

Automation Fair 2009


Lab 14 – Introduction to Logix



Commercial Engineering - Control Platforms

For Classroom Use Only!

LISTEN.
THINK.
SOLVE.®



 Allen-Bradley • Rockwell Software

**Rockwell
Automation**

Copyright © 2009 Rockwell Automation, Inc.

Contents

About This Hands-On Lab	5
Lab Materials	5
Document Conventions	7
About Logix Controllers.....	8
ControlLogix: Perfect for high-speed, high-performance, multidiscipline control.....	8
CompactLogix: Perfect for smaller, machine-level control applications.....	9
About This Lab	10
Launching RSLogix 5000 Configuration Software	10
Creating a New Controller Project.....	12
Adding Ladder Logic to the Main Routine.....	14
Creating Tags for the Ladder Code.....	21
Monitoring/Editing Tags	29
Lab 2: Configuring I/O (20 minutes).....	33
About this Lab	33
Adding the AB_ETHIP-1 (Ethernet/IP) Driver	55
Lab 4: Downloading the Project from the Computer to the Controller (10 minutes).....	57
About This Lab	57
Downloading the Project to the Controller	58
Lab 5: Testing Your Logic Program (5 Minutes)	61
About This Lab	61
Switching the Controller into Run Mode and Testing the Program.....	61
Lab 6: Adding Logic and Tags Online (15 Minutes)	65

About This Lab	65
Adding the Timer to the Logic	65
Testing Your Logic.....	70
Lab 7: Creating and Running a Trend (5 Minutes)	71
About This Lab	71
Creating and Running a Trend	72
Lab 8: Using RSLogix 5000 Help (15 minutes).....	77
About This Lab	77
Instruction Help.....	77
Viewing I/O Module Wiring Diagrams	79
Using Start Page	80
Learning Center 	81
Resource Center 	82

Before you begin

About This Hands-On Lab

This session provides you with an opportunity to explore the ControlLogix or CompactLogix platforms, depending on the station at which you find yourself seated. The following sections explain what you'll be doing in this lab session, and what you will need to do to complete the hands-on exercises.

What You Will Accomplish In This Lab

As you complete the exercises in this hands-on session, you will:

1. learn the primary advantages of Logix based controllers
2. design, create and download programs to a Logix controller
3. examine a controller executing a program

Who Should Complete This Lab

This hands-on lab is intended for:

Controller users who want to become familiar and comfortable with the basics of RSLogix 5000 programming software.

Lab Materials

For this Hands-On lab, we have provided you with the following materials that will allow you to complete the labs in this workbook.

Hardware

This hands-on lab requires one of the following Demo boxes:

4. ControlLogix Demobox



This is your hardware.

5. CompactLogix demobox (either L43 or L35E)



Software

This hands-on lab uses the following software:

6. RSLogix 5000 programming software
7. RSLinx Classic software

Files

There are no starting project files for this lab; you will be creating your own file as you go.

Document Conventions

Throughout this workbook, we have used the following conventions to help guide you through the lab materials.

This style or symbol:	Indicates:
Words shown in bold italics (e.g., <i>RSLogix 5000</i> or <i>OK</i>)	Any item or button that you must click on, or a menu name from which you must choose an option or command. This will be an actual name of an item that you see on your screen or in an example.
Words shown in Courier text, enclosed in single quotes (e.g., ' <i>Controller1</i> ')	An item that you must type in the specified field. This is information that you must supply based on your application (e.g., a variable). Note: When you type the text in the field, remember that you do not need to type the quotes; simply type the words that are contained within them (e.g., Controller1).
<i>FYI</i>	The text that follows this symbol is supplemental information regarding the lab materials, but not information that is required reading in order for you to complete the lab exercises. The text that follows this symbol may provide you with helpful hints that can make it easier for you to use this product.

Note: If the mouse button is not specified in the text, you should click on the left mouse button.

About Logix Controllers

ControlLogix: Perfect for high-speed, high-performance, multidiscipline control

ControlLogix brings together the benefits of the Logix platform — common programming environment, common networks, common control engine — to provide the high-performance your application requires in an easy-to-use environment. Tight integration between the programming software, controller and I/O reduces development time and cost at commissioning and during normal operation.

ControlLogix offers the following benefits:

- Premier high-speed, high-performance control platform for multidiscipline control (sequential, process, drive, and motion)
- Fully-redundant controller architecture provides bumpless switchover and high availability
- Widest range of communication options and analog, digital and specialty I/O.
- Select ControlLogix products are TUV-certified for use in SIL 2 applications

With memory options ranging up to 32MB, ControlLogix controllers support intensive process applications and provide fast processing of motion instructions in a single integrated solution.

ControlLogix provides modular network communications that let you purchase only what you need. Interface using ControlLogix communication modules via a ControlLogix gateway, without the need for a processor in the gateway chassis, or interface directly to a ControlLogix controller.

The ControlLogix solution also provides time synchronization capabilities, which is particularly useful in first fault and process sequencing applications.

CompactLogix: Perfect for smaller, machine-level control applications

CompactLogix brings together the benefits of the Logix platform — common programming environment, common networks, common control engine — in a small footprint with high performance. Combined with Compact I/O, the CompactLogix™ platform is perfect for tackling smaller, machine-level control applications, with or without integrated motion, with unprecedented power and scalability. CompactLogix is ideal for systems that require standalone and system-connected control over EtherNet/IP, ControlNet, or DeviceNet. Think CompactLogix when you need economical, reliable control.

CompactLogix offers the following benefits:

- rackless I/O for flexible installation
- high functionality in an economical platform
- analog, digital and specialty modules cover a wide range of applications
- advanced system connectivity to EtherNet/IP, ControlNet, and DeviceNet Networks
- truly integrated motion control capability

With a user memory ranging from 512K to 3Mb, CompactLogix controllers offer integrated serial (integrated RS-232-C ports for SCADA, ASCII, or peer-to-peer communication), EtherNet/IP or ControlNet channels, modular DeviceNet communications and local I/O capacity that can range from 8 to 30 I/O modules.

Use CompactLogix for small- to medium-size axes solutions. Typically, these applications are machine-level control applications with motion axes, I/O requirements and network connectivity requirements. Using the CompactLogix 1768-L43 or 1768-L45 controllers with the 1768-M04SE or 1768-M08SE SERCOS adapter module for motion control of SERCOS drives provides a truly integrated motion control solution at an economical price. The 1768-L3x controllers offer a built in Ethernet port, you can also add an optional 1768-ENBT communication module for L4x controllers for EtherNet/IP communications for plant-wide control.

About This Lab


In this lab, we will introduce you to the Logix product family. In this lab you will:

8. Create a new project
9. Write ladder logic
10. Use symbolic tag names
11. Use the tag monitor/editor

Launching RSLogix 5000 Configuration Software

In this section of the lab, you will launch the RSLogix 5000 software, which will allow you to configure and program a controller.

1. Follow the instructions in the **Before You Begin** section of this document before proceeding.

2. Double-click on the **RSLogix 5000**  icon on the Desktop to launch RSLogix 5000 software.

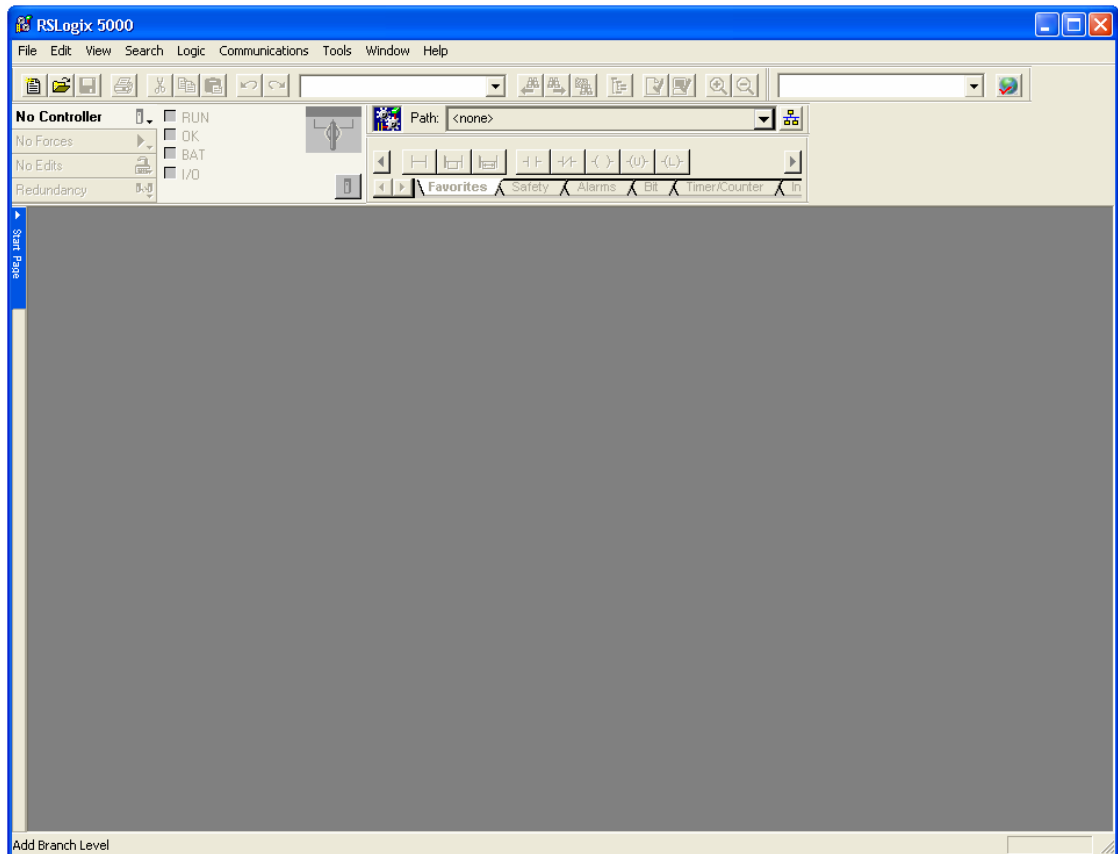
The RSLogix 5000 Splash Screen appears.

