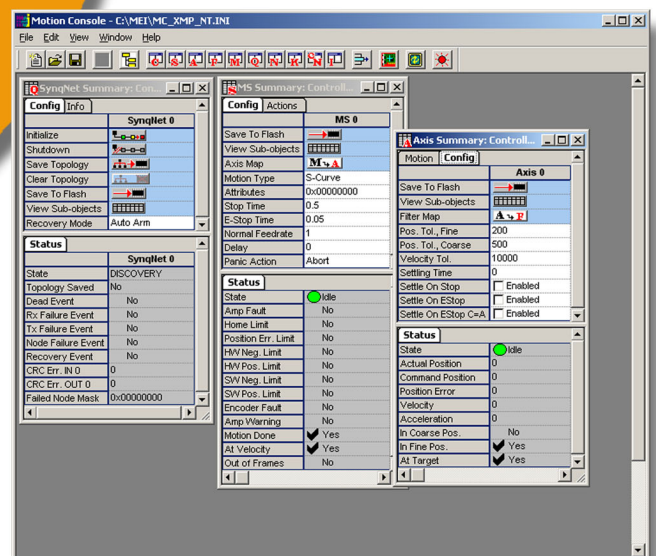
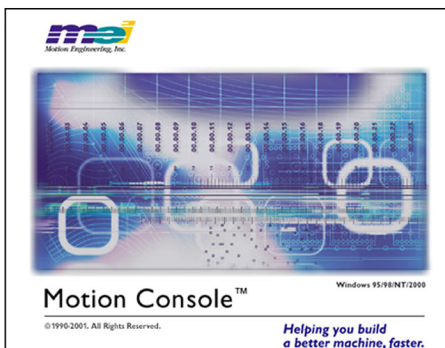


Motion Console

User Guide



Contents

Requirements and Safety

Introduction

Required Materials

Safety Precautions

Hardware Requirements and Installation

Frequently Asked Questions (FAQs)

Motion Console

Object-oriented Motion Control

Motion Console Objects

Motion Console User Interface

- Main Motion Console Frame

- Adding a New Controller

- Object Summary Windows

- Object Explorer

- Object List Configuration Dialog Boxes

Configuring New Systems with Motion Console

- Getting Started with Motion Console

- Saving Parameter Settings

- Saving to Flash Table

- Associating Objects with Motion Console

- Configuring a Motion Supervisor

- Mapping One Object to Another

- Removing (Deleting) a Mapped Object

- Safety Reminders

Motion Console Objects

- Controller Object

- Motion Supervisor Object

- Axis Object

- Filter Object

- Motor Object

- SynqNet Object

- SqNode Object

- CAN Network Object

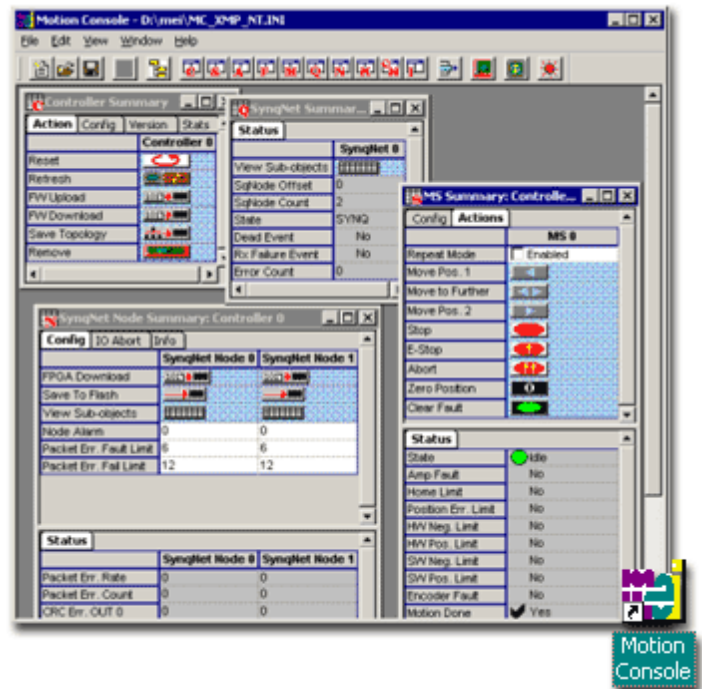
- CAN Node Object

- I/O Object

Motion Console Command Line Options

Reading Versions with Motion Console

Troubleshooting



Version 04.01.03

Introduction

To fully understand Motion Console, Motion Scope and the VM3 utility, you will need to experiment with hardware. Ideally, a test stand has been obtained for this purpose so that you may experiment with the XMP to the fullest extent possible. This reduces risks to personnel and property. (If you are new to XMP, your first motion experiments should not be with a 20-ton wrecking crane!) This section will assist you in getting your hardware connected. You should then proceed to the [Motion Console](#) section to begin using your XMP motion controller.

Required Materials

- [ZMP-SynqNet](#) or [XMP-SynqNet](#) motion controller, [RMB-10V2](#), [Cables](#), [STC-136](#).

(servo amplifier(s) or stepper motors)

-OR-

[ZMP-SynqNet](#) or [XMP-SynqNet](#) motion controller, [SynqNet drives](#), motors.

- Hand tools (sufficient to open host computer and install controller card, secure cables, etc.)

Safety Precautions

Mechanical Hazards

The following mechanical precautions should always be observed, especially when operating large, powerful components:

Crush-Collision Danger to Personnel -- Even small mechanical components can exert sufficient force to crush or injure you! Before powering your motion control system, verify that personnel, including their hands, fingers and feet are clear of the safety zone. If you have not already done so, define a safety zone, then employ barriers and/or clearly marked boundary lines so that personnel are excluded from the zone. Before powering motion control components, verify that all personnel remain outside the safety zone.

Crush-Collision Hazard to Equipment -- The same precautions applied to personnel should be observed for equipment. Components can be damaged or destroyed due to collision with other components or fixed hardware.

Component Hazards -- Motion control may be applied to components which themselves present hazards. These include welders, saws, lasers, presses, cutters, etc. Be aware that the introduction of movement to a component extends the reach and range of that component's hazards. In addition, any object can be made dangerous if suddenly jarred loose or pitched due to sudden, wild movements. When testing new applications, it is recommended that you begin at very low speeds, then slowly increase to the design speed.

Electrical Hazards

Electrocution Hazard -- Some motion control systems utilize high voltages and currents which present hazards to personnel. During installation and servicing of components:

- Power off all electrical components and accessories connected to your motion control system.
- Unplug electrical devices from their source of power. Safe-lock all power plugs to prevent them from being plugged in during servicing.
- Allow sufficient time for components using capacitors to discharge fully before making connections.
- Do not repower any components until installation is complete and personnel are clear of component spaces.

ESD -- Due to the use of sensitive microelectronics, all XMP boards are subject to damage due to electrostatic discharge (ESD). When handling XMP components, users are advised to observe basic ESD precautions, including the use of grounding straps and grounding mats. Removed XMP components should always be stored in protective ESD bags and/or cases.

Hardware Requirements and Installation

Hardware: Controllers

Controller hardware requirements and detailed installation instructions can be found in the following hardware sections. [SynqNet-XMP](#), [SynqNet-ZMP](#)

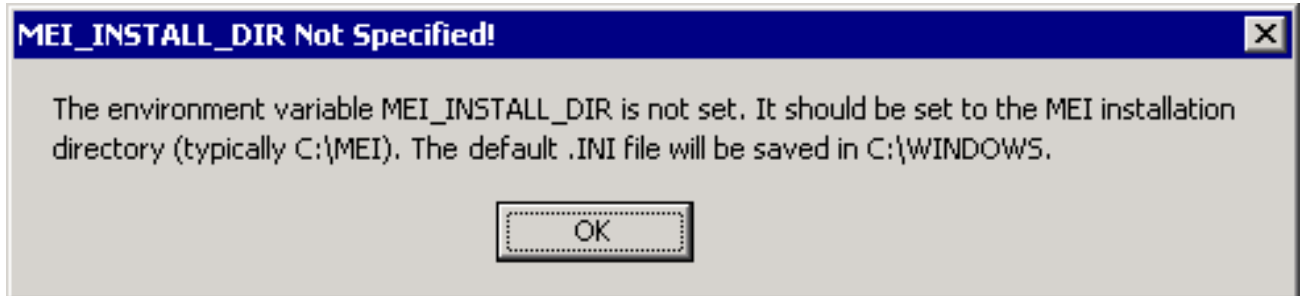
Software

Operating System -- Operates on WindowsNT 4.0, 2000, or XP. Detailed software installation instructions are included with the current release note of your software version.

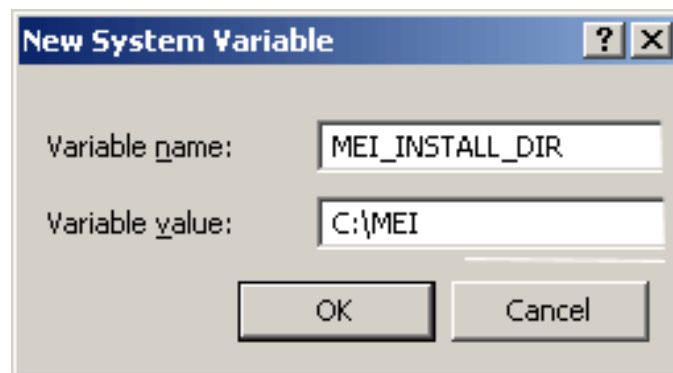
See Also: [Release Notes](#)

FAQs for Motion Console

How do I resolve the "MEI_INSTALL_DIR Not Specified" error?



In Windows, open the Control Panel > System. In the System Properties window, under the Advanced tab, click on Environment Variables button. Under the System Variables section, click the New button. Type in the following variable information.



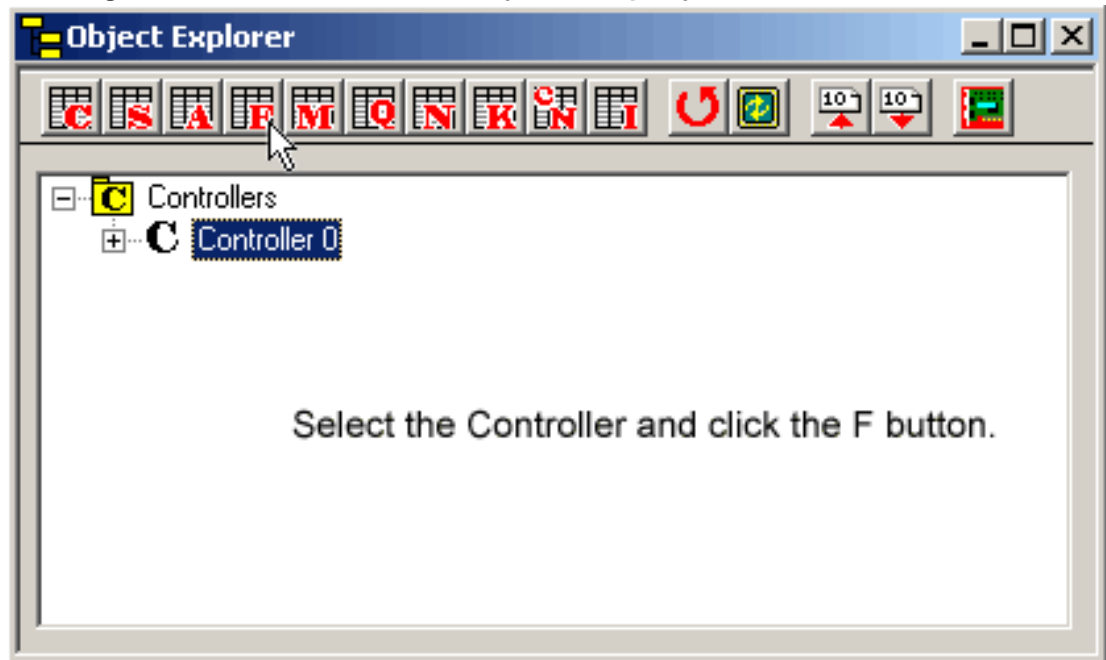
NOTE: If you have installed the MDK into a different directory, you must change the Variable Value accordingly.

Is there a way to save a copy of what is displayed in a Summary window?

Selected contents of a Summary window can be copied into the clipboard and pasted into any application that accepts text from the clipboard. Furthermore, data copied from other applications, such as Microsoft Excel, can be pasted into a Summary window. Below is an example of

saving the filter coefficients to an Excel spreadsheet.

1. Configure the Filter Summary to display all filters on a controller



2. In the Filter Summary, select coefficients 0 – 12.

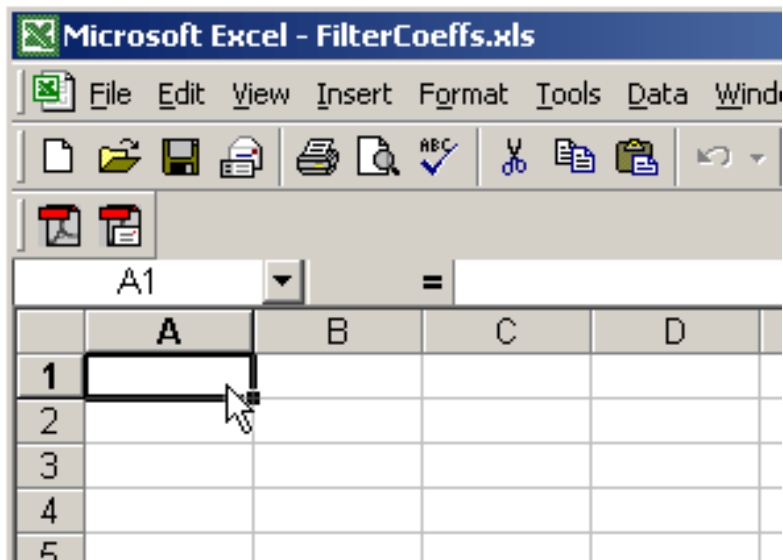
The screenshot shows the 'Filter Summary: Controller 0' window with the 'Coeffs' tab selected. The table below shows the coefficients for four filters. A mouse cursor is hovering over the cell for Coefficient 0 in Filter 0.

	Filter 0	Filter 1	Filter 2	Filter 3
Coefficient 0	256	256	256	256
Coefficient 1	0.5	0.5	0.5	0.5
Coefficient 2	2048	2048	2048	2048
Coefficient 3	0	0	0	0
Coefficient 4	0	0	0	0
Coefficient 5	0	0	0	0
Coefficient 6	0	0	0	0
Coefficient 7	0	0	0	0
Coefficient 8	32767	32767	32767	32767
Coefficient 9	0	0	0	0
Coefficient 10	32767	32767	32767	32767
Coefficient 11	32767	32767	32767	32767
Coefficient 12	-32768	-32768	-32768	-32768
Coefficient 13	0	0	0	0

Select the data by clicking the top left-hand cell and then the bottom right-hand cell while holding down the Shift key.

3. Click the Edit/Copy menu item or type the Copy shortcut, Ctrl+C.

4. Open up Microsoft Excel. Click on a cell.



5. Click on the Edit/Paste menu item or type the Paste shortcut, Ctrl+V. The coefficients will be copied into the spreadsheet.

The screenshot shows the same Microsoft Excel interface, but now the spreadsheet is populated with data. The formula bar shows 'A1' and the value '256'. The data is as follows:

	A	B	C	D
1	256	256	256	256
2	0.5	0.5	0.5	0.5
3	2048	2048	2048	2048
4	0	0	0	0
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0
8	0	0	0	0
9	32767	32767	32767	32767
10	0	0	0	0
11	32767	32767	32767	32767
12	32767	32767	32767	32767
13	-32768	-32768	-32768	-32768
14				

6. The reverse procedure can be used to copy and paste data into a Summary window in Motion Console.

Object-oriented Motion Control

The XMP motion control system is distinguished by its unparalleled flexibility in design. This is largely the result of two features:

1. A firmware-configured DSP processor which can be customized for efficient tasking.
2. Actual-library software architecture containing both general (MPI) and specialized (MEI) software objects.

These objects encompass complex motion control functions, enabling the programmer to quickly build programs without writing hundreds or thousands of lines of code. Objects lend extraordinary portability to the software environment, allowing complex functions to be executed by calling them as single objects.

Motion Console embodies this object-oriented approach to motion control through its unique graphical user interface. It has been designed as a utility to assist programmers. It does not write custom applications. Rather, it provides an environment for the testing and monitoring of motion control components. This will prove extremely helpful to you when writing new applications, because it allows you to check your code against a proven interface. (For example, if your application fails to move a component, you can try Motion Console to see whether the problem lies within the hardware or software.)

This section will guide you through the basics of Motion Console, and includes a description of its objects and attributes. If you would like to try configuring a simple motion control system with Motion Console, please see the [Quick-Start Guide](#), which accompanies MEI's Firmware Developer's Kit. Another helpful section is the [Configuring New Systems with Motion Console](#) section.

Motion Console Objects

Motion Console divides motion control into several distinct software objects. Some of these objects have direct hardware equivalents, such as Motor objects. Other objects are more abstract and complicated, such as Filter objects. Motion Console provides direct configuration and monitoring of XMP motion control objects, including:

Motion Controller – A single XMP controller, capable of controlling motion supervisors, and served by a host computer. A Controller folder lists the controllers served by the host computer, of which there may be more than one. However, each Controller demands its own separate motion controller hardware (i.e., one main controller board, with or without an attached expansion board).

Motion Supervisor – Topmost level of motion control associated with a Controller. Each Motion Supervisor, in turn, has 0 (zero) or more Axes mapped to it.

Axis – A motion vector associated with either linear (e.g., linear slide), or rotary (e.g., turntable) motion, and associated with a Motion Supervisor. Each Axis has 0 (zero) or more Filters mapped to it.

Filter – Attributes applied to closed-loop motor control, such as gains and motion algorithms. Each Filter has 0 (zero) or more Motors mapped to it.

Motor – A motor, which may be either rotary (such as a rotary motor shaft), or linear (such as a linear motor, pneumatic cylinder, hydraulic actuator, etc.).

SynqNet – A SynqNet object manages a single SynqNet network that is connected to a motion controller. It represents the physical network. It contains information about the network state, number of nodes, and status.

SqNode – A SqNode object manages a single SynqNet network node that is connected to a SynqNet network. It represents the physical network node. It contains information about the node, as well as its status and configuration. It provides read/write access to the node via network cyclic data and service commands. It also provides an interface to any drives connected to the node.

CAN Network – A CAN Network object manages a single CANOpen network connected to a motion controller. It represents the physical network. It contains information about the network state and configuration.

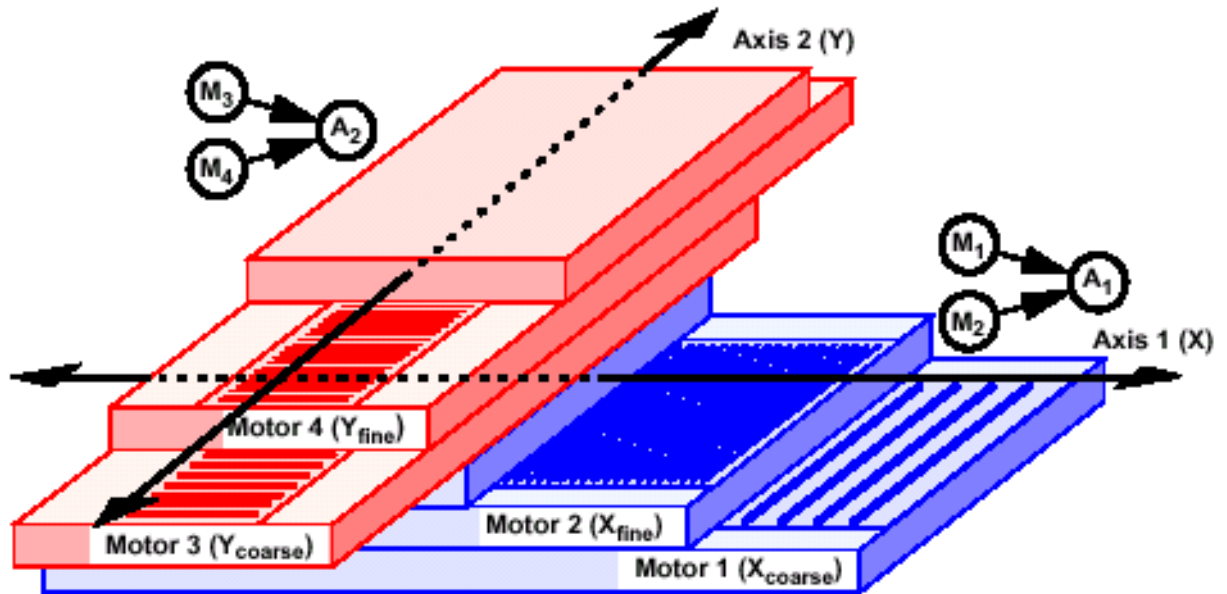
CAN Node – A CAN Node object manages a single CANOpen network node connected to a CANOpen network. It represents the physical network node. It contains information about the node, as well as its status and configuration.

I/O – I/O objects in Motion Console represent the individual I/O associated with other objects. Currently, Control, SqNode, CAN Node, and Motor objects support I/O, so the I/O Summary contains separate tabs for the I/O associated with those objects.

Mapping


"Mapping" means associating one object with another. For example, one axis may have one motor mapped to it (such as a motor-driven leadscrew on a slide), or one axis may have two motors mapped to it (such as a gantry crane).

Frequently, motion designers think of each axis in terms of a single motor; however, the XMP environment allows you to expand this model. For example, the X-Y table on a three-axis machine is simply controlled by two motors, each of which represents a single, independent axis. However, it may be advantageous to add a second motor to each axis, in order to obtain independent, rapid-coarse motion and slow-fine motion. In this model, each axis has two motors associated with it: one coarse, one fine. The choice of when to use what motor is performed by a **Filter** object, written into the custom application code. If the fine motor is commanded to move to a point outside its range, the filter object can be configured to use the coarse motor to reposition the axis. Once the coarse motor has positioned the axis at its starting point, the filter will switch to the fine motor. This level of flexibility is made possible by object mapping.



Motors 1 and 2 (M1 and M2) are mapped to Axis 1 (A1) on the X-axis. Motors 3 and 4 (M3 and M4) are mapped to Axis 2 (A2) on the Y-axis.

This allows one component to be associated with another by simply dragging one object to another on the computer screen with a mouse.

IMPORTANT! Object mappings, along with all other Motion Console settings, are NOT saved until the **Save to Flash Memory**  function is used on the Object Explorer or object summary window. If you do not save your settings to flash memory, your settings will be lost when the system is powered down, or when the controller is reset.

Before using Motion Console to map objects, it will help to review how the user interface is designed.

Motion Console User Interface

Main Motion Console Frame

The outermost Motion Console window frame presents a toolbar at the top with several buttons. Each button function is summarized below.



Toolbar Functions



<Alt> E

View Error List – MPI library function errors are displayed in the Library Function Errors window. When there are no errors to display, the View Error List button is disabled. Otherwise, it is enabled and clicking on it opens the Library Function Errors window.

To empty the list, click on the Reset button. To close the list, click on either the Close button, or the close ("X") icon in the upper-right corner of the window. Some MPI error messages are described in the Software Reference.



<Ctrl> N

New – Clicking on the New Profile button will create a new initialization file (.INI), or profile. The default setting will initially display only the Object Explorer box with no controllers.



<Ctrl> O

Open – Clicking the Open Existing Profile button will open a specified .INI file.



<Ctrl> A

Save As – Clicking the Save Profile As button will save the current profile into the specified .INI file.



<Alt> O

View Object Explorer – Clicking on the View Object Explorer button displays the Object Explorer window. See the [Object Explorer](#) section for more information.



<Alt> C

Open and Configure Controller Summary – Clicking on this button opens and configures the Controller Summary window. See the [Controller Objects](#) section for more information.



<Alt> S

Open and Configure Motion Supervisor Summary – Clicking on this button opens and configures the Motion Supervisor Summary window. See the [Motion Supervisor Objects](#) section for more information.



<Alt> A

Open and Configure Axis Summary – Clicking on this button opens and configures the Axis Summary window. See the [Axis Objects](#) section for more information.



<Alt> F

Open and Configure Filter Summary – Clicking on this button opens and configures the Filter Summary window. See the [Filter Objects](#) section for more information.



<Alt> M

Open and Configure Motor Summary – Clicking on this button opens and configures the Motor Summary window. See the [Motor Objects](#) section for more information.



<Alt> Q

Open and Configure SynqNet Summary – Clicking on this button opens and configures the SynqNet Summary window. See the [SynqNet Objects](#) section for more information.



<Alt> N

Open and Configure SqNode Summary – Clicking on this button opens and configures the SqNode Summary window. See the [SqNode Objects](#) section for more information.



<Alt> K

Open and Configure CAN Network Summary – Clicking on this button opens and configures the CAN Network Summary window. See the [CAN Network Objects](#) section for more information.



<Alt> D

Open and Configure CAN Node Summary – Clicking on this button opens and configures the CAN Node Summary window. See the [CAN Node Objects](#) section for more information.



<Alt> I

I/O Summary – Clicking on this button opens the I/O Summary window. See the [I/O Objects](#) section for more information.



<Ctrl> R

Configure Grid Rows – Clicking on this button displays a window to configure the Row Configuration of the highlighted Object Summary window. This allows the configuration of each window to display only the required parameters. This will simplify the Motion Console interface and hide parameters that should not be modified. The Row Configuration settings can be saved to an initialization file.



<Alt> NumKey +

Add Controller – Clicking on this button adds a new Controller object. See "Controller Objects" section for more information.



F5

Refresh – Clicking on this button will refresh the Motion Console display. While the Status tabs at the bottom of the Object Summary windows are constantly updated, the Configuration tabs at the top are only updated when manually changed by the user, when Motion Console is restored after being minimized, or when the Refresh button is clicked. This is particularly helpful if another application is also accessing

the XMP controller and changing the controller configuration.



<F 12>

Panic Button – Clicking on this button activates a Panic action for each Motion Supervisor.



WARNING! The Panic Stop behaves according to the user-specified Panic Action parameter located in the Motion Supervisor Summary / Config tab. Depending upon how Panic Action is configured, a panic stop may initiate a slow or rapid motion stop, an error state, or no error at all. For details on configuring your Panic Stop, see the "Motion Supervisor Object" section below in this chapter. **You must custom-configure your Panic Stop according to the level of control and safety required by your motion system!**

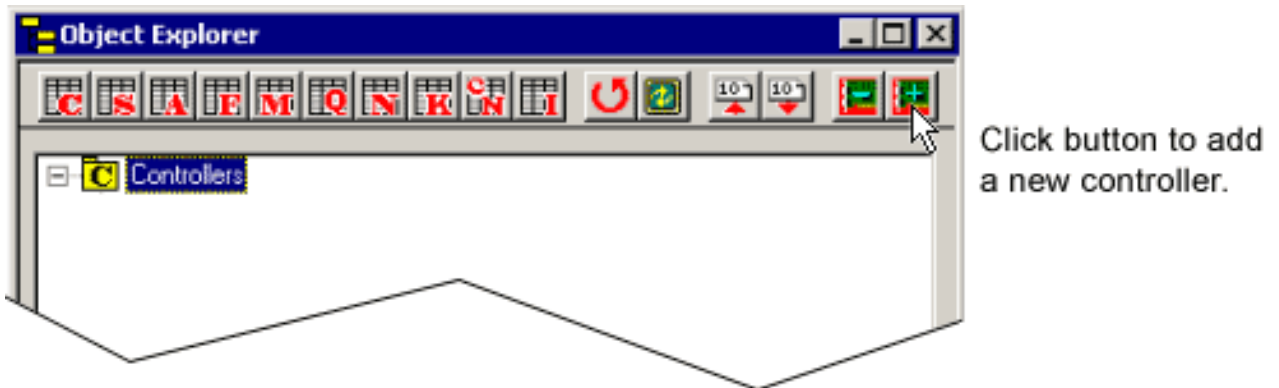
NOTE– The Shift and Ctrl keys can be used to modify the action performed when clicking on a toolbar button that opens a window. The modified actions are listed below:

- **Shift + button** – Changes the state of the window. If the window is closed, it will be opened. If the window is open, it will be closed.
- **Ctrl + button** – If the window is already open, it activates the window and brings it to the foreground. This is useful for locating a window that is buried under other windows.

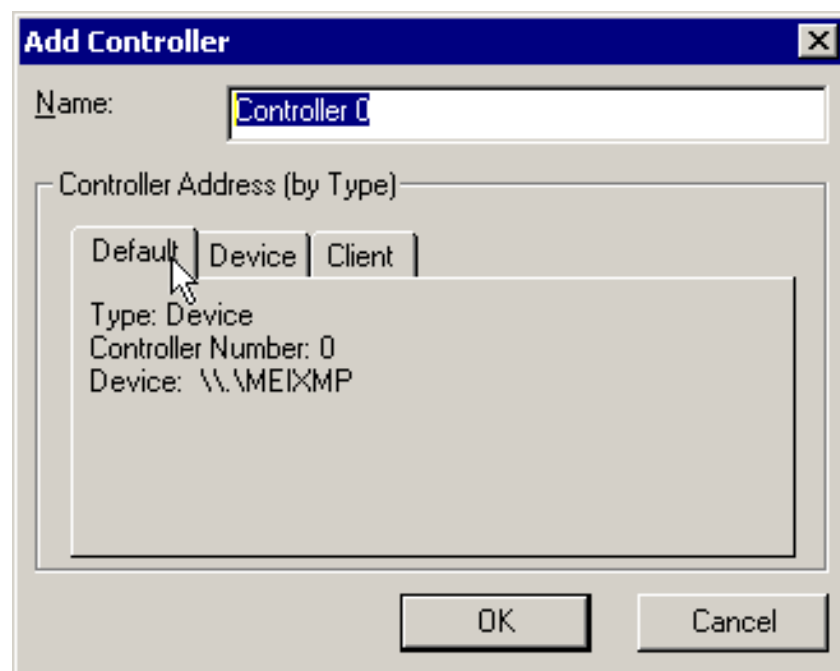
Furthermore, for those buttons that open object summary windows, the Ctrl and Shift keys will cause the Summary Object List Configuration dialog box to be bypassed. The Summary window will be opened to the current configuration.

Adding a New Controller

Recall that each Controller is represented by an actual, physical controller. Therefore, a new Controller cannot be added until a new motion controller has first been installed in the computer. After installation is complete, click on the **Add Controller** button on the toolbar of the Object Explorer to add a new controller.



Clicking on the Add Controller button displays the Add Controller dialog box. Enter the name of the new Controller in the Name field. This may be any alphanumeric label up to 27 characters long ("Controller 1," "XY Table," "Julie Ann," etc.). The Name does not need to be the same as the Controller Number, but many users find this less confusing.

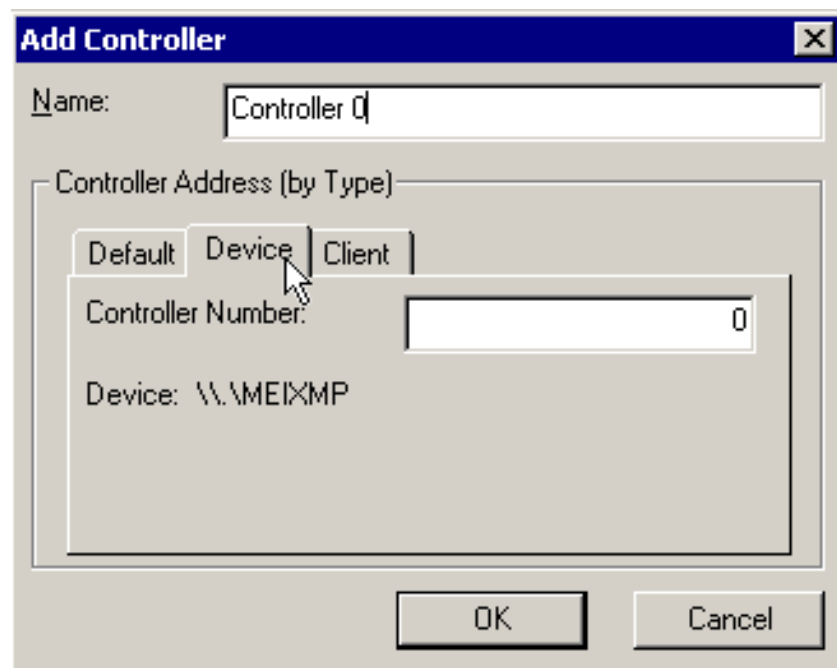


There are three ways to address a new controller:

- If only one controller is being installed on the host computer, simply assign a Name, then click on the **OK** button. The Controller number will be 0. This is the Default option.
- Click on the **Device** tab if there is more than one controller card in the computer.
- Click on the **Client** tab if the controller card resides inside a separate client computer (i.e., not in the same computer running Motion Console), linked across a TCP/IP network with it.

