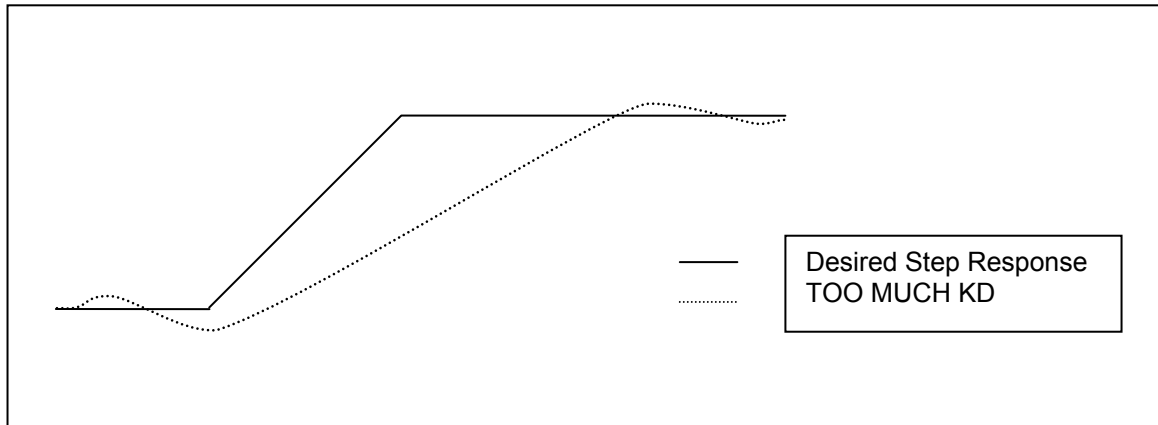


FIGURE 2-4

[KP](#), [KI](#) and [KD](#) are the primary parameters of concern when tuning a servo system. Once the optimum values for these variables have been determined, then you can adjust some of the secondary parameters that will help fine tune your system's performance. These other variables are described in the subsequent steps.

The [KV](#) variable is used when tuning velocity controlled servos (voltage mode servo amplifiers). This is the velocity feedforward coefficient. [KV](#) determines how closely the system follows the desired constant velocity portion of the motion profile. By increasing this term, the following error of the system's response can be minimized. However, too large of a value may result in unstable behavior after command velocity changes.

The values for [KV](#) range from 0 to 32767.

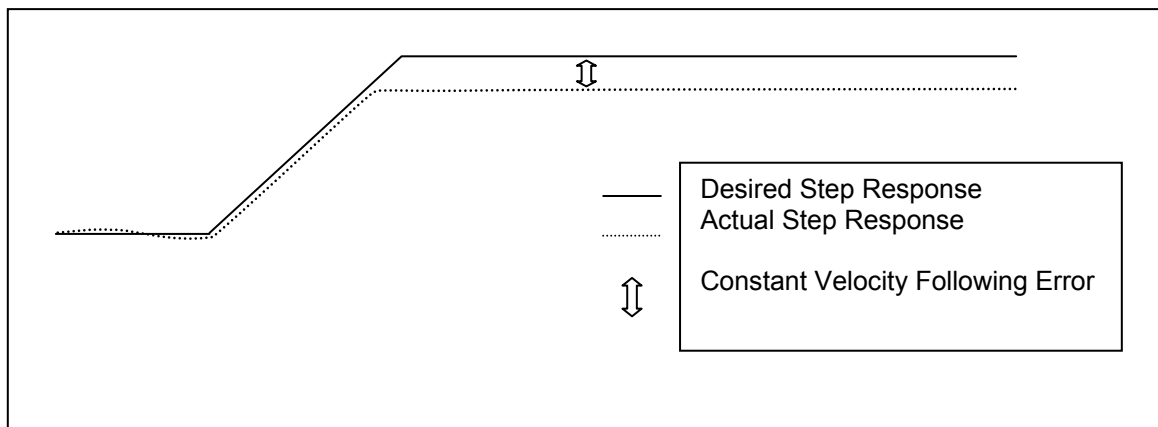


FIGURE 2-5

The [KA](#) variable is used when tuning torque controlled servos (current mode servo amplifiers). This is the acceleration feedforward coefficient. Systems with high inertial loads may require additional torque during acceleration or deceleration to achieve optimum performance. [KA](#) determines how closely the system follows the desired acceleration and deceleration portions of the motion profile. Increasing this term reduces the following error occurring during acceleration and deceleration of the system. Although, if [KA](#) is too large, instability may occur.

The values for [KA](#) range from 0 to 32767.

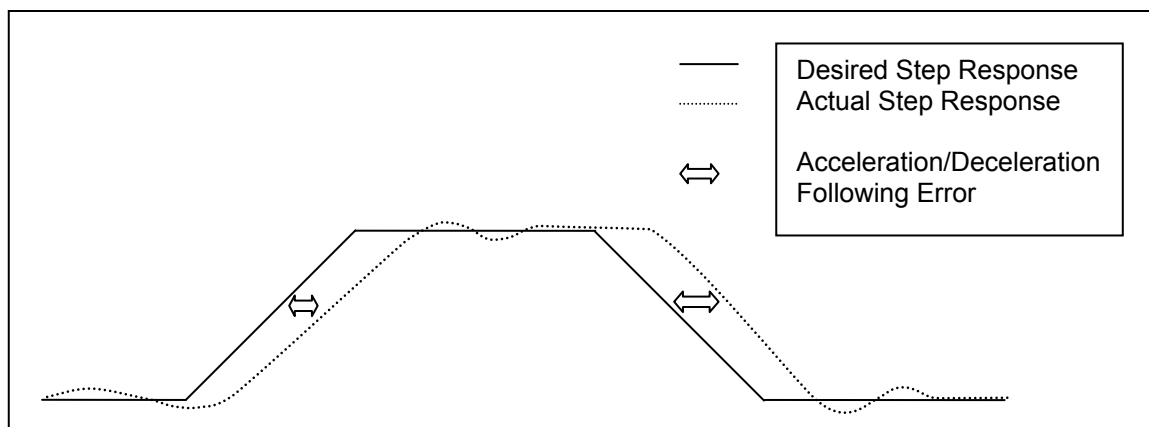


FIGURE 2-6

The block diagram below describes the feedback loop that is taking place in the servo system:

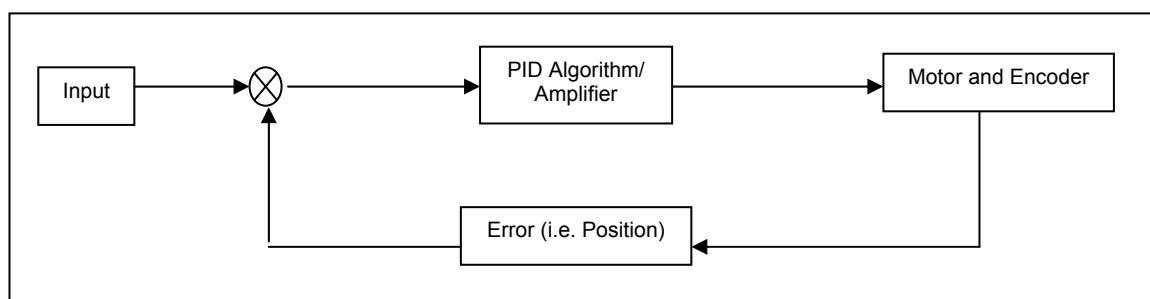


FIGURE 2-7 FEEDBACK LOOP

You may want to save the values for [KP](#), [KI](#), [KD](#), etc., for future reference. These values can be saved in the board's Flash Memory, so they can be accessed easily on reset or power-up. The command [AP](#) will store your current parameter assignments, such as [KP](#), [KI](#), [KD](#), etc., into flash memory. These saved parameters will then be used as the power up default set of values. Refer to page 5-33 for more detailed information regarding how to use the commands to save and load parameter sets from flash memory.

Once the [KP](#) term has been obtained, continue executing the motion while raising the [KI](#) term until the long-term following error is acceptable. This error is to be measured at the two knees of the profile. If the motor becomes unstable before obtaining the optimum following error then increase the [KD](#) term until the motor becomes stable.

To verify that your motor is tuned properly after you have completed the first 11 steps perform the following test to test the holding torque: Send [LP0;HN](#) commands and check the shaft of the motor to make sure it is stiff. If there is play in the motor shaft when you turn it then you may have to re-adjust your PID filter.

Once you are satisfied with the static holding torque you could check for position error. Send the command "[AC100000;VL5000;MR64000;GO](#)";. With an 8000 line encoder this move would be equivalent to 8 revolutions of the motor. After the move is complete check the position error by sending the [RE](#) and [RP](#) commands for the specific axis you are moving. Compare the difference in the two responses. If they are the same then you are on the right track, if the error was greater than 32768 then the controller will disable the PID so that you don't have a runaway motor and major changes to the PID parameters may be required. For minor differences in the encoder and the position reading you can fine-tune your PID filter according from the earlier steps.

2.5. SETTING THE USER DEFAULT CONFIGURATION

There are several parameters that can be defined by the user as default. These parameter values supersede the factory default values and are stored in Flash Memory for power-up configuration. Most of these parameters consist of axis specific values, i.e. velocity, acceleration, limit switch logic sense, etc. The configuration of the User I/O must be configured once the communication interface has been established.

2.5.1.SETTING THE USER I/O DEFAULTS

The factory default of these signals sets I/O bits 0 through 3 as inputs and 4 through 7 as outputs. If these signals are to be used in a configuration other than the factory defaults then they must be reconfigured before any hardware is connected. If an input device, such as a position sensor or limit switch, is attached to a User I/O signal configured as an output then the logic gate on the UMX may be destroyed and additional damage to the UMX could occur. The “[IO](#)” command is used to change the User I/O from input to output and visa-versa. User I/O bits are configured in blocks of four. The syntax for the [IO](#) command uses a 0 to set the corresponding four I/O bits to inputs and a 1 to set them to outputs.

Example: [IO](#)0,1;

This will set I/O bits 0 – 3 as inputs and 4 – 7 as outputs (factory default).

Example: [IO](#)1,1;

This will set all bits (0 – 7) to outputs.

Refer to page 5-102 for more on the [IO](#) command.

The “[RB](#)” Report Bits command is used to check the configuration of the I/O bits. This command should be executed to ensure that the configuration of the I/O bits is as required by the system. This should be performed PRIOR to connecting hardware to the User I/O signals of the UMX.

The eight User I/O bits all have a 2.2K Ohm pull-up resistor connected to them. By default, the active state of the inputs is a low true. When the I/O bits are configured as inputs it is only necessary to have a switch closure to ground to activate the input. The [BX](#) command is used to check the status of the input bits. The use of a 100-Ohm pull-down resistor can be used to change the state of the inputs for testing. Refer to page 5-42 for more on the [BX](#) command.

The “[BH](#)” Bit High and “[BL](#)” Bit Low commands are used to toggle the logic state of the User I/O bits. Before connecting the User I/O signals to exterior hardware, you should be certain that the bits are configured correctly. An LED connected through a 100 Ohm resistor to +5V can be used to test the I/O bits when toggling them. Refer to page 5-37 for more on the [BH](#) and page 5-39 for more on the [BL](#) commands.

2.5.2.OTHER USER DEFINABLE DEFAULT PARAMETERS

The UMX comes from the factory with default values for all parameters. For instance, the default value for the velocity of all axes is 100,000 counts per second. (A count is equivalent to a step pulse or one count of an encoder.) In a typical application, when the system is powered up, the main host computer would initialize all of the peripherals, such as the UMX, sending to each of the axes the peak velocity. When the User Definable Default Parameter value is defined for the velocity then the initialization of the system can skip initializing the velocities of the defined axes. This feature can greatly simplify the software and initialization process.

Once the values for all of the associated parameters are defined, i.e. velocity, acceleration, PID values, etc. then the [“AP”](#) Archive Parameters command is executed to place the values into Flash Memory. From this point forward these defined values will be used after reset or power-up. The individual parameters can be over-written at anytime by using the associated command, i.e. [VL](#), [AC](#), etc. To restore the factory defaults the command [“RF”](#) Restore Factory Defaults is executed. To restore the User Defined Default Parameters the command [“RD”](#) Restore Defaults is executed. Refer to Section 5 COMMAND STRUCTURE for more information on these commands.

The following is a list of parameters that can be defined as part of the User Definable Power-Up Default Parameters.

| | Factory Default Settings |
|--|----------------------------------|
| Baud Rate for serial communication (?SB) | 9600 |
| I/O bit configuration (BX) | 00 |
| Overtravel limit (soft limit or hard limit) (?SL) | sf |
| Overtravel limit (enabled or disabled) | enabled (0) |
| Overtravel limit polarity (active high or active low) (?LS) | ll |
| Software based overtravel for each axis (?TF) | tf |
| Direction bit polarity (?DB) | dbn |
| Acceleration value for each axis (?AC) | ac2000000 |
| Trajectory profile for each axis (linear, parabolic, custom) (?RT) | la |
| Velocity Peak (?VL) | vl2000000 |
| Velocity Base(?VB) | vb0 |
| User Unit values for each axis (?UU) | uf |
| Auxiliary output settle time for each axis | 0 |
| Automatic auxiliary control axis by axis | disabled |
| Encoder Ratio for each axis (?ER) | 1:0 |
| Encoder Slip/stall tolerance for each axis (?ES) | es1 |
| Position Maintenance Dead-Band, Hold Gain and Hold Velocity. (?HD) | hd0 |
| Servo axis unipolar/bipolar output (?SO) | bi |
| Servo PID values: KP , KD , KI , KO , KV , KA , KF , KB , KU | 10, 20, .04, 0, 0, 0, 0, 10, 200 |
| Servo zero value: (?DZ) | dz0 |

NOTE: Use the [AP](#) command sparingly since it writes to Flash Memory. There is a finite number of times the Flash can be re-written (i.e. less than 10,000 times, typical.)

2.6. POWER SUPPLY REQUIREMENTS

The UMX is designed to operate from the power supplied by an external power supply which must be capable of supplying +5V at 1 amp (typical) to the UMX, J3. Servo models of the UMX require +/-12V at 0.1 amps (typical) and these voltages must be present for proper servo operation.

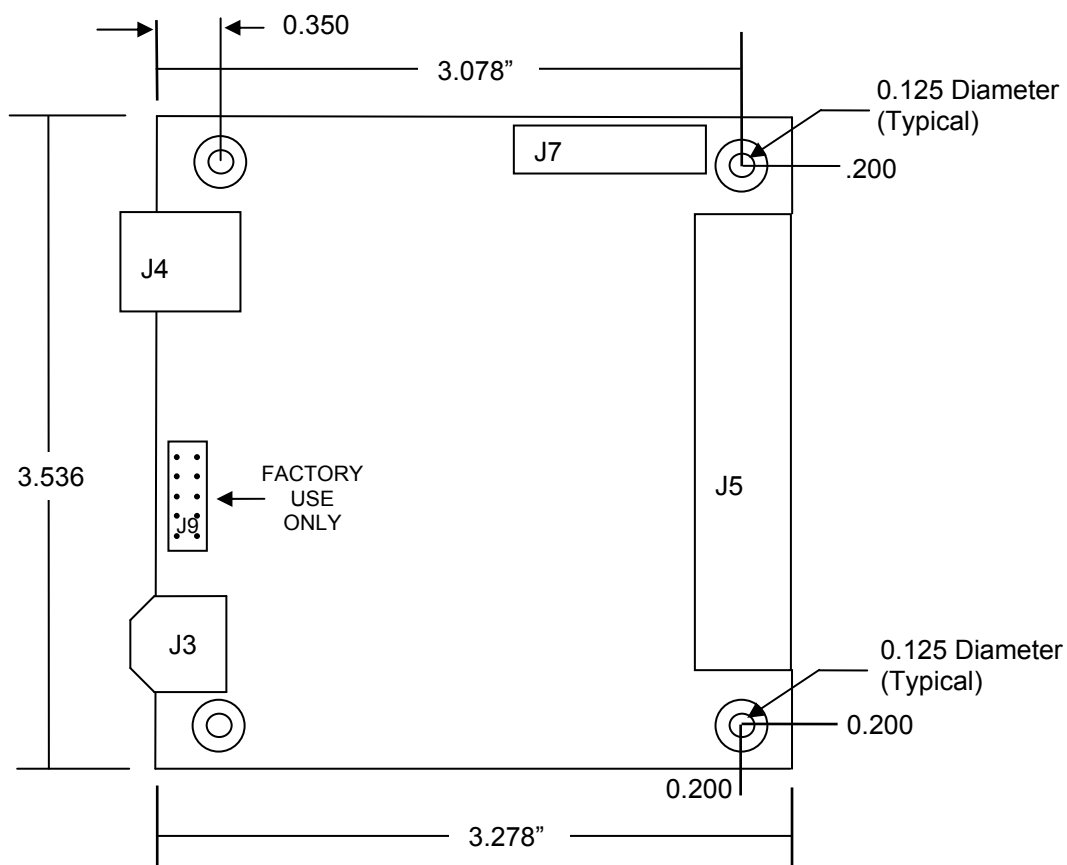


FIGURE 2-8 UMX DIMENSIONAL LAYOUT

* = NOTE: The UMX has a notched connector for J5 to better accommodate the SCSI3 connector.

3. COMMUNICATION INTERFACE

3.1. INTRODUCTION

The UMX is a compact multi-axis motion controller for Servo or Stepping motors or Stepping motors with encoders. The UMX communicates across a USB 2.0 interface. The software drivers are supplied with the controller and support most Windows operating systems. The UMX will appear as a serial device on a COM port of the computer for a simple communications interface though the hardware is a USB port.

The UMX can control up to four (4) axis of motion that can be coordinated, independent or any combination of coordinated and independent axes. The UMX utilizes Flash memory technology that provides for the ability to store parameters and command macros for execution. The power-up default settings can even be customized and stored and the UMX can be programmed to execute the macros based on a general purpose input to provide for a completely stand-alone system. The stepper motor version requires a single +5VDC supply while the servo motor version requires an additional low power +/-12VDC supply for the control of the analog signals.

Along with a 32-bit microprocessor and patented, proprietary technology the UMX controls the trajectory profile, acceleration, velocity, deceleration, and direction of all selected axes. In response to commands from the host computer, the UMX controller will calculate the optimum velocity profile to reach the desired destination in the minimum time while conforming to the programmed acceleration and velocity parameters.

The stepper control of the UMX produces a 50% duty cycle square wave step pulse at velocities of 0 to 1,044,000 pulses per second and an acceleration of 0 to 8,000,000 pulses per second, per second. The encoder feedback control can be used as feedback or position maintenance for the stepper axes or as strictly a position feedback of any axis. The encoder input supports either differential or single-ended quadrature TTL signals at a rate of up to 12MHz and counts at a 4 times resolution. This means a 1000 line encoder will produce 4000 counts per revolution in the UMX controller.

The UMX is commanded using virtually any programming language to the pass simple ASCII command strings to the UMX through USB port. For a typical motion requirement of 1,000,000 counts at 400,000 counts/second and an acceleration of 500,000 counts/second/second the following string would be sent from the host computer to the UMX:

VL400000;

AC500000;

MR1000000;

GO

For additional command programming examples see Chapter 5.

The UMX returns status information to the host via special characters inserted into the response stream. If a “Done” or “Command Error” condition occurs, the UMX will send one or more of these characters to the host. These characters are:

- # Command Error
- \$ Motor Slip/stall (with encoders only)
- @ Overtravel Limit
- ! Done

These characters are not axis-specific. It is necessary for the host to query the UMX for that information if it is required.

4. CONTROL SIGNAL INTERFACE

4.1. INTRODUCTION

The UMX family contains a two or four axes stepper with encoder feedback motion controller or a two or four axis servo motion controller. A single 68-pin shielded SCSI-3 connector incorporates all the control signals of the UMX. The mating connector is an AMP, Inc. part number 749621-7 with a 749195-2 hood and strain relief. As a convenience in system integration, connections to the UMX for +5VDC power, +12 VDC, -12 VDC, digital ground and analog ground are provided at the output connector (J5).

The red LED is used to assist in fault diagnosis, and if it is “on” steady, a fault exists in the UMX.

4.2. LIMIT AND HOME INPUTS

To facilitate system implementation, limit and home inputs are provided for each axis. Limits may be activated by mechanical switches using contact closures or other suitable active switches, such as a Hall Effect switch or opto-isolator that connects to ground.

If the motor travels beyond its allowable limits and trips the switch, the limit switch closure removes the excitation from the affected axis. You can select the limit switch active signal state with the [LH](#) and [LL](#) command on an axis by axis basis. The behavior of the limit functionality can be controlled with the System Control Commands.

The home switch provides a means to synchronize the motor controller with the load at some home, or reference position. The home switch, when used with the software commands [HM](#) or [HR](#), causes the motor to decelerate to a stop when the switch closes. On finding the home position, the position counters are initialized to the parameter supplied with the command. You can change the sense of the home switches to TRUE when open by use of the [HH](#) command (described on page 5-88).

4.3. CONTROL OUTPUT

The stepper version of the UMX is configured at the factory for control of stepper axes with encoder motors. The servo output may be either unipolar analog (0 to +10V) or bipolar analog ($\pm 10V$). (See the [UN](#) and [BI](#) commands). Step pulse output is TTL open-collector which will wire directly into most driver inputs but may require a 2K Ohm pull-up resistor to +5 VDC to operate some other drives. See wiring diagrams that follow.

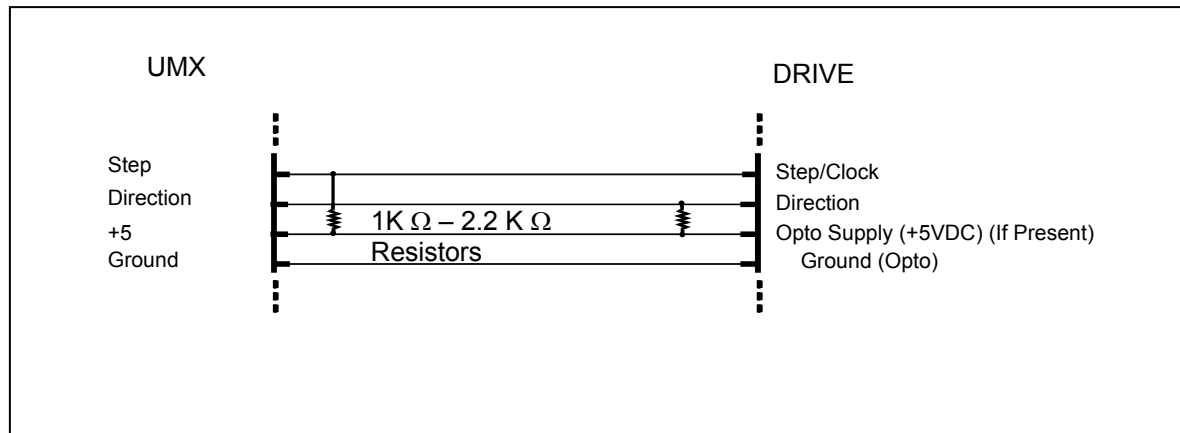


FIGURE 4-1 CONNECTION TO STEP DRIVES WITH INTERNAL PULL-UP RESISTORS AND OPTO-ISOLATION

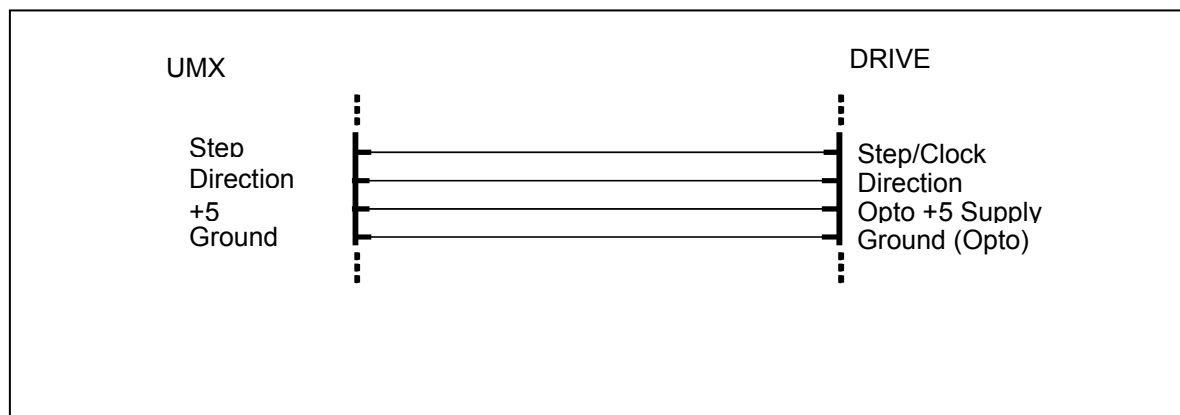


FIGURE 4-2 CONNECTION TO STEP DRIVES WITHOUT PULL-UP RESISTORS

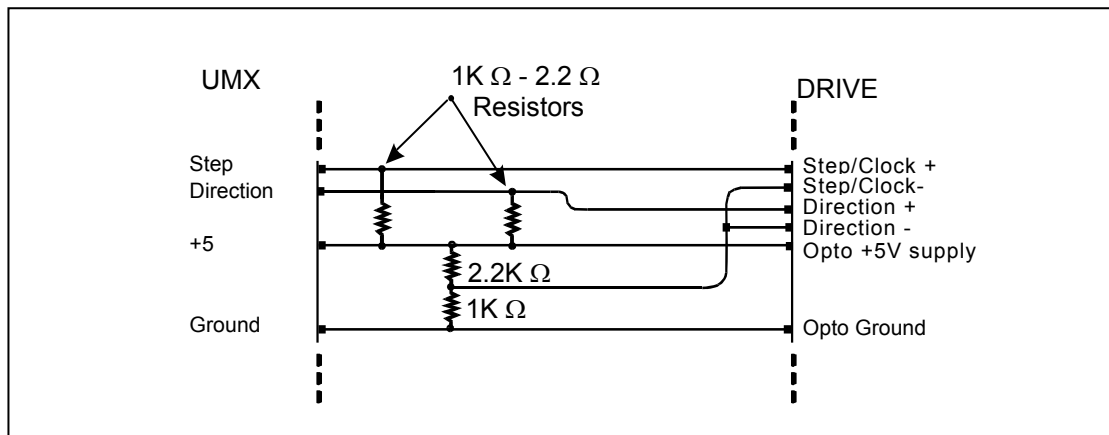


FIGURE 4-3 CONNECTION TO STEP DRIVES WITH DIFFERENTIAL INPUTS

TABLE 4-1 OUTPUT CONNECTOR PIN LIST (J5) (AT CIRCUIT BOARD)

| SIGNAL CONNECTOR (J5) | | | |
|------------------------------|--------------------|-------------|--------------------|
| Pin# | Description | Pin# | Description |
| 1 | Digital Ground | 35 | +5VDC |
| 2 | I/O-1 | 36 | I/O-0 |
| 3 | I/O-3 | 37 | I/O-2 |
| 4 | I/O-5 | 38 | I/O-4 |
| 5 | I/O-7 | 39 | I/O-6 |
| 6 | Digital Ground | 40 | +5VDC |
| 7 | X Index + | 41 | X Servo |
| 8 | X Index - | 42 | X Step |
| 9 | X Phase A + | 43 | X Auxiliary |
| 10 | X Phase A - | 44 | X Direction |
| 11 | X Phase B + | 45 | X Limit + |
| 12 | X Phase B - | 46 | X Limit - |
| 13 | Y Servo | 47 | X Home |
| 14 | Y Index + | 48 | Y Step |
| 15 | Y Index - | 49 | Y Auxiliary |
| 16 | Y Phase A + | 50 | Y Direction |
| 17 | Y Phase A - | 51 | Y Limit + |
| 18 | Y Phase B + | 52 | Y Limit - |
| 19 | Y Phase B - | 53 | Y Home |
| 20 | Analog Ground | 54 | +5VDC |
| 21 | Z Index + | 55 | Z Servo |
| 22 | Z Index - | 56 | Z Step |
| 23 | Z Phase A + | 57 | Z Auxiliary |
| 24 | Z Phase A - | 58 | Z Direction |
| 25 | Z Phase B + | 59 | Z Limit + |
| 26 | Z Phase B - | 60 | Z Limit - |
| 27 | T Servo | 61 | Z Home |
| 28 | T Index + | 62 | T Step |
| 29 | T Index - | 63 | T Auxiliary |
| 30 | T Phase A + | 64 | T Direction |
| 31 | T Phase A - | 65 | T Limit + |
| 32 | T Phase B + | 66 | T Limit - |
| 33 | T Phase B - | 67 | T Home |
| 34 | Digital Ground | 68 | +5VDC |

4.4. IO68 ADAPTER MODULE

The optional IO68-M is an adapter module designed to provide easy connection for each signal of the UMX. It incorporates a three row terminal block and some on-board filters for the limit inputs. A cable (CBL68-10) is available with the mating connector on each end to fit the UMX connector (J5). The +5VDC on the IO68-M is supplied by the UMX and is protected by a resettable fuse on the UMX. A green LED lights up when 5VDC is present.

This supply voltage is intended to be utilized with accessories used in conjunction with the UMX such as sensors, motor driver modules, etc., and is specified to supply a maximum current of 0.5 amps for these purposes.

If the fuse detects an over current situation (such as an external short circuit), the supply will shut down. It can be re-activated by powering down the UMX, ensuring the over current situation has been removed, and by powering the UMX up again after 3 seconds.

As the fuse is a semiconductor device, it never has to be replaced and requires no maintenance.

4.4.1. EXPLANATION OF ADDITIONAL CIRCUITRY ON THE IO68-M

The IO68-M board is to be used in conjunction with a UMX. The IO68-M board contains low-pass filtering circuitry for the positive and negative limit signals. The default values for the RC constants are 100Ω and $1.0\mu\text{F}$. With these RC constants used the average cutoff frequency is 21.5 kHz, in a worse case scenario, where all other signals are left unterminated.

In addition to the above mentioned circuitry, there are also 2.2K pull-up resistors that have been added to the step output signals for each axis. These resistors have been added to the IO68-M to help reduce noise that may occur on the step output signals.

The resistors and capacitors are packaged as through hole devices. Should your particular application require different RC constants, these devices can be removed from the board and replaced with the appropriate components. When changing these components, use appropriate methods to desolder and solder the components to avoid causing damage to the board (i.e. lifting pads from the board).

There are two switches on the IO68-M (S43 and S45) that are used in regards to encoder signals. If your system uses differential encoders, then these switches should be in the OFF position (except where indicated). The switches are set to the OFF position as default from factory. If, however, your system employs single-ended encoders, then the negative encoder signals need to be fed a biasing voltage. Since a biasing voltage has been made available on the IO68-M, simply change the position of the switches to ON to tap into this voltage. Be sure to change the position of these switches only when power to the board is OFF (the green LED should be off).

TABLE 4-2

IO68-M ENCODER BIAS SWITCH (S43)

| Switch Number | Signal Description |
|---------------|--------------------|
| 1 | Y Phase B- |
| 2 | Y Phase A- |
| 3 | Y Index- |
| 4 | X Phase B- |
| 5 | X Phase A- |
| 6 | X Index- |
| 7 | (Leave ON) |
| 8 | (Leave ON) |

IO68-M ENCODER BIAS SWITCH (S45)

| Switch Number | Signal Description |
|---------------|--------------------|
| 1 | T Phase B- |
| 2 | T Phase A- |
| 3 | T Index- |
| 4 | Z Phase B- |
| 5 | Z Phase A- |
| 6 | Z Index- |
| 7 | (Leave OFF) |
| 8 | (Leave OFF) |

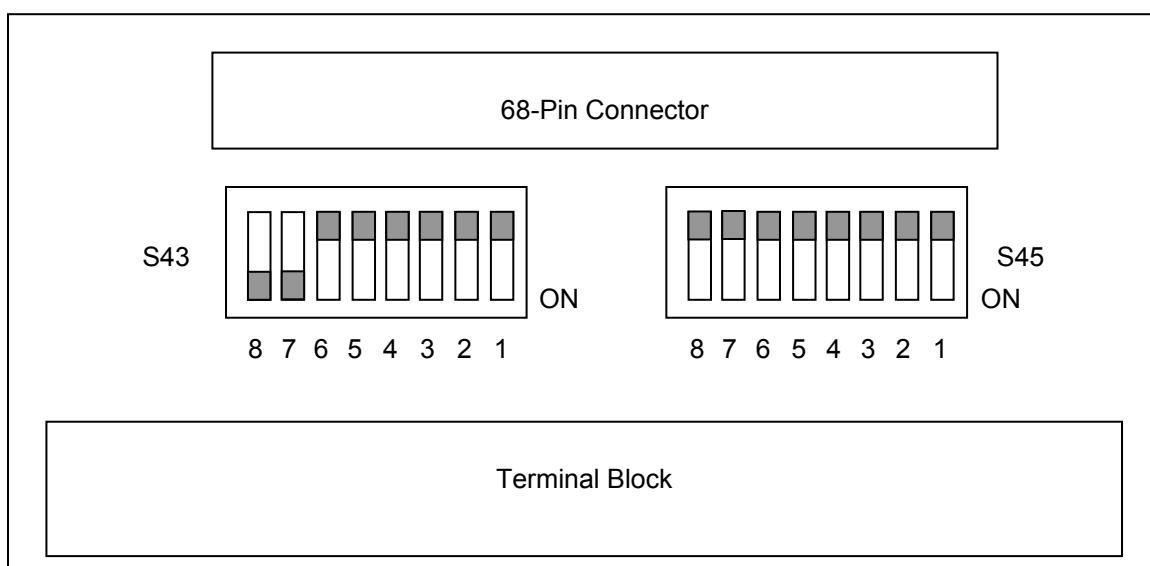


FIGURE 4-4 IO68-M DEFAULT SWITCH SETTING

If only one of the axes is using a single-ended encoder, while the others are using a differential encoder, you need only turn on the switch positions that relate to the axis with the single-ended encoder.

Besides biasing the negative encoder signals, these switches also configure the IO68-M. Placed in the position shown in Figure 4-4, the IO68-M is configured to work with the UMX. For the IO68-M, S43, switches 7 and 8 are to be ON, while S45, switches 7 and 8 are to be OFF.

All other signals on the IO68-M are straight through connections, with no additional circuitry added. Should you need filtering circuitry on any of these other signals, it would have to be added external to the IO68-M. See the OMS Motion, Inc. (<http://www.omsmotion.com/>) website if you need further instruction.

TABLE 4-3 IO68-M TERMINAL BLOCK PIN-OUT

| Row 1 | Description | Row 2 | Description | Row 3 | Description |
|-------|-------------|-------|-------------|-------|-------------|
| 1 | X Step | 24 | X Direction | 47 | X Auxiliary |
| 2 | X Phase A+ | 25 | X Phase B+ | 48 | X Index + |
| 3 | X Phase A- | 26 | X Phase B- | 49 | X Index - |
| 4 | X Limit + | 27 | X Limit- | 50 | X Home |
| 5 | 5VDC | 28 | X Servo | 51 | Digital GND |
| 6 | Y Step | 29 | Y Direction | 52 | Y Auxiliary |
| 7 | Y Phase A+ | 30 | Y Phase B+ | 53 | Y Index + |
| 8 | Y Phase A- | 31 | Y Phase B- | 54 | Y Index - |
| 9 | Y Limit + | 32 | Y Limit- | 55 | Y Home |
| 10 | 5VDC | 33 | Y Servo | 56 | Digital GND |
| 11 | I/O-0 | 34 | I/O-3 | 57 | I/O-5 |
| 12 | I/O-1 | 35 | No Connect | 58 | I/O-6 |
| 13 | I/O-2 | 36 | I/O-4 | 59 | I/O-7 |
| 14 | 5VDC | 37 | Z Servo | 60 | Analog GND |
| 15 | Z Step | 38 | Z Direction | 61 | Z Auxiliary |
| 16 | Z Phase A+ | 39 | Z Phase B+ | 62 | Z Index + |
| 17 | Z Phase A- | 40 | Z Phase B- | 63 | Z Index - |
| 18 | Z Limit + | 41 | Z Limit- | 64 | Z Home |
| 19 | 5VDC | 42 | T Servo | 65 | T Auxiliary |
| 20 | T Step | 43 | T Direction | 66 | T Index + |
| 21 | T Phase A+ | 44 | T Phase B+ | 67 | T Index - |
| 22 | T Phase A- | 45 | T Phase B- | 68 | T Home |
| 23 | T Limit + | 46 | T Limit- | 69 | Digital GND |

4.5. UIO ADAPTER MODULE

The optional UIO is an adapter module designed to provide easy connection for each signal of the UMX. It is the same size as the UMX and be attached to the UMX with the included hardware and cable. The UIO incorporates four 15-pin D-sub connectors, for each axis one, and an additional 15-pin Dsub for I/O signals. +5VDC is provided on each connector for powering accessories such as sensors, motor driver modules, encoders, etc. The supplied voltage is limited to 1.25 A overall by a resettable fuse on the UMX. A blue LED lights up when 5VDC is present.

