QuickMotion QuickStart Guide

Doc. No. 951-530018-006

25 South Street
Hopkinton, MA 01748

Phone: 508.425.9595
Fax: 508.435.2373

Tuesday, December 11, 2007

Control Technology Corp.

Table of Contents

QuickMotion QuickStart Guide

Control Technology Corp.
25 South Street
Hopkinton, MA 01748
Phone: 508.435.9595 • Fax 508.435.2373

Document No. 951-530018-006

⚠ WARNING: Use of CTC Controllers and software is to be done only by experienced and qualified personnel who are responsible for the application and use of control equipment like the CTC controllers. These individuals must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and/or standards. The information in this document is given as a general guide and all examples are for illustrative purposes only and are not intended for use in the actual application of CTC product. CTC products are not designed, sold, or marketed for use in any particular application or installation; this responsibility resides solely with the user. CTC does not assume any responsibility or liability, intellectual or otherwise for the use of CTC products.

See www.ctc-control.com for the availability of firmware and software updates or contact CTC Technical Support.

Conventions used in this guide:

When you need to type some text it will appear like this:

Type: **`StrtInput`**

where the characters you need to type are shown in a larger Bold Courier font.

1 Chapter 1: Overview

This document is a step-by-step guide to building a simple motion project using the QuickBuilder Automation Suite. It covers the basic setup, programming, and startup steps necessary to get the application up and running smoothly. The resulting project will cause a servo motor to execute a simple back and forth motion and will use the tuning wizard to optimize this motion.

It is assumed that the reader is already familiar with QuickBuilder and its use with the 5300 automation controller. If that it is not the case, please refer to *Document No. 951-530030: QuickBuilder QuickStart Guide* and *Document No. 530020: QuickBuilder Reference Guide* before continuing.

The following CTC hardware is used for this example project:

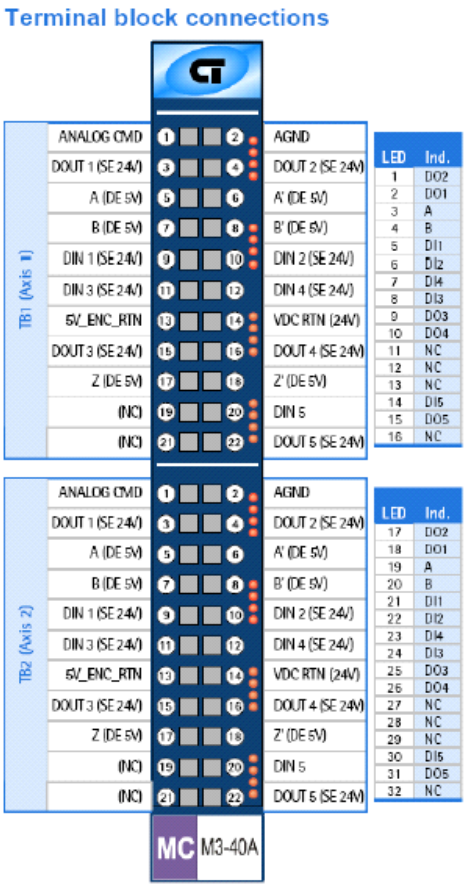
- Model 5300 automation CPU, rack and power supply.
- M3-40A dual axis servo module
- M3-18A digital switch / LED module
- QuickBuilder Automation Suite

☑ **Note:** It is recommended that the reader have a copy of *Document No.530017: QuickMotion Reference Guide* for this exercise.

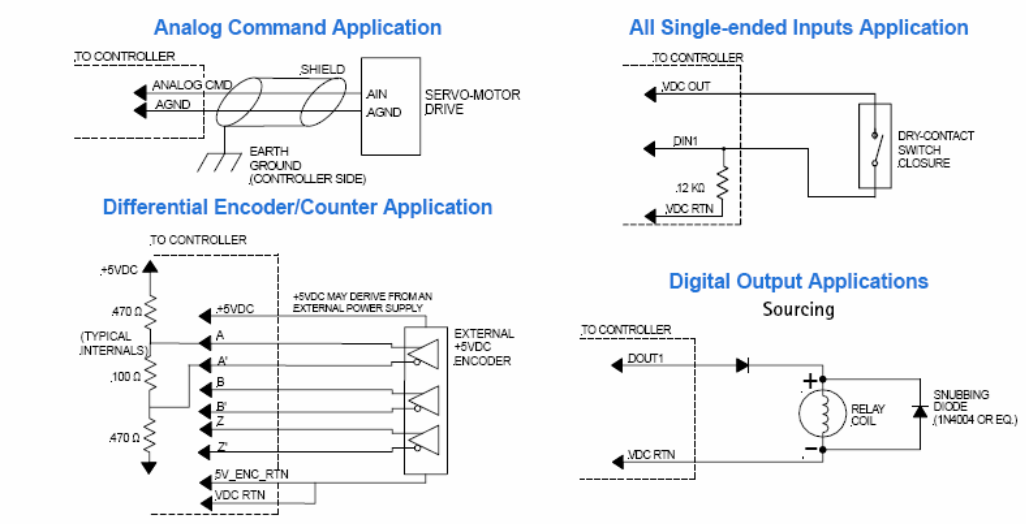
1.1 M3-40A Servo Control Hardware Features Overview

- Two axes of servo control per module
- Up to 64 axes per 5300 system
- Position loop update times of 500us / 2 axes
- Encoder Feedback up to 20 MHz
- 64-bit floating point loop control
- 16-bit analog command
- 5 user assignable inputs / axis
- 5 user assignable outputs / axis
- High speed registration capture
- High speed PLS outputs

1.1.1 M3-40A Servo Control Hardware Connections



1.1.2 M3-40A - Typical Hardware Connections



See also...

- APPENDIX B for more wiring information and good wiring practices

1.2 Adding Motion to the Blue Fusion 5300

There are three elements that need to be addressed to add motion to a control application:

1. **The Axis Module:** The physical motion module in the Model 5300 rack
2. **The Axis Object:** The QuickBuilder resource representing an axis on a given physical Axis Module. This is a software description of the Axis Module.
3. **The MSB:** The Motion Sequence Block containing one or more motion statements that execute on the Axis Module's CPU under the supervision of QuickStep on the main 5300 CPU.

2 Chapter 2: Building a Motion Program

2.1 Motion Sequence Blocks (MSB's)

To create motion on an axis, one or more motion commands are placed in an object called a Motion Sequence Block (MSB). The programming language used to enter commands into an MSB is an extension of QuickStep4 called QuickMotion. After an MSB is created, that MSB can be used by any of the axes in the 5300 system at anytime. A simple example would be a homing MSB – write it once, and then use it on as many axes as desired.

The initial MSB must be started from QuickStep4 within QuickBuilder for any given axis.

MSB's can also be called from other MSB's. So once an MSB is started on an axis, it can in turn start additional MSBs directly. Although a given MSB can be running on many axes at the same time, only one instance of a particular MSB can be running on a given axis. If an MSB is running on an axis, additional attempts to start the same MSB will be ignored.

2.2 Quickstep Motion Commands

There are only three motion related commands at the QuickStep level: **start**, **stop**, and **sync**. They are all used to start MSBs running on an axis, or to stop all MSBs running on an axis.

start will start an MSB for the stated axis:

```
start Axis1 MSB1;          //will start an MSB called MSB1 for Axis1
```

stop will stop execution of all MSBs and halt motion on the named axis:

```
stop Axis1;                //will stop execution of all MSB's for Axis1
```

⚠ **Note: Use this stop command with Caution.** It will stop will stop all MSB's for that axis, stop all motion for that axis, disable the tuning loop and turn off the drive enable output if one is assigned. There is a QuickMotion stop command that can be used from within an MSB to allow you to make controlled stops.

sync will start multiple MSB's at the exact same time.

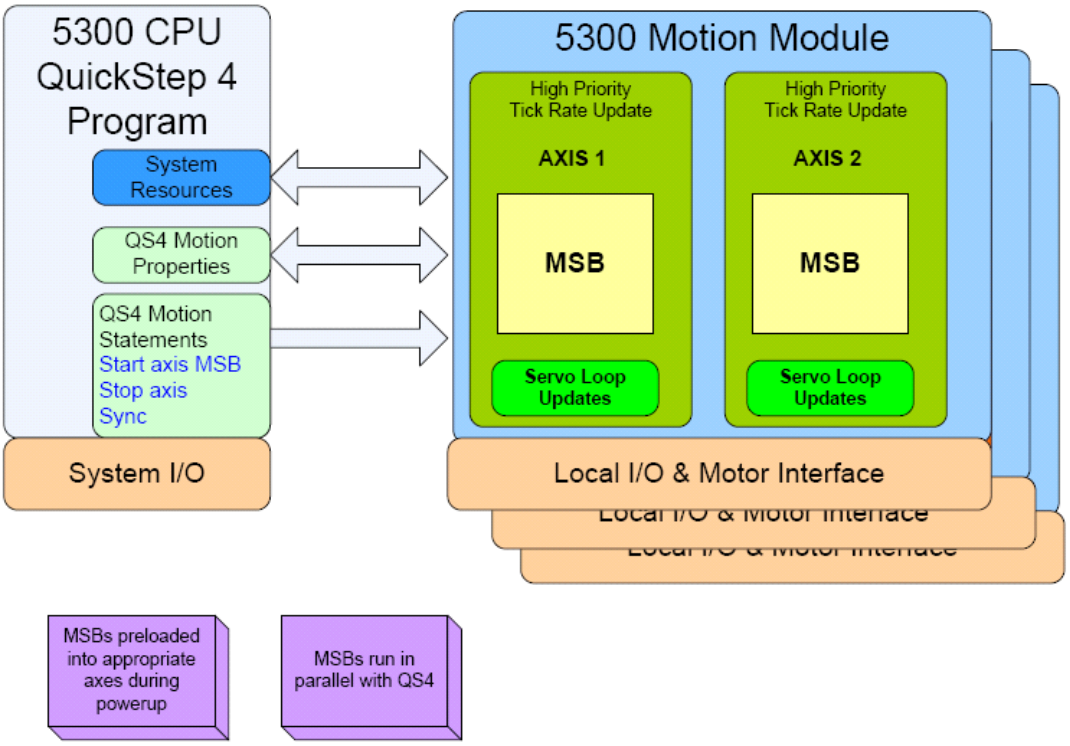
```
sync {                     //will start the Conv MSB for axis1 and
                           axis 2 simultaneously

start axis1  Conv;

start axis2  Conv;

}
```

⚠ **Note:** The MSBs that are started by the QuickStep program can be as simple as a single command, or they can be very sophisticated motion routines containing hundreds of commands. It all depends on the desired results

and the programmer's preference. Remember one of the big benefits of the MSB architecture is that the MSBs run independently on the Axis Module's CPU. This allows the MSB statements to run very quickly without bogging down the main system processor. And since every axis module has its own processor each Axis Module can be running its MSBs in parallel without slowing down the processing of another axis or the main processor. Keep this in mind when deciding how to optimize your application.