Device Settings

Device settings should be used whenever more than one XMP controller card is in the host system. If ALL controllers reside inside the host, you may ignore settings on the **Client** tab.



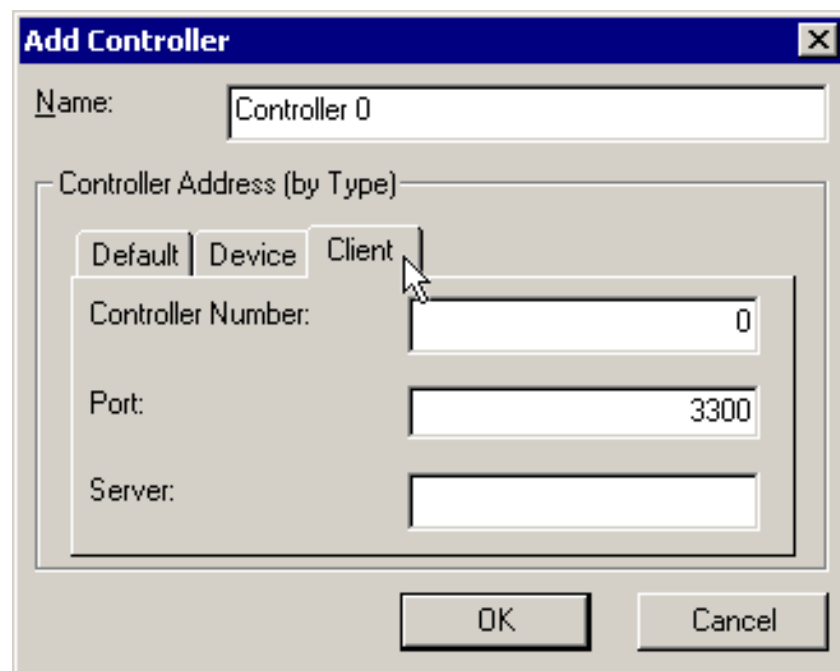
Controller Number – the MEI-specific index, inside the device list, that is used by the computer to identify each controller. Although you may customize the name of the controller, you cannot customize the "Controller Number," because it is already preconfigured to coordinate with the

appropriate XMP controller board. For example, you can call a controller "Martha," but if the Controller Number is 3, Motion Console will be referencing the 3rd board installed on your system.

Client Settings

NOTE: Before a client-based controller can be added to the object tree, the client computer must be running the [Server.exe](#) utility. If you have not already done so, start the utility now before proceeding. Once server.exe is running, it can be "minimized" on the client computer's monitor and kept running in the background.

Client settings apply to XMP controllers which reside on separate client computers (i.e., computers other than the one running the Motion Console utility). To access a client controller, its client computer must be networked with the host computer and be running the server.exe application.



The screenshot shows a Windows-style dialog box titled "Add Controller". It has a "Name:" label followed by a text box containing "Controller 0". Below this is a section titled "Controller Address (by Type)" which contains three tabs: "Default", "Device", and "Client". The "Client" tab is currently selected, and a mouse cursor is pointing at it. Inside the "Client" tab, there are three labels with corresponding text boxes: "Controller Number:" with the value "0", "Port:" with the value "3300", and "Server:" with an empty text box. At the bottom of the dialog are "OK" and "Cancel" buttons.

Controller Number – the MEI-specific index, inside the device list, that is used by the computer to identify each controller. Although you may customize the name of the controller, you cannot customize the "Controller Number," because it is already preconfigured to coordinate with the appropriate XMP controller board. For example, you can call a controller "Martha," but if the Controller Number is 3, Motion Console will be

referencing the 3rd board installed on your system.

Port – Socket connection to use for the client. In most cases this will be the same for host and client(s), and the default value (3300) can be used.

Server – IP address of the client computer serving the XMP controller.

Error Messages

If error messages are encountered while adding a new controller, refer to the table below.

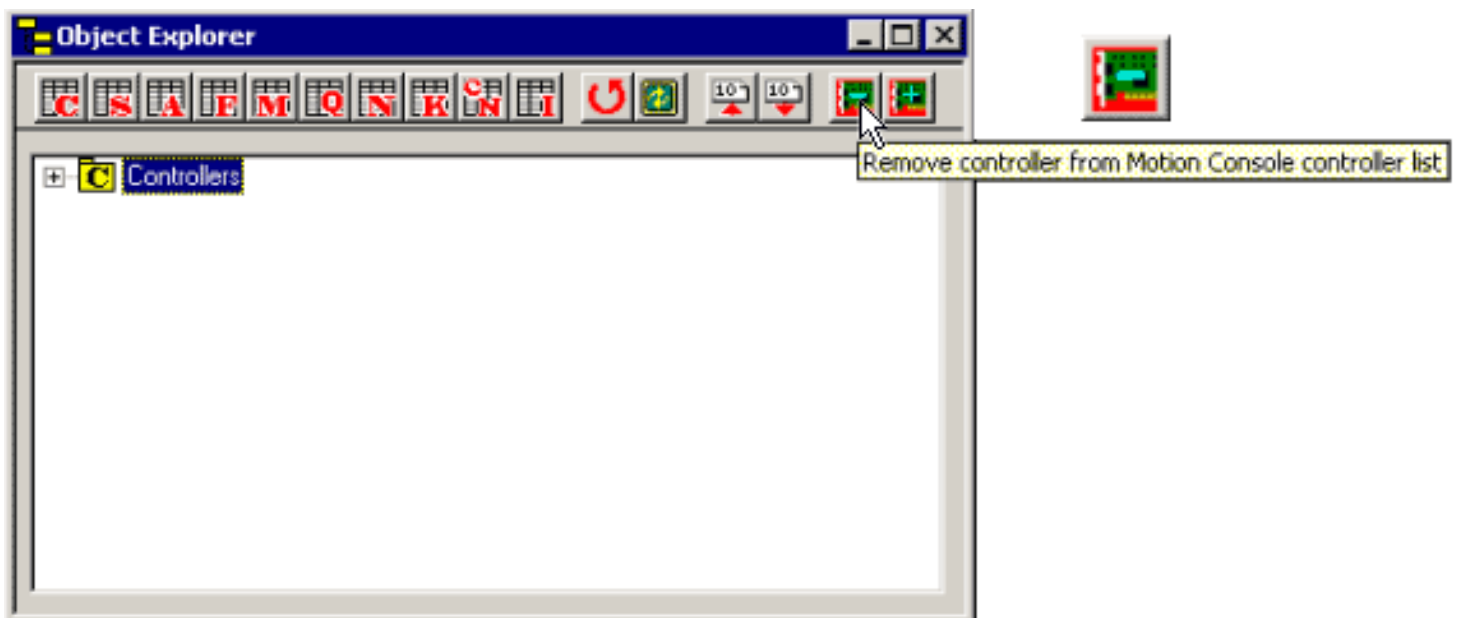
Controller Error Messages

	Error Message	Controller Type	Action	
A	mpiControllnit failed while creating (controller name)	Device	Recheck hardware installation, especially PCI bus socket.	A
B			Open the "Control Panel" and double-click "Devices." Find the "MeiXMP" device and make sure its status is "started."	B
C			Swap XMP board with a reliable card in a reliable host computer. If card fails to respond, contact MEI.	C
D		Client	Recheck client address.	D
E			Verify that client is running server.exe application.	E

F			Refer to Actions A and C above.	F
G		Device	Verify Controller Number field under the Device tab is different from existing controller(s).	G
H		Client	Verify that Controller Number, Port and Server fields under the Client tab are different from existing controller(s).	H
I	MPIControllInit() returns 0x8603:Packet:communication error	All	Verify that client is running server.exe application.	I

Removing Controllers

Controllers may be removed quickly with the Object Explorer by highlighting the controller to be removed and clicking on the Remove Controller icon. A controller can also be removed by clicking on the Remove button for the controller on the Operations tab of the Controller Summary window.



A Note Regarding Synchronized Motion

Designers should bear in mind that Motion Console does NOT support synchronized motion between separate controllers. (For example, you cannot perform a precision, interpolated move between Controller 0 and Controller 1 using only Motion Console.) However, coordinated motion between controllers on the order of milliseconds is supportable from user-written applications. Please contact an MEI applications engineer.

Object Summary Windows

Object Summary windows display the attributes of individual objects using a grid control, much like a spreadsheet. Each object is represented by a column and the attributes are represented by rows. For some objects, there are so many attributes that they are divided and categorized into several tab windows. The user can configure which objects are to be displayed by an Object Summary.

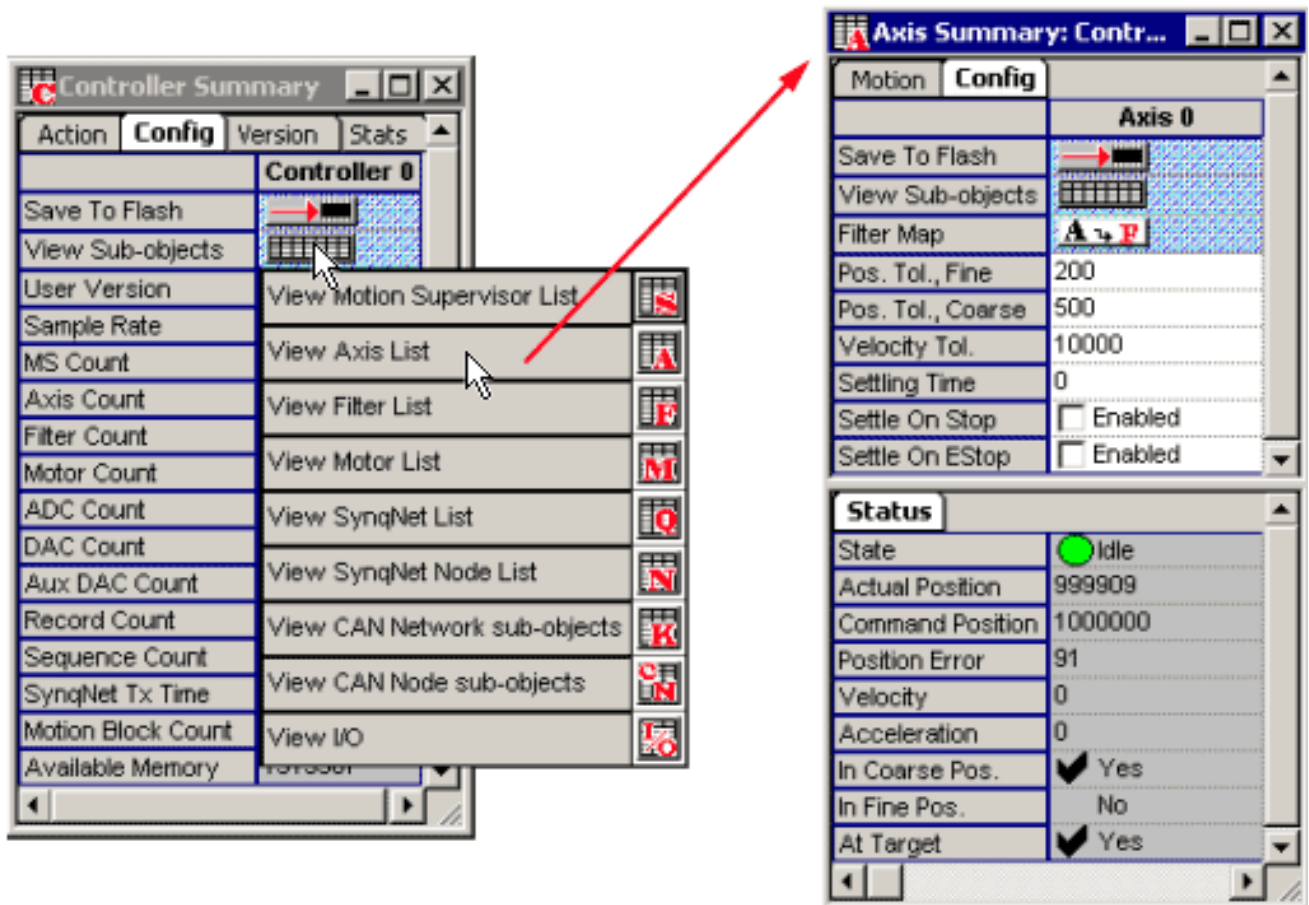
Configuring Object Summaries

Summary windows are configured to display objects directly or by association. For example, the Axis Summary can be configured to display a set of axes, such as Axis 0, Axis 1, and Axis 2, or it can be configured to display all the axes associated with a set of super-objects, such as MS 0, MS 1, and MS 2.

Object Summaries can be configured by using 1) the Object Explorer, or 2) the Object List Configuration dialog box, or 3) the View Sub-objects buttons on another Summary window. See the sections related to these windows for more information. The View Sub-objects buttons are described below.

View Sub-Objects

The View Sub-objects button is displayed as a general configuration item on most object windows. A "sub-object" is meant to signify any object that is mapped to another object, either directly or indirectly. The summaries shown are identical to those presented by clicking on summary icons in the Object Explorer. For example, clicking on the **View Sub-objects** button within the Controller Summary window displays eight types of subordinate objects:



A lower-level object such as the Axis object will display fewer sub-object types. Sub-object summaries are displayed by selecting them with the mouse. More than one summary can be selected from a list by simultaneously holding down the key while clicking on both.

Anatomy of an Object Summary

Configurable Attributes" versus "Status Parameters

Some summary windows are divided into two sections. The top section of the window lists attributes which may be altered through direct data entry; these are configurable. The bottom section of the window contains read-only information regarding the status of objects; these are not configurable. An example of one such panel (an Axis summary) is shown below.

The screenshot shows a software window titled "Axis Summary: Controller 0". It contains two main sections: "Motion" and "Status".

Configurable Section (Motion): This section is labeled "Configurable" with a blue bracket. It contains a table with columns for "Axis 0", "Axis 1", and "Axis 2". The rows represent various motion parameters that can be manually entered.

	Axis 0	Axis 1	Axis 2
Position 1	30000	30000	30000
Position 2	0	0	0
Relative Distance	0	0	0
Velocity	10000	10000	10000
Acceleration	100000	100000	100000
Deceleration	100000	100000	100000
Jerk Percent	0	0	0
AccelJerk	1e+006	1e+006	1e+006
DecelJerk	1e+006	1e+006	1e+006

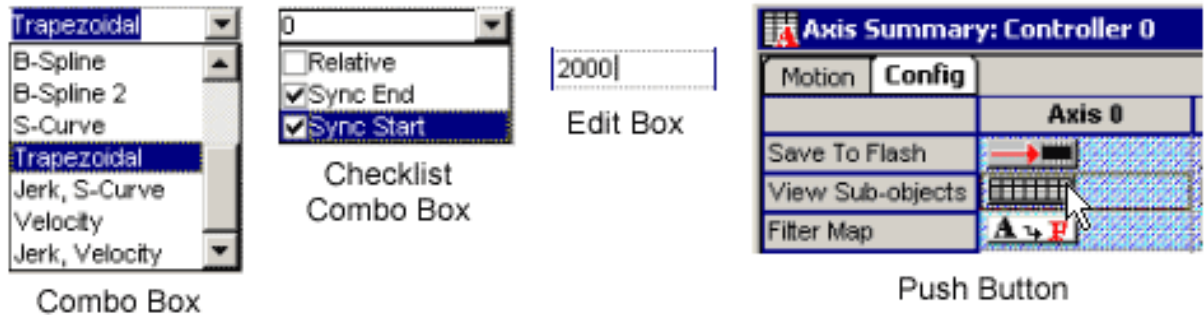
Read-only Status Section (Status): This section is labeled "Read-only Status" with a blue bracket. It contains a table with columns for "Axis 0", "Axis 1", and "Axis 2". The rows represent status information that cannot be manually changed.

	Axis 0	Axis 1	Axis 2
State	Idle	Idle	Idle
Actual Position	0	0	0
Command Position	0	0	0
Position Error	0	0	0
Velocity	0	0	0
Acceleration	0	0	0
In Coarse Pos.	No	No	No
In Fine Pos.	Yes	Yes	Yes
At Target	No	No	No

In this example, the bottom, read-only parameters provide status information about the axis; they cannot be manually changed. Attributes in the upper portion of summary windows may be altered by manually entering values.

Cell Controls

Each grid cell contains a user interface control that displays data and, for configurable attributes, allows the user to enter data. Below are shown some controls found within an object grid.



Edit Box – Data is entered manually into the cell using the keyboard.

Combo Box – Clicking on a cell yields a pull-down menu of options, which can be selected, or data can be manually entered via the keyboard. Combo boxes consist of two parts: 1) the edit box; 2) a drop-down button. If you choose to type in your selection manually, you must enter it exactly as shown in the list of options. Clicking on drop-down button will cause a window to appear that displays the option list. You may then click on a selection. If you do not want to use a mouse for entry, type to display the choice list and then use the navigation keys to make a selection. Every control type has a method for entering data without using the mouse.

Checklist Combo Box – Similar to the regular combo box, except it has selectable check boxes.

Push Buttons – Clicking on the button will cause an action to be performed on the selected object. If multiple buttons are selected in a row, then the action will be executed on all selected objects. Selecting buttons on multiple rows is not allowed. Buttons can also be activated by selecting the cell containing the button and pressing the Space bar.

Button Grid – This is a set of Push Buttons displayed in a grid, with each button appearing in a separate row. Each individual button behaves exactly as a regular push button. Multiple buttons may be selected.

CAUTION! Parameter values take effect immediately. If you are entering a value which influences machine movement, be certain to keep clear of moving components!

Navigating Within a Summary

Mouse-less Navigation within a Summary Window

Tab – Move the focus to the next control in the window.

Shift + Tab – Move the focus to the previous control in the window.

<Arrow Key> – Within a grid control, set the current cell to the adjacent cell in the direction of the arrow key. The arrows keys are also used to change the tab page on a tab control.

Ctrl + <Arrow Key> – Set the current cell to the extreme cell in the direction of the arrow key. For example, Ctrl + sets the current cell to the right-most cell in the current row.

Ctrl + Home – Set the current cell to the top, left-most cell.

Ctrl + End – Set the current cell to the bottom, right-most cell.

Selecting Cells

To select a single cell within the table, click on the cell. To toggle the selection state of a single cell, hold down the Ctrl key while clicking on the cell. To select a range of cells, click on the first cell in the range, then hold down the Shift key while clicking on the last cell in the range.

An entire row of cells can be selected by clicking on the row header button. An entire column can be selected by clicking on the column header button.

Cells can also be selected with the keyboard:

Shift + <Arrow Key> – Set the current cell to the next cell in the direction of the arrow key and select both the new and previous current cell.

Shift + Ctrl <Arrow Key> – Set the current cell to the extreme cell in the direction of the arrow key and select all cells between the former current cell and the new current cell.

Shift + Ctrl + Home – Set the current cell to the top, left-most cell and select

all cells in the range between the former current cell and the new current cell.

Shift + Ctrl + End – Set the current cell to the bottom, right-most cell and select all cells in the range between the former current cell and the new current cell.

Ctrl + Enter – Toggle the selection state of the current cell.

Copying Cell Data


Selected cells can be copied to another group of cells by dragging the selection with the mouse and dropping them onto the destination cells.

Selected cells can also be copied into the clipboard by clicking the Edit/Copy menu item or by typing the keyboard shortcut Ctrl+C. Once the cell contents are in the clipboard, they can be pasted into another set of cells by selecting the first target cell and clicking the Edit/Paste menu item or by typing the keyboard shortcut Ctrl+V. Cells copied into the clipboard can also be copied into a separate application, such as Microsoft Excel or a text editor.

For an example, see [Is there a way to save a copy of what is displayed in a Summary window?](#).

Configure Grid Rows



The Configure Grid Rows button (Ctrl+R)  will open a window that customizes the Object Summary display. This allows the user to select which parameters to display. This provides a simplified display for Motion Console that omits many unneeded or pre-configured parameters. It also hides parameters that should not be changed by users less familiar with the system.

In order to click the Configure Grid Rows button, one of the Object Summary windows must first be selected. Clicking the button will display a new window with all the parameters for the Object. Each parameter has a Check Box in the Hidden column. To hide a Row, simply select the appropriate Check Box.

In the example below, the Axis Summary Row Configuration window is open. Note that the Relative Distance, AccelJerk and DecelJerk rows have been checked as Hidden. This causes the Axis Summary window to no longer displays these rows. To display these rows, simply un-check the appropriate box in the Row Configuration window.

Axis 0	
Position 1	10000
Position 2	0
Relative Distance	0
Velocity	10000
Acceleration	100000
Deceleration	100000
Jerk Percent	0
AccelJerk	0
DecelJerk	0

Status	
State	Idle
Actual Position	0
Command Position	0
Position Error	0
Velocity	0
Acceleration	0
In Coarse Pos.	No
In Fine Pos.	Yes
At Target	No

BEFORE

Motion	Config	Hidden
Position 1		<input type="checkbox"/>
Position 2		<input type="checkbox"/>
Relative Distance		<input checked="" type="checkbox"/>
Velocity		<input type="checkbox"/>
Acceleration		<input type="checkbox"/>
Deceleration		<input type="checkbox"/>
Jerk Percent		<input type="checkbox"/>
AccelJerk		<input checked="" type="checkbox"/>
DecelJerk		<input checked="" type="checkbox"/>

Status	Hidden
State	<input type="checkbox"/>
Actual Position	<input type="checkbox"/>
Command Position	<input type="checkbox"/>
Position Error	<input type="checkbox"/>
Velocity	<input type="checkbox"/>
Acceleration	<input type="checkbox"/>
In Coarse Pos.	<input type="checkbox"/>
In Fine Pos.	<input type="checkbox"/>
At Target	<input type="checkbox"/>

Select which parameters
are to be hidden.

Axis 0	
Position 1	10000
Position 2	0
Velocity	10000
Acceleration	100000
Deceleration	100000
Jerk Percent	0

Status	
State	Idle
Actual Position	0
Command Position	0
Position Error	0
Velocity	0
Acceleration	0
In Coarse Pos.	No
In Fine Pos.	Yes
At Target	No

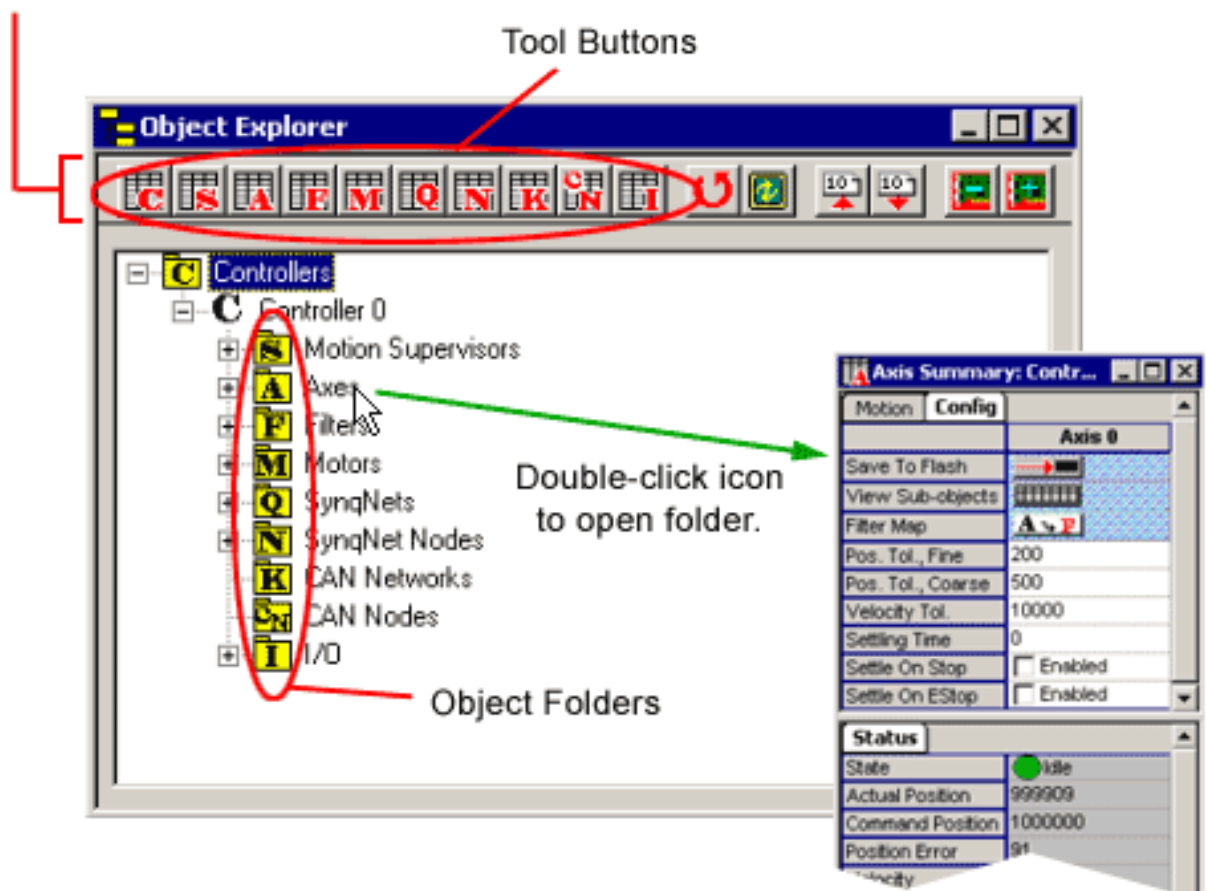
AFTER

Notice that the *Relative Distance*, *AccelJerk*, and *DecelJerk* parameters are now hidden.

Object Explorer

The Object Explorer provides the topmost, "bird's eye" view of the XMP motion control system. Objects such as Motion Supervisors and Motors are grouped together in their own separate folders. When opened, folders reveal objects available for mapping.

Object Summary Configuration Tools



In this example, the Filter icon is double-clicked to reveal currently available Filter objects (or you can click once on the file tree's "+" junction).

Tool Buttons – Clicking on a tool button causes an action to be performed in relationship to the object or folder currently selected in the object explorer. Another way to act on a selection is to right-click on the selected icon. Actions that can be performed on a specific type of object are unique to that object. Therefore, the user will notice that the tool buttons in the toolbar will change when a different object is selected.

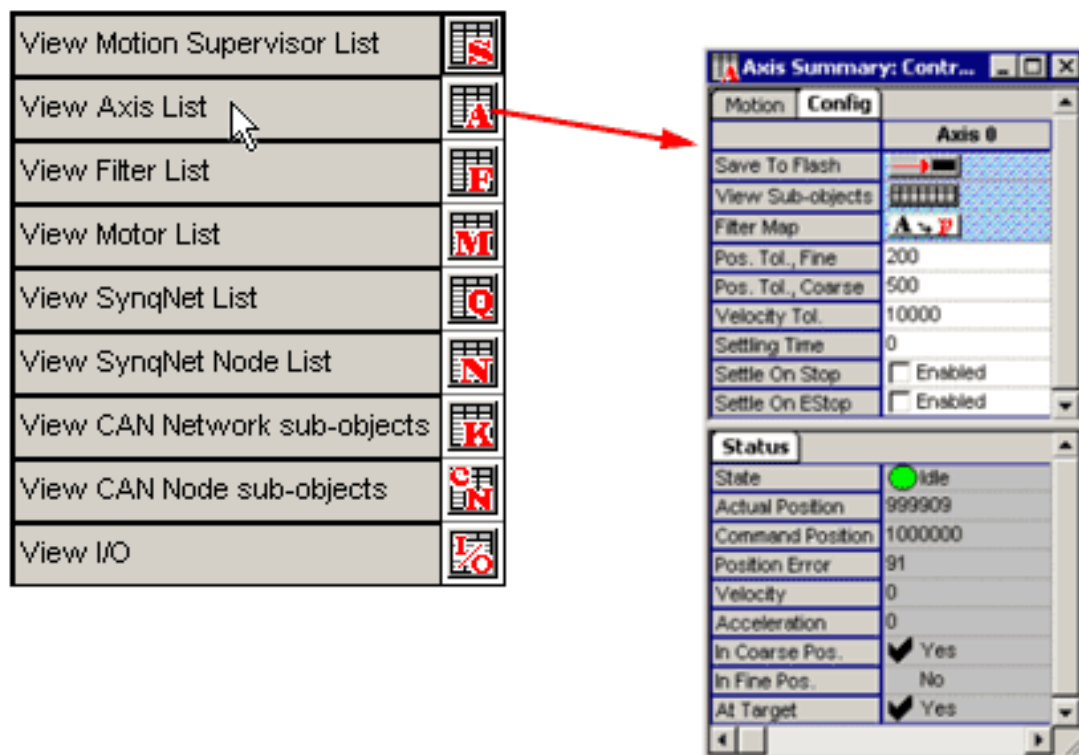
The Tool Buttons can be roughly divided up into two groups:

- buttons that open and configure object Summaries
- buttons that perform an action on the selected object

Summary of Configuration Buttons

When you click on a summary button, it opens and configures the selected object's summary window (see below).

For example, suppose Axis 1 is selected on the Object Explorer. Clicking on the Filter Summary button will configure the Filter Summary window to display the Filter objects that are mapped to Axis 1. Clicking on the Axis Summary button will program the Axis Summary window to display Axis 1.



Summary buttons are lettered with the first letter of the object they represent. In the case of Motion Supervisor and Motor objects, the Motor

is represented by an M, while the Motion Supervisor is represented with an S.

Within the View menu, summaries can also be displayed by typing their hot keys ("c" for Controller, "a" for Axis, etc.).

The Other Tool Buttons include:



- Reset the selected controller(s)



- Refresh display of the selected controller(s) and sub-objects



- Upload the firmware of the selected controller(s) to a file



- Download firmware from a file to the selected controller(s)



- Remove the selected controller(s)



- Add a controller

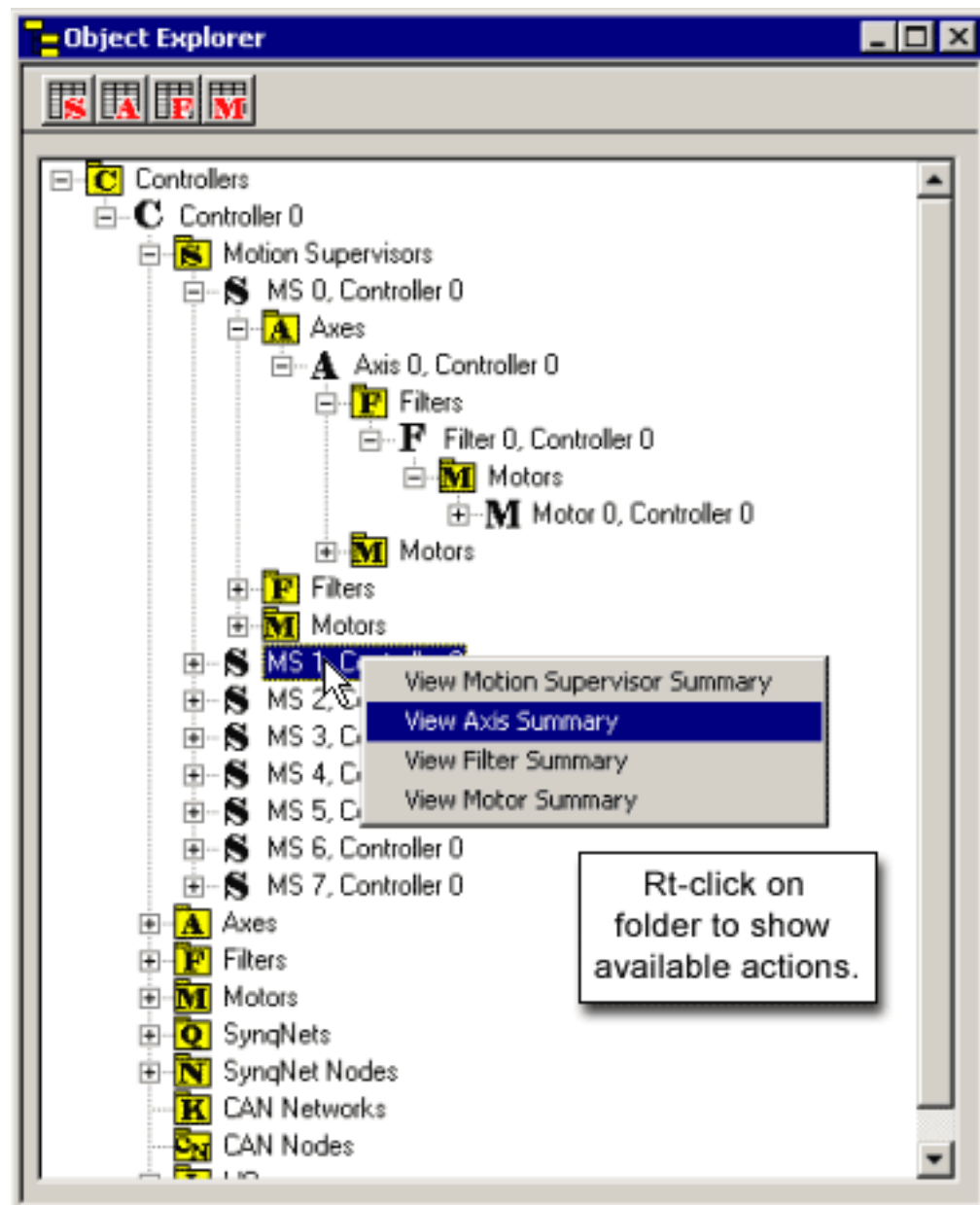


- Disassociate (un-map) the selected sub-object from the object containing it.

