CHAPTER 6
Marker Motion™ MOTION CONTROL

Section I: Overview

Marking systems are often required to control one or more motors or actuators to execute complex production sequences. Typical uses of *Marker Motion*™ in a laser marking environment are:

- Adjust the XY table to implement step and repeat marking or locate parts with respect to the marker lens.
- Adjust the rotary axis to locate parts with respect to the marker lens.
- Adjust the height of the marker head to pre-programmed static positions to accommodate marking surfaces at different heights above the tooling plane.
- Rotate a circular cylindrical part while marking to ensure mark completely wraps around part with no distortion or areas out of focus. This feature can be used with a basic implementation ("chuck on a stick") or an advanced implementation which includes a stage with gearing and index (home) sensor.

The LMF family integrated *Marker Motion*™ system allows the user to control up to four stepper motors with integrated controllers using a the *WinLase* software interface for easy configuration and control without extensive hardware requirements. The system consists of two major components:

- The LMF fiber laser marker running *WinLase* software is responsible for controlling motors and issuing serial commands to the individual motors and supplying power and safety control features. Markers configured as 8-79-xBx-xxx are capable of performing motion in API standalone jobs.
- The motors themselves with integrated serial communication hardware, encoders, and sensor inputs.

The fiber laser marker is currently capable of controlling motion in streaming implementations and standalone modes depending on marker configuration. Various cable sets and hardware accessories are available to meet functionality requirements for each of the X, Y, Z, and rotary axes.

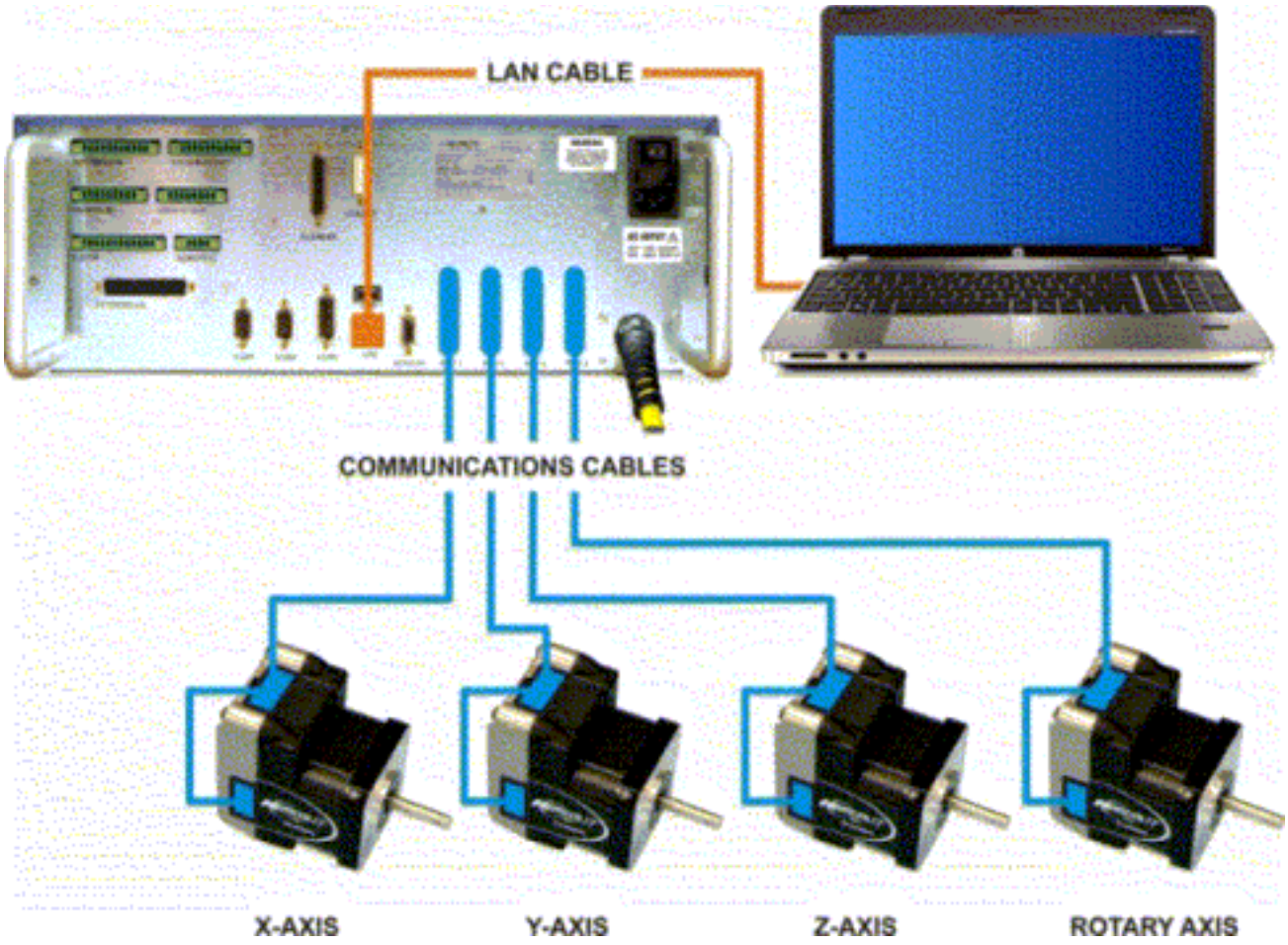
The motors selected are from the SCHNEIDER ELECTRIC MDrivePlus family, and communication between motors and laser marker is conducted using the SCHNEIDER ELECTRIC communication protocol.

The entire SCHNEIDER ELECTRIC MDrivePlus MDI1 family of motors is supported. Each motor communicates with the laser marker using the RS-422 protocol in a daisy-chain or parallel configuration. Power is supplied to each motor directly from the power supply.

General Motor Specifications

Parameter	Specification	Comment
Minimum reliable position step size (10% RMS error)	With Miyachi Unitek linear stage utilizing 4 mm per revolution C5 screw pitch: 10 microns	Specifications validated on 8”x8” travel XY stage.
	Rotary direct drive: +/-5 arc-min in open loop mode, +/- 7 arc-min closed loop resolution	
	Rotary with standard Miyachi Unitek 20:1 5” rotary stage: +/-10 arc-min	
Homing, index, and bi-directional move Repeatability	With linear stage, 4 mm/rev C5 screw pitch: 10 microns;	Specifications validated on 8”x8” travel XY stage.
	Rotary direct drive: +/-5 arc-min	
	Rotary with standard Miyachi Unitek 20:1 5” rotary stage: +/-7 arc-min	
Standard Torque	SCHNEIDER ELECTRIC standard torque	Motor dependent, see SCHNEIDER ELECTRIC specifications to select correct motor
Power supply capacity	24VDC 2A RMS, 4A peak per motor for coil drive 200mA maximum per motor for motor logic backup supply	Allocate 4A per motor maximum, regulated switching power supply. Max torque of NEMA23 single length drive at 1500rpm and 24V is 9 N-cm. 4A peak will drive single or double length motors
Sensor inputs	Linear stage: adjustable end limits; adjustable home limits; Rotary state: adjustable home limits; Sensors integrated with SCHNEIDER ELECTRIC IO functions	Standard with MDrive23Plus motor
	Rotary direct drive index mark	Standard with MDrive23Plus -EQ motor
Motor slewing speed range, settable through WinLase	1 – 1500+ rpm	Motor maximum 1750rpm
Slewing acceleration	Motor torque / load limited	From SCHNEIDER ELECTRIC motor specifications
Operating ambient temperature and humidity	0-40 deg C, 5-95% RH non-condensing	

Marker Motion™ System Connection Diagram – Stepper Motors



Section II: Fiber Laser *Marker Motion*™ Features

System Specifications

PARAMETER	SPECIFICATION	COMMENT
1-4 Axis Capacity	Full performance using SCHNEIDER ELECTRIC MDrivePlus motors in NEMA23 size at single or double coil length.	
Certifications	Incomplete Machine under Machinery Directive. Meets CE LVD and EMC specifications.	Maintaining certification requires approved cable harnesses and implementation techniques.
Safety	Controlled by fiber marker emergency stop and interlock circuitry.	Emergency stop deactivates motor coil but motor logic and encoders remain powered up. Motors retain position and configuration parameters after fault or emergency stop. Emergency Stop safety meets same performance level and category as rest of machine. Motion prevention with door open is implemented to lesser standards. Protecting users from machine motion is the responsibility of the final integrator.
Connectivity	Each motor connects to fiber marker and stage limits if applicable with Miyachi Unitek marker motion cables.	
Number of axes	1, 2, 3, or 4	Axes can be configured with any combination of linear or rotary stages.
Standard motor configuration	RS-422 controlled SCHNEIDER ELECTRIC MDrivePlus with optional rotary encoder feedback.	SCHNEIDER ELECTRIC MDrivePlus family with RS-422 motion control interface, part numbers beginning with MDI1xRLxxxx-xx. Chose -EQ option for encoder feedback if desired. NEMA23 size standard, other NEMA size MdrivePlus motors are supported with appropriate external power supply.

Section III: Standard Cable Harness Configurations with Part Numbers

Introduction

- The user should select appropriate standard cables depending on the application.
- Motors and cables are selected separately
- Use one cable per axis
- Each cable can interface directly with Miyachi Unitek standard stage limits

Part Numbers – LMF with *Marker Motion™*

P/N 10-501-02-01 Motor, NEMA23, with Encoder

P/N 10-502-02-xx

Description	Tab Number	Length	
Laser Marker to Motor Cable	-01	3m	One per axis. Choose length as necessary.
	-02	5m	One per axis. Choose length as necessary.
	-03	8m	One per axis. Choose length as necessary.

Section IV: Connection Using Customer-Supplied Stages

If desired, or necessary due to the system limitations or customer requirements, it is possible to use your own motion stages instead of those supplied by Amada Miyachi America. In this case it is necessary to wire the limits to the motors correctly. Standard motors are NEMA23 form factor.

It is the user's responsibility to take all necessary precautions to make sure the user supplied equipment is safe to use and does not damage the fiber marker or motion systems. Damage to the fiber marker system or motion hardware caused by user implemented non-approved systems is not covered by the fiber marker warranty. If in doubt, please use the Amada Miyachi America supplied fiber marker motion control power supply.

WARNING

Do ***not*** hot plug the motors (plug or unplug them while voltage is present). Doing so will damage the motors.

Do ***not*** ground the aux logic supply at communications ground or the drive will be damaged. Ground aux logic at power ground ***only***. There is a 100Ω resistor between communications ground and the motor common to prevent ground loops. Grounding any signal other than communications may damage the drive.

Do ***not*** unplug motors with the system power on. Doing so *will* destroy the motor.

Motor Input Wiring

Use the following table to wire to the Limit connector attached to the 10-502-02-xx cable.

Pin Number	Function	Description
1	+24V	Power for limits
2	GND	Ground for limits
3	N/C	Do not connect
4	Home Limit	Motor I/O 1, Axis Home
5	N/C	Do not connect
6	N/C	Do not connect
7	CW Limit	Motor I/O 3, Axis CW Limit
8	N/C	Do not connect
9	CCW Limit	Motor I/O 2, Axis CCW Limit

Stage Limit Wiring

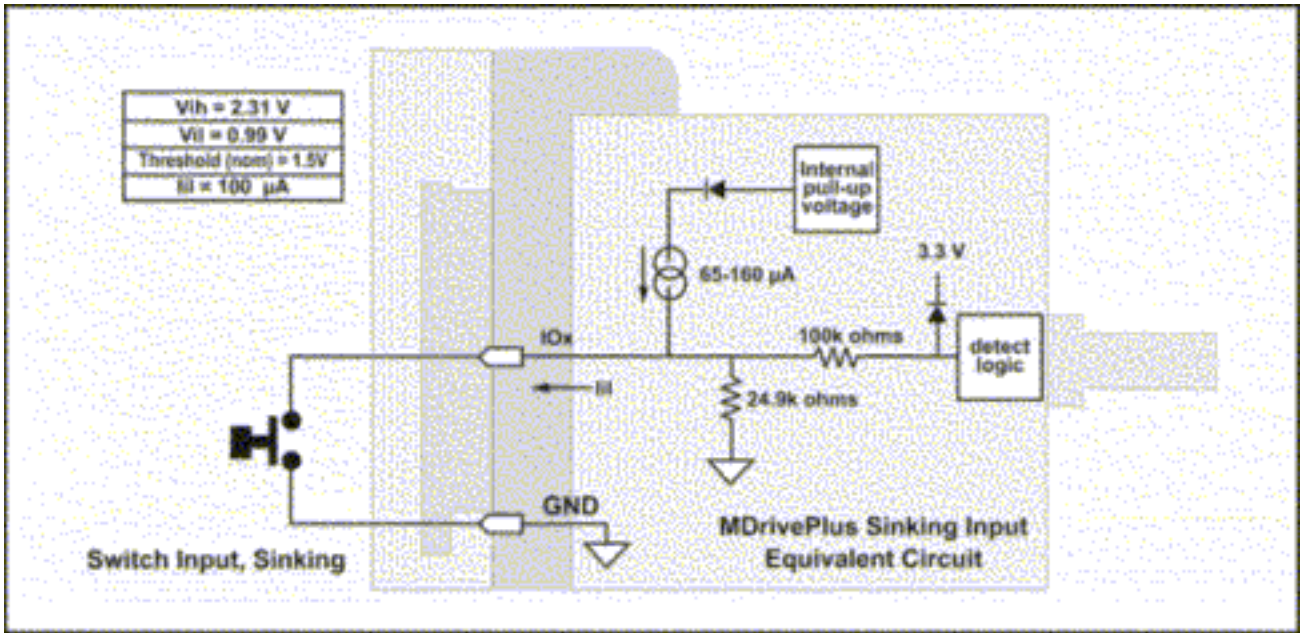
The SCHNEIDER ELECTRIC MDrivePlus motors are very flexible and have four input bits that are commonly used for stage limit input signals. These can be wired in a "pull up", "pull down", or "floating" configuration. Typically optical flag sensors are used. When powering external sensors be cognizant of the behavior of the power source used to bias the sensor in all situations.

It is recommended to use limits which are "low active" or "pull down" since a break in the limit cable will cause the limit to be detected for safety.

The *WinLase* software package interprets the inputs as follows:

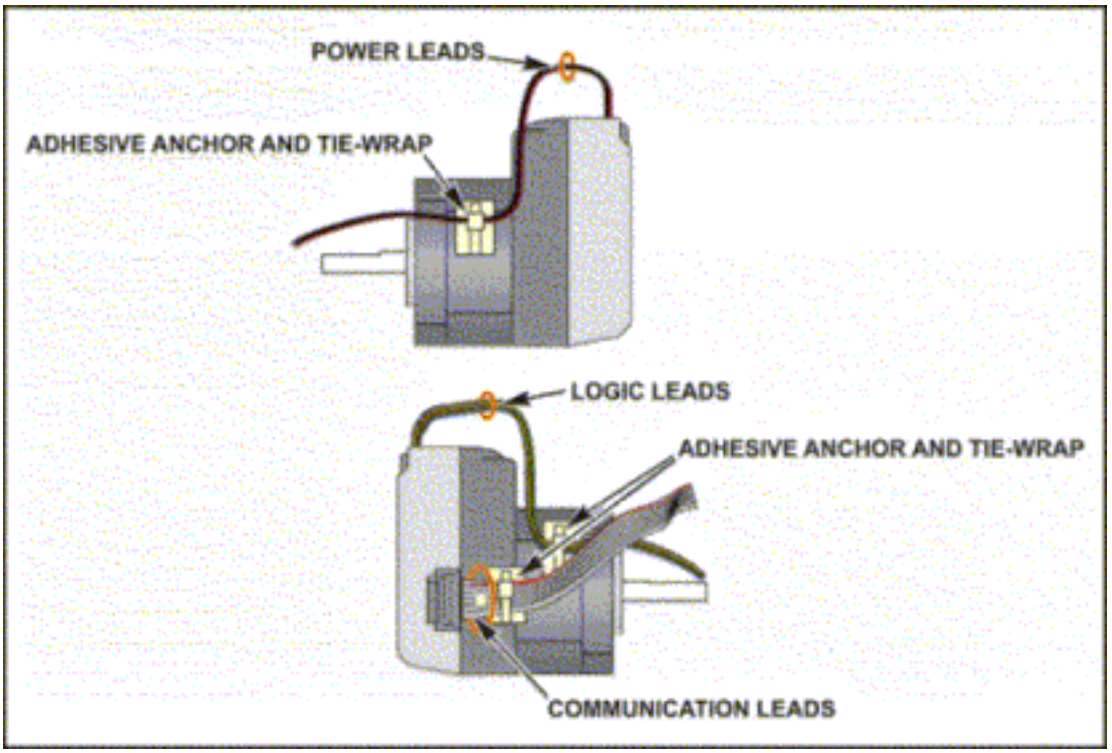
- I/O 1: Stage Home Limit Flag Input
- I/O 2: Stage CCW Flag Input
- I/O 3: Stage CW Limit Flag Input

EXAMPLE: Sinking input optical flag sensor using +24V from the auxiliary logic supply:



Securing Power and Communications Leads

NOTE: Motor connections ***must*** be strain relieved or failure will occur.