FYI

At the bottom of the RSLogix 5000 splash screen you will see all the different versions of RSLogix 5000 software that are currently installed on the computer.

Next, the RSLogix 5000 screen appears.

3. Click the arrow on the Start Pages label to hide the start pages – we will discuss these later in the lab.



Creating a New Controller Project

In this portion of the lab, you will create an offline project using a CompactLogix 1769-L35E controller.

1. From the **File** menu, choose **New**.

The New Controller dialog Box appears.

The screenshot shows the 'New Controller' dialog box. The 'Type' dropdown menu is highlighted with a red circle, showing '1769-L35E CompactLogix5335E Controller'. Other fields include 'Vendor: Allen-Bradley', 'Revision: 17', 'Name: Controller1', 'Create In: C:\RSLogix 5000\Projects', and buttons for 'OK', 'Cancel', 'Help', and 'Browse...'.

2. Verify that your entries match those shown based on the lab equipment at your station and then click on **OK**.

Important Note! All the Logix controllers use RSLogix 5000 software. Be sure to choose the correct controller type that matches the equipment at your lab station. If you are unsure of the equipment at your station, refer to the pictures at the beginning of the lab to verify your hardware. The Controllers have revision 17 firmware.

FYI

New Controller

From the **New Controller** window you are defining the project.

Type: This is the type of Logix controller you will use. This could be a ControlLogix, CompactLogix, DriveLogix, or SoftLogix controller. Only one programming software package is need for all Logix Controllers.

Revision: Here you are selecting the firmware revision that the project will be created with. Lab computer's include revision 17.

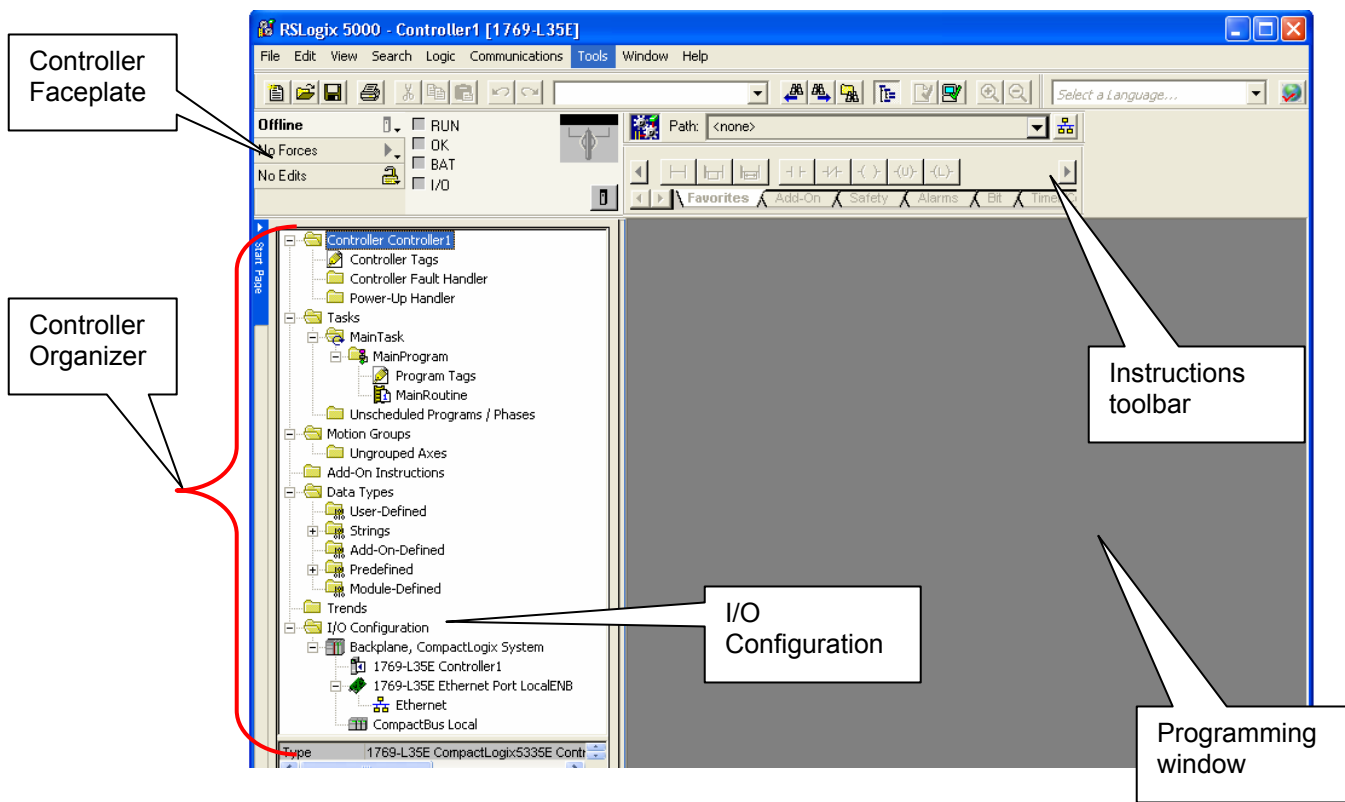
Name: The name of the controller and project.

Chassis Type: Select the size of the chassis you will use. This is not applicable for all controller types.

Slot: The slot number that you wish the controller to be installed in. This is not applicable for all controller types, In this case, CompactLogix is fixed at slot zero.

The **Controller Organizer** appears on the left side of the RSLogix5000 window, with a folder called **Controller Controller1**.

At this time, there is no I/O, tag database, or logic associated with the controller.



You have now created your first controller project!

FYI

The **Controller Organizer** is a graphical representation of the contents of your controller file. This display consists of a tree of folders and files that contain all of the information about the programs and data in the current controller file. The default main folders in this tree are:

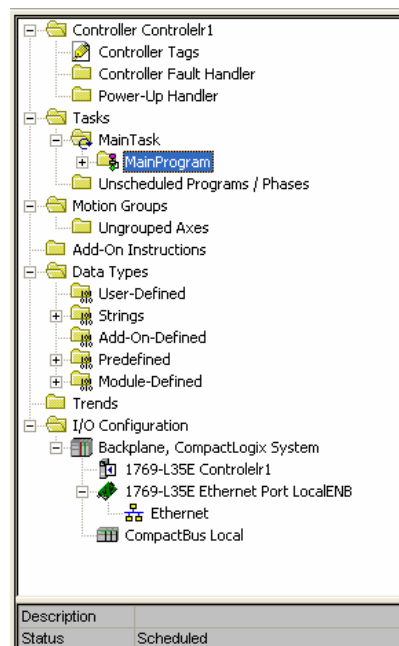
- Controller File Name
- Tasks
- Motion Groups
- Add-On Instructions
- Data Types
- Trends
- I/O Configuration
- The square containing a '+' or '-' indicates whether a folder is open or closed. Click on it to expand the tree display and display the files in the folder. The - sign indicates that the folder is already open and its contents are visible. By default the Add-On instructions folder is empty as none are installed by default.

Adding Ladder Logic to the Main Routine

In this lab you will add code for a simple motor start/stop seal-in circuit. You will experience the ease of programming with RSLogix 5000 software. During the labs we will only utilize ladder logic programming, but Logix controllers also can be programmed using Function Block, Sequential Function Charts, and Structured Text. This allows you to select the program language that best fits an application.

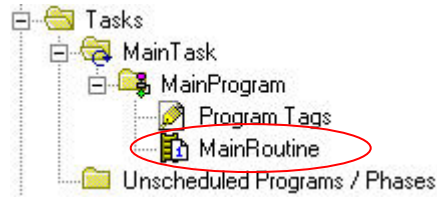
You will continue to use the project already opened.