2.3 Model 5300 Motion Architecture

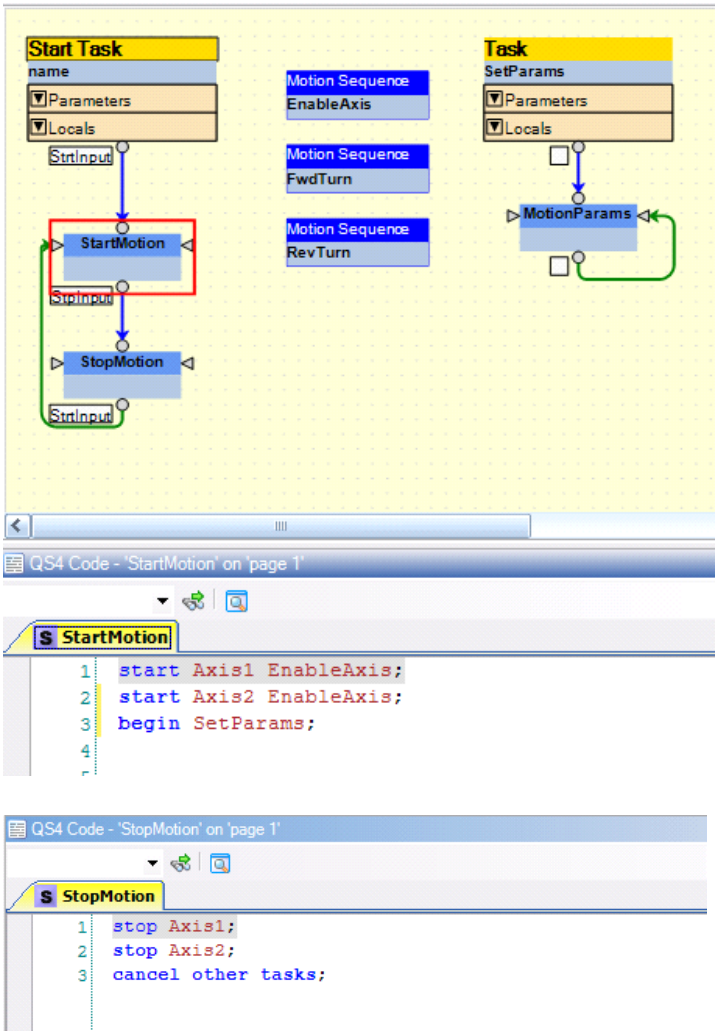
Here is a pictorial view of the overall architecture of how the QuickStep4 Program in the CPU interacts with QuickMotion MSB's in the M3-40A cards.



2.4 QuickBuilder and MSB Usage

We will now take a look at a sample QuickBuilder program incorporating both QuickStep Code and QuickMotion MSB's. This program simply waits for an input, then moves both Axis1 and 2 forward and back with a brief delay between and then repeats this motion until another input is turned on to stop the motion.

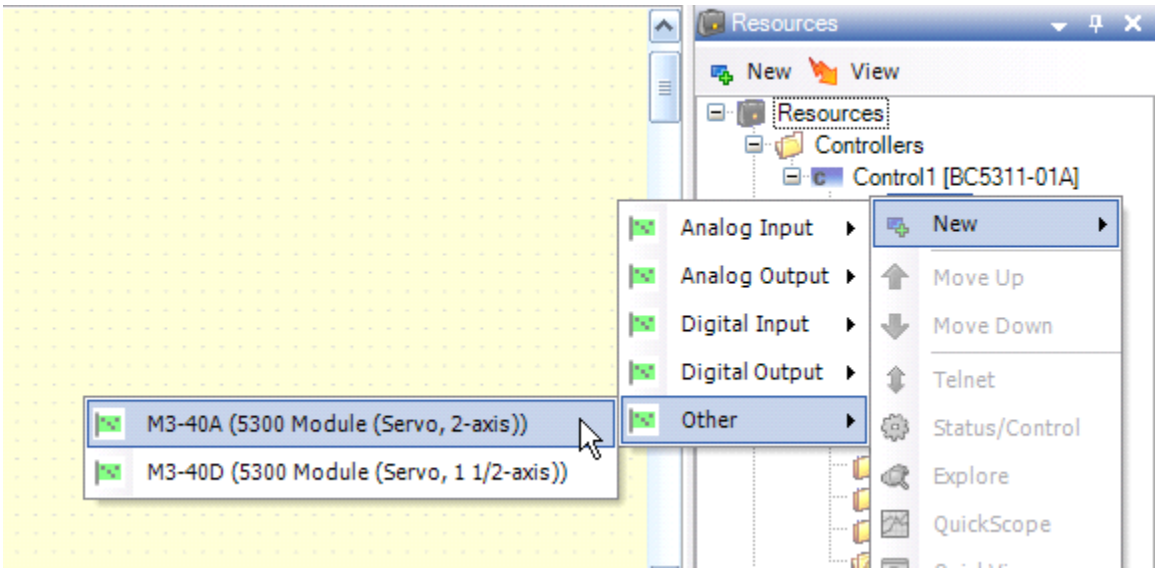
The **SetParams** task on the right simply allows changes to the velocity and distance of the moves being made on the fly from within QS4, an operator interface, or a watch window.



2.5 A Simple Tuning Program

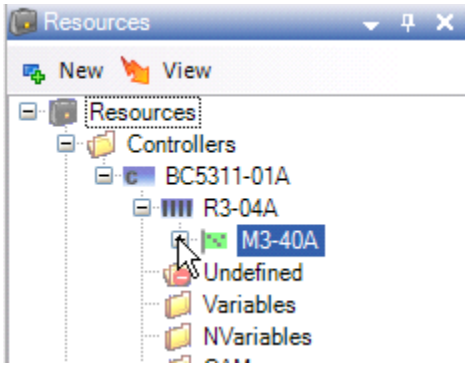
Let's write this program: (remember it is assumed you are familiar with QuickBuilder and Quickstep4. Refer to the QuickBuilder QuickStart guide if you are not already familiar with QuickBuilder)

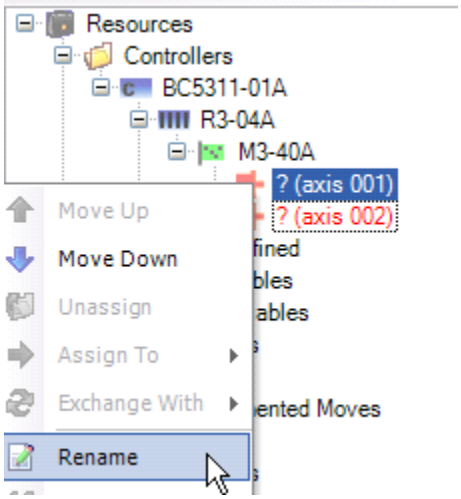
First: Add and name your controller. Set up the IP address for the controller. Add the type rack you have, your M3-40A and all the other cards you have in your system.



NOW:

1. Click on plus next to the M3-40A card.
2. Right click on axis 001.
3. Select Rename.
4. Type: **Axis1**.
5. Repeat this for axis 002, naming it **Axis2**.





NOW: We need to set the critical properties for each of your servos. Remember to set them for both Axis1 and Axis 2 if you are using both of them, and to use values for your system. *The values on the right are shown just for reference* .

Required — When setting up an axis the following properties must be set up in order for the Servo Control module to properly interface with the connected motor and drive (defaults are listed in [ ])

- **cmode:** Determines what type of analog command signal the controller sends out—Set to [Torque] or Velocity. This option is hard coded to Torque in Release 1 of QuickMotion and is not displayed in the menu at left.
- **tmax / vmax:** Depending on the drive type, set the maximum torque in Nm or velocity in RPM that will be realized by a 10V command from the controller. [1Nm /1000RPM]
- **ppr:** The number of post-quadrature feedback counts per revolution [4000]

Recommended — Once the required properties have been entered the axis can be tuned. However it is recommended that the following properties also be checked and adjusted as necessary.

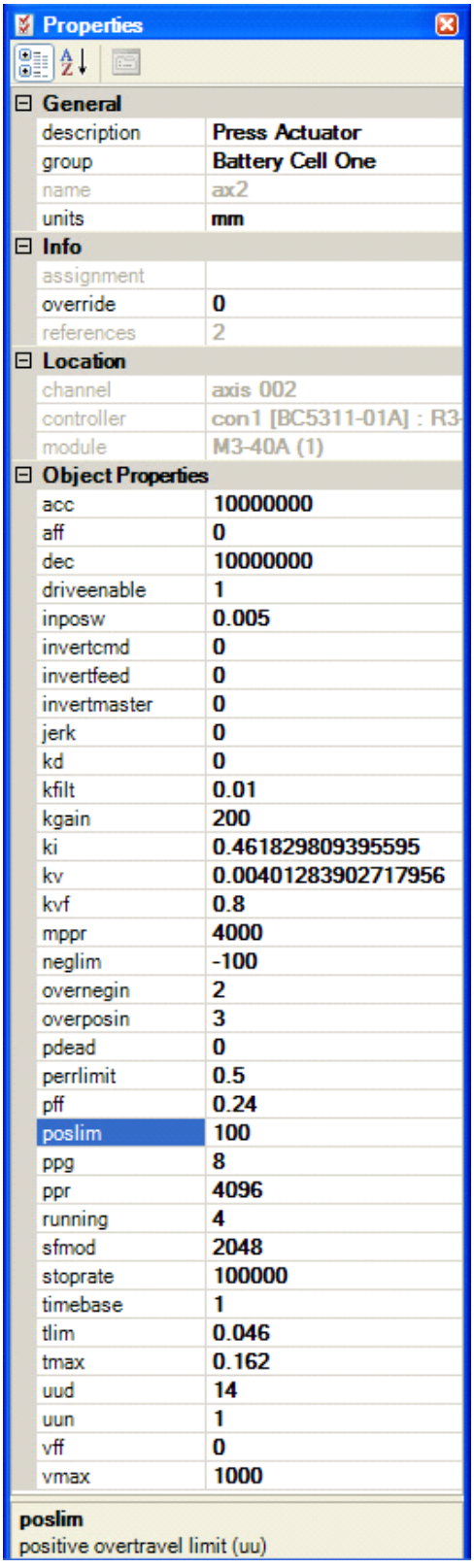
- **acc / dec:** Check that the acceleration / deceleration rates are appropriate [10000000/10000000]
- **driveenable:** Set this to the output number that will be used to enable the servo drive. (Highly recommended that this be used) [0=not used]
- **inposw:** The in-position window scaled in user units. This is used to determine when the drive has reached the commanded position. [0.01]
- **overnegin / overposin:** (Hardware over-travel limits) Set these to the input number to be used to signal positive and negative over-travel. [0=not used]

☑ **Note:** overnegin and overposin are not required in some applications but it is highly recommended for all linear applications.