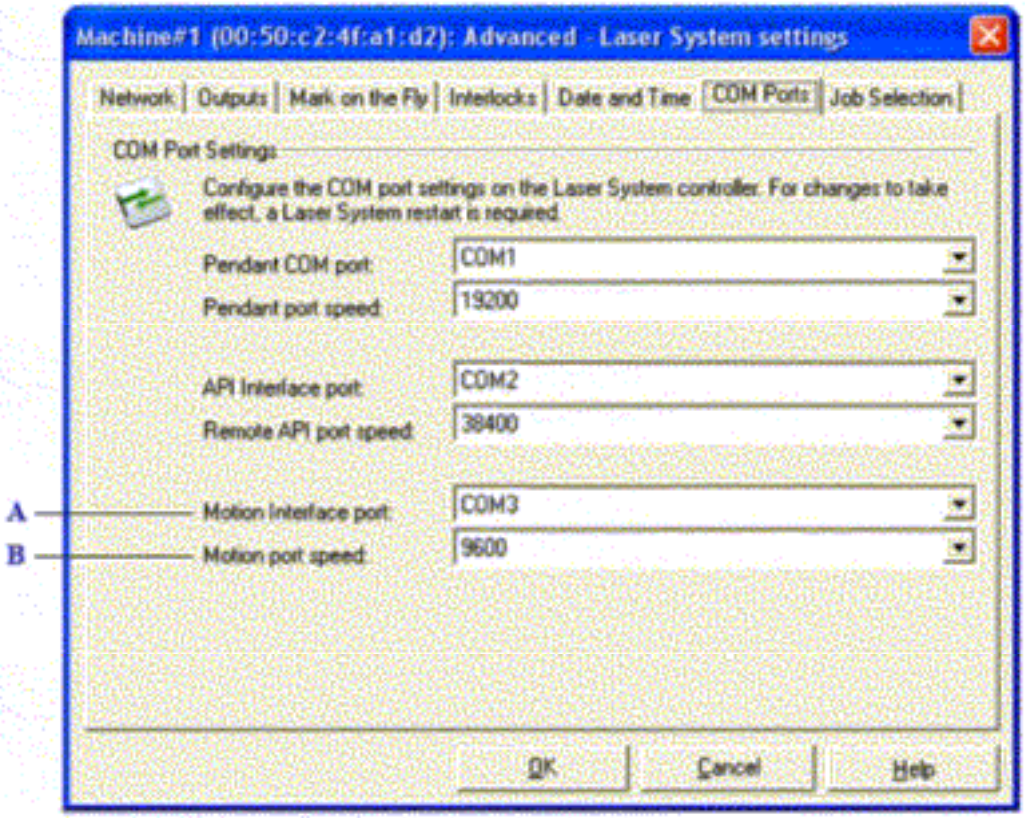
Section V. Software Configuration

System Settings

Follow these steps to configure WinLase LAN for *Marker Motion*™. More detail on motion configuration is described after the configuration steps below.

First, configure the machine COM port. The LMF fiber laser marker by default communicates over COM3 at 9600 baud.

1. Create a connection to the laser marker.
2. In the Laser System **Viewer,** right-click on the connected LEC-1 device, and select **Settings**.
3. Select the **COM Ports** tab. The COM ports page appears.
4. Set the **Motion Interface Port** to **COM3** and **Motion Port Speed** to **9600** baud.



The LEC-1 COM Ports settings page

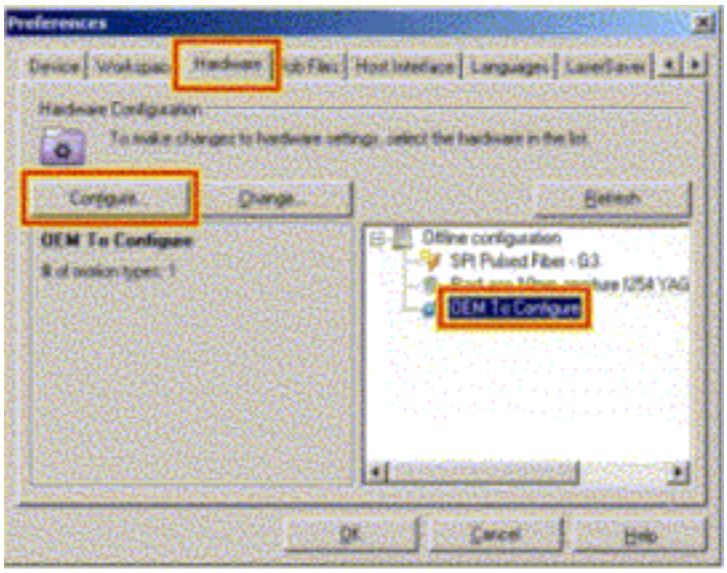
A Configure which COM port to use on the LEC-1 device for motion control.

B Configure the port speed (default = 9600).

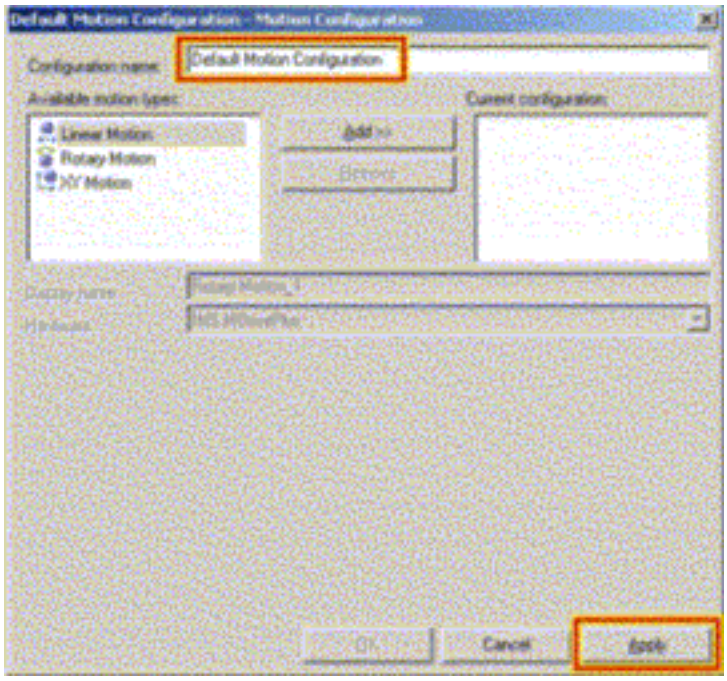
5. In *WinLase*, navigate to the **System** pull-down menu, then select **Preferences**.

7. Select the **Hardware** tab.
8. Select **OEM to Configure** (or similar) and click the **Configure** button.

   NOTE: Make sure that you follow these steps for the motion settings first under the specific connected laser as well as under **Offline Configuration**. (**System > Preferences)**



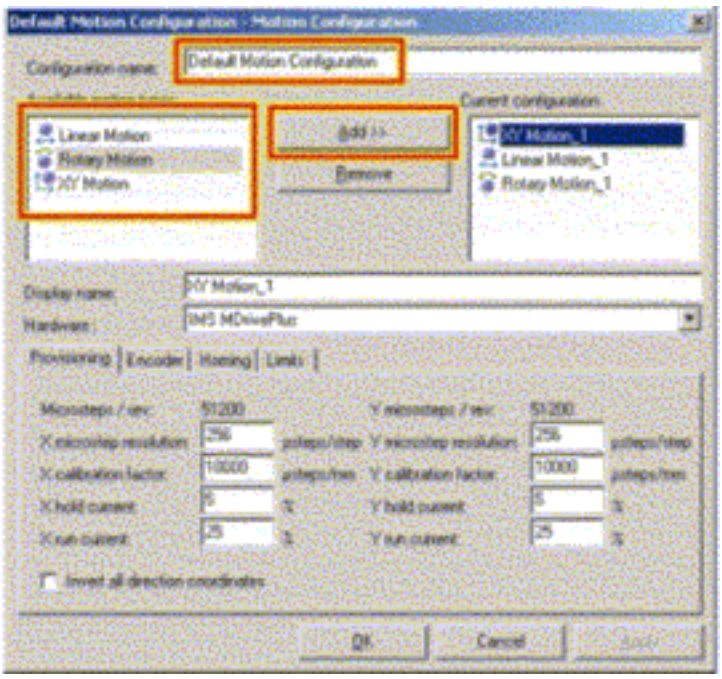
9. Type **Default Motion Configuration** in the **Configuration Name** field as shown on the right.
10. Select **Apply**.



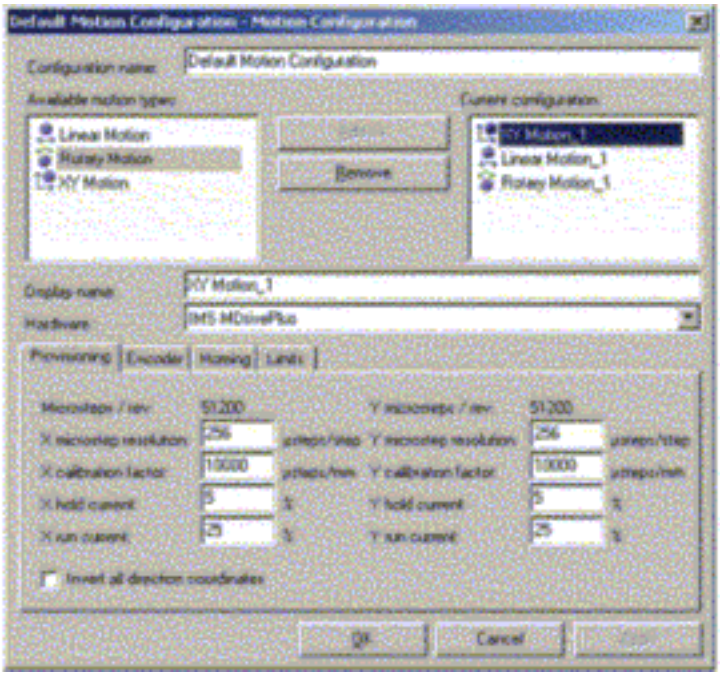
11. Configure the axes in the order desired by selecting from "**Available Motion Types**" and clicking "**Add**" until up to 4 axes are populated.

    Axes will be assigned a letter from the top of the list beginning with A. For example, a system with a XY table, a Z-axis, and a rotary would look like the configuration shown on the right. The letter assigned can not be seen in this window.

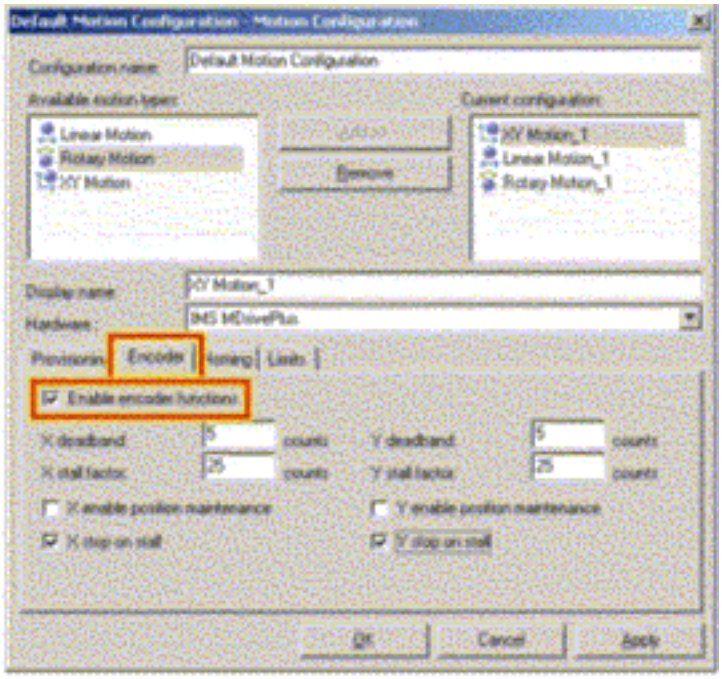
    CAUTION: Do not add more axes to this list than you actually plan to connect as errors will occur if not all axes can be communicated with.



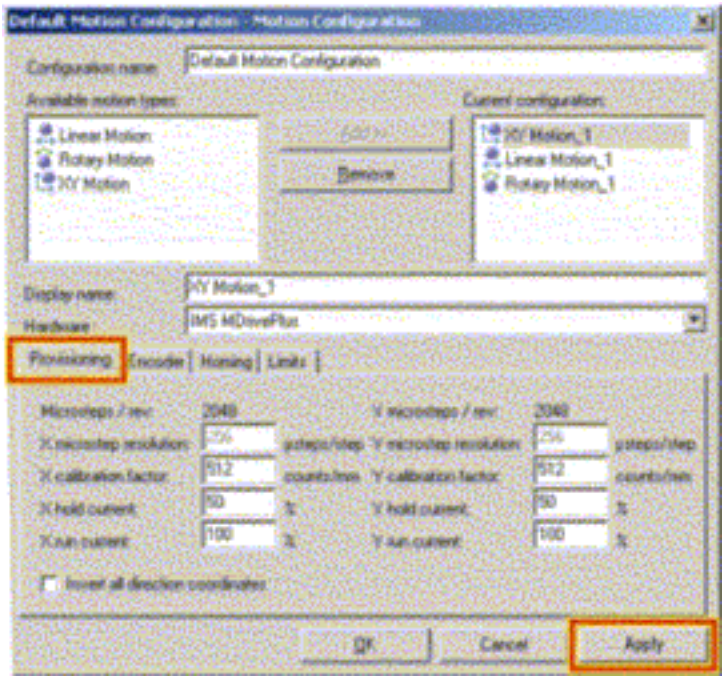
11. Select the first axis from the list **XY Motion_1** as shown on the right and rename it if desired. Make sure that the **Hardware** option is **SCHNEIDER ELECTRIC MDrivePlus**.



12. Select the **Encoder** tab as shown on the right. If the motor is equipped with an internal encoder (-EQ as the last two characters in the motor part number on the motor sticker), check **Enable Encoder Functions**.
13. Set the **deadband** to 5 and the **stall factor** to 25. Do ***not*** enable position maintenance.
14. Select **Apply**.



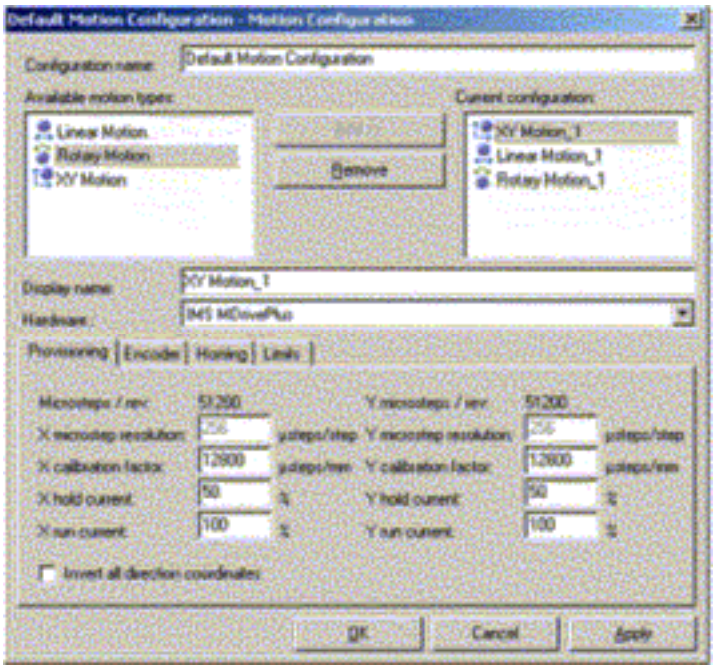
15. Select the **Provisioning** tab.
16. Set the calibration values to the correct settings. With the encoder enabled, one revolution is 2048 steps. Note that these values will depend on the stage selected. Default values provided are for common Miyachi Unitek XY stages.
17. Select **Apply** when finished.



Linear Axis Example:

For a 4mm/revolution lead screw (standard Miyachi Unitek XY table) you would set the µsteps/mm value to 512 for an encoder enabled system or 12800 for an open loop stepper system without encoders. Set the **Hold** current to 50% and Run Current to 100%.

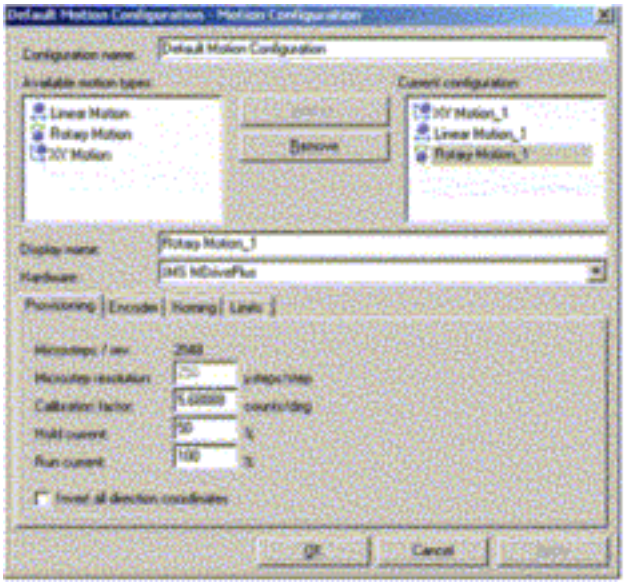
The illustration in Steps 16 and 17 (above) show the Encoder example, the illustration on the right shows an open loop example.



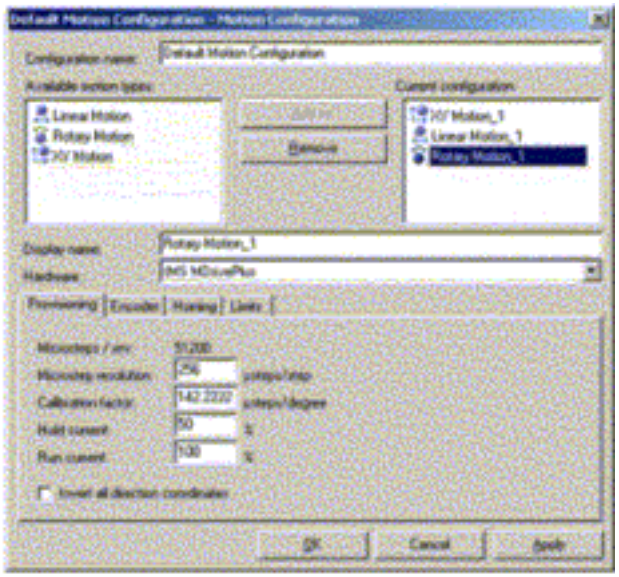
Rotary Axis Example:

For a 4mm/revolution lead screw (standard Miyachi Unitek XY table) you would set the µsteps/mm value to 5.68888 for an encoder enabled system or 142.2222 for a open loop stepper system without encoders. Set the **Hold current** to 50% and **Run current** to 100%.