1. In the Controller Organizer expand the **MainProgram** folder by clicking on the +.

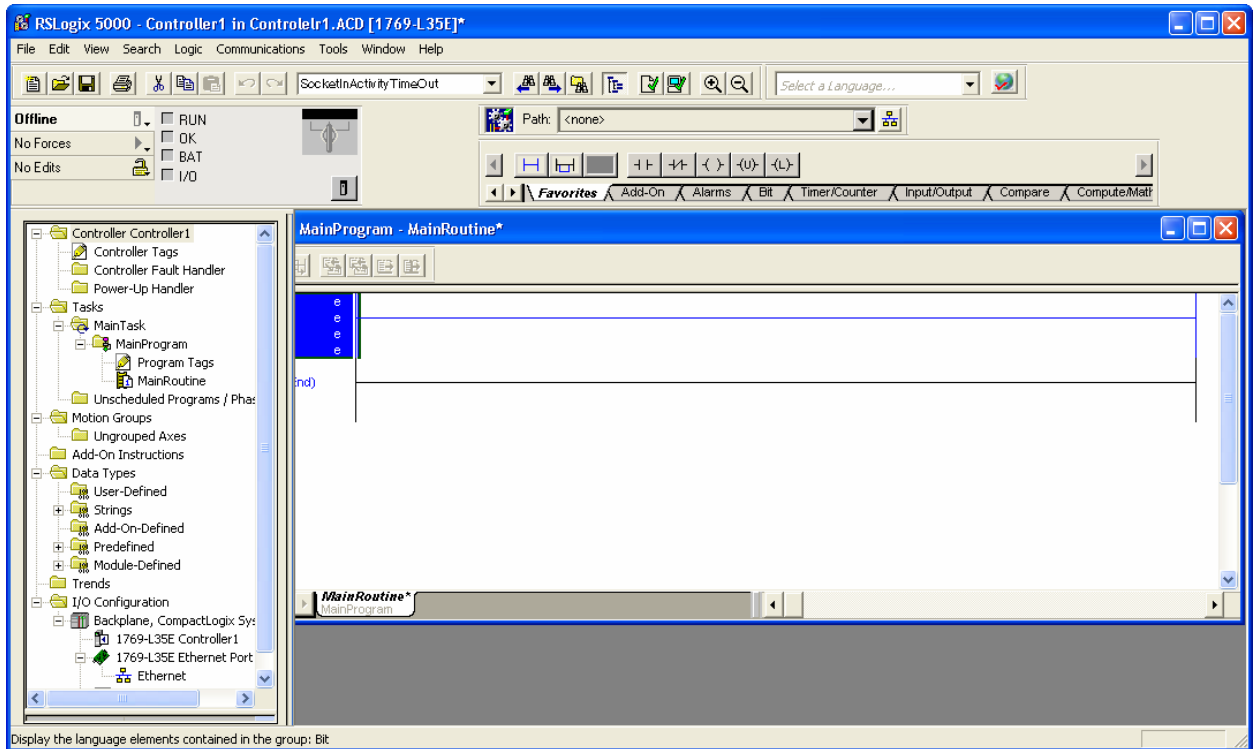



Once expanded, the **MainProgram** will appear as shown below:

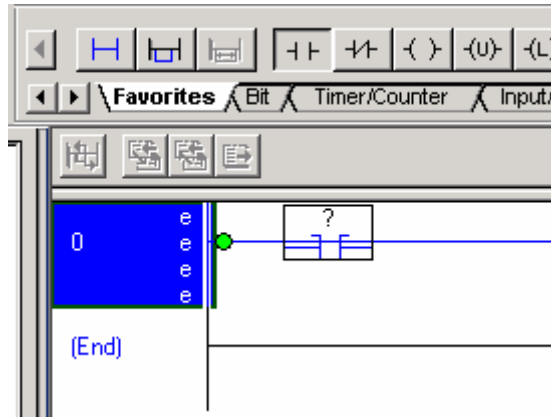
2. Double-click the MainRoutine icon.



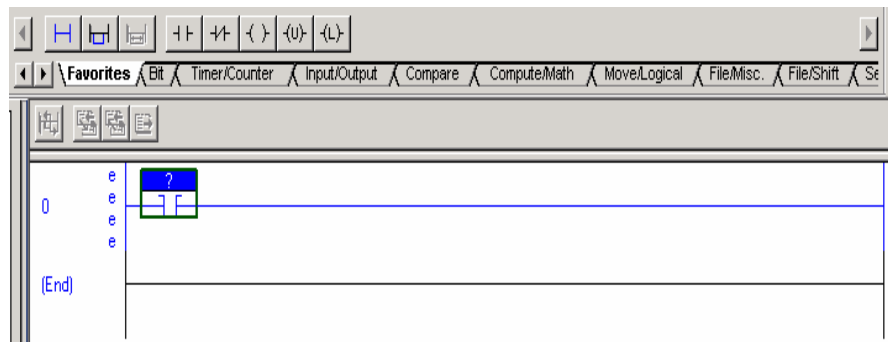
This will open the routine editor. An empty rung will be added for you as shown below:




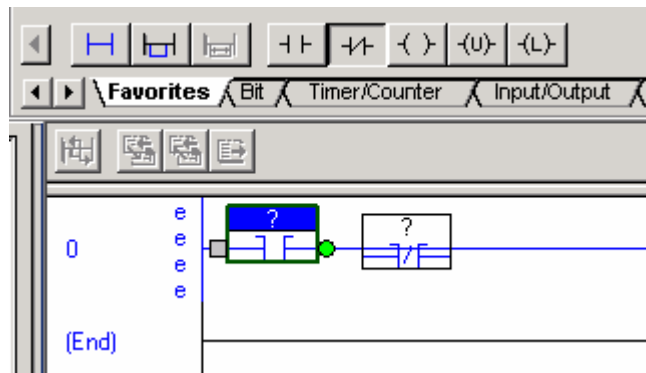
3. From the instruction toolbar, left click and hold on the **Examine if Closed (XIC)** instruction. 



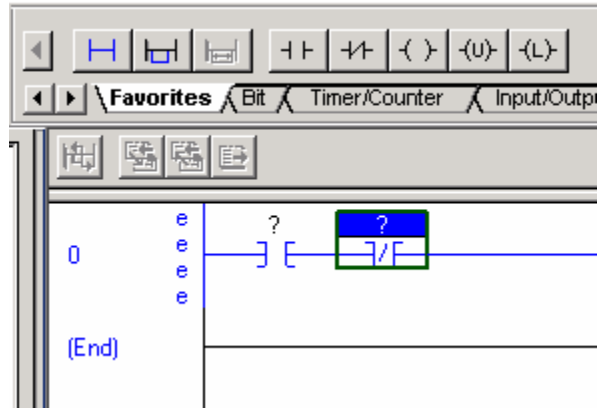
4. Drag the **XIC** onto rung 0 until the **green** dot appears as shown above. Release the mouse button at the location you wish to place your instruction.
5. Verify your rung appears like the figure below:



6. From the instruction toolbar left click and hold on the **Examine if Open (XIO)** instruction. 




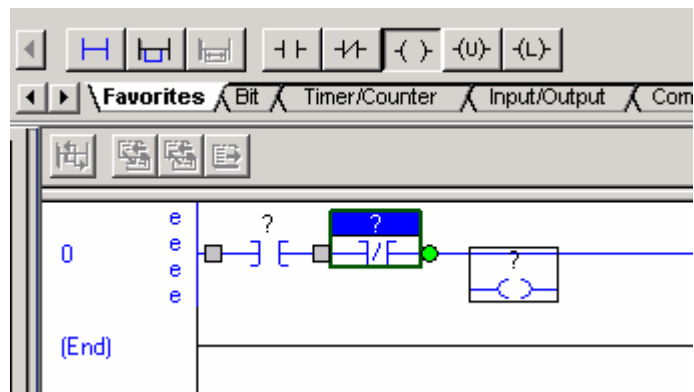
7. Drag the **XIO** onto rung 0 to the right of the XIC instruction as shown above. Again a green dot will appear to the right of the XIC instruction indicating where your new instruction will be inserted. Release the mouse button at the location you wish to place your instruction.
8. Verify your rung appears like the figure below:



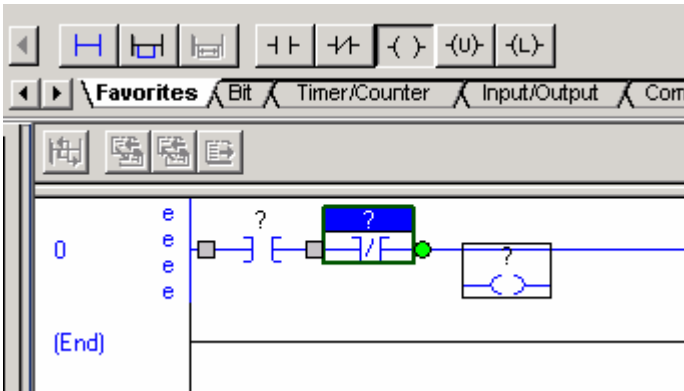
FYI

If you place an instruction in the wrong location on a rung simply click and hold on the instruction and drag it to the correct location.