The UIO can be used with CBL15-5, a mating cable to the 15-pin connectors.

TABLE 4-4 UIO 15-PIN CONNECTOR PIN OUT

| J1 (X), J2 (Y), J3 (Z) , J4 (T) Connector pin outs | | J8 (IO) Connector pin outs | |
|---|---------------|---------------------------------------|---------------|
| Pin | Signal | Pin | Signal |
| 1 | GND | 1 | IO7 |
| 2 | Phase B- | 2 | IO6 |
| 3 | Phase B+ | 3 | IO5 |
| 4 | Index - | 4 | IO4 |
| 5 | Index + | 5 | IO3 |
| 6 | Phase A - | 6 | IO2 |
| 7 | Phase A + | 7 | IO1 |
| 8 | 5V | 8 | IO0 |
| 9 | Home | 9 | GND |
| 10 | Limit - | 10 | 5V |
| 11 | Limit + | 11 | GND |
| 12 | Servo | 12 | 5V |
| 13 | Auxiliary | 13 | GND |
| 14 | Direction | 14 | 5V |
| 15 | Step | 15 | GND |

TABLE 4-5 CBL15-5 WIRE COLOR TO PIN TABLE

| Pin | Wire Color |
|------------|-------------------|
| 1 | Black |
| 2 | Brown |
| 3 | Red |
| 4 | Orange |
| 5 | Yellow |
| 6 | Dark Green |
| 7 | Blue |
| 8 | Purple |
| 9 | Grey |
| 10 | White |
| 11 | Pink |
| 12 | Light Green |
| 13 | Black/White |
| 14 | Brown/White |
| 15 | Red/White |

TABLE 4-6 34-PIN CONNECTOR PIN OUTS

| J5 Connector | | J6 Connector | |
|--------------|--------------------|--------------|------------------|
| Pin | End Signal | Pin | End Signal |
| 1 | GND | 1 | 5V |
| 2 | J8-Pin7 IO1 | 2 | J8-Pin8 IO0 |
| 3 | J8-Pin5 IO3 | 3 | J8-Pin6 IO2 |
| 4 | J8-Pin3 IO5 | 4 | J8-Pin4 IO4 |
| 5 | J8-Pin1 IO7 | 5 | J8-Pin2 IO6 |
| 6 | GND | 6 | 5V |
| 7 | J1-Pin5 X Index+ | 7 | J1-Pin12 X Servo |
| 8 | J1-Pin4 X Index- | 8 | J1-Pin15 X Step |
| 9 | J1-Pin7 X Phase A+ | 9 | J1-Pin13 X Aux |
| 10 | J1-Pin6 X Phase A- | 10 | J1-Pin14 X Dir |
| 11 | J1-Pin3 X Phase B+ | 11 | J1-Pin11 X Lmt+ |
| 12 | J1-Pin2 X Phase B- | 12 | J1-Pin10 X Lmt- |
| 13 | J2-Pin12 Y Servo | 13 | J1-Pin9 X Home |
| 14 | J2-Pin5 Y Index+ | 14 | J2-Pin15 Y Step |
| 15 | J2-Pin4 Y Index- | 15 | J2-Pin13 Y Aux |
| 16 | J2-Pin7 Y Phase A+ | 16 | J2-Pin14 Y Dir |
| 17 | J2-Pin6 Y Phase A- | 17 | J2-Pin11 Y Lmt+ |
| 18 | J2-Pin3 Y Phase B+ | 18 | J2-Pin10 Y Lmt- |
| 19 | J2-Pin2 Y Phase B- | 19 | J2-Pin9 Y Home |
| 20 | GND | 20 | 5V |
| 21 | J3-Pin5 Z Index+ | 21 | J3-Pin12 Z Servo |
| 22 | J3-Pin4 Z Index- | 22 | J3-Pin15 Z Step |
| 23 | J3-Pin7 Z Phase A+ | 23 | J3-Pin13 Z Aux |
| 24 | J3-Pin6 Z Phase A- | 24 | J3-Pin14 Z Dir |
| 25 | J3-Pin3 Z Phase B+ | 25 | J3-Pin11 Z Lmt+ |
| 26 | J3-Pin2 Z Phase B- | 26 | J3-Pin10 Z Lmt- |
| 27 | J4-Pin12 T Servo | 27 | J3-Pin9 Z Home |
| 28 | J4-Pin5 T Index+ | 28 | J4-Pin15 T Step |
| 29 | J4-Pin4 T Index- | 29 | J4-Pin13 T Aux |
| 30 | J4-Pin7 T Phase A+ | 30 | J4-Pin14 T Dir |
| 31 | J4-Pin6 T Phase A- | 31 | J4-Pin11 T Lmt+ |
| 32 | J4-Pin3 T Phase B+ | 32 | J4-Pin10 T Lmt- |
| 33 | J4-Pin2 T Phase B- | 33 | J4-Pin9 T Home |
| 34 | GND | 34 | 5V |

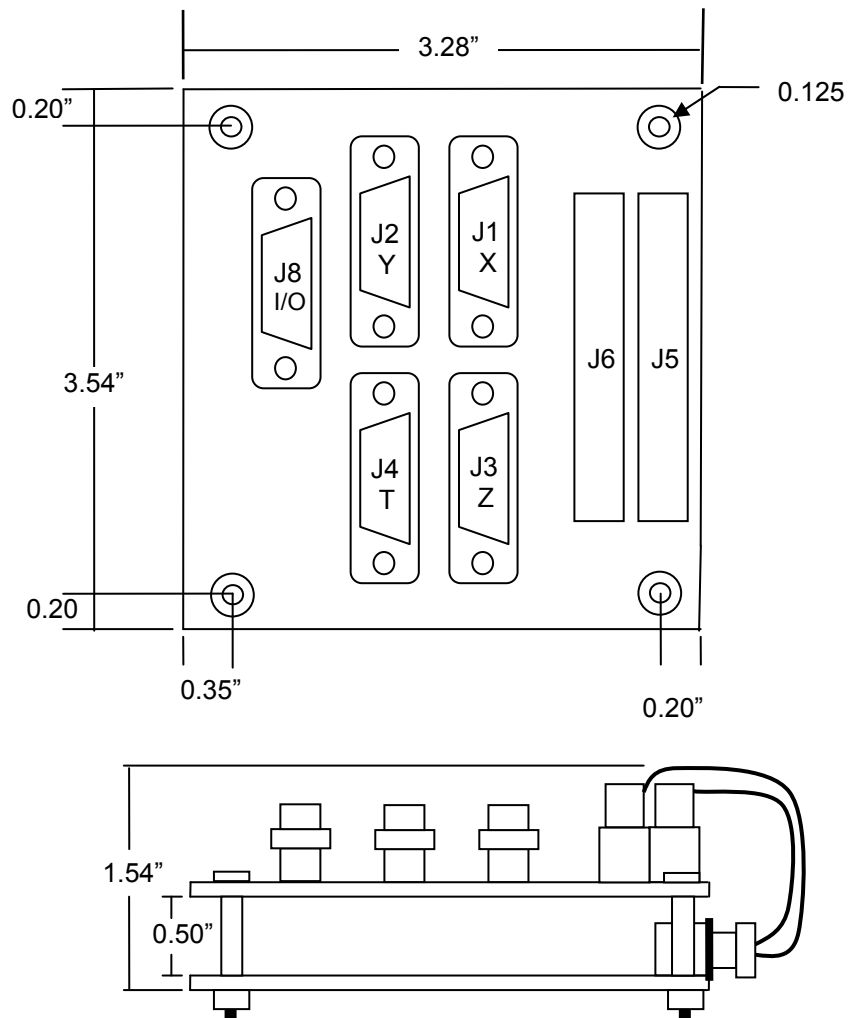


FIGURE 4-5 UIO DIMENSIONAL DRAWING

4.6. ENCODER FEEDBACK

Incremental encoder feedback is provided for each axis. The UMX encoder option accepts quadrature pulse inputs from high resolution encoders at rates up to 12 MHz (after quadrature detection). The encoder monitors actual position through the encoder pulse train. It then continuously calculates the position error on the servo axes and for servo axes the output based on that error through the PID filter. The stepper axes can monitor the error and correct the position after the move is finished. The encoder input can also be used as an independent feedback source. All modes are capable of slip/stall detection and encoder tracking with electronic gearing. These options are selectable by the user through software commands.

4.7. ENCODER SELECTION AND COMPATIBILITY

The UMX is compatible with virtually any incremental encoder which provides quadrature outputs. Times four quadrature detection is used to increase resolution. This means that an encoder rated for 1024 counts (or lines) per revolution will result in 4096 counts. The inputs are compatible with encoders, which have single ended or differential TTL outputs. The UMX inputs have built in hysteresis to minimize effects of noise pickup. The UMX has differential line receivers to accommodate encoders with differential line driver outputs.

4.8. HOME PROCEDURES

When single ended encoders are used the unused negative inputs, i.e. Phase A-, Phase B-, etc. must be biased at or near +1.5V. The IO68 provides convenient switches for this. Single ended encoders with long leads are not recommended.

Two modes are provided to synchronize the physical hardware with the UMX controller, i.e. put the controlled motor in the home position.

[HS](#) mode (factory default):

The home switch input is a TTL level input signal that can be used to physically home a mechanical stage. This signal can be either a logic HIGH or logic LOW true by using the [HH](#) and [HL](#) commands. The [HM](#) or [HR](#) commands are used after reducing the velocity to no more than 2048 pulses per second. This limit on velocity is necessary to avoid ambiguity of the home position if more than one pulse occurs per sample interval. When this functionality is used the axis position counter will be reset to a select value when the switch is activated. At this point the UMX can either ramp the axis to a stop or stop the axis immediately. The control of the direction of travel, the logic active state, and the response to the active switch are controlled through commands.

[HE](#) mode:

UMX home inputs can be used with encoders which provide one home pulse for the complete travel of the stage. The index input uses internal logic to establish the home position when used with the [HE](#) command mode. The default home position consists of the logical AND of the encoder index pulse, the home enable external input (LOW true only) and a single quadrant from the encoder logic. The home enable pulse must be true for less than one revolution of the encoder thus allowing only one home for the complete travel of the stage. The home logic expressed in Boolean terms is:

$$\text{Home} = \text{PhaseA} \times \overline{\text{PhaseB}} \times \text{Index} \times \overline{\text{Home Switch}}$$

It is necessary that the above quadrant occur within the index pulse as provided by the encoder for this logic to function properly. It may be necessary with some encoders to shift the phase of this quadrant by inverting one or both of the phases. Inverting one phase or swapping Phase A for Phase B will also reverse the direction. The encoder counter (read by a [RE](#) command) must increase for positive moves or the system will oscillate due to positive feedback. This is the default for the [HE](#) mode and is compatible with the PC68 family.

NOTE: The UMX has additional options for the [HE](#) mode. (See details in the [HE](#) command description.)

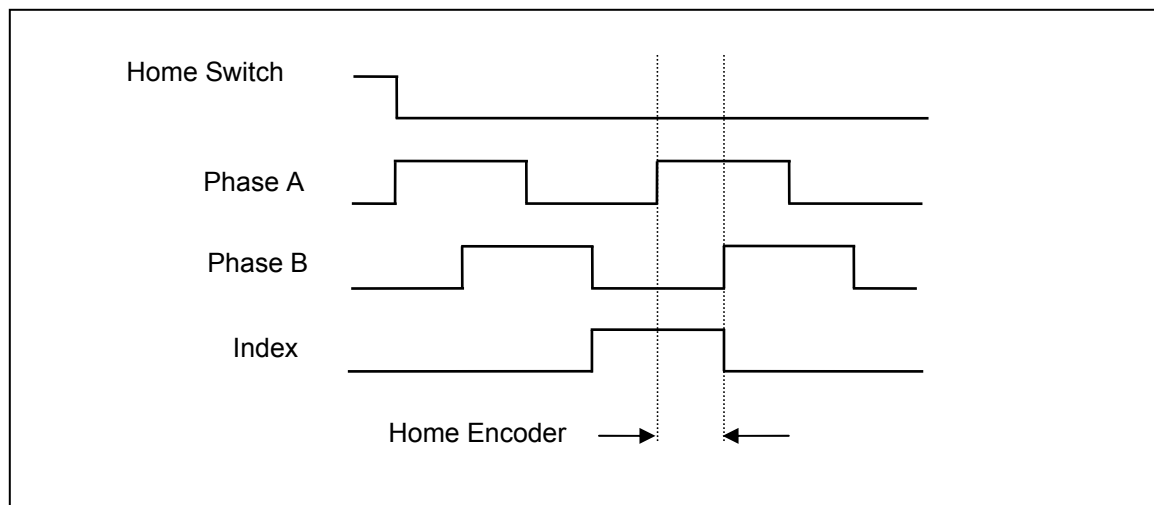


FIGURE 4-6 ENCODER HOMING STATE DETECTION (DEFAULT)

This page is intentionally left blank.

5. COMMAND STRUCTURE

5.1. INTRODUCTION

An extensive command structure is built into the UMX family of intelligent motor controls. It includes over 200 commands, a parameter buffer for each axis, and a command loop counter which allows for multiple executions of almost any command string. Please visit the <http://www.omsmotion.com/> website for the latest product releases.

The following commands in this section are included in the UMX family of controllers. Most of the commands are two ASCII characters and may be in upper or lower case. Some of the commands expect a numerical operand to follow. These commands are identified with “#” after the command. The operand must be terminated by a space, carriage return, or semi-colon to indicate the end of the number. No terminator is required on the other commands, but it is recommended to improve readability. Semi-colons are the preferred termination character because they are visible in your code. The operand must immediately follow the command with no space or separation character. The “#” indicates a signed integer input parameter or a signed fixed point number of the format ##.# when user units are enabled. With user units enabled distances, velocity and acceleration parameters may be input in inches, revolutions, etc.

Some commands that are usable in both single-axis and multi-axis modes do not take a parameter in single-axis mode. These commands require numeric parameters in multi-axis modes, and the parameters indicate whether or not to take action for each axis. If a parameter exists for an axis, then the command affects that axis and if the parameter does not exist for that axis, then the command has no affect on that axis. For example, the single-axis format of [LN](#) (Limits On) is simply "[LN](#)" without any parameters of any kind. The multi-axis format of [LN](#) is "[LN](#)b,b,b,b,b;" for 4-axis systems where 'b' represents the parameter for the corresponding axis. Like other multi-axis commands, a 'b' parameter may be omitted if that axis is to remain unchanged and the command may be prematurely terminated with a semicolon. Each 'b' position, if used, can be any numeric value. For example, to enable the Y and Z axes limit switches and leave the X and T axes unchanged, send the command "[LN](#),1,100;". The 1 and 100 parameters could be any numeric value whatsoever, and the effect of the command would be the same. For example, the following commands are equivalent:

```
"LN,1,1;"
```

```
"LN,0,0;"
```

```
"LN,50,99;"
```

Synchronized moves may be made by entering the [AA](#) command. This command performs a context switch which allows entering the commands in the format [MR](#)x#,y#,z#,t#, Numbers are entered for each axis which is to be commanded to move. An axis may be skipped by entering a comma with no parameter. The command may be prematurely terminated with a “;”, i.e. a move requiring only the X and Y axes would use the command [MR](#)x#,y#; followed by the [GO](#) command. Each axis programmed to move will start together upon executing the [GO](#) command. The UMX can be switched back to the unsynchronized mode by entering the desired single axis command such as [AX](#).

The [AM](#) command is provided for complex applications where the host manages multiple motion processes by a multitasking operating system. This mode shares the same


instructions as the [AA](#) mode, but allows starting a task while some other task involving one or many axes is active. For example, the X and Y axes could be doing linear interpolation while the Z axis is making an unrelated move simultaneously.


Constant velocity contouring provides another mode wherein the move parameters are predefined by entering [AA](#) then [CD](#). The UMX will then calculate the move profile in advance and move at constant velocity in the prescribed pattern. It can do linear interpolation on as many as 4 axes between the predefined points or it can do circular interpolation mixed with linear on two axes.

5.2. COMMAND QUEUES

The input characters are placed in a character buffer on input then removed and interpreted. The commands are then placed in separate command queues for each axis. As they are executed the space is reclaimed allowing the host to pass commands ahead of the moves actually being processed. Most of the commands are placed in the appropriate command queue for execution, while others are executed immediately allowing return of status information in a timely manner rather than when encountered in the command stream. This information is provided in a table for each command which shows the queue requirements, if any, and indicates immediate in those cases where the command is not queued. The queue requirements shown in the tables are typical. Depending on the circumstances in which the command is issued, the actual queue requirement may vary slightly. The single axis cases are indicated by the mode reference indicating the appropriate axis. The synchronized mode is indicated by the mode identifier [AA](#) or [AM](#). The contouring case is indicated by [AA/CD](#) for multiple axes in contour definition mode. The [RQ](#) command may be used to determine the actual space available at any time. The queues operate independently allowing each axis to perform separate processes concurrently. The synchronized modes ([AA](#)) insert special wait op-codes which allow the axes to be synchronized in this mode. When the commands are nested within loops, the queue space is not reclaimed until after the loop has been executed the programmed number of times. For loops larger than the queue space, the loop may never be completed since it cannot reclaim the queue space and cannot accept the loop terminator. The [RQ](#) command may be used to examine the remaining queue space. A Control-D may clear this condition if the input character queue is not also filled since it bypasses the command interrupter.


Some commands are valid only for stepper axes, others for stepper axes with encoder feedback, and still others for servo axes. Most are valid for all three types or some combination of types. A set of symbols to the right of each command identifies which motor types with which each command may be used. The symbols' meanings are as follows:

 Stepper motor with or without an encoder

 Stepper motor with an encoder

 Servo motor

If a command is usable with one of these motor types, the symbol will appear in black. If the command is not usable with a motor type, that motor symbol will be displayed in gray:

 This command is not usable with servo motors

 Indicates an example.

5.3. COMMAND SUMMARY

The following commands are included in the UMX family of motor controllers.

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| AA | 5-25 | C | Any following commands are for the AA mode |
| AC | 5-26 | C | Set acceleration/deceleration register |
| ?AC | 5-27 | Q | Report AC command |
| ?AD | 5-27 | Q | Report default auxiliary bit state |
| ADH | 5-28 | C | Set auxiliary default to high |
| ADL | 5-29 | C | Set auxiliary default to low |
| AF | 5-30 | C | Auxiliary off |
| AM | 5-31 | C | Axes multitasking mode |
| AN | 5-32 | C | Auxiliary on |
| AP | 5-33 | C | Make the current parameter set the power up default values |
| ?AQ | 5-34 | Q | Query current axis |
| AT | 5-34 | C | Any following commands are for the T axis |
| AX | 5-35 | C | Any following commands are for the X axis (default on reset) |
| AY | 5-35 | C | Any following commands are for the Y axis |
| AZ | 5-36 | C | Any following commands are for the Z axis |
| ?BD | 5-36 | Q | Report the direction of a general purpose I/O |
| BH | 5-37 | C | Set selected I/O bit high (off) |
| BI | 5-38 | C | Bipolar, set the analog torque outputs to bipolar |
| BL | 5-39 | C | Set selected I/O bit low (on) |
| BS | 5-40 | C | Set all bits of the general purpose output port to the state specified by the hex argument |
| ?BS | 5-41 | Q | Report the state of the specified I/O bit |
| BW | 5-41 | C | Wait for input to go low |
| BX | 5-42 | Q | Report bit status in hex format |
| CA | 5-43 | C | Clear done flag of currently addressed axis |
| CB | 5-43 | C | Clear all macro links to input bits |
| CD | 5-44 | C | Define a contour |
| CE | 5-46 | C | End contour definition, ramp to a stop |
| CG | 5-47 | C | Execute a constant velocity contour and prevent MT parsing |
| CK | 5-49 | C | End contour definition, immediately stop step pulses |
| CN | 5-50 | C | Cosine on, enable cosine velocity profiles |
| CR | 5-51 | C | Circular interpolation, move in a circle |
| CV | 5-52 | C | Contouring velocity, definition |
| CW | 5-52 | C | Clear while flag, i.e. terminate WH/WG loop |
| CX | 5-53 | C | Contour execute |
| ?DA | 5-54 | Q | Report a custom ramp |
| DAB | 5-55 | C | Define custom ramp breakpoint |

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| DAE | 5-55 | C | End custom ramp definition |
| DAR | 5-56 | C | Begin custom ramp definition |
| ?DB | 5-57 | Q | Report direction bit logic |
| DBI | 5-58 | C | Invert direction bit |
| DBN | 5-59 | C | Normalize direction bit |
| DC | 5-60 | C | Set the deceleration rate that will be used by the GU command |
| ?DE | 5-61 | Q | Report an acceleration ramp definition table entry |
| ?DS | 5-62 | Q | Report the size of a custom acceleration ramp table |
| DZ | 5-63 | C | Offset coefficient, used in open-loop mode |
| ?DZ | 5-64 | Q | Report DAC open-loop offset |
| EA | 5-65 | Q | Report encoder status axis |
| EF | 5-66 | C | Echo off, (default at power up) |
| EH | 5-67 | C | Encoder home definition |
| ?EH | 5-68 | Q | Report encoder home |
| EN | 5-69 | C | Echo on, turn on echo to host |
| ER | 5-70 | C | Set encoder count to motor count ratio |
| ?ER | 5-71 | Q | Report motor:encoder ratio |
| ES | 5-72 | C | Encoder slip/stall tolerance |
| ?ES | 5-73 | Q | Report encoder slip/stall tolerance |
| ET | 5-74 | C | Encoder tracking |
| FL | 5-75 | C | Flush an axis command queue |
| FP | 5-76 | Q | Force position, flush queue and attempt to stop |
| FX | 5-77 | C | Enable axis gantry mode |
| GD | 5-78 | C | Go and reset done flags |
| GN | 5-80 | C | Go and notify when done |
| GO | 5-81 | C | Go command, start execution of motion |
| GS | 5-82 | C | Go and use the home switch to monitor for motor slip/stall |
| GU | 5-83 | C | Go and use the AC values to accelerate and the DC values to decelerate |
| HD | 5-84 | C | Hold deadband, specify tolerance for position hold |
| ?HD | 5-84 | Q | Report position maintenance deadband |
| HE | 5-85 | C | Encoder home mode, set home on encoder logic |
| HF | 5-86 | C | Hold off, disable position hold, slip/stall detection and tracking modes |
| HG | 5-87 | C | Hold gain, specify position hold gain parameter |
| ?HG | 5-87 | Q | Report position maintenance gain |
| HH | 5-88 | C | Home high, home switches are active high |
| HL | 5-89 | C | Home low, home switches are active low |
| HM | 5-90 | C | Home, find home and initialize the position counter |

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| HN | 5-92 | C | Hold on, enable position correction after move |
| HR | 5-93 | C | Home reverse, find home in reverse direction and initialize position counter |
| HS | 5-95 | C | Home switch, enable home switch mode |
| HV | 5-96 | C | Hold velocity, specify maximum position hold correction velocity |
| ?HV | 5-97 | Q | Report position maintenance velocity |
| IC | 5-98 | C | Interrupt clear, clear done interrupt status and error flags |
| ID | 5-99 | C | Interrupt host when done and set done flag |
| II | 5-100 | C | Interrupt independent |
| IN | 5-101 | C | Interrupt when nearly done |
| IO | 5-102 | C | Designates I/O bits as inputs or outputs |
| IP | 5-103 | C | Interrupt when in position |
| IS | 5-104 | C | Interrupt slip/stall, interrupts host on slip/stall detection |
| IX | 5-105 | C | Interrupt when done. Sends Hex character via RS-232 |
| JF | 5-106 | C | Jog at fractional rates |
| JG | 5-107 | C | Jog command, run motor at specified velocity |
| KA | 5-109 | C | Acceleration feedforward coefficient, used in tuning servo systems |
| ?KA | 5-109 | Q | Report acceleration feedforward |
| KB | 5-110 | C | PID upper bound limit coefficient |
| ?KB | 5-110 | Q | Report axis PID upper bound limit |
| KD | 5-111 | C | Derivative gain coefficient, used for PID filter |
| ?KD | 5-111 | Q | Report PID derivative gain |
| KF | 5-112 | C | Set servo axis PID friction coefficient |
| ?KF | 5-112 | Q | Report servo axis friction offset |
| KI | 5-113 | C | Integral gain coefficient, used for PID filter |
| ?KI | 5-113 | Q | Report PID integral gain |
| KL | 5-114 | C | Kill |
| KM | 5-115 | C | Home and kill pulse generation |
| KO | 5-116 | C | Offset coefficient, used in closed-loop mode |
| ?KO | 5-116 | Q | Report PID closed-loop offset |
| KP | 5-117 | C | Proportional gain coefficient, used for PID filter |
| ?KP | 5-117 | Q | Report PID proportional gain |
| KR | 5-118 | C | Home in reverse and kill pulse generation |
| KS | 5-119 | C | Kill selected axes |
| KU | 5-120 | C | PID integration sum upper limit |
| ?KU | 5-120 | Q | Report PID integration sum upper limit |
| KV | 5-121 | C | Velocity feedforward coefficient, used in tuning servo systems |
| ?KV | 5-121 | Q | Report velocity feedforward |
| LA | 5-122 | C | Linear ramp selection per axis |
| LE | 5-123 | C | Loop end, terminate most recent LS command |
| LF | 5-124 | C | Disable limit switches for selected axis |

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| LH | 5-125 | C | Limit high, limit switch is active high |
| LL | 5-126 | C | Limit low, limit switch is active low |
| LN | 5-127 | C | Enable limit switches for selected axis |
| LO | 5-128 | C | Load monitor position |
| LP | 5-129 | C | Load position |
| LS | 5-130 | C | Loop start |
| ?LS | 5-132 | Q | Report limit active state |
| MA | 5-133 | C | Move absolute, move to absolute position |
| MD | 5-135 | C | Define a temporary macro |
| ML | 5-136 | C | Move linear |
| MM | 5-137 | C | Move minus |
| MP | 5-137 | C | Move plus |
| MR | 5-138 | C | Move relative |
| MT | 5-140 | C | Move to |
| MV | 5-141 | C | Move velocity |
| MX | 5-143 | C | Execute a macro command string |
| NV | 5-143 | C | Set a new contour velocity from within a contour |
| PA | 5-144 | C | Power automatic |
| ?PA | 5-145 | Q | Report power automatic state |
| PE | 5-146 | Q | Report encoder positions |
| PF | 5-146 | C | Parabolic off |
| PM | 5-147 | C | Print a macro command string |
| ?PM | 5-147 | Q | Report PID state |
| PN | 5-149 | C | Parabolic on |
| PP | 5-150 | Q | Report motor positions of all axes |
| PR | 5-151 | C | Parabolic ramp selection per axis |
| PS | 5-152 | Q | Reports macro link to specified UMX input bit |
| PT | 5-153 | C | Preserve a temporary macro by copying it to non-volatile memory |
| QA | 5-154 | Q | Report status of switches and flags for addressed axis without affecting flags |
| QI | 5-154 | Q | Report status of switches and flags on all axes without affecting flags |
| QL | 5-155 | Q | Report the state of the limit sensor inputs |
| RA | 5-156 | Q | Report status of switches and flags and reset flags |
| RB | 5-157 | Q | Report programmed direction of I/O bits in hex format |
| RC | 5-157 | Q | Report current acceleration or deceleration of the current axis |
| RD | 5-158 | C | Restore the current parameter set to the power up default values |
| RE | 5-158 | Q | Request encoder position, return current encoder position |
| RF | 5-159 | C | Restore factory defaults |
| RI | 5-159 | Q | Report status of switches and flags for all axes and reset flags |
| RL | 5-160 | Q | Report slip/stall status of each axis |

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| RM | 5-161 | Q | Report remainder of position divided by parameter in position counter |
| RP | 5-162 | Q | Request position, returns current position |
| RQ | 5-163 | Q | Request queue status, return number of queue entries available |
| RS | 5-163 | C | Software reset of UMX |
| ?RT | 5-164 | Q | Report ramp type |
| RU | 5-165 | Q | Return current position in user units |
| RV | 5-166 | Q | Return current velocity at which the axis is moving |
| SA | 5-167 | C | Stop all, flush queue and stops all axes with deceleration |
| SB | 5-168 | C | Sets the controller's serial communications baud rate |
| ?SB | 5-168 | Q | Report current baud rate setting |
| SC | 5-169 | C | Cosine ramp selection per axis |
| SD | 5-170 | C | Stop all axes and clear any done flags |
| SE | 5-171 | C | Set settling time before power is reduced in PA mode |
| ?SE | 5-171 | Q | Report settling time |
| SF | 5-172 | C | Soft limit off, restore normal overtravel operation |
| SI | 5-173 | C | Stop selected motors |
| SK | 5-174 | C | Links KILL function to specified UMX input bit |
| SL | 5-175 | C | Soft limit mode, allow pulse train to ramp down on overtravel |
| ?SL | 5-176 | Q | Report soft limit status |
| SM | 5-176 | C | Enables/Disables stand-alone mode |
| SO | 5-177 | C | Stop at a designated position using a specified ramp down distance |
| ?SO | 5-178 | Q | Report analog output mode |
| SP | 5-178 | C | Stop at position, stop at specified position if possible after all commands have been executed |
| SR | 5-179 | C | Selects custom ramp |
| ST | 5-180 | C | Stop, flush queue and decelerate to stop |
| ?SV | 5-181 | Q | Report servo voltage inversion state |
| SVI | 5-182 | C | Invert servo voltage |
| SVN | 5-182 | C | Normalize servo voltage |
| SW | 5-183 | C | Sync wait, wait for the input bit to be released by other controllers |
| SX | 5-185 | C | Links macro to specified UMX input bit |
| TF | 5-187 | C | Turn encoder slip/stall kill off |
| TL | 5-188 | C | Set software travel limits |
| ?TL | 5-189 | Q | Report software overtravel ranges |
| TM | 5-190 | C | Jog at the current velocity for the specified number of milliseconds |
| TN | 5-191 | C | Turn encoder slip/stall kill on |
| TX | 5-192 | C | Track the X axis |

| ALPHABETICAL COMMAND SUMMARY C = COMMAND, Q = QUERY | | | |
|--|--------|------|--|
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| UF | 5-193 | C | User units off, turn off user unit translation |
| UN | 5-194 | C | Unipolar, set the analog torque outputs to unipolar |
| UU | 5-195 | C | User units, multiply acceleration, velocity and distance parameters by specified parameter |
| ?UU | 5-196 | Q | Report axis user units' axis assignment |
| VB | 5-197 | C | Base velocity, set base velocity |
| ?VB | 5-198 | Q | Report axis base velocity |
| VL | 5-199 | C | Set maximum velocity to be used in profile |
| ?VL | 5-200 | Q | Report axis velocity limit |
| VS | 5-201 | C | Velocity stream, slave velocity mode for profiling |
| WA | 5-202 | C | Wait until all moves on all axes are finished |
| WD | 5-203 | C | While end, WS loop terminator |
| WG | 5-203 | C | Terminate WH loop |
| WH | 5-204 | C | While, execute all commands until WG loop terminator, until flag cleared by CW command |
| WQ | 5-206 | C | Wait until current axis queue is empty |
| WS | 5-207 | C | While sync, execute while sync is true |
| WT | 5-208 | C | Wait, wait for specified number of milliseconds |
| WY | 5-209 | Q | Who are you, returns model and software revision |

5.3.1.AXIS SPECIFICATION COMMANDS

The following commands set the context to direct the commands which follow to the appropriate axis. They remain in effect until superseded by another command of the same type, specifying a different axis.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| AA | 5-25 | C | Any following commands are for the AA (All Axes) mode |
| AM | 5-31 | C | Axes multitasking mode |
| AT | 5-34 | C | Any following commands are for the T axis |
| AX | 5-35 | C | Any following commands are for the X axis (default on reset) |
| AY | 5-35 | C | Any following commands are for the Y axis |
| AZ | 5-36 | C | Any following commands are for the Z axis |
| ?AQ | 5-34 | Q | Query current axis |

5.3.2.SYSTEM CONTROL COMMANDS

These commands allow control of various system parameters and operating modes to allow the user to optimize the response of the system for his/her application needs.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| AP | 5-33 | C | Make the current parameter set the power up default values |
| ?DB | 5-57 | Q | Report motor direction bit logic |
| DBI | 5-58 | C | Invert motor direction bit |
| DBN | 5-59 | C | Normalize motor direction bit |
| EF | 5-66 | C | Echo off, turn off echo to host (default at power up) |
| EN | 5-69 | C | Echo on, turn on echo to host |
| HL | 5-89 | C | Home low, home switches are active low |
| HH | 5-88 | C | Home high, home switches are active high |
| LF | 5-124 | C | Disable limit switches for selected axis |
| LH | 5-125 | C | Limit high, limit switch is active high |
| LL | 5-126 | C | Limit low, limit switch is active low |
| LN | 5-127 | C | Enable limit switches for selected axis |
| ?LS | 5-132 | Q | Report limit active state |
| PR | 5-151 | C | Parabolic ramp selection per axis |
| RD | 5-158 | C | Restore the current parameter set to the power up default values |
| RF | 5-159 | C | Restore the current parameter set to the factory default values |
| RS | 5-163 | C | Software reset of UMX |
| SF | 5-172 | C | Soft limit off, restore normal overtravel operation |
| SL | 5-175 | C | Soft limit mode, allow pulse train to ramp down on overtravel |
| ?SL | 5-176 | Q | Report soft limit status |
| ?SV | 5-181 | Q | Report servo voltage invert selection |
| SVI | 5-182 | C | Invert servo voltage |
| SVN | 5-182 | C | Normalize servo voltage |
| TL | 5-188 | C | Set software travel limits |
| ?TL | 5-189 | Q | Report software overtravel ranges |

5.3.3.ACCELERATION RAMP COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| CN | 5-50 | C | Cosine on, enable cosine velocity profiles |
| ?DA | 5-54 | Q | Report a custom ramp |
| DAB | 5-55 | C | Define custom ramp breakpoint |
| DAE | 5-55 | C | End custom ramp definition |
| DAR | 5-56 | C | Begin custom ramp definition |
| ?DE | 5-61 | Q | Report an acceleration ramp definition table entry |
| ?DS | 5-62 | Q | Report the size of a custom acceleration ramp table |
| LA | 5-122 | C | Linear ramp selection per axis |
| PF | 5-146 | C | Parabolic off, disable parabolic ramps, i.e. linear ramps will be generated |
| PN | 5-149 | C | Parabolic on, enable parabolic ramps |
| ?RT | 5-164 | Q | Report ramp type |
| RS | 5-163 | C | Software reset of UMX |
| SC | 5-169 | C | Cosine ramp selection per axis |
| SR | 5-179 | C | Selects custom ramp |

5.3.4.USER I/O COMMANDS

The following commands are for accessing the bit I/O functions of the board. See also the [SW](#) (page 5-183) and [WS](#) (page 5-207) commands.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| ?AD | 5-27 | Q | Report default auxiliary bit state |
| ADH | 5-28 | C | Set auxiliary default to high |
| ADL | 5-29 | C | Set auxiliary default to low |
| AF | 5-30 | C | Auxiliary off |
| AN | 5-32 | C | Auxiliary on |
| ?BD | 5-36 | Q | Report the direction of a general purpose I/O |
| BH | 5-37 | C | Set selected I/O bit high (off) |
| BL | 5-39 | C | Set selected I/O bit low (on) |
| BS | 5-40 | C | Set all bits of the general purpose output port to the state specified by the hex argument |
| ?BS | 5-41 | Q | Report the state of the specified I/O bit |
| BX | 5-42 | Q | Return bit status in hex format |
| IO | 5-102 | C | Designates I/O bits as inputs or outputs |
| PA | 5-144 | C | Power automatic |
| ?PA | 5-144 | Q | Report power automatic state |
| RB | 5-157 | Q | Report programmed direction of I/O bits in hex format |
| SE | 5-171 | C | Set settling time before power is reduced in PA mode |
| ?SE | 5-171 | Q | Report settling time |