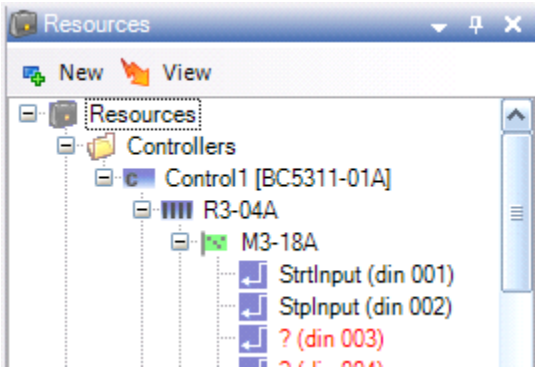
- **neglim / poslim:** (Software over-travel limits)

Set these to the input number to be used to signal positive and negative overtravel. [-1E+50 /1E+50]

- **peerlimit:** This is the maximum allowed following error in user units before a fault is generated. [0=disable checking]
- **uun/uud:** User-units numerator and denominator. This fraction is used to convert revolutions to user units. [1/1]

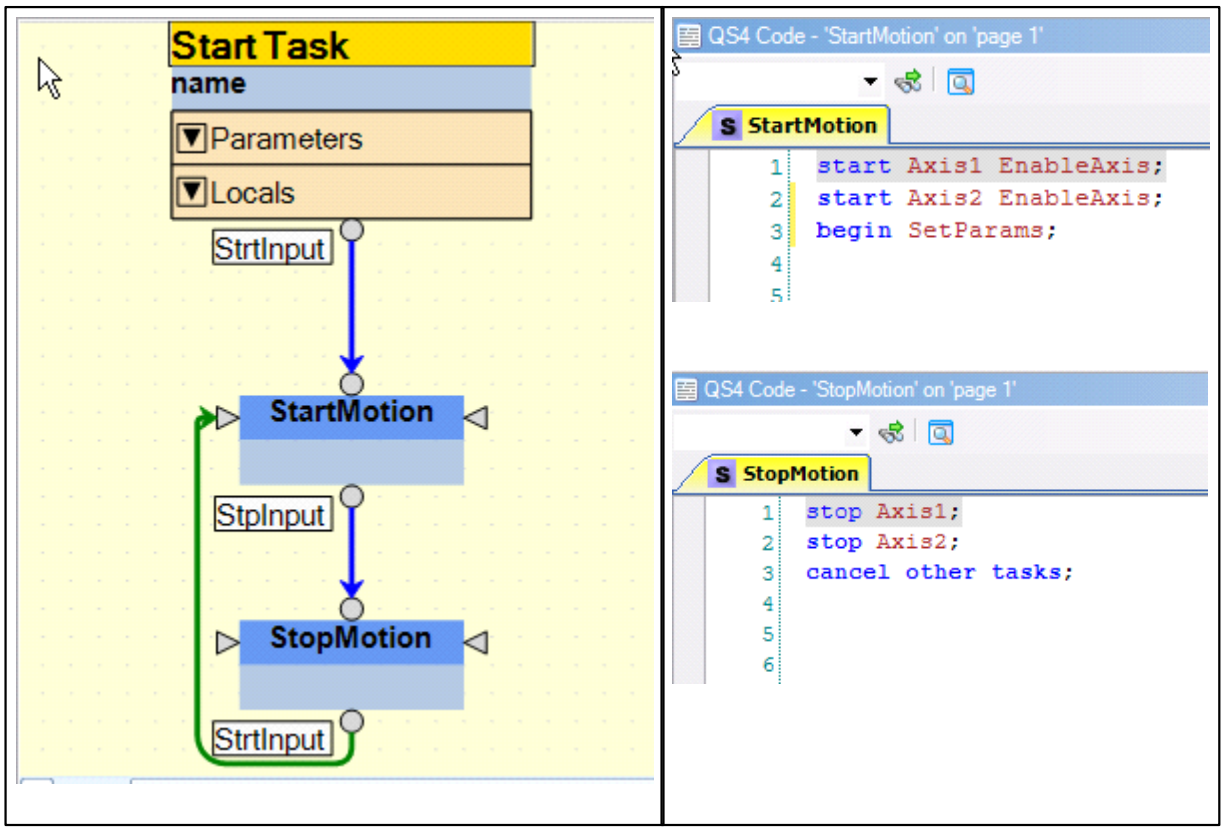


NOW: Define the following inputs on one of your input cards.



NOW:

1. Create the following flow chart using QuickBuilder as shown below left.
2. Add the QS4 Code for the **StartMotion** and **StopMotion** steps as shown below right.



☑ **Notes:**

1. Both Axis1 and 2 will be using the same exact MSBs in this example. Using MSBs allows you to write

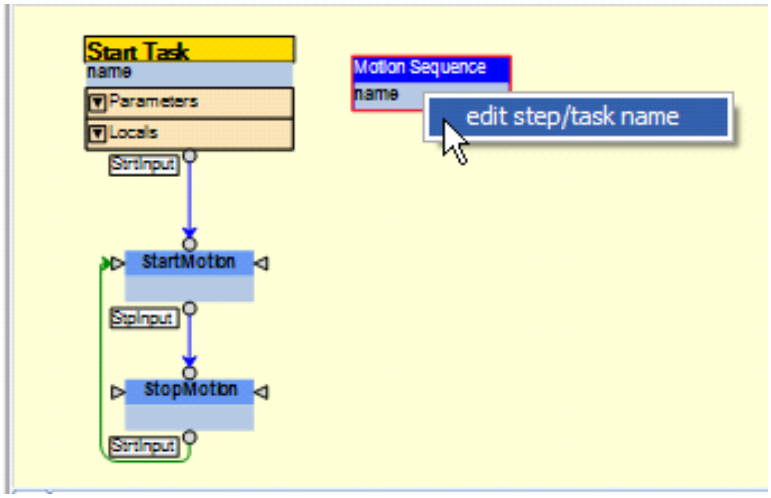
code only once when multiple axes are doing the same thing. This is particularly useful for homing routines.

2. If you are only using one axis, omit any references to Axis2. (i.e., **start Axis2 EnableAxis;** and **stop Axis2;**)

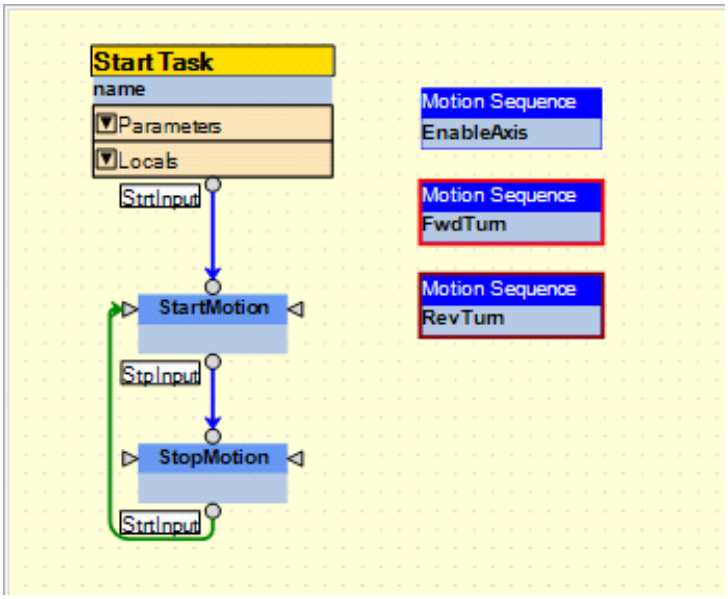
NOW: Click on the Motion Sequence Block Icon to add an MSB.

NOW:

1. Move the Motion Sequence Block as shown below.
2. Right click on the MSB.
3. Select **edit step/task name**.
4. Type: **EnableAxis** to name the MSB.

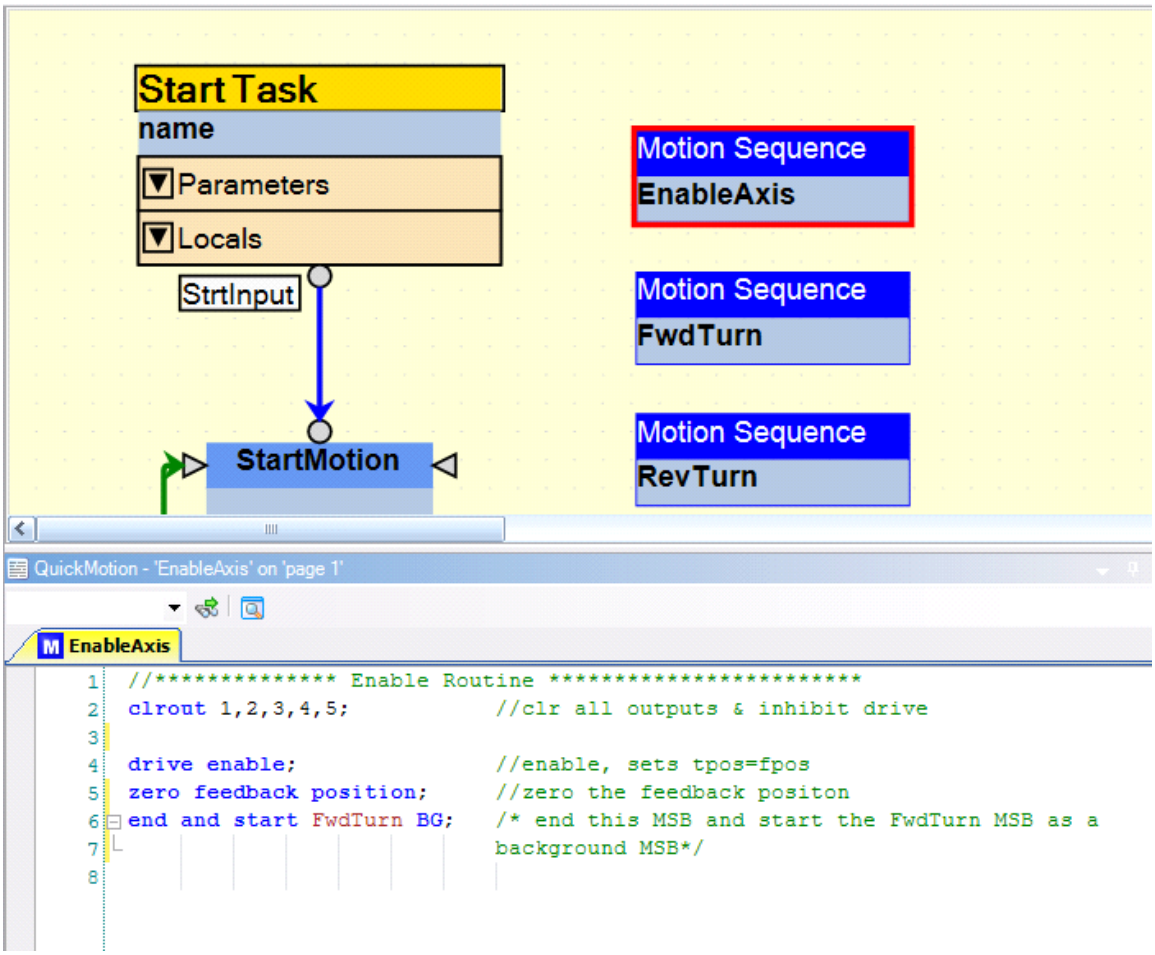


NOW: Repeat the above steps to create the additional 2 MSBs shown below, naming them **FwdTurn** and **RevTurn**.