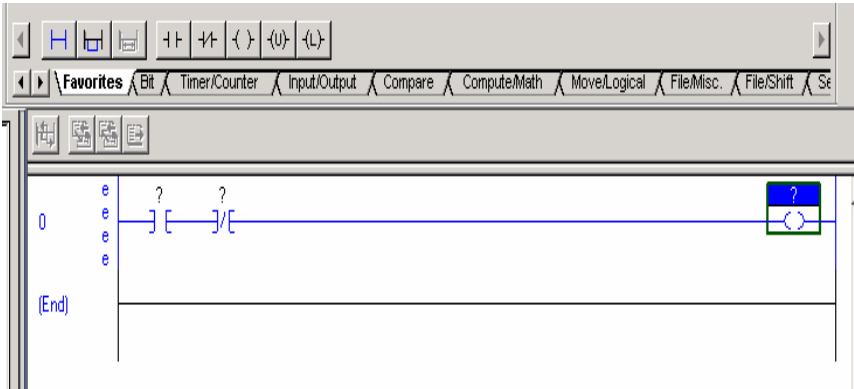
9. From the instruction toolbar, left click and hold on the **Output Energize (OTE)**  instruction.



- 10. Drag the **OTE** onto rung 0 to the right of the XIO instruction as shown above. Again a green dot will appear to the right of the XIO instruction indicating where the OTE instruction will be inserted. Release the mouse button at the location you wish insert the instruction.

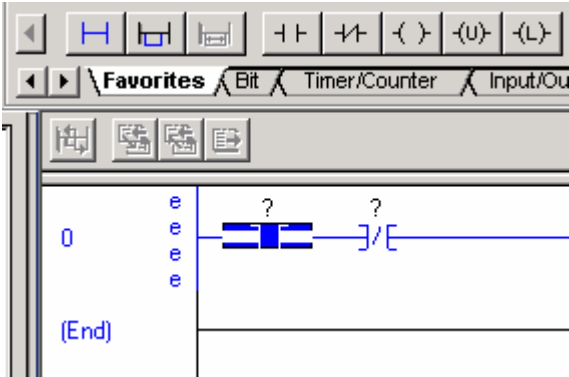


- 11. Verify your rung appears as shown below:



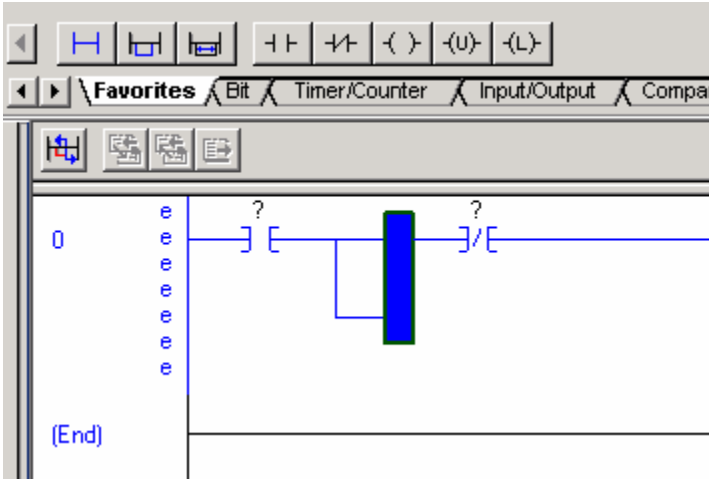
We will now add a branch around the XIC instruction.

- 12. Click on the **XIC** instruction to select it as shown below:



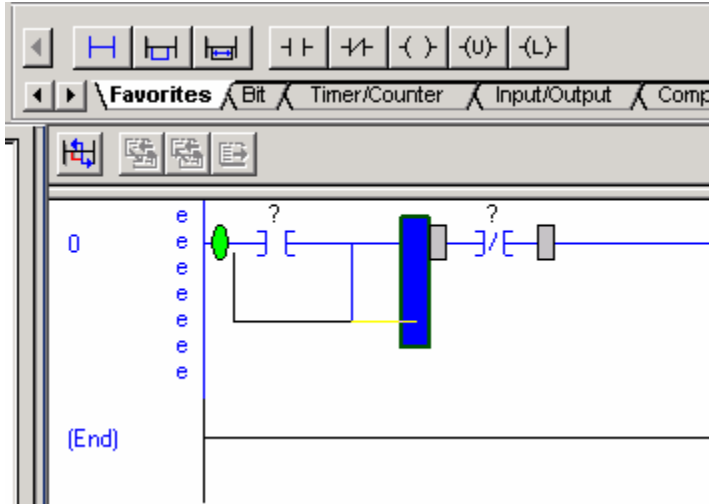
13. From the instruction toolbar click on the **Branch** instruction. 


A branch will be inserted on the rung.

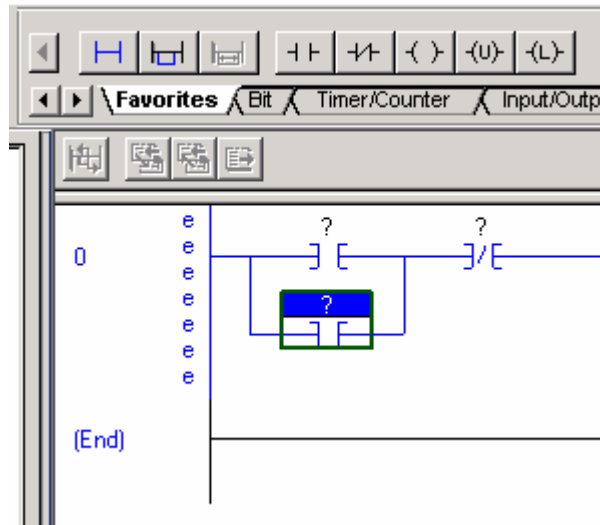


14. Left-click and hold on the **blue highlighted part of the branch** and drag your selected leg of the branch to the left side of the XIC instruction.

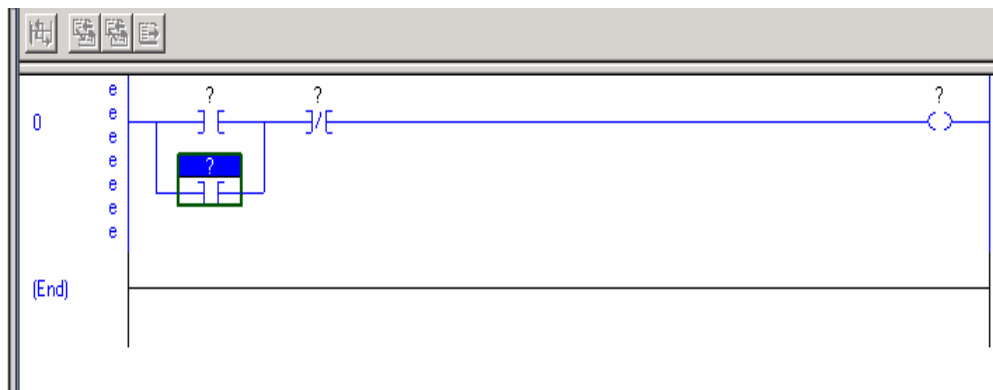
15. Place the branch over the green dot and release the mouse button.



16. From the instruction toolbar, left click and hold on the **XIC**  instruction.
17. Drag the **XIC** onto your newly created branch until the green dot appears.
The rung should now appear as shown below.



18. Verify that the entire rung appears like the figure below.



19. **Save** the program by clicking on the Save icon  on the toolbar. This will save the program in the default program directory, which is C:\RSLogix 5000\Projects\.

As you can see the free form editing in RSLogix 5000 can help speed development. You no longer have to place an instruction and tie an address to it before you add the next instruction.

Creating Tags for the Ladder Code

In this section of the lab you will create the tags needed for the program. In traditional PLCs, a physical memory address identifies each item of data, for example N7:0. In Logix controllers, there is no fixed numeric format. We use tags.

You will continue to use the project already opened.

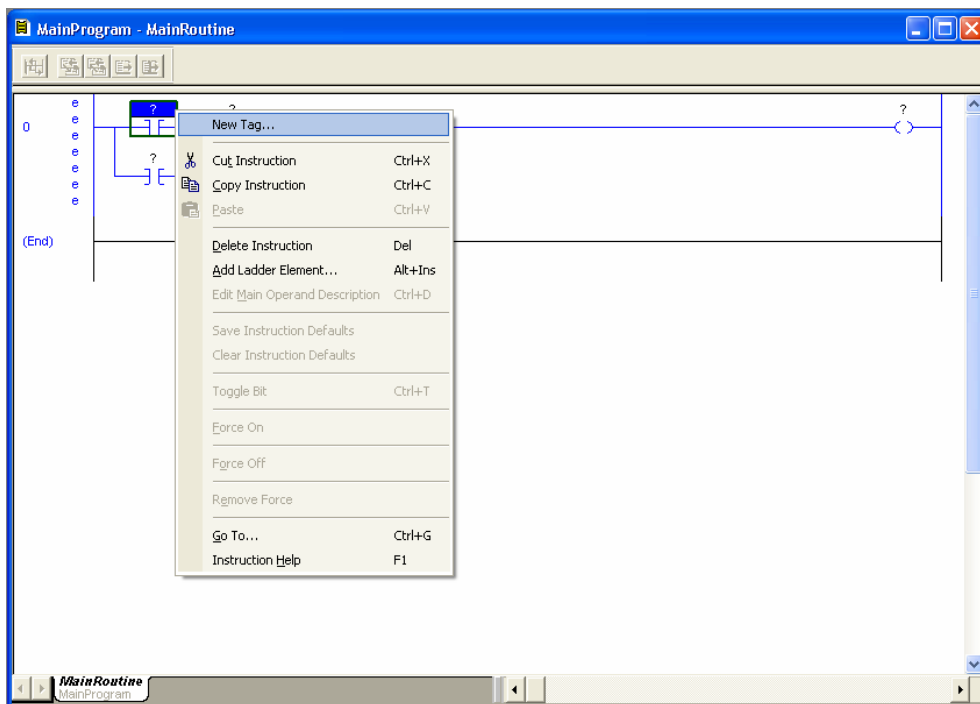
FYI

What is a tag and why are they better?