5.3.5.MOVE SPECIFICATION COMMANDS

These commands allow specification of move parameters. They allow move parameters to be tailored to the user's system requirements.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| AC | 5-26 | C | Acceleration, set acceleration/deceleration register |
| ?AC | 5-27 | Q | Report AC command |
| DC | 5-60 | C | Set the deceleration rate that will be used by the GU command |
| LP | 5-129 | C | Load position, load position counter with parameter |
| MA | 5-133 | C | Move absolute, move to absolute position |
| ML | 5-136 | C | Move linear, move specified distance relative from current position |
| MR | 5-138 | C | Move relative, move specified distance from current position |
| MT | 5-140 | C | Move to, move to specified position |
| VB | 5-197 | C | Base velocity, set base velocity |
| ?VB | 5-198 | Q | Report axis base velocity |
| VL | 5-199 | C | Set maximum velocity to be used in profile |
| ?VL | 5-200 | Q | Report axis velocity limit |

5.3.6.MOVE EXECUTION COMMANDS

These commands allow execution of the moves which have been previously specified.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| GD | 5-78 | C | Go and reset done flags |
| GN | 5-80 | C | Go and notify when done |
| GO | 5-81 | C | Go command, start execution of motion |
| GS | 5-82 | C | Go and use the home switch to monitor for motor slip/stall |
| GU | 5-83 | C | Go and use the AC values to accelerate and the DC values to decelerate |
| JG | 5-107 | C | Jog command, run motor at specified velocity until a new velocity command is sent or it is stopped by a stop or kill command |
| JF | 5-106 | C | Jog at fractional rates |
| TM | 5-190 | C | Jog at the current velocity for the specified number of milliseconds |
| VS | 5-201 | C | Velocity stream, slave velocity mode for profiling |

5.3.7.MOVE TERMINATION COMMANDS

The following commands allow termination of move sequences in process.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| FL | 5-75 | C | Flush an axis command queue |
| KL | 5-114 | C | Kill, flush queue and terminate pulse generation immediately on all axes without decelerating |
| KS | 5-119 | C | Kill selected axes |
| SA | 5-167 | C | Stop all, flush queue and stops all axes with deceleration |
| SD | 5-170 | C | Stop all axes and clear any done flags |
| SI | 5-173 | C | Stop selected motors |
| SO | 5-177 | C | Stop at a designated position using a specified ramp down distance |
| ST | 5-180 | C | Stop, flush queue and decelerate to stop |

5.3.8.LOOP CONTROL COMMANDS

These commands allow move sequences to be repeated within loops. Loops can be nested up to four levels deep on each axis.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| CW | 5-52 | C | Clear while flag, i.e. terminate WH / WG loop |
| LE | 5-123 | C | Loop end, terminate most recent LS command |
| LS | 5-130 | C | Loop start, set loop counter, from 1 to 32000 loops; (may be nested to 4 levels) |
| WD | 5-203 | C | While end, WS loop terminator |
| WG | 5-203 | C | Terminate WH loop |
| WH | 5-204 | C | While, execute all commands until WG loop terminator, until flag cleared by CW command |
| WS | 5-207 | C | While sync, execute while sync is true |

5.3.9. INITIALIZATION AND HOME CONTROL COMMANDS

These commands allow the initialization of the physical stage with the controller.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| HM | 5-90 | C | Home, find home and initialize the position counter |
| HR | 5-93 | C | Home reverse, find home in reverse direction and initialize position counter |
| KM | 5-115 | C | Home and kill pulse generation |
| KR | 5-118 | C | Home in reverse and kill pulse generation |

5.3.10. MOVE SYNCHRONIZATION COMMANDS

These commands allow the synchronization of moves with external events or multiple axis sequences.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| BW | 5-41 | C | Wait for input to go low |
| CA | 5-43 | C | Clear done flag of currently addressed axis |
| IC | 5-98 | C | Interrupt clear, clear done interrupt status and error flags |
| ID | 5-99 | C | Interrupt host when done and set done flag |
| II | 5-100 | C | Interrupt independent |
| IN | 5-101 | C | Interrupt when nearly done |
| IP | 5-103 | C | Interrupt when in position |
| IX | 5-105 | C | Interrupt when done. Sends Hex character via RS-232 |
| SW | 5-183 | C | Sync wait, wait for the input bit to be released by other controllers |
| WA | 5-202 | C | Wait until all moves on all axes are finished |
| WQ | 5-206 | C | Wait until current axis queue is empty |
| WT | 5-208 | C | Wait, wait for specified number of milliseconds |

5.3.11. SYSTEM STATUS REQUEST COMMANDS

These commands allow the host to request the status of various move parameters, including the status of limit and home switches.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| BX | 5-42 | Q | Return bit status in hex format |
| PE | 5-146 | Q | Report encoder positions of all encoder and servo axes |
| PP | 5-150 | Q | Report motor positions of all axes |
| QA | 5-154 | Q | Report status of switches and flags for addressed axis without affecting flags |
| QI | 5-154 | Q | Report status of switches and flags on all axes without affecting flags |
| QL | 5-155 | Q | Report the state of the Limit Sensor inputs |
| RA | 5-156 | Q | Report status of switches and flags and reset flags |
| RC | 5-157 | Q | Report current acceleration or deceleration of the current axis |
| RI | 5-159 | Q | Report status of switches and flags for all axes and reset flags |
| RM | 5-161 | Q | Report remainder of position divided by parameter in position counter |
| RP | 5-162 | Q | Request position, returns current position |
| RQ | 5-163 | Q | Request queue status, return number of queue entries available |
| RU | 5-165 | Q | Return current position in user units |
| RV | 5-166 | Q | Return current velocity at which the axis is moving |
| WY | 5-209 | Q | Who are you , returns model and software revision |

5.3.12. AXIS SCALING COMMANDS

The following commands allow specification of move parameters in user defined units. The OMS controls will automatically convert all move parameters to these units once they have been initialized.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| ER | 5-70 | C | Encoder ratio, set encoder count to motor count ratio |
| ?ER | 5-71 | Q | Report motor:encoder ratio |
| UF | 5-193 | C | User units off, turn off user unit translation |
| UU | 5-195 | C | User units, multiply acceleration, velocity and distance parameters by specified parameter |
| ?UU | 5-196 | Q | Report axis user units' axis assignment |

5.3.13. PID FILTER CONTROL COMMANDS

The following commands are valid only for servo axes and should never be executed while the specific axis is in motion.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| BI | 5-38 | C | Bipolar, set the analog torque outputs to bipolar |
| DZ | 5-63 | C | Offset coefficient, used in open-loop mode |
| ?DZ | 5-64 | Q | Report DAC open-loop offset |
| HF | 5-86 | C | Hold off, disables PID filter |
| HN | 5-92 | C | Hold on, enables PID filter |
| KA | 5-109 | C | Acceleration feedforward coefficient, used in tuning servo systems |
| ?KA | 5-109 | Q | Report acceleration feedforward |
| KB | 5-110 | C | PID Upper Bound Limit Coefficient |
| ?KB | 5-110 | Q | Report AXIS PID upper Bound limit |
| KD | 5-111 | C | Derivative Gain coefficient, used for PID filter |
| ?KD | 5-111 | Q | Report PID derivative gain |
| KF | 5-112 | C | Set Servo axis PID friction coefficient |
| ?KF | 5-112 | Q | Report servo axis friction offset |
| KI | 5-113 | C | Integral Gain coefficient, used for PID filter |
| ?KI | 5-113 | Q | Report PID integral gain |
| KO | 5-116 | C | Offset coefficient, used in closed-loop mode |
| ?KO | 5-116 | Q | Report PID closed-loop offset |
| KP | 5-117 | C | Proportional Gain coefficient, used for PID filter |
| ?KP | 5-117 | Q | Report PID proportional gain |
| KU | 5-120 | C | PID integration sum upper limit |
| ?KU | 5-120 | Q | Report PID integration sum upper limit |
| KV | 5-121 | C | Velocity feedforward coefficient, used in tuning servo systems |
| ?KV | 5-121 | Q | Report velocity feedforward |
| ?PM | 5-147 | Q | Report PID state |
| ?SO | 5-178 | Q | Report analog output mode |
| UN | 5-194 | C | Unipolar, set the analog torque outputs to unipolar |

5.3.14. POSITION MAINTENANCE COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| FX | 5-77 | C | Enable axis gantry mode |
| HD | 5-84 | C | Hold deadband, specify deadband tolerance for position hold |
| ?HD | 5-84 | Q | Report position maintenance deadband |
| HG | 5-87 | C | Hold gain, specify position hold gain parameter |
| ?HG | 5-87 | Q | Report position maintenance gain |
| HF | 5-86 | C | Hold off, disable position hold, slip/stall detection and tracking modes |
| HN | 5-92 | C | Hold on, enable position correction after move |
| HV | 5-96 | C | Hold velocity, specify maximum position hold correction velocity |
| ?HV | 5-97 | Q | Report position maintenance velocity |
| IP | 5-103 | C | Interrupt when in position |
| LO | 5-128 | C | Set axis motor position but not the encoder position |
| TX | 5-192 | C | Track the X axis |

5.3.15. SLIP AND STALL DETECTION COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| ES | 5-72 | C | Encoder slip/stall tolerance, set tolerance before slip/stall is flagged |
| ?ES | 5-73 | Q | Report encoder slip/stall tolerance |
| HF | 5-86 | C | Hold off, disable position hold, slip/stall detection and tracking modes |
| IS | 5-104 | C | Interrupt slip/stall, interrupts host on slip/stall detection |
| RL | 5-160 | Q | Report slip/stall status of each axis |
| TF | 5-187 | C | Turn encoder slip/stall kill off |
| TN | 5-191 | C | Turn encoder slip/stall kill on |

5.3.16. ENCODER TRACKING COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| ET | 5-74 | C | Encoder tracking, set encoder tracking mode |
| HF | 5-86 | C | Hold off, disable position hold, slip/stall detection and tracking modes |

5.3.17. ENCODER HOME CONTROL COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| EH | 5-67 | C | Encoder home definition |
| ?EH | 5-68 | Q | Report encoder home |
| HE | 5-85 | C | Encoder home mode, set home on encoder logic |
| HS | 5-95 | C | Home switch, enable home switch mode |

5.3.18. ENCODER STATUS REQUEST COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| EA | 5-65 | Q | Encoder status, return encoder status of currently addressed axis |
| PE | 5-146 | Q | Report encoder positions of all encoder and servo axes |
| RE | 5-158 | Q | Request encoder position, return current encoder position |

5.3.19. VELOCITY STAIRCASE COMMANDS

The following commands describe the velocity staircase mode. This mode is useful in applications requiring a change in velocity at a prescribed position without stopping.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| FP | 5-76 | C | Force position, flush queue and attempt to stop at specified position |
| MM | 5-137 | C | Move minus, set minus direction for MV type move |
| MP | 5-137 | C | Move plus, set positive direction for MV type move |
| MV | 5-141 | C | Move velocity, move to first parameter (absolute position) at second parameter velocity without stopping at end of move |
| SP | 5-178 | C | Stop at position, stop at specified position if possible after all commands have been executed |

5.3.20. CONSTANT VELOCITY CONTOURING COMMANDS

The UMX will attempt to generate any profile which it is asked to do. It is the responsibility of the host to be sure the acceleration required when generating a circle or any other change in direction is possible within the mechanical constraints of the system. All corners must be defined by arcs and tangents to those arcs, else the change in direction will be instantaneous and generate very large accelerations. The arc radius must be chosen so that the acceleration constraints of the system are met.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| CD | 5-44 | C | Define a contour |
| CE | 5-46 | C | End contour definition, ramp to a stop |
| CG | 5-47 | C | Execute a constant velocity contour and prevent MT parsing |
| CK | 5-49 | C | End contour definition, immediately stop step pulses |
| CR | 5-51 | C | Circular interpolation, move in a circle |
| CV | 5-52 | C | Contouring velocity, definition |
| CX | 5-53 | C | Contour execute |
| MT | 5-140 | C | Move to, move to specified position |
| NV | 5-143 | C | Set a new contour velocity from within a contour |
| RQ | 5-163 | Q | Request queue status, return number of queue entries available |

5.3.21. MACRO CONTROL COMMANDS

Macros are typically used as a shortcut to save some keystrokes. They can be used to save common parameter settings that may need to be recalled. They can also be used to store common command sequences that may be used for a particular process.

Once macros are defined, an entire command sequence can be sent to the controller through just executing the [MX](#) command and the appropriate macro number. As a result, sending a stream of frequently used commands to the controller is done simply through the use of one command. Detailed information regarding the macro commands is shown below.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| MD | 5-135 | C | Define a temporary macro |
| MX | 5-143 | C | Execute a macro command string |
| PM | 5-147 | Q | Print a macro command string |
| PT | 5-153 | C | Preserve a temporary macro by copying it to non-volatile memory |

5.3.22. STAND-ALONE COMMANDS

The Stand-alone mode allows a UMX Motion Controller to run in a completely independent operation mode when powered by a separate +5 VDC power supply (for steppers) and ± 12 VDC (for servos). This mode has several commands that can establish links to macros. When set up properly in this mode the UMX can scan for a predefined I/O Input bit, until it changes to the specified state. Upon sensing that this condition has been met, it will execute the permanent Macro from Non-volatile flash memory that had been previously associated or linked with this I/O bit and its state.

A common application the stand-alone mode is to incorporate the KILL ([KL](#)) function. Reference the [SK](#) command. This will allow the user to stop motion of the device.

All of these selections are temporary. They can be made permanent by executing the [AP](#) command, which assigns the current parameter values as the Power Up defaults.

Note: The [AP](#) command should be used sparingly as it causes a write to the on board Flash Memory and there is a finite amount of times that it can be re-written to (i.e. less than 10,000 times, typical).

Application Overview:

The setup of the stand-alone mode is performed through the communication interface by the use of the commands. The user would define the required motion and processes and store them in a macro. Then, with the use of the commands below, the execution of the specific macros would be defined. Once all of the setup is completed the controller would be put in the stand-alone mode ([SM](#)) and the execution of the macros controlled by the defined input bits.

There are no queue requirements for these commands.

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|--------|------|--|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| CB | 5-43 | C | Clear all macro links to input bits |
| PS | 5-152 | Q | Reports macro link to specified UMX input bit |
| SK | 5-174 | C | Links KILL function to specified UMX input bit |
| SM | 5-176 | C | Enables/Disables stand-alone mode |
| SX | 5-185 | C | Links macro to specified UMX input bit |

5.3.23. SERIAL COMMUNICATION CONFIGURATION COMMANDS

| COMMANDS SUMMARY BY SECTION | | | |
|-----------------------------|-----------|------|---|
| C = COMMAND, Q = QUERY | | | |
| COMMAND | PAGE # | TYPE | COMMAND DESCRIPTION |
| SB | 5-168 | C | Sets the controller's serial communications baud rate |
| ?SB | 5-168 | Q | Report current baud rate setting |

5.3.24. COMMAND DEFINITIONS

AA ALL AXES



The AA command performs a context switch to multi-axis mode. All commands entered after this one will be treated as “all axes” commands which must be formatted for multi-axis use rather than single-axis use. Each command will be executed in the order in which it is received. This is true even if the second command affects axes other than those affected by the first command. For example, if AA mode is entered followed by a move of the X axis and then a move of the Y axis, the Y axis move will not begin until the X axis move has completed.



Example: Perform an absolute move using the X and Y axes. When this move is complete, perform a relative move using the Y, Z, and T axes.

Enter: AA ;
 [MA](#)12000,14000;
 [GO](#);
 [MR](#),5000,1500,100000;
 [GO](#);

Response: None

NOTE: This command changes the axis mode immediately, but has axis queue requirements because it places synchronization entries in all axis queues.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AA; | AX - AT | 1 | 1 |
| AA; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [AM](#), [?AQ](#), [AT](#), [AX](#), [AY](#), [AZ](#), [CD](#)

AC ACCELERATION



The AC command sets the acceleration/deceleration register to the operand which follows the command. The parameter must be greater than zero (zero is not valid) and less than 8,000,000, and the unit is in steps per second per second. All the following move commands for the axis being programmed will accelerate and decelerate at this rate until another AC command is entered. See the [AP](#) command, page 5-33 to preserve the AC settings as the power-up/reset values. The default value is 2,000,000.

RANGE: $1 \leq AC \leq 8000000$

Note: The range denotes values after User Units have been applied. (See [UU](#) command.)



Example: In the single axis mode, set the Y axis acceleration to 200,000 counts per second per second.

Enter: [AY](#);
AC200000;

Response: None



Example: In the [AA](#) mode, set the acceleration of the X axis to 200,000 and the Z axis to 50,000 and leave the other axes with their previous values.

Enter: [AA](#) ;
AC200000,,50000;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|----------------|---|---------|--------------------------|
| FORMAT | MODE | Axis Ramp Type | COMMAND | ARGUMENT |
| AC#; or AC#,#,#,; | AX-AT or AA-AM | Linear (LA or PF) | 3 | 3 |
| AC#; or AC#,#,#,; | AX-AT or AA-AM | Short Parabolic (PN0 ; or PR0 ;)) | 3 | 5 |
| AC#; or AC#,#,#,; | AX-AT or AA-AM | All other parabolic forms | 3 | 12 |
| AC#; or AC#,#,#,; | AX-AT or AA-AM | Custom Ramps (SR) | 3 | No. of ramp segments +12 |

Related commands: [?AC](#), [DC](#), [RC](#), [VB](#), [VL](#)

?AC REPORT AC COMMAND

This command will reply with the current acceleration value for the selected axis.



Example: Report the current [AC](#) value for this axis.

Enter: ?AC

Response: ac200000<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?AC | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [AC](#)

?AD REPORT DEFAULT AUXILIARY BIT STATE

This command reports the power up default selection for the axis aux bit.



Example: Report the power up state of the Y axis auxiliary bit

Enter: [AY](#);
?AD

Response: adh<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?AD | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ADH](#), [ADL](#)

ADH SET AUXILIARY DEFAULT TO HIGH



The ADH command sets the default power up or reset state of the auxiliary line for the current axis to high. This change is stored as a power up parameter in flash automatically and need not be stored via the AP command. Since this command writes to non-volatile memory it should be used only when necessary and not in repeatedly called functions.

NOTE: This command will also archive all other parameter values as power up defaults.



Example: Set the power up state of the Z axis auxiliary line to high

Enter: [AZ;](#)
ADH;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ADH; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?AD](#), [ADL](#)

ADL SET AUXILIARY DEFAULT TO LOW



The ADL command sets the default power up or reset state of the auxiliary line for the current axis to low. This change is stored in nonvolatile memory automatically and need not be stored via the [AP](#) command. Since this command writes to non-volatile memory it should be used only when necessary and not in repeatedly called functions.

NOTE: This command will also archive all other parameter values as power up defaults.



Example: Set the power up state of the Y axis auxiliary line to low

Enter: [AY](#);
ADL;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ADL; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?AD](#), [ADH](#)

AF AUXILIARY OFF

The AF command turns off the selected auxiliary outputs. That is, it causes the signal to be driven low. The AF command may be used to change power level on driver modules so equipped or as a user specified output. Note that this command will turn power automatic ([PA](#)) mode off.



Example: Turn off the Y axis auxiliary output in the single axis mode.

Enter: [AY](#);
AF;

Response: None



Example: Turn off the X and Z axes auxiliary outputs when in the [AA](#) command mode. The Y axis is unchanged in this example.

Enter: [AA](#);
AF1,,1;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|---------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| AF; | AX - AT | 1 | 0 | N/A |
| AFb,b,b,b; | AA-AM | 1 | 0 | N/A |
| AFb,b,b,b; | AA/CD | N/A | N/A | 2* |

*When AF is used in a contour definition the aux bits of all axes included in that definition will be turned off when the contour is executed.

Related commands: [AN](#), [BH](#), [BL](#), [BS](#), [PA](#)

AM AXES MULTITASKING



The AM mode allows several tasks to be managed simultaneously. This command changes the mode of all future commands to multi-axis mode. In this mode, a task may be performing coordination motion on 2 axes, while a second task is performing unrelated but simultaneous motion on another axis. All commands sent in this mode must be formatted for multi-axis mode rather than single-axis mode.



Example: Perform a coordinated relative move on the X and Y axes, while moving the T axis as a separate move at the same time.

Enter: AM;
[ML](#)2000,3000;
[GO](#);
[MA](#),,,10000;
[GO](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AM; | AX - AT | Immediate | |
| AM; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AA](#), [?AQ](#), [AT](#), [AX](#), [AY](#), [AZ](#)

AN AUXILIARY ON

The AN command turns on the selected auxiliary output ports. That is, it allows the signal to be pulled high. This is the default mode for the auxiliary line at power up or reset. The AN command may be used to change power level on driver modules so equipped, trigger another board's input or as a user specified output.

A parameter must be supplied for the desired axes when used in the [AA](#) mode so that the other axes are not affected. No parameter is required in a single axis mode. Note this command will turn power automatic ([PA](#)) mode off.



Example: Turn on the Y axis auxiliary output in the single axis mode.

Enter: [AY](#);
AN;

Response: None.



Example: Turn on the X and Z axes auxiliary outputs when in the [AA](#) command mode. The Y axis is unchanged in this example.

Enter: [AA](#);
AN1,,1;

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|---------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| AN; | AX - AT | 1 | 0 | N/A |
| ANb,b,b,b; | AA-AM | 1 | 0 | N/A |
| ANb,b,b,b; | AA/CD | N/A | N/A | 2* |

*When AN is used in a contour definition the Aux bits of all axes included in that definition will be turned on when the contour is executed.

Related commands: [AF](#), [BH](#), [BL](#), [BS](#), [PA](#)

AP**ASSIGN CURRENT
PARAMETERS AS POWER-UP DEFAULTS**

The AP command stores the current parameter set as the power-up default set of values. This is done by writing the current parameter set to flash memory. The following list of parameters will have their values saved to flash memory when this command is used: [AC](#), [BI/UN](#), [BR](#), [EH](#), [ER](#), [ES](#), [HD](#), [HG](#), [HH/HL](#), [HV](#), [KA](#), [KB](#), [KD](#), [KF](#), [KI](#), [KO](#), [KP](#), [KU](#), [KV](#), [LA/SC/PR](#), [LH/LL](#), [LN/LE](#), [PA0/PA1](#), [SE](#), [SF/SL](#), [VB](#), [VL](#), and [UU](#).

Note: This command should not be issued when an axis is in motion and it should be used sparingly because the flash memory has a limited number of write cycles.



Example: Save the current parameter set to be the power up default set of values.

Enter: [AP](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AP; | AX - AT | Immediate | |
| AP; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [RF](#)

?AQ QUERY CURRENT AXIS

The ?AQ command reports the mode that the current axis is in, i.e. [AA](#) mode, [AM](#) mode, etc.



Example: Determine the mode of the X axis.

Enter: [AX](#);
?AQ

Response: ax<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?AQ | AX - AT | Immediate | |
| ?AQ | AA-AM | Immediate | |
| ?AQ | AA/CD | Immediate | |

Related commands: [AA](#), [AM](#), [AT](#), [AX](#), [AY](#), [AZ](#)

AT AXIS T

The AT command directs all following commands to the T axis. Commands sent in this mode will be executed in the order in which they are received (subject to queued versus immediate execution) and are expected to be formatted for single-axis use.



Example: Move the T axis to absolute position -2468.

Enter: AT;
[MA](#)-2468;
[GO](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AT; | AX - AT | Immediate | |
| AT; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AA](#), [AM](#), [?AQ](#), [AX](#), [AY](#), [AZ](#)

AX**AXIS X**

The AX command directs all following commands to the X axis. This is the default mode at power up or reset. Commands sent in this mode will be executed in the order in which they are received (subject to queued versus immediate execution) and are expected to be formatted for single-axis use.



Example: Make the X axis step at a rate of 5,000 steps/second.

Enter: AX;
[JG](#)5000;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AX; | AX - AT | Immediate | |
| AX; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AA](#), [AM](#), [?AQ](#), [AT](#), [AY](#), [AZ](#)

AY**AXIS Y**

The AY command directs all following commands to the Y axis. Commands sent in this mode will be executed in the order in which they are received (subject to queued versus immediate execution) and are expected to be formatted for single-axis use.



Example: Examine the status of the Y axis.

Enter: AY;
[RA](#)

Response: PNNN<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AY; | AX - AT | Immediate | |
| AY; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AA](#), [AM](#), [?AQ](#), [AT](#), [AX](#), [AZ](#)

AZ**AXIS Z**

The AZ command directs all following commands to the Z axis. Commands sent in this mode will be executed in the order in which they are received (subject to queued versus immediate execution) and are expected to be formatted for single-axis use.



Example: Move the Z axis 2,000 steps at a rate of 500 steps per second.

Enter: AZ;
 VL500;
 MR2000;
 GO;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| AZ; | AX - AT | Immediate | |
| AZ; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AA](#), [AM](#), [?AQ](#), [AT](#), [AX](#), [AY](#)

?BD**REPORT BIT DIRECTION**

The ?BD command reports if a general purpose I/O bit is an input or output.



Example: Find out whether I/O bit 2 is configured as an input or an output.

Enter: ?BD2;

Response: If the bit is an input the response will be: io0<LF>
 An output bit response is: io1<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?BD | AX - AT | Immediate | |
| ?BD | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [BH](#), [BL](#), [BX](#), [IO](#), [RB](#)

BH BIT HIGH

The BH command turns the selected general purpose output off (i.e. logic high). The default state of general purpose outputs is off at power up or reset.

Note: Output bits should not be used as triggers for applying power to any device unless master power is applied separately and after the UMX is fully configured. The states of the outputs are unpredictable during power-up and reset and can toggle several times before settling at a high level.

Range: $0 \leq \text{bit number} \leq 7$



Example: Set general purpose bits 4 and 5 to high.

Enter: BH4;
BH5;

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|---------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| BH#; | AX - AT | 1 | 1 | N/A |
| BH#; | AA-AM | 2 | 1 | N/A |
| BH#; | AA/CD | N/A | N/A | 2 |

Related commands: [AF](#), [AN](#), [BL](#), [BS](#), [BX](#)

BI BIPOLAR

The BI command sets the analog servo output of the current axis to bipolar. When bipolar is selected, a zero torque reference will result in a 0VDC output (+/- offset voltage). The analog output will range between +10VDC and -10VDC when bipolar is enabled. The BI command is valid only in the single axis mode and is the default mode at power up or reset.



Example: Set up servo axis X for bipolar operation.

Enter: [AX](#);
BI;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| BI; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [DBI](#), [DBN](#), [?SO](#), [SVI](#), [SVN](#), [UN](#)

BL BIT LOW

The BL command turns the selected general purpose output line on (i.e. logic low). The default states of all output bits at power-up are logic high (off). The [BS](#) command can be used to set all outputs to a known state at once.

Note: Output bits should not be used as triggers for applying power to any device unless master power is applied separately and after the UMX is fully configured. The states of the outputs are unpredictable during power-up and reset and can toggle several times before settling at a high level.

RANGE: 0 < bit number < 7



Example: Turn on output bits 4 and 5 after a move. Note that this is only valid for output bits; input bits cannot be modified.

Enter: [AX](#);
[MA](#)1000;
[GO](#);
 BL4;
 BL5;

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|---------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| BL#; | AX - AT | 1 | 1 | N/A |
| BL#; | AA-AM | 2 | 1 | N/A |
| BL#; | AA/CD | N/A | N/A | 2 |

Related commands: [AF](#), [AN](#), [BS](#), [BX](#)

BS BIT SET

Set all of the output bits to a known state at the same time. This command will affect all output bits, setting their states to the specified bit mask nearly simultaneously. The mask must be in ASCII hex format where the least significant bit (bit 0) is on the right. To set a line low, the corresponding bit in the hex mask must be a 0. A one (1) in any bit position will set the corresponding bit high.

Note: Output bits should not be used as triggers for applying power to any device unless master power is applied separately and after the UMX is fully configured. The states of the outputs are unpredictable during power-up and reset and can toggle several times before settling at a high level.

Range: 00 < Hex number < FF



Example: Assume I/O bit direction [IO](#) = FFFF, all outputs. Set output 0 high, 1-3 low, and 4-6 high (71 = (hex) 01110001)

Enter: BS71;

Response: None.

NOTE: Data written to input bits has no effect

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| BS#; | AX – AT | 1 | 1 |
| BS#;- | AA-AM | 3 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [AF](#), [AN](#), [BH](#), [BL](#), [BX](#)

?BS REPORT BIT STATE

The ?BS command reports the state of the specified general purpose I/O bit.



Example: Determine the state of I/O bit 6.

Enter: ?BS6;

Response: If the bit is set to a TTL high the response will be:
1<LF>
If the bit is set to a TTL low the response will be:
0<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?BS | AX - AT | Immediate | |
| - | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [BH](#), [BL](#), [BX](#)

BW WAIT FOR INPUT TO GO LOW

The BW command is just like the [SW](#) command except that it waits for the input line to reach a TTL low rather than a TTL high. Refer to the [SW](#) command for more detail.

RANGE: $0 \leq \text{Bit Number} \leq 7$



Example: See the examples for the [SW](#) (see page 5-183) command

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| BW#; | AX - AT | 1 | 1 |
| BW#; | AA-AM | 3 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [SW](#), [WA](#), [WT](#), [WQ](#)

BX BIT REQUEST IN HEX

The BX command returns the states of the general purpose I/O bits in a hex format. The rightmost character represents the least-significant-nibble (4 bits) and, if the nibble is rewritten as bits, the rightmost bit is the least-significant bit. An input set low will be represented as a binary 0 and a high as binary 1, similarly for an output.



Example: Assuming the default I/O bit direction of [IO](#) = 00, bits 0-7 inputs
User input lines 0 and 2 are high and the remaining 6 inputs are low.
Use the BX commands to verify these states.

Enter: BX

Response: 05<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| BX | AX - AT | Immediate | |
| BX | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [BH](#), [BL](#), [BS](#)

CA CLEAR AXIS DONE FLAG

The CA command operates like the [IC](#) command, except it clears the done flag of the addressed axis only. In multi-axis modes, the CA command clears the flags of all selected axes. Unlike the [IC](#) command, CA will not clear other error flags in the status register such as slip and limit.



Example: After a multi-axis move, clear the Z axis done flag only.

Enter: [AA](#);
[MR](#)1000,2000,3000,4000;
[GO](#);
[ID](#)
[AZ](#);
 CA;

Response: None.



Example: After a multi-axis move, clear the Y and Z axis done flags only.

Enter: [AA](#);
[MR](#)1000,2000,3000,4000;
[GO](#);
[ID](#);
 CA,1,1;

Response: None.

NOTE: In [AA](#) or [AM](#) mode, a null value in the argument list specifies the done bit of that axis is not to be cleared.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| CA; | AX - AT | Immediate | |
| CAb,b,b,b; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [GD](#), [IC](#), [ID](#), [IL](#), [IN](#), [IP](#)

CB CLEAR MACRO LINKS

This command clears all macro links of input bits to macro executions and/or the KILL ([KL](#)) function.



Example: Clear all previously defined macro links to input bits.

Enter: CB;

Response: None

CD CONTOUR DEFINE



The CD command enters contour definition mode and defines a constant velocity contour. The only way to exit this mode is to issue a [CE](#) or [CK](#) command. Commands following the CD command must be in multi-axis format and be commands valid in CD mode. All multi-axis commands entered in CD mode can address only those axes assigned in the CD command. No commands entered will be executed until a [CX](#) command is received which must be issued outside of CD mode.

The CD command takes up to eight parameters. Each parameter specifies a starting point for each axis to be involved in the contour in absolute coordinates. If an axis is not to be involved in the contour, its parameter position should be skipped just as it would be in any other multi-axis command. Commands issued within CD mode must be formatted to include only those axes indicated in the CD command. Those that are not in the CD command simply do not exist in the formatting of commands entered in CD mode. For example, if a CD command is issued that uses the X, Y, and T axes (such as CD100,300,,400;), commands entered within CD mode will only consider the X, Y, and T axes. An [MT](#) command that moves X to 200, Y to 600, and T to 200 would take the form: [MT](#)200,600,200;. Note the lack of a placeholder comma for the Z axis.

Contours that will include circular interpolation ([CR](#)) must be defined for only 2 axes in the CD command. Contours involving more than 2 axes may not use the [CR](#) command. The size of the contour definition buffer for the UMX is 32,763 positions.

When the contour is executed, the UMX will use the distance between the current position and the contour starting point to linearly ramp up each axis such that all involved axes reach a combined, vectored velocity equal to the value set with the [CV](#) command. If this distance is zero, no ramp will be generated resulting in an instantaneous jump to contour velocity. Most stepper systems cannot achieve this and servos will tend to oscillate wildly before settling down if at all. Care should be taken to allow sufficient ramping distance between the contour starting position and the current position when the [CX](#) command is issued.

Once the contour is completely executed, the UMX will ramp the axes to a stop using the rate defined with the [AC](#) command. This ramp down will take each axis beyond the final point of the contour. Without manually calculating the ramp down distance for each axis, there is no way to force the contour to come to a complete stop at a predetermined point.



Example:

The following demonstrates cutting a hole with a 10,000 count radius using constant velocity contouring and circular interpolation. The contouring velocity is set to 1000 pulses per second. A contour is then defined beginning at coordinates 0,0 on the X and Y axes. The auxiliary output of the Z axis is turned on, which could turn on the cutting torch or laser starting the cut at the center of the circle. A half circle is cut from the center to the outside of the hole, positioning the cutting tool at the start of the desired hole. The hole is then cut, the torch turned off, the stage stopped and the definition is complete. The stage is then positioned and the hole cut with the [CX](#) command. The [AN](#) and [AF](#) commands must have commas for all axes since they can all be addressed from within the contour definition.

Enter:

```
AA;  
CV1000;  
CD0,0;
```

```

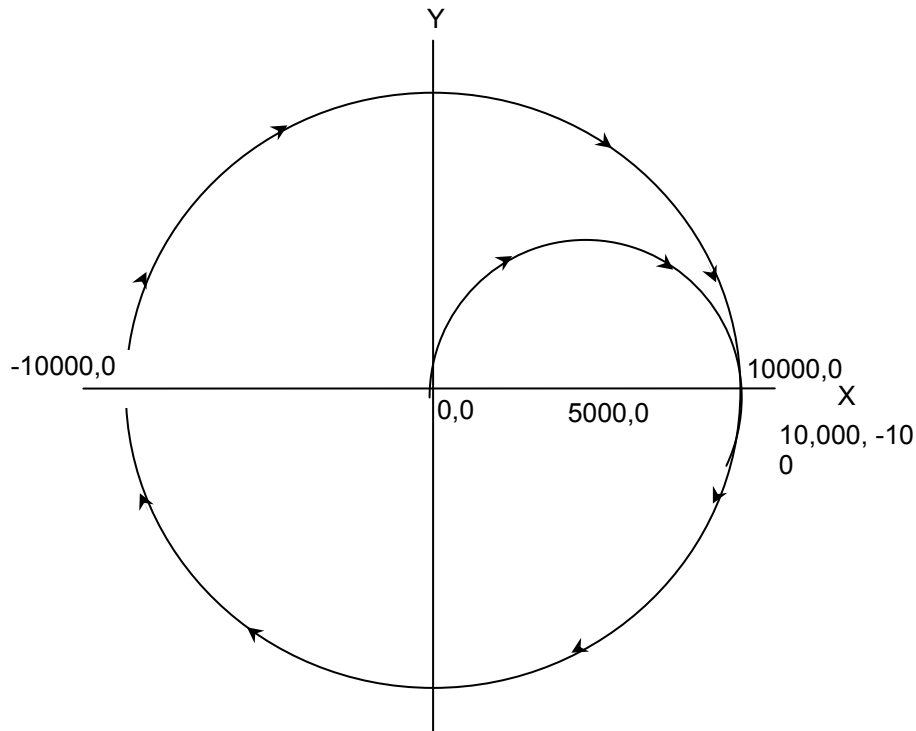
AN,,1;
CR5000,0,-3.1415926;
CR0,0,-6.2831853;
AF,,0;
MT10000,-1000;
CE;
MT10000,0;
GO;
CX;

```

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------------------------------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX - AT | Not Valid | | |
| CD#,#,#,#; | AA-AM | 5 + number of axes in the contour | | |
| - | AA/CD | N/A | | |

Related commands: [AF](#), [AN](#), [BH](#), [BL](#), [CE](#), [CK](#), [CR](#), [CV](#), [CX](#), [NV](#), [MT](#)



CE CONTOUR END

The CE command marks the end of the contour sequence. It will terminate the [CD](#) mode and, when executed, ramp to a stop and exit to the [AA](#) command mode. The end of the contour should contain at least a short linear segment just prior to the CE command to initialize the parameters for the deceleration of the stage.