Encoder Example

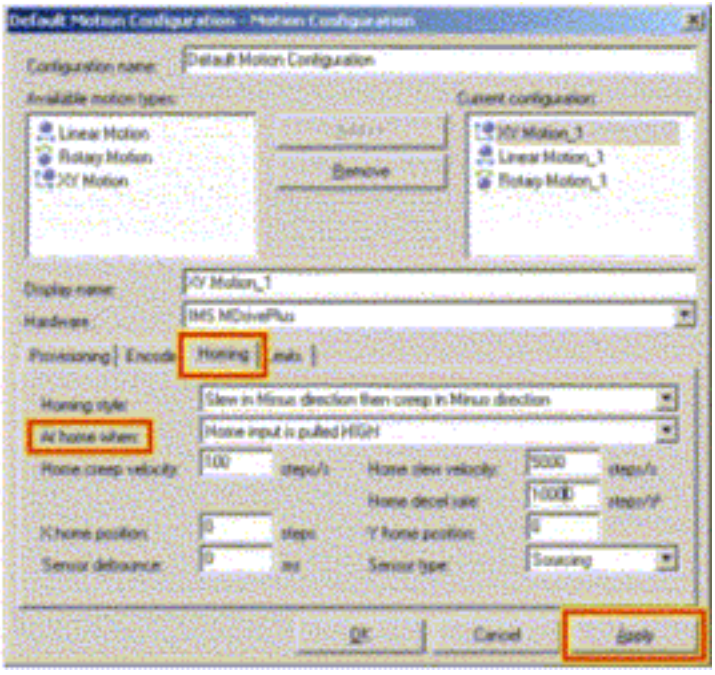


Open Loop Example (Encoder Disabled)



18. Select the **Homing** tab.
19. Set **Sensor Type** to **Sourcing** and set **At home when:** to **Home Index is pulled High** as shown on the right using the pull-down menu if using a stage with limits, or **on Encoder Index mark** for a chuck-on-a-stick or similar application with no physical home limit.

    Set the remaining parameters as shown on the right for the situation when nothing is connected to the motor shaft. If a stage is connected you will need to set these parameters depending on the individual stage's requirements.

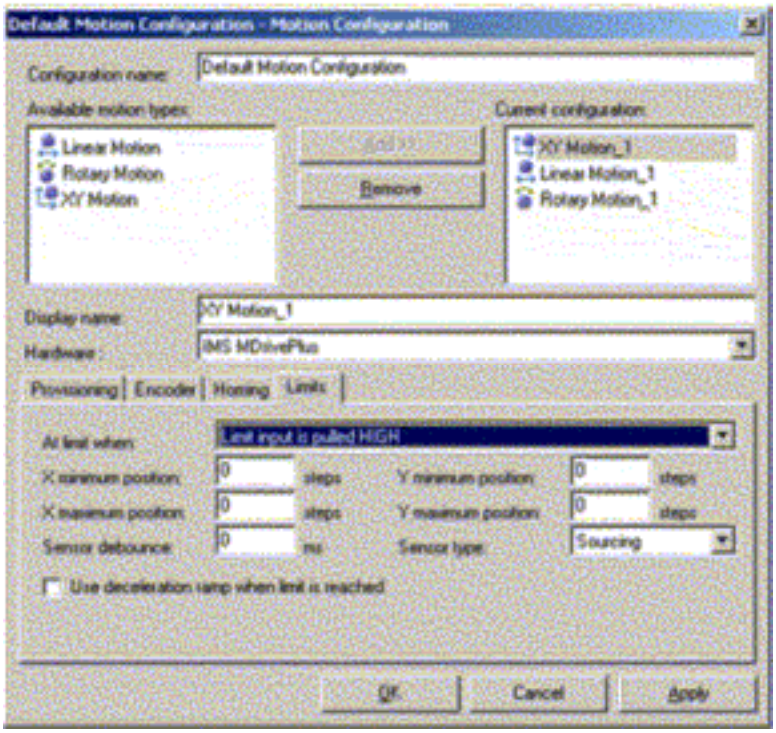


20. Select **Apply** when finished.

21. Select the **Limits** tab.
22. Set "**Sensor type:"** to **Sourcing** and set the **At limit when:** to **Limit Input is pulled HIGH.**
23. Set the soft limits (minimum/maximum positions) shown on the right to a very negative and very positive number, like

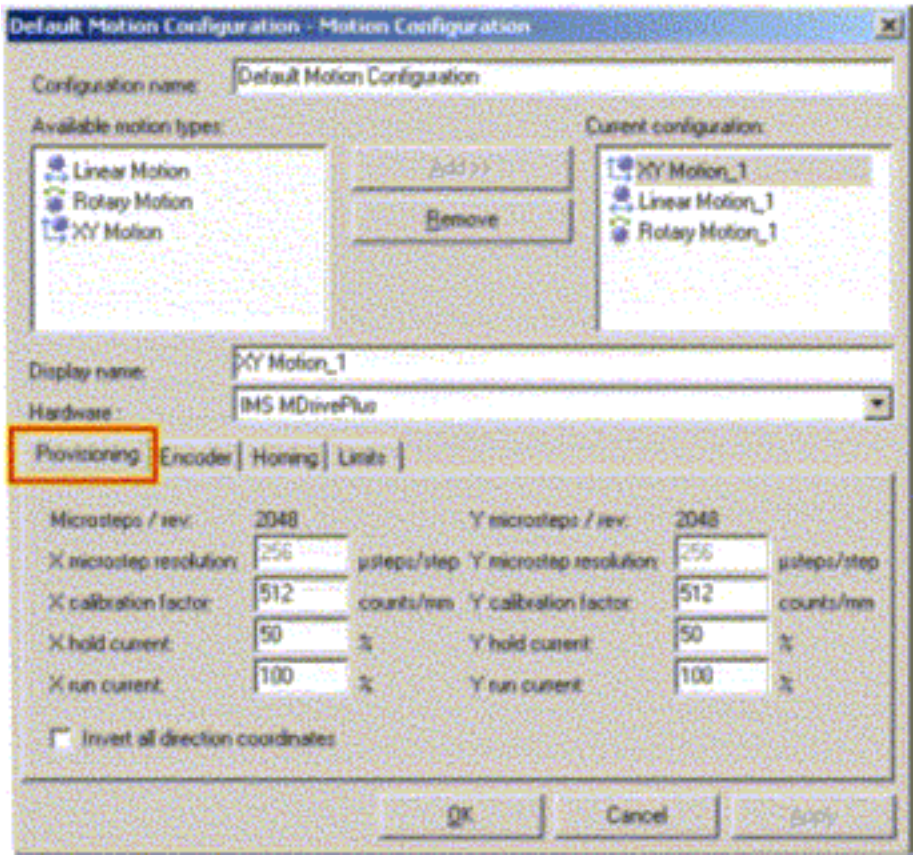
    -100000000 and **100000000.**

    This value can be set more precisely if the user desires a software limit to motion stage position, but for most applications it is best to set these values outside the range of the stage so that the optical sensors provide the limits of travel.



24. Repeat this procedure for all axes, and for the **Offline Configuration** settings in the hardware tab of the **System > Preferences** window.
25. Click **OK** to exit the Motion Configuration screen.

Provisioning Tab



Provisioning **Tab**

Microsteps/rev

The number of microsteps per motor revolution is displayed to the user. This value is calculated differently based on the motor's encoder setting. If the encoders are disabled (open loop mode) the microsteps per revolution are the (Microstep Resolution)*200 steps/rotation. This gives 51200 microsteps per resolution at the highest microstep resolution setting. If the encoders are enabled there is a fixed value of 2048 microsteps/rev based on the encoder resolution.

Acceptable values in open loop mode are: 200, 400, 800, 1600, 2000, 3200, 5000, 6400, 10000, 12800, 20000, 21600, 25000, 25400, 25600, 36000, 40000, 50000, and 51200.

Microstep Resolution

This value selects the level of microstepping in open loop mode. Acceptable values are 2, 4, 5, 8, 10, 16, 25, 32, 50, 64, 100, 128, 200, 250, 256.

Other specialty values are:

18, corresponds to .01 degrees / µstep
108, corresponds to 1 arc minute / µstep
127, corresponds to .001mm / µstep

When encoders are enabled 256 microsteps/step is the only option.

Calibration Factor

This parameter selects the number of microsteps per unit distance at the part. It will be in units of µsteps/degree, µsteps/mm, µsteps/inch, etc. This multiplier is used to accommodate for any gearing changes and calibrate the system to the specific motion stage setup. For a typical 1:1 rotary stage, for example, this value will be the same as the number of microsteps/revolution divided by 360, or 142.2 µsteps/degree. A 20:1 high-precision stage would require this value to be multiplied by 20 to get the correct calibration factor. You know you have the calibration factor correct when you can move the stage a given distance and the actual traveled distance matches the given distance.

Hold Current

This parameter sets the motor hold current as a percentage of total current. **Hold current** is the amount of current that the motor uses to keep the motor static between movements. A stage which loads the motor at idle (like a Z-axis stage with a marker head attached) requires more current to maintain its current position, and thus a higher hold current value than a stage with lower load. A higher **Run current** uses more electricity and causes the motor to heat up, so it is important to balance the need for a stiff motor against the additional heat introduced into the system and the movement duty cycle. At high **Run current** values the motor can become hot to the touch.

Run Current

This parameter sets the motor run current as a percentage of total current. **Run current** is the amount of current the motor uses during movements. Faster movements, higher accelerations, and high stage loads increase the necessary run current. Set the **Run current** to a setting that allows the desired speeds and accelerations to run without stalling, but do not set the run current higher than is necessary to prevent the motor from becoming unnecessarily hot. It is important to balance the speeds and accelerations with the run current and duty cycle to find the optimal setting.

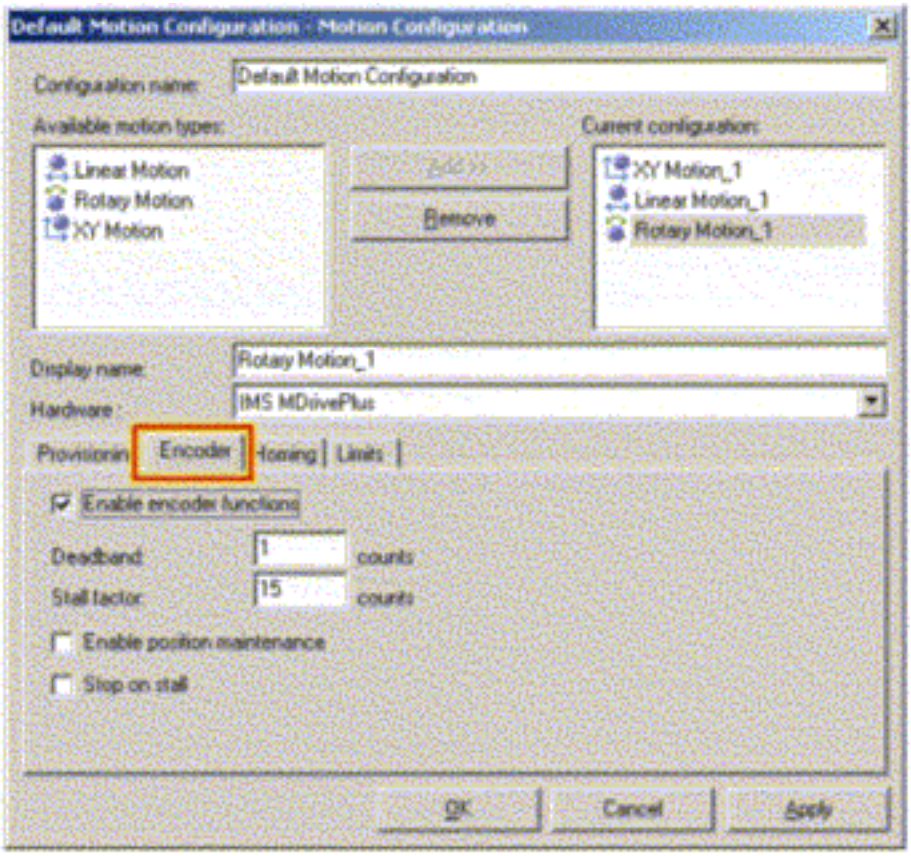
Invert All Direction Coordinates

This parameter reverses the direction the motor traverses with respect to the positive axis of the *WinLase* mark field. This setting is determined by the stage configuration.

Program Hardware

This button starts the hardware programming wizard used to download configuration data into SCHNEIDER ELECTRIC MDrivePlus motors. Please see the *Motor Configuration and Programming* section of this manual for details.

Encoder Tab



Encoder **Tab**

Enable Encoder Functions

Enables the built-in encoders on equipped drives. Choose drives with the **-EQ** option to get the built-in 2048 line encoder.

Deadband

This parameter sets the number of counts that determine the width of the "dead band". In closed loop feedback mode the motor will compare the requested position with the encoder position and automatically make adjustments to ensure that the motor is in the correct place. The deadband is the zone around the requested point at which the motor considers it "close enough".

It is important to balance the desire for a tight tolerance against the physical parameters of the stage and the extra current required to maintain a location against static load. This value is only meaningful when **Enable position maintenance** is checked.

Stall Factor

The stall factor sets the number of counts required for a stall error to be reported to the laser marker. If the motor's current position is different from the current requested position by greater than the number of counts set in this stall factor the motor will error out and stop moving. This is useful to minimize damage in the case of motor crashes, part jams, or other issues. It is also useful in position maintenance mode because it causes a fault if the maintenance is unable to maintain the desired position.

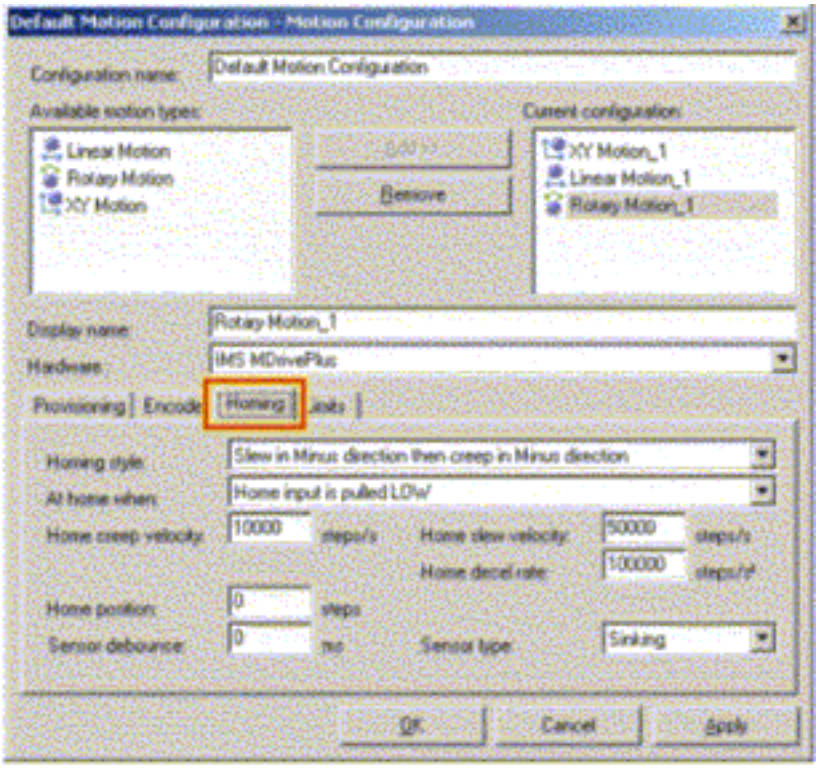
Enable Position Maintenance

This checkbox turns on the closed-loop position maintenance algorithm. The motor will compare the requested position with the current position reported by the encoders and make any movements necessary to correct errors within the tolerance of the deadband. If the position errors are so great that they exceed the stall factor an error will be reported and the motor will stop moving if **Stop on Stall** is checked. When position maintenance is enabled it is important to properly set the deadband and stall factor to reasonable values, and ensure that the run current is high enough to allow the stage to move if position maintenance is required. This option should not be selected when safety is a concern since the motors can operate autonomously attempting to maintain their position and potential injury could result. Typically this option is only enabled for low torque motors in applications that do not represent a danger of pinching or other human danger.

Stop on Stall

This checkbox causes detected stalls to stop the motor from moving until the motor is reset. Use this option to protect the system from damage in the case of stage crashes or part jams.

Homing Tab



Homing **Tab**

Sensor Type

This pull-down menu is where the wiring style of the limit sensors is configured. Choices are sinking, where the limit I/O pin at 24V is pulled down to 0V to activate, and sourcing where the limit I/O pin at 0V is pulled up to 24V to activate.

Changing this option will change some choices in the **At home when** pull-down menu to accommodate the differing polarity of sinking and sourcing modes.

Home Creep Velocity

This parameter sets the "creep" velocity in steps/seconds. The homing stage first slews until it finds the appropriate limit, then creeps in a manner determined by the setting of the **Homing Style** pull-down menu. Set this value to a low setting for accurate homing.

Home Slew Velocity

This parameter sets the "creep" velocity in steps/seconds. The homing stage first slews until it finds the appropriate limit, then creeps in a manner determined by the setting of the **Homing Style** pull-down menu. Set this value to a slow to medium setting for accurate homing. Slower home slews slow homing time, while longer home slews require increased creep distances due to stage inertia.

Home Decel Rate

This parameter sets the deceleration rate of the stage during the homing process in units of steps/s^2. This is the rate at which the stage will decelerate when it finishes its slew and creep movements. Select a high value for accurate and quick homing.

Home Position

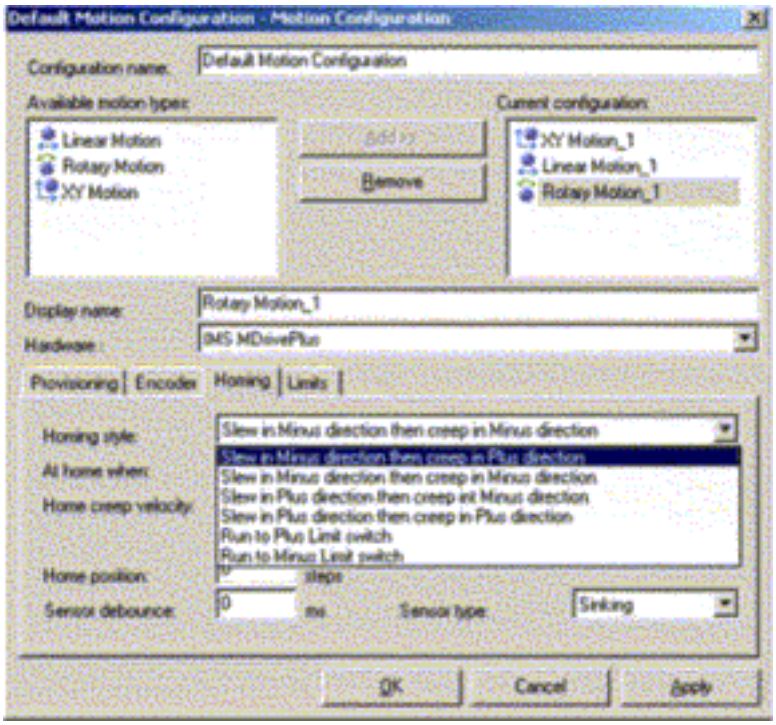
This parameter sets the offset position from the home flag in steps. This value can be positive or negative. When the stage has reached home it will move the desired number of steps at the home creep speed. This function is particularly useful when trying to execute the same *WinLase* LAN job on two different machines. Each machine will have a different necessary home offset, which can be corrected using this parameter.