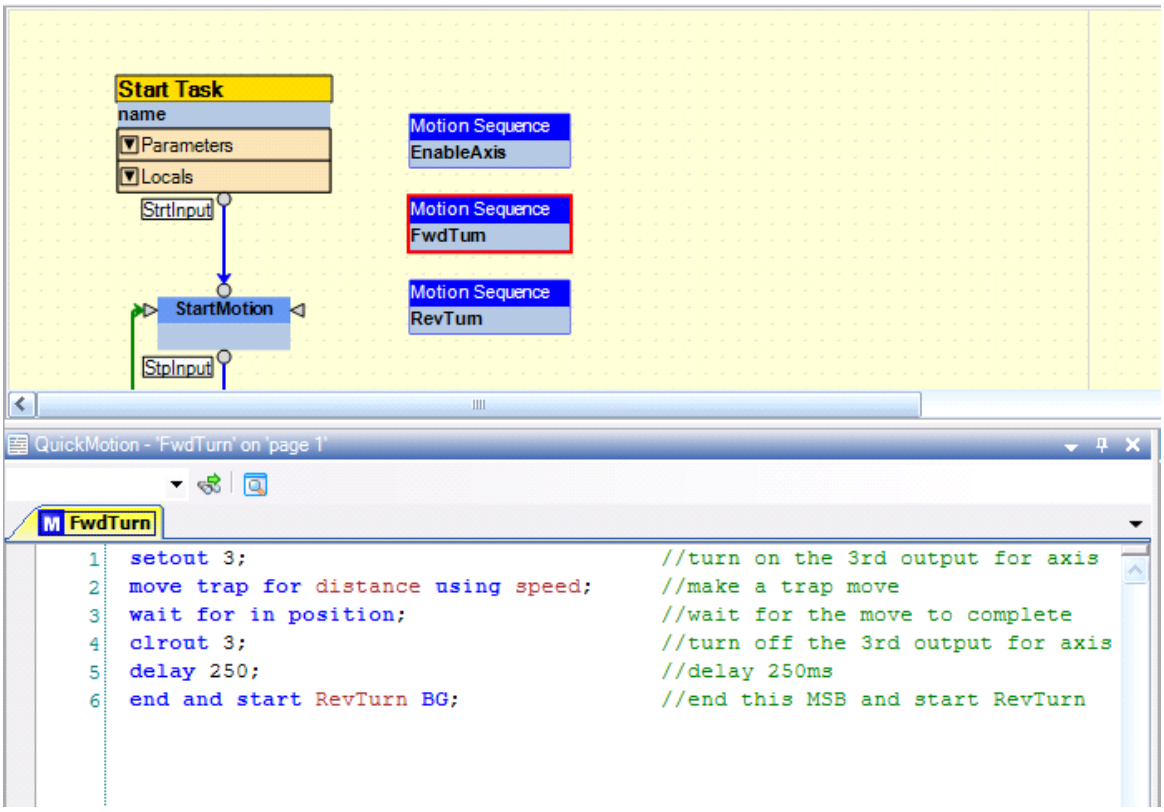
NOW:

1. Click on the **EnableAxis** MSB.
2. Type in the QuickMotion commands as shown below in the Code Editor:



NOW:

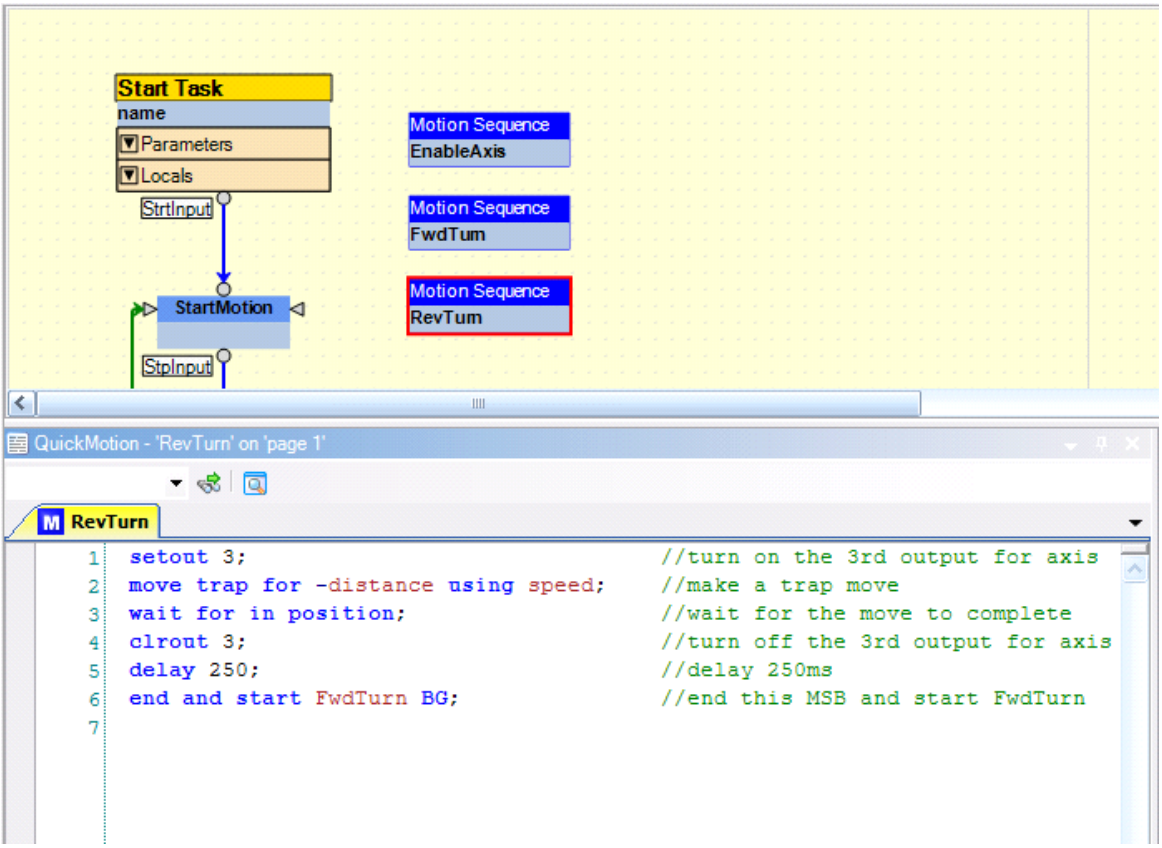
1. Click on the **FwdTurn** MSB.
2. Type in the QuickMotion Commands as shown below in the Code Editor:



☑ **Note:** There are 64 floating point variables available for each axis. They are automatically created when you type them and there is no need to define them. Both the distance and speed variable defined above will allow the QS4 task **SetParams** set to a value.

NOW:

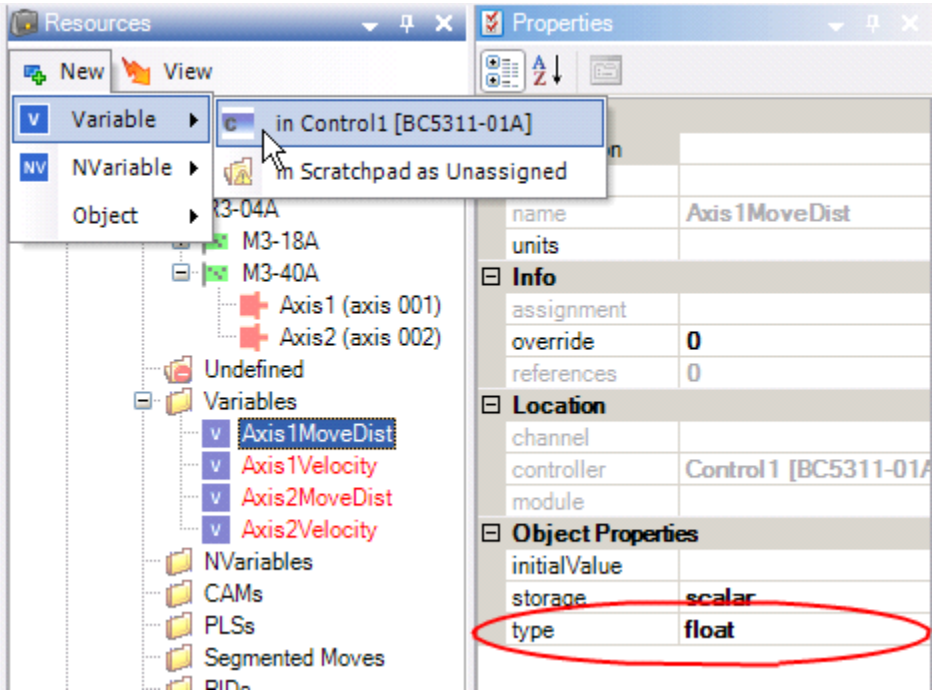
1. Click on the **RevTurn** MSB.
2. Type in the QuickMotion Commands as shown below in the Code Editor:



Create the 4 following floating point variables (shown below):

NOW:

1. Click on **New**.
2. Select **Variable**.
3. Select your controller.
4. Name them **Axis1Velocity**, **Axis2Velocity**, **Axis1MoveDist** and **Axis2MoveDist**.
5. Set their **type** Property to **float**.