Example: (see [CD](#) command on page 5-44)

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX – AT | Not Valid | | |
| - | AA-AM | Not Valid | | |
| CE; | AA/CD | N/A | N/A | 2 |

Related commands: [CD](#), [CK](#)

CG CONTOUR PRIORITY



The CG command is a form of the [CX](#) command, use if [CE](#) is followed by a [MT](#) command. When CG is used to execute the contour, the following [MT](#) commands will be on hold until the contour execution is complete. After the contour has been executed, the [MT](#) commands that follow can be parsed.

CG is preferred over the [CX](#) when the [MT](#) commands are issued after a contour is executed. The CG command ensures that the [MT](#) command that follows starts from a known position. This makes for a more accurate calculation of the [MT](#) move.



Example: See page 5-44 ([CD](#)), Make sure that the hole is completely cut before executing the move to the new command position.

Enter:

```

AA;
CV1000;
CD0,0;
AN,,1;
CR5000,0,-3.1415926;
CR0,0,6.2831853;
AF,,0;
MT10000,-1000;
CE;
MT-10000,0;
GO;
CG;

```

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX – AT | Not Valid | |
| CG; | AA-AM | 7* | 2* |
| CG; | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is selected add 2 to the command queue.
- If an aux bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.
- If the axis is stepper and encoder or servo add 1 to the command queue.
- Add the following queue requirements for the ramp types listed.

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

Related commands: [AF](#), [AN](#), [BH](#), [BL](#), [CD](#), [CK](#), [CR](#), [CV](#), [CX](#), [MT](#), [NV](#)

CK CONTOUR END AND KILL

The CK command will end the contour sequence, like the [CE](#) command, except there is no ramp down; i.e. motion will stop abruptly. It is used in place of the [CE](#) command.

NOTE: This command should be used with caution to prevent the stage from slipping or losing its correct position.



Example: Same scenario as [CD](#) command, but we want to end the contour with the minimum ramp down.

Enter: [AA](#);
[CV](#)1000;
[CD](#)0,0;
[AN](#),,;
[CR](#)5000,0,-3.1415926;
[CR](#)0,0,-6.2831853;
[AE](#),,0;
[MT](#)10000,-1000;
 CK;
[MT](#)-1000,0;
[GO](#);
[CX](#)

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX – AT | Not Valid | | |
| - | AA-AM | Not Valid | | |
| CK; | AA/CD | N/A | N/A | 2 |

Related commands: [CD](#), [CE](#)

CN COSINE ON

The CN command enables cosine velocity ramps; i.e. half sinusoid acceleration profiles, for all axes. The cosine profile is not truncated in moves cannot reach full velocity, but instead the velocity is reduced sufficiently to preserve the cosine profile. This command should not be given while an axis is in motion or the results may be unpredictable. This command affects all axes even if issued in the single axis mode. The [PF](#) command is used to return to linear motion profiles. See the [AP](#) command, page 5-33, to preserve the CN setting as the Power up/Reset ramp.



Example: Set the board to be in cosine mode.

Enter: CN;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| CN; | AX - AT | 5 | 24 |
| CN; | AA-AM | 5 | 24 |
| - | AA/CD | Not Valid | |

Related commands: [LA](#), [PF](#), [PN](#), [PR](#), [?RT](#), [SC](#), [SR](#)

CR CIRCULAR INTERPOLATION

The CR command defines a move in a circular pattern from the entry position. The first two parameters are the center of the circle in absolute units and the third parameter is the distance to move in radians. Positive radians equal counterclockwise movement. Negative radians equal counter clockwise movement. The radius of the circle is the linear distance between the current position and the first two parameters of the CR command.

The CR command generates a circle by breaking the circle into a series of linear segments. The number of segments will be equal to the total distance traveled divided by the contour velocity divided by the update rate. If finer resolution is required, the [MT](#) command may be used to produce smaller segments but the segments must be calculated by the user

RANGE:

Min Pos. Value ≤ Parameter 1 (First Coordinate for Center of Circle) ≤ Max Pos. Value
Min Pos. Value ≤ Parameter 2 (Second Coordinate for Center of Circle) ≤ Max Pos. Value

Note: The minimum and maximum position value depends on the settings used on UMX, see the specifications for more detail.



Example: (see [CD](#) command on page 5-44)

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX - AT | Not Valid | | |
| - | AA-AM | Not Valid | | |
| CR#,#, #; | AA/CD | N/A | N/A | 8 |

Related commands: [CD](#), [MT](#)

CV CONTOUR VELOCITY

The CV command allows specification of a contouring velocity. It is executed from the [AA](#) mode before a contour definition. A contour defined by a [CD](#) command cannot be executed if followed by a CV command. Changing this parameter will make any previously defined contours invalid. The contour velocity defaults to 1000 at power up or reset. Use [WQ](#) between contour definitions to avoid having a CV associated with a second contour definition affect a prior contour still in motion. A CV cannot be issued between a [CD](#) and [CE](#) command. To change the contour velocity within a contour definition, use the [NV](#) command.

RANGE: $1 \leq CV \leq 4,000,000$



Example: (see [CD](#) command on page 5-44)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Not Valid | |
| CV#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [CD](#), [NV](#)

CW CLEAR WHILE

The CW command breaks the [WH](#) loop upon execution of the remaining commands in the loop; i.e. the current execution of the loop is finished. The [WH](#) loop is always executed at least one time since the test for the flag is at the bottom.



Example: (see [WH](#) command page 5-204)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| CW; | AX - AT | Immediate | |
| CW; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [WG](#), [WH](#)

CX CONTOUR EXECUTE



The CX command will execute the previously defined contour sequence. The stage must be positioned such that it can accelerate to speed by the absolute position specified by the [CD](#) command it is executing and must be traveling in the proper direction. Once a contour is defined it may be executed at any time by executing a CX command until it is replaced by another contour definition. The CX command cannot be placed within a loop or while construct.



Example: (see [CD](#) command on page 5-44)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX – AT | Not Valid | |
| CX; | AA-AM | 6* | 2* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo add 1 to the command queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

Related commands: [CD](#), [CE](#), [CK](#), [ID](#)

?DA PRINT A CUSTOM RAMP

This command will report the entries of a previously defined custom ramp table.

RANGE: $1 \leq \text{Ramp Table Numbers} \leq 8$



Example: Print out custom ramp table #2

Enter: ?DA2;

Response: [DAR](#)2<LF>
 [DAB](#)0.10000000,0.20000000<LF>
 [DAB](#)90000,0.80000000<LF>
 [DAB](#)10000,1.00000000<LF>
 [DAE](#)<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?DA#; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [DAB](#), [DAE](#), [DAR](#), [?DE](#), [?DS](#)

DAB DEFINE CUSTOM RAMP BREAKPOINT



The DAB command sets a breakpoint in a custom ramp table. This is the only command that should be used after [DAR](#) and before [DAE](#). Each custom ramp may contain up to 25 breakpoints, each defined by a DAB command.

The DAB command takes two parameters; the first specifies the acceleration level that should be used to achieve the second parameter, velocity level. Both levels are expressed in terms of percentage in decimal format; i.e. 1.00 is 100%. At no time should a DAB command be entered in which the velocity parameter is less than the velocity parameter of the prior DAB. The UMX will not flag this as a command error but the results of such a ramp will be unpredictable. Each DAB command sent should be equal to or greater than the DAB command that preceded it. It is the user's responsibility to make sure this command is used properly.

RANGE:

$0.0000000 \leq \text{Parameter 1} \leq 1.00000000$

$0.0000000 \leq \text{Parameter 2} \leq 1.00000000$



Example: See the [DAR](#) command (page 5-56) for a complete example of a custom profile.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DAB#,#; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?DA](#), [DAR](#), [DAE](#), [?DS](#), [?DE](#), [SR](#)

DAE END CUSTOM RAMP DEFINITION



The DAE command terminates a custom ramp table definition initiated by the [DAR](#) command.



Example: See the [DAR](#) command (page 5-56) for a complete custom ramp table definition.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DAE; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?DA](#), [DAR](#), [DAB](#), [?DS](#), [?DE](#), [SR](#)

DAR BEGIN CUSTOM RAMP DEFINITION



The DAR command starts the definition of a custom ramp table. A parameter supplied with this command, from 1 to 8, specifies which ramp table to create. If a ramp table by that number has already been defined, it will be overwritten.

Once the DAR command has been issued, only the [DAB](#) and [DAE](#) commands will be valid. A series of ramp table breakpoints may be entered using the [DAB](#) command which define the profile breakpoints for this ramp table. Up to 25 breakpoints may be defined but a smaller number may be used. A ramp table containing no breakpoints is invalid and will result unpredictably if used.

RANGE: $1 \leq \text{DAR} \leq 8$



Example: Create a ramp table definition resembling a jerk-limited linear profile.

Enter:

| | |
|-----------------------------|--|
| DAR3; | * Store as table #3 |
| DAB .1,.05; | * Ramp at 10% of AC until 5% of VL |
| DAB .3,.1; | * Ramp at 30% of AC until 10% of VL |
| DAB .9,.9; | * Ramp at 90% of AC until 90% of VL |
| DAB .3,.95; | * Ramp at 30% of AC until 95% of VL |
| DAB .1,1; | * Ramp at 10% of AC until 100% of VL |
| DAE ; | * End table definition |

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DAR#; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?DA](#), [DAB](#), [DAE](#), [?DE](#), [?DS](#), [SR](#)

?DB**REPORT DIRECTION BIT LOGIC**

The ?DB command reports the direction bit logic: inverted or normal. If the direction bit is low when moving positive, this command will return 'normal'. If the direction bit has been inverted, this command will return inverted.



Example: Report whether the direction bit for the T axis is low or high when making positive moves

Enter: [AT](#);
?DB

Response: dbi<LF> (The inverted result indicates the T axis direction bit is high for positive moves)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?DB | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [DBI](#), [DBN](#), [?SV](#)

DBI INVERT DIRECTION BIT

The DBI command inverts the logic of the direction control output of the addressed axis or axes. By default, the direction output of an axis is a TTL low when traveling in the positive direction and high when traveling negative. After using the DBI command, the direction bit will be high when traveling positive and low when traveling negative. This is useful for inverting the logical direction of a motor when the encoder counts opposite the motor direction. This command can be canceled using the [DBN](#) command. To make this the default at power up or reset, use the [AP](#) command.



Example: Set the direction outputs for axes Z and T to output high when traveling positive and low when traveling negative. Leaves X and Y as they are.

Enter: [AZ](#);
 DBI;
 [AT](#);
 DBI;
 Or
 [AA](#);
 DBI,,1,1;

Response: None.

Note: In [AA](#) or [AM](#) mode a null value in the argument list specifies that the sense of that direction bit is not to be changed.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DBI; | AX - AT | 1 | 0 |
| DBIb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [?DB](#), [DBN](#), [SVI](#), [SVN](#), [UN](#)

DBN**NORMALIZE DIRECTION BIT**

The DBN command normalizes the logic of the direction control output of the addressed axis or axes, returning their output logic to default; i.e. TTL low when traveling in the positive direction and high when traveling negative. This command negates the effect of the [DBI](#) command. To make this the default at power up or reset when [DBI](#) has already been made the default, use the [AP](#) command.



Example: Set the direction outputs for axes Z and T to default output logic; i.e. output high when traveling positive and low when traveling negative. Leave X and Y as they are.

Enter: [AZ](#);
DBN;
[AT](#);
DBN;
Or
[AA](#);
DBN,,1,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DBN; | AX - AT | 1 | 0 |
| DBNb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [?DB](#), [DBI](#), [SVI](#), [SVN](#), [UN](#)

DC**DECELERATION**

The DC command sets a deceleration rate overriding the [AC](#) parameter when the [GU](#) command is used to initiate a move. Only the [GU](#) command will use the DC value. The deceleration rate defaults to 200,000 and will take on whatever value is entered via the [AC](#) command. Therefore, the DC command must be reentered after using [AC](#) if a different deceleration rate is desired.

RANGE: $1 \leq DC \leq 1044000$

Note: The range denotes values after User Units have been applied. (See [UU](#) command.)



Example: Send the Y axis on a 100,000 count move that accelerates at 100,000 counts per second per second up to 50,000 counts per second and decelerates at 20,000 counts per second per second.

Enter: [AY](#);
[AC](#)100000;
 DC20000;
[VL](#)50000;
[MR](#)100000;
[GU](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| DC#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [AC](#), [GU](#), [RC](#), [VB](#), [VL](#)

?DE REPORT A CUSTOM RAMP TABLE ENTRY



The ?DE command will return a specific entry from a specific custom ramp table. The first parameter specifies the table to examine and the second parameter specifies the entry to return from the table.

RANGE:

$1 \leq \text{Parameter1} \leq 8$

$1 \leq \text{Parameter2} \leq 25$



Example: We can't remember what the 23rd breakpoint in table 4 was set to. Use the ?DE command to find out.

Enter: ?DE4,23;

Response: <LF> (there is no 23rd entry in table 4)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?DE#,#; | AX - AT | Immediate | |
| - | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [?DA](#), [DAB](#), [DAE](#), [DAR](#), [?DS](#)

?DS REPORT THE SIZE OF A CUSTOM RAMP TABLE



The ?DS command returns the number of segments in the specified custom ramp table.

RANGE: $1 \leq ?DS \leq 8$



Example: The 3rd custom ramp should be 17 breakpoints long. Make sure this is true.

Enter: ?DS3;

Response: 17<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?DS#; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?DA](#), [DAB](#), [DAE](#), [DAR](#), [?DE](#)

DZ DEFINE ZERO POSITION IN OPEN-LOOP MODE



The DZ command defines the offset coefficient needed to produce a zero or stationary position by the servo motor. This command is used in the open-loop mode, hold off ([HF](#)). The factory default value is zero. See the [AP](#) Command on page 5-33 to preserve the DZ settings as the Power up/Reset values.

Power up/Reset values.

Value range: -32640 to +32640



Example: Define the offset coefficient to be 250 for the X axis.

Enter: [AX](#);
DZ250;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AY – AT | 2 | 2 | Not valid |
| AA-AM | Not Valid | | |
| AA/CD | Not Valid | | |

Related commands: [?DZ](#), [?KO](#)

?DZ REPORT DAC OPEN-LOOP OFFSET



The ?DZ command reports the current setting of the [DZ](#) command in [DZ](#) command format.



Example: The closed-loop offset needs to be set the same as the open-loop offset. We've already set the open-loop offset but forgot what value we used. Send the ?DZ command to find out.

Enter: ?DZ;

Response: dz28<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|-----|
| FORMAT | MODE | Min | Max |
| ?DZ | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [DZ](#), [?KO](#)

EA ENCODER STATUS

The EA command returns encoder status of the currently addressed axis or axes in the following format:

| EA COMMAND RESPONSE DESCRIPTION | | |
|---------------------------------|------|---|
| CHAR | SENT | DESCRIPTION |
| 1 | LF | Line feed |
| 2 | CR | Carriage return |
| 3 | CR | Carriage return |
| 4 | E | Slip detection enabled |
| | D | Slip detection disabled |
| 5 | E | Position maintenance enabled |
| | D | Position maintenance disabled |
| 6 | S | Slip or stall detected (reset by execution of EA command) |
| | N | No slip or stall detected |
| 7 | P | Position Maintenance within deadband |
| | N | Position not within deadband |
| 8 | H | Axis is home |
| | N | Axis is not home |
| 9 | N | Unused/reserved |
| 10 | LF | Line feed |
| 11 | CR | Carriage return |
| 12 | CR | Carriage return |



Example: Examine the status of the Y axis encoder.

Enter: [AY](#);
EA

Response: DDNNNN<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| EA | AX - AT | Immediate | |
| EA | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [QA](#), [QI](#), [RA](#), [RI](#)

EF**ECHO OFF**

The EF command disables echoing from the UMX motion system. This is the default mode at power up or reset.



Example: Stop echoing to the host.

Enter: EF;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AY – AT | Immediate | | |
| AA-AM | Immediate | | |
| AA/CD | Not Valid | | |

Related commands: [EN](#)

EH DEFINING ENCODER HOME

[HH](#) and [HL](#) commands are active in [HE](#) mode. This allows for an encoder home operation with the Home Switch being active HIGH. Previous OMS controllers required an active LOW Home Switch to do an encoder home operation, and this will be the "default" setting here as well. This command sets the true states of the A, B, and Index encoder signals and the true state of the Home Switch for the Encoder Home condition.

The new EHbbbb command provides variants that will completely determine the logic states of encoder signals A, B, Index, and Home Switch that make for a true encoder home event. This allows the user more flexibility to adapt to the schemes of most encoder manufacturers. All bbbb are ones (1) or zeros (0) representing the states of the Home Switch, Index, Phase B and Phase A, respectively, or X to indicate a "don't care" condition.

Format: [AX-AT](#): EHbbbb

All b's are 0, 1, or X representing the states of Home Switch, Index, Phase B, and Phase A respectively. A 1 represents a high true state, a 0 represents a low true state, and an X represents a "don't care" state.



Example: [AX-AT](#): for the currently selected axis, the EH command: sets true states as follows:

```
EH1000;    HomeHigh/Index low/PhaseB low/PhaseA low
EH0100;    HomeLow/Index high/PhaseB low/PhaseA low
EH0010;    HomeLow/Index low/PhaseB high/PhaseA low
EH0001;    HomeLow/Index low/PhaseB low/PhaseA high
EH01XX;    HomeLow/Index high/PhaseB "don't care"/PhaseA "don't care"
```

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| EHbbbb; | AX – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?EH](#)

?EH**QUERY ENCODER HOME**

The ?EH reports the state of the Home Switch, Index, Phase A and Phase B true states in both single axis and multi-axis modes. It is an immediate command and does not require any queue space.



Example: ?EH [AX](#)- [AT](#) Single axis mode

The controller's response to the ?EH command has the following meaning:

| | |
|--------|--|
| eh1000 | HomeHigh/Index low/PhaseB low/PhaseA low |
| eh0100 | HomeLow/Index high/PhaseB low/PhaseA low |
| eh0010 | HomeLow/Index low/PhaseB high/PhaseA low |
| eh0001 | HomeLow/Index low/PhaseB low/PhaseA high |
| eh01XX | HomeLow/Index high/PhaseB "don't care"/PhaseA "don't care" |

NOTE: The encoder home pattern set by the [EH](#) command and the true logic state of the home switch as set by the [HL](#) or [HH](#) command are archived in the flash with the parameter archive commands.

The factory default encoder home pattern and home switch state would be the same as the current OMS standard home pattern, that is, Encoder Home pattern is Index HIGH, Phase A HIGH, Phase B LOW and the Home Switch is active LOW.

EN ECHO ON

The EN command enables echoing. All commands and parameters will be echoed to the host. This mode is useful for debugging command strings from a terminal. This mode also outputs an English readable error message to the host which may be echoed to the terminal or computer to aid in debugging.



Example: Enable echoing by the UMX so that commands are echoed and the error message is returned to the host as a readable ASCII string. This command would probably be the first command executed after turning on the system when this mode is desired.

Enter: EN;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | Immediate | | |
| AA - AM | Immediate | | |
| AA/CD | Not Valid | | |

Related commands: [EF](#)

ER ENCODER RATIO

The ER command allows specification of encoder:motor ratio for position maintenance mode. This command is not designed for use with servo motors. ER takes two arguments: encoder counts and motor counts. Both parameters must be integers unless user units ([UU](#)) are enabled. The ratio need not be per full revolution; reduce the fraction as far as possible and use those values.

The factory default ratio is 1:1. See the [AP](#) command, page 5-33, to preserve the ER settings as the Power up/Reset values.

NOTE: That if an encoder ratio has been defined by the ER command, then the slip tolerance defined by the [ES](#) command is defined in encoder units and not in motor units.

Parameter 1 = Encoder Count**Parameter 2 = Motor Counts**

Example: You have an encoder connected to a stepper motor through a series of gears. When the motor steps 25,000 times, the encoder produces 10,000 counts. Set up an encoder ratio so hold mode will work correctly.

Enter: ER10000,25000;
or
ER2,5;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ER#,#; | AX - AT | 1 | 0 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?ER](#)

?ER REPORT MOTOR:ENCODER RATIO



The ?ER command reports the motor-to-encoder ratio as set with the [ER](#) command.



Example: Find out what the last [ER](#) command sent was.

Enter: ?ER

Response: er2.00000000<LF>
(The encoder produces 1 count for every 2 steps of the motor.)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?ER | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ER](#)

ES ENCODER SLIP TOLERANCE

The ES command parameter specifies the slip detection tolerance before slip or stall is flagged in the status register and in the [RL](#) command response. The encoder may get off target by as much as this value before the UMX will consider the axis slipped. This mode must be turned on with an [IS](#) command and off with an [HF](#) command. The factory default value is 1. For servo motors, the default is 32,767. This value determines when the PID is disabled due to excessive following error. See [AP](#) command (see page 5-33) to preserve the ES settings as the Power up/Reset values.

NOTE: If an encoder ratio is defined with the [ER](#) command, then the slip tolerance defined by the ES command is in encoder units and not in motor units.

NOTE: The [GS](#) command allows an open-loop stepper axis to perform a limited form of slip detection by using the home switch as a reference when executing a move command.

STEPPER RANGE: $0 \leq ES \leq 65,535$

SERVO RANGE: $0 \leq ES \leq 327,670$



Example: Your application can tolerate being up to 5 steps from the desired position before the controlling program should be notified of a slip condition.

Enter: ES5;
[IS](#);

Response: None.

NOTE: That if an encoder ratio has been defined by the [ER](#) command, then the slip tolerance defined by the ES command is defined in encoder units and not in motor units.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ES#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?ES](#), [IS](#), [RL](#), [TF](#), [TN](#)

?ES REPORT ENCODER SLIP TOLERANCE



The ?ES command reports the current value of the slip detect tolerance assigned to an axis.



Example: Report the current value for encoder slip detection tolerance

Enter: ?ES

Response: es15<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?ES | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ES](#)

ET ENCODER TRACKING

The ET command turns on the encoder tracking mode. The axis will track its encoder input, thus allowing one axis to follow the activity of another or a thumbwheel for manual positioning or the movement of another device that produces a signal compatible to the encoder inputs. No acceleration or deceleration ramps are generated. The axis will duplicate the encoder input. The [ER](#) command allows the user to scale the motor's movements relative to the encoder. This command is intended to be used with stepper motors with encoders and not with servo motors.



Example: Set up the Y axis so it will follow its encoder input.

Enter: [AY](#);
 ET;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ET; | AY – AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ER](#), [HF](#)

FL**FLUSH**

The FL command will flush an individual axis' queues. This command is similar in operation to the [KL](#) and [ST](#) commands except that current motion will remain unaffected by the FL command. All unexecuted commands remaining in the current axis queue will be flushed upon receipt of this command.



Example: Several motion commands have been sent to the X axis but a situation arose and now those commands must be cleared out. The currently executing motion must be allowed to complete to avoid damage to the product.

Enter: [AX](#);
FL;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| FL; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [KL](#), [KS](#), [SA](#), [ST](#)

FP**FORCE POSITION**

The FP command will flush the command queue and attempt to stop at the specified position. The axis will overshoot if there is insufficient distance left to stop at the programmed acceleration. This command should not be given to a servo axis while it is in motion; the results may be unpredictable.



Example: Force axis to stop at 25,000.

Enter: FP25000;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| FP#; | AX - AT | 2* | 1* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [MM](#), [MP](#), [MV](#), [SP](#)

FX**ENABLE AXIS GANTRY MODE**

The FX command pairs the current axis with the X axis to form a gantry crane. (i.e. the axis follows the X axis.)



Example: Pair the X and Y axis to form a gantry crane.

Enter: [AY](#);
FX;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| FX; | AY – AT | 1 | 0 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: None

GD**GO AND RESET DONE**

The GD command may be substituted for a [GO](#) command. It will reset the done flags and then initiate the move which has been previously programmed with such commands as [MA](#), [MR](#), [MT](#), and [ML](#) just as the [GO](#) command does. In single axis mode, only the done flag for the selected axis will be reset.

In [AA](#) mode, all the done flags will be reset. In the [AM](#) mode, only the axes involved in the move will be reset. This allows the host to reset the interrupts on the axis involved in the next move without affecting other axes which may be still active. Note that this command is probably only useful in applications where commands are queued in advance since the interrupt may be reset before the host has the opportunity to service it if the GD command is waiting in the queue.

If this command is issued without having defined a move, the results are undefined. Issuing a GD command to execute an already-executed move also has undefined results. Only one GD command should be issued per defined move.



Example: In the single axis mode, move the Y axis 12345 counts in the negative direction and set the done flag when the move is completed. Then clear the done flag, move the motor 12345 counts in the positive direction, and set the done flag again when the move is completed.

Enter: [AY](#);
[MR](#)-12345;
[GO](#);
[ID](#);
[MR](#)12345;
 GD;
[ID](#);

Response: None.



Example: In [AA](#) mode, perform a linear absolute move with the X and Y axes to the position 10000, 20000 and set the done flag when the move is completed. Then clear the done flag, perform a linear relative move on both axes moving the X axis 10000 steps in the negative direction and the Y axis 20000 steps in the negative direction, and set the flag once again.

Enter: [AA](#);
[MT](#)10000,20000;
[GO](#);
[ID](#);
[ML](#)-10000,-20000;
 GD;
[ID](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| GD; | AX - AT | 6* | 0* |
| GD; | AA-AM | 8* | 0* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [GO](#), [GN](#), [GS](#), [GU](#), [MA](#), [ML](#), [MR](#), [MT](#)

GN**GO AND NOTIFY WHEN DONE**

The GN command will initiate a move which has been previously programmed with such commands as [MA](#), [MR](#), [MT](#) and [ML](#) and set the axis done flags when the move is complete. No operand is required with the GN command. If this command is issued without having defined a move, the results are undefined. Issuing a GN command to execute an already-executed move also has undefined results. Only one GN command should be issued per defined move.



Example: In the single axis mode, move the X axis to absolute position 12345.

Enter: [AX](#);
[MA](#)12345;
 GN;

Response: None



Example: In the AA mode, move the X axis 2468 steps in the positive direction and the Y axis 2468 steps in the negative direction.

Enter: [AA](#);
[MR](#)2468,-2468;
 GN;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| GN; | AX - AT | 6* | 0* |
| GN; | AA-AM | 7* | 0* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [GD](#), [GN](#), [GS](#), [GU](#), [MA](#), [ML](#), [MR](#), [MT](#)

GO**GO**

The GO command will initiate a move which has been previously programmed with such commands as [MA](#), [MR](#), [MT](#), and [ML](#). No operand is required with the GO command. If this command is issued without having defined a move, the results are undefined. Issuing a GO command to execute an already-executed move also has undefined results. Only one GO command should be issued per defined move.



Example: In the single axis mode, move the X axis to absolute position 12345.

Enter: [AX](#);
[MA](#)12345;
 GO;

Response: None



Example: In the [AA](#) mode, move the X axis 2468 steps in the positive direction and the Y axis 2468 steps in the negative direction.

Enter: [AA](#);
[MR](#)2468,-2468;
 GO;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| GO; | AX - AT | 5* | 0* |
| GO; | AA-AM | 7* | 0* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [GD](#), [GN](#), [GS](#), [GU](#), [MA](#), [ML](#), [MR](#), [MT](#)

GS**GO AND MONITOR SLIP
TRIGGER**

The GS command works exactly like the [GO](#) command except that the home switch will be monitored during the motion. If the home switch becomes active the slip flag will be set for the axis. The host application can read the slip flag and see that the home switch was encountered during the move. This is useful in applications that register slip conditions by means other than encoder position verification; in fact, this command is not valid in controls with encoder feedback which includes servo motors.

If this command is issued without having defined a move, the results are undefined. Issuing a [GD](#) command to execute an already-executed move also has undefined results. Only one [GD](#) command should be issued per defined move.



Example: Move the X axis 50,000 counts in the positive direction. If the motor slips it will close a switch wired to the home input of the X axis. Monitor this switch during the move and set the slip flag for axis X if the switch becomes active.

Enter: [AX](#);
[MR](#)50000;
[GS](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| GS; | AX - AT | 5* | 0* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [GD](#), [GN](#), [GO](#), [GU](#), [MA](#), [MR](#)

GU**GO ASYMMETRICAL**

The GU command initiates a previously defined move using the [AC](#) value for acceleration and the [DC](#) value for deceleration. This command may be used with only one axis at a time; i.e. it is not valid with the [ML](#) and [MT](#) commands.

If this command is issued without having defined a move, the results are undefined. Issuing a GU command to execute an already-executed move also has undefined results. Only one GU command should be issued per defined move. GU is used only with linear acceleration ramps



Example: Move the Y axis to position 1,500 using the current acceleration and velocity and a deceleration of 5,000 counts per second per second.

Enter: [AY](#);
[DC](#)5000;
[MA](#)1500;
 GU;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| GU; | AX - AT | 3* | 0* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [AC](#), [DC](#), [GD](#), [GN](#), [GO](#), [GS](#), [MA](#), [MR](#)

HD HOLD DEADBAND

The HD command specifies dead band counts for position maintenance mode. If the encoder count is within this distance of target, it is considered in position and no further correction will be made. This parameter interacts with the [HG](#) and [HV](#) commands; i.e. a larger dead band will allow a larger gain parameter in many applications. This command is designed to work with stepper motor applications using encoders and is not designed for use with servo motors. The factory default value is zero. See the [AP](#) command to preserve the HD settings as the Power up/Reset values.

RANGE: $0 \leq HD \leq 64,000$



Example: (see [HN](#) command page 5-92)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HD#; | AX - AT | 1 | 1 |
| HD#,#,#,#; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [?HD](#), [HE](#), [HG](#), [HN](#), [HV](#)

?HD REPORT POSITION MAINTENANCE DEADBAND

The ?HD command reports the current setting of the [HD](#) command. This command will only work on stepper axes with encoders.



Example: Find out what [HD](#) was last set to.

Enter: ?HD

Response: hd5<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?HD | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HD](#), [?HG](#), [?HV](#)

HE HOME ENCODER



The HE command enables encoder index, and phase A and B signals to be considered in addition to the home switch input when an [HM](#) or [HR](#) command is executed. This command does not execute the home motion which must be initiated with an [HM](#), [HR](#), [KM](#), or [KR](#) command. Default home behavior (considering only the home switch for home commands) may be reestablished via the [HS](#) command. In the HE mode, home is defined as the logical AND of the encoder index, the external home enable and the encoder quadrant where channel A is positive and channel B is negative. The default home logic expressed in Boolean terms is:

$$\text{Home} = \text{Phase_A} \times / \text{Phase_B} * \text{Index} \times / \text{Home_Switch}$$

The default logic can be changed to any required combination with the use of [HL](#) and [HH](#) commands to set the true home switch state and the [EH](#) command to set the encoder signal (Index, PhaseA and PhaseB) states required for a true home condition.



Example: Set up the Y axis so it will use the encoder signals to recognize the home position.

Enter: [AY](#);
HE;
or
[AA](#);
HE,1;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HE; | AX - AT | Immediate | |
| HEb,b,b,b; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [EH](#), [HH](#), [HL](#), [HM](#), [HR](#), [HS](#), [KM](#), [KR](#)

HF**HOLD OFF**

The HF command disables position hold, stall detection and tracking modes, for stepper with an encoder axis. If the current axis is a servo, this command will open the loop and turn off the PID. If the current mode is multi-axis, all axes will go into open-loop mode. This is the default mode at power up or reset.



Example: Turn off encoder hold mode on the X axis.

Enter: [AX](#);
HF;

Response: None

NOTE: In [AA](#) or [AM](#) mode, a null value in the argument list specifies feedback for that axis is not to be disabled.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HF; | AX - AT | 2* | 0* |
| HF; | AA-AM | 2* | 0* |
| - | AA/CD | Not Valid | |

*Values in table are for a stepper with encoder axis. For a servo axis the command queue requires 1, and the argument queue requires 1.

Related commands: [HN](#), [?PM](#)

HG HOLD GAIN

The HG command allows the user to specify the position hold gain parameter. This gain parameter is multiplied by the position error in determining the velocity during correction. The velocity used will not exceed the value set with the hold velocity ([HV](#)) command. This command is designed to work with stepper motor applications using encoders and is not designed for use with servo motors. The parameter should be set experimentally by increasing it until the system is unstable then reducing it slightly below the threshold of stability. The factory default value is 1. See the [AP](#) command, page 5-33, to preserve the HG settings as the Power up/Reset values.

RANGE: $1 \leq HG \leq 32,000$



Example: (see [HN](#) command (page 5-92))

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HG#; | AX - AT | 1 | 1 |
| HG#,#,#,; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [HD](#), [HE](#), [?HG](#), [HN](#), [HV](#)

?HG REPORT POSITION MAINTENANCE GAIN

The ?HG command reports the current setting of the position maintenance hold gain constant for the current axis. This command works only with stepper + encoder axes.



Example: Position corrections seem slow. Check the setting of [HG](#) to be sure it is correct.

Enter: ?HG

Response: hg100<LF >

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?HG | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?HD](#), [HG](#), [?HV](#)

HH HOME HIGH

The HH command sets the sense of the home switch on the current axis to active TTL high. This command allows TTL logic high to be treated as the “true” state for applications where this is more convenient. Once this command has been sent to the UMX, active high homes can be made the power-up default by using the [AP](#) command; refer to the [AP](#) command for details. This command sets the home switch “true” state to TTL logic high for both the [HS](#) and [HE](#) modes of the home operation.



Example: (see [HL](#) command on page 5-89)

Note: In [AA](#) or [AM](#) modes, a null argument in the parameter list specifies that axis home switch to be unchanged.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HH; | AX - AT | 1 | 0 |
| HHb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [EH](#), [HE](#), [HL](#), [HM](#), [HR](#), [HS](#), [KM](#), [KR](#)

HL HOME LOW

The HL command sets the sense of the home switch on the current axis to active TTL low. This is the power-up and reset default “true” state unless [HH](#) has been saved as a user power-up default (See the [AP](#) command, page 5-33.). This command sets the “true” state of a home input to a TTL logic low.

This command sets the home switch “true” state to TTL logic low for both [HS](#) and [HE](#) modes of home operation.



Example: The stage is moved through home with the home switch set for active low at low speed to meet the less than 2048 steps per second requirement of the home command.

Enter: [AX](#);
[VL](#)2000;
 HL;
[HM](#)0;

Response: None

Note: In [AA](#) or [AM](#) modes, a null argument in the parameter list specifies that axis home switch to be unchanged.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HL; | AX - AT | 1 | 0 |
| HLb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [EH](#), [HH](#), [HE](#), [HS](#), [HM](#), [HR](#), [KM](#), [KR](#)

HM**HOME**

The HM command will cause the current axis or specified axes to move in the positive direction at the predefined velocity until the home is detected for each axis. The position counters will be initialized to the positions supplied as parameters.

How home is detected depends on whether the [HS](#) or [HE](#) modes has been selected. The default mode, [HS](#), detects home based on an external home input only. The [HE](#) mode detects home based on the external home input and the encoder signals (Index, PhaseA and PhaseB).

The velocity should be less than the update rate to maintain accuracy of the home position loaded. A velocity set to less than or equal to the update rate will provide a homing accuracy of +/-0 counts. Every multiple of the update rate adds +/-1 count to the error range.

Each axis will, when home is detected, reset its position counter to the parameter specified in the HM command. Once the counter is reset, the axis will ramp to a stop at the rate specified previously via the [AC](#) command. This will result in the axis stopping beyond the home switch so care should be taken to ensure adequate stopping distance is available beyond the switches. The axis can be easily returned to the precise home position by using an [MA](#) command after the HM command.

If no parameter is specified in single axis modes, the HM command will use zero as a default value. Parameters must be specified in multi-axis modes to inform the UMX which axes are to be homed.



Example:

Find the physical home position of the X axis of the stage. The motor runs until the home switch input is activated and then initializes the position counter to the parameter supplied. Since the motor decelerates to a stop after reaching home, it is necessary to do an [MA](#) to the same position as specified in the home command if it is desired to physically position the device at home. The following commands will find home, initialize it to 1000 counts, and then return to home. In many cases it will not be necessary to return home, only find the position and synchronize the controller to it.

Enter:

```
AX;  
VL1000;  
HM1000;  
MA1000;  
GO;
```

Response:

None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HM#; | AX - AT | 3* | 1* |
| HM#,#,#,; | AA-AM | 3* | 11* |
| - | AA/CD | Not Valid | |

| Axis Ramp Type | Command queue | Argument queue |
|---|---------------|----------------------------------|
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

- If the axis is a stepper and encoder or servo axis add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and 1 to the argument queue.
- If the last profile move, just prior to this home command was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the home. This will add to the queue requirements under the following conditions:

Related commands: [HE](#), [HH](#), [HL](#), [HR](#), [HS](#), [KM](#), [KR](#), [LO](#), [LP](#), [RP](#)

HN**HOLD ON**

For servo axes, the HN command closes the loop, enabling the PID. For servo axes, this mode is disabled when the [HF](#) command is entered, when the servo error becomes too large or a limit is encountered.