Object Explorer: Mouse Controls

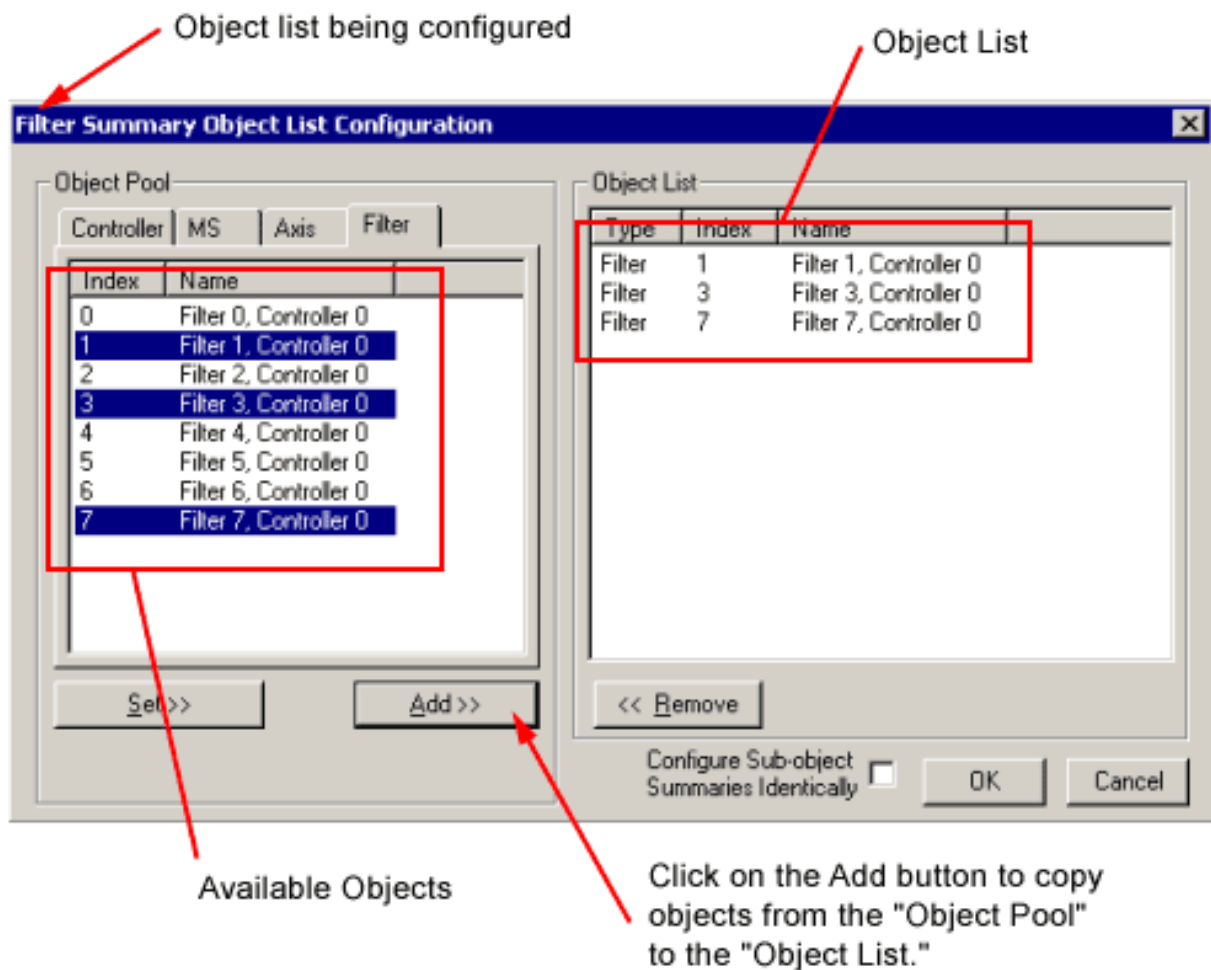
Shift-click – When holding down the Shift key and clicking on a Summary window icon, the selected object is either: 1) added to the Summary window (if it is not already there); or 2) removed from the Summary window (if it is already being displayed).

Right mouse button – The functions performed by the toolbar buttons can also be accessed by clicking on the selected object or folder with the right mouse button. This provides a drop-down menu listing of all actions that can be applied to the selected item.



Each item on the pull-down menu corresponds to an icon on the tool bar.

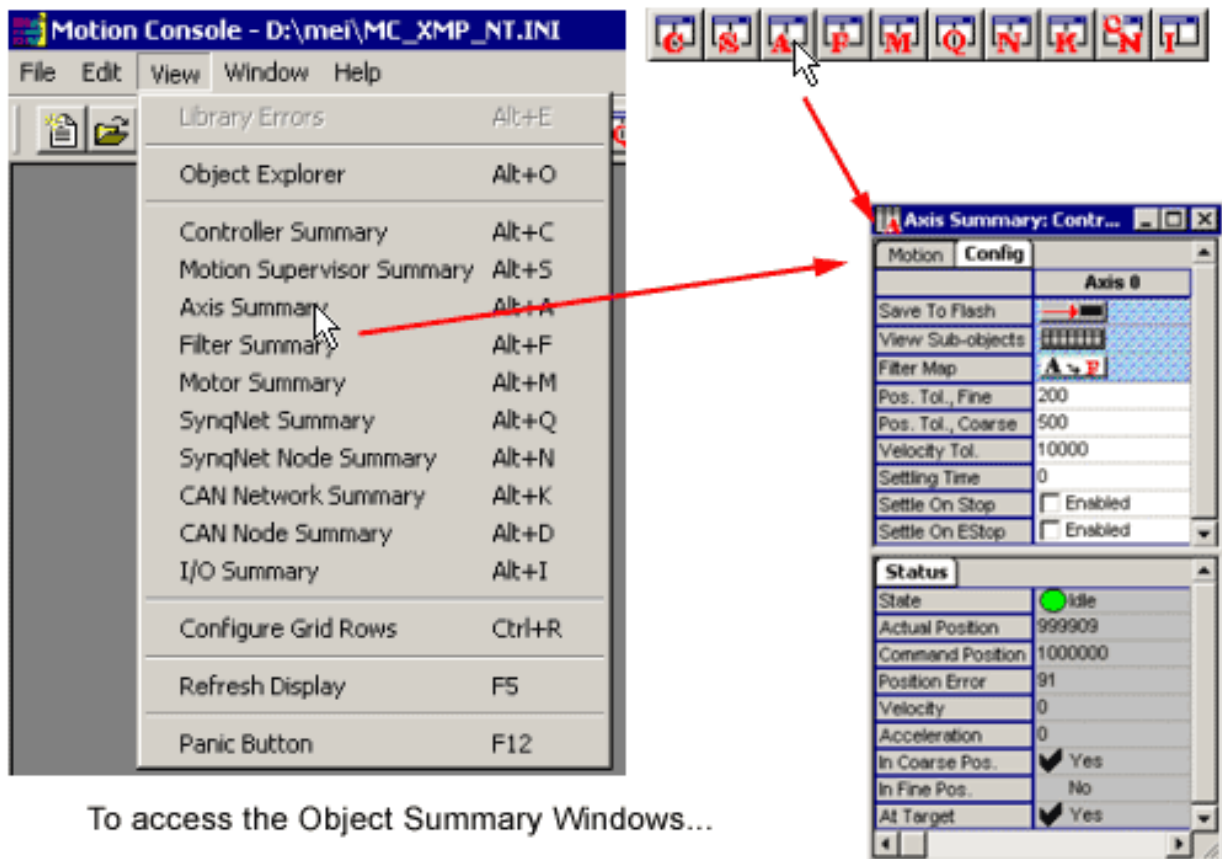
Object List Configuration Dialog Boxes



When using Motion Console, it is sometimes necessary to define a list of objects. The Object List Configuration Dialog Box is used for this purpose. To select more than one, hold Ctrl + Left Click. Object lists are used for three different purposes:

To display all objects mapped to a list of objects

An Object Summary can be programmed to display all the objects mapped directly or indirectly to a list of objects. For this purpose, the Object List Configuration Dialog Box is opened by clicking on one of the Open and Configure Object Summary buttons on the main frame toolbar, or by selecting one of the View/Object Summary menu items.



To map sub-objects to a super-object

A super-object (Motion Supervisor 1, for example) can be mapped to a list of sub-objects (Axis 0-3, for example). The Object Explorer can be used to map sub-objects one at a time to a super-object, but the Object List Configuration Dialog Box is used to define the entire list of sub-objects all at once. For this purpose, the Object List Configuration Dialog Box is opened by clicking on the sub-object map button on the Config tab page of the Object Summary. These buttons are listed below for each object that has sub-objects:

 **Motion Supervisor: Axis Map**

 **Axis: Filter Map**

 **Filter: Motor Map**

Save objects to flash memory

A list of objects can be defined to save to flash memory. There is a Save To Flash button on the Config tab page of each Object Summary. Clicking on this button will open the Object List Configuration Dialog Box, allowing multiple objects to be selected and saved to flash memory.

The title bar of the Object List Configuration Dialog Box tells the user what function is currently being performed. It may be one of the following:

- **[Object] Summary Object List Configuration:** For example: MS Summary Object List Configuration. In this case, the Object List Configuration Dialog Box is being used to configure the MS Summary to display a set of Motion Supervisors.
- **[Object] [Sub-Object] List Configuration:** [Object Name]: For Example: "MS Axis List Configuration: MS 0, Controller 0". In this case, the Object List Configuration Dialog Box is being used to map a set of axes to MS 0, Controller 0.
- **Save To Flash Memory Object List Configuration:** The Object List Configuration Dialog Box is being used to define a list of objects to save to flash memory.

Object List

The Object List displays the list of objects that will be the result of clicking the OK button. In most cases, the order of objects in the Object List is insignificant. The only exception to this is when configuring the Motion Supervisor Axis List. When the order of objects in the Object List is significant, then the order of the list can be modified using the Up and Down buttons, or by using the mouse to drag selected objects to a new position. (The Up and Down buttons are hidden if the order of objects in the Object List is insignificant.)

Object Pool

The Object Pool displays all objects that are valid candidates for the Object List. It consists of a set of tab pages, one for each type of object that can be in the Object List. The set of tab pages will vary, depending on the function of the

object list that is being defined.

Summary Object List Configuration

For this purpose, the Object Pool will include all objects hierarchically greater than or equal to the object type that is being displayed. For example, when configuring the Motor Summary Object List, the Object Pool will contain all Motors, Filters, Axes, Motion Supervisors, and Controllers. This is because the Motor Summary window can be configured to display motor objects directly, or it can be configured to display all motors that are associated with a set of super-objects.

Sub-object List Configuration

For this purpose, the only objects displayed in the Object Pool will be those of the same type as the sub-object list that is being configured. For example, when the Axis List of a Motion Supervisor is being configured, then only axes will appear in the Object Pool.

Save to flash memory object list configuration

Every object will appear in Object Pool.

Controller Summary

1 Tab:
Controller

Controller Summary Object List Configuration

Object Pool

Index	Name
0	Controller 0

Object List

Type	Index	Name
Controller	0	Controller 0

Set >> Add >> << Remove

Configure Sub-object Summaries Identically ☐ OK Cancel

Motor Summary

5 Tabs:
Controller
Motion Supervisor
Axis
Filter
Motor

Motor Summary Object List Configuration

Object Pool

Index	Name
0	Controller 0

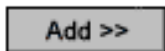
Object List

Type	Index	Name
------	-------	------

Set >> Add >> << Remove

Configure Sub-object Summaries Identically ☐ OK Cancel

Buttons



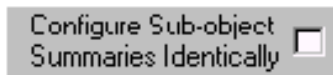
Add – Adds the highlighted object(s) in the Object Pool to the Object List.

NOTE: You may select more than one object in the Object Pool by holding down the Ctrl button while clicking on individual objects with the mouse. To select ALL objects between two items, hold down the <Shift> button while using the mouse to click on both items. Objects can also be selected without the mouse by using the navigation keys in conjunction with <Ctrl> and <Shift>. <Ctrl><Enter> will toggle selection of the current object.

The entire list can be selected by double-clicking on any member in the list. Selected items in either list can be dragged and dropped onto the other list. **NOTE:** Clicking the Add button is equivalent to dragging the selection in the Object Pool and dropping it into the Object List.



Set – Sets the Object List to those objects selected in the Object Pool. **NOTE:** Clicking on the Set button is equivalent to dragging the selection in the Object Pool and dropping it into the Object List while simultaneously holding down the Alt key.



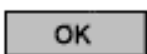
Configure Sub-object Summaries Identically – This check box is only visible when configuring an Object Summary and the object type for the Summary that is being configured has sub-objects associated with it. Checking the box will cause all the Object Summaries that display objects that are hierarchically lower than the object type of the Summary that is currently being configured, to be configured identically. For example, if the Motion Supervisor Summary is being configured to display MS 0, then the Axis, Filter, and Motor Summary windows will be configured to display all of their associated object types that are directly and indirectly mapped to MS 0. **NOTE:** The sub-object Summary windows must be open for them to be configured.



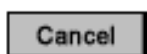
Up and Down – Clicking on the Up or Down button will move objects that are selected in the Object List up or down one position. The same result can be achieved by dragging the selected objects and dropping them into the desired position. These buttons are hidden when the order of objects in the Object List is insignificant.



Remove– Removes highlighted object(s) from the Object List. **NOTE:** Clicking on the Remove button is equivalent to dragging the selection in the Object List and dropping it into the Object Pool.



OK – Closes the dialog box and retains the Object List settings.



Cancel – Closes the dialog box without retaining the Object List settings.

Configuring New Systems with Motion Console

Introduction

This section describes how to configure a new motion control system using Motion Console. If you are new to Motion Console, you should read this section first, then proceed to the other sections in this manual.

Getting Started with Motion Console

Motion Console is a utility designed to assist software and hardware designers with MEI motion controllers. There are two major releases of Motion Console: one for the XMP Series motion controllers, and another, older release for DSP Series motion controllers. This section discusses only the XMP version of Motion Console; the DSP release is outlined in separate documents available from MEI.

What is Motion Console?

Briefly stated, Motion Console is a software interface linking the programmer to your MEI motion controller. It gives you the ability to perform simple motions with hardware and verify that controller-hardware wiring is working properly. Motion Console is NOT a software authoring tool. It does not compile programs or check code. However, using Motion Console alongside a line editor and compiler will allow you to quickly troubleshoot programs and determine whether problems lie in hardware or software. Motion Console also provides a quick way to demonstrate the full range of mechanical movement in your system, so that you may perfect hardware designs while software is still being developed.

What is Motion Scope?

Another MEI utility, separate from Motion Console, is Motion Scope. Motion Scope provides a virtual "oscilloscope" to plot the movements of your motion system. Using Motion Scope in conjunction with Motion Console allows you to command and plot the movement of any axis. See the [Motion Scope](#) section.

First Things First: Safety

Before configuring your system, you must consider the safety aspects of your motion control system. Such configurations as "amplifier enabling" impact every movement that your system makes, and dramatically redefine the degree of safety. Specifically, the following safety issues must be addressed when configuring new systems:

- What hazards are posed by your motion control system to personnel and material? Has a safety zone been established to exclude personnel, their hands and fingers, etc. from dangerous spaces? Are warnings posted to alert personnel when and where hardware is in motion?
- Are safety interlocks in place to automatically protect personnel from exposed high voltage and mechanical movements? Are emergency switches in place to perform instant, electro-mechanical shut-downs when needed?
- How will your drive amplifiers be wired? Are they wired in a fail-safe mode which leaves your system safe when power is suddenly or unexpectedly lost? The Amp Enable box is located under the Motor Object Summary window -> Config tab.

Designers of new systems are sometimes tempted to rush and ignore basic safety issues; however, safety is integral to your system's usability. If you design your system with safety in mind from the beginning, you will alleviate major hardware and software design problems later, saving valuable lives and time.

CAUTION! Do *NOT* Connect Your Hardware Yet!



Before connecting your MEI controller to the hardware, you must first follow the steps outlined in the next section *Safety Parameters*. This will ensure a proper level of safety.

Saving Parameter Settings

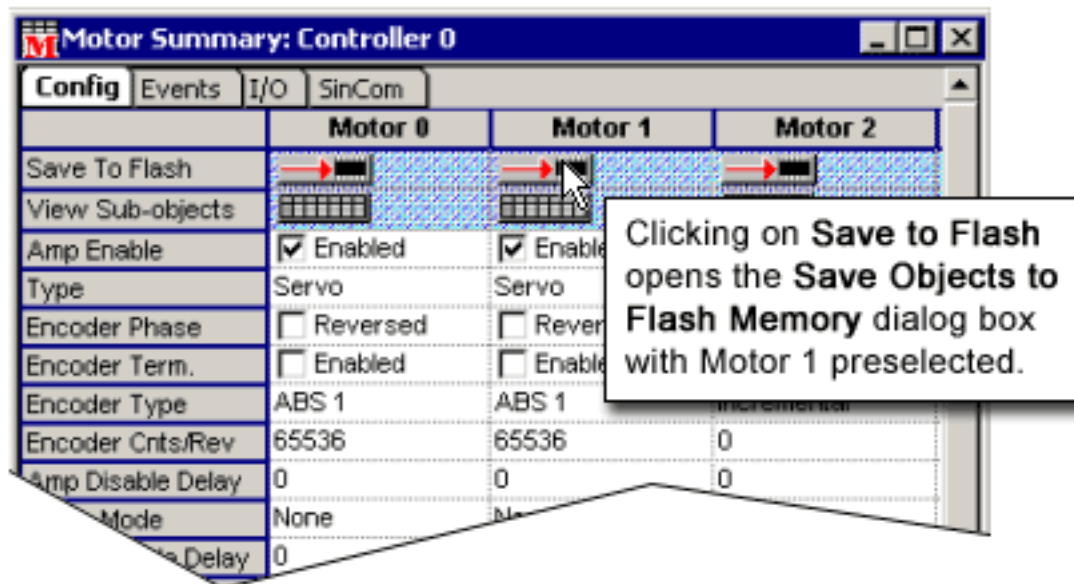
Before proceeding, you must learn to save your object settings and mappings. If you do not save your configurations, they will be lost the next time your controller is reset and/or powered down. Saving in Motion Console is NOT like saving files in other applications. (Remember, you are dealing with a lower-level hardware controller which interacts directly with hardware.) Motion Console saves parameter settings directly to its own flash memory, which ensures the same settings are reused when you repower your controller. You may also upload settings from flash memory to a firmware file, located on a hard disk or separate storage media. This permits the saving of more than one set of parameter settings, and allows multiple controllers to be loaded with the same parameter values. To ensure that your settings are protected, it is recommended that you both Save to Flash and save to an archived firmware file (Firmware Upload). Both functions are explained below.



Saving to Flash Memory

Each object has its own Save to Flash [Memory] button, which opens the Save Objects to Flash Memory dialog box. This dialog box configures a list of objects to be saved to flash memory. When the controller is powered down and restarted, the last-flashed settings are loaded automatically. For example, clicking on the Save to Flash button under Motor 2 will open the dialog box, with Motor 2 shown in the object list.

Each object has its own Save to Flash [Memory] button, which opens the Save Objects to Flash Memory dialog box. This dialog box configures a list of objects to be saved to flash memory. When the controller is powered down and restarted, the last-flashed settings are loaded automatically. For example, clicking on the Save to Flash button under Motor 2 will open the dialog box, with Motor 2 shown in the object list.

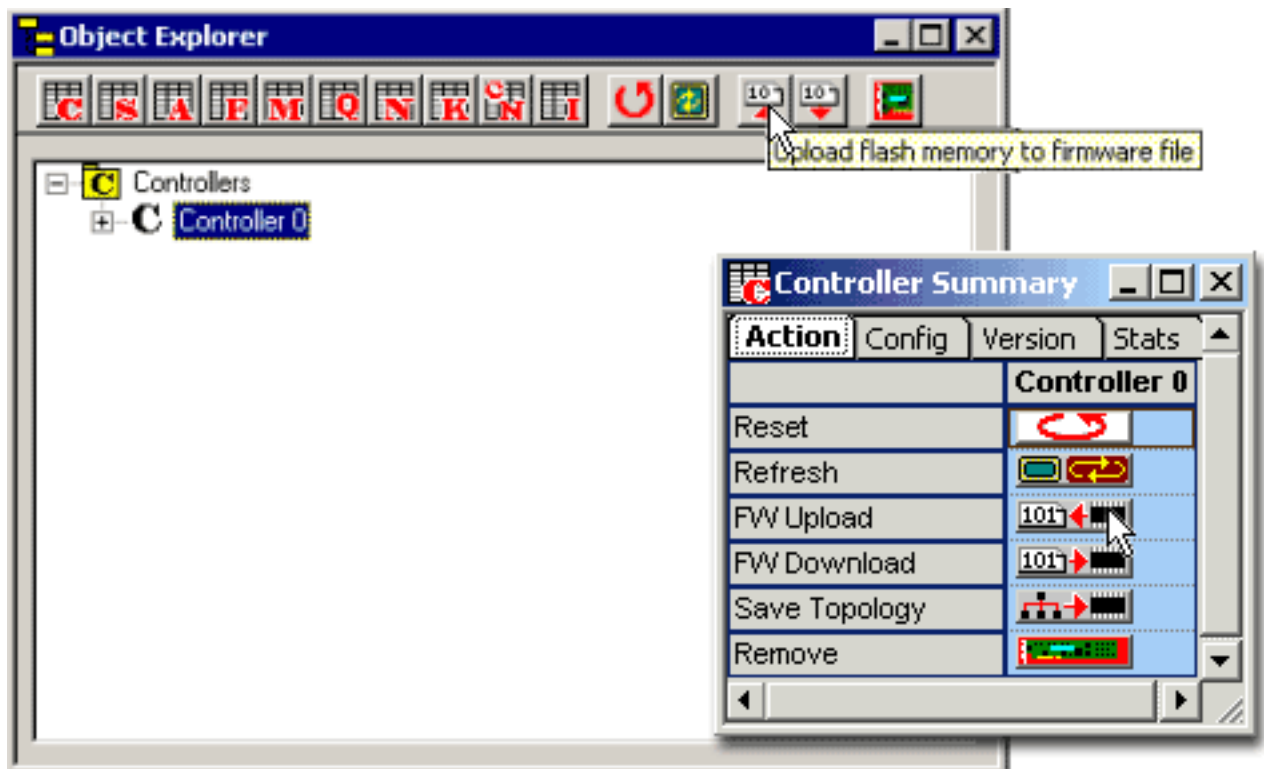


Uploading Firmware

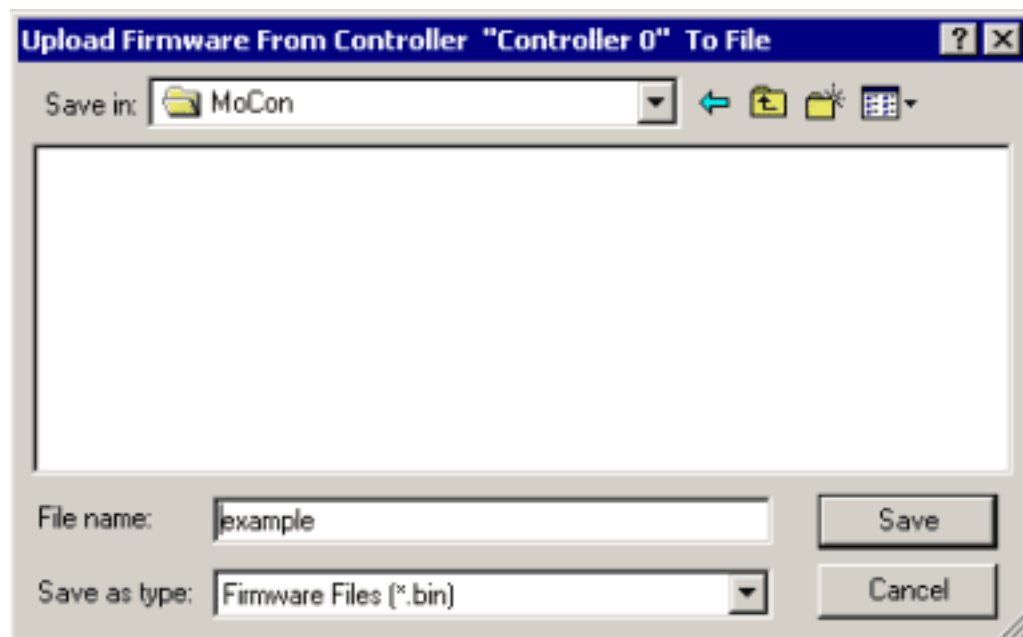
Firmware settings which are flashed to the controller's memory can also be saved as a file on separate media using the FW [Firmware] Upload function. It is highly recommended that settings be uploaded to an archive for secure storage. This ensures protection of your settings in case the controller is damaged or lost, and allows storage of multiple configurations.

IMPORTANT! The FW Upload function saves only what is stored in flash memory, not settings on the Motion Console screen. Therefore, you must first use the Save to Flash function (see above) before using the FW Upload function.

The FW Upload function can be accessed either from the Object Explorer window, or the Controller Summary window.



Clicking on the FW Upload button displays a file manager window:

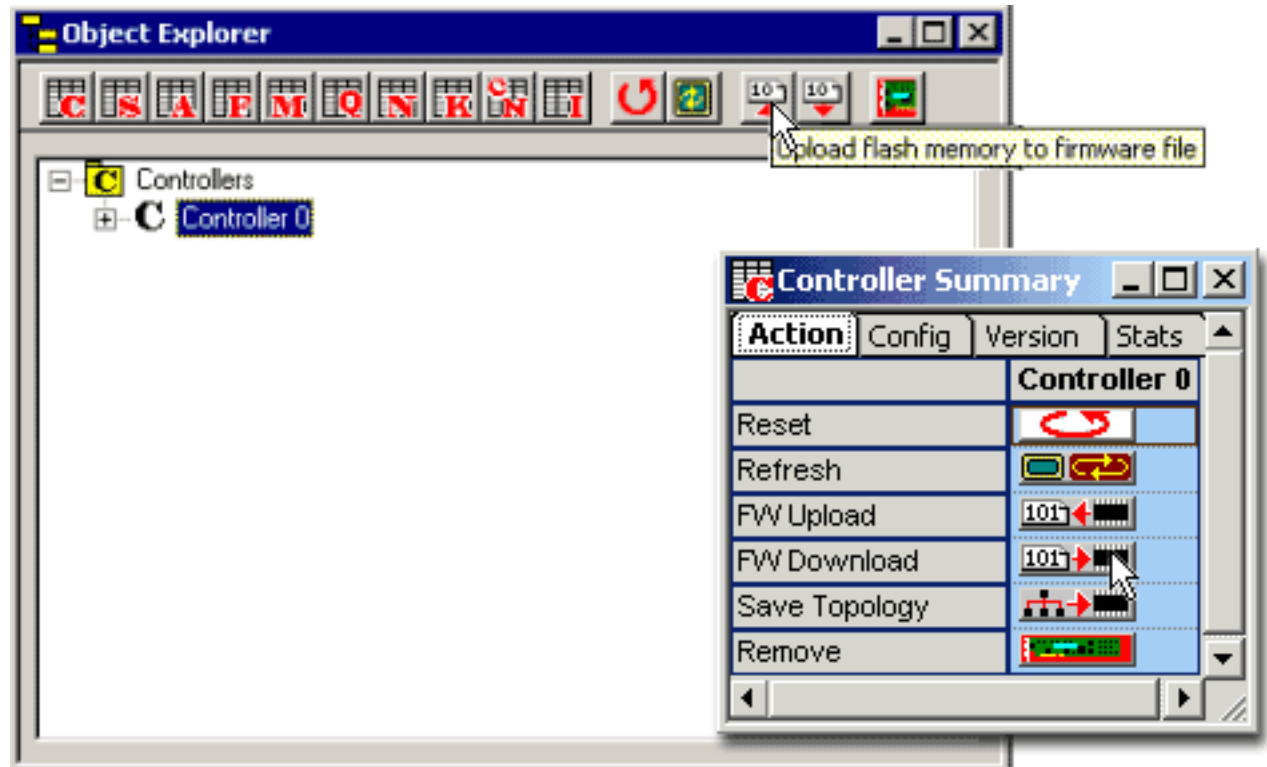


Enter the path and filename of the firmware file to be saved, then click on the Save button. The controller's firmware in flash memory will be saved to the file indicated.

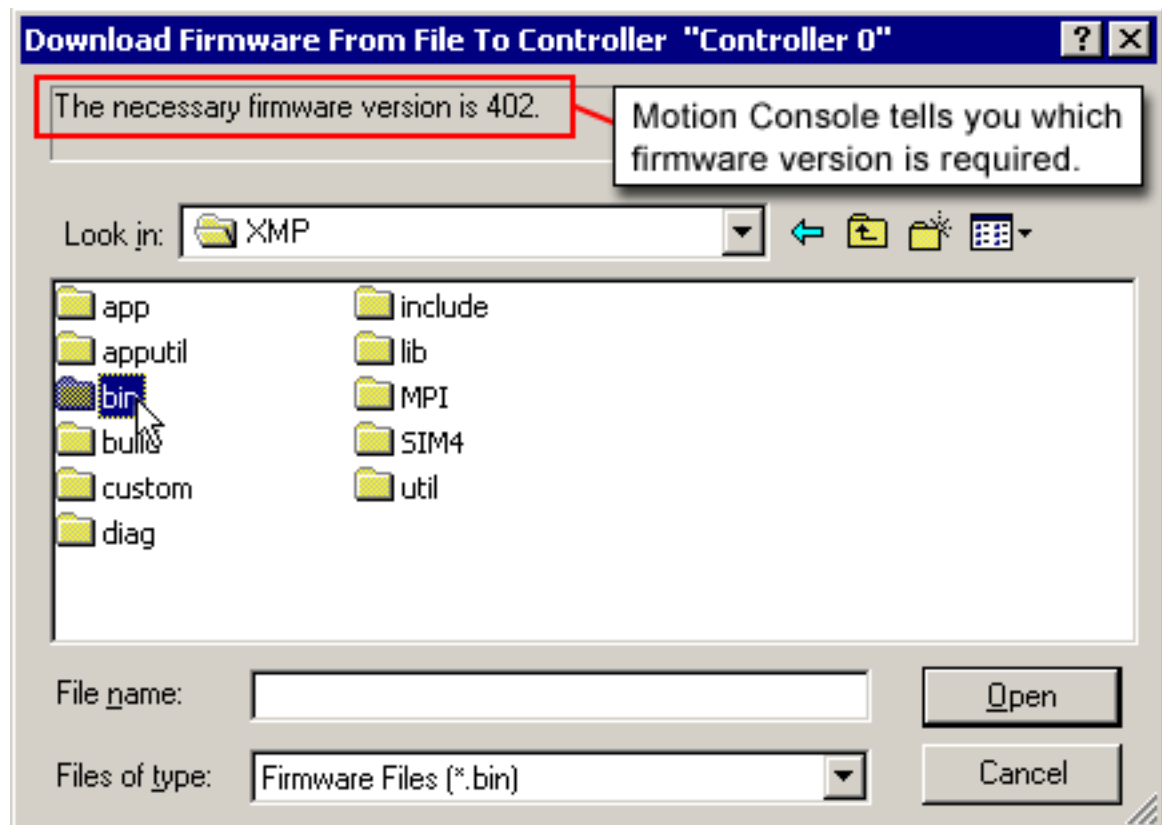


Downloading Firmware

The opposite of uploading is downloading. Once a firmware file has been saved, the FW Download function copies the saved firmware file into the controller's flash memory, and volatile memory as well. To download a firmware file into the controller's flash memory, click on the FW Download function in the Object Explorer or Controller Summary windows.



Clicking on the FW Download button displays a file manager window:



Select the firmware file (*.bin) to be loaded and click on the Open button. The selected file will be loaded into the controller's flash memory.