Sensor Debounce

The sensor debounce is a time in milliseconds required from when a limit switch is activated until the homing routine continues to the next phase. This is intended to compensate for switches with a lot of electrical "bounce" or instability on transition. Most mechanical switch systems require a debounce. 0ms is a typical setting for solid state optical sensors.

Homing Style

The homing style pull-down menu selects the homing behavior from a number of different options as described below. In all cases, if the high or low limit is reached before the home limit is reached the slew direction will reverse, but the creep direction will remain the same.



Slew in Minus direction, then creep in Plus direction

The stage will slew in the negative direction until the limit switch is reached, then creep in the positive direction until the edge of the limit switch is found.

Slew in Minus direction, then creep in Minus direction

The stage will slew in the negative direction until the limit switch is found, then creep in the minus direction across the length of the switch until the edge of the switch is found.

Slew in Plus direction, then creep in Minus direction

The stage will slew in the positive direction until the limit switch is reached, then creep in the negative direction until the edge of the limit switch is found.

Slew in Plus direction, then creep in Plus direction

The stage will slew in the positive direction until the limit switch is found, then creep in the positive direction across the length of the switch until the edge of the switch is found.

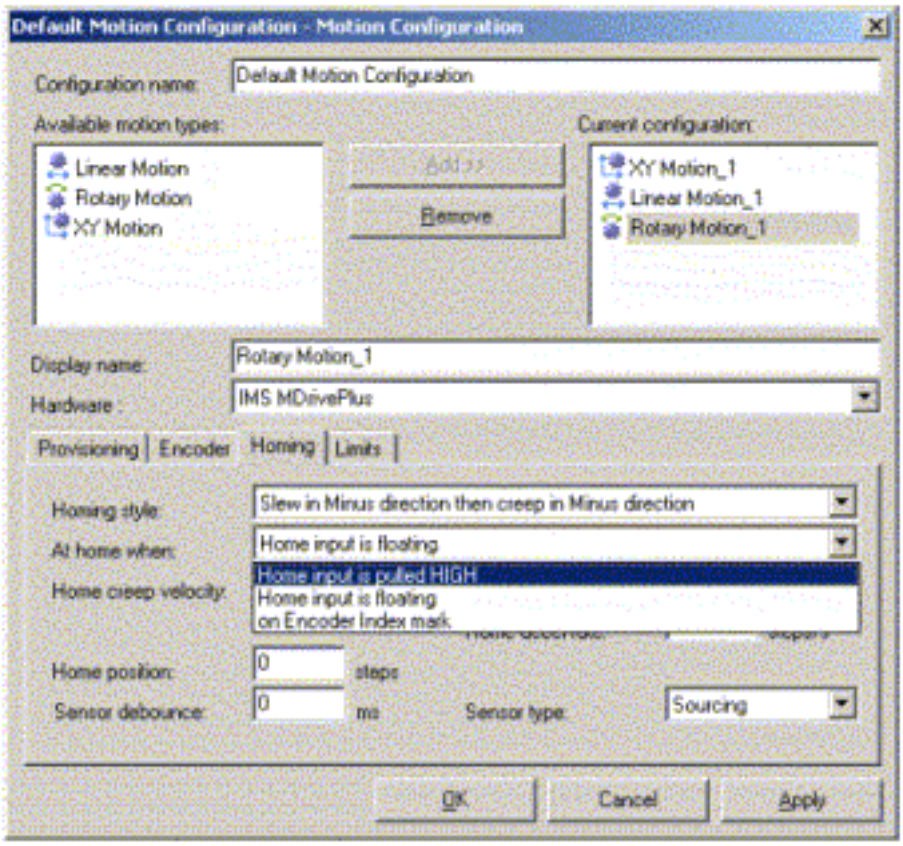
Run to Plus Limit Switch

The stage will run to the positive limit switch, then set the home position.

Run to Minus Limit Switch

The stage will run to the negative limit switch, then set the home position.

At Home When



At Home when **Pull-down Menu**

Home input is floating

This selection causes the system to interpret the I/O pin entering a floating state as activation. This option is available in both sinking and sourcing modes.

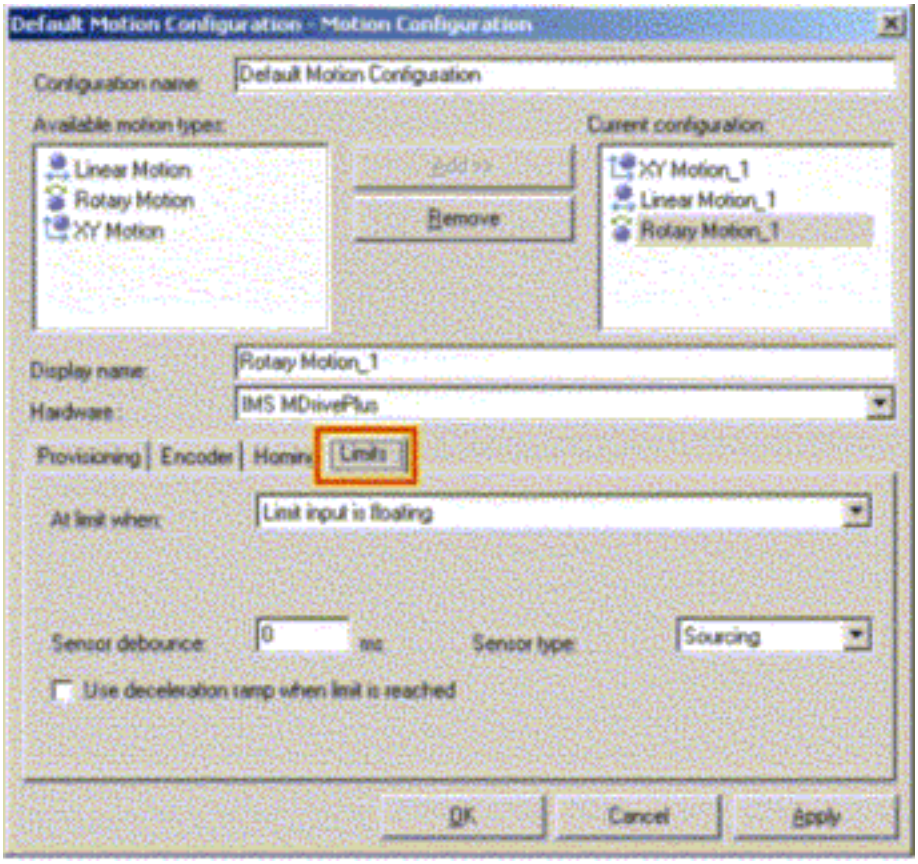
Home input is pulled HIGH (or LOW)

This selection causes the system to interpret the limit input as transitioning from Low to High (sourcing) or High to Low (sinking) as activation. This is the most common setting when using solid state optical limits.

On Encoder Index Mark

This selection causes the system to determine it has reached its home when the motor encoder reaches a built-in index mark. This is very useful for direct drive rotary systems without limit switches. It is important not to use this setting on systems with alternative gearing since there is one home index for every motor revolution which may not correspond to the same home location every time on an arbitrary system. For example, a 20:1 rotary stage would have 20 motor rotations and index indications for every 1 rotation of the actual stage.

Limits Tab



Limits tab

Sensor Debounce

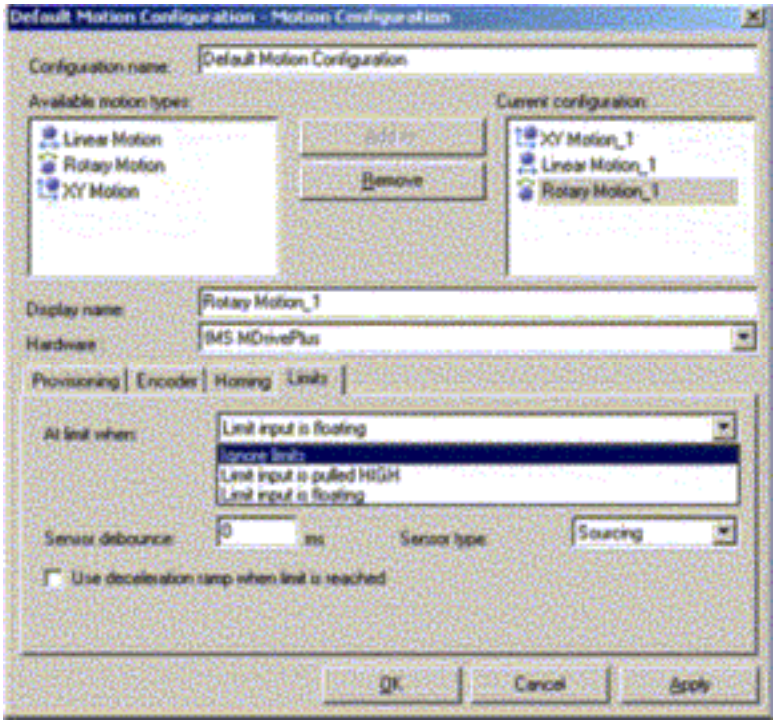
The sensor debounce is a time in milliseconds required from when a limit switch is activated until the limit is considered activated. This is intended to compensate for switches with a lot of electrical "bounce" or instability on transition. Most mechanical switch systems require a debounce. 0ms is a typical setting for solid state optical sensors.

Sensor Type

This pull-down menu is where the wiring style of the limit sensors is configured. Choices are sinking, where the limit I/O pin at 24V is pulled down to 0V to activate, and sourcing where the limit I/O pin at 0V is pulled up to 24V to activate.

At limit when

Changing this option will change some choices in the **At limit when** pull-down menu to accommodate the differing polarity of sinking and sourcing modes.



Limit is floating

This selection causes the system to interpret the I/O pin entering a floating state as activation. This option is available in both sinking and sourcing modes.

Home input is pulled HIGH (or LOW)

This selection causes the system to interpret the limit input as transitioning from Low to High (sourcing) or High to Low (sinking) as activation. This is the most common setting when using solid state optical limits.

Ignore limits

Ignores limit switches. Commonly used for direct drive rotary stages with no limit switches.

Use deceleration ramp when limit is reached

This checkbox causes the motor to decelerate using a ramp profile instead of a constant profile. This setting is used for stages with high inertia that do not respond well to rapid deceleration.

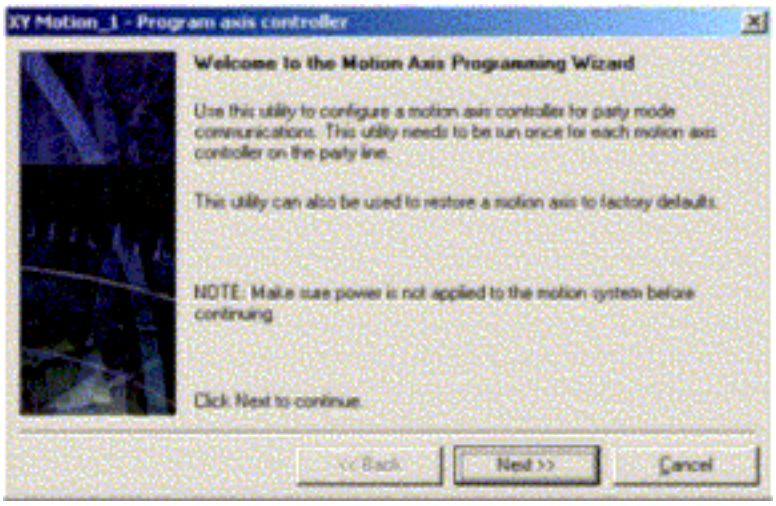
Motor Configuration and Programming

Each axis must be programmed before use. Once the motion configuration is completed, use the "Program Axis" button from the Provisioning window to store the configuration to the motors. Each motor will receive a letter name (A-D) during this process.

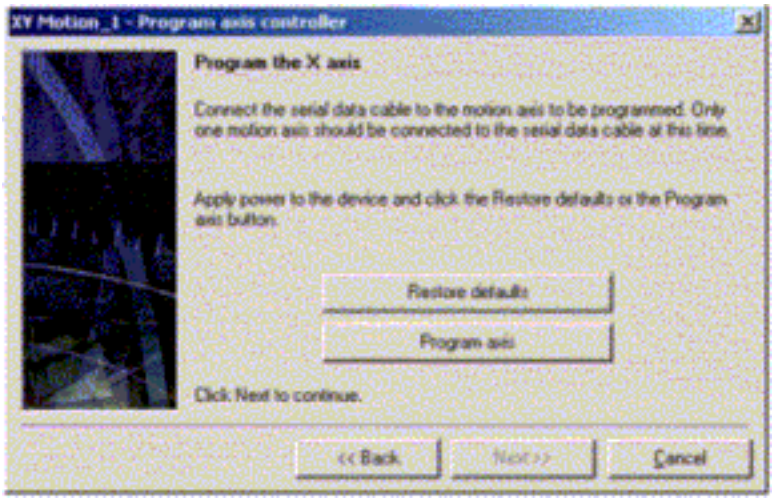
Motors ***must*** be programmed individually and all motor cables except the current one being programmed must be fully disconnected. **Turn the laser marker off before connecting or disconnecting any cables.**

To Program an Axis

1. Select the axis or axes to be programmed in the **Current Configuration** portion of the motion configuration screen and click **Program Hardware**. The program motors wizard will start.
2. Click **Next**.

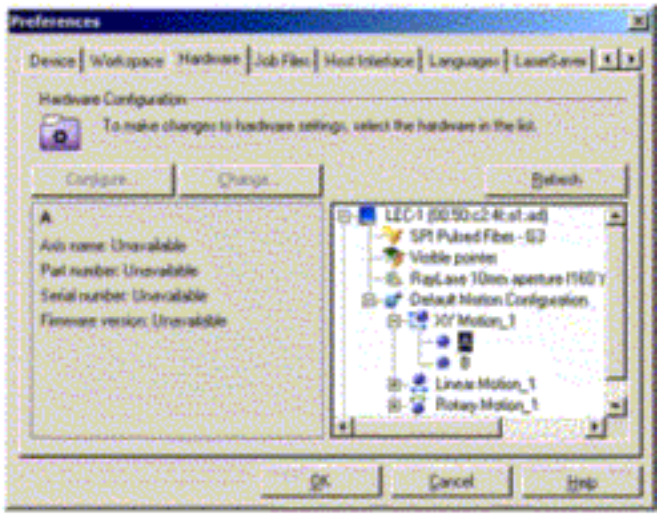


3. Ensure that only the motor you wish to program is connected, then click **Program Axis**. The programming process should take 3-4 seconds.
4. Once the motor has been programmed click **Next** and follow the directions in the wizard.
5. Once all motors have been programmed, return to the **System > Preferences** screen. Verify that the motors have been correctly programmed by looking to see if they are correctly reporting their serial number under the Hardware tab.

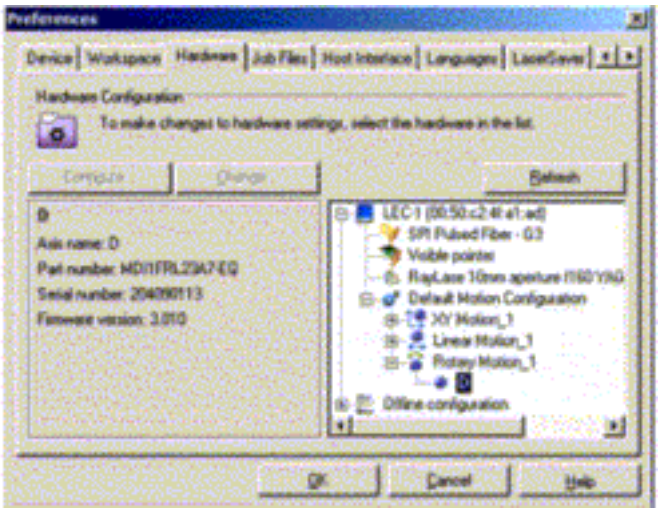


From the settings tree structure, navigate until the motors are visible and show their axis names (**A, B, C,** or **D**). Click on the axis, and see if the **Axis Name**, **Part number**, **Serial number**, and Firmware version are correctly reported. **Unavailable**, gibberish, or **????????** mean the programming has failed. Please check your power cables and communication cables and try again. Gibberish typically means that that more than one motor was plugged in when the programming was attempted.

Axis A: Unprogrammed



Axis D: Programmed and Communicating

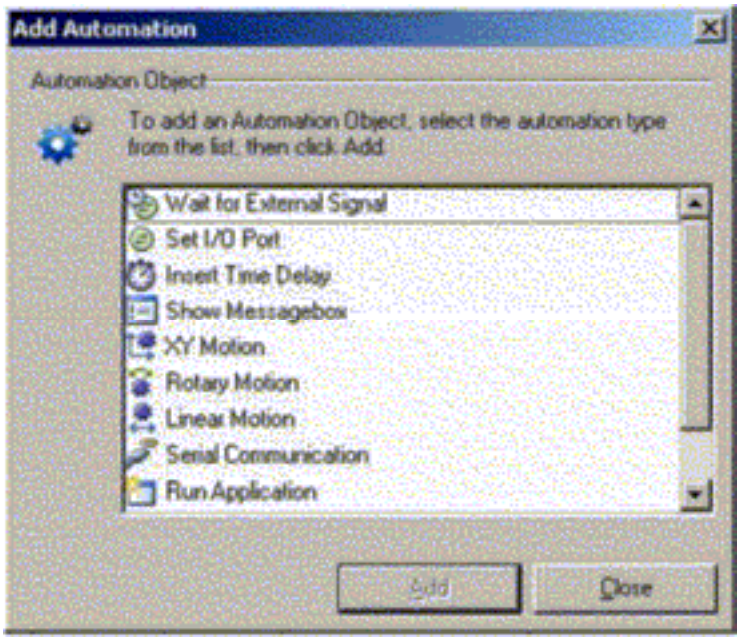


Section VI: Operation

Using System Motion

Adding Motion to a Job

Motion can be added to the job in a few ways. The standard method to add a stage movement is to add a motion object from the Automation toolbox. A motion automation object is added to the *WinLase* job just like any other marking object and has properties that can be changed in the same manner. Some regular objects have motion capability as well. These have a "motion" tab within their properties, and relevant motion settings can be changed within this tab. Please see *Section IV, Advanced Motion Applications* for more details. To run jobs with embedded motion simply use the **Run** command within *WinLase*.