A tag is a text-based name for an area of memory. By using a text-based system you can use the name of the tag to document your ladder code and organize your data to mirror your machinery. For example you could create a tag named North_Tank_Pressure. This helps to speed code generation and debugging. All tag names are stored in the controller.

We will create 3 tags for the program: Motor_Start, Motor_Stop, and Motor_Run.

1. First you will create the tag Motor_Start. Right click on the ? of the first **XIC** instruction and select **New Tag**.



The **New Tag** window will appear.

The screenshot shows the 'New Tag' dialog box with the following fields and values:

- Name: [Empty text box]
- Description: [Empty text area]
- Usage: <normal>
- Type: Base
- Alias For: [Empty dropdown]
- Data Type: BOOL
- Scope: MainProgram
- Style: Decimal

FYI

Creating a Tag

When you create a tag there are several attributes for a tag. The main attributes we are interested in for this lab are as follows:

Type: Defines how the tag operates within the project

Base: Stores a value or values for use by logic within a project

Alias: A tag that represents another tag

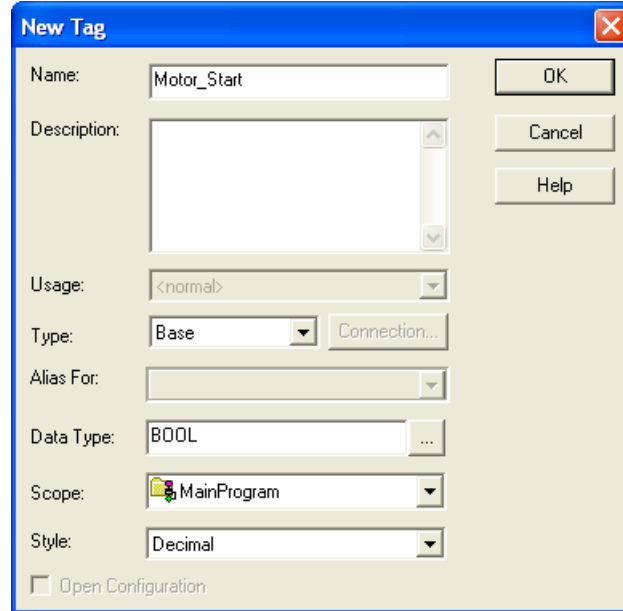
Produced: Send data to another controller

Consumed: Receive data from another controller

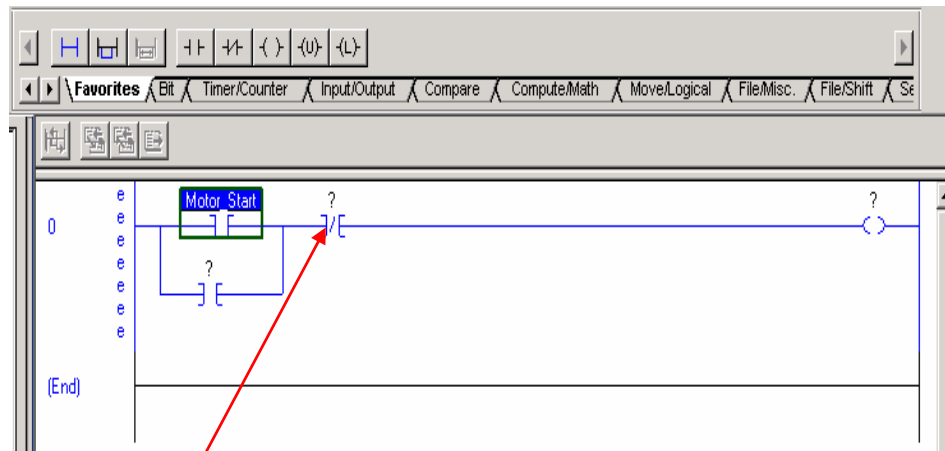
Data Type: Defines the type of data that the tag stores. For example: Boolean, Integer, Real, String, etc.

Scope: Defines how the data is accessed in the project. It is either controller scoped, global data accessible throughout the controller or program scoped, data accessible for a specific program.

3. Enter the parameters as shown below.



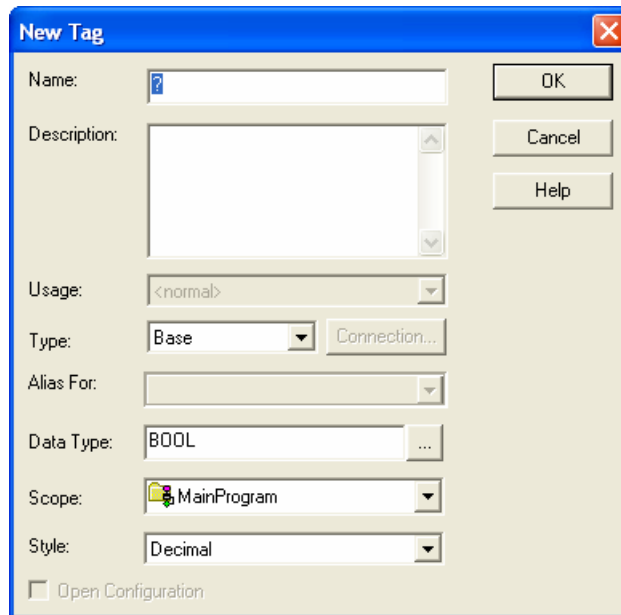
4. Click **OK** to accept and create the tag.
The rung will now look like the figure below.



Next you will create the tag `Motor_Stop`.

5. Right click on the ? of the **XIO** instruction and select **New Tag**.

Again, the **New Tag** window will appear:

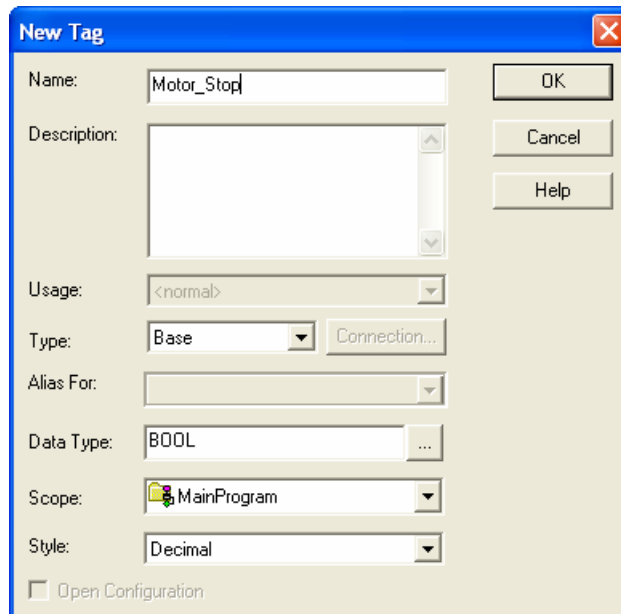


The 'New Tag' dialog box is shown with the following fields and values:

- Name: [?]
- Description: [Empty text area]
- Usage: <normal>
- Type: Base
- Alias For: [Empty dropdown]
- Data Type: BOOL
- Scope: MainProgram
- Style: Decimal
- Open Configuration:

Buttons: OK, Cancel, Help

6. Enter the parameters as shown below:



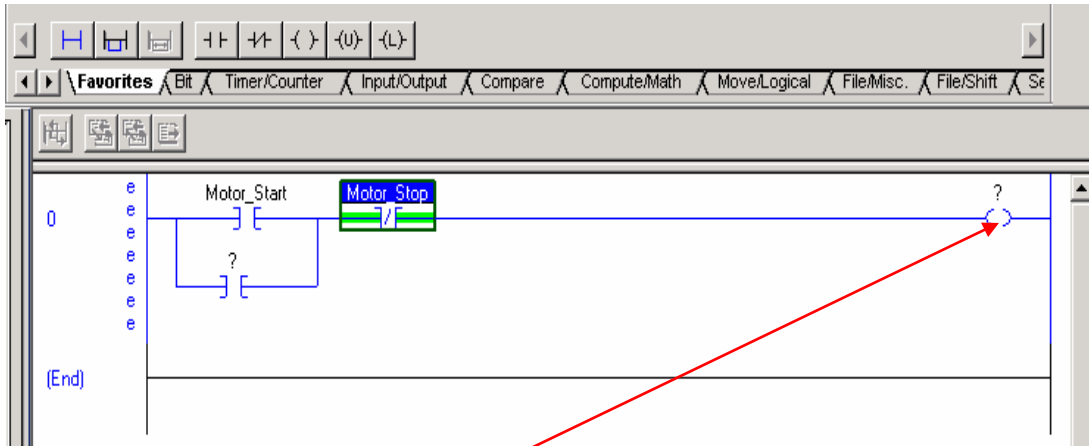
The 'New Tag' dialog box is shown with the following fields and values:

- Name: Motor_Stop
- Description: [Empty text area]
- Usage: <normal>
- Type: Base
- Alias For: [Empty dropdown]
- Data Type: BOOL
- Scope: MainProgram
- Style: Decimal
- Open Configuration:

Buttons: OK, Cancel, Help

7. Click **OK** to accept and create the tag.

8. Verify the rung appears like the figure below:



You will now create the tag Motor_Run.