Saving to Flash Table

It is important to know what is saved and what is not saved when you save an object in Motion Console. The table below summarizes what is actually saved when you use the Save to Flash feature.

NOTE: The mappings of objects to sub-objects are saved (ex: Motion Supervisor axes mappings), except in the case of SynqNet-to-Nodes-to-(I/O and Motors) and CAN-to-nodes-to-I/O.

Object	Saved	Not Saved	Notes
Controller	Config tab, Recorders tab, Axis Map	Action tab, Version tab, Stats tab	
Motion Supervisor	Stop Time, E-Stop Time, Normal Feedrate, Axis Map	Everything Else	
Axis	Config tab,	Motion tab, Status tab	
Filter	Config tab, Coeff tab		Post filters are saved, but are not shown in Motion Console.
Motor	Config tab (except Amp Enable), Events tab, SinCom tab, Upper I/O tab	Amp Enable, Home Trigger, Encoder Fault Trig., Status tab, (Lower) I/O tab	
SynqNet		Config tab, Info tab, Status tab, Axis Map (Save with SynqNet topology)	
SynqNet Node	Config tab (except Upstream Err. Fault/Fail Limits)	Upstream Err. Fault Limit, Upstream Err. Fail Limit, Info tab	

CAN	Config tab		
CAN Node	Config tab		
I/O	Motor I/O tab (Type only, not Out or In)	SqNode I/O tab, Motor I/O tab - Out and In, CAN I/O tab	

Associating Objects with Motion Console

Motion Console's Object Explorer can be used to associate objects comprising your motion control system. Objects are associated (mapped) in specific ways, for example:

- Motors are mapped to Filters
- Filters are mapped to Axes
- Axes are mapped to Motion Supervisor

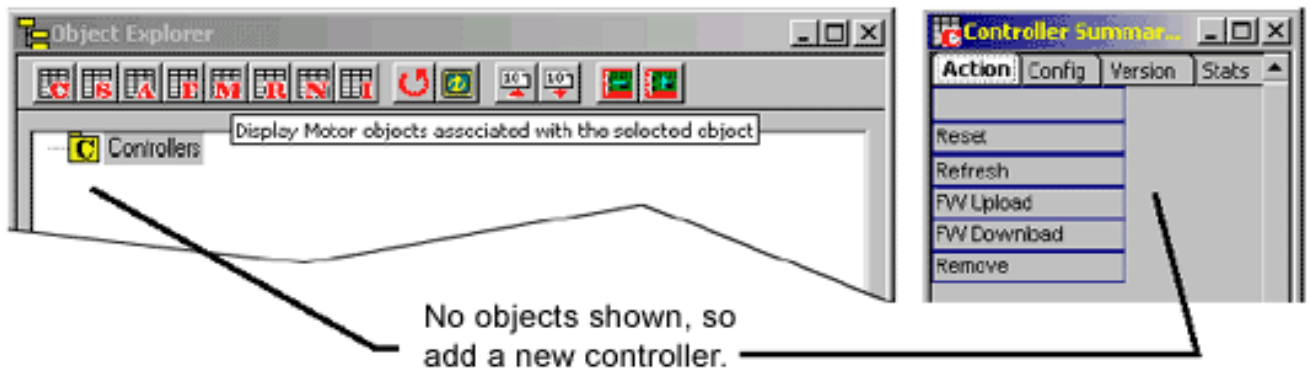
How Many Controllers?

In Motion Console, each "Controller" designates a physical piece of hardware: one XMP controller, with or without an expansion card. Depending upon how your XMP controller is configured, you will be able to command 1-32 axes per controller. If you require more axes, you will need to install additional XMP controllers.

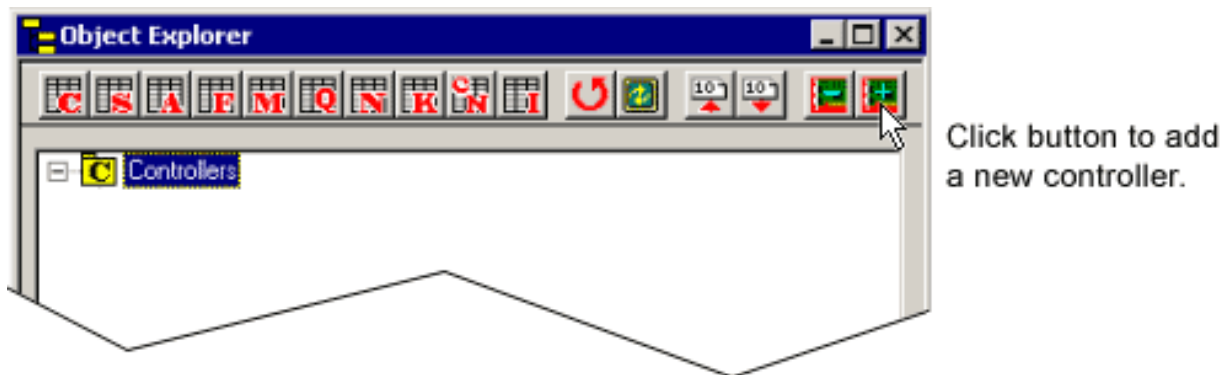
Creating a New Controller

Object Before any mapping can be done, you must first add a Controller. To determine whether a Controller object already exists, look at the Object Explorer. The Object Explorer may appear as shown below if a Controller has not yet been added. (No objects will be listed below the Controllers icon in Object Explorer, or listed within the Controller Summary.)

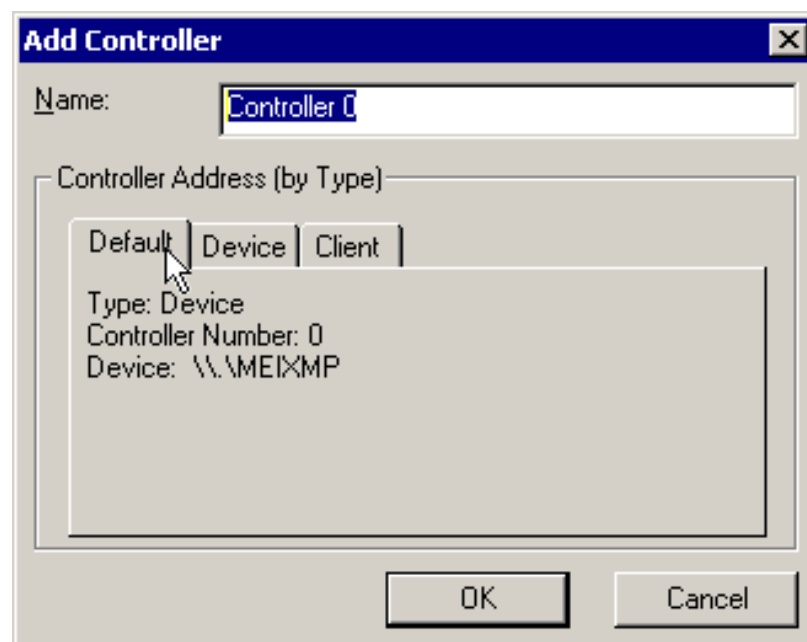
An unconfigured Motion Console.



To add a Controller to Motion Console, click on the Add Controller icon on the Object Explorer toolbar or the main toolbar.



Clicking on the Add Controller icon displays the Add Controller dialog box. In most cases, the default settings may be left as-is. If you desire to make changes to settings, refer to the "Controller Object" section of this chapter for a description of attributes.



Click on the OK button to add a new Controller. The new Controller will be added to the object tree of the Object Explorer, which now appears as shown here:

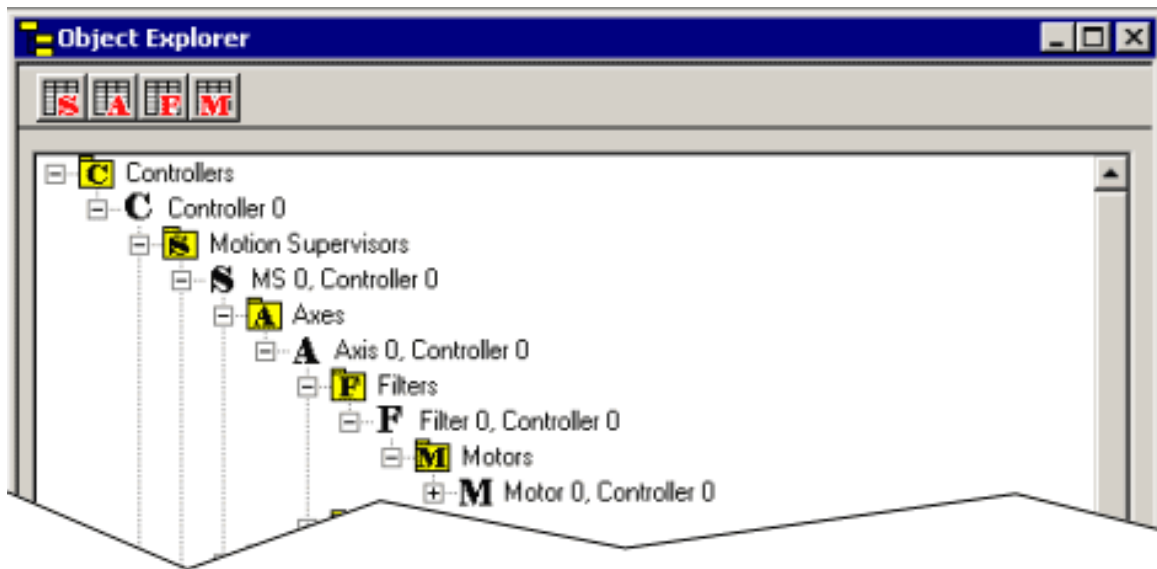


Whenever a Controller is added to the Object Explorer, it will have a number of sub-objects. These objects are explained below.

Mapping a Motion Supervisor

The Motion Supervisor is a high-level, motion "task master," which may be configured to manage any number of axes. In the majority of cases, however, it is advantageous to assign to each Motion Supervisor one Axis. This keeps the architecture simple, and offers maximum flexibility. In limited cases (e.g., gantry cranes), it may prove helpful to map additional axes to a single Motion Supervisor.

The mapping displayed below shows one configuration. In this case, Axis 0 has been mapped to MS 0. Motion Supervisors 1-7 (not shown below) are also mapped to the same Controller object, giving a total of eight (8) Motion Supervisors.

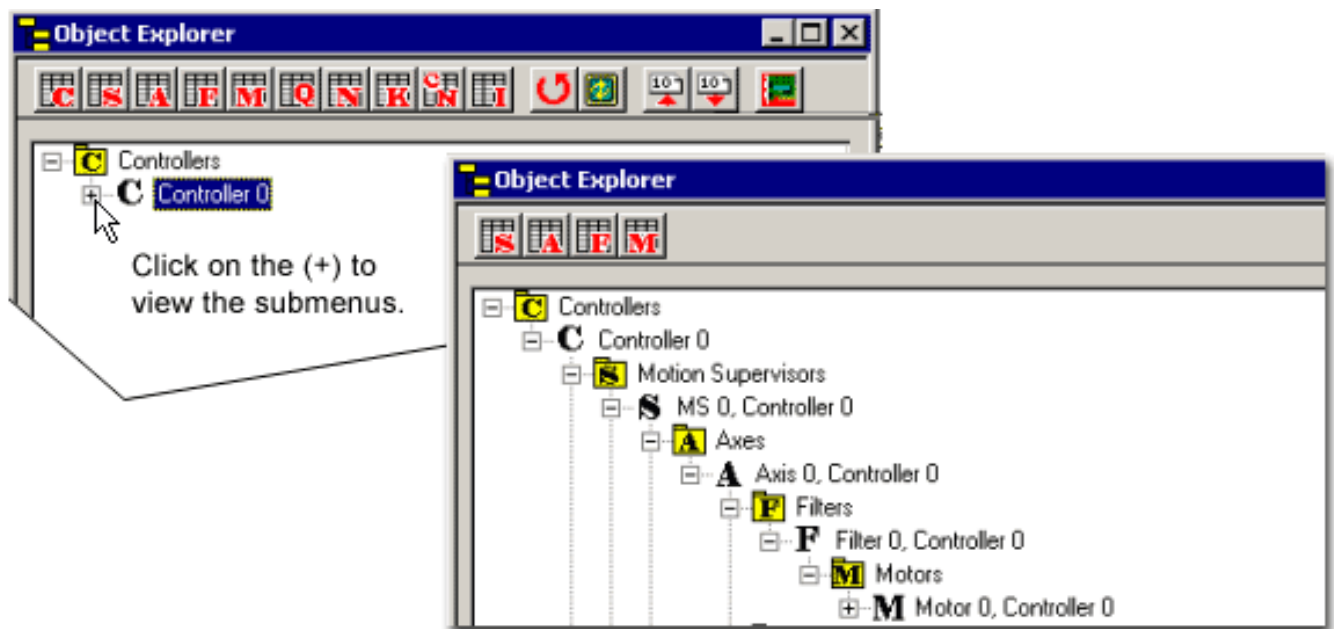


As shown here, Motion Supervisor 0 has the following sub-objects mapped to it: Axis 0; Filter 0; and Motor 0. Other mappings and combinations are possible too, subject to the user's control.

Configuring a Motion Supervisor

Because Motion Supervisors comprise the highest level of object organization (after the controller itself), they provide a convenient starting point from which to map. Mapping will require some forethought. Some of these objects are linked tightly together (e.g., Axes, Motors and Filters); therefore, it is useful to think of them as a single unit, operating under the direction of a given Motion Supervisor. Other objects may operate independently of others.

The Object Explorer gives the best single overview of your motion control system. To look at what you have configured, open the object tree below Motion Supervisors by clicking on the file "branch" at this point. It will look something like the object tree shown here:



Mapping Axes

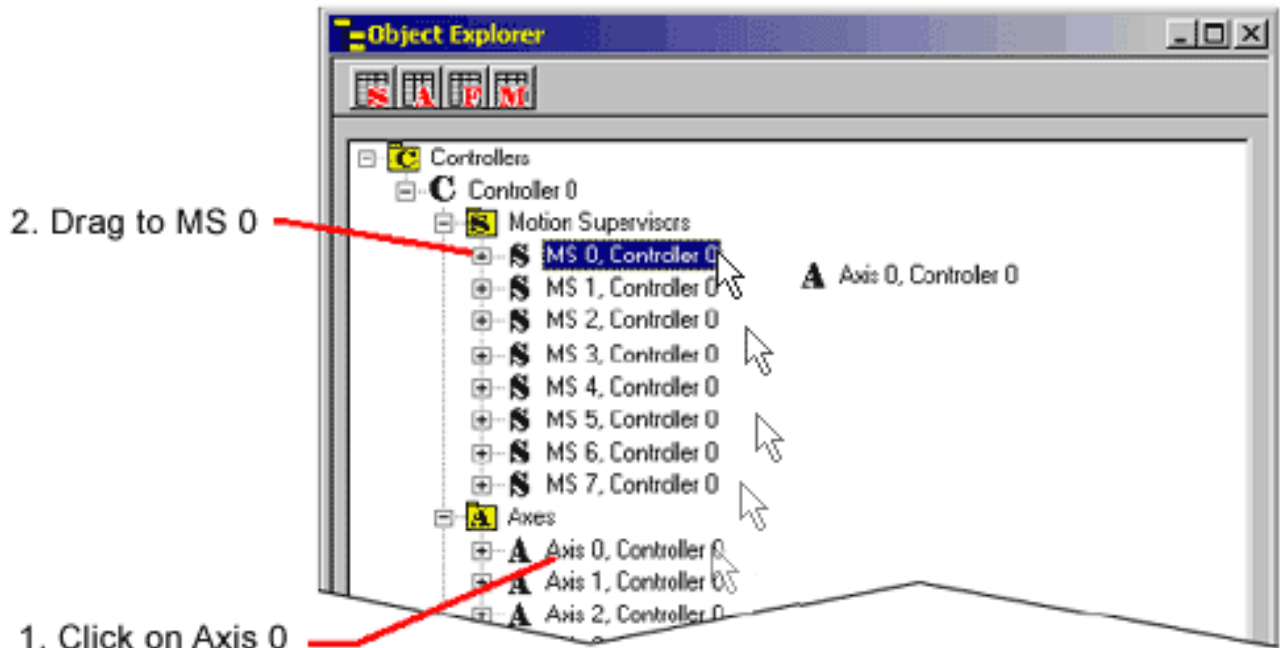
Classically, a single motion axis is powered by a single motor. The XMP's architecture accommodates this traditional arrangement, but also allows more than one motor to be mapped to a single axis. (Through the use of Filters, multiple motors can be configured to operate differently upon the same axis.) For now, do not worry about the type of motor(s) to be used. Think only of the axes of motion. Before further mapping, you should know the following about your motion control system:

- How many axes of motion are required?
- Do some axes have special requirements or limitations?
- Will some motors always operate with other motors? How?

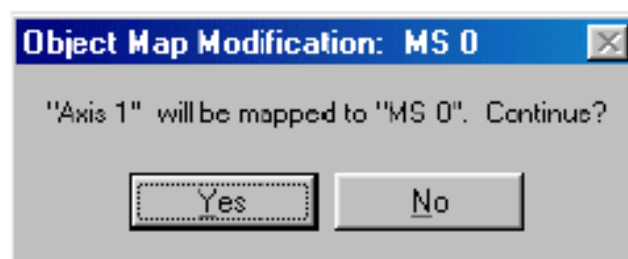
After these questions have been answered, you may begin mapping Axis objects to Motion Supervisors. In the examples which follow, it is assumed that objects will be mapped in numerical order (e.g., we will map Motor 1 before we map Motor 2). This is the usual order of progress, although you may do things differently if you prefer.

Mapping One Object to Another

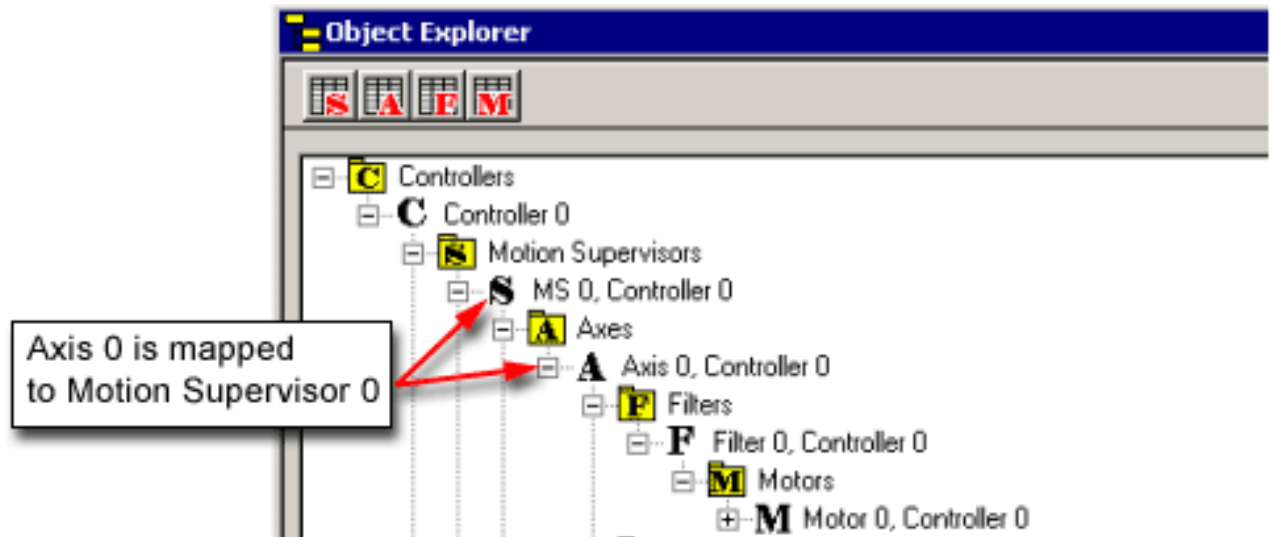
To map Axis 1 to MS 0, use the cursor to select the Axis 1 object, then drag it into MS 0.



A dialog box will be displayed to confirm the object mapping:



To confirm the object mapping, click on the Yes button. The Object Explorer will now show Axis 1 mapped to Motion Supervisor 0:



To map several objects to their "default" sub-object, i.e., the one with the same number, use the "Sub-Object Map" button on the Summary window. (For Motion Supervisors, this button is labeled "Axis Map.") If the entire row is selected and the button is clicked, then an Object List Configuration Dialog Box will be opened to map sub-objects to each object. If the key is held down when the button is clicked, then each object will be mapped to its default sub-object.

Mapping Filters, and Motors

Each Axis has at least one Motor and Filter object associated with it. Mapping of these objects is performed exactly as demonstrated above with Axis objects.

Before mapping these objects, it will help to answer the following questions:

- What types of motors are required by your system? Servo or stepper? If servo-type, what phase? What are the speed and travel constraints of each axis? How does this affect the torque capabilities of your selected motors? What duty cycle is expected of your motors?
- Is your system closed-loop or open-loop? If closed-loop, how many encoders? What levels of resolution and repeatability are required?
- Which motion path algorithm will best answer your needs (PID, PVT, PD, etc.)? How will you tune and adjust your system?

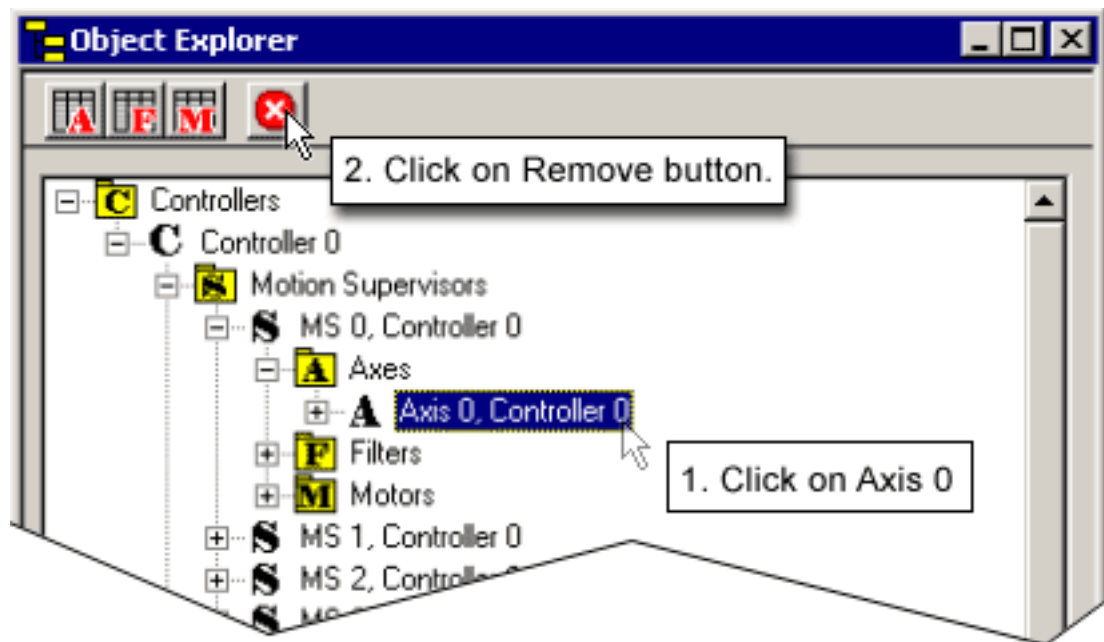
Based upon the mechanical requirements and overall architecture of your system, you should proceed to mapping objects with the simplest configuration possible. Motor objects correlate to specific pieces of hardware having connection and power constraints. These can be determined by referring to the [XMP Hardware section](#) and the manufacturer's specifications.

Troubleshooting

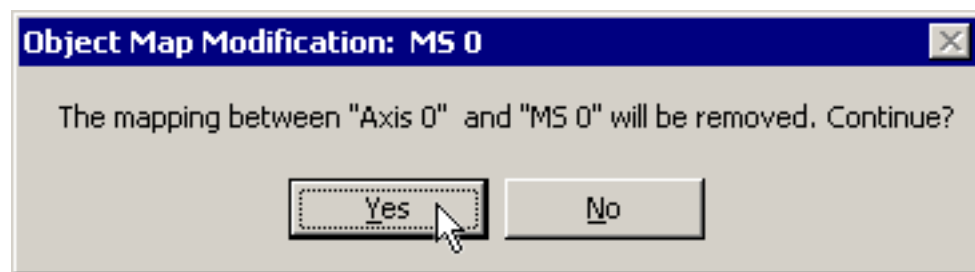
Are you experiencing unexpected behavior? It may be a result of improper axis mapping. See the [Troubleshooting](#) section.

Removing (Deleting) a Mapped Object

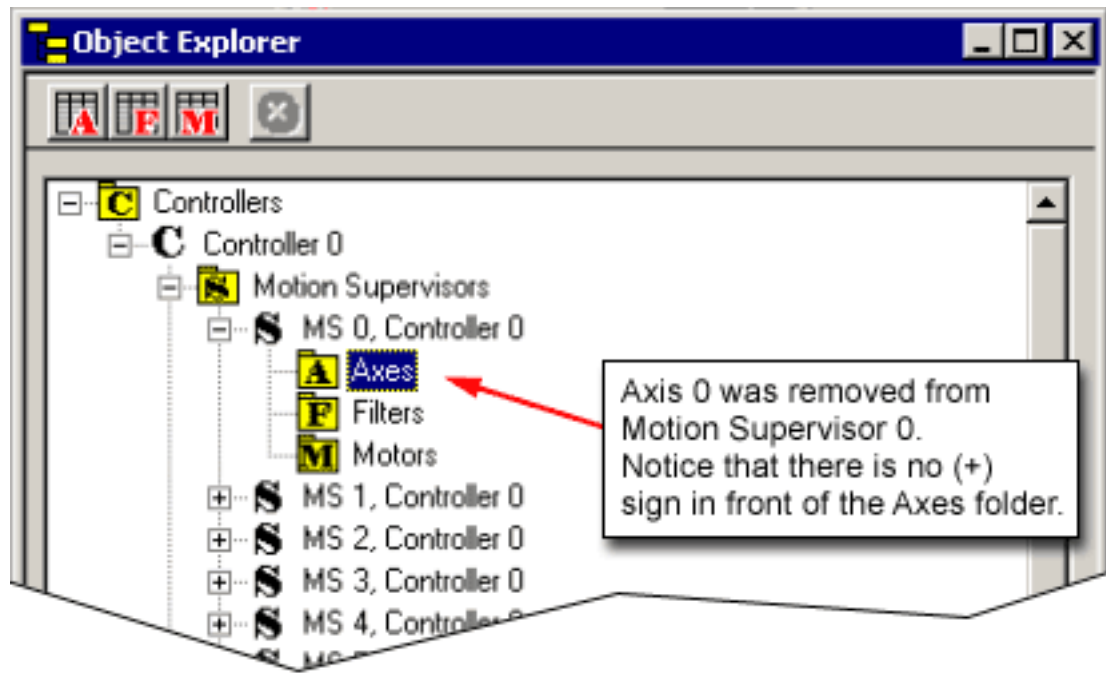
Recall that in the previous example, Axis 0 was mapped to Motion Supervisor 0. But what if you preferred to map Axis 1 to Motion Supervisor 0 (MS 0) instead? To do this, you must first remove Axis 0 from MS 0. Object removal is done by first selecting the object to be removed, then clicking on the Remove Axis From List icon (or use the <Delete> key).



A dialog box will be displayed to confirm the object removal:



To confirm the object removal, click on the Yes button. The Object Explorer will now appear as shown here:



Notice that Axis 0 is no longer mapped to MS 0. Moreover, all of the sub-objects associated with Axis 0 (e.g., Filter 0 and Motor 0) have also been removed from MS 0. This is because Filter 0, and Motor 0, were mapped to Axis 0. When we removed Axis 0, all its sub-objects were removed with it.

Note that Axis 0 has not been entirely removed from use. If you click on the Axes portion of the object tree, you will still find it there. But, it is no longer mapped to another object.

Safety Reminders

As you map out your motion control system using the Object Explorer, you must address specific safety factors. These should be addressed **BEFORE** you mechanically engage motors to axes, and connect motion hardware to the controller.

Establish a Safety Zone

If you have not already done so, define a safety zone around the motion system to be configured. Areas traversed by hardware should be clearly marked and/or partitioned to keep out personnel, hands, fingers, etc. Warning signs and labels should be clearly posted.

Mount an Emergency Off Switch

At least one emergency off (EMO) switch should be prominently mounted where operators and technicians can easily locate and reach it at all times. Depending upon your system, the EMO should effect a rapid and safe stop of all components, preferably wired through the power lines of the system components themselves. Consider using automatic mechanical braking where heavy and/or fast components might coast dangerously after the loss of a controller.

Wire all Drive Amplifiers for Fail Safe Operation

The wiring of your system's drive amplifiers is a critical design detail and is discussed in the XMP Hardware Installation Manual. As part of your design, you must decide whether to wire the drive's amplifier enable (Amp En) line as "active high" or "active low." Regardless of the method used, ensure that your motion system cannot run away when power to the controller is lost. Design for the worst possible power loss scenario, then ask yourself: "Is this machine safe with power to the controller turned off?" Amplifier enabling should be tested in crude hardware

fashion before the system is either mechanically coupled or connected to an XMP controller. Once hardware wiring has been tested for fail-safe design, proceed to the configuration of Motor Summary / General Configuration attributes (see below).

Configure Motor Summary / General Configuration Attributes

Attributes in the Motor Summary / Config tab page configure both motors and encoders. Of particular importance to safety are the following:

- Amp Enable
- Type
- Encoder Phase
- Encoder Term
- Encoder Type
- Encoder Cnts/Rev.
- Amp Disable Delay
- Brake Mode
- Brake Enable Delay
- Brake Disable Delay
- Fault Config
- DAC Offset
- Aux DAC Offset
- SIM4

The screenshot shows the 'Motor Summary: Controller 0' window with the 'Config' tab selected. The window displays a list of configuration attributes for Motor 0. The attributes are listed on the left, and their values are shown on the right. The 'Encoder Term.' attribute is highlighted with a blue background.

Motor 0	
Save To Flash	
View Sub-objects	
Amp Enable	<input type="checkbox"/> Enabled
Type	Servo
Encoder Phase	<input type="checkbox"/> Reversed
Encoder Term.	<input checked="" type="checkbox"/> Enabled
Encoder Type	Incremental
Encoder Cnts/Rev	0
Amp Disable Delay	0
Brake Mode	None
Brake Enable Delay	0
Brake Disable Delay	0
Fault Config	0x1f
DAC Offset	0
Aux DAC Offset	0
SIM4	<input type="checkbox"/> Enabled

Each of these attributes are described in a separate section of this chapter. The Amp Enable and Polarity attributes are treated in the XMP Hardware Installation Manual. Note especially that once a motor's Amp Enable line has been wired, the Amp Enable and Polarity attributes must

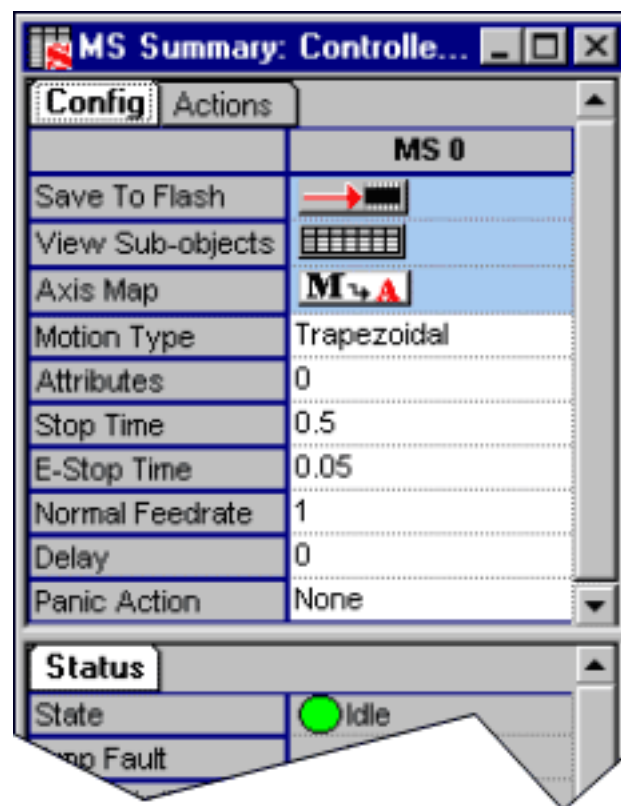
accord with system wiring to avoid problems.

Encoder Phase and Encoder Cnts/Rev affect the safety of your system by telling the controller which way an axis is moving, how far and how fast. If encoder attributes are set wrong, you may experience runaway problems.