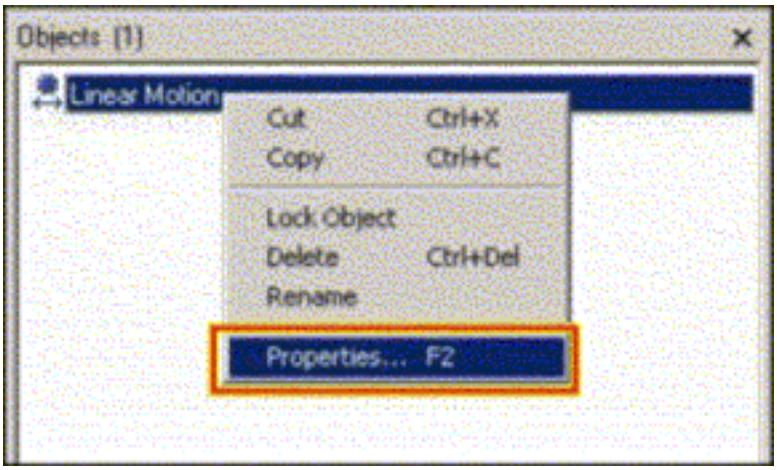
Useful motion automation objects are:

XY Motion	Used to control a 2-axis motion table
Linear Motion	Used to control a 1-axis motion table or Z-stage
Rotary Motion	Used to control a rotary stage

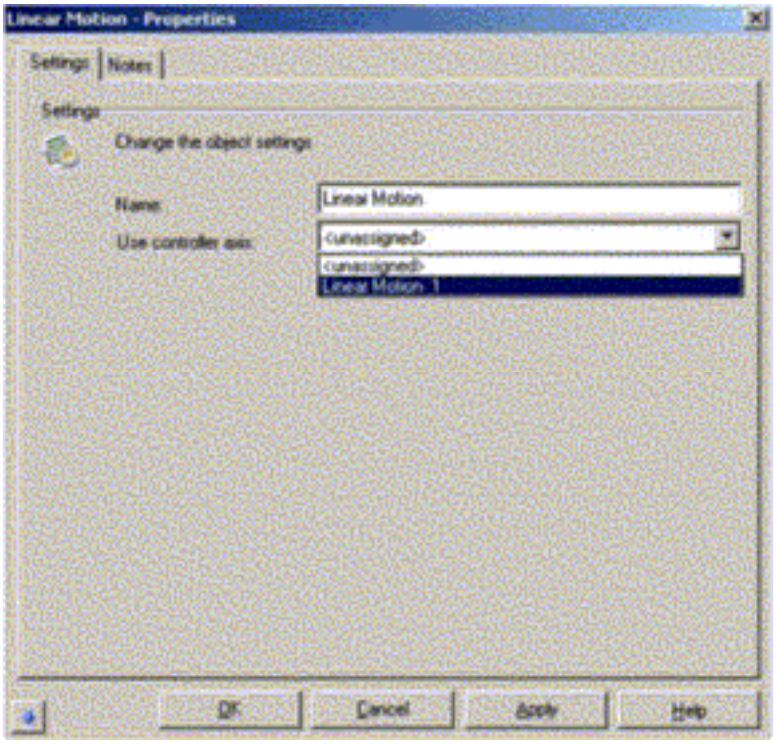
Any number of motion objects can be added to the job. If there are multiple axes of the same type (e.g., two rotary axes) simply add two **Rotary Motion** objects and select the specific axis from within the **Rotary Motion** preferences.

Upon first opening a new motion object it is necessary to select the desired controller axis for the motion object to act upon.

1. To modify motion objects, click on the desired object in the **Object Manager** screen.
2. Right-mouse click and select **Properties**.

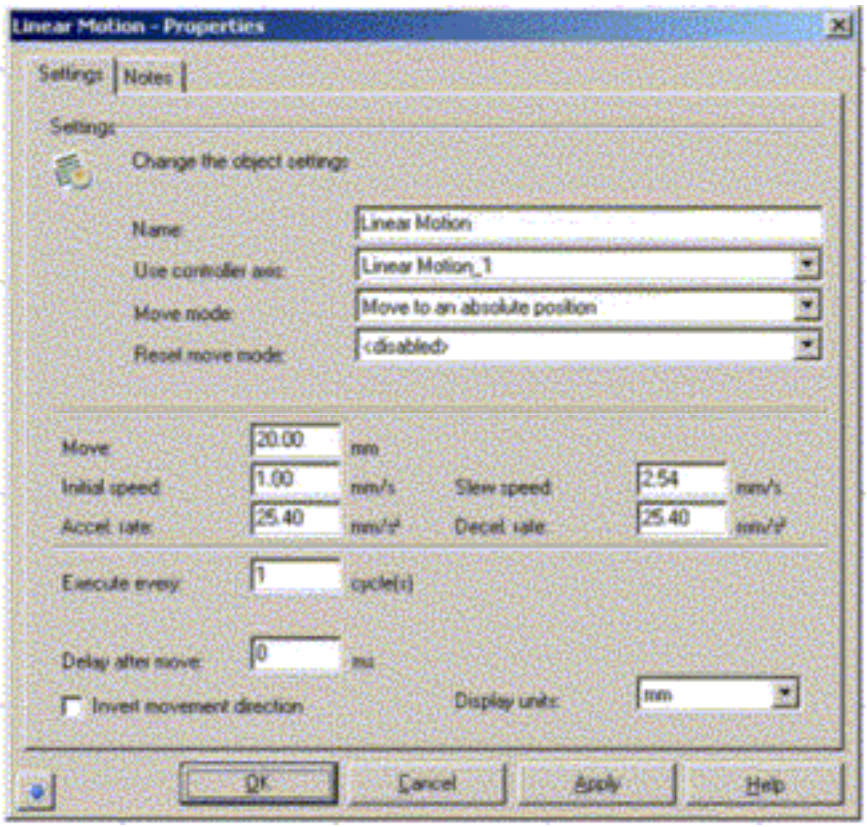


3. When you click on **Properties** the screen on the right will appear.



Motion Objects

Linear Motion Properties



Linear Motion Settings PROPERTY Page

Name: This is the axis name as it appears in the **Object Manager**.

Use Controller Axis: Assign this motion object to a specific hardware axis. There must be the correct motion type configured in the **Motion Configuration**.

Move: The amount to move the axis when performing a normal move.

Initial Speed: The initial speed the axis moves at while accelerating to the slew speed.

Accel. Rate: The acceleration rate of the axis when changing from the initial speed to the slew speed.

Execute every: The object can be configured to skip execution by setting this parameter greater than 1. For example, to execute the object every third job cycle set this value to 3.

Delay after move: The amount of time in ms (milliseconds) to delay after completing the move.

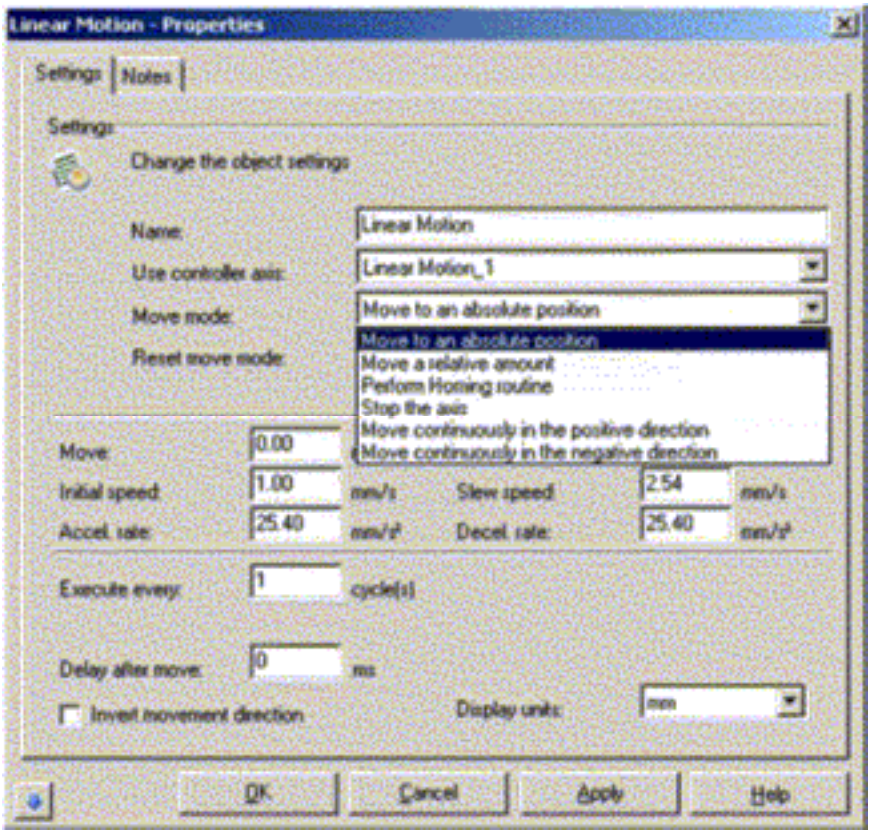
Invert movement direction: When selected all axis movement will be opposite normal movement.

Slew Speed: The maximum speed an axis will attain when executing a move.

Decel. Rate: The deceleration rate of the axis when changing from the slew speed to a stop at the end of the programmed travel.

Display Units: Change the display of all units in the current window. If changed all previously entered values will be converted automatically.

Move Mode:



Select from the following Move Modes:

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

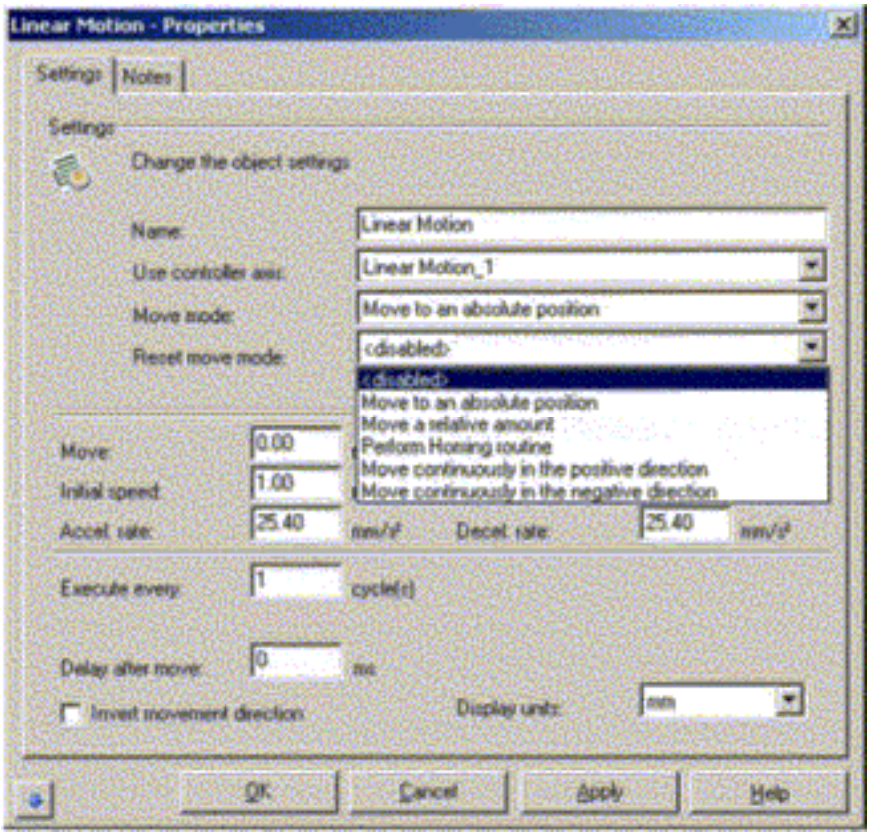
Stop the axis: The axis will be stopped if it was moving as a result of a previous motion object when the object executes.

Move continuously in the positive direction: The axis will move towards the positive limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the negative direction: The axis will move towards the negative limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Reset Move Mode:

A reset condition can be configured so that after the specified number of cycles the axis will perform a rest function.



Select from the following Reset Move Modes:

<disabled>: Disable the reset functionality.

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

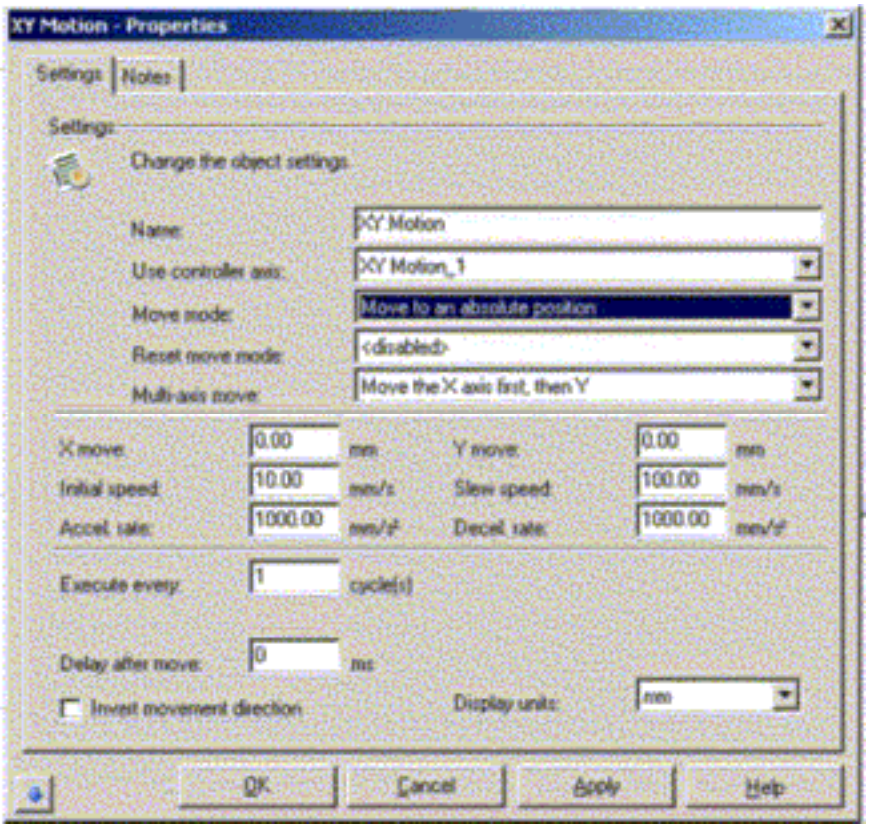
Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

Move continuously in the positive direction: The axis will move towards the positive limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the negative direction: The axis will move towards the negative limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

XY Motion Properties



XY Table Properties Page

Name: This is the axis name as it appears in the **Object Manager**.

Use Controller Axis: Assign this motion object to a specific hardware axis. There must be the correct motion type configured in the **Motion Configuration**.

X move, Y move: The amount to move the axis when performing a normal move.

Initial Speed: The initial speed the axis moves at while accelerating to the slew speed.

Accel. Rate: The acceleration rate of the axis when changing from the initial speed to the slew speed.

Execute every: The object can be configured to skip execution by setting this parameter greater than 1. For example, to execute the object every third job cycle set this value to 3.

Delay after move: The amount of time in ms (milliseconds) to delay after completing the move.

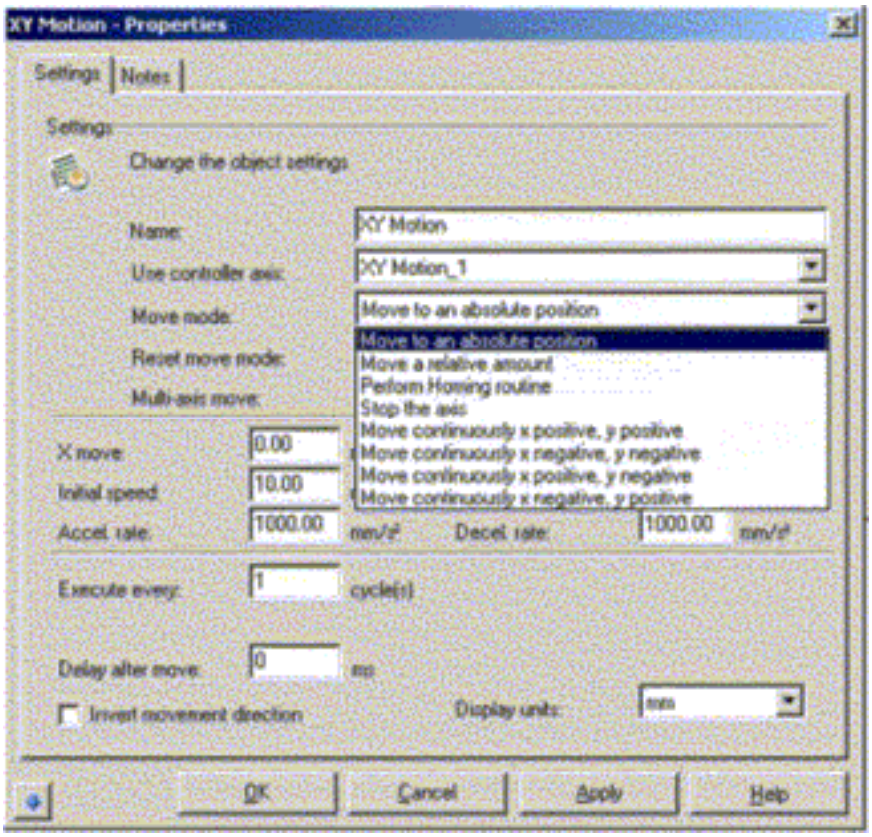
Invert movement direction: When selected all axis movement will be opposite normal movement.