For stepper with encoder feedback axes, the HN command enables position correction after a move and activates the [HV](#), [HG](#) and [HD](#) commands for stepper axes with encoders. For stepper axes with encoders, this mode will be cancelled (as though via an HF commands) is an [CG](#), [CX](#), [HM](#), [HR](#), [LP](#), [MV](#), [SA](#), [SO](#), [SP](#), or [ST](#) commands is entered, if a limit is encountered or the minimum allowable position error is executed.

(Stepper)

Example:

The following commands could be used to set up the position correction mode on a stepper axis. This sequence sets up a move velocity of 100,000 steps per second and an acceleration of 500,000 steps per second per second. The position correction velocity is set for 50,000 steps per second, a deadband of 10 steps and correction gain of 2,000. The correction is then enabled. A 200,000 step move is performed, then that position is maintained within the 10 step deadband until commanded to a new position.

Enter:

[AX](#);
[VL](#)100000;
[AC](#)500000;
[HV](#)50000;
[HD](#)10;
[HG](#)2000;
 HN;
[MR](#)200000;
[GO](#);

Response:

None

(Servo)

Example:

Close PID loop or X axis after having modified one of the PID parameters

Enter:

[AX](#);
 HN;

Response:

None.

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | 1 | 3 | 3 |
| AA | 1 | 3 | 3 |
| AM | 1 | 3 | 1 |
| AA/CD | Not Valid | | |

Related commands: [HF](#)

HR HOME REVERSE

The HR command will cause the current axis or specified axis to move in the negative direction at the predefined velocity, until the home is detected for each axis. When the home input is detected, the position and encoder counters are loaded with the parameter(s) supplied in the HR command. Then the axis is ramped to a stop. It behaves exactly like the [HM](#) command, except it travels in the reverse direction.

How home is detected depends on whether the [HS](#) or [HE](#) modes has been selected. The default mode, [HS](#), detects home based on an external home input only. The [HE](#) mode detects home based on the external home input and the encoder signals (Index, Phase A and Phase B).



Example: In a long stage it may be awkward to travel the full distance to home at less than the update rate. The following will get close to home at higher speed, and then refine the position at lower speed in the reverse direction.

Enter: [AX](#);
[VL](#)100000;
[HH](#);
[HM](#);
[VL](#)1000;
[HL](#);
 HR;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HR#; | AX - AT | 3* | 1* |
| HR#,#,#,#; | AA-AM | 3* | 1* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 2 to the argument queue.
- If [PA](#) (power automatic) mode is active add 2 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and 1 to the argument queue.
- If the last profile move, just prior to this home command, was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the Home. This will add to the queue requirements under the following conditions:

| Axis Ramp Type | Command queue | Argument queue |
|---|---------------|----------------------------------|
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

Related commands: [HE](#), [HH](#), [HL](#), [HM](#), [HS](#), [KM](#), [KR](#), [LO](#), [LP](#)

HS HOME SWITCH



The HS command disables [HE](#) mode and returns the UMX to the default home behavior. Default behavior defines a home state to be active when the home switch input is either low in default or HL mode or high when in [HH](#) mode. This mode can also be used with encoders which contain internal home logic by connecting their output to the UMX home input for the appropriate axis. The active level of this input may be controlled by the [HH](#) and [HL](#) commands.



Example: Set up the Y axis so it will ignore the encoder signals and only use the home input to recognize the home position.

Enter: [AY](#);
HS;

or

[AM](#);
HS,1;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HS; | AX - AT | Immediate | |
| HSb,b,b,b; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [HE](#), [HH](#), [HL](#), [HM](#), [HR](#), [KM](#), [KR](#)

HV HOLD VELOCITY



The HV command specifies the maximum velocity to be used when correcting position error. The factory default setting is zero; some value must be set for position correction to occur at all. See the [AP](#) command, page 5-33, to preserve the HV settings as the Power up/Reset values. This command is not designed for use with servo motors.

Hold gain ([HG](#)) will be used to scale the HV value based on the total error that must be corrected. In most cases the HV value will never be reached unless the position error is very wide or the [HG](#) value is set very high.

RANGE: $1 \leq HV \leq 1,044,000$



Example: (see [HN](#) command, see page 5-92)

NOTE: In [AA](#) or [AM](#) mode, a null value in the argument list specifies that the hold velocity parameter is not to be changed for that axis.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| HV#; | AX - AT | 1 | 1 |
| HV#,#,#,#; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [HD](#), [HE](#), [HG](#), [HN](#), [?HV](#)

?HV**REPORT POSITION
MAINTENANCE VELOCITY**

The ?HV command reports the current setting of the position maintenance hold velocity limits for the current axis. This command works only with stepper + encoder axes.



Example: Check the peak correction velocity for the T axis

Enter: [AT](#);
?HV

Response: hv20000<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?HV | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?HD](#), [?HG](#), [HV](#)

IC INTERRUPT CLEAR



The IC command or the ASCII character Control-Y (hex 19) is used to clear the done and error flags in the status register and the done flag register on UMX, otherwise the axis would always appear to be “done”. This command will be executed immediately and will usually be placed in the done and error handler interrupt service routine to clear the interrupt and the associated flags. The flags may be polled by an [RA](#) or [RI](#) command which will also reset the flags.

Note: That this command is not recommended in Windows environments using OMS-supplied device drivers and DLLs. The preferred method is to use the device driver and DLL to handle the reporting and clearing status flags.



Example: Clear the flags after an X axis move relative of 5000 steps was flagged as done when an [ID](#) executes.

Enter: [AX](#);
[MR](#)5000;
[GO](#);
[ID](#); (done flag set)
 IC; (done flag cleared on UMX)

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| IC; | AX - AT | Immediate | |
| IC; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [CA](#), [GD](#), [ID](#), [IJ](#), [IN](#), [IP](#)

ID INTERRUPT WHEN DONE

The ID command will set the done flag for the axis or axes to which the command was issued. An interrupt to the host will be generated if interrupts have been enabled in the UMX. In Windows environments, this interrupt is captured by the device driver which will store the flags to pass on to the host application when requested.

This command allows the UMX to signal the host when a string of commands has been completed. In [AA](#) mode, the done flag register bits will be set as each axis encounters the ID in its command stream, but the done flag in the status register will not be set until all axes have executed the ID command. In the [AM](#) mode, only the axes active in the most recent move will set their done flags.

Though move commands are most commonly used with the ID command, others may be used as well. The ID command may be sent to the UMX to tell it to set done flags whenever the host application could benefit from knowing an event or series of commands has completed.



Example: Interrupt the host CPU after the execution of Move Absolute is finished. When the move is finished the ID command will be encountered in the command queue and will set the done flags.

Enter: [AX](#);
[MA](#)100000;
[GO](#);
 ID;

Response: None



Example: Wait until an input line becomes false then notify the host that this occurred.

Enter: [SW](#)2;
 ID;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|---------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| ID; | AY – AT | 1 | 0 | N/A |
| ID; | AA-AM | 1 | 0 | N/A |
| ID; | AA/CD | 1 | 0 | 1 |

Related commands: [II](#), [IN](#), [IP](#)

II INTERRUPT INDEPENDENT



Like the [ID](#) command, the II command tells the UMX to interrupt the host when each axis finishes a move. Unlike the [ID](#) command, only those axes which have been supplied parameters in the most recent move command will get their done flags set and cause interrupts.



Example: The following command sequence would cause interrupts when the Y and T axes finish. If they do not complete at the same time, two separate interrupts would be generated.

Enter: [AM](#);
[MR](#),1000,,10000;
[GO](#);
 II;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Not Valid | |
| II; | AA - AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [ID](#), [IN](#), [IP](#)

IN INTERRUPT NEARLY DONE

The IN command allows the UMX to interrupt the host when the axis or combination of axes is nearly complete. When used in an application involving probing a part after a move, the probes could start accelerating down while the stage is finishing its move, improving the overall system throughput. This command is valid in all modes. The IN command must be entered before the [GO](#) or [GD](#) command since it is executed before the move is complete. The test is only performed during deceleration. If the IN parameter is greater than the ramp down distance, the interrupt will be generated when the control starts decelerating.

RANGE:

Min Position Range ≤ Parameter (Distance Before the End of Move) ≤ MAX Position Range

NOTE: The position range depends on the settings used on UMX, see [Specifications](#) for details.



Example: The following sequence would interrupt the host when the X axis is complete and the Z axis is within 10,000 counts of being complete. The Y axis completion would be ignored in this example.

Enter: [AA](#);
 IN0,,10000;
[MR](#)100000,100000;
[GO](#);
[MR](#),,50000;
[GO](#);

Response: None

NOTE: In [AA](#) or [AM](#) mode, a null value in the argument list specifies the position tolerance for nearly done is not to be set for that axis.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| IN#; | AX - AT | 1 | 2 |
| IN#,#,#,#; | AA-AM | 1 | 2 |
| - | AA/CD | Not Valid | |

Related commands: [ID](#), [II](#), [IP](#)

IO SET I/O BIT DIRECTION



The IO command defines the direction of the general purpose I/O bits 0 to 7 as inputs or outputs. The bit direction selection is made in groups of four bits. A null argument skips that bit group. A zero value sets that bit group to be inputs. A one value configures that bit group to be outputs. The factory default configuration is: bits 0-3 are inputs and bits 4-7 are outputs.

See the [AP](#) command on page 5-33 to preserve the IO setting as the Power up/Reset setting.

NOTE: Bits 8-15 are not configurable.



Example: Set bits 0-3 as output and bits 4-7 as inputs.

Enter: IO1,0;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | Immediate | | |
| AA - AM | Immediate | | |
| AA/CD | Not Valid | | |

Related commands: None

IP**INTERRUPT WHEN IN POSITION**

The IP command operates like the [ID](#) command except the interrupt is deferred until the stage is within the specified step encoder dead band. If the position hold ([HN](#)) is not enabled for an axis, the command will behave like an [ID](#) command for that axis.



Example: Set the done flag when axis is within its dead band.

Enter: [AX](#);
[HV](#)1000;
[HG](#)100;
[HD](#)10;
[HN](#);
[MR](#)1000;
[GO](#);
 IP;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| IP; | AX - AT | 1 | 0 |
| IP; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [HN](#), [ID](#), [II](#), [IN](#)

IS INTERRUPT ON SLIP



The IS command enables the UMX to interrupt the host on slip or stall detection if the appropriate bit has been set in the interrupt control register. Slip detection are disabled if an [ER](#), [ET](#), [HF](#), [HM](#), [HR](#), [JG](#), [LP](#), or [TM](#) command is entered or if a limit is encountered. If a slip occurs, slip detection must be re-enabled. The factory default slip tolerance ([ES](#)) value is 1. This command is intended to be used with stepper motors and not servo motors.



Example: (see [ES](#) command on page 5-72)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| IS; | AX - AT | 1 | 0 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ES](#), [RL](#), [TF](#), [TN](#)

IX INTERRUPT WHEN AXES ARE DONE



The IX command is a special form of the [ID](#) command. It is intended for use by the serial communications version of the UMX. Each axis which has been supplied a position in the most recent move command will send a special done character to the host as they complete their move.

| AXIS DONE INDICATOR CHARACTERS | |
|--------------------------------|-----------------------|
| AXIS | DONE CHARACTER IN HEX |
| X | 80 |
| Y | 81 |
| Z | 82 |
| T | 83 |



Example: The following command sequence would cause two characters to be sent to the host. An 81 hex character will be sent when the Y axis finishes and an 83 hex character when the T axis finishes.

Enter: [AM](#);
[MR](#),1000,,10000;
[GO](#);
 IX;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|-----|
| FORMAT | MODE | Min | Max |
| - | AY – AT | Not Valid | |
| IX; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [ID](#), [IN](#), [IP](#)

JF**JOG FRACTIONAL VELOCITIES**

The JF command will jog one or more axes at the velocities specified, like the [JG](#) command. The parameter may include a fractional part allowing better resolution at low speeds. The velocity set by this command will remain the default velocity until altered by a [VL](#), [JG](#) or another JF command.

RANGE: $-1,044,000 \leq JF \leq 1,044,000$



Example: Jog the Y axis at 2 2/3 steps per second.

Enter: [AY](#);
JF2.667;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| JF#; | AX - AT | 3* | 0* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If the axis is a stepper and encoder or servo axis add 1 to the command queue
- If the profile move, just prior to this home command was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the home. This adds to the queue requirements under the following conditions:

| AXIS RAMP TYPE | COMMAND QUEUE | ARGUMENT QUEUE |
|---|---------------|----------------------------------|
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

Related commands: [JG](#), [SA](#), [ST](#), [TM](#)

JG**JOG**

The JG command is a velocity mode and will move one or more axes at the velocities supplied as parameters. The JG command will accelerate to the programmed velocity at the current [AC](#) rate and run until altered by an [ST](#), [SA](#), [KL](#), [HF](#) (servo models), another JG command, or a limit switch is encountered while limits are enabled.

The jog velocity may be changed by following the command with another JG command of a different velocity. A change in direction between two JG commands will cause an axis to ramp to a stop then back up in the opposite direction.

This command modifies the move velocity parameter ([VL](#)) for the affected axis or axes. The JG command does not require any other command to start the motion.

RANGE: $-1,044,000 \leq \text{JG} \leq 1,044,000$



Example: Jog the motor at 100,000 counts per second then change to 35,000 counts per second when the second JG is entered, stay at that velocity for 5 seconds, then stop by decelerating to a stop. Next, jog the motor at 5,000 counts per second in the negative direction.

Enter: JG100000;
JG35000;
[WT](#)5000;
[ST](#);
JG-5000;

Response: None

Note: Output events waiting for completion of JG will begin when JG is up to its requested velocity. In this case, the motor will ramp from zero to 100,000, ramp back down to 35,000, flatten out at 35,000 for 5 seconds, then ramp to a stop, before moving in the negative direction at a velocity of 5,000.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| JG#; | AX - AT | 3* | 1* |
| JG#,#,#,; | AA-AM | 3* | 1* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the Command queue
- If the axis is a stepper encoder or servo axis add 1 to the Command queue
- If the profile move, just prior to this home command was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the home. This add to the queue requirements under the following conditions:

| Axis Ramp Type | Command queue | Argument queue |
|---|---------------|----------------------------------|
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

Related commands: [JF](#), [SA](#), [ST](#), [TM](#)

KA ACCELERATION FEEDFORWARD

KA is the acceleration feedforward gain coefficient used in the PID filter calculations. The factory default value is zero. See the [AP](#) command, page 5-33, to preserve the KA settings as the power up/reset values. The default value is 0.00

RANGE: $0.00 \leq KA \leq 32767.00$



Example: Define KA to be 2 on the T axis.

Enter: [AT](#);
KA2;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KA#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [?KA](#), [KD](#), [KI](#), [KP](#), [KV](#)

**?KA REPORT ACCELERATION
FEED FORWARD**

The ?KA command reports the current setting of the acceleration feed-forward constant ([KA](#)) for the current servo axis. Default value is 0.00



Example: Find out what the current [KA](#) value is for servo axis Y

Enter: [AY](#);
?KA

Response: ka10.50<LF>

| QUEUE REQUIREMENTS | | | |
|----------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ? KA | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [KA](#), [?KV](#)

KB PID UPPER BOUND LIMIT COEFFICIENT



The KB command sets the servo axis PID output value upper bound limit from DAC. (This can be useful in debugging servo drives.) The default value is 0.00

RANGE: 0.0 < KB < 10.0



Example: Limit the servo output from DAC to 9 max on Y axis. This is approximately half of its normal range.

Enter: [AX;](#)
KB9;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KB#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KB](#)

?KB REPORT AXIS PID UPPER BOUND LIMIT



The ?KB command requests the value of the [KB](#) parameters for the upper bound limit to the DAC.



Example: Determine PID upper bound limit for X axis

Enter: [AX;](#)
?KB

Response: kb5.00<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KB | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [DBI](#), [DBN](#), [?SV](#)

KD DERIVATIVE GAIN COEFFICIENT

KD is the derivative gain coefficient used in the PID filter calculations. See Section 2 for more information regarding this parameter. The factory default value is 20.00. See the [AP](#) command, page 5-33, to preserve the KD settings as the Power up/Reset values.

RANGE: 0.00 ≤ KD ≤ 32767.00



Example: Set KD to 56 on the Z axis.

Enter: [AZ](#);
KD56;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KD#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KD](#), [KI](#), [KP](#), [HF](#), [HN](#)

?KD REPORT PID DERIVATIVE GAIN

The ?KD command reports the current setting of the derivative gain coefficient ([KD](#)) in the PID of the current servo axis.



Example: Forgot to write down the Y axis [KD](#) setting which is working well. Report the setting so it can be recorded.

Enter: [AY](#);
?KD

Response: kd5.12<LF >

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KD | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [KD](#), [?KI](#), [?KP](#),

KF SET SERVO AXIS PID FRICTION COEFFICIENT



The KF command sets the friction offset coefficient of a servo axis PID. This assists in the smooth start up motion of the X axis, and compensates for friction. The default value is 0.

Range: $0 \leq KF \leq 32767$



Example: Set the Y axis friction coefficient to 100.

Enter: [AY](#);
KF100;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KF#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KF](#)

?KF REPORT SERVO AXIS PID FRICTION COEFFICIENT



The ?KF command reports the value of the frictional offset coefficient.



Example: Verify Z axis friction offset coefficient is 12

Enter: [AZ](#);
?KF

Response: kf12<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KF | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [KD](#), [?KI](#), [?KP](#)

KI INTEGRAL GAIN COEFFICIENT



KI is the integral gain coefficient used in the PID filter calculations. See the [AP](#) command, page 5-33, to preserve the KI settings as the power up/reset values. Default value is 0.04.

RANGE: $0.00 \leq KI \leq 32767.00$



Example: Define KI to be 3.42 on the X axis.

Enter: [AX](#);
KI3.42;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KI#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [KD](#), [?KI](#), [KP](#)

?KI REPORT PID INTEGRAL GAIN



The ?KI command reports the current setting of the integral gain constant ([KI](#)) in the PID of the current servo axis.



Example: Report the setting of the [KI](#) coefficient on the Z axis

Enter: [AZ](#);
?KI

Response: ki1.00<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KI | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KD](#), [KI](#), [?KP](#)

KL KILL

The KL command will flush the command queues and terminate pulse generation of all axes immediately. It is intended for emergency termination of any program and to reset the input queues to a known state.

Step motors may not stop immediately even though no more step pulses are delivered due to inertia of the motor and system load. This may result in slippage of the motor. Therefore, the position counter may not accurately reflect the true position of the motor following this command. All axes should be re-homed to return the position counters to a known state.

Due to the encoders used in servo systems, position will not be lost, so re-homing servo axes is unnecessary.



Example: Stop all previously defined movement and flush the queue of a partially entered incorrect move command (you wanted a negative move not a positive one), before [GO](#) is entered.

Enter: [AX](#);
[MR](#)5000; (oops!)
 KL;
[MR](#)-5000;
[GO](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KL; | AX - AT | 1* | 0* |
| KL; | AA-AM | 1* | 0* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [FL](#), [KS](#), [SA](#), [SD](#), [SI](#), [SO](#), [ST](#)

KM HOME AND KILL



The KM command will move the current axis in the positive direction until home is detected and then kill motion immediately; i.e. without using a deceleration ramp. The position counter will not be reset or cleared. Due to motor and/or payload inertia, the motor may not stop immediately but slip some distance instead. This will result in inaccurate position counters. This command's primary purpose is to move an axis out of the way quickly or to get the axis near home rapidly to speed up the homing process.



Example: Move the Y axis in the positive direction to the home sensor and stop movement as quickly as possible.

Enter: [AY](#);
KM;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KM; | AX - AT | 1* | 1* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 2 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and 1 to the argument queue.
- If the last profile move, just prior to this home command, was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the Home. This will add to the queue requirements under the following conditions:

| Axis Ramp Type | Command queue | Argument queue |
|---|---------------|----------------------------------|
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

Related commands: [HE](#), [HH](#), [HL](#), [HM](#), [HR](#), [HS](#), [KR](#), [LO](#), [LP](#)

KO OFFSET COEFFICIENT



The KO command defines the offset coefficient to cause the motor to remain stationary and compensate for additional torque on the motor from loading. The factory default value is zero. See the [AP](#) command, page 5-33, to preserve the KO settings as the power up/reset values.

The factory default value is zero. Full-scale, the KO command has a range of $\pm 32,767$ which corresponds directly to the 16-bit range of the DAC less a few counts as a buffer zone. Each increment/decrement of the KO value will result in an approximate change in the output voltage of 0.0003 volts.

RANGE: $-32767 \leq KO \leq 32767$



Example: Define the offset coefficient to be -2000 ($\approx -610\text{mV}$) on the Y axis.

Enter: [AY](#);
KO-2000;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KO#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [?KO](#)

?KO REPORT PID OFFSET COEFFICIENT



The ?KO command reports the voltage offset ([KO](#)) setting for the current servo axis.



Example: The open-loop offset is 218. Make sure the closed-loop offset is the same.

Enter: ?KO

Response: ko218<LF >

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KO | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [KO](#)

KP PROPORTIONAL GAIN COEFFICIENT



KP is the proportional gain coefficient used in the PID filter calculations. See the [AP](#) command, page 5-33, to preserve the KP settings as the power up/reset values. Default value is 10.00.

RANGE: $0.00 \leq KP \leq 32767.00$



Example: Define KP to be 45.6 on the Z axis.

Enter: [AZ](#);
KP45.6;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KP#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [KD](#), [KI](#), [?KP](#)

?KP REPORT PROPORTIONAL GAIN COEFFICIENT



The ?KP command reports the current setting of the proportional gain coefficient ([KP](#)) in the PID of the current servo axis.



Example: Find out what the X axis proportional gain is set to.

Enter: [AX](#);
?KP

Response: kp10.00<LF >

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KP | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KD](#), [?KI](#), [KP](#)

KR HOME REVERSE AND KILL

The KR command will find home by issuing a [JG](#) in reverse. When home is found, it will stop generating pulses immediately; i.e. no deceleration ramp will be generated. This command is identical to the [KM](#) command except that the direction of motion is reversed.



Example: Move the Y axis in a negative direction to the home sensor and stop movement as quickly as possible.

Enter: [AY](#);
KR;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KR#; | AX - AT | 1* | 1* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 2 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and 1 to the argument queue.
- If the last profile move, just prior to this home command, was either a [MT](#) or [ML](#) move then the axis acceleration and velocity values will be reset to the [AC](#) and [VL](#) values just prior to the execution of the Home. This will add to the queue requirements under the following conditions:

| QUEUE REQUIREMENTS | | |
|---|---------------|----------------------------------|
| Axis Ramp Type | Command queue | Argument queue |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 6 |
| All other Parabolic forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2 x number of ramp segments) +2 |

Related commands: [HE](#), [HH](#), [HL](#), [HM](#), [HR](#), [HS](#), [KM](#), [LO](#), [LP](#)

KS KILL SELECTED AXES

This command performs the same operation as the [KL](#) (kill) command except that individual axes can be killed without affecting others. KS will flush only the selected axes' command queues rather than the entire board. Refer to the [KL](#) command for more details.



Example: The Y axis has hit a limit switch and is now executing commands that were waiting in the queue. This axis must be reset but the other axes must be allowed to continue operation.

Enter: [AY](#);
KS;
or
[AA](#);
KS,1;

Response: None

NOTE: In [AA](#) or [AM](#) modes, null values in the argument list specify that motion on that axis is not to be killed.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KS; | AX - AT | 1* | 0* |
| KSb,b,b,b; | AA-AM | 1* | 0* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [FL](#), [KL](#), [SA](#), [SD](#), [SI](#), [SO](#), [ST](#)

KU PID INTEGRATION SUM UPPER LIMIT



This command sets servo axis PID integration sum upper limit.

RANGE: $0 < KU < 32767$



Example: Set the integration sum upper limit of X axis to 150

Enter: [AX](#);
KU150;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KU#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KU](#)

?KU REPORT PID INTEGRATION SUM UPPER LIMIT



Report servo axis PID integration sum upper limit.



Example: Check to make sure the integration sum upper limit of X axis is less than 200.

Enter: [AX](#);
?KU

Response: ku150<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KU | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [KD](#), [KP](#), [KU](#)

KV VELOCITY FEEDFORWARD



KV is the velocity feedforward coefficient used in the PID filter calculations. The factory default value is zero. See the [AP](#) command, page 5-33, to preserve the KV settings as the power up/reset values.

RANGE: $0.00 \leq KV \leq 32767.00$



Example: Set KV to 35.3 on the Y axis.

Enter: [AY](#);
KV35.3;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| KV#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#), [KA](#), [KD](#), [KI](#), [KP](#), [?KV](#)

?KV REPORT VELOCITY FEED FORWARD



The ?KV command reports the current velocity feed forward coefficient ([KV](#)) of the current servo axis. Default value is 0.00



Example: Make sure the velocity feed-forward setting of axis T is zero

Enter: [AT](#);
?KV

Response: kv0.00<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?KV | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?KA](#), [KV](#)

LA LINEAR RAMP PER AXIS

The LA command specifies that the linear acceleration ramp is to be used by the selected axes. This is the factory default for all axes. See the [AP](#) command, page 5-33, to preserve the LA settings as the power up/reset values. This command is similar to the [PF](#) command but can be used to switch a single axis rather than all axes at once.



Example: Select a linear ramp for the X axis.

Enter: [AX](#);
 LA;

Response: None



Example: Select the linear ramp for the Y and T axes.

Enter: [AA](#);
 LA,1,,1;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LA; | AX - AT | 5 | 6 |
| LAb,b,b,b; | AA-AM | 5 | 6 |
| - | AA/CD | Not Valid | |

Related commands: [CN](#), [PF](#), [PN](#), [PR](#), [?RT](#), [SC](#), [SR](#)

LE LOOP END

The LE command terminates the most recent [LS](#) command. The axis will loop back and repeat the commands within the loop the number of times specified in the [LS](#) command. The loop will start repeating as soon as this command is issued.



Example: Perform a relative move on axis Z five times. After each Z move, wiggle the T axis 20 times.

Enter:

```

AZ;
LS5;
MR5000;
GO;
AT;
LS20;
MR50;
GO;
MR-50;
GO;
LE; (terminates the LS20; command)
AZ;
LE; (terminates the LS5; command)

```

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LE; | AX - AT | 1 | 1 |
| LE; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [LS](#)

LF**LIMITS OFF**

The LF command disables the limit switches for the addressed axis or axes. This allows the stage to move beyond the limit switches and should be used with caution. The UMX will still recognize that a limit switch has been closed if the stage runs into one and will report this information via the query commands (See the [QA](#) command). However, the limit switch closure will have no effect on motion; i.e. the axis will not be forced to stop as a result.

NOTE: In systems not designed to handle motion beyond the limit switch points, this can potentially cause damage to the system and/or persons operating the system. This command should be used with extreme caution.



Example: Set up a board to ignore the Y axis limit switches.

Enter: [AY](#);
 LF;
 or
[AA](#);
 LF,1;

Response: None

Note: In [AA](#) or [AM](#) modes, a null argument in the parameter list specifies that axis home switch to be affected.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LF; | AX - AT | 1 | 0 |
| LFb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LH](#), [LL](#), [LN](#), [SF](#), [SL](#)

LH LIMITS HIGH

The LH command sets the senses of the limit switches on the current axis to active high. The default “true” states of the limits are TTL logic low. This command allows TTL logic high to be treated as the “true” state for applications where this is more convenient. Through the execution of the [AP](#) command, limits can be made to default as active high on power-up or reset; see the [AP](#) command for details.



Example: Select the limit switch high true condition for the X axis.

Enter: [AX](#);
LH;

Response: None



Example: Select a high true limit condition for the Z and T axes.

Enter: [AA](#);
LH,,1,1;

Response: None

Note: In [AA](#) or [AM](#) modes, a null argument in the argument list specifies that axis is not to be affected.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LH; | AX - AT | 1 | 0 |
| LHb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LL](#), [LN](#), [?LS](#), [SF](#), [SL](#), [TL](#)

LL LIMITS LOW

The LL command specifies that over travel occurs when the limit input signal is low (active low). This is the factory default mode. See the [AP](#) command for information on how to preserve the LL settings as the power up/reset values.



Example: Select the limit switch low true condition for the X axis.

Enter: [AX](#);
LL;

Response: None



Example: Select a low true limit condition for the Y and T axes.

Enter: [AA](#);
LL,1,,1;

Response: None

Note: In [AA](#) or [AM](#) modes, a null argument in the argument list specifies that axis is not to be affected.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LL; | AX - AT | 1 | 0 |
| LLb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LH](#), [LN](#), [?LS](#), [SF](#), [SL](#), [TL](#)

LN LIMITS ON



The LN command restores the operation of the limit switches for the addressed axis or axes. This is the default mode at power up or reset. With limit switches enabled, if the axis encounters a limit switch in the course of executing any motion, the axis will be instructed to stop and the host will be notified of the event. Limit conditions are treated as critical errors and should not be used as simple positioning inputs to the UMX.



Example: Set up the Y and T axes to stop immediately when a limit switch is encountered.

Enter: [AA](#);
LN,1,,1;
or
[AY](#);
LN;
[AT](#);
LN;

Response: None

Note: In [AA](#) or [AM](#) modes, a null argument in the parameter list specifies that axis is not to be affected.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LN; | AX - AT | 1 | 0 |
| LNb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LH](#), [LL](#), [SF](#), [SL](#)

LO LOAD MOTOR POSITION

The LO command sets the motor position independently of the encoder position unlike [LP](#) which sets both to the same supplied value. The [LP](#) command will override the LO command and reset the motor position. If the [LP](#) command is used and a different motor position value than the encoder position is desired, the LO command must be reentered. Any valid position within the allowable position range may be used.



Example: Set the motor position to 50,000 and the encoder position to 100,000 on the T axis

Enter: [AT](#);
[LP](#)100000;
 LO50000;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LO#; | AX - AT | 1 | 1 |
| LO#,#,##; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

NOTE: For [AA](#) and [AM](#) modes that any values in the argument list specify that axis is not to be affected.

Related commands: [LP](#), [RE](#), [RM](#), [RP](#), [RU](#)

LP LOAD POSITION



The LP command will immediately load the number supplied as a parameter in the absolute position registers of the axis. In models with the encoder option, the parameter will be loaded into the encoder position register and the parameter times the encoder ratio will be loaded into the position counter. If no parameter is supplied, the value of zero is used.

The [LO](#) command can be used after this command to set the motor position independently of the encoder position.

RANGE: Min. Position Value ≤ LP ≤ Max. Position Value

NOTE: The position value range varies with the specific settings used on UMX.



Example: The following would load the X axis position register with 1000, and the Z axis position register with 2000.

Enter: [AA](#);
LP1000,,2000;

Response: None



Example: The following would load the Y axis position register with 20,000 and the encoder position register with 30,000 counts, in encoder models.

Enter: [AY](#);
[ER3,2](#);
LP30000;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LP#; | AX - AT | 1* | 1* |
| LP#,#,#,#; | AA-AM | 1* | 1* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue and 1 to the argument queue.

Related commands: [LO](#), [RE](#), [RM](#), [RP](#), [RU](#)

LS LOOP START



The LS command sets the loop counter for the axis being programmed in the single axis mode and all axes in the [AA](#) mode. The command expects a loop counter operand following the command. The commands up to the [LE](#) loop terminator will be executed the number of times specified by the operand. Loops may be nested up to four levels deep on each axis. The parameter must be less than 32000.

The first loop of commands will occur immediately as they are entered. The remaining loops will be executed after the loop terminator ([LE](#)) has been entered.

The axis mode (e.g. [AX](#), [AY](#), and [AA](#)) must be the same when entering and exiting the loop, otherwise the matching loop termination command will not be found by the board's command processor. The axis mode may be switched within the loop provided the board is in the same mode when the [LE](#) command is sent as when the LS command was sent.

If you want one axis to wait for another in the loop, you must be in the [AA](#) mode throughout the loop. If you are in the single axis mode in the loop, each axis' commands will go into their separate queues and execute independently of each other.

If, when entering a looping sequence of commands, the command queue is filled before the [LE](#) loop terminator is entered, the board will hang. This is because there is no space for the [LE](#) command. When programming a loop of more than four or five moves, the queue size should be examined with the [RQ](#) command to see if it is nearing zero.

RANGE: $1 \leq LS \leq 32000$



Example: Execute a 100,000 count relative move on the Z axis 5 times.

Enter: [AZ](#);
[LS](#)5;
[MR](#)100000;
[GO](#);
[LE](#);

Response: None

NOTE: The first move will occur immediately after entering the [GO](#) command. The remaining 4 moves will be executed after the loop terminator [LE](#) has been entered.



Example: Execute a 100,000 count move relative on the X axis together with a 100 count move on the T axis, followed by a move absolute to 100 counts on the X axis and 200 counts on the T axis, four times.

Enter: [AA](#);
 LS4;
[MR](#)100000,,,100;
[GO](#);
[MA](#)100,,,200;
[GO](#);
[LE](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| LS#; | AX - AT | 1 | 1 |
| LS#; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [LE](#), [WH](#), [WS](#)

?LS**REPORT LIMIT ACTIVE STATE**

The ?LS command reports the active state of the limits for the current axis. The [LL](#) and [LH](#) commands are used to set this value.



Example: Find out whether the Y axis limits are active high or active low.

Enter: [AY](#);
?LS

Response: ll<LF> or lh<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?LS | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [LH](#), [LL](#)

MA MOVE ABSOLUTE



The MA command will set up one or more axes to move to the absolute positions supplied as parameters. In [AA](#) mode, an axis may remain stationary by entering a comma but omitting the parameter. The move is actually initiated by a [GO](#) or [GD](#) command.

In [AA](#) mode all axes begin their move at the same time. Each axis will use its predefined acceleration and velocity values to move to the new absolute position. Each axis may or may not get to the destination at the same time because each axis utilizes individual velocities and accelerations. The [MT](#) command will ensure all axes reach their target positions simultaneously.



Example: In the single axis mode, move the X axis to absolute position 100,000 counts with the previously entered acceleration and velocity parameters.

Enter: [AX](#);
MA100000;
[GO](#);

Response: None



Example: In the [AA](#) mode, move the Y axis to absolute position 10,000 counts and the T axis to absolute position 1,000 counts. The other axes will remain in their current positions.

Enter: [AA](#);
MA,10000,,1000;
[GO](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MA#; | AX - AT | 1* | 1* |
| MA#,#,#,; | AA-AM | 1* | 1* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper encoder or servo axis add 1 to both the command and argument queues.
- If the axis is using the cosine acceleration ramp, and velocity needs to be adjusted so the profile is not truncated, add 1 to the command queue and add 10 to the argument queue.
- If the acceleration and velocity values need to be reset to their [AC](#) and [VL](#) values because the last move just prior to this move was either a [MT](#) or [ML](#) move, add the following queue requirements:

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

Related commands: [GD](#), [GN](#), [GO](#), [GS](#), [GU](#), [ML](#), [MR](#), [MT](#)

MD**TEMPORARY MACRO DEFINE**

MD is used to begin defining a temporary macro. A macro can contain up to 250 characters. Macros 0 through 4 are temporary and they will be erased when the controller is reset or power is turned off. Macros 5 through 24 are stored in non-volatile memory and will be preserved when the controller is reset or powered off. This command cannot be used to define a macro directly into numbers 5 through 24. They must be defined with this command and then moved into the non-volatile space with the [PT](#) command.

Enter the macro number immediately after the MD command. The macro number must be between 0 and 4. Next enter the command string, which is made up of up to 250 ASCII characters. After entering the command string for the macro, enter a control Z to end the macro definition. The control Z may be ASCII value 26 or the string “^Z” (carat, shift 6 on a US keyboard, followed by a Z.)

Be careful not to exceed 250 ASCII characters or the size of the axis queue when working with macros.

RANGE: $0 \leq MD \leq 4$



Example: Define macro 2 to set velocities to 20000 on all axes of a two axes board.

Enter: MD2;
[AA](#);
[VL](#)20000,20000;
 ^Z

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MD#; | AX - AT | Immediate | |
| MD#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [MX](#), [PM](#)

ML**MOVE LINEAR**

The ML command uses linear interpolation to perform a straight line relative move. Input parameters are relative distances for each axis involved in the move. The ML command should be followed by a [GO](#) or [GD](#) to start the axes together. The velocity and acceleration parameters are scaled to allow the axes to move and finish together. All axes are scaled to the axis with the longest move time (the master axis). At the end of the move, all involved velocities and accelerations will be restored to their pre-move values.