Configure Motion Supervisor Summary / General Configuration Attributes

Attributes in the Motion Supervisor Summary / Config tab page include those used to configure a motion system's panic actions. These tell the controller what to do when an error or panic stop command is encountered. If they are not correctly configured, your controller may ignore an operator's command to stop! Attributes include the following:

- Stop Time
- E-Stop Time
- Normal Feedrate
- Delay
- Panic Action



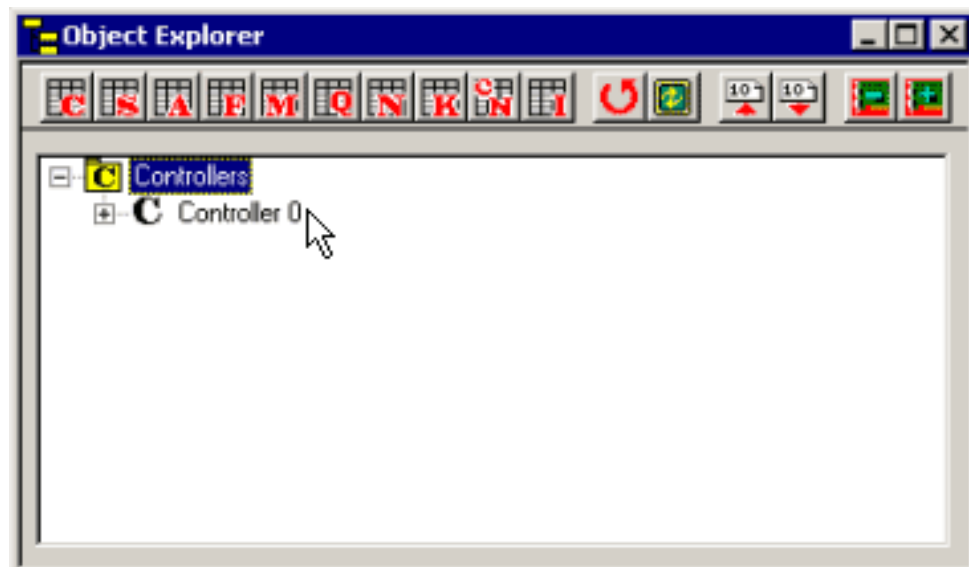
The Panic Action parameter is of particular importance. When switched

to None, the controller will take no action after panic events! Please note that the <F12> panic stop key generates a panic action for all Motion Supervisors, according to how Panic Action is configured.

Controller Objects

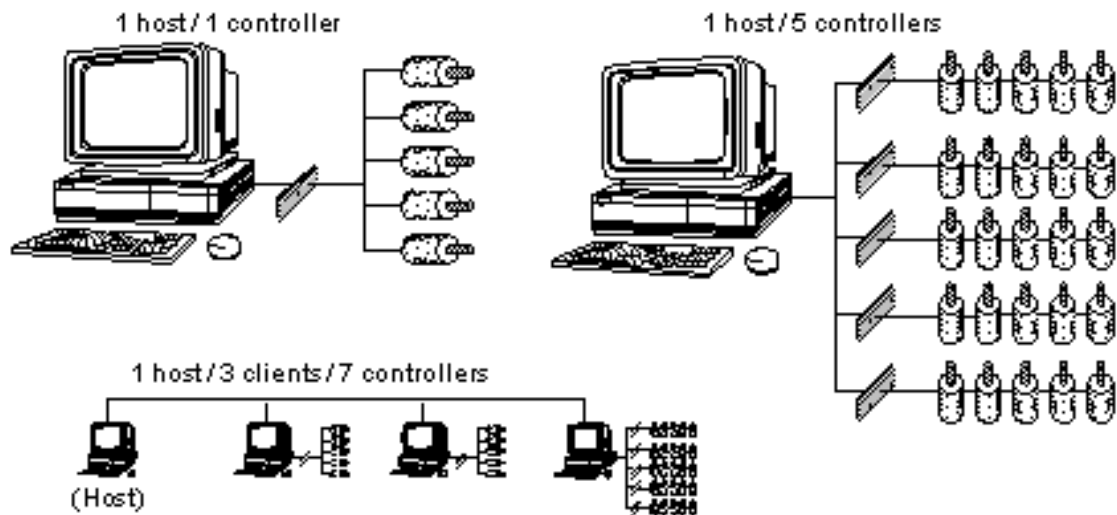
Within the Motion Console program, Controller objects occupy the highest level on the motion control tree. This prominent software position reflects the hardware reality: each controller represents a separate, physical piece of hardware (i.e., an XMP card) having local control. Moreover, each controller is capable of operating multiple motion supervisors, each with its own subordinate array of axes, motors, etc.

If you have not already added a Controller, please see the [Add Controller](#) section.

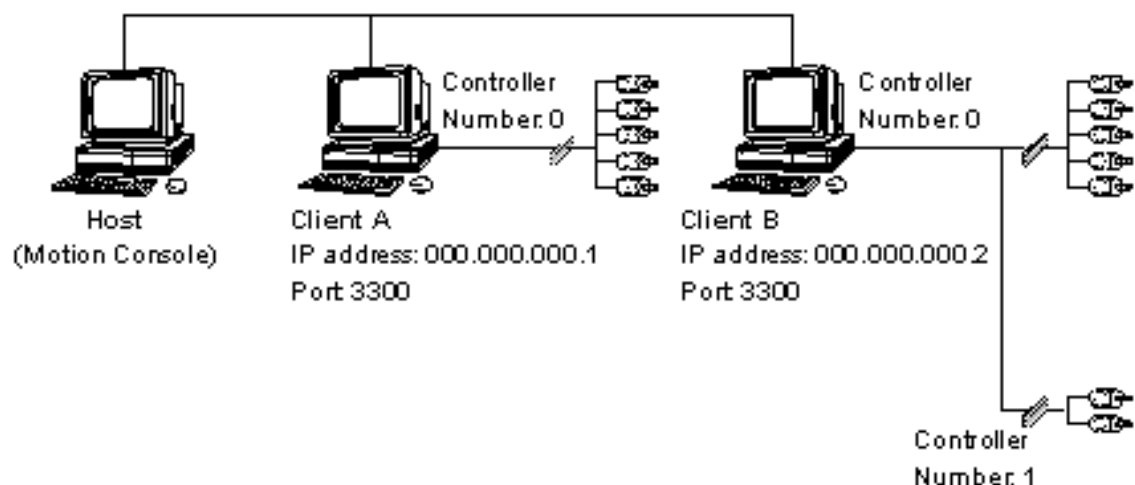


Controller-Client Configuration

A Controller is physically connected with a host computer via PCI bus. Hosts may be linked to clients, which allows virtually unlimited configuration combinations.



Each controller in a machine is assigned a unique device identification number by the operating system. Motion Console uses this number to communicate with the controller. When all controllers reside within a single host computer, Motion Console requires only Controller numbers (0, 1, 2, etc.) to differentiate between them. When controllers reside in separate client computers linked through a network to the Motion Console host, each controller must also be assigned a unique IP address and Port Number.



XMP Controller Placement on PCI Buses

PCI bus architecture imposes constraints which must be observed during card installation. When adding controllers to Motion Console, each XMP board is assigned a physical location on the PCI bus in the form of a

Controller Number. This is a device index number beginning at "0," which indexes with each new addition of a controller board, and which ignores non-MEI devices. Motion Console uses the device index number to associate configuration/status information with a particular controller.

Adding, removing or rearranging PCI devices can cause a motion controller device number to change. For systems with only a single controller, this is not a problem. For systems with multiple controllers, Motion Console will not know whether the device index to the physical controller map has changed. This is because some data is stored in the Motion Console initialization file and is associated with the device number. The following data will be associated with the wrong controller in Motion Console if the device number changes:

- Controller name
- Summary object list configuration
- Axis attributes:
 - Position 1
 - Position 2
 - Velocity
 - Acceleration
 - Deceleration
 - Jerk Percent

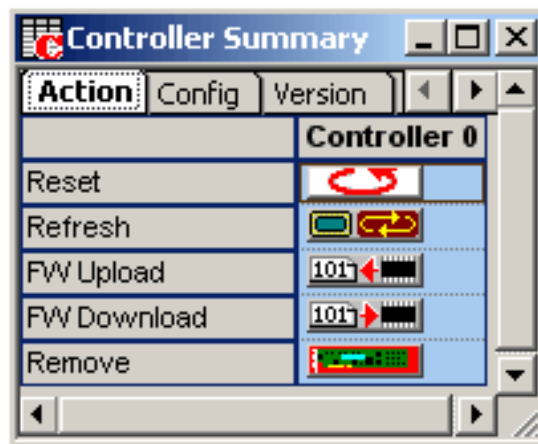


WARNING! When using Motion Console with multiple controller systems, make sure you know exactly which physical controller is associated with which device index. Be sure to always verify controller-to-device index relationships before commanding any motion.

The Controller Summary window lists Controller attributes in four tab pages: Actions, Config, Version, and Stats.

Configuration Attributes: "Action" Tab Page

"Action" Filter attributes include configurations for standard and custom filter features.



Reset – Resets the controller to the power-up configuration saved in flash memory.

Refresh – Refreshes display to reflect any external modifications.

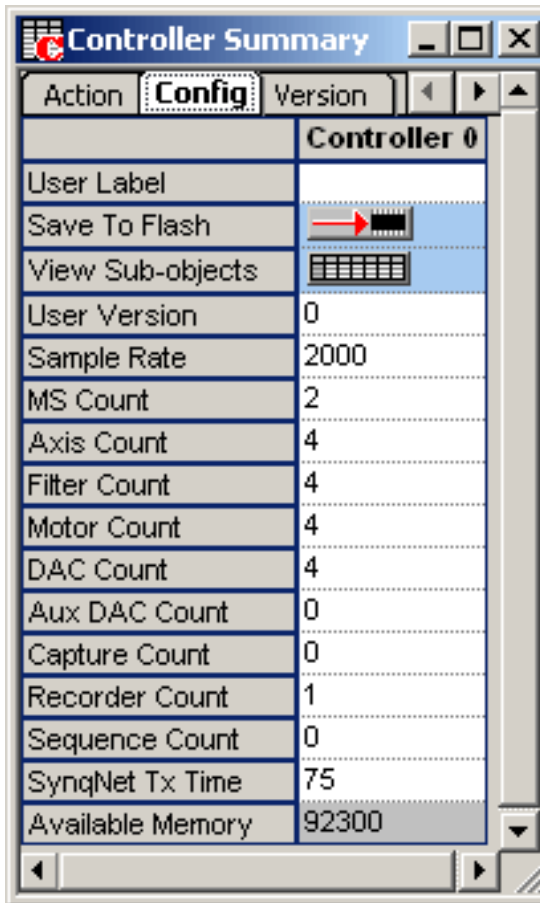
FW Upload – Uploads Flash memory to the firmware file. See the [Uploading Firmware](#) section earlier in this chapter for more information.

FW Download – Downloads the firmware file to Flash memory. See the [Downloading Firmware](#) section earlier in this chapter for more information.

Save Topology – Saves the current network topology to Flash memory.

Remove – Removes controller from Motion Console controller list.

Controller Configuration Attributes: "Config" Tab Page



User Label – User-defined label for the object.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for the Controller. For more information, see the [View Sub-object](#) section.

User Version – User defined version. See [MPIControlConfig](#).

Sample Rate – Sample Rate (samples/sec). See [MPIControlConfig](#).

MS Count – Number of enabled Motion Supervisor objects. See [MPIControlConfig](#).

Axis Count – Number of enabled Axis objects. See [MPIControlConfig](#).

Filter Count – Number of enabled Filter objects. See [MPIControlConfig](#).

Motor Count – Number of enabled Motor objects. See [MPIControlConfig](#).

DAC Count – Number of enabled DAC objects. See [MPIControlConfig](#).

Aux DAC Count – Number of enabled Auxiliary DAC objects. See [MPIControlConfig](#).

Capture Count – Number of enabled Capture objects. See [MEIMotorInfo](#)

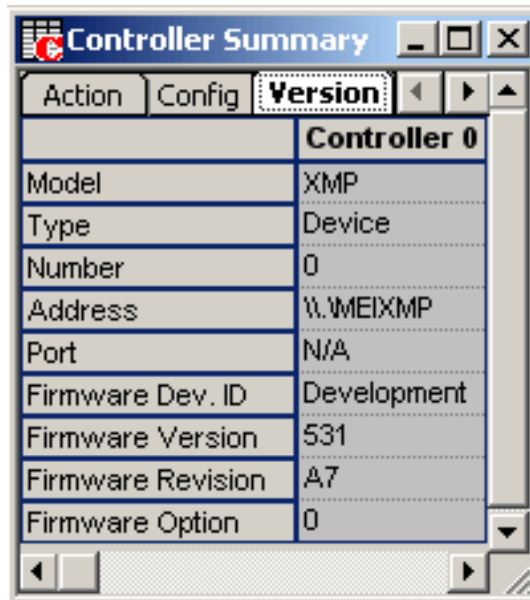
Record Count – Number of enabled Record objects. See [MPIControlConfig](#).

Sequence Count – Number of enabled Program Sequence objects. See [MPIControlConfig](#).

SynqNet Tx Time – The percentage of the sample rate to devote to transmitting SynqNet telegrams. This should be set in such a way that the transmissions occur after the firmware has completed its foreground cycle. (0-100%).

Available Memory – The amount of external memory available on the controller. The maximum value for Record Count and Axis Count is limited by the available memory.

Configuration Attributes: "Version" Tab Page



Model – They type of motion controller: XMP or ZMP.

Type – Currently either "Device" or "Client." See [MEIControllInfoHardware](#).

Number – The device number of the controller on the IDE bus.

Address – The device driver for "Device" type controllers, the IP address for "Client" type controllers.

Port – Port for "Client" type controllers only.

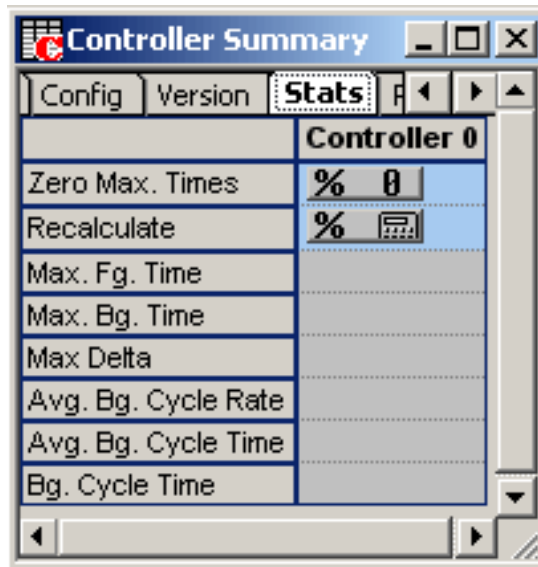
Firmware Dev. ID – Firmware Development ID.

Firmware Version – Firmware Version. See [MEIControllInfoFirmware](#).

Firmware Revision – Firmware Revision. See [MEIControllInfoFirmware](#).

Firmware Option – Firmware Option. See [MEIControllInfoFirmware](#).

Configuration Attributes: "Stats" Tab Page



Zero Max. Times – Zeros the maximum foreground and background timers.

Recalculate – Triggers a recalculation of the timing statistics.

Max Fg. Time – Maximum foreground time in microseconds. **Max Bg. Time**– Maximum background time in microseconds.

Max Delta – Maximum Delta (samples/ background cycle).

Avg. Bg. Cycle Rate – Average interrupted background cycle rate (cycles/sec).

Avg. Bg. Cycle Time – Average interrupted background cycle time in microseconds.

Bg. Cycle Time – Uninterrupted background cycle time in microseconds.

Configuration Attributes: "Recorders" Tab Page

Controller Summary	
Stats	Recorders
Controller 0	
Record Count 0	16384
Record Count 1	N/A
Record Count 2	N/A
Record Count 3	N/A
Record Count 4	N/A
Record Count 5	N/A
Record Count 6	N/A
Record Count 7	N/A
Record Count 8	N/A
Record Count 9	N/A

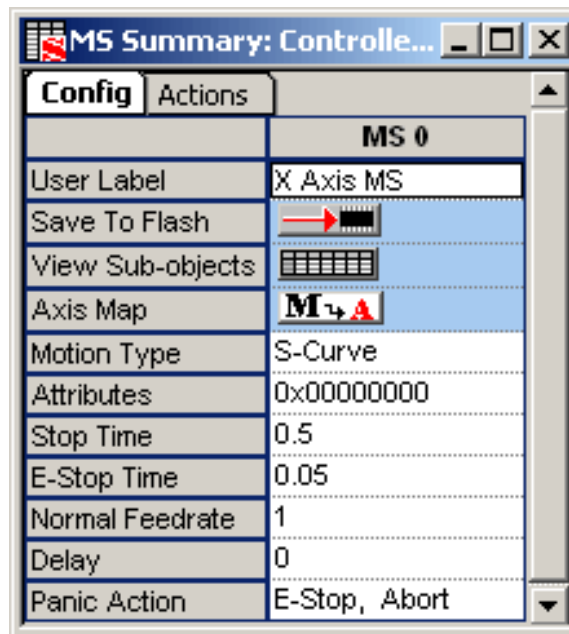
Record Count 0-31 – Number of data recorder objects enabled for a controller. The controller's recorder object handles collecting and buffering any data in controller memory. The enabled data recorders can collect up to a total of 32 addresses each sample. See [MPIControlConfig](#).

Motion Supervisor Object

Motion Supervisors can be thought of as specialized task-masters for motion control. Many system designers find it convenient to divide their system into specialized sub-tasks, each handled by a separate motion supervisor. As you organize your motion control system, the advantages of this approach will become obvious.

It is important to know that the Motion Supervisor is a host-based object. Please see the [Motion Supervisor and Axes Mapping: MS is a Host-based Object](#) animation.

The Motion Supervisor summary window is divided into two configuration tab pages, plus a General Status page as shown below, and includes such attributes as Motion Type and Stop Time:



User Label – User-defined label for the object.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

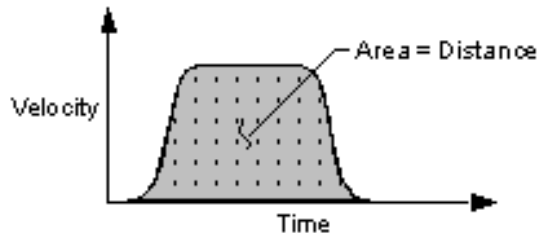
View Sub-objects – Shows all sub-objects for the Motion Supervisor. For more information, see [Object Summary Windows](#).

Axis Map – Displays the MS Axis List Configuration window. For more

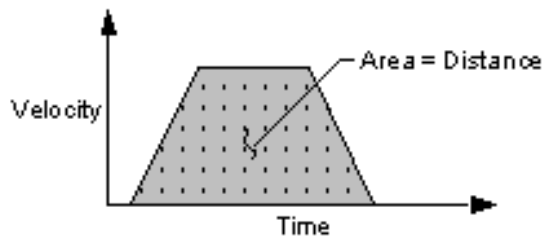
information, see the [Object List Configuration Dialog Boxes](#) section.

Motion Type – There are ten Motion Type settings; however, only three are supported in Motion Console. All are supported by the MPI library.

- **S-Curve** – Implements an S-curve velocity trajectory, using the Jerk Percent parameter to reach the target position.



- **Trapezoidal** – (Default) Implements a trapezoidal velocity trajectory to reach the target position. Does not use the Jerk Percent parameter.



- **Velocity** – Implements an S-curve velocity trajectory using the Jerk Percent parameter to reach the target velocity.
- Jog – (Unsupported)
- PT – (Unsupported)
- PVT – (Unsupported)
- Spline – (Unsupported)
- B-Spline 2– (Unsupported)
- Bessel– (Unsupported)
- B-Spline– (Unsupported)

Attributes – Applies special attributes to motion objects via mpiMotionModify.

Stop Time –Time (in seconds) to decelerate axis to a stop.



CAUTION! Actual stopping time is limited by mass and speed! If you are moving a heavy object at high speed and command a very rapid Stop Time, your motion control or mechanical system may be unable to comply. Verify that drive and motor manufacturer's specifications meet the demands placed upon your system **BEFORE** assuming a stop time.

E-Stop Time – Time to decelerate to an emergency stop. When E-Stop is activated, the State will be flagged with a Stopping Error while the axis is decelerated. An Error state is flagged when the axis comes to a complete stop. However, amp enable outputs are NOT disabled.

NOTE: If the axis is close to completing a move when E-Stop is activated, the axis may halt before the E-Stop Time has expired.

Normal Feedrate – Speed scaling factor applied to movement. 0 = stopped; 1 = normal speed (unscaled); 1.5 = 150%; 2 = 200%, -1 = reversed normal speed; etc.

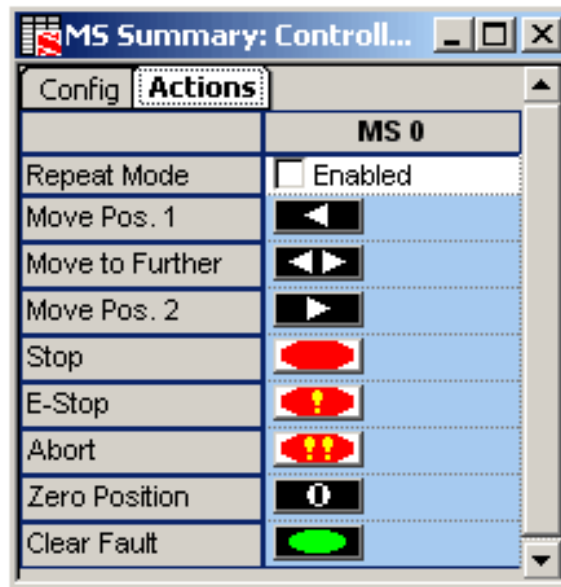
Delay – Time delay (in seconds) before execution of next move command. If operating in Repeat Mode, the delay will be applied to the beginning of each move segment.

Panic Action – Action to execute when the panic key is activated.

NOTE: Some panic action states (E-Stop and Abort) flag a fault condition. To reset the system after a fault, click on the Clear Fault button (MS Summary / Actions tab page).

- **None** – Motion is halted at the end of the currently commanded move. (No-fault condition.)
- **Stop** – Motion is halted using a Stop. (No-fault condition.)
- **E-Stop** – Motion is halted using an E-Stop, and a fault condition is flagged.
- **E-Stop, Abort** – Motion is halted using both an E-Stop and an Abort, and a fault condition is flagged. A fault condition is flagged.
- **Abort** – Motion is halted using an Abort.

Configuration Attributes: "Actions" Tab Page



Repeat Mode – [Enabled; (not enabled)] Repeats programmed motion.

NOTE: Each repeated motion will be delayed by the Delay amount (see description above on Delay).



Move Pos. 1 – Command each axis associated with the Motion Supervisor to Position 1.



Move to Further – Command each axis associated with the Motion Supervisor to either Position 1 or Position 2, whichever is further away.



Move Pos. 2 – Command each axis associated with the Motion Supervisor to Position 2.



Stop – Stops movement of objects under control of current Motion Supervisor within Stop time (seconds).



E-Stop – Emergency stop. Halts movement of objects under control of current Motion Supervisor within E-Stop time (seconds) and leaves the axes in an Error state.

NOTE: E-Stop does NOT disable closed-loop control and does not disable the amp enable output(s).



Abort – Immediately stops movement of objects under control of current Motion Supervisor, disables closed-loop control, and disables the amp enable output(s), leaving the axes in an Error state.



CAUTION! Do not use Abort when testing equipment which depends upon servos or stepper motors to secure personnel or equipment (e.g., lifting cranes, elevators, etc.). The Abort command disables servos and steppers and, in some circumstances, may present a hazard to personnel or equipment due to sudden loss of power.




Zero Position – Sets the origin of the axis to the current actual position and the command position to 0. The algorithm for setting the origin is: **new origin = current origin + actual position**



Clear Fault – Clears the Motion Supervisor's error state and all sub-object statuses. If you are having problems clearing the faults, it may be a result of improper axis mapping. Please see the [Troubleshooting](#) section.

Status Attributes








Status fields are read-only:

Status	
State	 Idle
Amp Fault	No
Home Limit	No
Position Err. Limit	No
HW Neg. Limit	No
HW Pos. Limit	No
SW Neg. Limit	No
SW Pos. Limit	No
Encoder Fault	No
Amp Warning	No
Motion Done	<input checked="" type="checkbox"/> Yes
At Velocity	No
Out of Frames	No

Motion States

State – Current motion state. See Motion States table below.

Motion States

State	Description
 (green)	Idle (State parameter). Awaiting a move command.
	In motion (State parameter). Performing a move command.
 (yellow)	Stopping. (State parameter). Motion has been commanded to stop.
 Moving	Moving when not commanded to move. NOTE: This occurs when another application commands a move.
	Initializing.
	Unknown state.
 Error	Fault or limit detected. (See Event Status Flags below.)

Error States



Yes

A black checkmark indicates something positive.
(ex: Motion Done, At Target)



Yes

A red checkmark indicates something potentially negative.
(ex: error states, limits, amp fault, etc.)

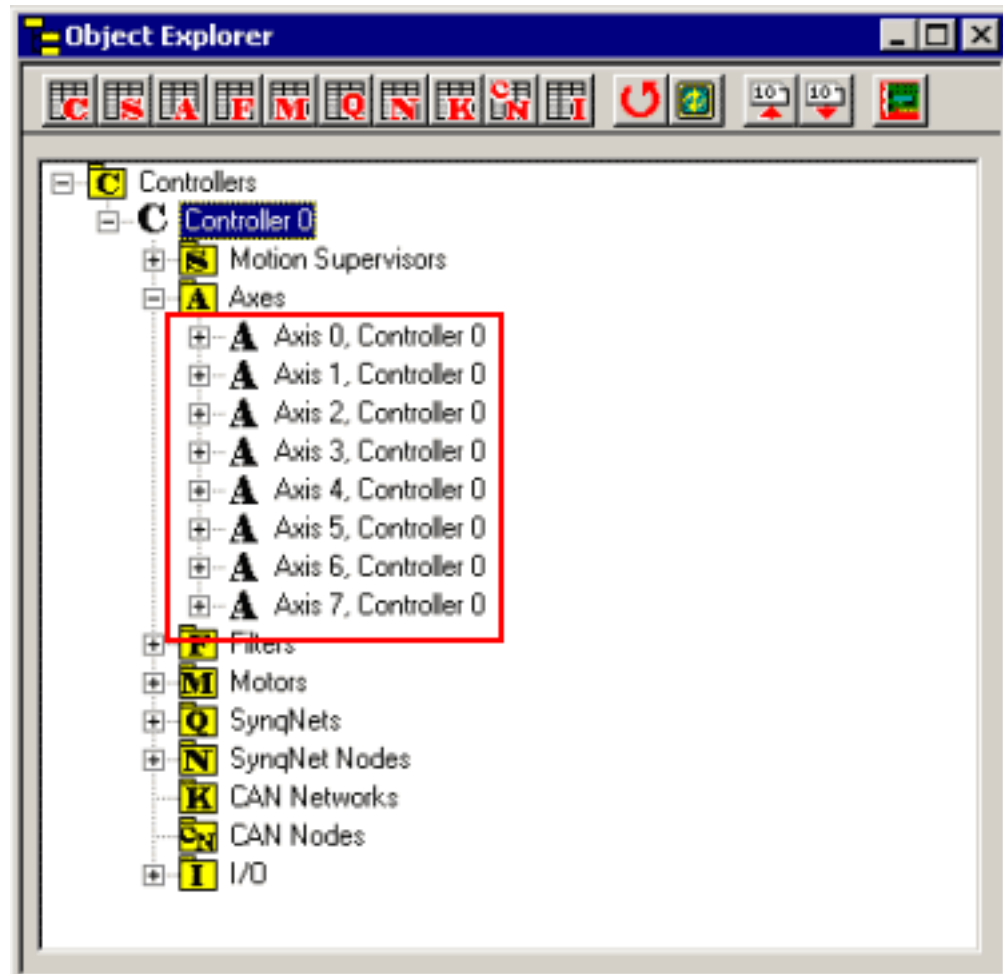
Event Status Flags

Error	Flag	Description
Amp[lifier] Fault	No Yes	Yes - Amplifier fault.
Home Limit	No Yes	Yes - Encoder home pulse exceeded or not found.
Position Error Limit	No Yes	Yes - Error between command and actual position exceeds user-defined limit.
HW Neg. limit	No Yes	Yes - Hardware negative limit activated.
HW Pos. limit	No Yes	Yes - Hardware positive limit activated.
SW Neg. limit	No Yes	Yes - Software negative limit exceeded.
SW Pos. limit	No Yes	Yes - Software positive limit exceeded.
Encoder Fault	No Yes	Yes - Encoder fault detected.
Amp Warning	No Yes	Yes - Amp warning.
Motion Done	No Yes	Yes - Move command completed.
At Velocity	No Yes	Yes - Commanded velocity attained.
Out of Frames	No Yes	Yes - Out of Frames. A Motion Done event was generated from a Motion Supervisor object. See MEIEventType .

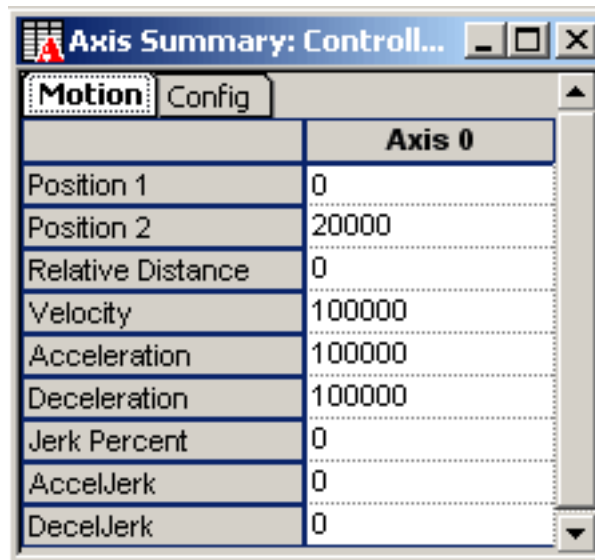
Axis Objects

Axis objects are classically associated with a single vector of motion, particularly as it relates to a specific motor. In Motion Console, Axis objects bridge the control link between Filters and Motion Supervisors.

Object Explorer Panel



Axis Summary Window



Configuration Attributes: "Motion" Tab Page

The following attributes appear on the Motion configuration tab page:

Position 1 – (counts) The home position. Used for non-relative moves.

Position 2 – (counts) The target position. Used for non-relative moves.
NOTE: Either Position 1 or 2 can be the target position at the end of the move.

Relative Distance – Position Offset (counts). Used for relative moves.

Velocity – Command velocity during move (in encoder counts / second).

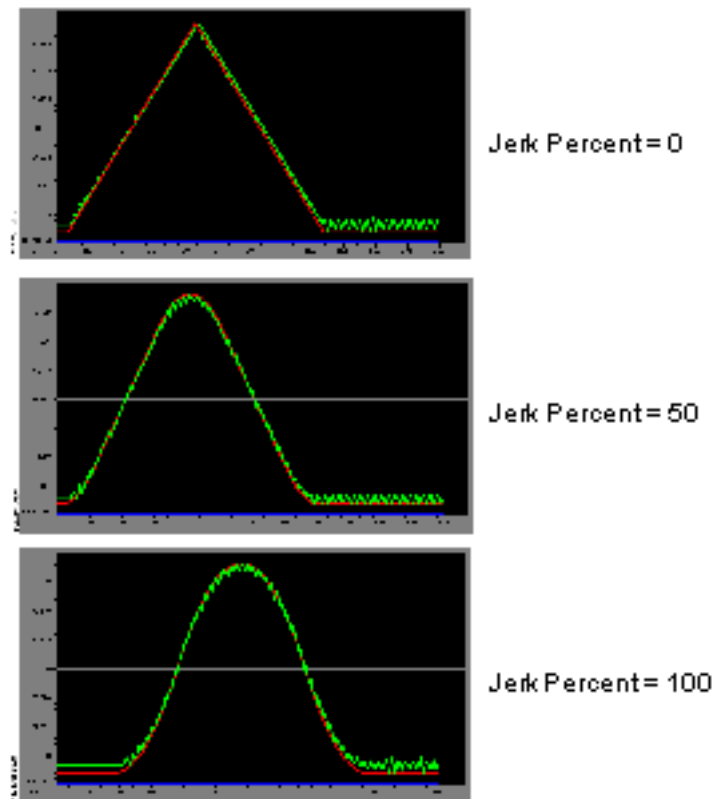
Acceleration – Command acceleration during move (in encoder counts / sec²).

Deceleration – Command deceleration during move (in encoder counts / sec²).

Jerk Percent – (0 - 100%) Percentage of acceleration / deceleration curve utilized for transitioning between Acceleration, Velocity, and Deceleration components. In the figure below, the effects of various Jerk Percent values are illustrated for a simple triangular profile move.