9. Right click on the ? of the **OTE** instruction and select **New Tag**.

The New Tag window will appear.

10. Enter the parameters as shown below:

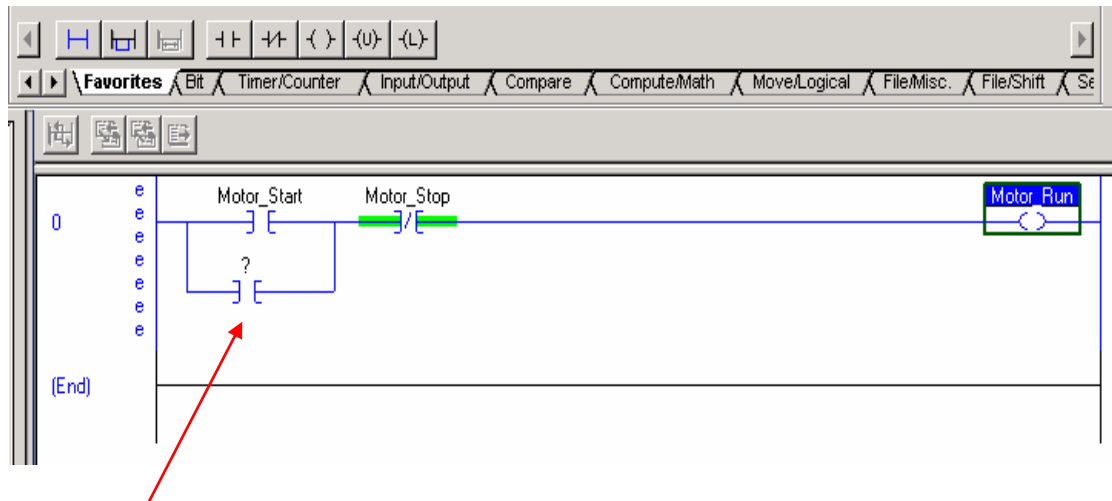
The 'New Tag' dialog box is shown with the following parameters:

- Name: Motor_Run
- Description: (empty text area)
- Usage: <normal>
- Type: Base
- Alias For: (empty dropdown)
- Data Type: BOOL
- Scope: MainProgram
- Style: Decimal
- Open Configuration: (unchecked checkbox)

Buttons for OK, Cancel, and Help are visible on the right side of the dialog.

11. Click **OK** to accept and create the tag.

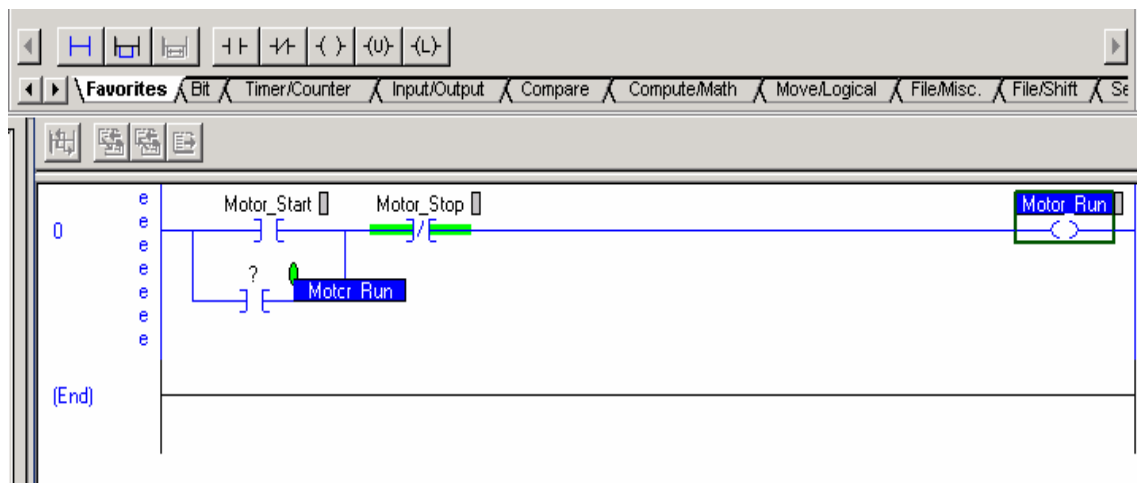
Your rung should now appear as shown below:



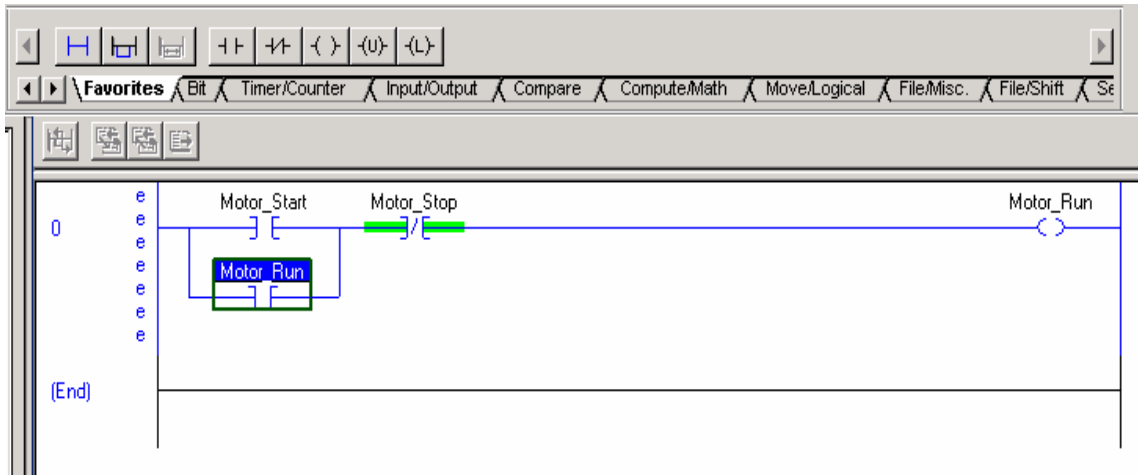
For the **XIC** instruction in the branch we do not have to create a tag. You will use the tag **Motor_Run**.

12. Left click and hold the mouse button over the tag **Motor_Run** on the **OTE** instruction.

13. Drag the tag **Motor_Run** tag over to the **XIC** instruction until a green dot appears next to the ?. Then release the mouse button.



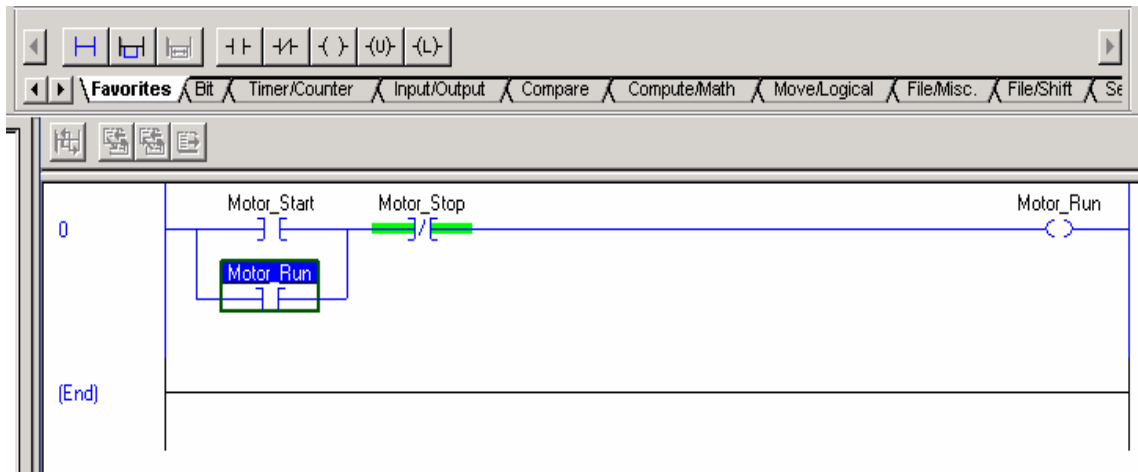
Your rung should now appear as shown below. Notice the 'e's next to rung zero. These indicate that the rung is in edit mode.