Example: In the [AA](#) mode, move the Y, Z and T axes 10000, 100 and 1000 counts respectively with all axes starting and finishing together. The other axes remain in their previous positions.

Enter: [AA](#);
ML,10000,100,1000;
[GO](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Not Valid | |
| ML#,#,#,; | AA-AM | 2* | 2* |
| - | AA/CD | Not Valid | |

- If this is the master axis for this move and its acceleration and velocity values need to be reset to their original [AC](#) and [VL](#) values, because a previous move altered them, add the following queue requirements:
- If this is not the master axis, then the acceleration and velocity values will be modified, and the following queue requirements will be added:

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

If the axis is using the cosine acceleration ramp, and velocity needs to be adjusted so the profile is not truncated, add 1 to the command queue and 10 to the argument queue:

Related commands: [GD](#), [GN](#), [GO](#), [MA](#), [MR](#), [MT](#)

MM MOVE MINUS

The MM command sets the direction logic to move in the negative direction. The direction output for the current axis will change (if necessary) to reflect the new direction. All non-direction-specific move commands will now move in the negative direction.



Example: Set the direction line to move in the negative direction on the Y axis.

Enter: [AY](#);
MM;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MM; | AX - AT | 1 | 0 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [FP](#), [MO](#), [MP](#), [MV](#), [SP](#)

MP MOVE POSITIVE

The MP command sets the direction logic to move in the positive direction. The direction output for the current axis will change (if necessary) to reflect the new direction. All subsequent non-direction-specific motion commands will now move in the positive direction.



Example: Set the direction line to move in the positive direction on the Y axis.

Enter: [AY](#);
MP;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MP; | AX - AT | 1 | 0 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [FP](#), [MM](#), [MV](#), [SP](#)

MR MOVE RELATIVE

The MR command will set up one or more axes to move relative from their current positions at the time the move is executed. In the [AA](#) mode, an axis may remain stationary by entering a comma but omitting the parameter. The move is actually initiated by a [GO](#) or [GD](#) command.

In [AA](#) mode all axes will start at the same time. Each axis will use its predefined acceleration and velocity values to move to the new position. Each axis may, or may not, get to the destination at the same time, because each axis utilizes individual velocities and accelerations. To ensure all axes reach their destinations simultaneously use the [ML](#) command.



Example: In the single axis mode, move the X axis 2468 steps in the negative direction.

Enter: [AX](#);
MR-2468;
[GO](#);

Response: None



Example: In the [AA](#) mode, move the X axis 12345 steps in the positive direction and the Y axis 6789 steps in the positive direction. Both axes will start at the same time.

Enter: [AA](#);
MR12345,6789;
[GO](#);

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MR#; | AX - AT | 1* | 1* |
| MR#,#,#,##; | AA-AM | 1* | 1* |
| - | AA/CD | Not Valid | |

- If the axis is a stepper encoder or servo axis add 1 to both the command and argument queues.
- If the axis is using the cosine acceleration ramp, and velocity needs to be adjusted so the profile is not truncated, add 1 to the command queue and add 10 to the argument queue.
- If the acceleration and velocity values need to be reset to their [AC](#) and [VL](#) values because the last move just prior to this move was either a [MT](#) or [ML](#) move, add the following queue requirements:

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

Related commands: [GO](#), [GD](#), [GN](#), [GS](#), [GU](#), [MA](#), [ML](#), [MT](#)

MT**MOVE TO**

The MT command uses linear interpolation to move two or more axes to the specified absolute positions. The syntax is similar to the [ML](#) command. This command is invalid if loops are being used due to the overhead involved. The command will become valid again after executing an [ST](#) or [KL](#) command. When used in the contour definition mode, only the axes being used in the contour must be provided for in the MT syntax. A [GO](#) or [GD](#) command initiates the move.

The axis that will reach its destination first (the master axis) is used as a gauge to modify the acceleration and velocity values of the other axes. This is done to ensure all involved axes arrive at their targets simultaneously. At the end of the move, any velocity or acceleration value that was modified is restored to its pre-move value.



Example: In the [AA](#) mode, move the X, Y and T axes to absolute positions 1000, 10000 and 100 counts, respectively, with each starting and finishing together. The unused axis remains in its previous position.

Enter: [AA](#);
MT1000,10000,,100;
[GO](#);

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------|----------|---|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX - AT | Not Valid | | |
| MT#,#,##,; | AA-AM | 2* | 2* | N/A |
| MT#,#,##,; | AA/CD | 2* | 2* | 4 + (number of axes included in contour definition) |

- If this is not the master axis, then the acceleration and velocity values will be modified, and the following queue requirements will be added:

| QUEUE REQUIREMENTS | | |
|---|---------------|---------------------------------|
| Axis Ramp Type | COMMAND QUEUE | ARGUMENT QUEUE |
| Linear (PF , LA) | 4 | 4 |
| Short Form Parabolic (PN0 , PR0) | 4 | 8 |
| All other Parabolic Forms (PN , PR) | 4 | 22 |
| Cosine (CN , SC) | 4 | 22 |
| Custom (SR) | 4 | (2*number of ramp segments) + 2 |

- If the axis is using the cosine acceleration ramp, and velocity needs to be adjusted so the profile is not truncated, add 1 to the command queue and 10 to the argument queue:

Related commands: [GD](#), [GO](#), [GN](#), [MA](#), [ML](#), [MR](#)

MV**MOVE VELOCITY**

The MV command causes the current axis to move to a new absolute position (parameter 1) at a new velocity (parameter 2). When the destination is reached control will be passed to the next command which should be another MV command or a [SP](#) command. If the command is not received in time the controller will continue to move at the specified velocity. Note that this is a slave mode and it is the responsibility of the user to provide the commands in time. They may be queued ahead of time. If a new MV command is sent after the controller has already passed the destination specified in the command, the controller will continue to move at the old velocity.

The UMX will ramp up or down as needed at the rate previously set with the [AC](#) command and travel at the new velocity until the new position is reached. The controller will not reverse direction if the position has already passed, but will behave as explained above. Thus the direction of the move must be specified before starting the move with the [MP](#) or [MM](#) commands. All destinations must be in absolute position; no position relative moves are allowed due to the nature of these commands. Cosine and parabolic acceleration will not apply.

RANGE:

Min Position Range ≤ Parameter 1 (Absolute Position) ≤ Max Position Range

1 ≤ Parameter 2 (Velocity) ≤ 1044000

NOTE: The position range depends on the settings used on UMX, see specifications for more details.



Example: Generate a velocity staircase with the breakpoints given in absolute position. Default acceleration ([AC](#)) of 200,000

Enter: [MP](#);
MV10000,30000;
MV20000,50000;
MV30000,10000;
[SP](#)35000;

Response: None

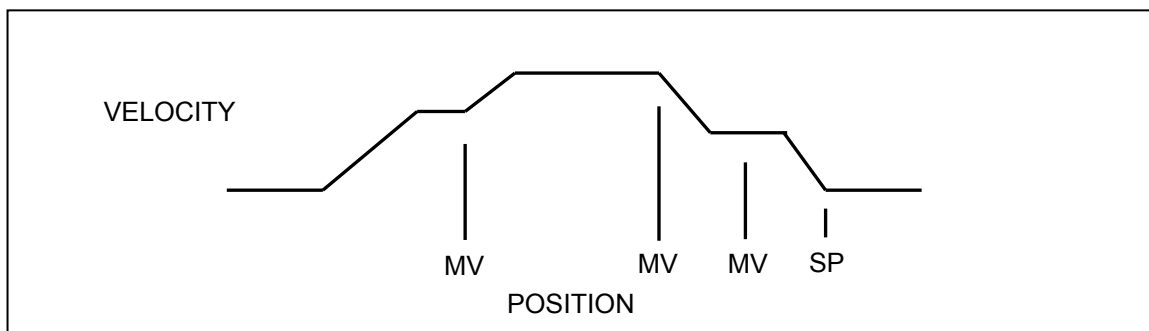


FIGURE 5-1 VELOCITY STAIRCASE PROFILE

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MV#,#; | AX - AT | 2* | 2* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If this axis is a stepper encoder or servo axis add 2 to the command queue

Related commands: [FP](#), [MM](#), [MP](#), [SP](#)

MX MACRO EXECUTE

The MX command will execute the command string stored in the specified macro. The macro number that is entered as the argument of the command must be between 0 and 24.

RANGE: $0 \leq MX \leq 24$



Example: Execute macro number 6.

Enter: MX6;

Response: None

NOTE: MX itself is an immediate command. However, the commands contained within the macro may have queue requirements.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| MX#; | AX - AT | Immediate | |
| MX#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [MD](#), [PM](#), [PT](#)

NV NEW CONTOUR VELOCITY

The NV command will set a new velocity for a contour currently in execution. When the velocity changes, the UMX will switch to the new velocity at the start of the next command in the contour queue without ramping. An NV command issued without a contour currently executing will have no effect.

RANGE: $1 \leq NV \leq 4,000,000$



Example: The contour is executing at 25,000 counts per second. Change the velocity to 30,000 counts per second upon execution of the next command in the contour queue.

Enter: NV30000;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|-----------|----------|---------|
| FORMAT | MODE | COMMAND | ARGUMENT | CONTOUR |
| - | AX - AT | Not Valid | | |
| - | AA-AM | Not Valid | | |
| NV#; | AA/CD | Immediate | | |

Related commands: [CD](#), [CV](#)

PA POWER AUTOMATIC

The PA command will turn on or off the auxiliary outputs at the beginning of each [GO](#) or [GD](#) command execution and complement the outputs after the move is executed. The auxiliary will be turned on; i.e. pulled high, upon the execution of the [GO](#) or [GD](#) and off at the end of that move if the parameter is zero or not specified in the single axis mode. If the parameter is non-zero, the sense is reversed; i.e. the auxiliary output is turned off (driven low) upon the execution of the [GO](#) or [GD](#) command and on at the end of the move.

The [SE](#) command can be used to apply a settling time at the end of each move before complementing the auxiliary bit. This is useful for systems that need to retain torque for some specific amount of time before allowing the motor drive to reduce current output.

This mode need only be set once and can be turned off by using the [AN](#) or [AF](#) commands. Axes can be selectively affected in the [AA](#) mode. The values of the included parameters set the state of the auxiliary line during the move. The following queue requirements apply to each [GO](#) or [GD](#) command in the command stream in the [AA](#) and single axis modes. This mode is off by factory default. See the [AP](#) command to preserve the PA settings as the Power up/Reset values.



Example: Turn on the Y axis auxiliary output at the beginning of a move and turn the T axis output off at the beginning of a move, while in the [AA](#) command mode. (Note - the reversed logic; i.e. 0 = on, 1 = off. "On" pulls the signal line to ground. "Off" lets it rise to 5 volts or its pull-up reference voltage.)

Enter: [AA](#);
PA,0,,1;

Response: None

NOTE: PA selects the mode immediately but places entries in the axis command queue to set the state of the aux bit to the idle state.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| PAb; | AX - AT | 1* | 0 |
| PAb,b,b,b; | AA-AM | 1* | 0 |
| - | AA/CD | Not Valid | |

*If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.

Related commands: [AF](#), [AN](#), [SE](#)

?PA**REPORT POWER AUTOMATIC
STATE**

The ?PA command reports whether the current axis has power automatic mode enabled. The [PA](#) command is used to set this value. If power automatic mode is enabled, it will report whether the aux bit goes high or low during a move. The axis is stationary if the aux bit is low.



Example: Determine if X axis power automatic is high or low.

Enter: [AX](#);
?PA

Response: pa0<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?PA | AX – AS | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

*Power automatic mode is enabled for X axis such that the aux bit is high during a move.

Related commands: [PA](#), [?SE](#)

PE REPORT ENCODER POSITIONS

The PE command reports the encoder positions of all encoder and/or servo axes. All encoder positions will be reported even in single axis mode. (This is the same as [AA;RE;](#))



Example: Report the encoder positions of a four axis servo board.

Enter: PE

Response: 0,50,156,0<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| PE | AX - AT | Immediate | |
| PE | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [PP](#), [RE](#), [RP](#)

PF LINEAR ON

The PF command restores all axes to linear acceleration and deceleration ramps. This command should not be given while an axis is in motion or the results may not be predictable. This command affects all axes even if issued in the single axis mode. PF is the factory default setting. See the [AP](#) command to restore the PF setting as the power up/reset mode.



Example: Turn off cosine or parabolic ramps, returning to linear.

Enter: PF;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| PF; | AX - AT | 5 | 6 |
| PF; | AA-AM | 5 | 6 |
| - | AA/CD | Not Valid | |

Related commands: [CN](#), [LA](#), [PN](#), [PR](#), [?RT](#), [SC](#), [SR](#)

PM**PRINT MACRO**

The PM command will return the command string stored in the specified macro number as a command response. The macro number entered as the argument for this command must be between 0 and 24.

RANGE: $0 \leq PM \leq 24$



Example: Print the command string contained in macro 19.

Enter: PM19;

Response: If a macro string is defined for macro 19, the macro string will be the response. If no macro is defined there will be no response.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| PM#; | AX - AT | Immediate | |
| PM#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

NOTE: Macros are stored as ASCII character strings. If <LF> character is used as command terminator it will be sent back to the host computer by the PM command. If the application software stops reading a character string first it will appear that the PM command did not return the macro contents. To avoid this issue, save macros without <LF> terminator. Use semi-colons instead to terminate commands

Related commands: [MD](#), [MX](#), [PT](#)

?PM REPORT HOLD STATE

The ?PM command reports whether the PID for the current servo axis is enabled.



Example: A limit switch was hit by servo axis Y. See if the PID is still enabled for that axis.

Enter: [AY](#);
 ?PM

Response: hf<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?PM | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [HF](#), [HN](#)

PN PARABOLIC ON

The PN command sets all axes to truncated parabolic ramps. This acceleration profile starts at 100% of the programmed acceleration and decreases in steps of 10% of the initial acceleration down to as low as 10%. The parameter supplied selects the number of steps. It must be in the range of 3 to 10 corresponding to 70% and 10% acceleration at the peak respectively. A parameter out of this range or no parameter supplied defaults to 70% or 3 steps. Note that the parameter is the number of steps, not the acceleration values. The larger number is a lower acceleration at the peak. This command should not be given while an axis is in motion or the results may not be predictable. This command affects all axes even if issued in the single axis mode. The [PF](#) command is used to return to the default linear motion profiles. See the [AP](#) command to preserve the PN setting as the Power up/Reset ramp.

See [PR](#) for single axis.

RANGE: $3 \leq \text{PN} \leq 10$



Example: Set the board to be in the smoothest parabolic acceleration ramp.

Enter: PN10;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|--|---------|----------|
| FORMAT | MODE | AXIS RAMP TYPE | COMMAND | ARGUMENT |
| PN#; | AX - AT | Short Form Parabolic (PN0 ;)) | 5 | 10 |
| PN#; | AX - AT | All Other Parabolic (PN , PR) | 5 | 24 |
| PN#,#,#,#; | AA-AM | Short Form Parabolic (PN0 ;)) | 5 | 10 |

Related commands: [CN](#), [LA](#), [PF](#), [PR](#), [?RT](#), [SC](#), [SR](#)

PP**REPORT MOTOR POSITIONS**

The PP command reports the motor positions of all axes in ASCII format. All axes will be reported even in single axis mode. (This is the same as [AA](#); [RP](#))



Example: Report the motor positions of a four axis controller.

Enter: PP;

Response: 0,0,0,125 <LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| PP; | AX - AT | Immediate | |
| PP; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [PE](#), [RE](#), [RP](#)

PR**PARABOLIC RAMP PER AXIS**

The PR command defines parabolic acceleration/deceleration ramps for use with one or more axes. This command is similar to the [PN](#) command except that only the specified axes are affected. The [AP](#) command can be used to store the settings of PR as the power-up/reset defaults.

RANGE: $3 \leq PR \leq 10$



Example: Select a 10 step parabolic ramp for the T axis.

Enter: [AT](#);
PR10;

Response: None



Example: Select a 10 step parabolic ramp for the Y axis and an 8 step parabolic ramp for the T axis and the R axis.

Enter: [AA](#);
PR,10,,8,,,8;

Response: None

| QUEUE REQUIREMENTS | | | | |
|--------------------|---------|--|---------|----------|
| FORMAT | MODE | AXIS RAMP TYPE | COMMAND | ARGUMENT |
| PR#; | AX - AT | Short Form Parabolic (PN 0;) | 5 | 10 |
| PR#; | AX - AT | All Other Parabolic (PN ,PR) | 5 | 24 |
| PR#,#,#,; | AA-AM | Short Form Parabolic (PN 0;) | 5 | 10 |
| PR#,#,#,; | AA-AM | All Other Parabolic (PN ,PR0) | 5 | 24 |
| - | AA/CD | Not Valid | | |

Related commands: [CN](#), [LA](#), [PE](#), [PN](#), [?RT](#), [SC](#), [SR](#)

PS REPORT MACRO LINK



This command reports the macro link or KILL ([KL](#)) function link to the specified input bit and bit state.

First Parameter

This specifies the standard input bit number. The factory default bits are 0, 1, 2, and 3, but it can be configured by the user to be the I/O bits 0 to 7

Second Parameter

Valid bit states are 0 and 1

If the value of the selected bit state is ZERO, the selected macro will be executed if the selected bit changes from a TTL high to a TTL low.

If the bit state is ONE, then the selected macro will be executed when the selected bit changes from a TTL low to a TTL high.

NOTE: Each bit and state can be linked with a macro. So, up to two macros can be assigned to an input bit. For example, macro 15 could be executed when I/O 20 goes low, and macro 16 could be executed when I/O 20 goes high.

Output Formats:

If the bit is linked to the execution of a macro, the text response is the text of the [SX](#) command used to link the bit with the macro



Example: <LF><CR>[SX](#)1,0,23;<LF><CR>

If the bit is linked to the [KL](#) (KILL) function, the output will be the text response of the [SK](#) command used to link the bit with the kill function



Example: <LF><CR>[SK](#)1,0,1;<LF><CR>

If the bit is not linked the output will be: <LF><CR><LF><CR>



| | |
|-----------|---|
| Example: | This will report if there are any macro or KILL (KL) function links to input bit 2 when it goes from low to high. |
| Enter: | PS2,1; |
| Response: | None. |

PT PRESERVE A TEMPORARY MACRO



Use PT to save a temporary macro permanently by copying it to non-volatile memory. The temporary macro number, which is entered as an argument for this command, must be between 0 and 4. The non-volatile macro number, which is also entered as an argument for this command, must be between 5 and 24.

RANGE:

$0 \leq \text{Parameter 1} \leq 4$

$5 \leq \text{Parameter 2} \leq 24$



Example: Copy temporary macro 3 to non-volatile macro 19.

Enter: PT3,19;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|-----|
| FORMAT | MODE | Min | Max |
| PT#,#; | AX – AT | Immediate | |
| PT#,#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [MD](#), [MX](#), [PM](#)

QA QUERY AXIS STATUS



The QA command returns the status of the single addressed axis like the [RA](#) command except the limit and done flags will not be cleared. Refer to the [RA](#) command for details.



Example: Check the status of the X axis.

Enter: [AX](#);
QA

Response: PNNH<LF>

Refer to the table "Character Meaning" in the [RA](#) command on page 5-156.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| QA | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [EA](#), [QI](#), [RA](#), [RI](#)

QI QUERY INTERRUPT STATUS



The QI command returns the same information as the [QA](#) command but for all axes at once. The 4 character fields for each axis are separated by commas. The state of the status flags for all axes with out clearing the controller's copy of the done flags.



Example: Check the status of a four axis board.

Enter: [AA](#);
QI

Response: PNNN,MNNN,PDNN,MNLN<LF>

Refer to the table "Character Meaning" in the [RA](#) command on page 5-156.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Not Valid | |
| QI | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [EA](#), [QA](#), [RA](#), [RI](#)

QL QUERY ALL LIMIT SENSORS



QL is an immediate command and is valid in single axes, [AA](#), and [AM](#) modes of operation.

The response of the QL command is a hexadecimal value representing the state of the response of the positive and negative limit sensors of each axis. A limit sensor in the TTL “High” state will have a value of ONE (1) and a limit sensor in a TTL “Low” state will have value of Zero (0). The order of the limit sensors is as follows:

| | | | |
|--------|-----------------------|--------|-----------------------|
| 0xf8f0 | T Axis Positive Limit | 0xf0f8 | T Axis Negative Limit |
| 0xf4f0 | Z Axis Positive Limit | 0xf0f4 | Z Axis Negative Limit |
| 0xf2f0 | Y Axis Positive Limit | 0xf0f2 | Y Axis Negative Limit |
| 0xf1f0 | X Axis Positive Limit | 0xf0f1 | X Axis Negative Limit |



Example: Query all limit sensors

Enter: [AA](#);
QL

Response: ffff<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| QL | AX - AT | Immediate | |
| - | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [LH](#), [LL](#), [?LS](#)

RA REQUEST AXIS STATUS

The RA command returns the state of the limit and home switches and the done and direction flags for the currently addressed axis. The done flag register will be reset by this command.

See [RI](#) command for multi-axis mode.

The status is returned in the following format:

| CHARACTER MEANING | | |
|-------------------|------|---|
| CHAR | SENT | DESCRIPTION |
| 1 | LF | Line feed |
| 2 | CR | Carriage return |
| 3 | CR | Carriage return |
| 4 | P | Moving in positive direction |
| | M | Moving in negative direction |
| 5 | D | Done (ID , II or IN command has been executed, set to N by this command or IC command) |
| | N | No ID executed yet |
| 6 | L | Axis in overtravel. Char 4 tells which direction. Set to N when limit switch is not active. |
| | N | Not in overtravel in this direction |
| 7 | H | Home switch active. Set to N when home switch is not active. |
| | N | Home switch not active |
| 8 | LF | Line feed |
| 9 | CR | Carriage return |
| 10 | CR | Carriage return |



Example: The Y axis just encountered a limit, verify its status.

Enter: [AY](#);
RA

Response: PNLN<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RA | AX – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [EA](#), [QA](#), [QI](#), [RI](#)

RB REPORT BIT DIRECTION

The RB command reports the bit direction of the general I/O bits.



Example: Report direction of all I/O's. In this example I/O bits 0-3 are set as outputs and I/O bits 4-7 are set as inputs.

Enter: RB

Response: 0f<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Immediate | |
| RB | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: None

RC REQUEST ACCELERATION

The RC command will return the current acceleration rate of the current axis. This may differ from the programmed acceleration if a cosine ([CN](#)) or parabolic ([PN](#)) ramp is being generated. When the stage is stopped, the parameter returned will be zero (0). When the stage is running at programmed speed; i.e. not accelerating, the parameter returned will be zero (0). While a contour is executing, the value computed to generate the appropriate lead in will be returned. The response to the RC command is surrounded by linefeed + carriage return pairs.



Example: Display current acceleration values for all axes on a four axis board.

Enter: [AA](#);
RC

Response: <LF><CR>2000000, 2000000, 2000000, 2000000<LF><CR>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RC | AX - AT | Immediate | |
| RC | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [?AC](#), [AC](#), [RV](#)

RD RESTORE DEFAULT VALUES

RD assigns the current parameter set to be the default values that are in flash memory.

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | Immediate | | |
| AA - AM | Immediate | | |
| AA/CD | Not Valid | | |



Example: Assign the current parameter set to be the default values.

Enter: RD;

Response: None.

Related commands: [AP](#), [RF](#)

RE REPORT ENCODER POSITION

The RE command returns the current encoder position of the currently addressed axis or axes in encoder counts.



Example: Examine the current encoder position of the Y axis.

Enter: [AY](#);
RE

Response: 12345<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RE | AX - AT | Immediate | |
| RE | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [PE](#), [PP](#), [RP](#)

RF RESTORE FACTORY DEFAULT VALUES



RF assigns the current parameter set to be the factory default values.



Example: Assign the current parameter set to be the factory default values.

Enter: RF;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | Immediate | | |
| AA - AM | Immediate | | |
| AA/CD | Not Valid | | |

Related commands: None

RI REQUEST INTERRUPT STATUS



The RI command is a multi-axis mode command that returns the same status information for all axes as the [RA](#) command does in single axis mode. The 4 character fields for each axis are separated by commas. The done flag is reset by this command as it would be via the [RA](#) command.



Example: Check the status of a 4 axis board.

Enter: [AA](#);
RI

Response: MDNN,MDNN,PNLN,PNNN<LF>

Refer to the table "Character Meaning" in the [RA](#) command on page 5-156.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AX - AT | Not Valid | |
| RI | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [EA](#), [QA](#), [QI](#), [RA](#)

RL REPORT SLIP STATUS



The RL command returns the slip detection status of all axes. S is returned if a slip condition has occurred for that axis, or else an N is returned. The number of characters returned corresponds to the number of axes available on the board. This command is intended to be used with stepper motors with encoders and not with servo motors. Open-loop stepper always returns “n” and servo axes always returns “N” in their RL response.



Example: On a four axis board, see if any axis has slipped.

Enter: RL

Response: NNSN<LF> (The Z axis has slipped.)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RL | AX - AT | Immediate | |
| RL | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [ES](#), [IS](#), [TF](#), [TN](#)

RM**REMAINDER**

The RM command divides the position counter by the parameter supplied and replaces the position counter with the resulting remainder. The parameter must be greater than zero and less than 32,000. This command is used in applications where the controller is managing the motion of a continuously rotating object. It allows the position counter to keep track of the absolute position without regard to the number of revolutions it may have rotated. This command has the same effect on the encoder position register on axes with the encoder feedback enabled.

RANGE: $0 < RM < 32,000$



Example: The current position of a rotating stage with a full-revolution count of 6000 is needed. Since this stage has been rotated several times without regard for the position, the position counter has reached 163,279. Send an RM6000 command to find out what the real position of the axis is.

Enter: (Current position is 163,279)
RM6000;
(Current position is now 1,279)

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RM#; | AX - AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [LO](#), [LP](#), [RE](#), [RP](#), [RU](#)

RP**REQUEST POSITION**

The RP command returns the current position of the currently addressed axis in single axis mode or all positions separated by commas in [AA](#) or [AM](#) mode. The position will be returned to the host in ASCII format. This command is not queued; i.e. the current position will be returned immediately even if the axis is in motion.



Example: The current position on the Y axis is 12345. Use the RP command to verify the position.

Enter: [AY](#);
RP

Response: 12345<LF>



Example: Verify the positions of all axes.

Enter: [AA](#);
RP

Response: 100,200,300,400<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RP | AX - AT | Immediate | |
| RP | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [RE](#), [PE](#), [PP](#)

RQ REPORT QUEUE SIZE

The RQ command returns the number of entries available in the queue of the currently addressed axis in single axis mode or all axes separated by commas in multi-axis modes. The ASCII string is delimited by linefeed + carriage return pairs. The maximum available in each command queue is 800 except the contour queue which is 7160. The response is at a fixed length of 3 characters for all modes except contour mode which is fixed at 4 characters. For example, if the current free queue space is 67 in AM mode, the response from the controller to the RQ command is <LF><CR>067<LF><CR>.



Example: Report the command queue space remaining in contour mode.

Enter: RQ

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|-----------|
| FORMAT | MODE | COMMAND | MAX Value |
| RQ | AX - AT | Immediate | 200 |
| RQ | AA-AM | Immediate | 200 |
| RQ | AA/CD | Immediate | 7160 |

Related commands: None.

RS RESET

The RS command is a software reset which causes the local UMX microprocessor to reset. All previously entered data and commands are lost. All internal parameters are initialized to defaults. All interrupts are disabled. This command is intended for catastrophic failure recovery only. The [KL](#) command should be used to reset queues or return the system to a known state. Monitor the INIT flag in the status register for completion of the initialization process; see Table 3-4. The "Initializing" in process bit goes high during the initialization process. You should not try to communicate to the controller for approximately 5 seconds after issuing the RS command.



Example: Clear everything in the board and stop all movement. Reset all hardware registers.

Enter: RS;

Response: None

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AY – AT | Immediate | | |
| AA-AM | Immediate | | |
| AA/CD | Not Valid | | |

Note: The Polling for the Dir Flag should be at a frequency of 1500ms or more.

Related commands: None.

?RT REPORT RAMP TYPE

The ?RT command reports the current acceleration ramp assigned to the active axis. Possible responses are:

| | |
|-----|--|
| la | Default linear ramp |
| prn | Parabolic where n specifies number of segments |
| sc | Cosine ramp |
| srn | Custom ramp where n specifies the table number |
| pr | Parabolic ramps |



Example: Make sure custom ramp #3 was assigned to the Y axis

Enter: [AY](#);
 ?RT

Response: sr3<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?RT | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [CN](#), [LA](#), [PF](#), [PN](#), [PR](#), [SC](#), [SR](#)

RU REPORT POSITION IN USER UNITS



The RU command returns the current position in user units (see [UU](#) command on page 5-195). The format of the response is a floating-point number.



Example: One revolution of a motor is 2000 steps. Define user units so moves can be referenced in revolutions. Move the Z axis 3 1/2 revolutions. Use RU to display the position when the move is complete.

Enter: [AZ](#);
[UU](#)2000;
[LP](#)0;
[MR](#)3.5;
[GO](#); (Wait until move is complete.)
 RU;

Response: 3.50000<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RU; | AX - AT | Immediate | |
| RU; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [RE](#), [RP](#), [UU](#)

NOTE: While the user units' mode provides a certain level of convenience to the user, it does so at a cost, namely accuracy and control. User unit conversions may cause round off error and may possibly truncate key information.

RV REQUEST VELOCITY

The RV command will return the current velocity at which the axis is moving. This may differ from the programmed maximum velocity if the axis is accelerating or decelerate. If the [JF](#) command is executing, the command only reports the integer part of the velocity.



Example: Jog the Y axis at 12345 steps per second.
Display the current velocity.

Enter: [AY](#);
[JG](#)12345;
RV

Response: 12345<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| RV | AX - AT | Immediate | |
| RV | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [RC](#), [?VL](#), [VL](#)

SA**STOP ALL**

The SA command flushes all queues and causes all axes to decelerate to a stop at the rate previously specified in an [AC](#) command. All status and position information is retained. Even when executed in a single axis mode, this command will cause all axes to stop.



Example: Send all axes on a move and then ramp them to a stop before they finish.

Enter: [AA](#);
[VL](#)100,100,100,100;
[MR](#)1000,2000,3000,4000;
[GO](#);(wait awhile)
 SA;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SA; | AX - AT | 1* | 0* |
| SA; | AA-AM | 1* | 0* |
| - | AA/CD | Not Valid | |

Related commands: [KL](#), [KS](#), [SD](#), [SI](#), [SO](#), [ST](#)

SB SET BAUD RATE

The SB command sets the controller's serial communications baud rate. Valid baud rates include 300, 600, 1200, 2400, 9600, 19200 and 38400. The factory default baud rate is 9600. See the [AP](#) Command to preserve the SB your baud rate as the Power up/Reset rate.

NOTE: The UMX uses CTS/DTR hardware handshake for flow control.



Example: Sets the serial communications baud rate to 19200.

Enter: SB19200;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AY – AT | Immediate | | |
| AA - AM | Immediate | | |
| AA/CD | Not Valid | | |

Related commands: [?SB](#)

?SB QUERY THE BAUD RATE

The ?SB command queries the board to determine the current baud rate setting



Example: Query the board to determine its current baud rate setting.

Enter: ?SB

Response: If the board is set for 9600 baud rate: sb9600<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|-----------|-------------|-------------|
| MODE | Min (pf) | Max (pn/cn) | Custom ramp |
| AX - AT | Immediate | | |
| AA - AM | Not Valid | | |
| AA/CD | Not Valid | | |

Related commands: [SB](#)

SC COSINE RAMP PER AXIS

The SC command specifies that the standard cosine acceleration ramp is to be used by the selected axis/axes. This command is similar to the [CN](#) command except that only the selected axis or axes are affected. The [AP](#) command will store the settings of SC as the power-up/reset defaults.



Example: Select the cosine ramp for the X axis.

Enter: [AX](#);
SC;

Response: None.



Example: Select the cosine ramp for the Y and T axes.

Enter: [AA](#);
SC,1,,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SC; | AY – AT | 5 | 24 |
| SCb,b,b,b; | AA-AM | 5 | 24 |
| - | AA/CD | Not Valid | |

Related commands: [CN](#), [LA](#), [PE](#), [PN](#), [PR](#), [?RT](#), [SR](#)

SD**STOP AND RESET DONE**

The SD command may be substituted for the [SA](#) command. It will reset the done flags for all axes, stop all axes at the rates previously specified via the [AC](#) command, and then flush all axis command queues. This allows the host to be interrupted when all axes have stopped by using the [ID](#) command after the SD. The [SA ID](#) combination may flag the completion early if one of the axes is already done from a previously executed [ID](#).



Example: Stop all axes and reset all done flags. When all axes have stopped set all done flags.

Enter: [AA](#);
SD;
[ID](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SD; | AY – AT | 2* | 0* |
| SD; | AA-AM | 2* | 0* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.

Related commands: [KL](#), [KS](#), [SA](#), [SI](#), [SO](#), [ST](#)

SE SETTLING TIME

The SE command allows specification of a settling time, in milliseconds, to be used before the auxiliary output is complemented when using [PA](#) mode. The parameter may be any value up to 1000 milliseconds. Specification of a parameter of zero disables SE mode.

The factory default settling time is zero. See the [AP](#) command to preserve the SE settings as the Power up/Reset values.

RANGE: $0 \leq SE \leq 1000$



Example: Turn on the Z axis auxiliary output upon execution of a move and have it remain on for 500 milliseconds after the move is complete.

Enter: [AZ](#);
[PA](#);
 SE500;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SE#; | AY – AT | Immediate | |
| SE#,#,#,#; | AA - AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [AF](#), [AN](#), [PA](#), [?SE](#)

?SE REPORT SETTLING TIME

The ?SE command reports the settling time setting ([SE](#)) used with power automatic mode ([PA](#)) for the current axis.



Example: Report the current settling time for axis X

Enter: [AX](#);
 ?SE

Response: se250<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?SE | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?PA](#), [SE](#)

SF**SOFT LIMITS OFF**

The SF command restores the default operation of the limit switches; i.e. causes the affected axis or axes to abruptly halt when a limit switch is encountered. If soft limits have been made the power-up default, the [AP](#) command must be used to store hard limit operation as the default.



Example: Set up a board to make the X axis stop immediately when a limit is encountered.

Enter: [AX](#);
SF;

Response: None.



Example: Set up a board to make the Y and T axes to stop immediately when a limit is encountered.

Enter: [AA](#);
SF,1,,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SF; | AX - AT | 1 | 0 |
| SFb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LH](#), [LL](#), [LN](#), [?SL](#), [SL](#), [TL](#)

SI STOP INDIVIDUAL



This command can be used to stop only certain axes. In a single axis mode, the SI command behaves identically to [ST](#). In a multi-axis mode, however, SI can be used to stop any number of axes and can be used in place of [SA](#). Like [SA](#), SI will ramp those axes to be stopped using the rate previously specified via the [AC](#) command. This command is useful for stopping a specific axis when the current axis mode is unknown and for stopping several axes without affecting current motion on other axes.

Each parameter represents an axis from X through S. Any non-zero value in a parameter will cause the corresponding axis to be stopped.



Example: Start a motion on all four axes. When input bit 1 becomes true, stop axes Y and T without affecting X and Z.

Enter: [AM](#);
[MR](#)15000,30000,20000,40000;
[GO](#);
[SW](#)1;
 SI,1,,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SI; | AX - AT | 1* | 0* |
| SIb,b,b,b; | AA-AM | 1* | 0* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.

Related commands: [KL](#), [KS](#), [SA](#), [SD](#), [SO](#), [ST](#)

SK**DEFINE KILL LINK ON UMX**

The SK command links the execution of the KILL function ([KL](#)) to the defined change of state of the assigned input bit.

First Parameter

This specifies the standard input bit number. The factory default bits are 0, 1, 2, and 3, but it can be configured by the user to include bits 0 to 7

Second Parameter

Valid bit states are 0 and 1

If the value of the selected bit state is ZERO, the selected macro will be executed if the selected bit changes from a TTL high to a TTL low.

If the value of <Bit State> is ONE, then the selected Macro will be executed when the selected bit changes from a TTL low to a TTL high.