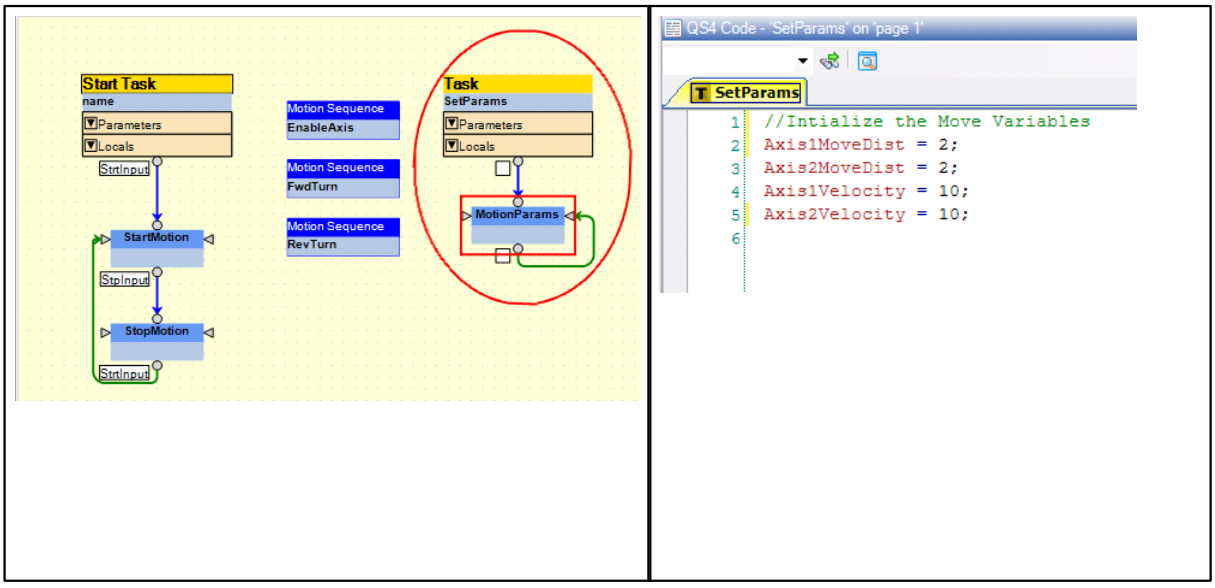


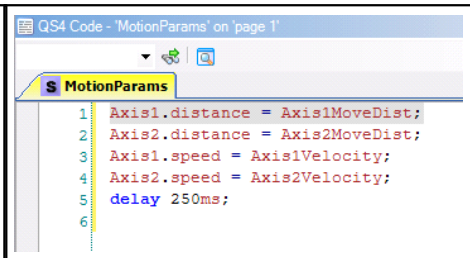
☑ **Note:** These QS4 Variable will be used in the QS4 task **SetParams** (that we will create next) to set the value of the 64 floating point variables distance and speed used in the MSB **FwdTurn** and RevTurn.

NOW:

1. Add the additional **SetParams** task to your existing program, using QuickBuilder as shown below left.

2. Add the QS4 Code for the **SetParams** task and **MotionParams** step as shown below right.





☑ **Notes:**

1. This task allows the QS4 variables to set the variables `distance` and `speed` used in the MSBs **FwdTurn** and **RevTurn**. Notice that you can set the distance and speed for the MSBs running for Axis1 and Axis2 separately, because each axis runs its own instance of that same MSB, conveniently allowing for modular programming.
2. If you are only using one axis, omit any references to Axis2. (i.e., `Axis2.distance = Axis2Movedist;` and `Axis2.speed = Axis2Velocity;`)

Now that the program is complete, translate and download the program to the controller:

☑ **Note:** During the translation process, QuickBuilder determines which MSBs are needed for which axes. It prepares a separate copy of the MSB for each axis that will need it. Also during this translation process, QuickBuilder will assign the MSB variables to the axes objects in the Resource Manager. So if you want to see which MSB variables are assigned to a particular axis, just translate the project, and the tree will be automatically updated.

NOW:

1. *Click* on the Translate icon.
2. *Click* on the Publish and Run icon.

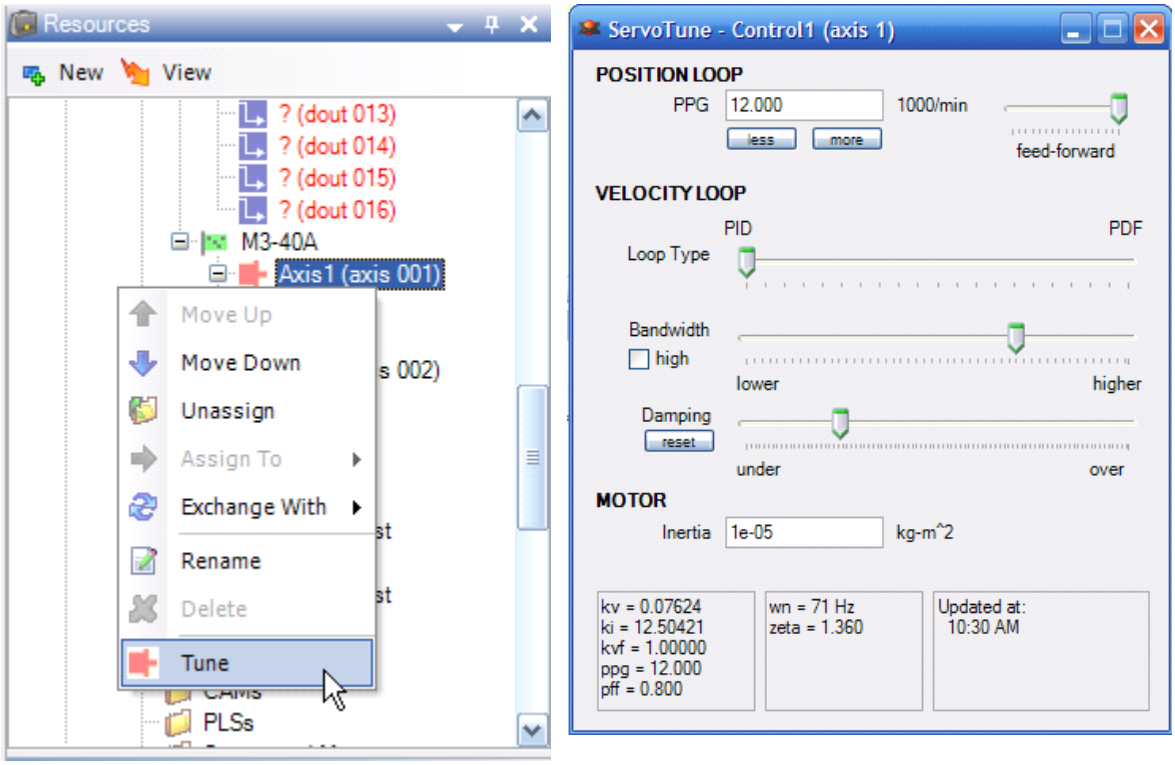
2.6 The Tuning Wizard

Once the program is in the controller you can start motion by turning on input 1, and you can open up the Tuning Wizard to tune your motor (note that input 2 will stop motion and disable the drive(s) at any time):

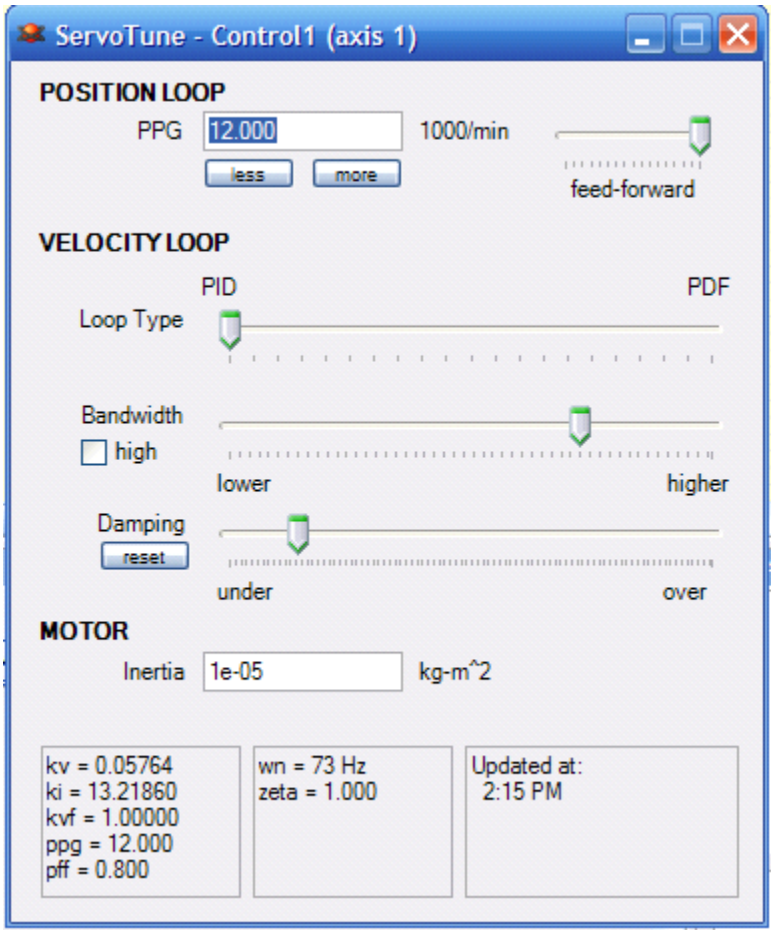
NOW:

1. Right Click on **Axis 1** in the Properties Window.
2. Select **Tune**.

☑ **Note:** The ServoTune Wizard shown on the right will pop up.


Resources
New
View
? (dout 013)
? (dout 014)
? (dout 015)
? (dout 016)
M3-40A
Axis1 (axis 001)
Move Up
Move Down
Unassign
Assign To
Exchange With
Rename
Delete
Tune
PLSs
ServoTune - Control1 (axis 1)
POSITION LOOP
PPG
12.000
1000/min
less
more
feed-forward
VELOCITY LOOP
PID
PDF
Loop Type
Bandwidth
high
lower
higher
Damping
reset
under
over
MOTOR
Inertia
1e-05
kg-m^2
kv = 0.07624
ki = 12.50421
kvf = 1.00000
ppg = 12.000
pff = 0.800
wn = 71 Hz
zeta = 1.360
Updated at:
10:30 AM

2.6.1 Basic Tuning



☑ **Notes:**

1. Each axis will have its own Tuning Wizard window. Multiple windows may be active and displayed simultaneously.

2. Tuning parameters adjusted using the wizard are updated in volatile memory.

   To save them to the non-volatile memory of the controller it is necessary to download the project to the controller after tuning.