Changes in Jerk Percent parameter change the acceleration-

deceleration curve characteristics. The graphs below display commanded and actual velocities.

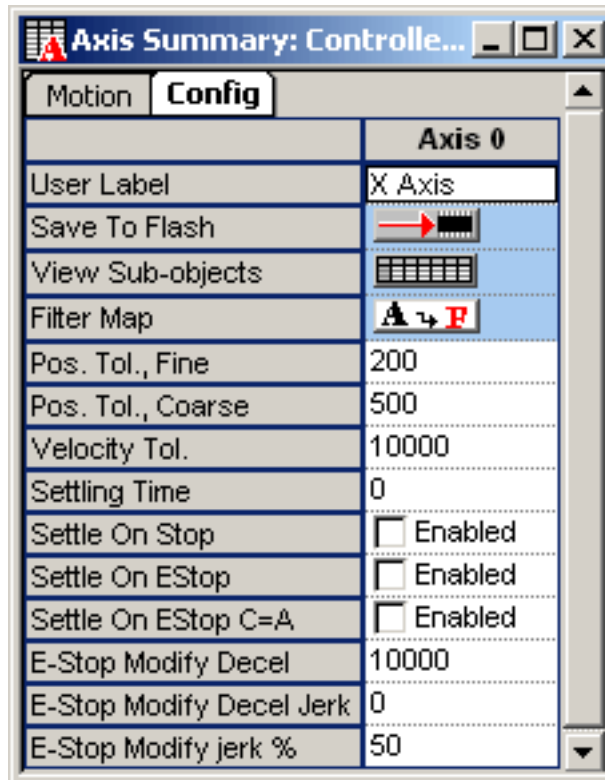


AccelJerk – Acceleration Jerk. User for move types, S-Curve Jerk and Velocity Jerk.

DecelJerk – Deceleration Jerk. User for move types, S-Curve Jerk and Velocity Jerk.

Configuration Attributes: "Config" Tab Page

The following attributes appear on the General Config tab page:



User Label – User-defined label for the object.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for Axis object. For more information, see [Object Summary Windows](#).

Filter Map – Displays the Axis Filter Map Configuration window. For more information, see [Object List Configuration Dialog Boxes](#). See [MEIAxisConfig](#).

Pos. Tol., Fine – Fine Positional Tolerance. Maximum allowable positional deviation (error) from target position to be considered done. See [MPIAxisInPosition](#).

Pos. Tol., Coarse – Coarse Positional Tolerance. Arbitrarily set deviation (error) from target position.

NOTE: The Pos. Tol., Coarse parameter is useful for triggering auxiliary events just prior to attaining target position. (e.g. On an automated milling machine, Pos. Tol., Coarse can be used to trigger the flow of

cutting fluid when the tool is close to the object being milled.) See [MPIAxisInPosition](#).

Velocity Tol. – Velocity Tolerance. Maximum allowable velocity deviation (error) from target velocity to be considered "at velocity." See [MPIAxisInPosition](#).

Settling Time – Minimum time interval for both the Fine Pos[itional] Tol[erance] and Velocity Tol[erance] attributes to qualify as "done." NOTE: As they approach their target positions / velocities, all axes will spend some interval of time settling. Only when an axis attains both its Fine Positional Tolerance and Velocity Tolerance within the Settling Time is that axis regarded as truly "done"; this triggers a Motion Done event (See [Motion Supervisor](#) objects). See [MPIAxisInPosition](#).

Settle on Stop – When checked, the settling criteria is applied to moves halted by a stop. See [MPIAxisInPosition](#).

Settle on EStop – When checked, the settling criteria is applied to moves halted by an E-Stop. See [MPIAxisInPosition](#).


Settle on EStop C=A – When checked, the settling criteria is applied to moves halted by an E-Stop. During an Estop, the command position is set equal to the actual position from the previous servo sample. This mode is not recommended. See [MPIAxisInPosition](#).

E-Stop Modify Decel – See [MPIAxisEstopModify](#).

E-Stop Modify Decel Jerk – See [MPIAxisEstopModify](#).

E-Stop Modify Jerk % – See [MPIAxisEstopModify](#).

Status Parameters

Status	
State	 Idle
Actual Position	3
Command Position	3
Position Error	0
Velocity	0
Acceleration	0
In Coarse Pos.	No
In Fine Pos.	<input checked="" type="checkbox"/> Yes
At Target	No

State – n/a

Actual Position – Real, current position of axis. Where the axis actually is.

Command Position – Commanded axis position. Where the axis should be.

Position Error – Difference between Command and Actual positions.

Velocity – Command velocity.

Acceleration – Command acceleration.

In Coarse Position – No | ☒ Yes

Yes - An In Coarse Position event was generated from an Axis object. See [MEIEventType](#).






In Fine Position – No | ☒ Yes

Yes - An In Fine Position event was generated from an Axis object. See [MEIEventType](#).

At Target – No | ☒ Yes

Yes - See [MPIStatus](#) and [MEIEventType](#).

Motion States and Descriptions

Motion State	Description
 Moving	Moving toward target position 1.
 Moving	Moving toward target position 2.
 Moving	Moving when not commanded to move. NOTE: this occurs when another application commands a move while Motion Console is displaying the MS.
 Error	Error state
 Idle	Idle

Filter Objects

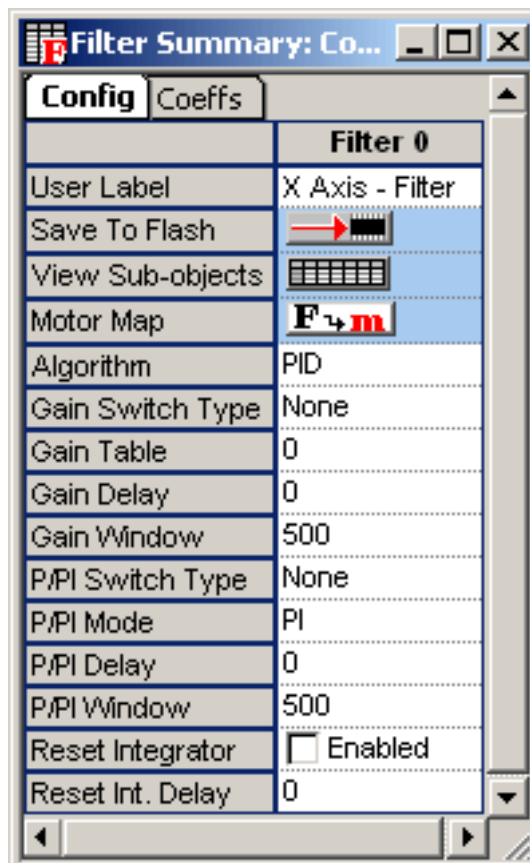
Filters provide linkage between motors and axes: motors are mapped to filters, and filters are mapped to axes. (It is not possible to map a motor directly to an axis without a filter.) Depending upon the algorithm employed, filters use such coefficients as gains, feed forward and torque limits.

Filter Configuration Attributes

The Filter Summary window lists Filter attributes in two tab pages: "Config" and "Coeffs."

Configuration Attributes: "Config" Tab Page

"Config" Filter attributes include configurations for standard and custom filter features.



User Label – User-defined label for the object.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for Filter object. For more information, see [Object Summary Windows](#).

Motor Map – Displays the Filter Motor Map Configuration window. For more information, see [Object List Configuration Dialog Boxes](#).

Algorithm – Software filter algorithm. There are five types:

- **None** – Filter is OFF. No algorithm is applied.
- **PID** – Proportional Integral Differential algorithm.
- **PIV** – Proportional Integral Velocity.
- **PIV1** – Proportional Integral Velocity.
- **User** – User-defined algorithm defined in firmware. (For more information, please contact MEI.)

Gain Switch Type – Used to switch filter parameters from the gain tables.

- **None** – Gain switching is disabled.
- **Motion Only** – Switch gains based on controller's switching algorithm.
- **Window** – **Requires custom firmware**. Switch gains based on position window.
- **User** – **Requires custom firmware**. Switch gains based on user criteria.

Gain Delay – **Requires custom firmware**. It specifies the time (seconds) between the controller's calculated gain switch and the applied gain switch.

Gain Window – Requires custom firmware. It specifies the position window (counts) to apply the gain switch.

P/PI Switch Type – Requires custom firmware.

P/PI Mode – Requires custom firmware.

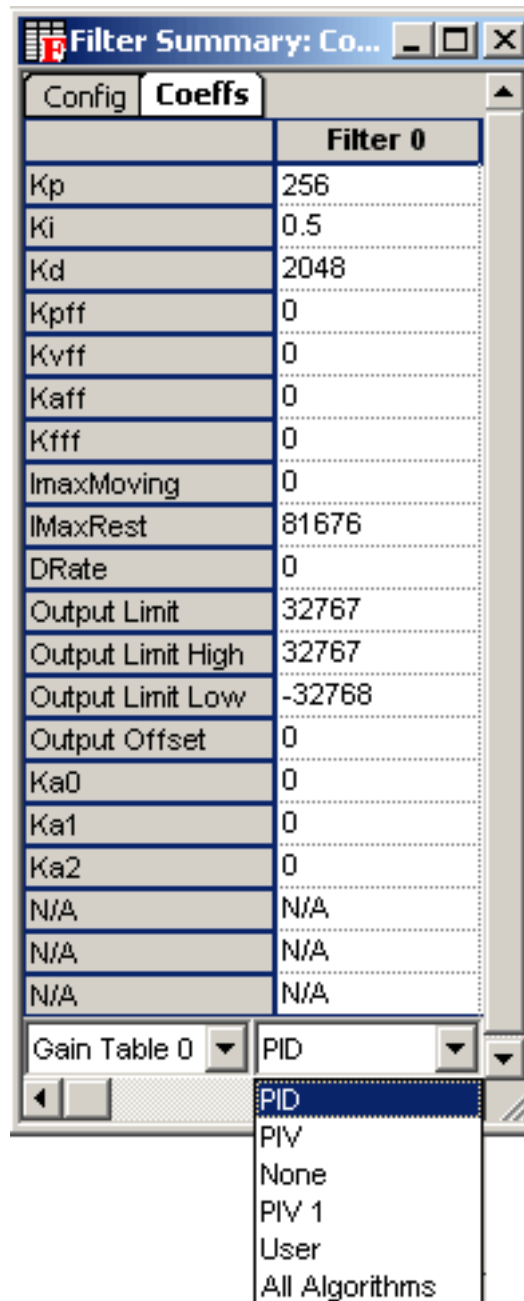
P/PI Delay – Requires custom firmware.

P/PI Window – Requires custom firmware.

Reset Integrator – Requires custom firmware.

Reset Int. Delay – Requires custom firmware.

Configuration Attributes: "Coeffs" Tab Page



The display of filter coefficients changes according to the Algorithm chosen (selectable from the Filter / General panel). The algorithm combo box on the Coeffs tab merely changes how the coefficients are displayed. It changes the row headers and displays N/A for coefficients that are not used for the selected algorithm. If the algorithm of a filter object does not match the algorithm being displayed, then N/A is displayed for all coefficients.

Each set of coefficients is described in separate tables below.

PID Coefficients

PID coefficients are defined in the table below. See [MEIFilterGainPID](#). See the [Scaling PID Tuning Parameters for Different Controller Sample Rates](#) application note.

Filter Coefficients (PID Algorithm)

Coefficient	Description
Kp	Proportional gain
Ki	Integral gain
Kd	Derivative gain
Kpff	Position feed forward gain
Kvff	Velocity feed forward gain
Kaff	Acceleration feed forward gain
Kfff	Friction feed forward gain
ImaxMoving	Position loop integration maximum (while moving)
IMaxRest	Position loop integration maximum (while resting)
DRate	Derivative sub-sampling rate. See The PID Algorithm and DRate .
Output Limit	Voltage command output limit
Output Limit High	Maximum command output voltage limit.
Output Limit Low	Minimum command output voltage limit.
Output Offset	Offset added to output.
Ka0	Fast Fourier transform (FFT) testing gain (0)
Ka1	Fast Fourier transform (FFT) testing gain (1)

Ka2	Fast Fourier transform (FFT) testing gain (2)
-----	---

PIV Coefficients

PIV coefficients are described in the table below. See [MEIFilterGainPIV](#).

Filter Coefficients (PIV Algorithm)

Coefficient	Description
Kpp	Position loop proportional gain
Kip	Position loop integral gain
Kpv	Velocity loop proportional gain
Kvff	Velocity feed forward gain
Kaff	Acceleration feed forward gain
Kfff	Friction feed forward gain
ImaxMoving	Position loop integration maximum (while moving)
IMaxRest	Position loop integration maximum (while resting)
Kdv	Velocity estimate feedback gain
Output Limit	Voltage command output limit
Output Limit High	Upper voltage command output limit.
Output Limit Low	Lower voltage command output limit.
Output Offset	Offset added to output
Kiv	Velocity loop integral gain
VintMax	Velocity loop integration maximum

Ka0	Fast Fourier transform (FFT) testing gain (0)
Ka1	Fast Fourier transform (FFT) testing gain (1)

None Coefficients

Selecting the None option means that no filter parameters will be used. This is common for stepper motors.

PIV1 Coefficients

PIV1 coefficients are defined in the table below.

Filter Coefficients (PID Algorithm)

Coefficient	Description
Kpp	Position loop proportional gain
Kip	Position loop integral gain
Kpv	Velocity loop proportional gain
Kpff	Position feed forward gain
Kvff	Velocity feed forward gain
Kaff	Acceleration feed forward gain
Kfff	Friction feed forward gain
ImaxMoving	Position loop integration maximum (while moving)
IMaxRest	Position loop integration maximum (while resting)
Kdv	Velocity estimate feedback gain

Output Limit	Voltage command output limit
Output Limit High	Maximum command output voltage limit.
Output Limit Low	Minimum command output voltage limit.
Output Offset	Offset added to output.
Kiv	Velocity loop integral gain
VintMax	Velocity loop integration maximum
Ka0	Fast Fourier transform (FFT) testing gain (0)
Ka1	Fast Fourier transform (FFT) testing gain (1)
Ka2	Fast Fourier transform (FFT) testing gain (2)

User Coefficients

These coefficients are used with a custom algorithm developed using the Firmware Development Kit.

All Algorithm Coefficients

Displays all coefficients for the different filter objects. Because different filter objects may use different algorithms, Motion Console must display generic labels rather than algorithm specific labels.

Motor Objects

Motor objects are mapped to Filter objects. Because motor type determines many other attributes found in Motion Console, it is essential that attributes in the Motor Summary window are the first ones configured.

Motor Summary Window

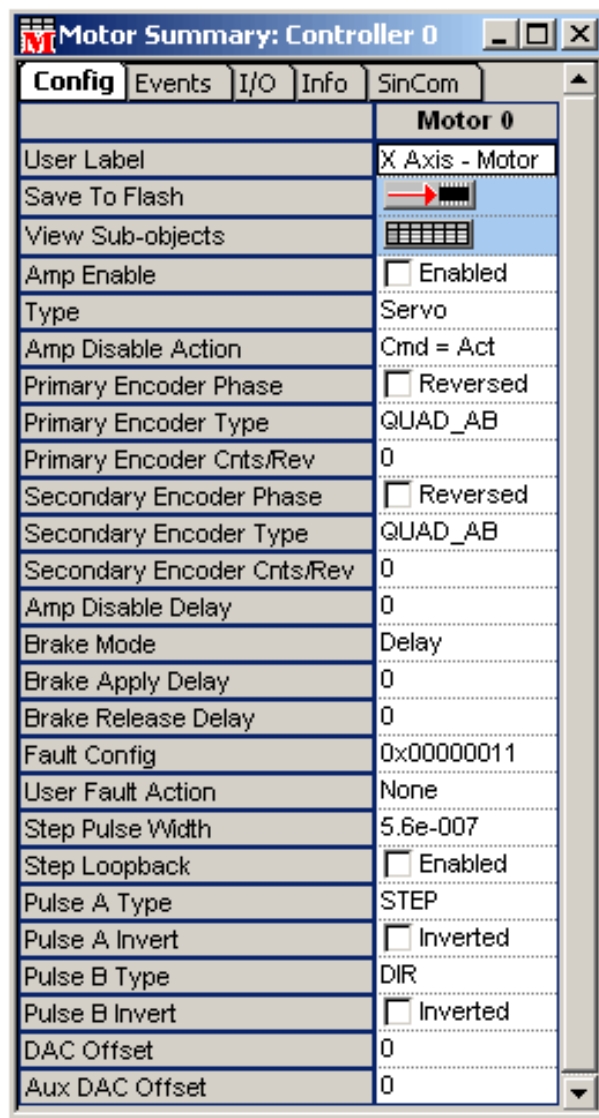
The four configuration pages are:

- **Config**– General motor, stepper, and encoder configuration table.
- **Events**– Dedicated I/O and encoder event configuration table.
- **I/O**– Transceiver and User I/O configuration table.
- **Info**– Resource information about the motor. Data is read-only.
- **SinCom**– Sinusoidal Commutation configuration table.

There are also two status pages:

- **Status**– Fault and Limit event status display.
- **I/O**– Dedicated I/O, Transceiver I/O and User I/O status display.

Configuration Attributes: "Config" Tab Page



User Label – User-defined label for the object.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for Motor object. For more information, see [Object Summary Windows](#).

Amp Enable – When checked, the amp is enabled. When unchecked, the amp is disabled.

Type – See [MPIMotorType](#).

- **Servo** – Configure a servo-type motor (including DC brush/brushless and linears) for $\pm 10V$ servo output.
- **Stepper** – Configure for step motor operation (step/direction, or clockwise/counterclockwise). See the Step and Pulse sections for additional configurations.

Amp Disable Action – See [MEIMotorDisableAction](#).

- **None** – No action.
- **Cmd = Act** – Action is taken when amp is enabled.

Primary Encoder Phase – Determines which direction of motor movement results in incrementing or decrementing encoder counts. It is very important to use the correct encoder phasing so that positive voltage will result in increasing counts and vice-versa. **NOTE:** phasing can also be reversed by reconfiguring the A+/- and B+/- encoder wires, e.g., swapping the A+ and A- wires.

- **(not reversed)** – Encoder phasing is passed to the controller directly, as wired.
- **Reversed** – Inverts direction of actual position traveled for given encoder feedback.

Primary Encoder Type –

- **Quad A_B** – Standard incremental-type quadrature encoders. No absolute position information is stored.
- **Drive** – Reads feedback from SynqNet node to drive memory interface. Either incremental or absolute position information, depending on the SynqNet drive implementation.
- **SSI** – Synchronous Serial Interface (absolute position serial encoder interface). Requires SynqNet node hardware support and FPGA support. Please contact MEI for details.

Primary Encoder Cnts/Rev – Encoder counts per revolution. When available, the counts per rev is read from the SynqNet drive. At present, SynqNet drives do not support configurable counts per revolution for feedback devices. See [MEIMotorEncoder](#).

Secondary Encoder Phase – Determines which direction of motor movement results in incrementing or decrementing encoder counts. It is very important to use the correct encoder phasing so that positive voltage will result in increasing counts and vice-versa. **NOTE:** phasing can also be reversed by reconfiguring the A+/- and B+/- encoder wires, e.g., swapping the A+ and A- wires.

- **(not reversed)** – Encoder phasing is passed to the controller directly, as wired.
- **Reversed** – Inverts direction of actual position traveled for given encoder feedback.

Secondary Encoder Type –

- **Quad A_B** – Standard incremental-type quadrature encoders. No absolute position information is stored.
- **Drive** – Reads feedback from SynqNet node to drive memory interface. Either incremental or absolute position information, depending on the SynqNet drive implementation.
- **SSI** – Synchronous Serial Interface (absolute position serial encoder interface). Requires SynqNet node hardware support and FPGA support. Please contact MEI for details.

Secondary Encoder Cnts/Rev – Encoder counts per revolution. When available, the counts per rev is read from the SynqNet drive. At present, SynqNet drives do not support configurable counts per revolution for feedback devices. See [MEIMotorEncoder](#).

Amp Disable Delay – Delay (in seconds) before disabling amplifier on events that cause an Abort.

Brake Mode (None or Delay) –

- **None** – None disables the motor brake feature.
- **Delay** – Delay enables the motor brake feature, and sets functionality to use a delay between the amp enable output and brake output. Setting Brake Mode to Delay does not require a non-zero delay.

Brake Apply Delay – Delay (in seconds) between releasing the brake and enabling

the motor. For more information, please see the [Motor Brake \(brake enable/disable delay\)](#) page. See [MPIMotorBrake](#).

Brake Release Delay – Delay (in seconds) between enabling the brake and disabling the motor. For more information, please see the [Motor Brake \(brake enable/disable delay\)](#) page. See [MPIMotorBrake](#).

Fault Config – A mask of motor fault bits. The masked motor fault bits will be monitored by the controller's motor object. If any masked motor fault bit is active (TRUE), the motor's dedicated amp fault input (MEIMotorDedicatedInAMP_FAULT) is set active (TRUE). The support for motor fault bits is node/drive specific. During SynqNet network initialization, the SqNodeLib automatically configures the faultMask based on the node type. See the node/drive manufacturer's documentation for details. See [MEIMotorFaultConfig](#).

User Fault Action – The action taken in the event of a User Fault. The criteria for a User Fault can be set on the SqNode Config Summary page. For more information, see [MEIMotorConfig](#) and [MEISqNodeConfigUserFault](#).

Step Pulse Width – Value is determined by pulse drive specifications.

Step Loopback – Enable for open loop, Disable (default) for encoder feedback.

Pulse A Type – Configure for STEP, DIR, CW, CCW, QUADA, or QUADB.

Pulse A Invert – Enable or Disable.

Pulse B Type – Configure for STEP, DIR, CW, CCW, QUADA, or QUADB.

Pulse B Invert – Enable or Disable.

DAC Offset – Adds an offset value to the Motor's DAC Output signal. Valid values are +10V through -10V.

AUX DAC Offset – Adds an offset value to the Motor's Auxiliary DAC Output signal. Valid values are +10V through -10V.

Configuration Attributes: "Events" Tab Page

The generation of Motor events is based upon the Motor configuration attributes. Motor events are state changes in the motor object that can be programmed to trigger a particular action. The status of motor events can be monitored in the Event Status page of the Motor Summary window. Motor events are passed up to the Motion

Supervisor object (via Filter and Axis objects) and can be monitored as event status flags on the Status tab page on the Motion Supervisor Summary window. If action is taken when the event is triggered, then the event status flag is "sticky," i.e., stays on until explicitly cleared by the user. This allows the user to determine which motor(s) caused an error. Events have some or all of the below configurable attributes.

Motor 0	
Amp Fault Trigger	<input checked="" type="checkbox"/> High
Amp Fault Action	Abort
Amp Fault Duration	0
Amp Warning Trigger	<input checked="" type="checkbox"/> High
Amp Warning Action	None
Amp Warning Duration	0
Home Trigger	<input checked="" type="checkbox"/> High
Home Action	None
Home Duration	0
Error Limit Trigger	1000
Error Limit Action	Abort
Error Limit Duration	0
H/W Neg. Lim. Trig.	<input checked="" type="checkbox"/> High
H/W Neg. Lim. Act.	E_Stop
H/W Neg. Lim. Dir.	<input type="checkbox"/> Enabled
H/W Neg. Lim. Dur.	0
H/W Pos. Lim. Trig.	<input checked="" type="checkbox"/> High
H/W Pos. Lim. Act.	E_Stop
H/W Pos. Lim. Dir.	<input type="checkbox"/> Enabled
H/W Pos. Lim. Dur.	0
SW Neg. Lim. Trig.	-1073741824
SW Neg. Lim. Act.	E_Stop
SW Pos. Lim. Trig.	1073741823
SW Pos. Lim. Act.	E_Stop
Encoder Fault Trig.	Primary
Encoder Fault Act.	None
Encoder Fault Dur.	0

Triggers (Trig.) can be a state change (e.g., Amp Fault Trigger or Hardware Negative Limit Trigger) or positional information (e.g., Error Limit Trigger). For check boxes, enable (default) for active high logic, disable for active low logic.

Actions (Act.) are taken when an event is triggered (see "Triggers" section above). In all cases, the following actions are available:

None – No action taken. The event status will be flagged only while the condition of the event is met. For all other actions, the event status is "sticky."

Stop – Stop the motor within the Stop Time defined in the Motion Supervisor / General Config tab page associated with the motion of this motor. There is no effect on the state of the axis (axes) or Motion Supervisor(s) associated with the motor. The event status is "sticky," but it will clear the next time motion is commanded.

E-Stop – Stop the motor within the E-Stop Time defined in the Motion Supervisor / General Config tab page associated with the motion of this motor. The Axis (or Axes) and Motion Supervisor(s) associated with the motor will be left in an error state.

E-Stop, Abort – Same as E-Stop action (above), except that the motor amplifier is disabled after the motor stops. An additional delay can be defined by setting the Amp Disable Delay attribute on the Motor Summary / General Config tab page.

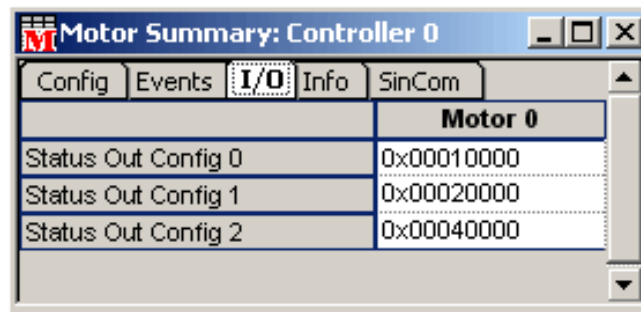
Abort – Disable the motor amplifier. A delay can be inserted before disabling the amplifier by setting the Amp Disable Delay attribute on the Motor Summary / General Config tab page.

Duration (Dur.) - The minimum amount of time, in seconds, that the Trigger condition must exist for the event to trigger.

NOTE: The duration parameter for the home limit should be set to zero. A non-zero value will result in the home limit being missed.

Direction (Dir.) - When enabled, it characterizes the Hardware Negative or Positive Limit Trigger directionally. When not enabled, the event will always be triggered, even when the motor is not in motion. When enabled, both hardware and software limit events are triggered when the motor is commanded to move only in the Direction the limit is associated with.

Configuration Attributes: "I/O" Tab Page



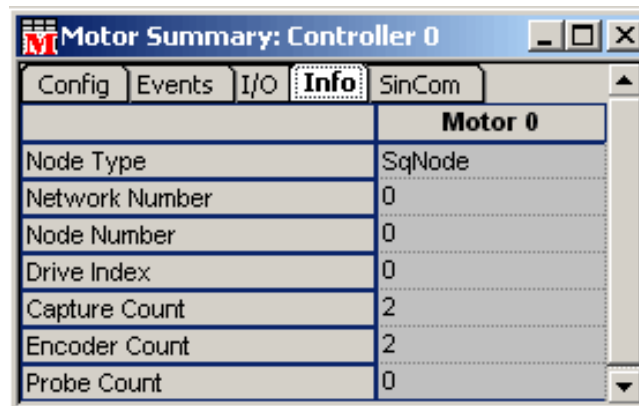
The screenshot shows the 'Motor Summary: Controller 0' window with the 'I/O' tab selected. The table displays status output configurations for Motor 0.

Motor 0	
Status Out Config 0	0x00010000
Status Out Config 1	0x00020000
Status Out Config 2	0x00040000

Status Out Config – Requires custom firmware.

- **Status Out Config 0** – n/a
- **Status Out Config 1** – n/a
- **Status Out Config 2** – n/a

Configuration Attributes: "Info" Tab Page



The screenshot shows the 'Motor Summary: Controller 0' window with the 'Info' tab selected. The table displays configuration attributes for Motor 0.

Motor 0	
Node Type	SqNode
Network Number	0
Node Number	0
Drive Index	0
Capture Count	2
Encoder Count	2
Probe Count	0

Node Type – Identifies the type of node. See [MEIMotorInfoNodeType](#).

Network Number – An index to the SynqNet network (0, 1, 2 etc.).

Node Number – An index to the node (0, 1, 2, etc.).

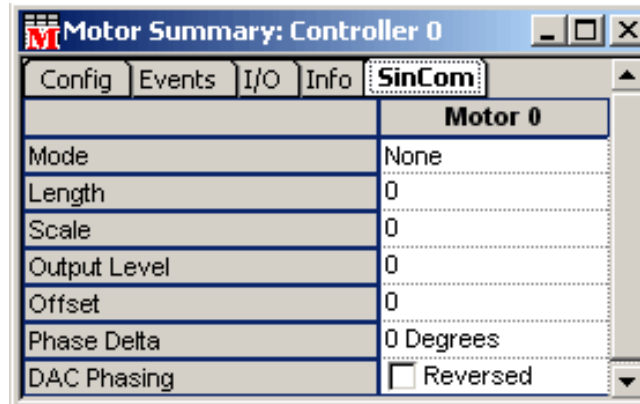
Drive Index – An index to the drive interface (0, 1, 2, etc.).

Capture Count – The number of captures for the motor.

Encoder Count – The number of encoders for the motor.

Probe Count – The number of hardware Probe engines for the motor.

Configuration Attributes: "SinCom" Tab Page



Mode –

- **None** – Non-commutated.
- **Closed Loop Mode** – Commutated, closed loop.
- **Open Loop Mode** – Commutated, open loop.
- **Simulate** – Sets the motor commutation mode to MEIXmpCommModeSIMULATE.

Length – Number of encoder counts per revolution for a rotary motor and the number of counts of one electrical cycle length for a linear motor.

Scale – (Type float) Total number of commutation points per motor revolution, divided by the total number of encoder counts per revolution (i.e., Length).

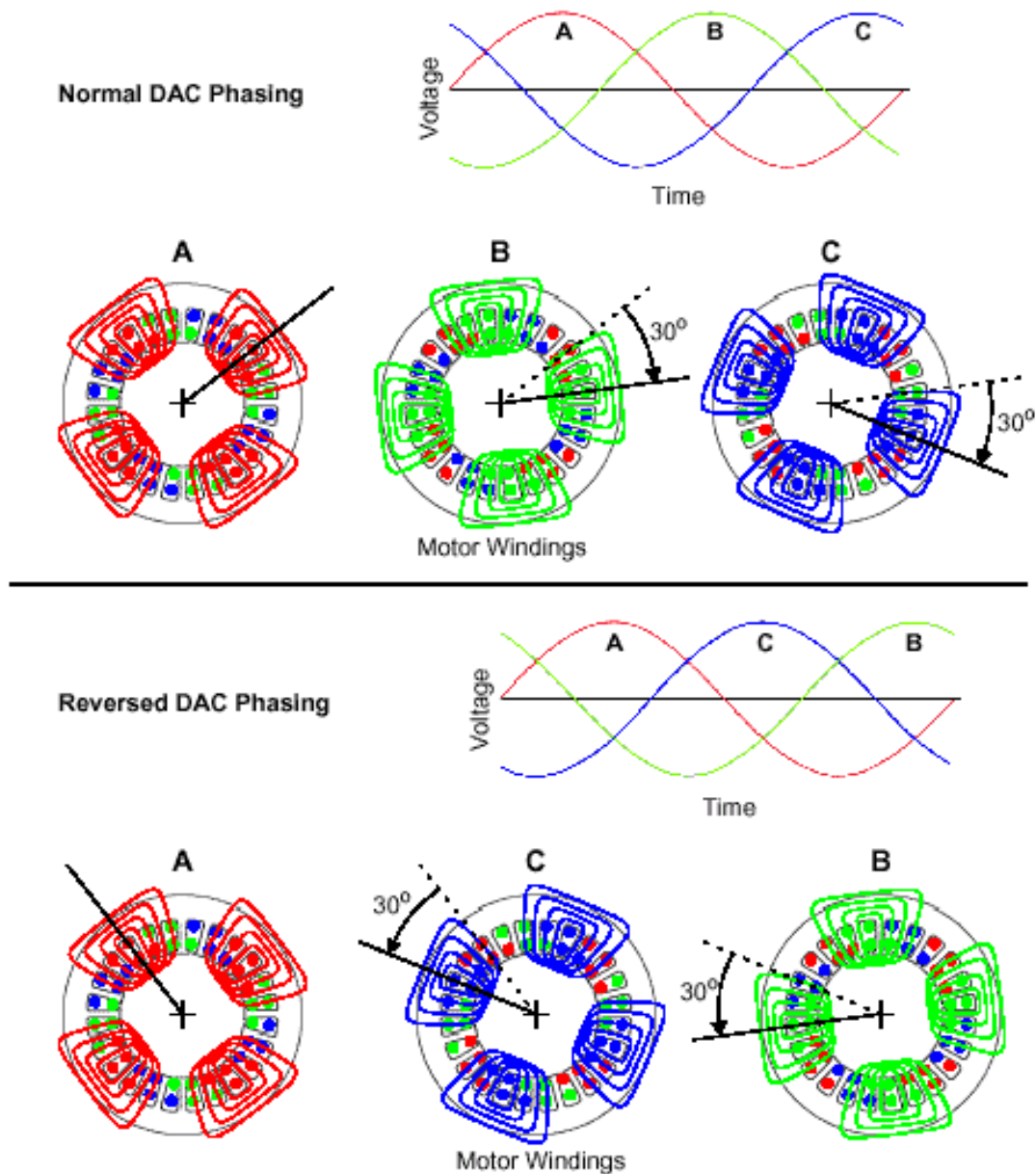
Output Level – Sets the DAC level during open-loop commutation. Output level is in DAC units (i.e., 32,767 = 10V).