Slew Speed: The maximum speed an axis will attain when executing a move.

Decel. Rate: The deceleration rate of the axis when changing from the slew speed to a stop at the end of the programmed travel.

Display Units: Change the display of all units in the current window. If changed all previously entered values will be converted automatically.

Move Mode:



Select from the following Move Modes:

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

Stop the axis: The axis will be stopped if it was moving as a result of a previous motion object when the object executes.

Move continuously in the x positive, y positive direction: The axis will move towards the positive limit in both x and y axes when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

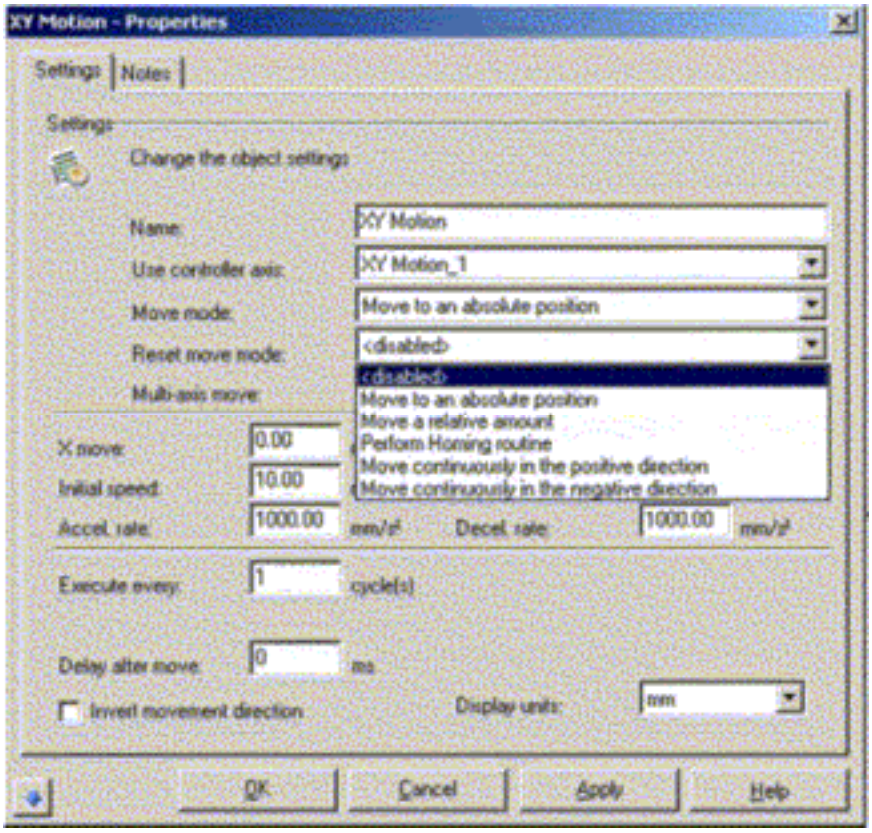
Move continuously in the x negative, y negative direction: The axis will move towards the negative limit in both x and y axes when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the x positive, y negative direction: The axis will move towards the positive limit in the x axis and negative limit in the y axis when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the x negative, y positive direction: The axis will move towards the negative limit in the x axis and positive limit in the y axis when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Reset Move Mode:

A reset condition can be configured so that after the specified number of cycles the axis will perform a rest function.



Select from the following Reset Modes:

<disabled>: Disable the reset functionality.

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

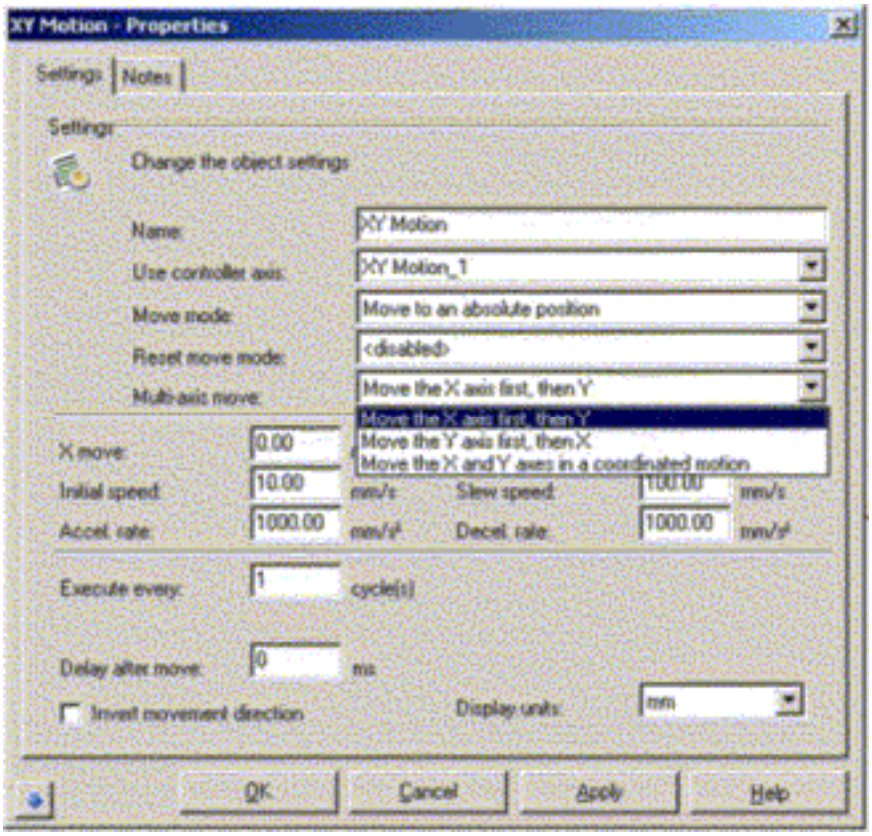
Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

Move continuously in the positive direction: The axis will move towards the positive limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the negative direction: The axis will move towards the negative limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Multi-Axis Move:



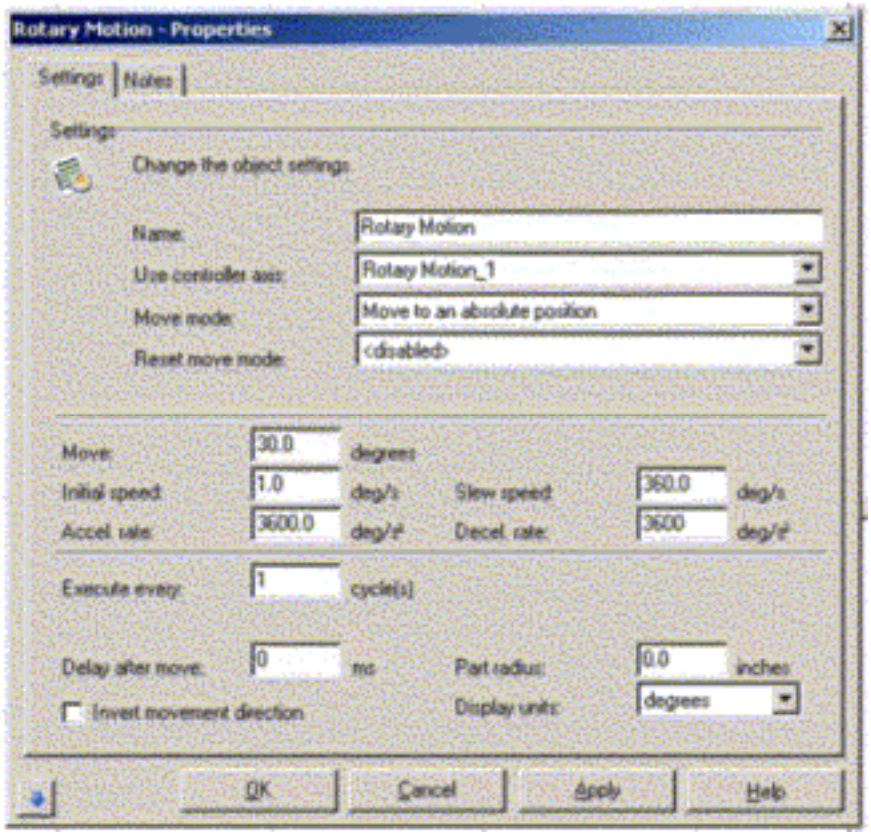
Determines the order in which the XY table will move the axes.

Move the X axis first, then Y: Moves the X axis the specified amount with the Y axis stationary, then stops the X axis and moves the Y axis the specified amount with the X axis stationary.

Move the Y axis first, then X: Moves the Y axis the specified amount with the X axis stationary, then stops the Y axis and moves the X axis the specified amount with the Y axis stationary.

Move the X and Y Axes in a Coordinated Motion: Both X and Y axes will begin their moves at the same time at the specified speeds. The axis with the shorter movement will finish first.

Rotary Motion Properties



Rotary Motion Settings Property Page

Name: This is the axis name as it appears in the **Object Manager**.

Use Controller Axis: Assign this motion object to a specific hardware axis. There must be the correct motion type configured in the **Motion Configuration**.

Move: The amount to move the axis when performing a normal move.

Initial Speed: The initial speed the axis moves at while accelerating to the slew speed.

Accel. Rate: The acceleration rate of the axis when changing from the initial speed to the slew speed.

Execute every: The object can be configured to skip execution by setting this parameter greater than 1. For example, to execute the object every third job cycle set this value to 3.

Delay after move: The amount of time in ms (milliseconds) to delay after completing the move.

Invert movement direction: When selected all axis movement will be opposite normal movement.

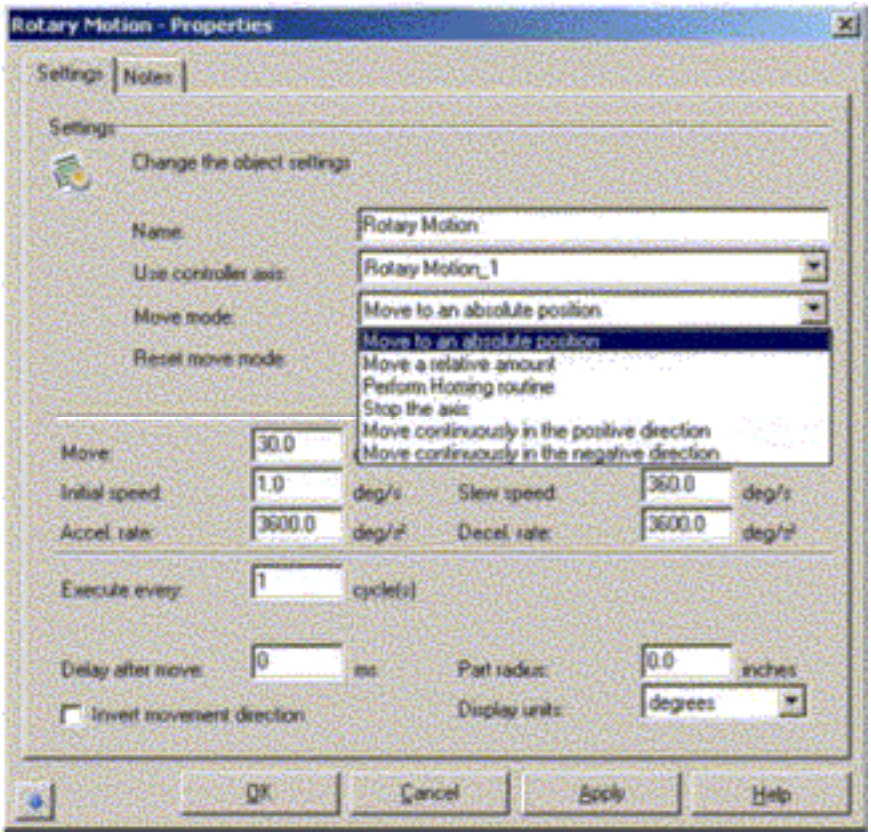
Slew Speed: The maximum speed an axis will attain when executing a move.

Decel. Rate: The deceleration rate of the axis when changing from the slew speed to a stop at the end of the programmed travel.

Part Radius: The radius of the part being marked. This value is useful to scale distances on the surface of the part.

Display Units: Change the display of all units in the current window. If changed all previously entered values will be converted automatically. **Rotary Movements** offer **Degrees** as a unit option.

Move Mode:



Select from the following Move Modes:

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

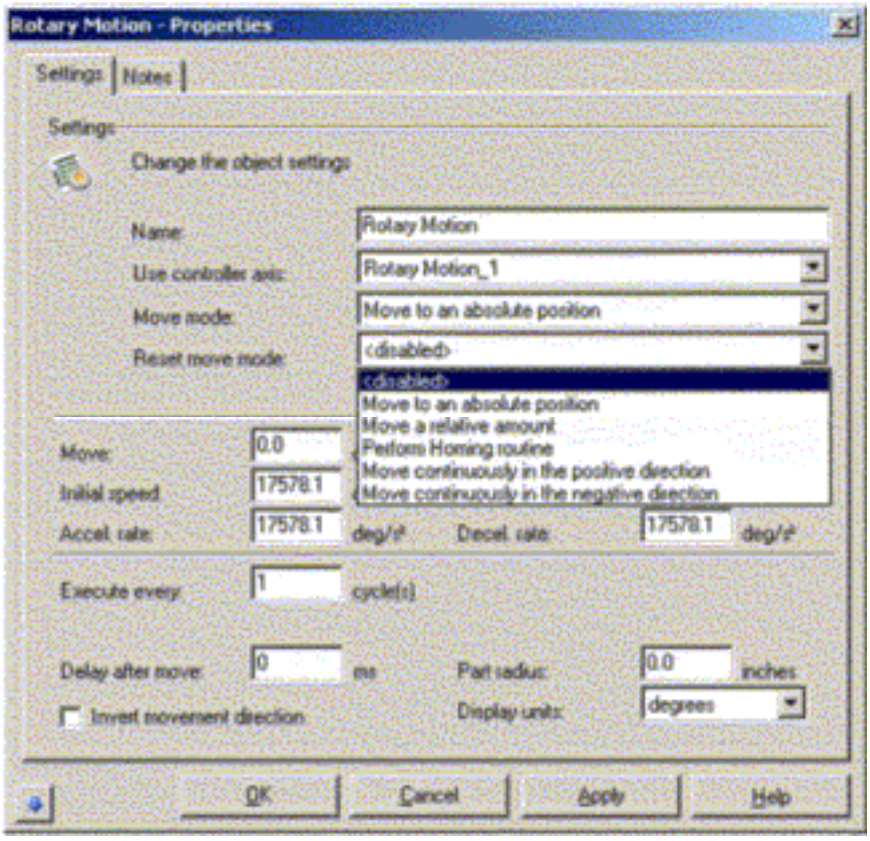
Stop the axis: The axis will be stopped if it was moving as a result of a previous motion object when the object executes.

Move continuously in the positive direction: The axis will move towards the positive limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the negative direction: The axis will move towards the negative limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Reset Move Mode:

A reset condition can be configured so that after the specified number of cycles the axis will perform a reset function.



Select from the following Reset Modes:

<disabled>: Disable the reset functionality.

Move to an absolute position: The axis will move to the actual absolute position indicated in "Move" when the object executes.

Move a relative amount: The axis will move the amount in "Move" from the current motor position when the object executes.

Perform Homing routine: The axis will perform a homing routine based on the home settings of the axis when the object executes.

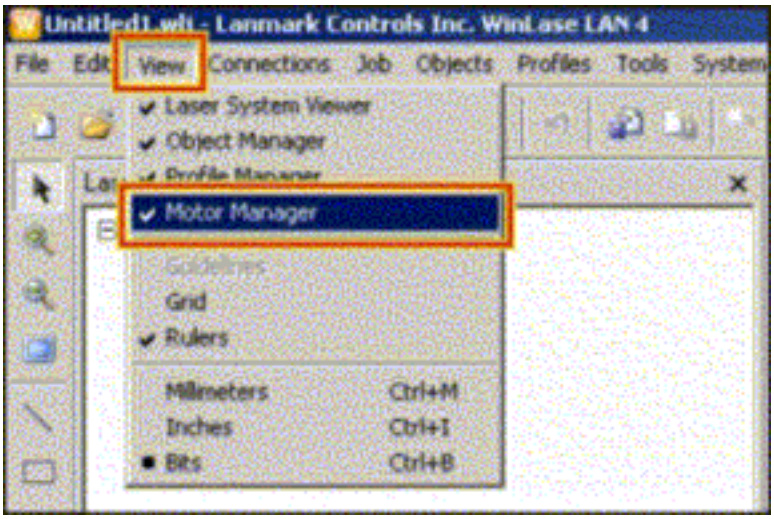
Move continuously in the positive direction: The axis will move towards the positive limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

Move continuously in the negative direction: The axis will move towards the negative limit when the object executes. The axis will continue to move until either hitting a limit switch or if stopped by another motion object.

The Motion Manager Window

Opening the Motion Manager

The **Motion Manager** is a window that allows the user to manually control motion stages by sending commands to the unit. This is usually used for process development and activities like jogging the stage manually.



Opening the Motor Manager

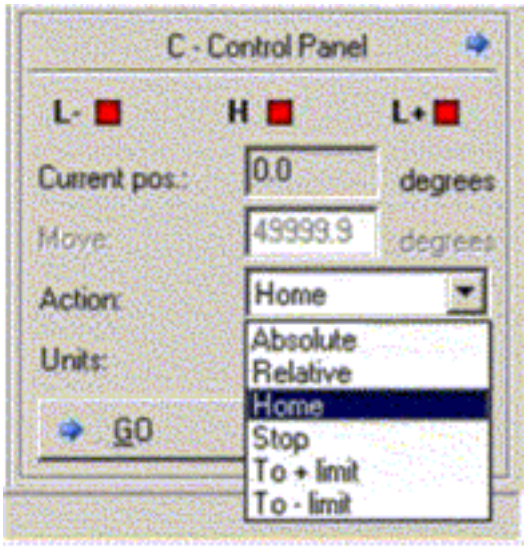
The motor manager appears on the right side of the screen. It is important that a specific laser be selected from the **Laser System Viewer**. To reset the connection to the motor manager it is possible to click elsewhere on the screen, then re-click on the laser in the **Laser System Viewer**.