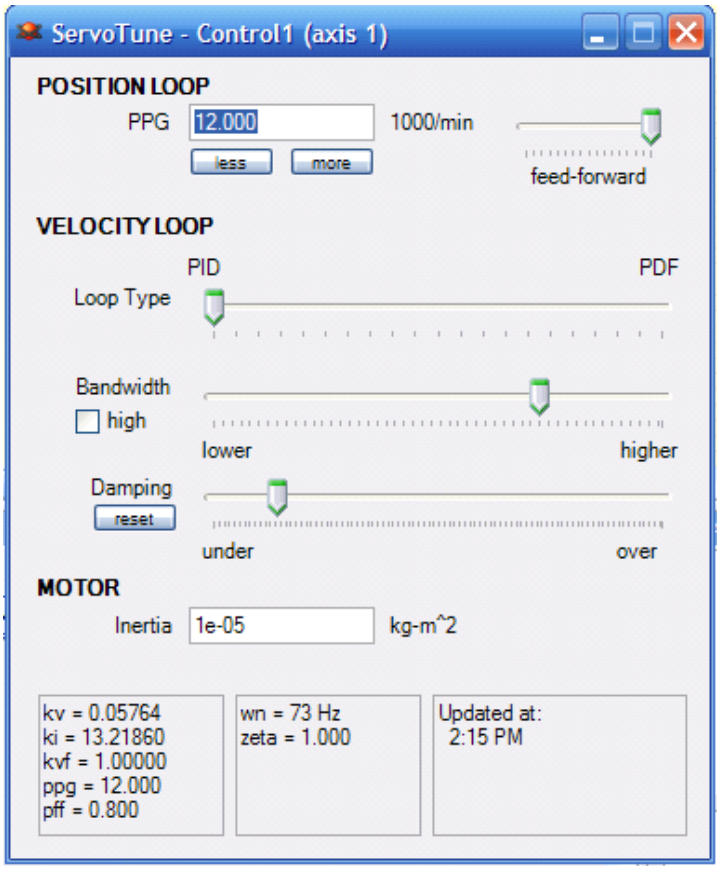
1. Enter the Inertia of **<u>your</u>** motor in kg-m^2. You can obtain this information from your motor manufacturer or supplier.
2. Adjust the Bandwidth slider until the desired performance is reached. Moving the slider to the right increases the servo loop bandwidth and hence the move performance. Checking the 'high' box causes the slider impact to be doubled.

☑ **Note:** If you move the Bandwidth slider too far to the right the motor will become unstable and begin to emit a buzzing sound and vibration even with the motor at rest. If this occurs, move the slider back to the left until this condition is eliminated.

Repeat these steps for Axis 2.

That is all that is required for Basic Tuning. If your system requires finer tuning continue on to Fine Tuning.

2.6.2 Fine Tuning



While the Basic Tuning method just discussed works well for most general purpose applications, higher performance applications or those with unusual loads or friction will typically require more adjustments.

For best results in fine tuning an axis it is useful to observe the velocity profile of the axis and how it responds to various adjustments to the tuning properties. This can be done by using QuickScope within QuickBuilder or by using an external oscilloscope to monitor the velocity output signal of the drive.

The fine Tuning Wizard adjustments are as follows:

PPG: This is the position loop proportional gain scaled in 1000/min units. This increases the response of the position loop and stiffness.

Damping: This has the affect similar to adding or removing friction from the system.

Feed-forward: This increases the position loop velocity feed-forward gain.

Loop Type: Adjust the loop type from 100% PID to 100% PDF structure.

For advanced applications, all of these parameters, with the exception of motor inertia, can be changed programmatically or interactively through a QuickBuilder Watch Window. There are also several other tuning variables available for the experienced motion engineer. See the Tuning Variables section in the QuickBuilder Reference Manual for details.

3 Appendix A: On Board IO

We did not cover the use of the onboard inputs so here is a summary of their use and some examples showing a few of the many possible uses of them within an MSB.

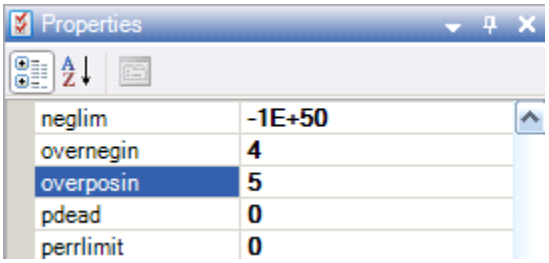
Each Servo axis card has its own 5 (2 axis) or 10 (1.5 axis) inputs and outputs assigned to it.

The inputs can be used as a home input, as hardware overtravel limits, as jog inputs, to initiate a pause, as a registration input and the list goes on.

The outputs can be used as moving/not moving outputs, as a pulsed output to a label machine, as a frequency output to a conveyor that accepts step and direction inputs, and many more uses.

Input examples

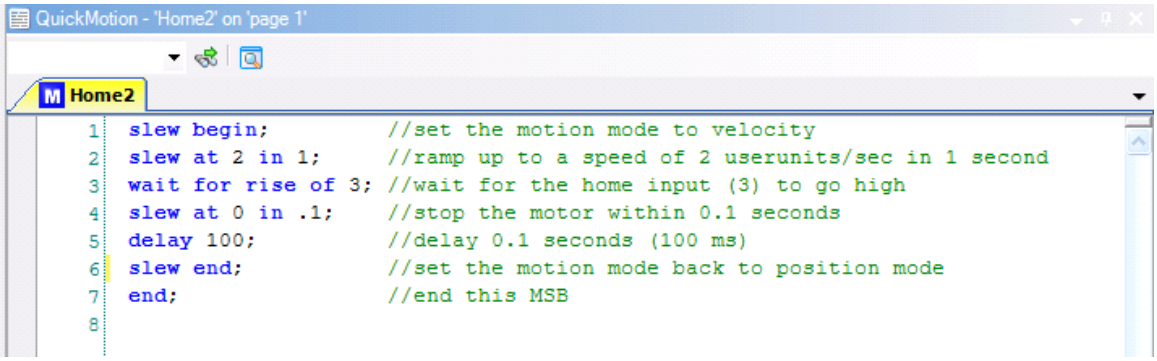
Hard limits are set up using the properties `overnegin` and `overposin` for each axis. The Properties Window shown below is set up for a negative overtravel limit on input 4 while the positive overtravel limit is set to input 5 on the selected axis.



Inputs can also be accessed using the wait for rise of and wait for fall of commands and if commands. We will now show you a few MSB examples of these commands in use.

Programmed inputs:

Here is an MSB using input 3 as a home input in a home routine:



Here is an MSB using input 1 as a pause input: (this will pause any motion for the axis while input 1 is on and allow it to continue once input 1 is off again)

QuickMotion - 'pause' on 'page 1'

pause

```
1  // This MSB will pause motion by moving the timebase to 0
2  // and then back to 1 based on input 1
3
4  [top]
5  wait for rise of 1;
6  timebase=0;
7  wait for fall of 1;
8  timebase=1;
9  goto top;
```

Here is an MSB using if statements to set-up a jog, based on three different inputs:

QuickMotion - 'jog' on 'page 1'

jog

```
1  // jog routine using inputs 1, 5, 2
2  JogSpeed=1;          // set the jogspeed to 1 user/unit
3  poslim=5;            // set a positive software travel limit
4  neglim=-10;          // set the negative software travel limit
5  slew begin;          // set motion mode to velocity mode
6  [loop]               //  loop here until input 2 is on
7  if !din1 && !din5 then speed=0; //don't jog until we see a jog input
8  if din1 then speed = JogSpeed;  //input 1 starts a positive jog
9  if din5 then speed = -JogSpeed; //input 5 starts a negitive jog
10 slew at speed in 0.5;
11 delay 510;                  // wait til at speed
12 if !din2 goto loop;         // if input 2 is on finish this MSB
13                             //if input 2 is not on go back to loop
14 slew end;                 //set the motion mode back to position mode
15 end;                      //end this MSB
```

Output example:

QuickMotion - 'MoveAt' on 'page 1'

MoveAt

```
1  move at speed for revs;      //set up an incremental move
2  wait until fpos > 1;         // move by hand to enable
3  pulse 2 for 500;             //pulse ouput 2 on for 500 ms
4  wait for in position;        // wait for the move to complete
5  setout 3;                    //turn on output 3
6  end;
7
8
```

4 Appendix B: Good Wiring Practices

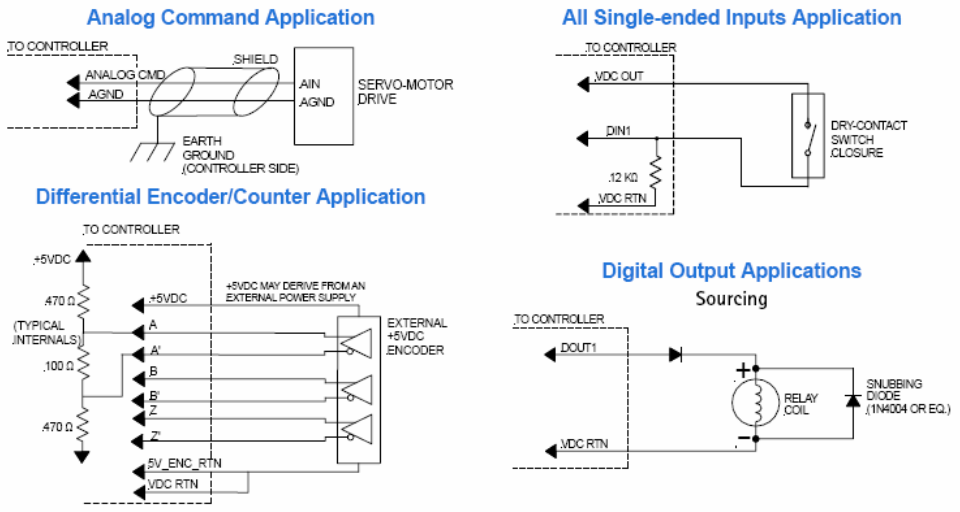
Following good wiring practices is your best course of action to ensure an "electrically" trouble free system. You can refer to our Reducing Noise Susceptibility Tech Note # 26 for more information on good grounding and wiring practices. Here are the basics:

Servo Motor/Drive Connections

You should refer to your servo motor and drive manufacturer for specific details, but in general you want to:

1. Make sure grounding and shield connections are made per the manufacturer's specifications. Remember to shield only one side of your cable to earth ground, typically on the side closest to your main grounding point.
2. Make sure motor cables are routed in a separate wireway or secured with tiedowns to ensure proper placement away from other cables, especially low voltage.
3. Try to keep a minimum of 12 inches separation between encoder or resolver feedback cables and the motor cables or any AC power cables, and place them in a separate conduit or wireway whenever possible.
4. If encoder or resolver cables must cross motor or AC power cables, try to arrange for them to cross at right angles to each other.