Offset – Expressed in units of commutation table points (i.e., 1024). 1024 table points equal 360 electrical degrees.

Phase Delta – Usually set to 120 degrees for a three-phase, brushless motor with 120 degrees of phase separation. [120 degrees is represented by 341 commutation table points (i.e., $1024/3$).] Switchable options include: 0 degrees; 90 degrees; 120 degrees. **NOTE:** 90 degrees is normally reserved for a two- or four-phase motor.

DAC Phasing – The digital-to-analog converter (DAC) provides signals for three-phase servo motor drives. Depending upon how the servo motor and drive are wired, the DAC Phasing Parameter can be used to operate a motor in Normal or Reversed phase.

- **(Normal)** – See diagrams below.
- **Reversed (checked)** – See diagrams below.



Status Attributes: "Status" Tab Page

This page displays the status of motor events, as well as several general stats flags. DAC output level is also displayed. For all event status attributes, see the corresponding EventType.

- ▼ Yes A gray checkmark indicates that the bit is on, but there is not an error.
(ex: Broken Wire 2)
- ▼ Yes A black checkmark indicates something positive.
(ex: Motion Done, At Target)
- ▼ Yes A red checkmark indicates something potentially negative.
(ex: error states, limits, amp fault, etc.)

Status	I/O
Fault Bit, Amp	No
Fault Bit, Drive	No
Fault Bit, Watchdog	No
Fault Bit, Checksum	No
Fault Bit, Amp Not Powered	No
Fault Bit, Drive Not Ready	No
Fault Bit, Primary Feedback	No
Fault Bit, Secondary Feedback	No
Amp Fault	No
Amp Warning	No
Home Limit	No
Position Err. Limit	No
HW Neg. Limit	No
HW Pos. Limit	No
SW Neg. Limit	No
SW Pos. Limit	No
Encoder Fault	No
Broken Wire	No
Illegal State	No
Broken Wire 2	No
Illegal State 2	No
Stepper Pulse Lock Lost	No
Stepper Pulse Bad	No
DAC Level	0
Aux DAC Level	0
Primary Feedback	1
Secondary Feedback	-2

Fault Bit, Amp – Status of the Amp Motor Fault bit. See [MEIMotorFaultMaskAMP](#).

Fault Bit, Drive – Status of the Motor Fault bit. See [MEIMotorFaultMaskDRIVE](#).

- ▼ Yes The fault bit is on, but it is not configured to trigger a fault
(see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Watchdog – Status of the Watchdog Motor Fault bit. See [MEIMotorFaultMaskWATCHDOG](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Checksum – Status of the Checksum Motor Fault bit. See [MEIMotorFaultMaskCHECKSUM](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Amp Not Powered – Status of the Amp Not Powered Motor Fault bit. See [MEIMotorFaultBitAMP_NOT_POWERED](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Drive Not Ready – Status of the Drive Not Ready Motor Fault bit. See [MEIMotorFaultBitDRIVE_NOT_READY](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Primary Feedback – Status of the Primary Feedback Motor Fault bit. See [MEIMotorFaultMaskFEEDBACK](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Fault Bit, Secondary Feedback – Status of the Secondary Feedback Motor Fault bit. See [MEIMotorFaultMaskFEEDBACK](#).

▼ Yes The fault bit is on, but it is not configured to trigger a fault (see MEIMotorConfig.faultConfig.faultMask).

Amp Fault – Amp Fault Event status. See [MPIEventTypeAMP_FAULT](#).

Amp Warning – Status of the Amp Motor Warning bit. See [MEIMotorFaultMaskWARNING](#).

Home Limit – Home Limit Event status. See [MPIEventTypeHOME](#).

Position Err Limit – Position Error Limit Event status. See [MPIEventTypeLIMIT_ERROR](#).

HW Neg. Limit – Hardware Negative Limit Event status. See [MPIEventTypeLIMIT_HW_NEG](#).

HW Pos. Limit – Hardware Positive Limit Event status. See [MPIEventTypeLIMIT_HW_POS](#).

SW Neg. Limit – Software Negative Limit Event status. See [MPIEventTypeLIMIT_SW_NEG](#).


SW Pos. Limit – Software Positive Limit Event status. See [MPIEventTypeLIMIT_SW_POS](#).

Encoder Fault – Encoder Fault event status. See [MPIEventTypeENCODER_FAULT](#).


Broken Wire – Broken wire on the primary encoder input signals. See [MEIStatusMaskBROKEN_WIRE](#).

Illegal State – Illegal encoder logic state on the primary encoder input signals. See [MEIStatusMaskILLEGAL_STATE](#).

Broken Wire 2 – Broken wire on the secondary encoder input signals. See [MEIStatusMaskBROKEN_WIRE_SECONDARY](#).

 **Yes** The status bit is on, but there is no secondary encoder.

Illegal State 2 – Illegal encoder logic state on the secondary encoder inputs. See [MEIStatusMaskILLEGAL_STATE_SECONDARY](#).

 **Yes** The status bit is on, but there is no secondary encoder.

Stepper Pulse Lock Lost – Indicates that the pulse engine has lost track of controller cycle timing.

Stepper Pulse Bad – Indicates that a pulse was lost to the drive because of overlapping pulse widths.

DAC Level – DAC output level in volts.

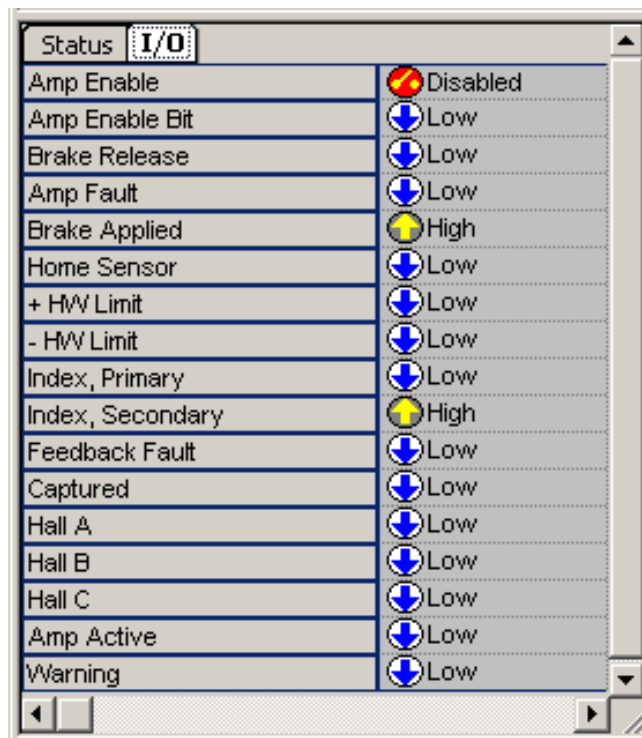
Aux DAC Level – Auxiliary DAC output level in volts.

Primary Feedback – Feedback position of the primary encoder. See [mpiMotorFeedback](#).

Secondary Feedback – Feedback position of the secondary encoder. See [mpiMotorFeedback](#).

Status Attributes: "I/O" Tab Page

This page displays Motor I/O status. For Dedicated I/O status.



Status	I/O
Amp Enable	Disabled
Amp Enable Bit	Low
Brake Release	Low
Amp Fault	Low
Brake Applied	High
Home Sensor	Low
+ HW Limit	Low
- HW Limit	Low
Index, Primary	Low
Index, Secondary	High
Feedback Fault	Low
Captured	Low
Hall A	Low
Hall B	Low
Hall C	Low
Amp Active	Low
Warning	Low

Amp Enable – Amp Enable output status.

Amp Enable Bit – Status of the Amp Enable dedicated motor output. See [MPIMotorDedicatedOutAMP_ENABLE](#).

Brake Release – Status of the Brake Release dedicated motor output. See [MPIMotorDedicatedOutBRAKE_RELEASE](#).

Amp Fault – Status of the Amp Fault dedicated motor input. See [MPIMotorDedicatedInAMP_FAULT](#).

Brake Applied – Status of the Brake Applied dedicated motor input. See [MPIMotorDedicatedInBRAKE_APPLIED](#).

Home Sensor – Status of the HOME dedicated motor input.

+ HW Limit – Status of the Positive Hardware Limit dedicated motor input. See [MPIMotorDedicatedInLIMIT_HW_POS](#).

– HW Limit – Status of the Negative Hardware Limit dedicated motor input. See [MPIMotorDedicatedInLIMIT_HW_NEG](#).

Index, Primary – Status of the Primary Index dedicated motor input. See [MPIMotorDedicatedInINDEX](#).

Index, Secondary – Status of the Secondary Index dedicated motor input. See [MPIMotorDedicatedInINDEX](#).

Feedback Fault – Status of the Feedback Fault dedicated motor input. See [MPIMotorDedicatedInFEEDBACK_FAULT](#).

Captured – Status of the Captured dedicated motor input. See [MPIMotorDedicatedInCAPTURED](#).

Hall A – Status of the Hall A dedicated motor input. See [MPIMotorDedicatedInHALL_A](#).

Hall B – Status of the Hall B dedicated motor input. See [MPIMotorDedicatedInHALL_B](#).

Hall C – Status of the Hall C dedicated motor input. See [MPIMotorDedicatedInHALL_C](#).

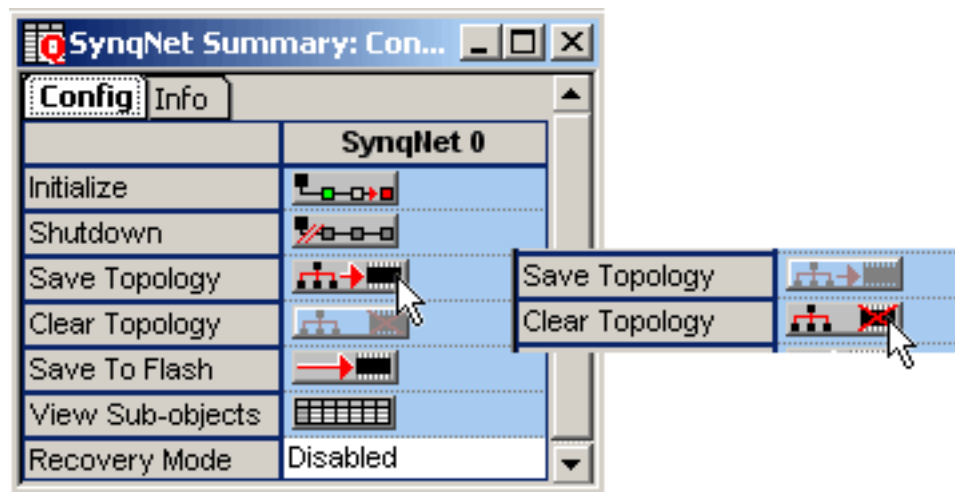
Amp Active – Status of the Amp Active dedicated motor input. See [MPIMotorDedicatedInAMP_ACTIVE](#).

Warning – Status of the Warning dedicated motor input. See [MPIMotorDedicatedInWARNING](#).

SynqNet Objects

SynqNet objects are the SynqNet networks that are supported by a motion controller. It represents the physical network. It contains information about the network state, number of nodes, and status.

Configuration Attributes: "Config" Tab Page



All SynqNet attributes are read-only in Motion Console.

Initialize – Press this button to bring up the SynqNet network. This will start the network initialization process. Please see the [Overview of Network Initialization](#) for details.

Shutdown – Press this button to perform an automatic shutdown of the SynqNet network. To bring up the network again, press the Initialize button.

Save Topology – Press this button to save the SynqNet topology information to the controller's flash memory.

NOTE: If you cannot click the Save Topology button, you must first click the Clear Topology button.

Clear Topology – Press this button to clear the saved SynqNet topology information from the controller's flash memory.

NOTE: If you cannot click the Clear Topology button, you must first click the Save Topology button.

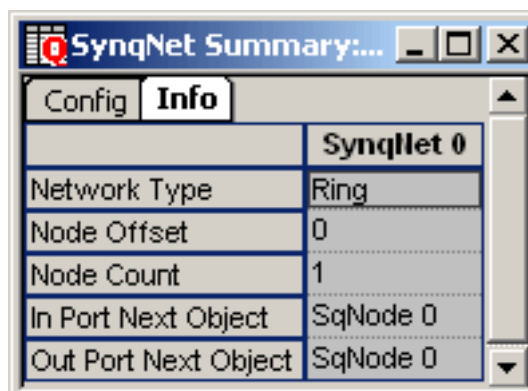
Save To Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for SynqNet object. For more information, see [Object Summary Windows](#).

Recovery Mode – Configures how the network is to respond to a fault condition. For more information, see [MEISynqNetRecoveryMode](#).

- **Disabled** – Recovery Mode is OFF. The network will not attempt to recover from a fault condition. This is the default mode for string topologies.
- **Single Shot** – The network will only attempt to recover from a fault condition one time. A second fault will be ignored.
- **Auto Arm** – The network will attempt to recover from a fault condition. After the fault recovery is complete, the network will automatically be re-armed to respond to another fault condition. This is the default mode for ring topologies.

Configuration Attributes: "Info" Tab Page



All Info attributes are read-only.

Network Type – The type of network topology discovered during network initialization (String, Ring, etc.). See [MEISynqNetInfo](#).

Node Offset – The starting number for the first node on the SynqNet network. For example, If SynqNet 0 has three nodes (Node 0-2), then the Node Offset for SynqNet 1 will be 3 because Node 3 is the next available node. See [MEISynqNetInfo](#).

Node Count – Number of Nodes on the SynqNet network. See [MEISynqNetInfo](#).

In Port Next Object – Name of the next network object connected to this object on the In Port. See [MEINetworkObjectInfo](#).

Out Port Next Object – Name of the next object connected to this object on the Out Port. See [MEINetworkObjectInfo](#).

Status Attributes: "Status" Tab Page

Status attributes are read-only. More information on events can be found under [MPIEventType](#) / [MEIEventType](#).

Status	
SynqNet 0	
State	SYNQ
Topology Saved	Yes
Dead Event	No
Rx Failure Event	No
Tx Failure Event	No
Node Failure Event	No
Recovery Event	No
CRC Err. IN 0	0
CRC Err. OUT 0	0
Failed Node Mask	0x00000000

State – The state of the SynqNet network. (DISCOVERY, ASYNQ, SYNQ, or SYNQ RECOVERING) See [MEISynqNetState](#) for more information.

Topology Saved – Whether or not the SynqNet topology has been saved.

Dead Event – The SynqNet network was shutdown due to a communication failure.

Rx Failure Event – Generated if the controller fails to receive packet data.

Tx Failure Event – Generated if the controller fails to transmit packet data.

Node Failure Event – Generated when any node's upstream or downstream packet error rate counters exceed the failure limit.

Recovery Event – Generated when data traffic is redirected around a faulty node.

CRC Err. IN 0 – The number of CRC errors that are received at the IN port. See [CRC Error Counters](#) for more information. See [MEISynqNetStatus](#).

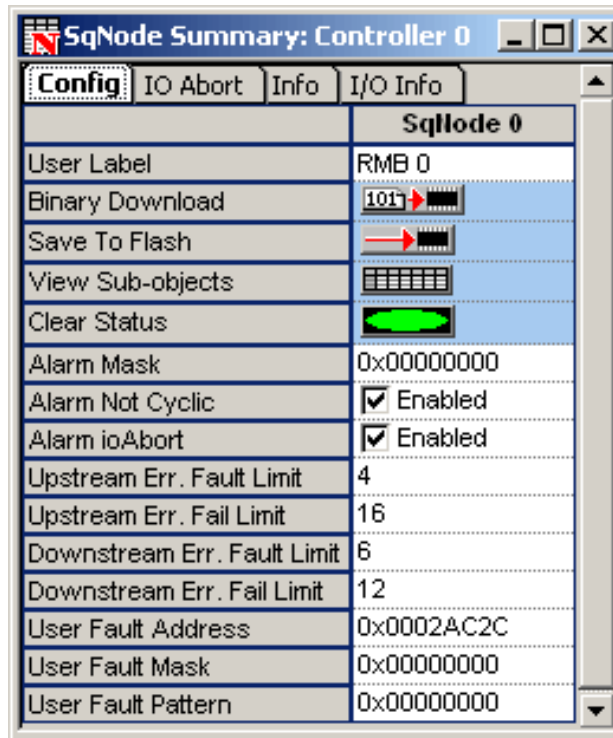
CRC Err. OUT 0 – The number of CRC errors that are received at the OUT port. See [CRC Error Counters](#) for more information. See [MEISynqNetStatus](#).

Failed Node Mask – Each bit in this mask represents a failed node (0x1 = node 0, 0x2 = node 2, 0x4 = node 3, etc.). See [MEISynqNetFailedNodeMask](#).

SqNode Objects

SqNode objects are the connected nodes on a SynqNet network. SynqNet supports up to 32 nodes supporting up to 32 motors on one SynqNet network.

Configuration Attributes: "Config" Tab Page



User Label – User-defined label for the object.

Binary Download – Downloads a binary file to the node which allows communication for normal operation between the controller and the node. Unless the Drive vendor has already installed the .sff file at the factory, an appropriate binary file will need to be downloaded. To determine which .sff file is appropriate for your node, please refer to the [Node Binary Files: Product Table](#). Multiple files may be downloaded to multiple nodes/drives. See Downloading Binary Files for more information.

Save to Flash – Pre-selects the current settings for saving to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for this object. For more information, see [Object Summary Windows](#).

Clear Status – Clears node CRC errors on all ports, clears node Packet errors, clears node ioAbort state, and resets SqNode events.

Alarm Configuration Attributes

The Alarm configuration attributes configure the trigger conditions for the Node Alarm output bit. See [MEISqNodeConfigAlarm](#) for more information.

Alarm Mask – One bit per drive/motor. Configures the node alarm output to trigger when the Amp Fault dedicated input of the motor is TRUE.

Alarm Not Cyclic – Specifies whether or not a node can receive an alarm when it is not in cyclic mode. When checked, a node alarm can be asserted in any mode. Otherwise, a node alarm can only be asserted in cyclic mode.

Alarm ioAbort – Specifies the effect an IO abort will have on the node alarm output. When checked, an I/O abort will trigger a node alarm.

Node Packet Error Rate Attributes

The following attributes specify the limit conditions for SynqNet node packet rate errors. See [MEISqNodeConfigPacketError](#) for more information.

Upstream Err. Fault Limit – Packet error rate limit on the IN port to generate a fault.

Upstream Err. Fail Limit – Packet error rate limit on the IN port to generate a failure.

Downstream Err. Fault Limit – Packet error rate limit on the OUT port to generate a fault.

Downstream Err. Fail Limit – Packet error rate limit on the OUT port to generate a failure.

User Fault Configuration Attributes

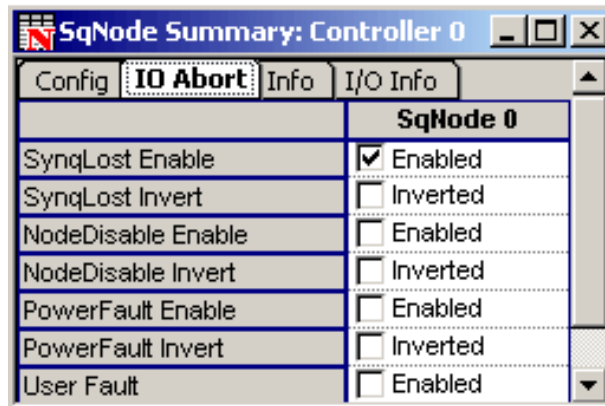
The User Fault configuration attributes specify the trigger conditions for a user fault. See [MEISqNodeConfigUserFault](#) for more information.

User Fault Address – The controller address to trigger off of.

User Fault Mask – A bit mask ANDed with the value at the controller address.

User Fault Pattern – A bit pattern compared to the masked value at the controller address. When the masked value equals the pattern, the user trigger is TRUE.

Configuration Attributes: "IO Abort" Tab Page



The I/O Abort configuration attributes specify the trigger conditions for a user fault. See [MEISqNodeConfigIoAbort](#) for more information.

SynqLost Enable – When checked, the controller will generate an I/O Abort Action when a SynqNet node drops out of SYNQ (cyclic) mode to SYNQ_LOST mode.

SynqLost Invert – When checked, the controller will invert the trigger polarity.

NodeDisable Enable - When checked, the controller will shutdown the node via IoAbort when the node disable is active.

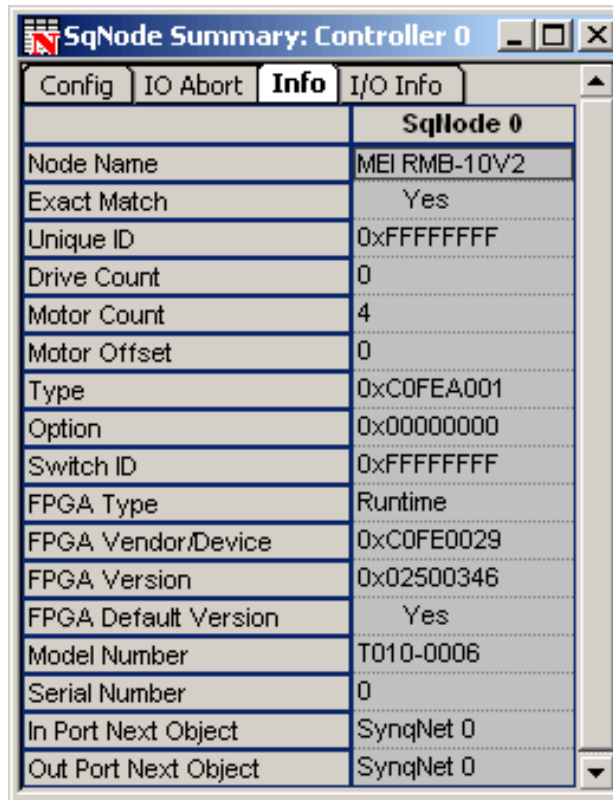
NodeDisable Invert – Inverts trigger polarity of the signal. When not checked, it will have normal polarity. If checked, it will have inverted polarity.

PowerFault Enable – An input bit to the SynqNet node. The power fault circuit is node-specific, but is usually connected to an analog power monitor. Typically, when the DAC power or other analog component power is either too high or drops below a threshold, the power fault is triggered.

PowerFault Invert – Inverts trigger polarity of the signal. When not checked, it will have normal polarity. If checked, it will have inverted polarity.

User Fault – The controller will generate an I/O Abort Action when the User Fault is triggered. The conditions for generating a User Fault are configured on the Config tab page.

Configuration Attributes: "Info" Tab Page



SqlNode 0	
Node Name	MEI RMB-10V2
Exact Match	Yes
Unique ID	0xFFFFFFFF
Drive Count	0
Motor Count	4
Motor Offset	0
Type	0xC0FEA001
Option	0x00000000
Switch ID	0xFFFFFFFF
FPGA Type	Runtime
FPGA Vendor/Device	0xC0FE0029
FPGA Version	0x02500346
FPGA Default Version	Yes
Model Number	T010-0006
Serial Number	0
In Port Next Object	SynqNet 0
Out Port Next Object	SynqNet 0

Node Name – A string that represents the SynqNet node type. See [MEISqNodeInfold](#).

Exact Match – "Yes" when all ID components have been matched to a supported configuration. See [MEISqNodeInfold](#).

Unique ID – The SynqNet node manufacturer determines this unique value to track a single product. See [MEISqNodeInfold](#).

Drive Count – The number of drives interfaces that the SynqNet node supports.

Motor Count – Number of motors mapped to the node.

Motor Offset – The starting number for the first motor on the SynqNet node. Example: If Node 0 has three motors (Motor 0-2), then the Motor Offset for Node 1 will be 3 because Motor 3 is the next available motor.

Type – This is a unique identification number that can be used to identify the type and manufacturer of the node.

Option – The product option code within a product series. See [MEISqNodeInfold](#).

Switch ID – If a node/drive has an address switch on its faceplate, Switch Id will be the value to which the switch is set. If an ID switch is not supported by the node, this value will be set to -1 (0xFFFFFFFF). See [MEISqNodeInfold](#).

FPGA Type – There are two FPGA Types that are used, Boot and Runtime. For more information about these FPGA types, please see the Binary Images section under [SynqNet Technology](#). See [MEISqNodeFpgaType](#).

Boot - The FPGA that is initially used when the SynqNet network is initialized for the first time. It enables the node to communicate with the controller. A runtime

image will need to be downloaded before any motors can move.

Runtime - The FPGA that is used for normal operation.

FPGA Vendor/Device – The unique identification number of the FPGA. For more information, please refer to the [Node Binary Files: Product Table](#). See [MEISqNodeInfoFpga](#).

FPGA Version – The version of the FPGA. See [MEISqNodeInfoFpga](#).

FPGA Default Version – Indicates if the default version of the SqNode FPGA image is loaded on this node. See [MEISqNodeInfoFpga](#).

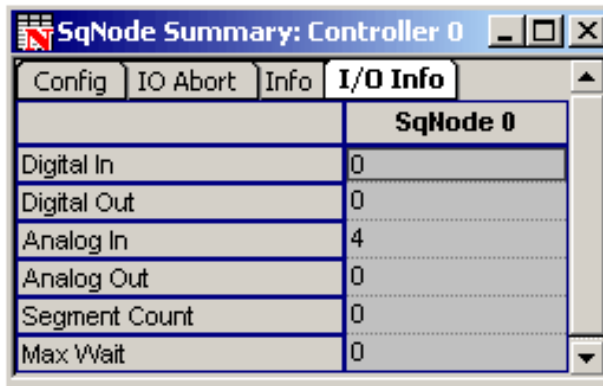
Model Number – The model number of the node (if available). See [MEISqNodeInfold](#).

Serial Number – The serial number of the node (if available). See [MEISqNodeInfold](#).

In Port Next Object – Name of the next network object connected to this object on the In Port.

Out Port Next Object – Name of the next object connected to this object on the Out Port.

Configuration Attributes: "Info" Tab Page



SqNode 0	
Digital In	0
Digital Out	0
Analog In	4
Analog Out	0
Segment Count	0
Max Wait	0

Digital In – The number of digital inputs on a SynqNet node. See [MEISqNodeInfo](#).

Digital Out – The number of digital outputs on a SynqNet node. See [MEISqNodeInfo](#).

Analog In – The number of analog inputs on a SynqNet node. See [MEISqNodeInfo](#).

Analog Out – The number of analog outputs on a SynqNet node. See [MEISqNodeInfo](#).

Segment Count – The total number of segments on a SynqNet node. See [MEISqNodeInfo](#).

Max Wait – This is the maximum amount of time between when the output bit is set in software and the hardware state takes effect. See [MEISqNodeInfo](#).

Status Attributes: "Status" Tab Page

SqNode status attributes display read-only data of each node. These are non-configurable.

Status	
SqlNode 0	
Upstream Err. Rate	0
Upstream Err. Count	0
Downstream Err. Rate	0
Downstream Err. Count	0
CRC Err. IN 0	0
CRC Err. OUT 0	0
IOAbort	No
Node Disable Input	<input checked="" type="checkbox"/> Yes
Node Alarm Output	No
Analog Power Fault	No
User Fault	No
Node Failure	No

Packet Errors

For more information on Packet Error Rates and Counters, see [MEISqNodeStatusPacketError](#).

Upstream Err. Rate – Number of upstream packet errors per cycle.

Upstream Err. Count – Number of upstream packet errors.

Downstream Err. Rate – Number of downstream packet errors per cycle.

Downstream Err. Count – Number of downstream packet errors.

CRC Errors

For more information on the CRC Error Counters, see the [CRC Error Counters](#) section.

CRC Err. IN 0 – CRC error counter for the IN 0 port.

CRC Err. OUT 0 – CRC error counter for the OUT 0 port.

Events

For more information concerning the following Events, see [MPIEventType](#) / [MEIEventType](#).

IO Abort – Generated when the node I/O Abort is activated. See [MEIEventTypeSQNODE_IO_ABORT](#).

Node Disable Input – Generated when the Node Disable input signal transitions from inactive to active. See [MEIEventTypeSQNODE_NODE_DISABLE](#).

Node Alarm Output – Generated when the node alarm output signal transitions from inactive to active. See [MEIEventTypeSQNODE_NODE_ALARM](#).

Analog Power Fault – Generated when the node's power failure input bit transitions from inactive to active. See [MEIEventTypeSQNODE_ANALOG_POWER_FAULT](#).

User Fault – Generated when the node's user configurable fault is triggered. See [MEIEventTypeSQNODE_USER_FAULT](#).

Node Failure – Generated when a node's upstream or downstream packet error rate counters exceed the failure limit. To recover from a node failure, the network must be shutdown and reinitialized. See [MEIEventTypeSQNODE_NODE_FAILURE](#).

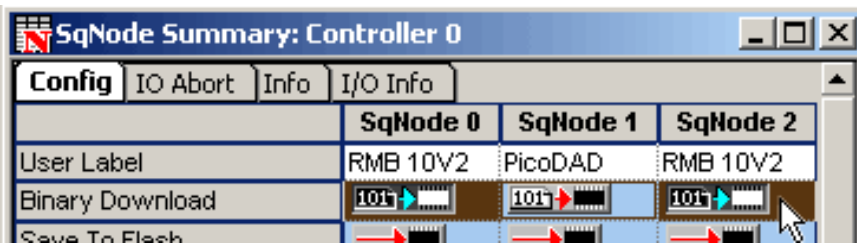
Downloading Binary Files

Follow the procedure below to download binary files to SynqNet nodes.

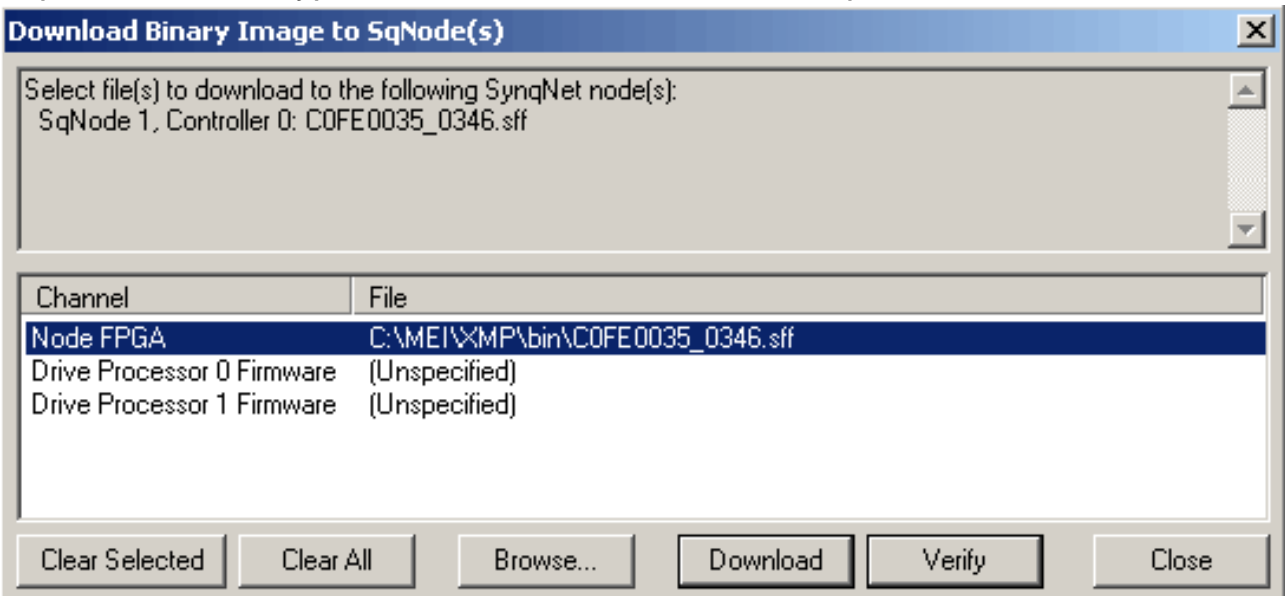
1. Click on the Binary Download button for the node on the SqNode Summary.



2. If you are downloading the same file(s) to multiple nodes, then you can select the nodes (Ctrl + click) and click the Binary Download button.
NOTE: All nodes in the selection must already have the same default FPGA file in order for the Binary Download button to be enabled.