14. Click on the **(End)** rung. The 'e's are now gone.

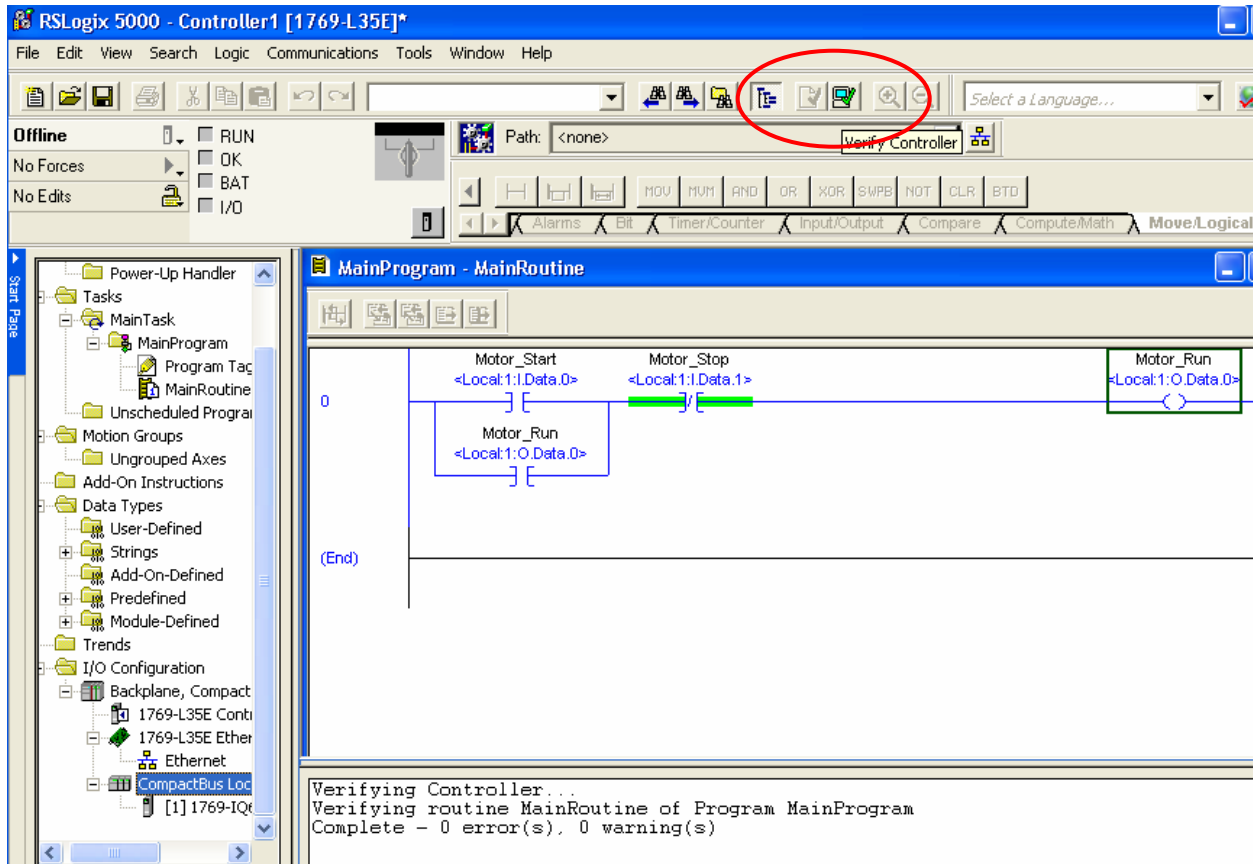
RSLogix 5000 software verifies each rung automatically when you click the mouse off of it. This makes programming easier!


Your rung should now appear as shown below:



15. Verify the program by clicking on the Verify Routine or Verify Controller  icon on the toolbar.

You will see if there are any errors in the status window.



16. **Save** the program by clicking on the Save icon  on the toolbar.

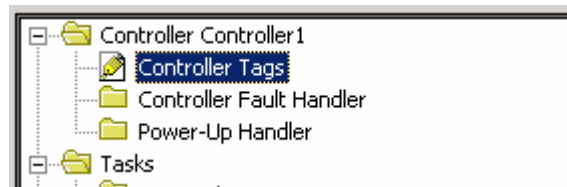
The tag database of Logix versus a traditional PLC's fixed memory addresses help you create self-documenting code. This means you do not have to use address descriptions or symbols to make code easy to read.

Monitoring/Editing Tags

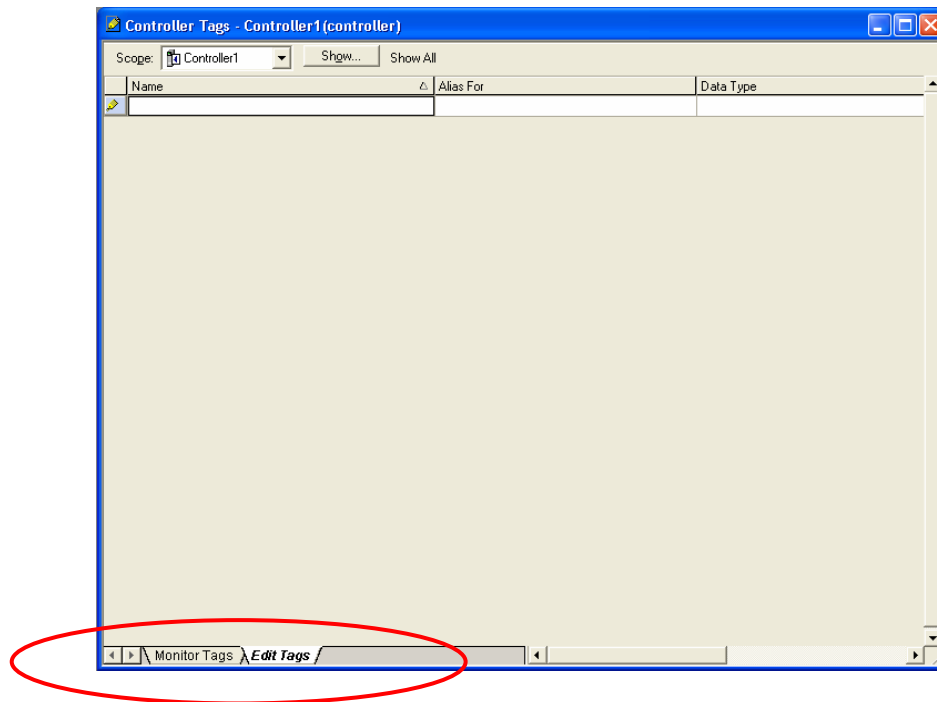
In this section of the lab, we will review the Tag Monitor/Editor in RSLogix 5000. We will also discuss the concept of Controller versus Program scoped tags.

You will continue to use the project already opened.

1. From the Controller Organizer double-click on **Controller Tags**.



The tag Monitor/Editor window appears. You notice in the lower left corner of the window two tabs labeled **Monitor Tags** and **Edit Tags** as shown below.



FYI

Monitor/Edit Tags Tabs

When the 'Monitor Tags' tab is selected the actual value(s) for the tags will be shown. For example, if you were to view an input button the software would show the button tag actively energized or de-energized.

When the 'Edit Tags' tab is selected, NEW tags may be created, and existing tag properties may be modified.

If you are having difficulty creating or modifying tag parameters, verify that the 'Edit Tags' tab is selected.

You notice first that there are no tags present, remember you just created 3 tags.



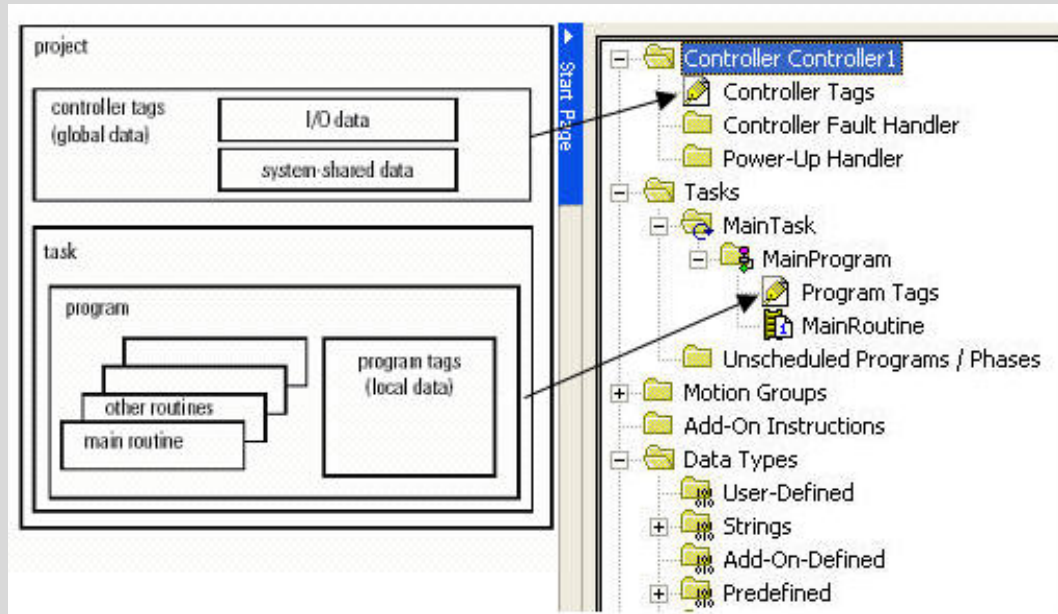
Scope:  Controller1		Show...	Show All
Name	Alias For	Base Tag	Data Type
			

Notice a field in the upper left corner of the Tag Editor window labeled **Scope**. Earlier in the lab we talked briefly about Controller and Program scoped tags. Currently the selection is **Controller1(controller)**.