NOTE: Each bit state can be linked with a macro. So, up to two macros can be assigned to an input bit. For example, macro 10 could be executed when I/O 0 goes low and macro 11 could be executed when I/O 0 goes high.

Third Parameter

If the value is ZERO the KILL function linkage for the specified Bit State is deleted.

If the value is ONE, the KILL function linkage for the specified Bit State will replace the current linkage.



Example: The current macro linked to I/O bit 2 as it goes from high to low is to be replaced with the KILL ([KL](#)) function.

Enter: SK2,0,1;

Response: None.

Related commands: None.

SL SOFT LIMITS ON

The SL command enables the UMX to ramp an axis to a stop rather than abruptly killing the motion when a limit switch is encountered on that axis. The output queue is not flushed except for the current move. This mode is effective for point to point and [JG](#) moves only. Soft limits can be made the power-up default via the [AP](#) command.



Example: Set up a board to allow the X axis to ramp to a stop when a limit is encountered.

Enter: [AX](#);
[SL](#);

Response: None.



Example: Set up a board I/O to allow the Y and T axes to ramp to a stop when a limit is encountered.

Enter: [AA](#);
[SL](#),1,,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SL; | AX - AT | 1 | 0 |
| SLb,b,b,b; | AA-AM | 1 | 0 |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LH](#), [LL](#), [LN](#), [SE](#), [?SL](#), [TL](#)

?SL REPORT SOFT LIMIT STATUS

The ?SL command reports whether soft limits are currently enabled for the active axis.



Example: Find out whether soft limits are enabled for axis Z

Enter: [AZ](#);
?SL

Response: sl<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?SL | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [SF](#), [SL](#)

**SM ENABLE/DISABLE
STAND-ALONE MODE**

This command enables or disables the SM mode (Stand-Alone Mode).

SM mode value of 1 enables the SM mode

A SM mode value of 0 disables the SM mode.



Example: Enable Stand-Alone Mode

Enter: SM1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SM#; | AX - AT | Immediate | |
| SM#,#,#,#; | AA-AM | Immediate | |
| SM#,#,#,#; | AA/CD | Immediate | |

Related commands: [CB](#), [PS](#), [SK](#), [SX](#)

SO STOP AT POSITION BY RAMPING FROM A DISTANCE



The SO command instructs the UMX to continue moving until reaching a specified distance (parameter 2) from a specified stop point (parameter 1). The axis will then ramp to a stop within the specified distance. This allows the user to control the point at which deceleration begins, the rate of deceleration, and the stop point, all with a single command.

RANGE:

Min. Position Range ≤ Parameter 1 (Stop Position) ≤ Max. Position Range
Min. Position Range ≤ Parameter 2 (Distance from Stop Position to Start Decelerating) ≤ Max. Position Range

NOTE: The position range is dependent on the UMX's settings, see Specifications for more detail.



Example: The X axis is jogging at 10,000 steps per second. We want the axis to stop at position 50,000 but it must not start ramping until reaching position 46,000.

Enter: SO50000,4000;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SO#,#; | AY – AT | 3* | 2* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis, add 1 to the command queue
- If [PA](#) (power automatic) mode is active, add 1 to the command queue
- If an aux bit settling time has been specified, add 2 to the command queue and add 1 to the argument queue

Related commands: [KL](#), [KS](#), [SA](#), [SD](#), [SI](#), [ST](#)

?SO REPORT ANALOG OUTPUT MODE



The ?SO command reports whether the analog output type for the current servo axis is bipolar or unipolar. The possible responses are [BI](#) and [UN](#), the same commands used to set one mode or the other.



Example: The Y axis should be setup with unipolar outputs. Use ?SO to make sure.

Enter: [AY](#);
?SO

Response: un<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?SO | AX - AT | Immediate | |
| - | AA - AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [UN](#)

SP STOP AT POSITION



The SP command will cause the controller to attempt to stop at the specified destination. If there is insufficient distance to stop at the previously specified deceleration when the command is received, the controller will stop as soon as possible at that deceleration. This command is not compatible with the [JG](#) command.



Example: (see [MV](#) command on page 5-141)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SP#; | AY – AT | 2* | 1* |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

- If the axis is a stepper and encoder or servo axis add 1 to the command queue
- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue

Related commands: [FP](#), [MM](#), [MP](#), [MV](#)

SR SELECT CUSTOM RAMP

The SR command selects a previously defined custom ramp profile for use with a currently active axis. This command will override previous ramp type selection for the given axis such as [PN](#) and [CN](#).

RANGE: $1 \leq \text{SR} \leq 8$



Example: Select custom ramp number 4 for use with axis Y.

Enter: [AY](#);
 SR4;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|--|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SR#; | AY – AT | 15 | 5 + (2* number of segments in ramp definition) |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [CN](#), [DAB](#), [DAE](#), [DAR](#), [LA](#), [PE](#), [PN](#), [PR](#), [?RT](#), [SC](#)

ST STOP

The ST command flushes the queue for the current axis or axes only and causes the axis/axes to decelerate to a stop at the rate previously specified via the [AC](#) command. This command is used to stop one or more motors in a controlled manner from jog mode or an unfinished [GO](#) or [GD](#) command. This command is executed immediately upon receipt. All status and position information is retained. When executed in a multi-axis mode, the ST command is equivalent to the [SA](#) command.



Example: Move the Y axis for a while at 1200 steps/second and then ramp to a stop.

Enter: [AY](#);
[JG](#)1200;
 (Wait awhile)
 ST;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ST; | AX - AT | 1* | 0* |
| ST; | AA-AM | 9* | 61* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue.
- If an auxiliary output bit settle time has been specified add 2 to the command queue and add 1 to the argument queue.

Related commands: [KL](#), [KS](#), [SA](#), [SD](#), [SI](#), [SO](#)

?SV REPORT SERVO VOLTAGE INVERSION STATE



The ?SV command reports the current logical direction for the current servo axis. The state is set with the [SVI](#) and [SVN](#) commands. The possible responses to the ?SV command are svn for normal and svi for inverted.



Example: Report whether servo voltage is positive for positive moves on axis X

Enter: [AX](#);
?SV

Response: svn <LF> (voltage is normal; i.e. positive for positive moves)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?SV | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?DB](#), [SVI](#), [SVN](#)

SVI INVERT SERVO VOLTAGE



The SVI command inverts the voltage output for the current axis. After receiving this command, the UMX will produce a negative voltage for positive motion and a positive voltage for negative motion. To cancel this command, issue an [SVN](#) command. To make inverted servo outputs the default at power up or reset, use the [AP](#) command.



Example: The Y axis encoder is counting opposite the expected direction. Setup the Y axis to produce a negative voltage when moving positive instead of a positive voltage to correct the problem.

Enter: [AY](#);
SVI;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SVI; | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [DBI](#), [DBN](#), [?SV](#), [SVN](#), [UN](#)

SVN NORMALIZE SERVO VOLTAGE



The SVN command normalizes the voltage output for the current axis, negating the effects of the [SVI](#) command. After receiving this command, the UMX will produce a positive voltage for positive motion and a negative voltage for negative motion, the default behavior. To make this the default behavior (if it has been changed via [SVI/AP](#)), use the [AP](#) command. (SVN is the factory default setting.)



Example: The Y axis encoder was rewired and now counts in the correct direction. Return the Y axis servo output to normal; i.e. output positive voltage for positive motion.

Enter: AY;
SVN;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SVN; | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [DBI](#), [DBN](#), [?SV](#), [SVI](#), [UN](#)

SW SYNC WAIT

The SW command allows synchronization of multi-axis moves or other tasks on one or more UMX boards by using one of the general purpose input lines. This command causes the UMX to stop processing new commands until the general purpose input line has been released (allowed to go high) before proceeding with the next command.

The SW command can also be used to cause an axis to wait until the others are finished. To do this, wire-OR the auxiliary lines from several axes together and connect them to a general purpose input line. Use power automatic ([PA](#)) mode with the SW command on that line. All commands after that will wait until all axes release their auxiliary lines; i.e. come to a complete stop.

RANGE: $0 \leq \text{Bit Number} \leq 7$

NOTE: The parameter used to specify Bit Number must be configured as an input. See, [RB](#) and [IO](#).



Example: The following command sequence will cause the X axis move to wait until the Y axis has finished its move and turned off its auxiliary output which has been wired to the general purpose input 0 line.

Enter: [AY](#);
[AN](#);
[MR](#)2000;
[GO](#);
[AF](#);
[AX](#);
 SW0;
[MR](#)10000;
[GO](#);

Response: None.

The SW command provides a way to synchronize moves on two or more controllers. The following example shows one way to do this.



Example: You have 3 four axis controllers, for a total of 12 axes to move together. Call board 1 the “master” and boards 2 and 3 the “slaves”. Wire board 1’s X axis auxiliary line to the two slave boards’ general purpose input 0 line. Send to the master the command “[AX PA](#)0”, setting the master’s X axis auxiliary line low until its move starts. This also sets the slaves’ general purpose input 0 line low. Enter the “SW0” command to the two slaves, followed by the move and [GO](#) commands. On the master, enter the move command, followed by the [GO](#) command. When the master’s move starts, the [PA](#) command will set the auxiliary line high releasing the wait on the slave boards. All three boards will start their moves. This provides synchronization to within 480us of each board.

Procedure: Wire board 1’s X axis auxiliary line to board 2’s and board 3’s general purpose input 0 line.

Enter: (Board 1) [AX](#);
[PA](#)0;
 (Board 2) [AA](#);
 SW0;
[MR](#)200,200,200,200;
[GO](#);
 (Board 3) [AA](#);
 SW0;
[MR](#)300,300,300,300;
[GO](#);
 (Board 1) [AA](#);
[MR](#)100,100,100,100;
[GO](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| SW#; | AX - AT | 1 | 1 |
| SW#; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [BW](#), [WA](#), [WQ](#), [WT](#)

SX DEFINE MACRO LINK ON UMX



This command creates the link to execute the previously defined permanent macro when it senses the change in state of the defined standard input bit.

First Parameter

This specifies the standard input bit number. The factory default bits are 0, 1, 2, and 3, but it can be configured by the user to include bits 0 to 7.

Second Parameter

Valid bit states are 0 and 1

If the value of the selected bit state is ZERO, the selected macro will be executed if the selected bit changes from a TTL high to a TTL low.

If the value of <Bit State> is ONE, then the selected Macro will be executed when the selected bit changes from a TTL low to a TTL high.

NOTE: Each bit state can be linked with a macro. So, up to two macros can be assigned to an input bit. For example, macro 10 could be executed when I/O 0 goes low and macro 11 could be executed when I/O 0 goes high.

Third Parameter

Specifies the macro number (5-24) to be executed when the conditions are met. A macro link can be deleted by specifying the bit number and the bit state along with a macro number zero. The macro assigned to a given input bit and bit state can be changed by issuing the same SX command using a different macro number.



Example: When I/O bit 1 goes from low to high, macro 20 will be executed.

Enter: SX1,1,20;

Example: Upon Power-up, the user presses an "Activate Button-short to ground", (linked to bit 1). This should cause the UMX to position the X-axis at a position 1000 steps from zero and the Y-axis at 2000 steps from zero. At that point, it should execute Macro 10. The stop button (short to ground) is linked to bit 2.

Enter: MD0; begin the definition of Macro #0

AX;

MA1000; move X-axis to position 1000

GO;

AY;

MA2000; move Y-axis to position 2000

GO;

<control Z> terminate the definition of macro #0

PT0,10; store macro #0 to non-volatile macro #10

SX1,0,10; define I/O bit #1, active low will execute macro #10

MD1; begin the definition of macro #1

KL;

send a KILL to stop everything

<control Z> terminate the definition of Macro #1

PT1,9; store macro #1 to non-volatile macro #9

SX2,0,9; define I/O bit #2, active low will execute macro #9

SM1; enable the stand-alone mode

AP;

set the current parameters to the power-up defaults

Related commands: [CB](#), [PS](#), [SK](#), [SM](#)

TF TURN OFF SLIP KILL MODE

The TF command disables slip kill mode (enabled with [TN](#).)



Example: Slip kill mode is enabled but a move needs to be performed where slip is likely and not important for this move. Disable slip kill mode.

Enter: TF;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| TF; | AY – AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ES](#), [IS](#), [RL](#) [TN](#)

TL**SET SOFTWARE OVERTRAVEL
LIMITS**

The TL command sets logical limits on the range of travel for an axis. Two parameters must be supplied; one for the upper travel limit and the other for the lower travel limit, both as absolute positions. If the axis reaches either of these logical limits, the UMX will flag a limit condition just as it would be using the physical limit switch inputs. Move Absolute ([MA](#)), Move Relative ([MR](#)), and Jog ([JF](#),[JG](#)) commands are subject to software travel limits because the UMX checks an internal absolute position register.

Note: Software overtravel limits and physical limit switch inputs can be enabled at the same time.

RANGE: + Position Range

Note: The position range on UMX is dependent on the adder rate and update rate selection. See [Specification](#) section for more detail.



Example: Set logical position limits for the X axis of +/-1,000,000.

Enter: [AX](#);
TL1000000,-1000000;

Response: None.

Note: TL0,0; Turns software limits off.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| TL#,#; | AY – AT | 1 | 2 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [LF](#), [LL](#), [LH](#), [LN](#), [SF](#), [SL](#), [?TL](#)

?TL**REPORT SOFTWARE
OVERTRAVEL LIMITS**

The ?TL command reports the software travel limits for the current axis assigned via the [TL](#) command. The first value returned is the upper (or "positive") limit and the second value is the lower (or "negative") limit. These are not physical limits but rather positional limits that an axis should not exceed. If one of these limits is exceeded, the UMX will set the current axis' limit flag and notify the host computer of the condition as though the axis encountered a hard limit.



Example: Find out what the software limits of the Y axis are currently set to.

Enter: [AY](#);
?TL;

Response: tf<LF>



Example: Find out what the software limits of the T axis are currently set to.

Enter: [AT](#);
?TL;

Response: tf<LF> (software limits for axis T are currently disabled)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?TL | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [TL](#)

TM**TIMED JOG**

The TM command performs a jog at the current velocity limits defined for the axis/axes for the specified number of milliseconds. In multi-axis mode, all axes begin moving at the same time and ramp to a stop when their respective jog times have elapsed. The overall jog time will be the parameter passed to the TM command plus deceleration time and acceleration time.

RANGE: $0 \leq TM \leq 200,000$



Example: Jog the X axis for 1000 milliseconds.

Enter: [AX](#);
TM1000;

Response: None.



Example: Jog the X axis for 1000 milliseconds and the Z axis for 2000 milliseconds, starting both at the same time.

Enter: [AA](#);
TM1000,,2000;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| TM#; | AX - AT | 5* | 1* |
| TM#,#,#,#; | AA-AM | 5* | 1* |
| - | AA/CD | Not Valid | |

- If [PA](#) (power automatic) mode is active add 1 to the command queue
- If the axis is a stepper and encoder or servo axis add 1 to the command queue

Related commands: [JF](#), [JG](#), [SA](#), [ST](#)

TN TURN ON SLIP KILL MODE



The TN command enables slip kill mode. In this mode, if the motor slips beyond the dead band set by the [ES](#) command, the UMX will kill motion on the axis that slipped as though a [KL](#) command had been issued to the axis. This mode can be disabled (default) with the [TF](#) command.



Example: X axis is sent on a move. Its encoder cable was not connected to the controller (oops!). The controller issues a [KL](#) (Kill) command to the X axis after receiving the slip interrupt. The slip interrupt is generated once the difference between the motor position counts and encoder counts exceed 20.

Enter: [AX](#);
[ES](#)20;
 TN;
[IS](#);
[LP](#)0;
[MA](#)30;
[GO](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| TN; | AY – AT | 1 | 1 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [ES](#), [IS](#), [RL](#), [TF](#)

TX TRACK THE X AXIS



Set up the current axis so that it tracks the X axis. The command argument is the tracking ratio specified as a floating point.

NOTE: If this is a negative, then the axis moves in the opposite direction of the X axis. The command is invalid for the X axis.



Example: Set the Z axis to track the X axis.

Enter: [AZ](#);
TX;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| TX; | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related Commands: [ET](#), [HF](#)

UF**USER UNITS OFF**

The UF command turns off user units returning all numeric commands and responses to their factory default raw representations. This command is equivalent to and preferred over [UU1](#); since it turns off the mode thus minimizing unnecessary overhead. See the [AP](#) commands to preserve the UF settings to flash memory.



Example: Turn off user unit conversion on the X, Y, and Z axes.

Enter: [AX](#);
 UF;
 [AY](#);
 UF;
 [AZ](#);
 UF;
 Or
 [AA](#);

UF1,1,1;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| UF; | AX - AT | Immediate | |
| UFb,b,b,b; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [?UU](#), [UU](#)

NOTE: While the user units' mode provides a certain level of convenience to the user, it does so at a cost, namely accuracy and control. User unit conversions may cause round off error and may possibly truncate key information.

UN UNIPOLAR

The UN command sets the analog torque outputs of servo axes to unipolar. The analog output will range between 0.0VDC and +10VDC when unipolar is enabled. At maximum positive velocity, the board outputs +10VDC. At maximum negative velocity, the board output approaches 0.0VDC. To maintain position the board outputs 5VDC. This command is valid only in single axis mode.



Example: Set up servo axis X for unipolar operation.

Enter: [AX;](#)
UN;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| UN; | AX - AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [BI](#), [DBI](#), [DBN](#), [?SO](#), [SVI](#), [SVN](#)

UU USER UNITS



The UU command converts all move velocities, distances, etc. to user specified units by multiplying by the specified parameter. The [UF](#) command is used to terminate this mode. Factory default is with this command off. See the [AP](#) command on to preserve the UU settings to flash memory.



Example: The motor, driver, and gear ratio you are using requires 10,000 steps to move one inch. Set up the X, Y, and Z axes so you can enter move information in inches.

Enter: [AX](#);
 UU10000;
[AY](#);
 UU10000;
[AZ](#);
 UU10000;
 Or
[AA](#);
 UU10000,10000,10000;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| UU#; | AY – AT | Immediate | |
| UU#,#,#,#; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [RU](#), [UF](#), [?UU](#)

NOTE: While the user units' mode provides a certain level of convenience to the user, it does so at a cost, namely accuracy and control. User unit conversions may cause round off error and may possibly truncate key information.

?UU**REPORT AXIS USER UNIT**

This command returns the current user units' multiplier as set via the [UU](#) command. Factory default is [UF](#).



Example: Make sure the [UU](#)512; command we sent earlier is still current. The command will return the [UU](#) value with six digits to the right of the decimal point. If the [UU](#) value exceeds six digits for the fractional value, the value will be rounded off to the sixth decimal place.

Enter: ?UU

Response: uu512.000000<LF>
If user units are turned off ([UF](#)) ?UU returns:
uf<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?UU | AY – AT | Immediate | |
| | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [UF](#), [UU](#)

NOTE: While the user units' mode provides a certain level of convenience to the user, it does so at a cost, namely accuracy and control. User unit conversions may cause round off error and may possibly truncate key information.

VB VELOCITY BASE



The VB command allows the acceleration ramp to start off at a specified velocity. This allows faster acceleration and the ability to pass through resonance quickly in some applications. The velocity jumps instantly to the specified velocity, and then accelerates as usual. The deceleration is the same in reverse. This mode is active only for linear ramps; it is ignored for cosine and parabolic ramps but not flagged as a command error. The parameter must be greater than zero and less than the programmed velocity, where the factory default is zero steps per second. This command is not valid with the [JG](#) command nor will it work in conjunction with the [DC](#) command. See the [AP](#) command to preserve the VB settings as the power-up/reset values.

If the [VL](#) command is used after the VB command and the velocity value set with [VL](#) is less than the previously set VB value, the initial velocity used at the start of a move will be the [VL](#) value minus one. This will result in a one-step acceleration ramp and must be taken into consideration in applications making use of the VB command.

RANGE: $0 \leq VB < \text{VL value}$



Example: In the single axis mode, set the Y axis velocity base to 200.

Enter: [AY](#);
VB200;

Response: None.



Example: In the [AA](#) mode, set the X and Y axes velocity bases to 200.

Enter: [AA](#);
VB200,200;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| VB#; | AY – AT | 1 | 1 |
| VB#,#,#,#; | AA-AM | 1 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [AC](#), [DC](#), [?VB](#), [VL](#)

?VB REPORT BASE VELOCITY

The ?VB command returns the base (starting) velocity setting for the current axis as set by the VB command. Note that the base velocity must be lower than the command velocity.



Example: The acceleration ramp should start at 0pps. Make sure we didn't leave it at some other value.

Enter: ?VB;

Response: vb1500<LF> (Oops! We forgot to set it back to zero)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?VB | AY – AT | Immediate | |
| | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [VB](#), [?VL](#)

VL VELOCITY



The VL command sets the maximum velocity register of one or more axes to the operands which follow the command. The factory default is 200,000 steps per second. See the [AP](#) command to preserve the VL settings as the power-up/reset values.

RANGE: $1 \leq VL \leq 1044000$



Example: In single axis mode, set the X axis velocity to 10,000 counts per second.

Enter: [AX](#);
VL10000;

Response: None.



Example: In the [AA](#) mode, set the peak velocities of the X and T axes to 5,000 and 50,000 respectively. Leave the other axes with their previous values.

Enter: [AA](#);
VL5000,,,50000;

Response: None.

| QUEUE REQUIREMENTS | | | | |
|--------------------|----------------|---|-----------|----------|
| Format | Mode | Axis ramp type | Command | Argument |
| VL#; or VL#,#,##; | AX-AT or AA-AM | Linear (LA or PE) | 1 | 1 |
| VL#; or VL#,#,##; | AX-AT or AA-AM | Short Parabolic (PN0 ; or PR0 ;)) | 2* | 26* |
| VL#; or VL#,#,##; | AX-AT or AA-AM | All other Parabolic Forms | Not Valid | |
| VL#; or VL#,#,##; | AX-AT or AA-AM | Cosine (CN or SC) | | |
| VL#; or VL#,#,##; | AX-AT or AA-AM | Custom Ramps (SR) | | |

Related commands: [AC](#), [DC](#), [?VB](#), [VB](#), [?VL](#)

?VL REPORT PEAK VELOCITY SETTING



The ?VL command returns the peak velocity setting for the current axis as set by the [VL](#) command.



Example: Make sure our "[AX;VL](#)50000;" command worked.

Enter: ?VL

Response: vl150000<LF>

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| ?VL | AY – AT | Immediate | |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [?VB](#), [VB](#), [VL](#)

VS VELOCITY STREAMING



The VS command will generate a pulse train without acceleration or deceleration at the rates specified using the X and Y axes. The parameters are time in 1 update rate second sample intervals, X velocity, and Y velocity. This is a slave mode and cannot be mixed or queued with other commands; i.e. no other motions involving the X and Y axes or [AA](#) mode may be currently in execution when this command is issued. [AX](#) mode is the only valid mode for use with this command. The VS command does not require a [GO](#) command to start the motion; motion begins immediately upon receipt of the complete VS command.

RANGES:

$1 \leq \text{Parameter 1 (Time in 1 update rate of a second)} \leq 200,000$

$0 \leq \text{Parameter 2 (X Axis Velocity)} \leq 1044000$

$0 \leq \text{Parameter 3 (Y Axis Velocity)} \leq 1044000$



Example: Create a stair step profile on the X and Y axes, with the X axis moving in the negative direction and the Y axis in the positive direction. Make each step last 1 second long and increase velocity by 1,000 steps/second, until a velocity of 3,000 steps/second is reached, then step down to 0 steps/second. (Example assumes an update rate of 1024.)

Enter: [AX](#);
VS1024,-1000,1000;
VS1024,-2000,2000;
VS1024,-3000,3000;
VS1024,-2000,2000;
VS1024,-1000,1000;
VS1,0,0;

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| VS#,#,#,; | AY – AT | 2 | 3 |
| - | AA-AM | Not Valid | |
| - | AA/CD | Not Valid | |

Related commands: [MV](#)

WA WAIT FOR AXES



The WA command, valid only in [AA](#) mode, allows a command to wait until all moves on all axes are finished before executing any further commands. Some commands which can affect a non-moving axis, such as [AN](#), [AF](#) and [PA](#), may execute before a previous move on other axes has finished, especially while in a looping ([LS-LE](#), [WH-WG](#)) mode. By preceding these commands with a WA, they will not execute until all previously defined moves have finished.



Example: The Z axis auxiliary line controls a laser beam that you only want on while the Z axis moves in a positive direction. The X and Y axes position the laser. You want to repeat the action 10 times.

Enter:

```

AA;
VL 1000,1000,1000;
AC 10000,10000,10000;
LS 10;
MR 1000,1000;
GO;
WA;
AN,,1;
MR,,500;
GO;
AF,,1;
MR,,-500;
GO;
LE;

```

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| - | AY – AT | Not Valid | |
| WA; | AA-AM | 2* | 0* |
| - | AA/CD | Not Valid | |

* This command places entries in all axes' queues.

Related commands: [BW](#), [SW](#), [WQ](#), [WT](#)

WD WHILE END

The WD command serves as the loop terminator for the [WS](#) command.



Example: (see [WS](#) command on page 5-207)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WD; | AY – AT | 1 | 1 |
| WD; | AA-AM | 1* | 1* |
| - | AA/CD | Not Valid | |

* In [AA](#) or [AM](#) mode, entries are made in all axes' queues.

Related commands: [WS](#)

WG WHILE FLAG

The WG command serves as the terminator for the [WH](#) command.



Example: (see [WH](#) command page 5-204)

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WG; | AY – AT | 1 | 1 |
| WG; | AA-AM | 2* | 1* |
| - | AA/CD | Not Valid | |

*In [AA](#) or [AM](#) mode, entries are made in all axes' queues.

Related commands: [CW](#), [WH](#)

WH**WHILE**

The WH command will execute all commands between it and the terminating [WG](#) command as a loop until terminated by a [CW](#) command. This allows repeated execution of a command sequence which can be terminated by the host. These commands may not be nested but may be executed sequentially.



Example: You have a 3 axis platform that you use to drill holes in the center of a ¼ inch thick sheet of metal. The sheet is 6 inch square. The driver / motor / lead - screw pitch provide 10000 steps per inch. The operator must manually insert and remove the square from the platform. The X and Y axes move a drill into the desired position. The Z axis lifts and lowers the drill. The operator presses a switch which tells the motion controller that the square is in place and ready to be drilled. The operator will continuously remove and replace the squares until ready to take a break.

The following is a description of how to set up an OMS board to perform this task.

Procedure: Connect a normally closed momentary switch between user I/O input line 0 and ground. This will be the “Ready to Drill” switch.

Enter:

```

AX;
UU10000;      *set up user units so we can reference move to inches
AY;
UU10000;      *10000 steps = 1 inch
AZ;
UU10000;      *10000 steps = 1 inch
AX;
VL.1;
AC10;         *set up X axis homing velocity and acceleration
AY;
VL.1;
AC10;         *set up Y axis homing velocity and acceleration
AZ;
VL.1;
AC10;         *set up Z axis homing velocity and acceleration
AX;
HR;
AY;
HR;
AZ;
HR;          *send each axis to home
AA;
VL3,3,.5;    *set normal move velocity for X, Y and Z axes
WH;          *start of loop to drill squares indefinitely
SW0;         *(operator removes/replaces square into platform)
MA3,3;       *wait until operator presses switch
GO;          *move to center of square
MA,,.5;
GO;          *move the drill through the square (1/2 inch move on the
              Z axis drill through the square)
MA,,0;

```

[GO](#); *lift the drill
[MA](#)0,0;
[GO](#); *move the platform to home position
[WG](#); *loop back to starting WH command
[\(CW](#);) *operator wants a break so he/she sends [CW](#) from keyboard and presses switch once more (since loop will most likely be waiting for the switch at this point)

[MA](#)0,0,0;
[GO](#); *the loop ends and the following commands execute
 *move to home position

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WH; | AY – AT | 1 | 1 |
| WH; | AA-AM | 2* | 1* |
| - | AA/CD | Not Valid | |

— In [AA](#) or [AM](#) mode, entries are made in all axes' queues.

Related commands: [CW](#), [LS](#), [WG](#), [WS](#)

WQ WAIT FOR QUEUE TO EMPTY

The WQ command is a special command that stops the board from processing any new commands until the command queue for the current axis mode is empty; i.e. all previous moves have finished. This command is not valid in looping ([LS-LE](#), [WH-WG](#)) modes.



Example: Move the Y axis 1,000 steps and wait until the move is complete before asking for the position.

Enter: [AY](#);
 [MR](#)1000;
 [GO](#);
 WQ
 [RP](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WQ; | AY – AT | Immediate | |
| WQ; | AA-AM | Immediate | |
| - | AA/CD | Not Valid | |

Related commands: [BW](#), [SW](#), [WA](#), [WT](#)

WS WHILE SYNC

The WS command will execute the commands between the WS and [WD](#) commands as a loop while the specified general purpose input line is true; i.e. low (default). When the line goes high the UMX will exit the loop and execute the commands which follow the [WD](#) command. The test is at the bottom of the loop; i.e. the commands between WS and [WD](#) will always be executed at least once.

If the input line specified is already in the specified state to exit the loop when the WS/[WD](#) loop is issued to the UMX, those commands will be executed only once.

#RANGE: $0 \leq \text{Parameter} \leq 15$



Example: Execute a continuous loop, moving the X axis 10,000 counts and then move the Y axis -1000 counts, until an external device terminates the loop by setting general purpose input 1 high.

Enter: [AA](#);
WS1;
[MR](#)10000;
[GO](#);
[MR](#),-1000;
[GO](#);
[WD](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WS#; | AY – AT | 1 | 1 |
| WS#; | AA-AM | 1* | 1* |
| - | AA/CD | Not Valid | |

* In [AA](#) or [AM](#) mode, entries are made in all axes' queues

Related commands: [LS](#), [WD](#), [WH](#)

WT**WAIT**

The WT command will wait for a specified number of milliseconds before proceeding with the next command in the queue. In the [AA](#) mode, all axes will wait and entries are made in all axis queues. Immediate commands will not wait since they are not queued.

RANGE: $1 \leq WT \leq 200,000$



Example: You want to produce pulses on the X axis at 5,000 steps/second for 2 seconds, then 10,000 pulses/second for 3 seconds, and then stop.

Enter: [AX](#);
[JG](#)5000;
 WT2000;
[JG](#)10000;
 WT3000;
[ST](#);

Response: None.

| QUEUE REQUIREMENTS | | | |
|--------------------|---------|-----------|----------|
| FORMAT | MODE | COMMAND | ARGUMENT |
| WT#; | AY – AT | 2 | 1 |
| WT#; | AA-AM | 2 | 1 |
| - | AA/CD | Not Valid | |

Related commands: [BW](#), [SW](#), [WA](#), [WQ](#)

This page is intentionally left blank.

6. HOST SOFTWARE

6.1. INTRODUCTION TO UMX SOFTWARE SUPPORT

A CD-ROM containing device drivers, application software, and demonstration code for OMS Motion, Inc. PC family controller is supplied with the initial purchase of an OMS UMX board. Reference the text files (i.e. README.TXT) on the disk for installation instructions and other information.

Some programs on the demo disk that include source code may be adapted for use in application programs that use OMS motion controls. No license is required.
An additional CD-ROM contains OMS User Manuals.

6.2. COMMUNICATION METHODS

There are two possible ports that may be used for communication on a UMX. They are virtual COM port and OmsUmxMC.dll.

6.2.1. VIRTUAL COM PORT COMMUNICATION

The UMX driver offers a serial virtual COM port that can be accessed as a character device. This way the application has to handle interrupt symbols itself (#, !, \$, @, 0x80-0x83).

6.2.2. OMSUMXMC.DLL

The OmsUmxMC.dll eases the use by presenting a simple interface to the controller's virtual COM port. See the DLL and Example sections in the UMX software package for more information.

There are several example applications on the support disk shipped with your UMX. Reference the source code and "README" files on the support disk.

This page intentionally left blank

7. SERVICE

7.1. USER SERVICE

The UMX family of controllers contains no user serviceable parts.

This page intentionally left blank

APPENDIX A.

LIMITED WARRANTY

The Seller warrants that the articles furnished are free from defect in material and workmanship and perform to applicable, published OMS Motion, Inc. specifications for one year from date of shipment. This warranty is in lieu of any other warranty express or implied. In no event will Seller be liable for incidental or consequential damages as a result of an alleged breach of the warranty. The liability of Seller hereunder shall be limited to replacing or repairing, at its option, any defective units which are returned f.o.b. Seller's plant. Equipment or parts which have been subject to abuse, misuse, accident, alteration, neglect or unauthorized repair are not covered by warranty. Seller shall have the right of final determination as to the existence and cause of defect. As to items repaired or replaced, the warranty shall continue in effect for the remainder of the warranty period, or for 90 days following date of shipment by Seller of the repaired or replaced part whichever period is longer. No liability is assumed for expendable items such as lamps and fuses. No warranty is made with respect to custom equipment or products produced to Buyer's specifications except as specifically stated in writing by Seller and contained in the contract.

LIMITED WARRANTY

This page intentionally left blank

APPENDIX B.

TECHNICAL SUPPORT

OMS Motion, Inc. can be reached for technical support by any of the following methods:

1. Email: support@omsmotion.com
2. Internet: <http://www.omsmotion.com/>
3. Telephone: 8:00 a.m. - 5:00 p.m. Pacific Standard Time
(503) 629-8081 or (800) 707-8111
4. Facsimile: 24 Hours
(503) 629-0688 or (877) 629-0688
5. USPS: OMS Motion, Inc.
15201 NW Greenbrier Pkwy, B-1
Beaverton OR 97006

RETURN FOR REPAIRS

Call OMS Motion, Inc. Customer Service at 503-629-8081 or (800) 707-8111 or email to: sales@omsmotion.com.

Explain the problem and we may be able to solve it on the phone. If not, we will give you a Return Materials Authorization (RMA) number.

Mark the RMA number on the shipping label, packing slip and other paper work accompanying the return. We cannot accept returns without an RMA number.

Please be sure to enclose a packing slip with the RMA number, serial number of the equipment, reason for return, and the name and telephone number of the person we should contact if we have further questions.

Pack the equipment in a solid cardboard box secured with packing material.

Ship prepaid and insured to:
OMS Motion, Inc.
15201 NW Greenbrier Pkwy, B1
Beaverton, OR 97006

This page intentionally left blank

APPENDIX C.

SPECIFICATIONS

Velocity

0 to 1,044,000 counts per second simultaneous on each axis

Acceleration

0 to 8,000,000 counts per second per second

Position range

67,000,000 counts ($\pm 33,500,000$)

Accuracy

Position accuracy and repeatability ± 0 counts for point to point moves

Velocity accuracy $\pm 0.01\%$ for step pulse output

Environmental Operating temperature range: 0 to 50 degrees centigrade

Storage temperature range: -20 to 85 degrees centigrade

Humidity: 0 to 90% non-condensing

Power

+5VDC at 1 amp typical

+12VDC at 0.1 amp for servo model

-12VDC at 0.1 amp for servo model

Dimensions

3.278X3.536X0.5 inches high

Communication

USB 2.0

Update Rate

488 μ s for UMX family

Limit switch inputs

TTL input levels with on board 2.2K pull up resistor, requires only external switch closure to ground or TTL level input signal.

Input sense (low or high true) selectable by command input for each axis.

Home switch inputs

TTL input levels with on board 2.2K pull up resistor, requires only external switch closure to ground or TTL level input signal. Input sense (low or high true) selectable by command input for each axis.

User definable I/O

Up to 12 bits of user definable I/O on UMX models with 4 axes or less. 8 bits are user configurable as inputs or outputs. One auxiliary output per axis and these are fixed as outputs. Factory default is 4 inputs, 4 outputs and 1 auxiliary output per axis

TTL input levels with on board 2.2K pull up resistor, requires only external switch closure to ground or TTL level input signal.

The auxiliary outputs are TTL open collector outputs (7406, max 48mA). The other outputs are TTL totem pole outputs (74LS243, max 24mA).

Analog outputs

+/-10V and 0 to +10V (servo models only)

Step pulse output

Pulse width 50% duty cycle. Open collector TTL level signal (7406, max 48mA).

Direction output

Open collector TTL level signal (7406, max 48mA).