Controller/ServoDrive/Encoder Connections



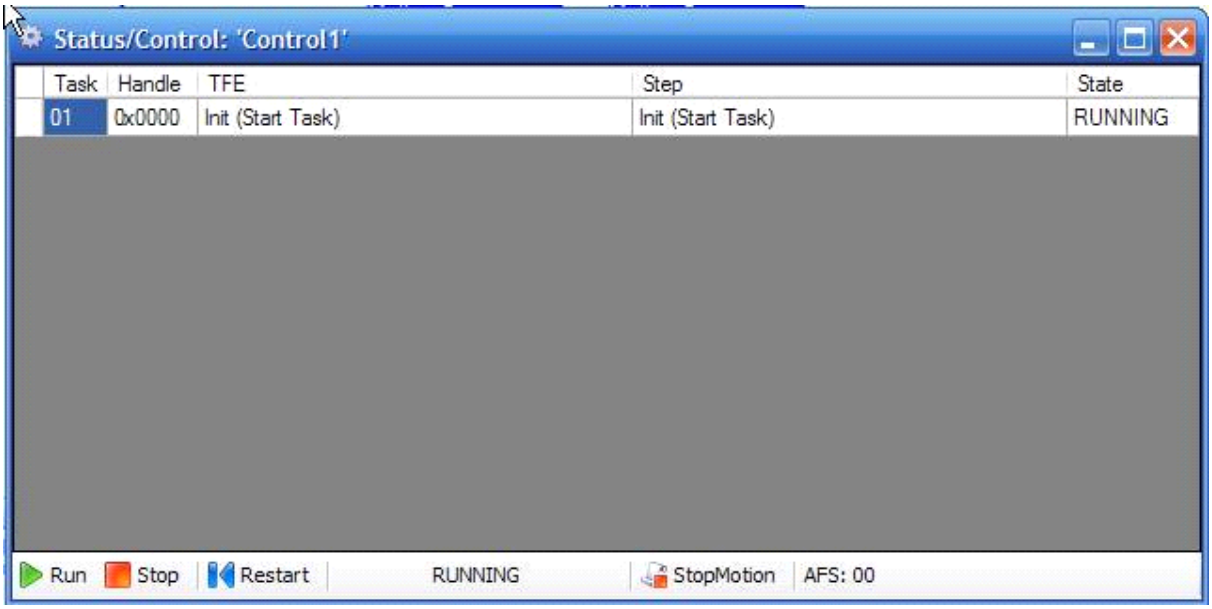
Your analog command and Encoder connections should be separated from both each other and from any high voltage/high current wires. If they must cross, see that they cross at right angles and do not run parallel. The use of shielded twisted pair wire over standard shielded wire is highly recommended, especially for the Servo/ Controller encoder and analog signal connections. Typically you want to connect your shields on the side closest to your main grounding point. Just be sure to connect only one side. Make sure you diode suppress any outputs connected to inductive loads including coil style relays. We recommend the use of a 1N4004 or equivalent, placed as close to the coil as possible.

5 Appendix C: Troubleshooting Tips

The Status/Control Window, Watch Window, along with QuickView and QuickScope all offer ways to troubleshoot your Motion application. For information on using QuickScope refer to the QuickScope™ User Guide: http://www.ctc-control.com/customer/techinfo/docs/5300_951/951-530032.pdf .

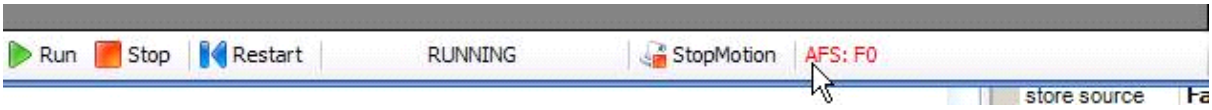
5.1 Status/Control Window

The Status/Control Window gives you access to the Axis Fault Status and allows you stop motion.



The StopMotion icon at the bottom of the Control/Status Window allows you to Stop motion on all motion axes.

The AFS:00 in the lower left corner of the Control/Status Window tells you there are two motion axis and both are okay.



The red AFS:F0 shown above tells you that the first motion axis has a fault. Clicking on the AFS:F0 will bring up the following Fault Display.

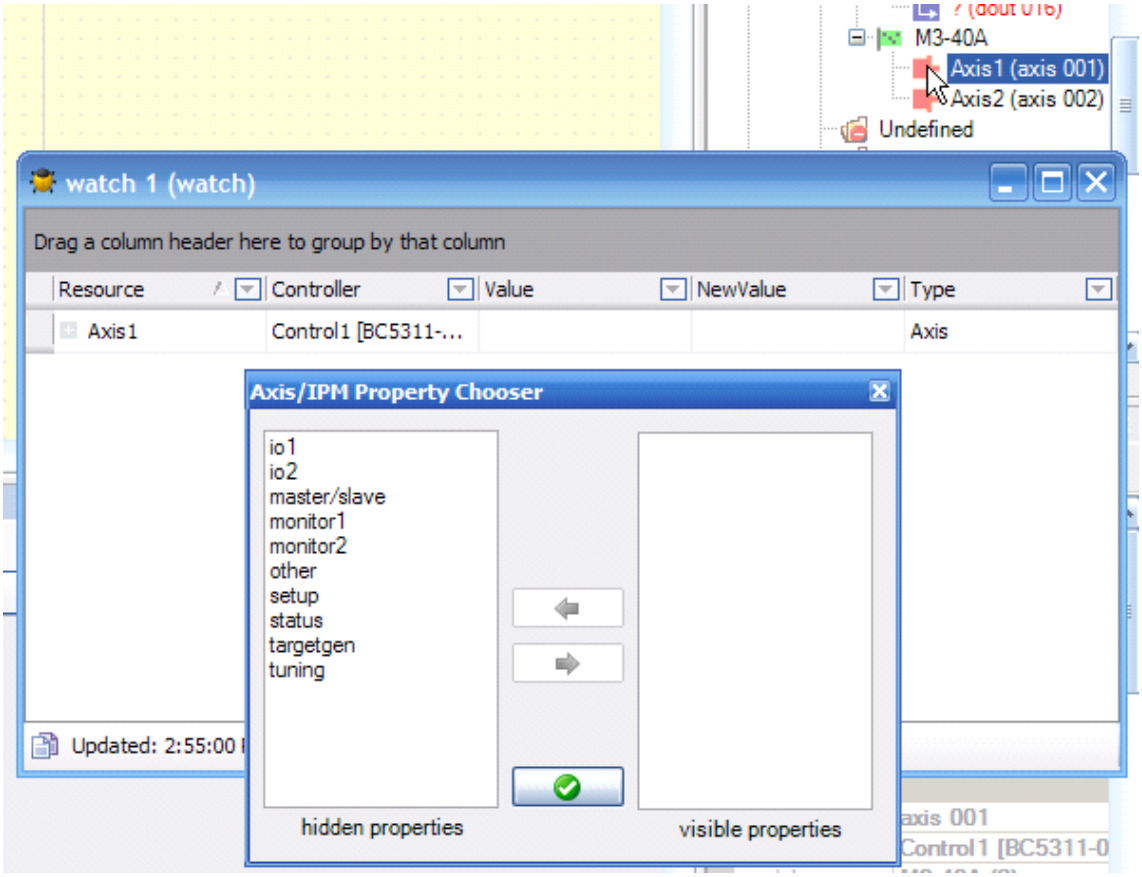
Clicking on **<EXIT FAULT DISPLAY>** in this window takes you back to the standard Status/Control Window.

5.2 Watch Window

Motion parameters can be monitored and changed via a Watch Window.

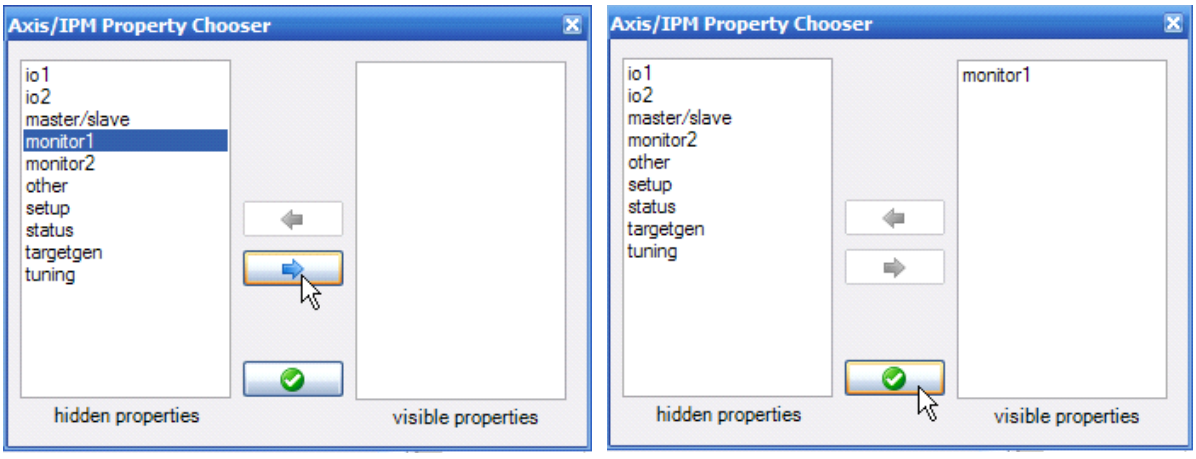
Adding axis parameters to a Watch Window is as simple as dragging and dropping the Axis from the Resources Window and to the Watch Window.

This will bring up an Axis/IPM Property Chooser Window as shown below.



This allows you to select which items will be displayed for that axis.

Simply select which selection of Properties you would like to display in left hidden properties list and use the Right arrow add them to the list on the right as shown below. To remove an item click on the item you want to remove in the visible properties list on the right and click on the Left arrow.