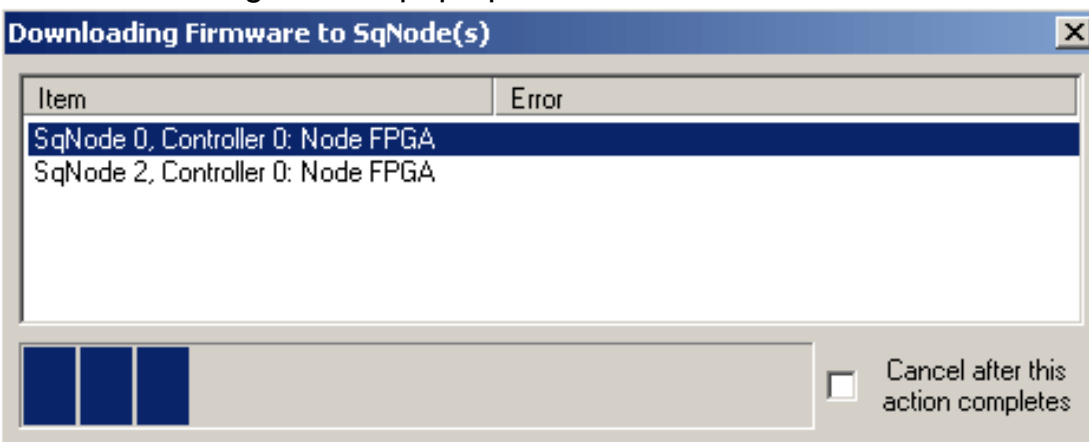


- After clicking the Binary Download button, the “Download Binary Image To SqNode(s)” dialog box pops up. In this dialog box, it is possible to select multiple files to download to the node. One FPGA file can be selected, as well as (possibly multiple) Drive Processor firmware files. The number of drive processors is dependent on the type of node. A node with two drive processors is shown below.



If the file is labeled “Unspecified”, then no file will be downloaded to that item. To specify a file for an item, click on the item to select it and then click on the **Browse** button. The default FPGA file will usually be pre-specified for the FPGA item. To “unspecify” a file for an item, select the item in the list and click the **Clear Selected** button. Click the **Clear All** button to “unspecify” a file for every item.

- After all the binary files are specified, click the **Download** button. The “Downloading Firmware” dialog box will pop up:



The item selected in the list indicates which item is currently being downloaded. The status bar at the bottom indicates the download progress for the current item. If there is no error for any item in the list, then the dialog box will close automatically.

Otherwise, an error message will be displayed in the Error column. The dialog box is left open to allow the user to read the error. Click on the “Close” button to exit the dialog box.

Downloading cannot be interrupted once it starts, but it is possible to cancel downloading after the current download is complete. To cancel the download, click on the “Cancel after this action completes” checkbox.

5. Once the download is complete and the “Downloading Firmware to SqNode(s)” dialog box is closed (either manually or automatically), close the “Download Binary Image To SqNode(s)” dialog box.

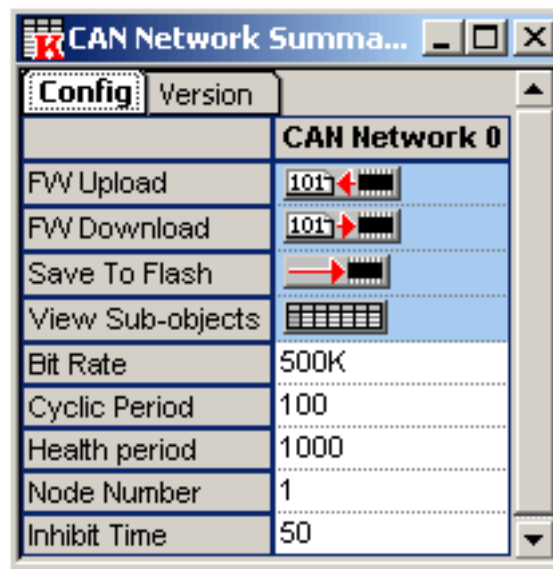
CAN Network Summary

CAN Configuration Attributes

The CAN Network Summary window lists CAN Network attributes in two tab pages: "Config" and "Version."

Configuration Attributes: "Config" Tab Page

The Config window displays the user configurable parameters of CAN.



FW Upload – allows the user to get a copy of the current CAN firmware.

FW Download – allows the user to upgrade the CAN firmware.

Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for CAN Network object. For more information, see [Object Summary Windows](#).

Bit Rate – The bit rate that the CAN bus uses. See [MEICanConfig](#). See

also [CAN Bit Rate](#).

Cyclic Period – the period between sending consecutive SYNC messages (milliseconds). A value of zero will disable the SYNC messages from being produced. See [MEICanConfig](#).

Health Period – the period (milliseconds) used for checking the health of nodes.

A value of zero will disable the health checking protocol.

For nodes that use the node guarding protocol, this is the node guarding period.

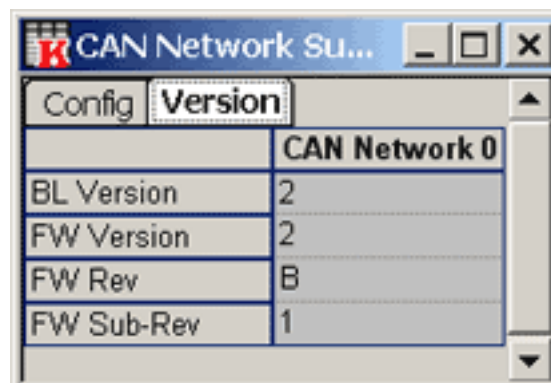
For nodes that use the heartbeating protocol, this is the heartbeat consumer time (the heartbeat producers are half this period). See [MEICanConfig](#).

Node Number – the node number of the XMP on the CAN network. See [MEICanConfig](#).

Inhibit Time – this coefficient defines the minimum time between two successive PDO messages. See [MEICanConfig](#).

Configuration Attributes: "Version" Tab Page

The Version window is not user-configurable.



The firmware version in the screenshot above is **002B1**.

BL Version – the version number of the CAN bootloader. See

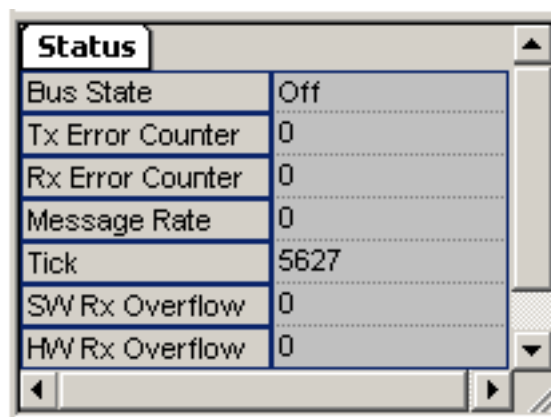
[MEICanVersion](#).

FW Version – the CAN firmware version number. See [MEICanVersion](#).

FW Rev – the CAN firmware revision number. See [MEICanVersion](#).

FW Sub-Rev – the CAN firmware sub-revision number. See [MEICanVersion](#).

Status Attributes: "Status" Tab Page



Status	
Bus State	Off
Tx Error Counter	0
Rx Error Counter	0
Message Rate	0
Tick	5627
SW Rx Overflow	0
HW Rx Overflow	0

Bus State – the CAN bus will be in one of the following states: Off, Operational, or Passive. See [MEICanStatus](#).

Tx Error Counter – the current value of the transmit error counter. See [MEICanStatus](#).

Rx Error Counter – the current value of the receive error counter. See [MEICanStatus](#).

Message Rate – allows the user to specify the Node Guard and Heartbeat times for the health protocols. See [MEICanStatus](#).

Tick – a Counter that is incremented every 1ms. See [MEICanStatus](#).

SW Rx Overflow – A flag indicating that the software receive buffer on

the XMP has overflowed. 0 indicates that the buffer has never overflowed and 1 indicates that the buffer has overflowed. See [MEICanStatus](#).

HW Rx Overflow – A flag indicating that the hardware receive buffer on the XMP has overflowed. 0 indicates that the buffer has never overflowed and 1 indicates that the buffer has overflowed. See [MEICanStatus](#).

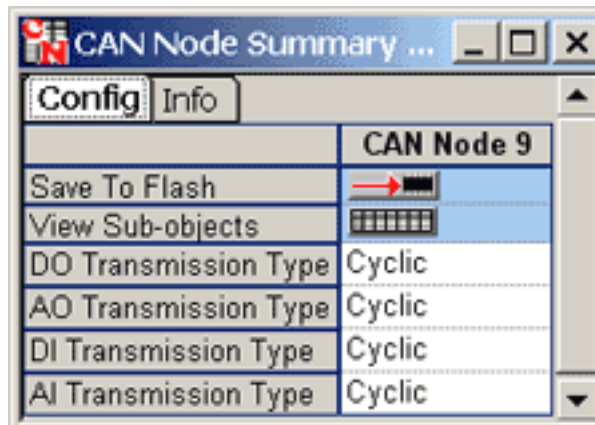
CAN Node Summary

CAN Node Configuration Attributes

The CAN Node Summary window lists CAN Node attributes in two tab pages: "Config" and "Info."

Configuration Attributes: "Config" Tab Page

The Config window displays the user configurable parameters of CAN.



Save to Flash – Pre-selects the current settings and saves them to flash memory. For more information, please see the [Save to Flash](#) section.

View Sub-objects – Shows all sub-objects for CAN Node object. For more information, see [Object Summary Windows](#).

DO Transmission Type – the current state of the digital output bit on the specified CAN node. See [MEICanNodeConfig](#).

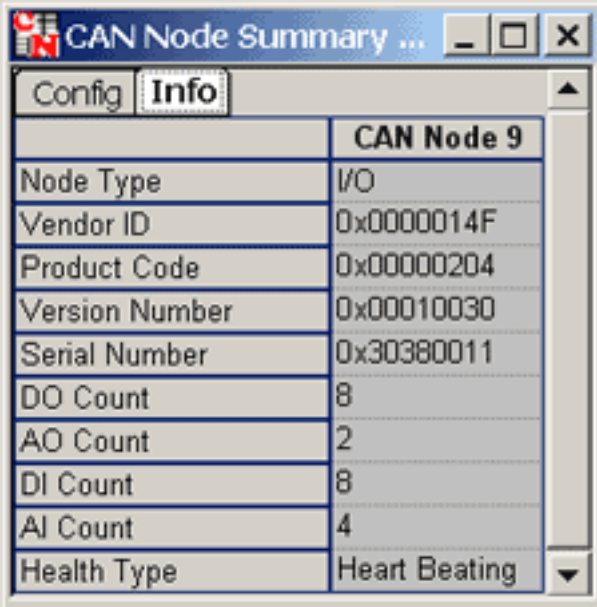
AO Transmission Type – the current state of the analog output bit on the specified CAN node. See [MEICanNodeConfig](#).

DI Transmission Type – the current state of the digital input bit on the specified CAN node. See [MEICanNodeConfig](#).

AI Transmission Type – the current state of the analog input bit on the specified CAN node. See [MEICanNodeConfig](#).

Configuration Attributes: "Info" Tab Page

The Info window is not user-configurable.



CAN Node 9	
Node Type	I/O
Vendor ID	0x0000014F
Product Code	0x00000204
Version Number	0x00010030
Serial Number	0x30380011
DO Count	8
AO Count	2
DI Count	8
AI Count	4
Health Type	Heart Beating

Node Type – Indicates the type of CANOpen node: I/O or None.

Vendor ID – Vendor ID numbers are unique numbers allocated to each manufacturer of CANOpen nodes. Not all CANOpen nodes support this feature, in which case, these nodes will return zero for this field. MEI CANOpen nodes always return x014Fh. See [MEICanNodeInfo](#).

Product Code – The product code is made up of numbers allocated by each manufacturer to uniquely identify their different types of nodes. Not all CANOpen nodes support this feature, in which case, these nodes will return zero for this field. MEI CANOpen SLICE nodes always return x0204h. See [MEICanNodeInfo](#).

Version Number – Identifies the version of code running on the CANOpen node. See [MEICanNodeInfo](#).

Serial Number – The serial number of the CANOpen node (if available). See [MEICanNodeInfo](#).

DO Count – the number of digital outputs supported on this node. The CANOpen protocol only allows the number of digital inputs and outputs to be interrogated in multiples of eight, i.e. if a node has one digital output the "digitalOutputCount" will return eight.

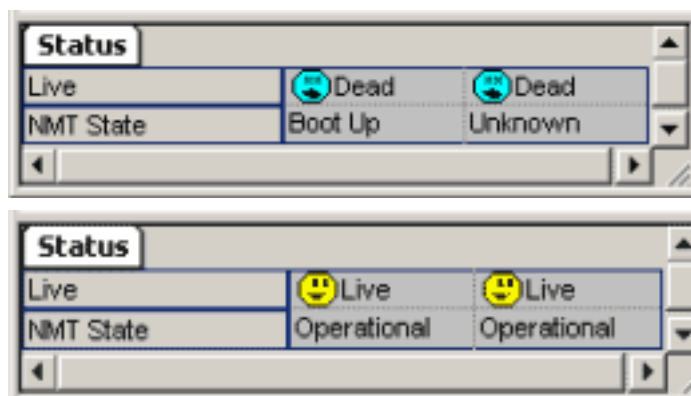
AO Count – the number of analog outputs supported on this node.

DI Count – the number of digital inputs supported on this node. The CANOpen protocol only allows the number of digital inputs and outputs to be interrogated in multiples of eight, i.e. if a node has one digital output the "digitalOutputCount" will return eight.

AI Count – the number of analog inputs supported on this node.

Health Type – The CANOpen protocol being used to check the health of this node. The health type is either Node Guarding or Heart Beating. See [MEICanHealthType](#).

Status Attributes: "Status" Tab Page



Live – the system will either be "live" or "dead."

NMT State – CANOpen protocol Network Management state. See

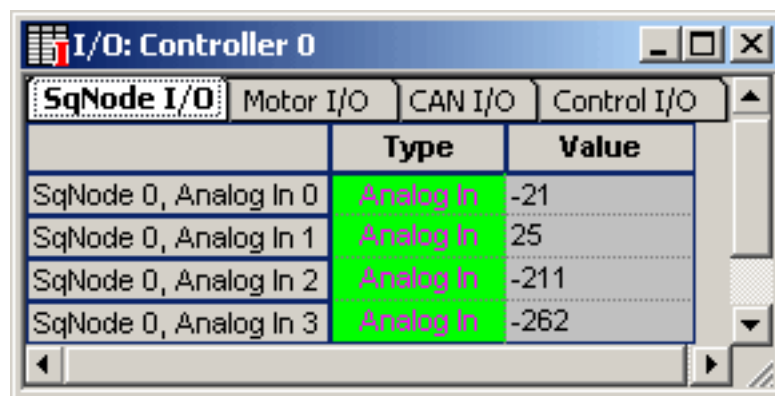
[MEICanNMTState.](#)

I/O Summary

The I/O Summary is different from other Summary windows in two ways. First, the type of I/O objects that can be displayed are broken out into sub-types. Each sub-type is displayed in its own tab page. For example, there are tab pages for CAN I/O and Motor I/O. Secondly, unlike other Summary windows, objects are represented as rows and attributes are represented in columns.

Configuration Attributes: "SqNode I/O" Tab Page

This window displays a breakdown for SynqNet I/O. See [MEISqNodeInfoLo](#).



	Type	Value
SqNode 0, Analog In 0	Analog In	-211
SqNode 0, Analog In 1	Analog In	25
SqNode 0, Analog In 2	Analog In	-211
SqNode 0, Analog In 3	Analog In	-262

Type – The SqNode I/O objects are sorted by type. Each type is color-coded. The table below describes the I/O types.

- Digital Output (DO) – red
- Analog Output (AO) – blue
- Digital Input (DI) – yellow
- Analog Input (AI) – green

Value – The Value attribute for an Input I/O object is read-only. For Digital I/O, the value is represented as a checkbox. Checked is HIGH, unchecked is LOW. The analog value is a 16-bit number that contains the raw bits going to/from the SynqNet node. The relationship between the 16-bit number and the analog value is described with the

documentation for each analog module or slice. For example, the analog data for the ADC4DAC4 module has the following relationship:

7FFFh	+10V
8000h	–10V

For more information, please see [Accessing Analog Data](#).

Configuration Attributes: "Motor I/O" Tab Page

This window displays the I/O associated with motor objects.

I/O: Controller 0			
SqNode I/O	Motor I/O	CAN I/O	Control I/O
	Type	Out	In
Motor 0, XCVR A	Input	<input type="checkbox"/>	High
Motor 0, XCVR B	Input	<input type="checkbox"/>	High
Motor 0, XCVR C	Input	<input type="checkbox"/>	High
Motor 0, XCVR D	Input	<input type="checkbox"/>	Low
Motor 0, XCVR E	Input	<input type="checkbox"/>	High
Motor 0, XCVR F	Input	<input type="checkbox"/>	Low
Motor 0, User In 0	Input	<input type="checkbox"/>	High
Motor 0, User Out 0	Output	<input type="checkbox"/>	High
Motor 1, XCVR A	Input	<input type="checkbox"/>	High
Motor 1, XCVR B	Input	<input type="checkbox"/>	High
Motor 1, XCVR C	Input	<input type="checkbox"/>	High
Motor 1, XCVR D	Input	<input type="checkbox"/>	Low
Motor 1, XCVR E	Input	<input type="checkbox"/>	Low
Motor 1, XCVR F	Input	<input type="checkbox"/>	Low
Motor 1, User In 0	Input	<input type="checkbox"/>	High
Motor 1, User Out 0	Output	<input type="checkbox"/>	High
Motor 2, XCVR A	Input	<input type="checkbox"/>	High
Motor 2, XCVR B	Input	<input type="checkbox"/>	High
Motor 2, XCVR C	Input	<input type="checkbox"/>	High
Motor 2, XCVR D	Input	<input type="checkbox"/>	Low
Motor 2, XCVR E	Input	<input type="checkbox"/>	Low
Motor 2, XCVR F	Input	<input type="checkbox"/>	Low
Motor 2, User In 0	Input	<input type="checkbox"/>	High
Motor 2, User Out 0	Output	<input type="checkbox"/>	High
Motor 3, XCVR A	Input	<input type="checkbox"/>	High
Motor 3, XCVR B	Input	<input type="checkbox"/>	High
Motor 3, XCVR C	Input	<input type="checkbox"/>	High
Motor 3, XCVR D	Input	<input type="checkbox"/>	Low
Motor 3, XCVR E	Input	<input type="checkbox"/>	Low
Motor 3, XCVR F	Input	<input type="checkbox"/>	Low
Motor 3, User In 0	Input	<input type="checkbox"/>	High
Motor 3, User Out 0	Output	<input type="checkbox"/>	High

The number of I/O objects associated with a motor depends on the type of SynqNet Node that the motor is mapped to. If the motor is not mapped to a SynqNet Node, then it has no I/O. See [Motor I/O](#) for more information.

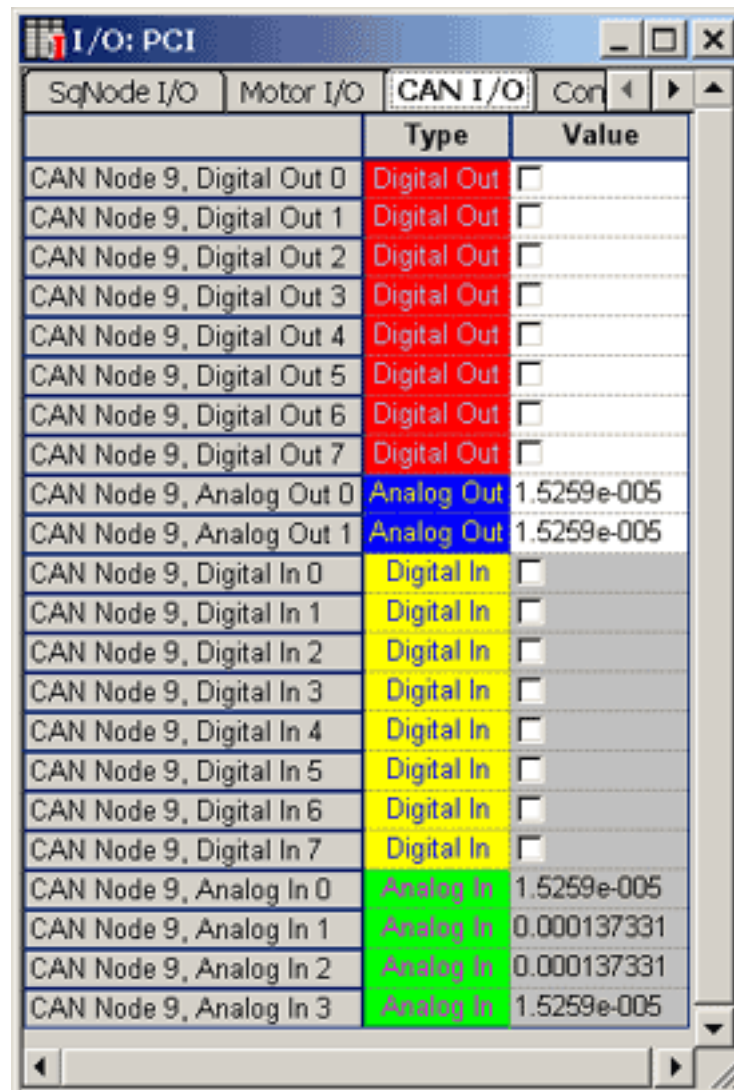
Type – The choices for the types of I/O depend on the type of SynqNet Node that the motor is mapped to.

Out – Value to write to the bit. Checked is HIGH, unchecked is LOW.

In – Value read from the bit.

Configuration Attributes: "CAN I/O" Tab Page

This window displays a breakdown of the I/O for a CAN system.



	Type	Value
CAN Node 9, Digital Out 0	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 1	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 2	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 3	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 4	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 5	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 6	Digital Out	<input type="checkbox"/>
CAN Node 9, Digital Out 7	Digital Out	<input type="checkbox"/>
CAN Node 9, Analog Out 0	Analog Out	1.5259e-005
CAN Node 9, Analog Out 1	Analog Out	1.5259e-005
CAN Node 9, Digital In 0	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 1	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 2	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 3	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 4	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 5	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 6	Digital In	<input type="checkbox"/>
CAN Node 9, Digital In 7	Digital In	<input type="checkbox"/>
CAN Node 9, Analog In 0	Analog In	1.5259e-005
CAN Node 9, Analog In 1	Analog In	0.000137331
CAN Node 9, Analog In 2	Analog In	0.000137331
CAN Node 9, Analog In 3	Analog In	1.5259e-005

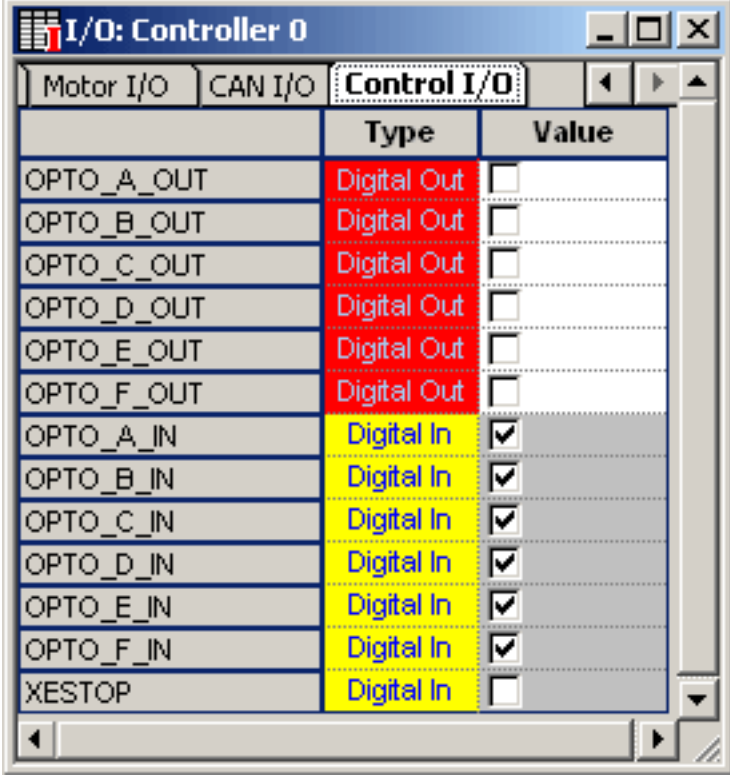
Type – The CAN I/O objects are sorted by type. Each type is color-coded. The table below describes the I/O types.

- Digital Out (red) – Digital Output
- Analog Out (blue) – Analog Output
- Digital In (yellow) – Digital Input
- Analog In (green) – Analog Input

Value – The Value attribute for an Input I/O object is read-only. For Digital I/O, the value is represented as a checkbox. Checked is HIGH, unchecked is LOW. For Analog I/O, the value is a floating point number normalized to be between -1 and 1.

Configuration Attributes: "Control I/O" Tab Page

This window displays the I/O associated with control objects.



	Type	Value
OPTO_A_OUT	Digital Out	<input type="checkbox"/>
OPTO_B_OUT	Digital Out	<input type="checkbox"/>
OPTO_C_OUT	Digital Out	<input type="checkbox"/>
OPTO_D_OUT	Digital Out	<input type="checkbox"/>
OPTO_E_OUT	Digital Out	<input type="checkbox"/>
OPTO_F_OUT	Digital Out	<input type="checkbox"/>
OPTO_A_IN	Digital In	<input checked="" type="checkbox"/>
OPTO_B_IN	Digital In	<input checked="" type="checkbox"/>
OPTO_C_IN	Digital In	<input checked="" type="checkbox"/>
OPTO_D_IN	Digital In	<input checked="" type="checkbox"/>
OPTO_E_IN	Digital In	<input checked="" type="checkbox"/>
OPTO_F_IN	Digital In	<input checked="" type="checkbox"/>
XESTOP	Digital In	<input type="checkbox"/>

Type – The Control I/O are sorted by type. Each type is color coded. The list below describes the I/O types.

- Digital Out (red) – Digital Output
- Digital In (yellow) – Digital Input

Value – The Value attribute for an Input I/O object is read-only. For Digital I/O, the value is represented as a checkbox. When checked, it's HIGH. When unchecked, it's LOW.

Command Line Options

`[-u updateRate][-l languageFile][-s showSplashScreen][-p profile]`

-u updateRate	Refresh rate in milliseconds.
-l languageFile	Alternate resource file. Used for changing text.
-s showSplashScreen	0 - To disable the display of the splash screen. 1 - To display it (default).
-p profile	Specify an alternate default profile.

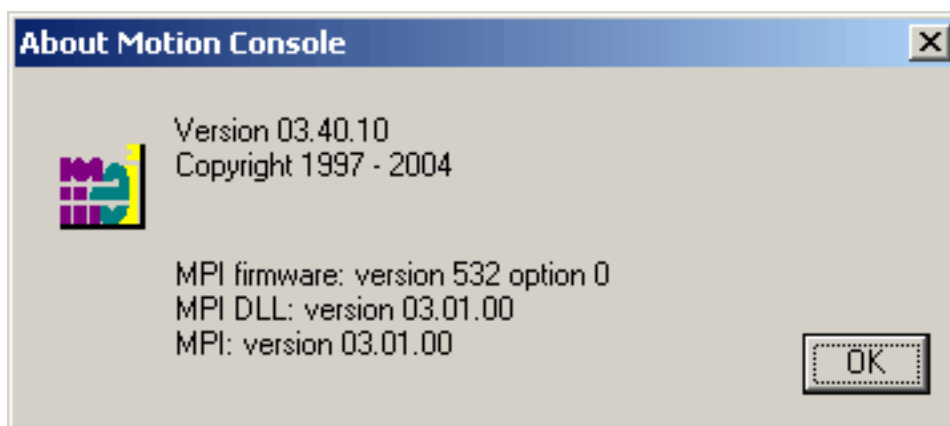
NOTE: Specifies a file to use for Motion Console's profile (i.e. how Motion Console displays windows and what positions they will first appear in). By default, this will be **MC_XMP_NT.ini**. A profile may also be saved from within Motion Console by selecting "File - > Save Profile As" from the menu or by using the shortcut key **ctrl-S**

Reading Versions With Motion Console

Motion Console can read version information from the controller, MPI, and drives.

To access the MPI version, firmware version, and Motion Console version, click Motion Console > Help > About Motion Console...

The About Motion Console window that pops up has several versions in it:

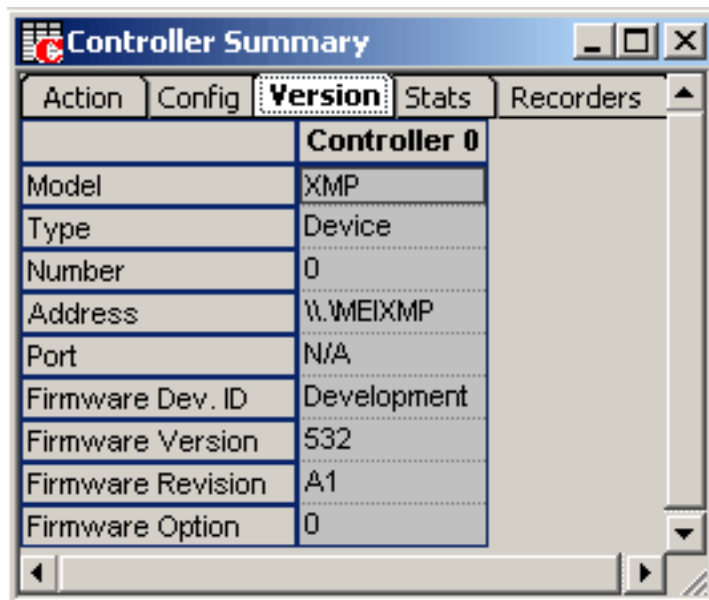


Using the screen shot above as an example:

- **Version** - is the version of Motion Console. (ex: 03.40.10)
- **MPI firmware**:- is the version of firmware that runs on the controller. (ex: version 532 option 0)
- **MPI DLL**: - is the version of the MPI that is being used. (ex: version 03.01.00)
- **MPI**: - is the version of MPI that Motion Console expects to see. (ex: version 03.01.00)

Reading The Firmware Version With The Controller Summary

The version of firmware on the controller can be read in Motion Console -> Controller Summary -> Version as is shown in the following screen shot.



The firmware version, revision, and option are shown in the Firmware Version, Firmware Revision, and Firmware Option cells, respectively.

For non zero Firmware Option versions, replace the M in XMP with the option number. The default firmware version (zero) as shown above would be called XMP532A1.bin. If the Firmware Option number was 1 the default firmware version would be called X1P532A1.bin. For ZMP firmware, replace XMP with ZMP.

Reading The Node FPGA Versions With the SynqNet Node Summary Window

The Node FPGA versions can be read in Motion Console > SqNode Summary > Info. The FPGA version is shown in the FPGA Version cell. The FPGA runtime version is included in the last four digits.

SqNode Summary: Controller 0		
Config IO Abort Info		
	SqNode 0	SqNode 1
Node Name	Trust TA801-D	MEI RMB-10V2
Exact Match	Yes	Yes
Unique ID	0x0000001E	0x00067CC7
Drive Count	0	0
Motor Count	1	4
Motor Offset	0	1
Type	0x00090001	0xC0FEA001
Option	0x00000000	0x00000000
Switch ID	0xFFFFFFFF	0xFFFFFFFF
FPGA Type	Runtime	Runtime
FPGA Vendor/Device	0xC0FE002D	0xC0FE0029
FPGA Version	0x02200311	0x02200311
FPGA Default Version	Yes	Yes
Model Number	TA801-D	T010-0004
Serial Number	TA801D01P314...	425159

Status		
	SqNode 0	SqNode 1
Upstream Err. Rate	0	0
Upstream Err. Count	0	0
Downstream Err. Rate	0	0
Downstream Err. Count	0	0
CRC Err. IN 0	0	0
CRC Err. OUT 0	0	0
IOAbort	No	No
Node Disable Input	No	No
Node Alarm Output	No	No
Analog Power Fault	No	No
User Fault	No	No

For example, the runtime version of both FPGA images shown above is 0311. The node device ID is shown in the last four digits of the FPGA Vendor/Device cell. SqNode 0, above, has a node device ID of 002D. When looking for a node FPGA file, the node device ID and FPGA runtime version are combined in the format C0FExxxx_yyyy.sff, where xxxx is the node device ID and yyyy is the FPGA runtime version. For example, the FPGA file for node 0, above, is C0FE002D_0311.sff.

Troubleshooting

Please see our [troubleshooting video tutorials](#) section on MEI's Technical Support site.