Encoder Feedback

Maximum 12 MHz after 4x quadrature detection

Differential TTL level signal MC26G32, max 150mA

| UMX CONNECTOR (J5) | | | |
|---------------------------|------|------|-------------|
| Description | Pin# | Pin# | Description |
| Digital GND | 1 | 35 | +5VDC |
| I/O-1 | 2 | 36 | I/O-0 |
| I/O-3 | 3 | 37 | I/O-2 |
| I/O-5 | 4 | 38 | I/O-4 |
| I/O-7 | 5 | 39 | I/O-6 |
| Digital Ground | 6 | 40 | +5VDC |
| X Index + | 7 | 41 | X Servo |
| X Index - | 8 | 42 | X Step |
| X Phase A + | 9 | 43 | X Auxiliary |
| X Phase A - | 10 | 44 | X Direction |
| X Phase B + | 11 | 45 | X Limit + |
| X Phase B - | 12 | 46 | X Limit - |
| Y Servo | 13 | 47 | X Home |
| Y Index + | 14 | 48 | Y Step |
| Y Index - | 15 | 49 | Y Auxiliary |
| Y Phase A + | 16 | 50 | Y Direction |
| Y Phase A - | 17 | 51 | Y Limit + |
| Y Phase B + | 18 | 52 | Y Limit - |
| Y Phase B - | 19 | 53 | Y Home |
| Analog GND | 20 | 54 | +5VDC |
| Z Index + | 21 | 55 | Z Servo |
| Z Index - | 22 | 56 | Z Step |
| Z Phase A + | 23 | 57 | Z Auxiliary |
| Z Phase A - | 24 | 58 | Z Direction |
| Z Phase B + | 25 | 59 | Z Limit + |
| Z Phase B - | 26 | 60 | Z Limit - |
| T Servo | 27 | 61 | Z Home |
| T Index + | 28 | 62 | T Step |
| T Index - | 29 | 63 | T Auxiliary |
| T Phase A + | 30 | 64 | T Direction |
| T Phase A - | 31 | 65 | T Limit + |
| T Phase B + | 32 | 66 | T Limit - |
| T Phase B - | 33 | 67 | T Home |
| Digital GND | 34 | 68 | +5VDC |

| IO68-M TERMINAL BLOCK PIN-OUT | | | | | |
|-------------------------------|-------------|-------|-------------|-------|----------------|
| Row 1 | Description | Row 2 | Description | Row 3 | Description |
| 1 | X Step | 24 | X Direction | 47 | X Auxiliary |
| 2 | X Phase A+ | 25 | X Phase B+ | 48 | X Index + |
| 3 | X Phase A- | 26 | X Phase B- | 49 | X Index - |
| 4 | X Limit + | 27 | X Limit- | 50 | X Home |
| 5 | +5VDC | 28 | X Servo | 51 | Digital Ground |
| 6 | Y Step | 29 | Y Direction | 52 | Y Auxiliary |
| 7 | Y Phase A+ | 30 | Y Phase B+ | 53 | Y Index + |
| 8 | Y Phase A- | 31 | Y Phase B- | 54 | Y Index - |
| 9 | Y Limit + | 32 | Y Limit- | 55 | Y Home |
| 10 | +5VDC | 33 | Y Servo | 56 | Digital Ground |
| 11 | I/O-0 | 34 | I/O-3 | 57 | I/O-5 |
| 12 | I/O-1 | 35 | No Connect | 58 | I/O-6 |
| 13 | I/O-2 | 36 | I/O-4 | 59 | I/O-7 |
| 14 | +5VDC | 37 | Z Servo | 60 | Analog Ground |
| 15 | Z Step | 38 | Z Direction | 61 | Z Auxiliary |
| 16 | Z Phase A+ | 39 | Z Phase B+ | 62 | Z Index + |
| 17 | Z Phase A- | 40 | Z Phase B- | 63 | Z Index - |
| 18 | Z Limit + | 41 | Z Limit- | 64 | Z Home |
| 19 | +5VDC | 42 | T Servo | 65 | T Auxiliary |
| 20 | T Step | 43 | T Direction | 66 | T Index + |
| 21 | T Phase A+ | 44 | T Phase B+ | 67 | T Index - |
| 22 | T Phase A- | 45 | T Phase B- | 68 | T Home |
| 23 | T Limit + | 46 | T Limit- | 69 | Digital Ground |

| UIO J1 (X), J2 (Y), J3 (Z) , J4 (T) Connector pin outs | |
|---|-----------|
| Pin | Signal |
| 1 | GND |
| 2 | Phase B- |
| 3 | Phase B+ |
| 4 | Index - |
| 5 | Index + |
| 6 | Phase A - |
| 7 | Phase A + |
| 8 | 5V |
| 9 | Home |
| 10 | Limit - |
| 11 | Limit + |
| 12 | Servo |
| 13 | Auxiliary |
| 14 | Direction |
| 15 | Step |

| UIO J8 (IO) Connector pin outs | |
|-----------------------------------|--------|
| Pin | Signal |
| 1 | IO7 |
| 2 | IO6 |
| 3 | IO5 |
| 4 | IO4 |
| 5 | IO3 |
| 6 | IO2 |
| 7 | IO1 |
| 8 | IO0 |
| 9 | GND |
| 10 | 5V |
| 11 | GND |
| 12 | 5V |
| 13 | GND |
| 14 | 5V |
| 15 | GND |

| UIO J5 Connector | |
|------------------|--------------------|
| Pin | End Signal |
| 1 | GND |
| 2 | J8-Pin7 IO1 |
| 3 | J8-Pin5 IO3 |
| 4 | J8-Pin3 IO5 |
| 5 | J8-Pin1 IO7 |
| 6 | GND |
| 7 | J1-Pin5 X Index+ |
| 8 | J1-Pin4 X Index- |
| 9 | J1-Pin7 X Phase A+ |
| 10 | J1-Pin6 X Phase A- |
| 11 | J1-Pin3 X Phase B+ |
| 12 | J1-Pin2 X Phase B- |
| 13 | J2-Pin12 Y Servo |
| 14 | J2-Pin5 Y Index+ |
| 15 | J2-Pin4 Y Index- |
| 16 | J2-Pin7 Y Phase A+ |
| 17 | J2-Pin6 Y Phase A- |
| 18 | J2-Pin3 Y Phase B+ |
| 19 | J2-Pin2 Y Phase B- |
| 20 | GND |
| 21 | J3-Pin5 Z Index+ |
| 22 | J3-Pin4 Z Index- |
| 23 | J3-Pin7 Z Phase A+ |
| 24 | J3-Pin6 Z Phase A- |
| 25 | J3-Pin3 Z Phase B+ |
| 26 | J3-Pin2 Z Phase B- |
| 27 | J4-Pin12 T Servo |
| 28 | J4-Pin5 T Index+ |
| 29 | J4-Pin4 T Index- |
| 30 | J4-Pin7 T Phase A+ |
| 31 | J4-Pin6 T Phase A- |
| 32 | J4-Pin3 T Phase B+ |
| 33 | J4-Pin2 T Phase B- |
| 34 | GND |

| UIO J6 Connector | |
|------------------|------------------|
| Pin | End Signal |
| 1 | 5V |
| 2 | J8-Pin8 IO0 |
| 3 | J8-Pin6 IO2 |
| 4 | J8-Pin4 IO4 |
| 5 | J8-Pin2 IO6 |
| 6 | 5V |
| 7 | J1-Pin12 X Servo |
| 8 | J1-Pin15 X Step |
| 9 | J1-Pin13 X Aux |
| 10 | J1-Pin14 X Dir |
| 11 | J1-Pin11 X Lmt+ |
| 12 | J1-Pin10 X Lmt- |
| 13 | J1-Pin9 X Home |
| 14 | J2-Pin15 Y Step |
| 15 | J2-Pin13 Y Aux |
| 16 | J2-Pin14 Y Dir |
| 17 | J2-Pin11 Y Lmt+ |
| 18 | J2-Pin10 Y Lmt- |
| 19 | J2-Pin9 Y Home |
| 20 | 5V |
| 21 | J3-Pin12 Z Servo |
| 22 | J3-Pin15 Z Step |
| 23 | J3-Pin13 Z Aux |
| 24 | J3-Pin14 Z Dir |
| 25 | J3-Pin11 Z Lmt+ |
| 26 | J3-Pin10 Z Lmt- |
| 27 | J3-Pin9 Z Home |
| 28 | J4-Pin15 T Step |
| 29 | J4-Pin13 T Aux |
| 30 | J4-Pin14 T Dir |
| 31 | J4-Pin11 T Lmt+ |
| 32 | J4-Pin10 T Lmt- |
| 33 | J4-Pin9 T Home |
| 34 | 5V |

| CBL15-5 | |
|----------------|-------------------|
| Pin | Wire Color |
| 1 | Black |
| 2 | Brown |
| 3 | Red |
| 4 | Orange |
| 5 | Yellow |
| 6 | Dark Green |
| 7 | Blue |
| 8 | Purple |
| 9 | Grey |
| 10 | White |
| 11 | Pink |
| 12 | Light Green |
| 13 | Black/White |
| 14 | Brown/White |
| 15 | Red/White |

| UMX Intelligent Motion Controls | | | | | | | |
|--|----------------------|------------|---|------------------|-----|------|----------|
| Model | Interface | Servo Axes | Stepper Axes (Open or Closed – loop) | Axes control I/O | | | User I/O |
| | Virtual COM over USB | | | Limits | Aux | Home | GPIO |
| UMX-25 | Yes | 2 | | 4 | 2 | 2 | 8 |
| UMX-45 | Yes | 4 | | 8 | 4 | 4 | 8 |
| UMX-26 | Yes | | 2 | 4 | 2 | 2 | 8 |
| UMX-46 | Yes | | 4 | 8 | 4 | 4 | 8 |

This page is intentionally left blank.

INDEX

?

| | |
|---|-------|
| ?AC, REPORT AC COMMAND | 5-27 |
| ?AD, REPORT DEFAULT AUXILIARY BIT STATE | 5-27 |
| ?AQ, QUERY CURRENT AXIS | 5-34 |
| ?BD, REPORT BIT DIRECTION | 5-36 |
| ?BS, REPORT BIT STATE | 5-41 |
| ?DA, PRINT A CUSTOM RAMP | 5-54 |
| ?DB, REPORT DIRECTION BIT LOGIC | 5-57 |
| ?DE, REPORT A CUSTOM RAMP TABLE ENTRY | 5-61 |
| ?DS, REPORT THE SIZE OF A CUSTOM RAMP TABLE | 5-62 |
| ?DZ, REPORT DAC OPEN-LOOP OFFSET | 5-64 |
| ?EH, QUERY ENCODER HOME | 5-68 |
| ?ER, REPORT MOTOR:ENCODER | 5-71 |
| ?ES, REPORT ENCODER SLIP | 5-73 |
| ?HD, REPORT POSITION MAINTENANCE DEADBAND | 5-84 |
| ?HG, REPORT POSITION MAINTENANCE GAIN | 5-87 |
| ?HV, REPORT POSITION MAINTENANCE VELOCITY | 5-97 |
| ?KA, REPORT ACCELERATION FEEDFORWARD | 5-109 |
| ?KB, REPORT AXIS PID UPPER BOUND LIMIT | 5-110 |
| ?KD, REPORT PID DERIVATIVE GAIN | 5-111 |
| ?KF, REPORT SERVO AXIS PID FRICTION COEFFICIENT | 5-112 |
| ?KI, REPORT PID INTEGRAL GAIN | 5-113 |
| ?KO, REPORT PID OFFSET COEFFICIENT | 5-116 |
| ?KP, REPORT PROPORTIONAL GAIN COEFFICIENT | 5-117 |
| ?KU, REPORT PID INTEGRATION SUM UPPER LIMIT | 5-120 |
| ?KV, REPORT VELOCITY FEEDFORWARD | 5-121 |
| ?LS, REPORT LIMIT ACTIVE STATE | 5-132 |
| ?PA, REPORT POWER AUTOMATIC STATE | 5-145 |
| ?PM, REPORT HOLD STATE | 5-148 |
| ?RT, REPORT RAMP TYPE | 5-164 |
| ?SB, QUERY THE BAUD RATE | 5-168 |
| ?SE, REPORT SETTLING TIME | 5-171 |
| ?SL, REPORT SOFT LIMIT STATUS | 5-176 |
| ?SO, REPORT ANALOG OUTPUT MODE | 5-178 |
| ?SV, REPORT SERVO VOLTAGE INVERSION STATE | 5-181 |
| ?TL, REPORT SOFTWARE OVERTRAVEL LIMITS | 5-189 |
| ?UU, REPORT AXIS USER UNIT | 5-196 |
| ?VB, REPORT BASE VELOCITY | 5-198 |
| ?VL, REPORT PEAK VELOCITY SETTING | 5-200 |

A

| | |
|--|-------|
| AA, ALL AXES | 5-25 |
| AC, ACCELERATION | 5-26 |
| ACCELERATION FEEDFORWARD, KA | 5-109 |
| ACCELERATION, AC | 5-26 |
| ADH, SET AUXILIARY DEFAULT TO HIGH | 5-28 |
| ADL, SET AUXILIARY DEFAULT TO LOW | 5-29 |
| AF, AUXILIARY OFF | 5-30 |
| ALL AXES, AA | 5-25 |
| AM, AXES MULTITASKING | 5-31 |
| AN, AUXILIARY ON | 5-32 |
| AP, ASSIGN CURRENT PARAMETERS AS POWER-UP DEFAULTS | 5-33 |
| ASSIGN CURRENT PARAMETERS AS POWER UP DEFAULTS, AP | 5-33 |

| | |
|-----------------------------|------|
| AT, AXIS T | 5-34 |
| AUXILIARY OFF, AF | 5-30 |
| AUXILIARY ON, AN | 5-32 |
| AX, AXIS X | 5-35 |
| AXES MULTITASKING, AM | 5-31 |
| AXIS T, AT | 5-34 |
| AXIS X, AX | 5-35 |
| AXIS Y, AY | 5-35 |
| AXIS Z, AZ | 5-36 |
| AY, AXIS Y | 5-35 |
| AZ, AXIS Z | 5-36 |

B

| | |
|---|------|
| BEGIN CUSTOM RAMP DEFINITION, DAR | 5-56 |
| BH, BIT HIGH | 5-37 |
| BI, BIPOLAR | 5-38 |
| BIPOLAR, BI | 5-38 |
| BIT HIGH, BH | 5-37 |
| BIT LOW, BL | 5-39 |
| BIT REQUEST IN HEX, BX | 5-42 |
| BIT SET, BS | 5-40 |
| BL, BIT LOW | 5-39 |
| BS, BIT SET | 5-40 |
| BW, WAIT FOR INPUT TO GO LOW | 5-41 |
| BX, BIT REQUEST IN HEX | 5-42 |

C

| | |
|----------------------------------|-------|
| CA, CLEAR AXIS DONE FLAG | 5-43 |
| CB, CLEAR MACRO LINKS | 5-43 |
| CD, CONTOUR DEFINE | 5-44 |
| CE, CONTOUR END | 5-46 |
| CG, CONTOUR PRIORITY | 5-47 |
| CIRCULAR INTERPOLATION, CR | 5-51 |
| CK, CONTOUR END AND KILL | 5-49 |
| CLEAR AXIS DONE FLAG, CA | 5-43 |
| CLEAR MACRO LINKS | 5-43 |
| CLEAR WHILE, CW | 5-52 |
| CN, COSINE ON | 5-50 |
| COMMAND LISTING BY SECTION | 5-10 |
| CONTOUR DEFINE, CD | 5-44 |
| CONTOUR END AND KILL, CK | 5-49 |
| CONTOUR END, CE | 5-46 |
| CONTOUR EXECUTE, CX | 5-53 |
| CONTOUR PRIORITY, CG | 5-47 |
| CONTOUR VELOCITY, CV | 5-52 |
| COSINE ON, CN | 5-50 |
| COSINE RAMP PER AXIS, SC | 5-169 |
| CR, CIRCULAR INTERPOLATION | 5-51 |
| CURRENT MODE | 2-5 |
| CV, CONTOUR VELOCITY | 5-52 |
| CW, CLEAR WHILE | 5-52 |
| CX, CONTOUR EXECUTE | 5-53 |

D

| | |
|--|------|
| DAB, DEFINE CUSTOM RAMP BREAKPOINT | 5-55 |
| DAE, END CUSTOM RAMP DEFINITION | 5-55 |

| | |
|--|-------|
| DAR, BEGIN CUSTOM RAMP DEFINITION | 5-56 |
| DBI, INVERT DIRECTION BIT | 5-58 |
| DBN, NORMALIZE DIRECTION BIT | 5-59 |
| DC, DECELERATION | 5-60 |
| DECELERATION, DC | 5-60 |
| DEFAULT ENCODER RATIO | 5-70 |
| DEFINE CUSTOM RAMP BREAKPOINT, DAB | 5-55 |
| DEFINE KILL LINK ON UMX, SK | 5-174 |
| DEFINE MACRO LINK ON UMX, SX | 5-185 |
| DEFINE ZERO POSITION IN OPEN-LOOP MODE, DZ | 5-63 |
| DEFINING ENCODER HOME, EH | 5-67 |
| DERIVATIVE GAIN COEFFICIENT, KD | 5-111 |
| DZ, DEFINE ZERO POSITION IN OPEN LOOP MODE | 5-63 |

E

| | |
|---|-------|
| EA, ENCODER STATUS | 5-65 |
| ECHO OFF, EF | 5-66 |
| ECHO ON, EN | 5-69 |
| EF, ECHO OFF | 5-66 |
| EH, DEFINING ENCODER HOME | 5-67 |
| EN, ECHO ON | 5-69 |
| ENABLE AXIS GANTRY MODE, FX | 5-77 |
| ENABLE/DISABLE STAND-ALONE MODE, SM | 5-176 |
| ENCODER RATIO, ER | 5-70 |
| ENCODER SLIP TOLERANCE, ES | 5-72 |
| ENCODER STATUS, EA | 5-65 |
| ENCODER TRACKING, ET | 5-74 |
| END CUSTOM RAMP DEFINITION, DAE | 5-55 |
| ER, ENCODER RATIO | 5-70 |
| ES, ENCODER SLIP TOLERANCE | 5-72 |
| ET, ENCODER TRACKING | 5-74 |

F

| | |
|-----------------------------------|------|
| FL, FLUSH | 5-75 |
| FLUSH, FL | 5-75 |
| FORCE POSITION. FP | 5-76 |
| FP, FORCE POSITION | 5-76 |
| FX, ENABLE AXIS GANTRY MODE | 5-77 |

G

| | |
|---------------------------------------|------|
| GD, GO AND RESET DONE | 5-78 |
| GN, GO AND NOTIFY WHEN DONE | 5-80 |
| GO | 5-81 |
| GO AND MONITOR SLIP TRIGGER, GS | 5-82 |
| GO AND NOTIFY WHEN DONE, GN | 5-80 |
| GO AND RESET DONE, GD | 5-78 |
| GO ASYMMETRICAL, GU | 5-83 |
| GS, GO AND MONITOR SLIP | 5-82 |
| GU, GO ASYMMETRICAL | 5-83 |

H

| | |
|-------------------------|------|
| HD, HOLD DEADBAND | 5-84 |
| HE, HOME ENCODER | 5-85 |
| HF, HOLD OFF | 5-86 |

| | |
|---------------------------------|-------|
| HG, HOLD GAIN | 5-87 |
| HH, HOME HIGH | 5-88 |
| HL, HOME LOW | 5-89 |
| HM, HOME | 5-90 |
| HN, HOLD ON | 5-92 |
| HOLD DEADBAND, HD | 5-84 |
| HOLD GAIN, HG | 5-87 |
| HOLD OFF, HF | 5-86 |
| HOLD ON | 5-92 |
| HOLD ON, HN | 5-92 |
| HOLD VELOCITY, HV | 5-96 |
| HOME AND KILL, KM | 5-115 |
| HOME ENCODER, HE | 5-85 |
| HOME HIGH, HH | 5-88 |
| HOME LOW, HL | 5-89 |
| HOME REVERSE AND KILL, KR | 5-118 |
| HOME REVERSE, HR | 5-93 |
| HOME SWITCH, HS | 5-95 |
| HOME, HM | 5-90 |
| HR, HOME REVERSE | 5-93 |
| HS, HOME SWITCH | 5-95 |
| HV, HOLD VELOCITY | 5-96 |

I

| | |
|--|-------|
| IC, INTERRUPT CLEAR | 5-98 |
| ID, INTERRUPT WHEN DONE | 5-99 |
| II, INTERRUPT INDEPENDENT | 5-100 |
| IN, INTERRUPT NEARLY DONE | 5-101 |
| INTEGRAL GAIN COEFFICIENT, KI | 5-113 |
| INTERRUPT CLEAR, IC | 5-98 |
| INTERRUPT INDEPENDENT, II | 5-100 |
| INTERRUPT NEARLY DONE, IN | 5-101 |
| INTERRUPT ON SLIP, IS | 5-104 |
| INTERRUPT WHEN AXES DONE, IX | 5-105 |
| INTERRUPT WHEN DONE, ID | 5-99 |
| INTERRUPT WHEN IN POSITION, IP | 5-103 |
| INVERT DIRECTION BIT, DBI | 5-58 |
| INVERT SERVO VOLTAGE, SVI | 5-182 |
| IO, SET I/O BIT DIRECTION | 5-102 |
| IO68 ADAPTER MODULE | 4-4 |
| IO68-M | 4-4 |
| IO68-M ENCODER BIAS SWITCH (S43) | 4-5 |
| IO68-M ENCODER BIAS SWITCH (S45) | 4-5 |
| IP, INTERRUPT WHEN IN POSITION | 5-103 |
| IS, INTERRUPT ON SLIP | 5-104 |
| IX, INTERRUPT WHEN AXES ARE DONE | 5-105 |

J

| | |
|-------------------------------------|-------|
| JF, JOG FRACTIONAL VELOCITIES | 5-106 |
| JG, JOG | 5-107 |
| JOG FRACTIONAL VELOCITIES, JF | 5-106 |
| JOG, JG | 5-107 |

K

| | |
|---|-------|
| KA, ACCELERATION FEEDFORWARD | 5-109 |
| KB, PID UPPER BOUND LIMIT COEFFICIENT | 5-110 |

| | |
|---|-------|
| KD, DERIVATIVE GAIN COEFFICIENT | 5-111 |
| KF, SET SERVO AXIS PID FRICTION COEFFICIENT | 5-112 |
| KI, INTEGRAL GAIN COEFFICIENT | 5-113 |
| KILL SELECTED AXES, KS | 5-119 |
| KILL, KL | 5-114 |
| KL, KILL | 5-114 |
| KM, HOME AND KILL | 5-115 |
| KO, OFFSET COEFFICIENT | 5-116 |
| KP, PROPORTIONAL GAIN COEFFICIENT | 5-117 |
| KR, HOME REVERSE AND KILL | 5-118 |
| KS, KILL SELECTED AXES | 5-119 |
| KV, VELOCITY FEEDFORWARD | 5-121 |

L

| | |
|--------------------------------|-------|
| LA, LINEAR RAMP PER AXIS | 5-122 |
| LE, LOOP END | 5-123 |
| LF, LIMITS OFF | 5-124 |
| LH, LIMITS HIGH | 5-125 |
| LIMITS HIGH, LH | 5-125 |
| LIMITS LOW, LL | 5-126 |
| LIMITS OFF, LF | 5-124 |
| LIMITS ON, LN | 5-127 |
| LINEAR ON, PF | 5-146 |
| LINEAR RAMP PER AXIS, LA | 5-122 |
| LL, LIMITS LOW | 5-126 |
| LN, LIMITS ON | 5-127 |
| LO, LOAD MOTOR POSITION | 5-128 |
| LOAD MOTOR POSITION, LO | 5-128 |
| LOAD POSITION, LP | 5-129 |
| LOOP END, LE | 5-123 |
| LOOP START, LS | 5-130 |
| LP, LOAD POSITION | 5-129 |
| LS, LOOP START | 5-130 |

M

| | |
|----------------------------------|-------|
| MA, MOVE ABSOLUTE | 5-133 |
| MACRO EXECUTE, MX | 5-143 |
| MD, TEMPORARY MACRO DEFINE | 5-135 |
| ML, MOVE LINEAR | 5-136 |
| MM, MOVE MINUS | 5-137 |
| MOVE ABSOLUTE, MA | 5-133 |
| MOVE LINEAR, ML | 5-136 |
| MOVE MINUS, MM | 5-137 |
| MOVE POSITIVE, MP | 5-137 |
| MOVE RELATIVE, MR | 5-138 |
| MOVE TO, MT | 5-140 |
| MOVE VELOCITY, MV | 5-141 |
| MP, MOVE POSITIVE | 5-137 |
| MR, MOVE RELATIVE | 5-138 |
| MT, MOVE TO | 5-140 |
| MV, MOVE VELOCITY | 5-141 |
| MX, MACRO EXECUTE | 5-143 |

N

| | |
|------------------------------------|-------|
| NEW CONTOUR VELOCITY, NV | 5-143 |
| NORMALIZE DIRECTION BIT, DBN | 5-59 |

| | |
|------------------------------------|-------|
| NORMALIZE SERVO VOLTAGE, SVN | 5-182 |
| NV, NEW CONTOUR VELOCITY | 5-143 |

O

| | |
|---|-------|
| OFFSET COEFFICIENT, KO | 5-116 |
| OUTPUT CONNECTOR PIN LIST (J5) (AT CIRCUIT BOARD) | 4-3 |

P

| | |
|--|-------|
| PA, POWER AUTOMATIC..... | 5-144 |
| PARABOLIC ON, PN | 5-149 |
| PARABOLIC RAMP PER AXIS, PR..... | 5-151 |
| PE, REPORT ENCODER POSITIONS | 5-146 |
| PF, LINEAR ON | 5-146 |
| PID INTEGRATION SUM UPPER LIMIT, KU | 5-120 |
| PID UPPER BOUND LIMIT COEFFICIENT, ?KB | 5-110 |
| PM, PRINT MACRO..... | 5-147 |
| PN, PARABOLIC ON | 5-149 |
| POWER AUTOMATIC, PA..... | 5-144 |
| PP, REPORT MOTOR POSITIONS..... | 5-150 |
| PR, PARABOLIC RAMP PER AXIS..... | 5-151 |
| PRESERVE A TEMPORARY MACRO, PT..... | 5-153 |
| PRINT A CUSTOM RAMP, ?DA | 5-54 |
| PRINT MACRO,PM..... | 5-147 |
| PROFILE DISTANCE FORMULA | 2-6 |
| PROPORTIONAL GAIN COEFFICIENT, KP | 5-117 |
| PS, REPORT MACRO LINK | 5-152 |
| PT, PRESERVE A TEMPORARY | 5-153 |

Q

| | |
|----------------------------------|-------|
| QA, QUERY AXIS STATUS | 5-154 |
| QI, QUERY INTERRUPT STATUS | 5-154 |
| QL, QUERY ALL LIMIT SENSORS..... | 5-155 |
| QUERY ALL LIMIT SENSORS, QL..... | 5-155 |
| QUERY AXIS STATUS, QA..... | 5-154 |
| QUERY CURRENT AXIS, ?AQ..... | 5-34 |
| QUERY ENCODER HOME?, EH..... | 5-68 |
| QUERY INTERRUPT STATUS, QI | 5-154 |
| QUERY THE BAUD RATE, ?SB | 5-168 |

R

| | |
|--|-------|
| RA, REQUEST AXIS STATUS..... | 5-156 |
| RB, REPORT BIT DIRECTION | 5-157 |
| RC, REQUEST ACCELERATION..... | 5-157 |
| RD, RESTORE DEFAULT VALUES | 5-158 |
| RE, REPORT ENCODER POSITION | 5-158 |
| REMAINDER, RM | 5-161 |
| REPORT A CUSTOM RAMP TABLE ENTRY, ?DE..... | 5-61 |
| REPORT AC COMMAND, ?AC | 5-27 |
| REPORT ACCELERATION FEED-FORWARD, ?AC | 5-109 |
| REPORT ANALOG OUTPUT MODE, ?SO..... | 5-178 |
| REPORT AXIS PID UPPER BOUND LIMIT, ?KB | 5-110 |
| REPORT AXIS USER UNITS', ?UU | 5-196 |
| REPORT BASE VELOCITY, ?VB | 5-198 |
| REPORT BIT DIRECTION, ?BD | 5-36 |

| | |
|--|-------|
| REPORT BIT DIRECTION, RB..... | 5-157 |
| REPORT BIT STATE, ?BS..... | 5-41 |
| REPORT DAC OPEN-LOOP OFFSET, ?DZ..... | 5-64 |
| REPORT DEFAULT AUXILIARY BIT STATE, ?AD..... | 5-27 |
| REPORT DIRECTION BIT LOGIC, ?DB..... | 5-57 |
| REPORT ENCODER POSITIONS, ?PE..... | 5-146 |
| REPORT ENCODER SLIP TOLERANCE, ?ES..... | 5-73 |
| REPORT HOLD STATE, ?PM..... | 5-148 |
| REPORT LIMIT ACTIVE STATE, ?LS..... | 5-132 |
| REPORT MACRO LINK, PS..... | 5-152 |
| REPORT MOTOR:ENCODER RATIO, ER..... | 5-71 |
| REPORT MOTOR POSITIONS, PP..... | 5-150 |
| REPORT PEAK VELOCITY SETTING, ?VL..... | 5-200 |
| REPORT PID DERIVATIVE GAIN, ?KD..... | 5-111 |
| REPORT PID INTEGRAL GAIN, ?KI..... | 5-113 |
| REPORT PID INTEGRATION SUM UPPER LIMIT, ?KU..... | 5-120 |
| REPORT PID OFFSET COEFFICIENT, ?KO..... | 5-116 |
| REPORT POSITION IN USER UNITS, RU..... | 5-165 |
| REPORT POSITION MAINTENANCE DEADBAND, ?HD..... | 5-84 |
| REPORT POSITION MAINTENANCE GAIN, ?HG..... | 5-87 |
| REPORT POSITION MAINTENANCE VELOCITY, ?HV..... | 5-97 |
| REPORT POWER AUTOMATIC STATE, ?PA..... | 5-145 |
| REPORT PROPORTIONAL GAIN, ?KP..... | 5-117 |
| REPORT QUEUE SIZE, RQ..... | 5-163 |
| REPORT RAMP TYPE, ?RT..... | 5-164 |
| REPORT SERVO AXIS FRICTION OFFSET, ?KF..... | 5-112 |
| REPORT SERVO VOLTAGE INVERSION STATE, ?SV..... | 5-181 |
| REPORT SETTLING TIME, ?SE..... | 5-171 |
| REPORT SOFT LIMIT STATUS, ?SL..... | 5-176 |
| REPORT SOFTWARE OVERTRAVEL LIMITS, ?TL..... | 5-189 |
| REPORT THE SIZE OF A CUSTOM RAMP TABLE, ?DS..... | 5-62 |
| REPORT VELOCITY FEED FORWARD, ?KV..... | 5-121 |
| REQUEST ACCELERATION, RC..... | 5-157 |
| REQUEST AXIS STATUS, RA..... | 5-156 |
| REQUEST ENCODER POSITION, RE..... | 5-158 |
| REQUEST INTERRUPT STATUS, RI..... | 5-159 |
| REQUEST POSITION, RP..... | 5-162 |
| REQUEST VELOCITY, RV..... | 5-166 |
| RESET, RS..... | 5-163 |
| RESTORE DEFAULT VALUES, RD..... | 5-158 |
| RESTORE FACTORY DEFAULT VALUES, RF..... | 5-159 |
| RETURN SLIP STATUS, RL..... | 5-160 |
| RF, RESTORE FACTORY DEFAULT VALUES..... | 5-159 |
| RI, REQUEST INTERRUPT STATUS..... | 5-159 |
| RL, REPORT SLIP STATUS..... | 5-160 |
| RM, REMAINDER..... | 5-161 |
| RP, REQUEST POSITION..... | 5-162 |
| RQ, REPORT QUEUE SIZE..... | 5-163 |
| RS, RESET..... | 5-163 |
| RU, REPORT POSITION IN USER UNITS..... | 5-165 |
| RV, REQUEST VELOCITY..... | 5-166 |

S

| | |
|---|-------|
| SA, STOP ALL..... | 5-167 |
| SB, SET BAUD RATE..... | 5-168 |
| SC, COSINE RAMP PER AXIS..... | 5-169 |
| SD, STOP AND RESET DONE..... | 5-170 |
| SE, SETTLING TIME..... | 5-171 |
| SELECT CUSTOM RAMP, SR..... | 5-179 |
| SET AUXILIARY DEFAULT TO HIGH, ADH..... | 5-28 |

| | |
|---|-------|
| SET AUXILIARY DEFAULT TO LOW, ADL | 5-29 |
| SET BAUD RATE, SB | 5-168 |
| SET I/O BIT DIRECTION, IO | 5-102 |
| SET SERVO AXIS PID FRICTION COEFFICIENT, KF | 5-112 |
| SET SOFTWARE OVERTRAVEL LIMITS, TL | 5-188 |
| SETTLING TIME, SE | 5-171 |
| SF, SOFT LIMITS OFF | 5-172 |
| SI, STOP INDIVIDUAL | 5-173 |
| SK, DEFINE KILL LINK ON UMX | 5-174 |
| SL, SOFT LIMITS ON | 5-175 |
| SM, ENABLE/DISABLE STAND ALONE MODE | 5-176 |
| SO, STOP AT POSITION BY RAMPING FROM A DISTANCE | 5-177 |
| SOFT LIMIT OFF, SF | 5-172 |
| SOFT LIMITS ON, SL | 5-175 |
| SOFTWARE SUPPORT | 6-1 |
| SP, STOP AT POSITION | 5-178 |
| SR, SELECT CUSTOM RAMP | 5-179 |
| ST, STOP | 5-180 |
| STOP ALL, SA | 5-167 |
| STOP AND RESET DONE, SD | 5-170 |
| STOP AT POSITION BY RAMPING FROM DISTANCE, SO | 5-177 |
| STOP AT POSITION, SP | 5-178 |
| STOP INDIVIDUAL, SI | 5-173 |
| STOP, ST | 5-180 |
| SVI, INVERT SERVO VOLTAGE | 5-182 |
| SVN, NORMALIZE SERVO VOLTAGE | 5-182 |
| SW, SYNC WAIT | 5-183 |
| SX, DEFINE MACRO LINK ON UMX | 5-185 |
| SYNC WAIT, SW | 5-183 |

T

| | |
|--|-------|
| TABLE OF COMMANDS, ALPHABETICAL | 5-4 |
| TEMPORARY MACRO DEFINE, MD | 5-135 |
| TF, TURN OFF SLIP KILL MODE | 5-187 |
| TIMED JOG, TM | 5-190 |
| TL, SET SOFTWARE OVERTRAVEL LIMITS | 5-188 |
| TM, TIMED JOG | 5-190 |
| TN, TURN ON SLIP KILL MODE | 5-191 |
| TRACK THE X-AXIS, TX | 5-192 |
| TURN OFF SLIP KILL MODE, TF | 5-187 |
| TURN ON SLIP KILL MODE, TN | 5-191 |
| TX, TRACK THE X AXIS | 5-192 |

U

| | |
|--------------------------|-------|
| UF, USER UNITS OFF | 5-193 |
| UIO ADAPTER MODULE | 4-6 |
| UN, UNIPOLAR | 5-194 |
| UNIPOLAR UN | 5-194 |
| USER UNITS OFF, UF | 5-193 |
| USER UNITS, UU | 5-195 |
| UU, USER UNITS | 5-195 |

V

| | |
|--------------------------------|-------|
| VB, VELOCITY BASE | 5-197 |
| VELOCITY BASE, VB | 5-197 |
| VELOCITY FEEDFORWARD, KV | 5-121 |

| | |
|------------------------------|-------|
| VELOCITY STREAMING, VS | 5-201 |
| VELOCITY, VL | 5-199 |
| VL, VELOCITY | 5-199 |
| VOLTAGE MODE | 2-5 |
| VS, VELOCITY STREAMING | 5-201 |

W

| | |
|------------------------------------|-------|
| WA, WAIT FOR AXES | 5-202 |
| WAIT FOR AXES, WA | 5-202 |
| WAIT FOR INPUT TO GO LOW, BW | 5-41 |
| WAIT FOR QUEUE TO EMPTY, WQ | 5-206 |
| WAIT, WT | 5-208 |
| WD, WHILE END | 5-203 |
| WG, WHILE FLAG | 5-203 |
| WH, WHILE | 5-204 |
| WHILE END, WD | 5-203 |
| WHILE FLAG, WG | 5-203 |
| WHILE SYNC, WS | 5-207 |
| WHILE, WH | 5-204 |
| WHO ARE YOU, WY | 5-209 |
| WQ, WAIT FOR QUEUE TO EMPTY | 5-206 |
| WS, WHILE SYNC | 5-207 |
| WT, WAIT | 5-208 |
| WY, WHO ARE YOU | 5-209 |