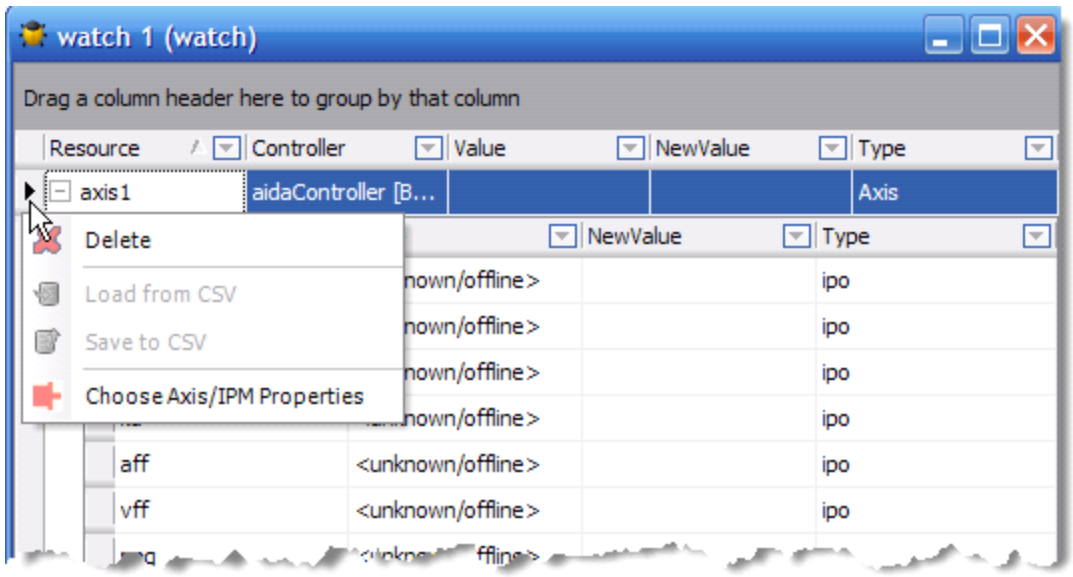


Below is the listing of properties for each Property group:

io1	din1, din2, din3, din4, din5 dout1, dout2, dout3, dout4, dout5 dins, douts	io2 (1-1/2 axis only)	din6, din7, din8, din9, din10 dout6, dout7, dout8, dout9 dout10
master/slave	sfposc, sfmod	other	time, tlim, ztpos, zfpos
monitor2	tposc, fposc, verr, ZPULSE_POS, ZPULSE_NEG theta, ztheta	monitor1	tpos, fpos, perr, trqc, vcmd, vel
setup	stoprate, tmax, vmax, ppr, mppr, uun, uud, inposw, poslim, neglim, invertfeed, invertmaster, invertcmd, overposin, overnegin, driveenable, perrlimit, running	status	enabled, inpos, overneg, overpos, overtrq, rmstrq, jogging, cmode, faulted, fault1, fault2, fault3, fault4, zpulse
targetgen	acc, dec, jerk, timebase	tuning	kvf, kv, ki, kd, aff, vff, ppg, pff, pdead, kgain, kfilt

For a description of these properties see the Model 5300 QuickMotion Reference Guide
http://www.ctc-control.com/customer/techinfo/docs/5300_951/951-530017.pdf

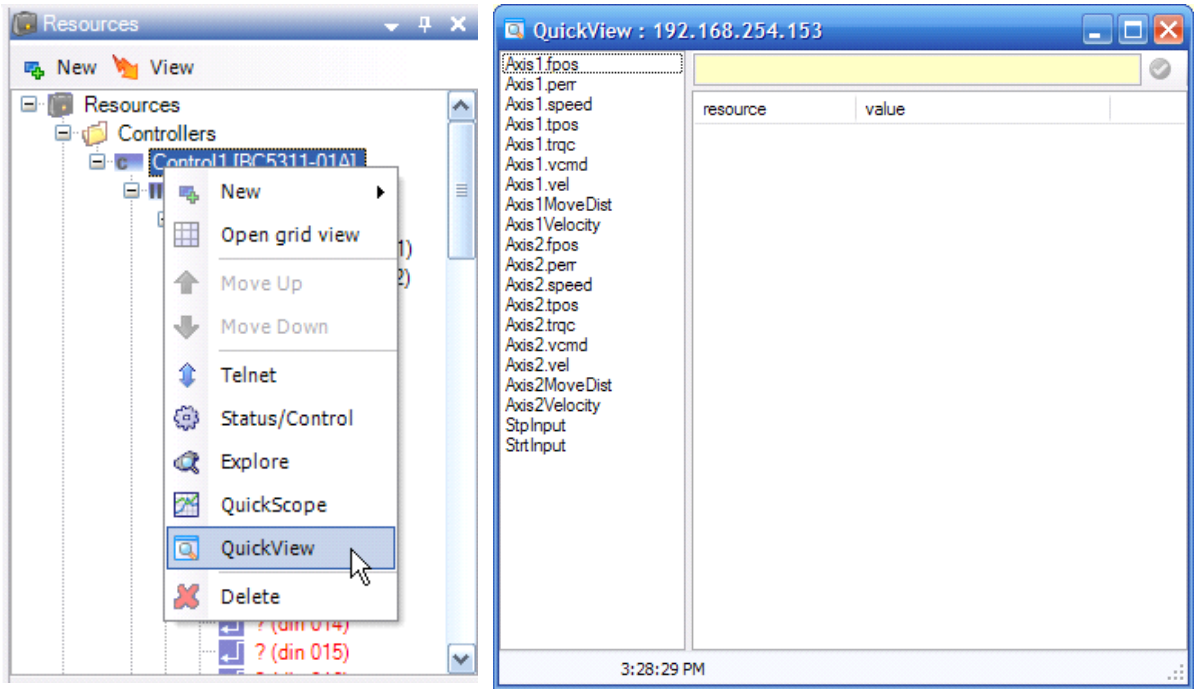
To add or remove lists from the axis property list at any time simply right click to the left of the axis in the Watch Window and select **ChoseAxis/IPM Properties** as shown above. This will bring up the **Axis/IPM Property Chooser** and allow you to add/remove items.



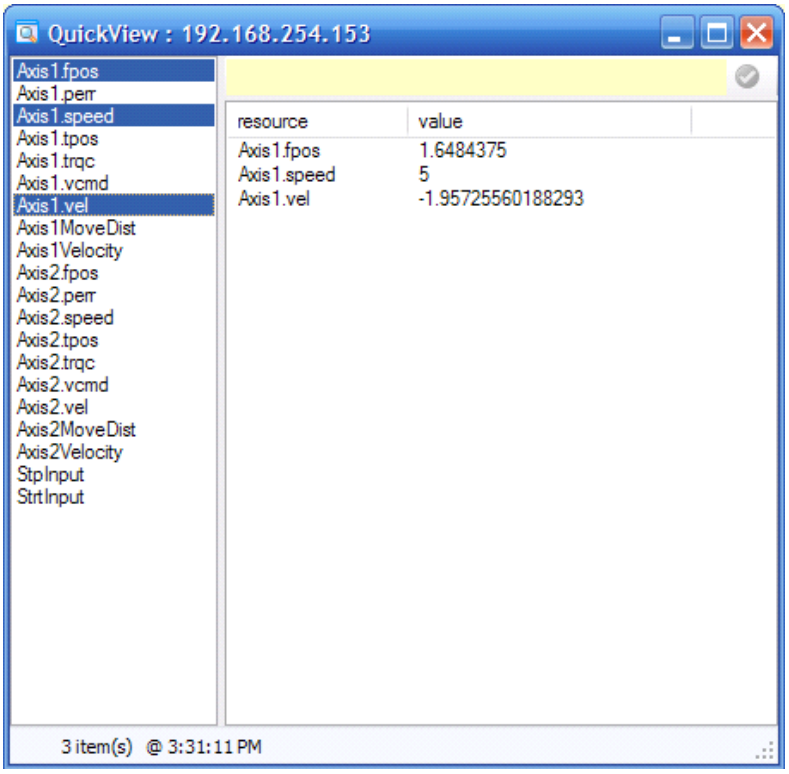
5.3 QuickView

QuickView gives you an easy way to individually select Motion Axis properties to view and edit.

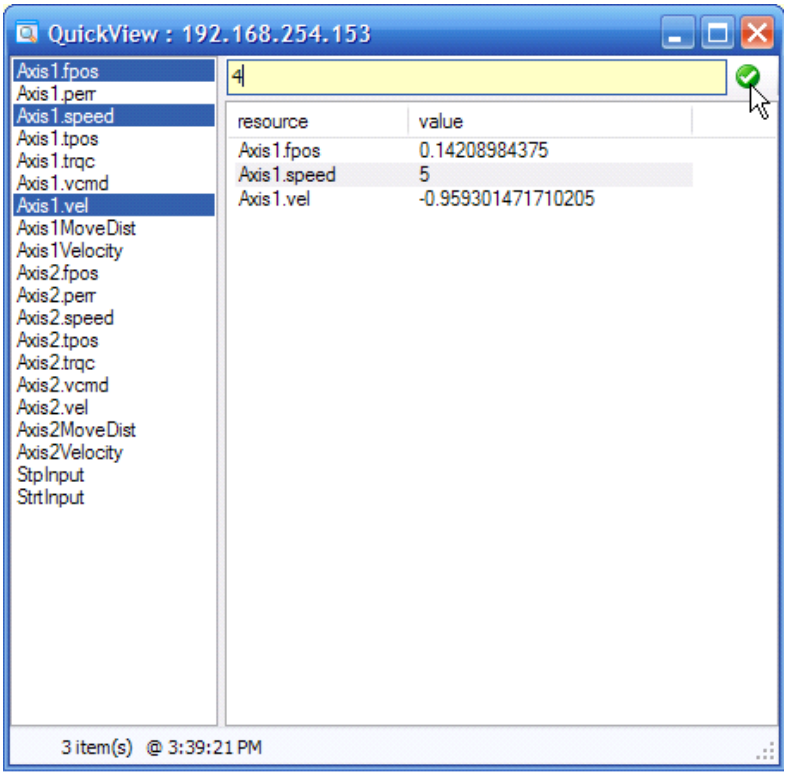
QuickView is initiated with a right-click on the Controller and then selecting Quickview as shown below left. QuickView Window comes up as shown on the right.



You can click on any of the Motion Axis properties or variables shown and they will be monitored on the right.



To change a read/write property being monitored, click on it in the monitor window, type the new value in the yellow box at the top and then click on the green arrow.


QuickView : 192.168.254.153
Axis1.fpos
Axis1.perr
Axis1.speed
Axis1.tpos
Axis1.trqc
Axis1.vcmd
Axis1.vel
Axis1MoveDist
Axis1Velocity
Axis2.fpos
Axis2.perr
Axis2.speed
Axis2.tpos
Axis2.trqc
Axis2.vcmd
Axis2.vel
Axis2MoveDist
Axis2Velocity
StpInput
StrtInput
4
resource
value
Axis1.fpos
0.14208984375
Axis1.speed
5
Axis1.vel
-0.959301471710205
3 item(s) @ 3:39:21 PM

Index

- A -

- M -

- Q -

- S -

- T -

- w -