The actual **Motion Manager** screen is on the right side of the *WinLase* GUI. To control an axis click on the letter address in the **Manager** screen. In the example below axis C is selected. Once selected the bottom section of the screen called the **Control Panel** will activate for that particular axis. It is ***not*** possible to move individual axes simultaneously other than XY tables.

L-, H, L+: These indications show the status of the axis stage limits. A red box indicates logic high. The polarity of the limits can be configured in the axis configuration.

Current Pos: This item expresses the current position in selected units. In the example above the motor is located at 920.2 degrees.

Move: This parameter controls the movement for the next action. If the next action does not require a Move parameter this will be grayed out and ignored.



Action Pull-down Menu

Action: This pulldown menu allows the user to select between a number of possible actions.

- **Absolute:** Performs an Absolute movement relative to the home position.
- **Relative:** Performs a Relative movement relative to the current position.
- **Home:** Performs a Home routine.
- **Stop:** Stops the axis.
- **To + limit:** Moves the axis to the positive limit.
- **To – limit:** Moves the axis to the negative limit.

Units: Allows the user to select appropriate units.

Go: Executes the selected Action.

Stop: Halts movement immediately.

Main View Of Motion Manger And Controller Panel Screen, Axis C Selected

Motion Manager Control Panel Settings

Settings for the control panel actions can be accessed by clicking the little blue “right arrow” in the upper right of the control panel. It is possible to return to the main control panel by clicking the blue “left arrow” in the settings screen.

Accel Rate: The rate of acceleration during movements executed through the control panel.

Decel rate: The rate of deceleration during movements executed through the control panel.

Init speed: The initial speed for control panel movements.

Slew Speed: The maximum speed attained after acceleration has been completed for control panel movements.

Part Radius: The radius of the part for rotary axes. This is a convenience that allows the user to command movements in linear measurement rather than angular units such as degrees.

Settings

Advanced Motion Applications

Choosing between Raster and Vector Marking

A laser marker by definition is a vector engine – the beam can only mark from one place to another using short line segments. This affects how certain marked objects must be configured and affects the final quality and speed of the marked part. It is important to understand this difference when setting up an advanced motion mark such as on a cylinder or other geometrical shape.

Vector graphics are composed of a series of lines – the marker completes these lines at the same power settings, and can perform direction changes at certain points. The letter **C**, for example, would be broken up into many tiny segments on the curved sections. In addition, if there is a closed polygon (like a square, or a letter) it is possible to use *WinLase's* built in functions to fill it very quickly. This is ideal for laser marking since the laser marker simply turns the beam on and off at the beginning/end of each vector, and gives the highest resolution and smoothest lines. A vector format doesn't mark empty areas. It is also substantially faster than a comparable raster mark.

Raster graphics like bitmap and jpg have a series of pixels depending on their resolution. Each pixel is individually marked at a different pulse width setting of the laser. Changing the mark characteristics of each pixel as the beam "rasters" back and forth marks the image. A bitmap with 256 shade grayscale is ideal since it gives the laser a range of "gray" colors to play with. Selecting the ideal resolution is the most important factor to get good results. The problem with raster marking is that it is necessary to mark a large number of pixels for good results. An extremely high resolution is usually necessary – ideal for a 160mm lens is well over 500dpi.

Typically a photo will look good to the naked eye above 250dpi. Although it's hard to estimate dpi based on JPG file size since jpg is a compressed format a good rule of thumb is that bitmap files must be >2 MB to look good, more if the final image will be larger. JPG files that look good are typically 500KB or more. Since each line of pixels (which would be 500 per inch) must be rastered across to get the resolution for ideal performance it can be extremely slow. Marking a monochrome company logo, for example, can take two minutes when it would take 10 seconds or less with a similar vector file. For this reason vector logos are preferred in a non-motion environment.

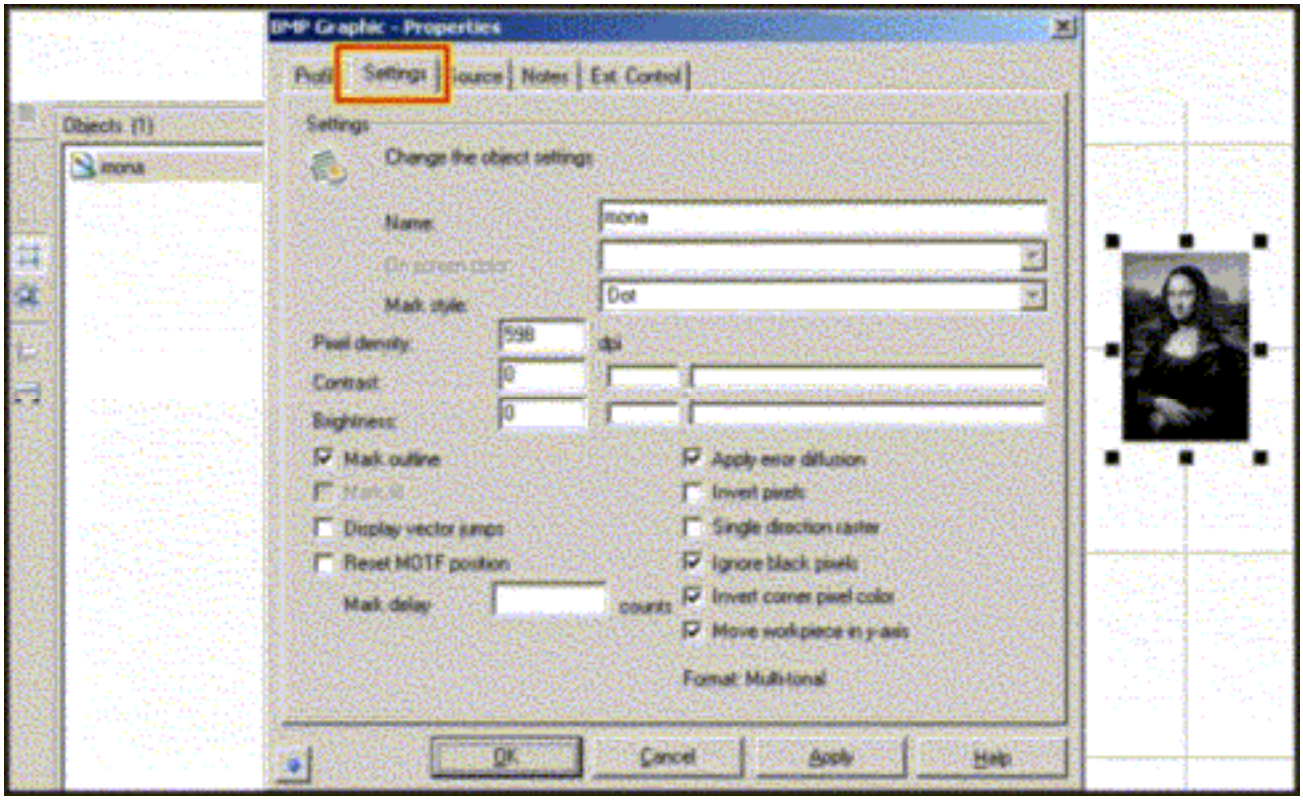
Applying motion to the raster/vector image introduces a new level of complexity. Since the surface presented to the laser beam must always be in focus using motion control it is important to design the mark within the limits of the motion system. In the example of the company logo marked on a cylinder it is very difficult to achieve a good mark with a vector image. In this situation a rastered bitmap is the best solution since the motion control can slowly rotate the part while the laser marks constantly at its focal point. In addition, it is possible to mark a few lines of the bitmap and increment for better speed when the part has a large radius.

Certain types of objects, specifically text, do not work well as raster objects due to the necessary sharpness required to look good to the human eye. In this situation there is a technique called "circumferential marking" used to mark vector text on a cylinder's circumference.

Raster Around a Cylinder

Rastering around a cylinder is a technique used to mark an image around a feature that moves during the mark process to expose areas that are otherwise inaccessible. The classic example of this is wrapping a bitmap image around a cylinder.

To mark a bitmap around a cylinder the bitmap should first be marked successfully on a flat surface, then motion can be employed to wrap the image. To adjust these settings right-click on the bitmap and access its properties screen.



The Bitmap Properties Screen

Name: The name displayed in the **Object Manager**.

Mark Style: Defined by the graphic itself, **Dot** for Bitmap files.

Pixel Density: Adjusts the spacing of pixels within the bitmap image in dots/in.

Contrast: Adjusts the range between the darkest and lightest pixel. A higher contrast setting will increase this range.

Brightness: Adjusts the overall lightness of the image. Increasing this value will cause the laser to stay on longer at each pixel.

Apply Error Diffusion: Applies the standard error diffusion algorithm to a grayscale bitmap, converting it to a monochrome bitmap. The pixels are placed in such a way that the image appears to be a grayscale. Typically not recommended for best quality but can increase overall speed.

Invert Pixels: Reverses the light and dark pixels in the image, using the middle of the range as a reference point.

Single Direction Raster: Forces the lines of pixels in the bitmap to all be marked in the same direction. This improves pixel alignment when marking at high speed.

Ignore Black Pixels: When marking, pixels that have a value of black (non-marking) will be jumped over by the laser beam, reducing overall mark time.

Invert Corner Pixel Color: When bitmaps are rotated black pixels may be added to the bitmap corners. To add white pixels instead, select this checkbox.

Move Workpiece in the Y-Axis: Select this option to lock the Y mirrors in place and only scan the X mirrors. The part to be marked is then translated under the marking head in the y-axis. This translation must be done using a motion axis. The speed is of axis movement is an important part of this equation and it is necessary to optimize it for proper bitmap appearance.

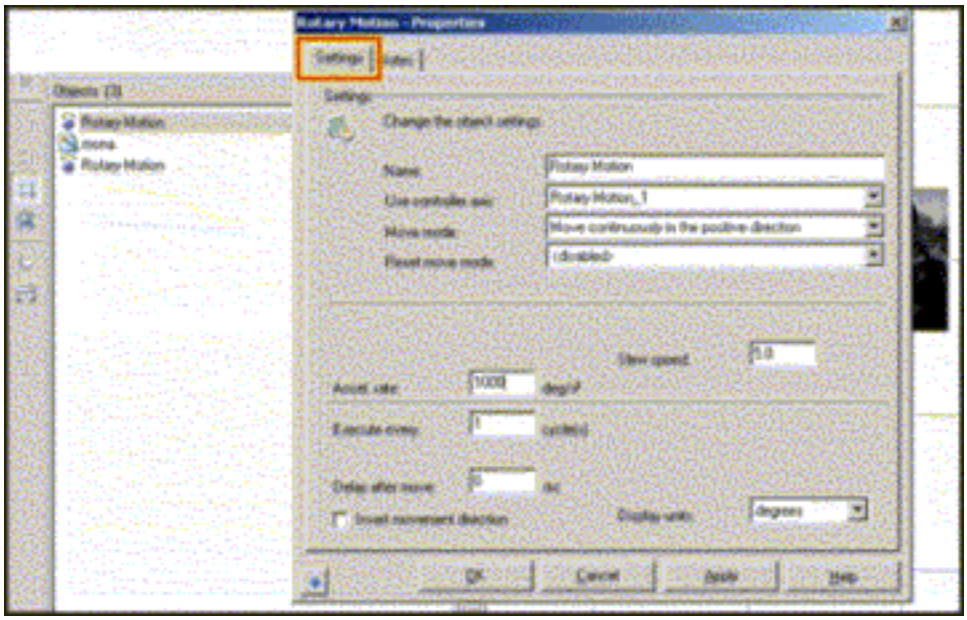
Bitmap Marking Methods

There are two control methods used for Bitmap marking around a cylinder: **Manual** and **Automatic**. Both methods require **Move Workpiece in Y-Axis** to be selected. In **Manual** mode the stage is rotated and the mark is placed continuously. In **Automatic** the stage is controlled by the object and a number of bitmap lines are marked, then the part is indexed and repeated until the bitmap is completely marked.

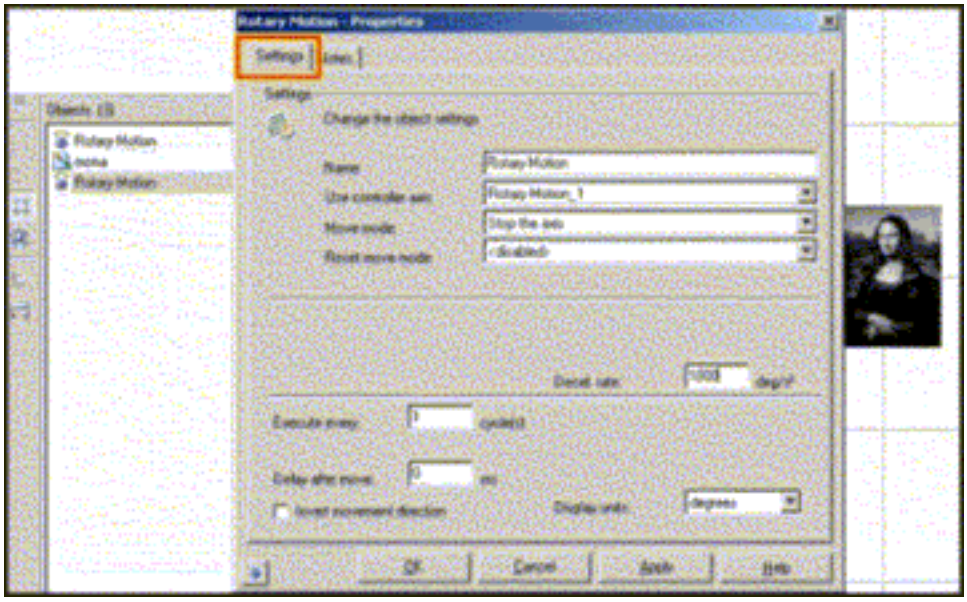
Manual

In this method the user controls the rotation of the stage as well as the bitmap. Once the bitmap has been configured and set to move in the Y axis the user adds a **Start Rotation** object before the bitmap using a move continuously command and a **Stop Rotation** object after using a stop the axis command.

In the following example, note the three objects in the **Object Manager**.



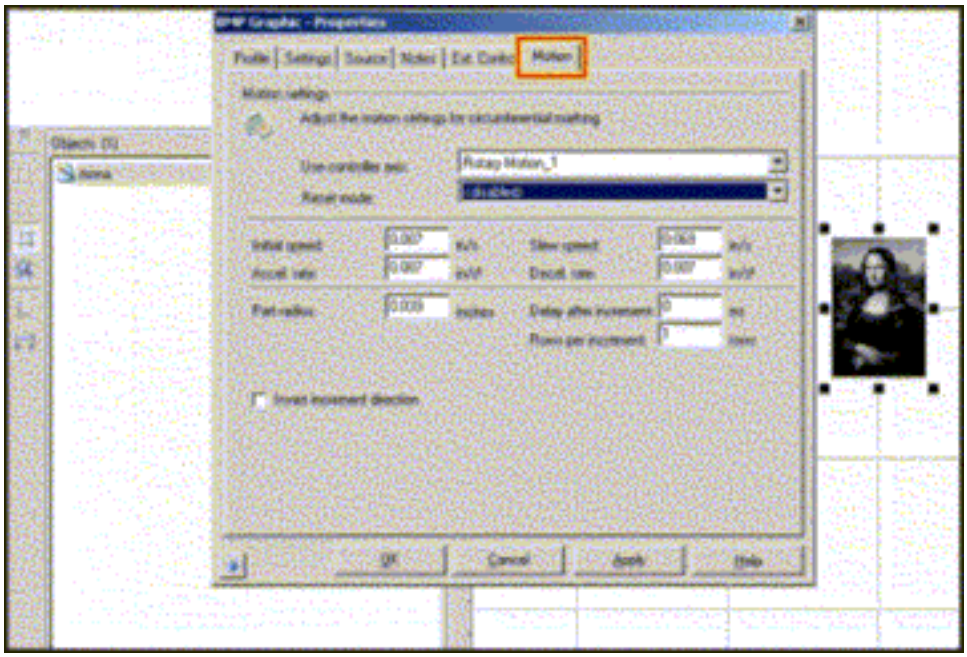
Rotary Motion Object Configured To Start Rotation For Bitmap Marking.



Rotary Motion Object Configured To Stop Motion After The Bitmap Is Marked

Automatic

It is also possible to use the **Motion** tab in the Bitmap settings to control the motion in a "step and repeat" mode rather than a continuous mode. In this case the bitmap controls the motion and no extra motion objects are required. To enable, check **Move Workpiece in Y-Axis** in the bitmap settings and a new **Motion** tab will appear.



The Bitmap Motion Tab

Configure this for the correct bitmap marking speed.

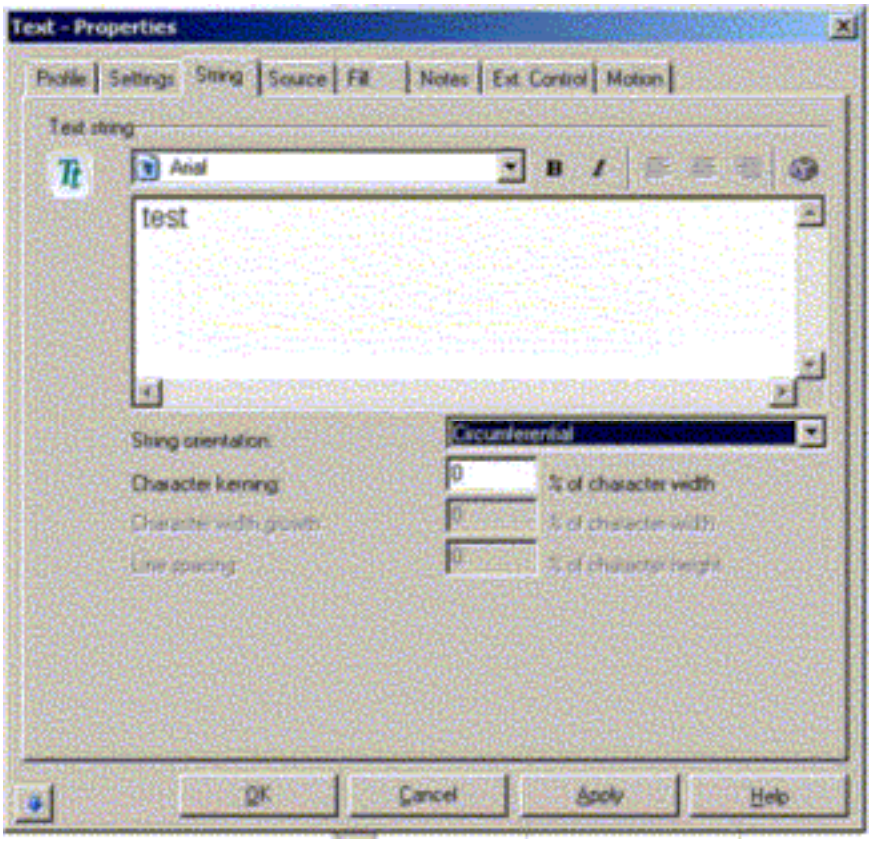
Part Radius: This specifies the radius of the part and is used to properly align the marked rows using stage motion.

Rows Per Increment: This specifies how many rows of the bitmap will be marked for each movement. Larger parts can handle more rows without the rows going out of focus. Make this value as large as possible such that good results are obtained.

Circumferential Marking

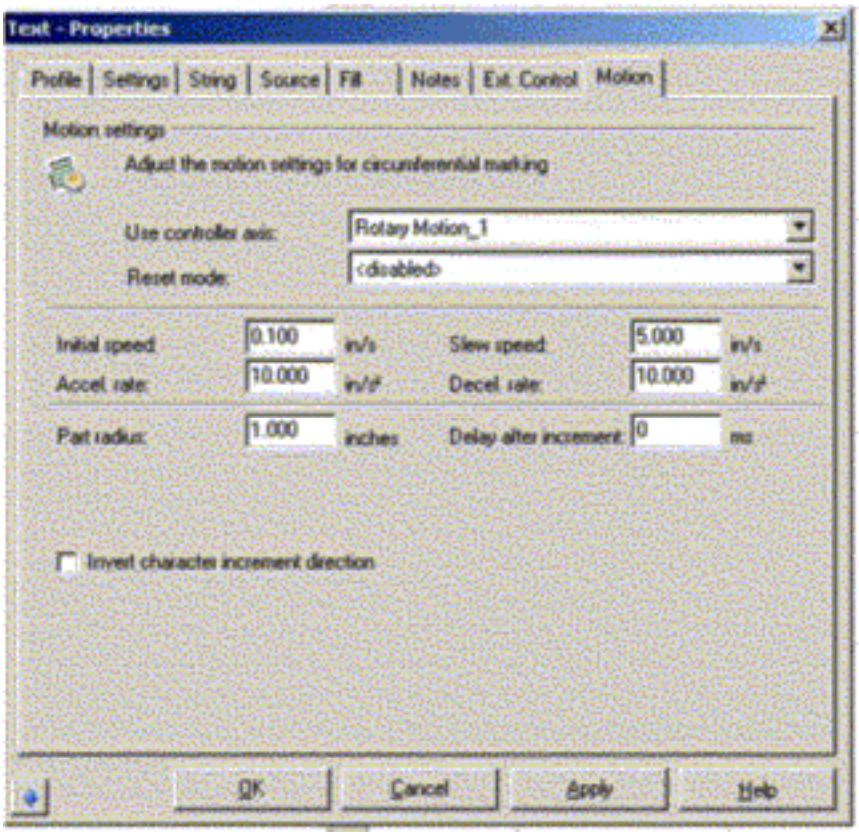
Marking Vector Text around a Circumference. The Circumferential mark feature allows the user to mark vector text around the circumference of a round part. In this case the rotary axis will index after each letter to maintain the work area in laser focus. The text to be marked will "compress" on the laser marking field screen to show the laser marker's view of the text to be marked. This is a fast and efficient way to mark high quality text around on a round or cylindrical object.

To use circumferential text with a compatible axis installed create a standard text object and open the **String** tab. Select **Circumferential** in the **String Orientation** pulldown. A **Motion** tab will appear



Selecting Circumferential Text Mode. Note the Motion tab.

The **Motion** tab has typical motion related commands.



The Motion tab

Initial Speed: Initial velocity of the motor.

Accel. Rate: The acceleration rate of the motor when changing velocity.

Part Radius: The radius of the part to be marked.

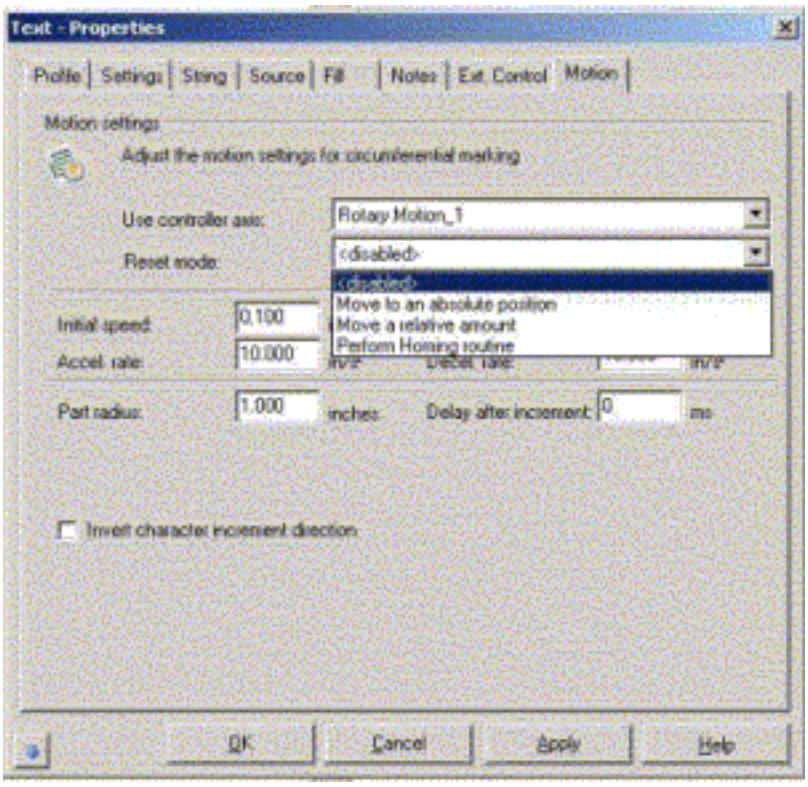
Invert character direction: Reverses the rotation direction of the motor when indexing between individual characters in text string.

Slew Speed: The maximum velocity the motor will reach during a move.

Decel. Rate: The deceleration rate when the motor is coming to a stop

Delay after increment: Change the delay between when the motor has stepped to its final position and the next character is marked. This delay allows settling time for the motor to come to a complete stop.

Reset Mode:



Reset Mode

Move to an absolute position: The motor will move to an absolute location after the mark object.

Move a relative amount: The motor will move a certain amount relative to the current location after the mark object.

Perform Homing Routine: The motor will return to home after the mark object.

Banding

Marking increment bands on tubes. A common application for rotary motion in the medical field is marking index bands on tubes. These bands are placed a set distance apart and used for reference. Often numbers will be marked indicating the exact distance from a reference point to a specific band.

Bands are typically marked on the top of a tube that is spinning at high velocity. A tiny but simple mark such as a line or square will be marked at the apex of the tube at extremely slow speeds. As the tube spins at high speed a small band is marked without etching the surface of the material. Surface damage can provide places for bacteria and debris to accumulate, so a clean mark is usually required.

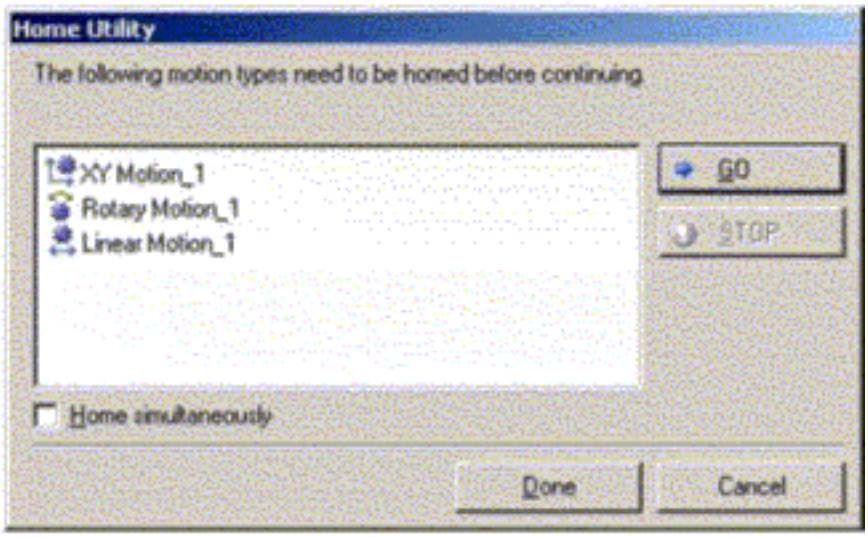
Homing

Homing is the process of referencing the motor position as seen by the motion controller to the physical location of the stage. This is done by moving the stages individually until each finds a known reference sensor or index mark and sets its "zero" position.

Homing is typically done in two common instances. First, any time a machine is turned on the home must be acquired. Second, any time a machine loses its position from a power loss, crash, or other issue will require a re-home. Motors in "open loop" mode without encoders connected are candidates for more frequent homing since any movement of the stage by forces outside the motor itself are not registered by the controller. Motors equipped with encoders ("closed loop") monitor their position and requires less frequent homing. These motors report all movements to the controller including those that occur by crashes, hard stops, or limit overruns.

The Miyachi Unitek Fiber Laser Motion Power Supply is equipped with auxiliary controls to power motors equipped with encoders so that home positions are not lost in the case of crashes or emergency stop events.

Homing can occur in three ways. The first time a job is executed with motion objects after a power up sequence a home prompt window will appear.



Home Prompt Window

The axes can be homed individually using the **Motor Manager**. Certain motion objects can perform a homing routine or execute a home routine as part of a reset move.

NOTE: For more information on **Homing**, see *Section IV* earlier in this chapter.

Troubleshooting

Clearing Errors. To attempt to clear a motion error the axis should be selected in the Motion Manager. Click on the laser name in the Laser System Viewer and open the Motion Manager. Click on the letter name of the axis that is causing problems. Send a command using the Motion Manager control panel and see if the runtime errors return.

When troubleshooting motion errors, verify:

- Are there any errors on the **Error Display** located in the **Motion Manager**?
- Is the motor communicating?
- Are all cables connected?
- Is the fiber marker turned on, booted, and connected to *WinLase* in the **Laser System Manager**?
- Is the fiber marker system in a fault state like **Emergency Stop** that would disable motors?
- Is the external power supply energized?
- Are there any *WinLase* faults?
- Is the appropriate COM port selected in the *WinLase* System Settings screen and set to 9600 baud? (LMF Laser Markers use COM3 by default)

Categories of Errors and Potential Resolution

- ***WinLase cannot communicate with the motion system*** (tried and failed)

  Most commonly caused by a disconnected or defective cable or power supply. Emergency Stop can result in this error as well if the auxiliary power supply is not connected.

- ***Motion System Not Found*** (*WinLase* doesn't even try and communicate)

  Most commonly caused by improper *WinLase* configuration. Check baud/COM settings.

- ***Limits have been* reached *unexpectedly***

  The high or low limits have been activated and the process has been aborted. Ensure that the motion program does not cause the stages to reach their limits, or troubleshoot defective limit switches or wiring.

- ***Motion has been aborted by user***

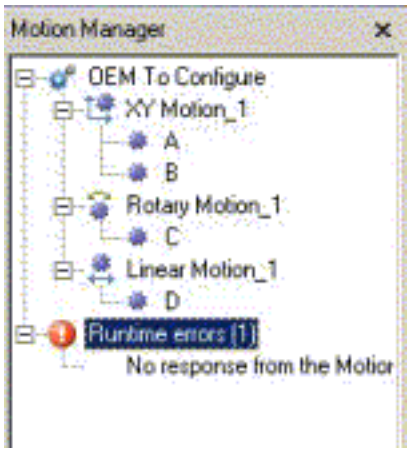
- ***Motion system has returned an unknown response***

  Bad response from the motion system. Try resetting the system and try the operation again. If problems continue to occur contact Amada Miyachi America for assistance.

- ***Stall Detected***

  An axis has stalled. This is nearly always caused by improper motion parameters or a stage which has mechanically locked up. Verify the acceleration, deceleration, initial speed, and slew speed of each motion axis and homing configuration. Any of these parameters can cause a stall if set outside the motors' ability to move the stage appropriately. Verify encoder settings including appropriate deadband and stall detection parameters. Ensure that the stage has not crashed, gotten stuck, or otherwise become mechanically bound. Reset the motion system, lower speeds and/or accelerations, and try again. If problems continue to occur contact Amada Miyachi America for assistance.

The Motion Manager Error Display. The **Motion Manager** in WinLase contains the primary error reporting mechanism for the intelligent motion system. The **Runtime Error** segment keeps track of any errors that occur during execution like motor stalls, limit hits, loss of communication, etc. Errors will accumulate in the window until a maximum of 15 are stored.

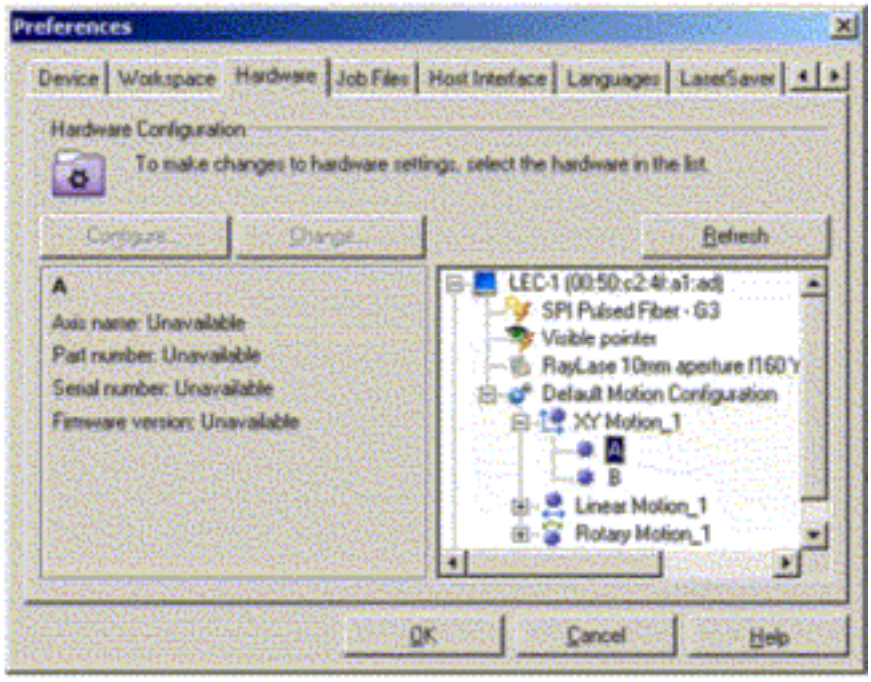


Motion Runtime Errors Displayed in the Motion Manager

Verifying Motor Communication. The **System > Preferences** screen provides a wealth of information about the current system. It is possible to look at each individual motion axis and verify that the individual motor is communicating, verifying the communication cables, power supply, and motors are functioning all at once.

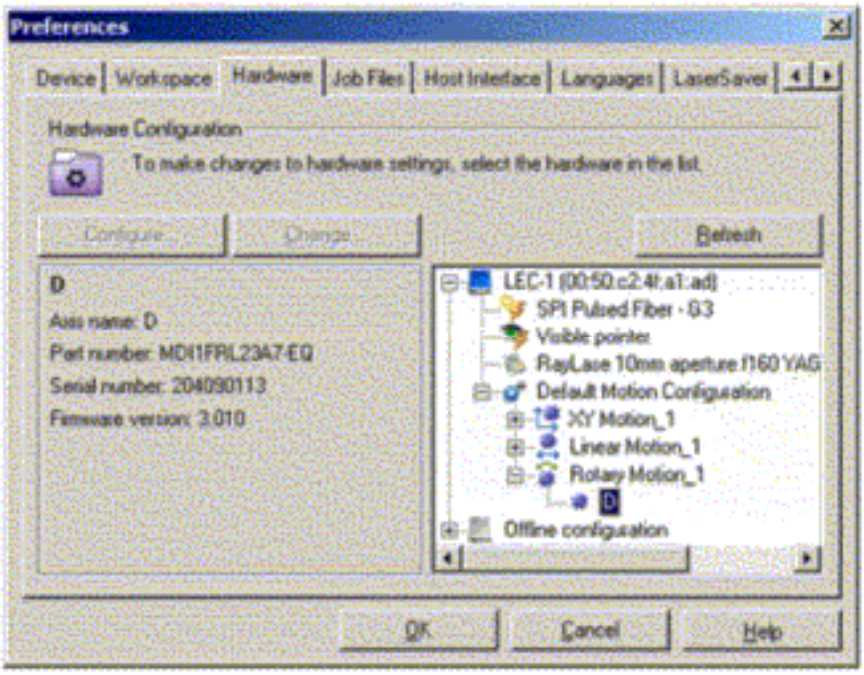
Isolating the problem axis is the first step towards resolving the problem. You can also verify that the motors have been correctly programmed by looking to see if they are correctly reporting their serial number.

In the *WinLase* main screen click **System > Preferences** and select the **Hardware** tab. From the settings tree structure, navigate until the motors are visible and show their axis names (A, B, C, or D).



Axis A Is Unprogrammed

Select each axis and see if the **Axis Name**, **Part number**, **Serial number**, and **Firmware version** are correctly reported. **Unavailable**, gibberish, or **????????** mean the programming has failed or a problem exists with the cables, power supply, or motor itself. Please check your power cables and communication cables and try again. Gibberish typically means that that more than one motor was plugged in when programming was attempted. Connect only one motor to a primary cable and try programming again.



Axis D Is Programmed and Communicating