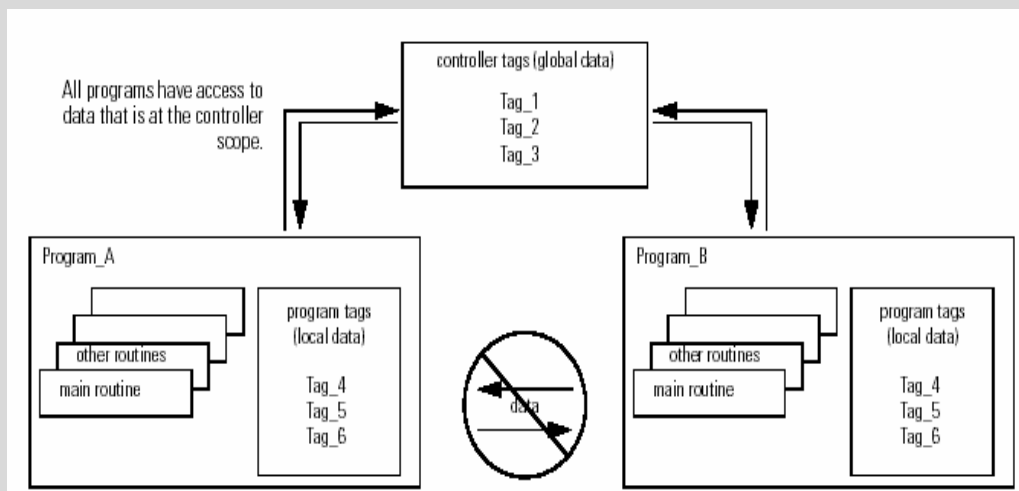
When we created the tags earlier we created them in the Program Scope.
FYI

Data Scoping

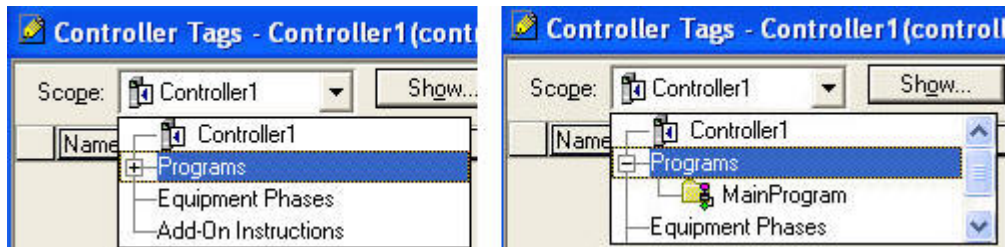
When you create a tag, you define it either as a controller tag (global data) or a program tag for a specific program (local data).



Data at the program scope is isolated from other programs. Routines cannot access data that is at the program scope of another program. Thus you can re-use the tag name of a program-scoped tag in multiple programs.




2. Click on the down arrow for the **Scope** selection box.
3. Select **Programs** → **MainProgram**



The Tag Editor now has switched views to the program level and you see the tags you created earlier.

Name	Alias For	Base Tag	Data Type
Motor_Run			BOOL
Motor_Start			BOOL
Motor_Stop			BOOL

4. **Save** the program by clicking on the Save icon  on the toolbar.

Congratulations! You have Completed Lab 1. Please move on to Lab 2.

Lab 2: Configuring I/O (20 minutes)

About this Lab

We will now look at configuring I/O for our project. To communicate with I/O modules you must add modules to the I/O Configuration folder.

During this off-line lab we will show adding 1769 I/O using the equipment at your lab station.

1769 I/O is native to both 1768-L43 and 1769-L35E controllers.

You will continue to use the project already opened.

For this lab we will add the following I/O modules. Please note the I/O that relates to the equipment at your lab station.

- 1769-IQ6XOW4/B Combination Digital I/O Module
- 1769-IF4XOF2 (Optional exercise) Combination Analog I/O Module

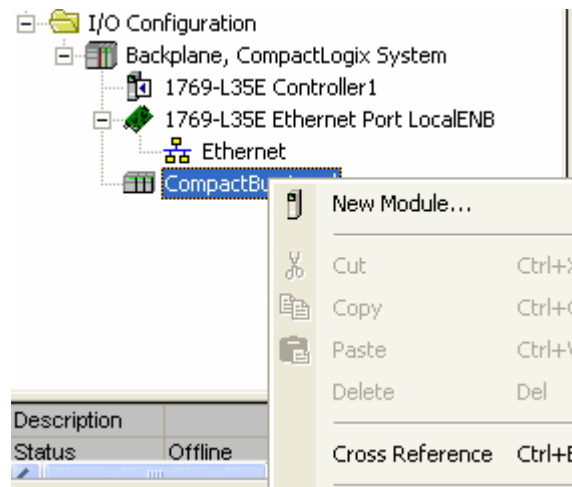
About this Lab

In this section of the lab, you will:

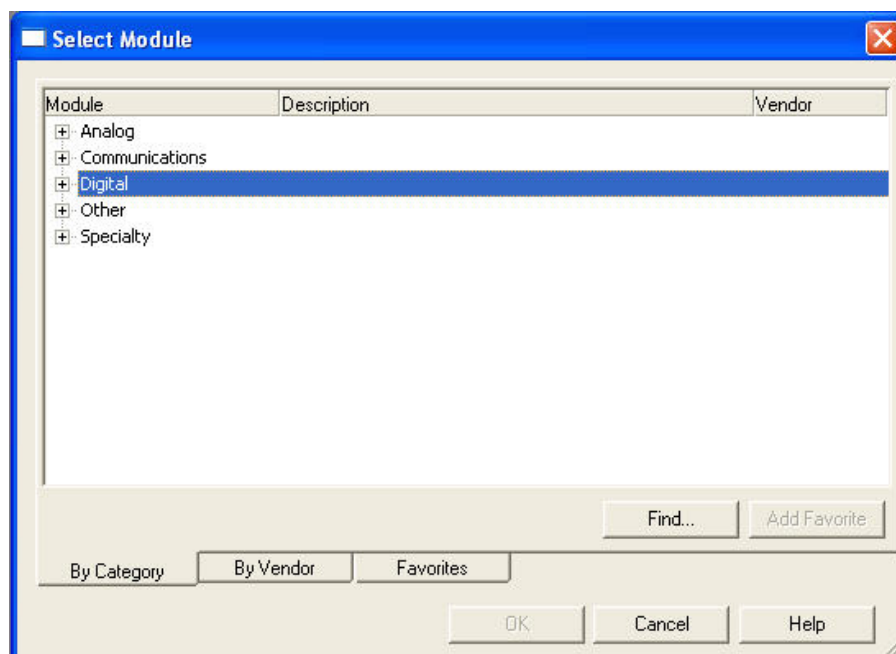
4. Add CompactLogix I/O to your application
5. View the I/O tags that were automatically created
6. Learn about tag aliasing

Adding CompactLogix I/O

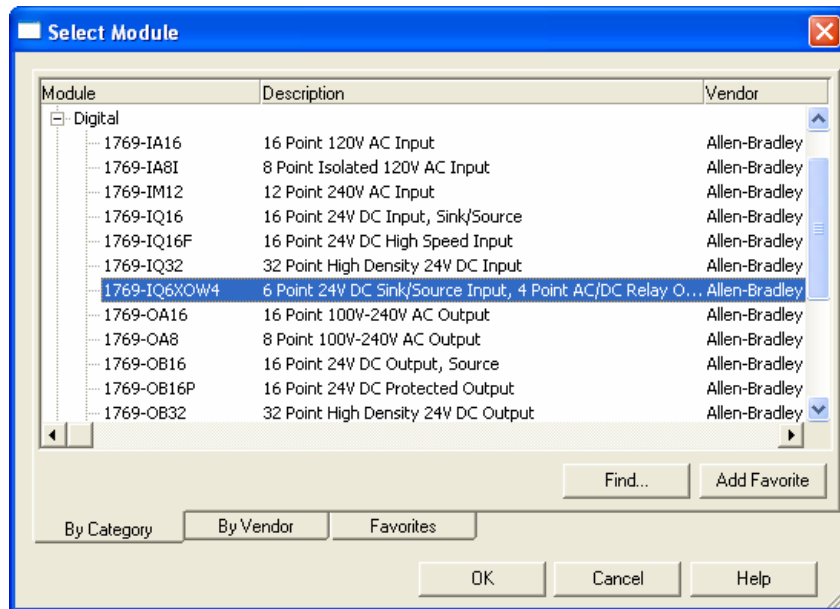
1. From the Controller Organizer right click on **CompactBus Local** and select **New Module**.



2. The **Select Module** window appears. Click and expand Digital.



3. Scroll through the list until you locate the **1769-IQ6XOW4**.
4. Select the **1769-IQ6XOW4** module and click **OK**.



The **Module Configuration Wizard** will appear for the **1769-IQ6XOW4**.

FYI

Module Configuration Wizard

Whenever you add an I/O module, to the system you will go through the Module Configuration Wizard. The Wizard allows you to step through the entire configuration needed for a module. You can access this information later by double clicking on a module in the I/O Configuration folder or through the tag monitor/editor.

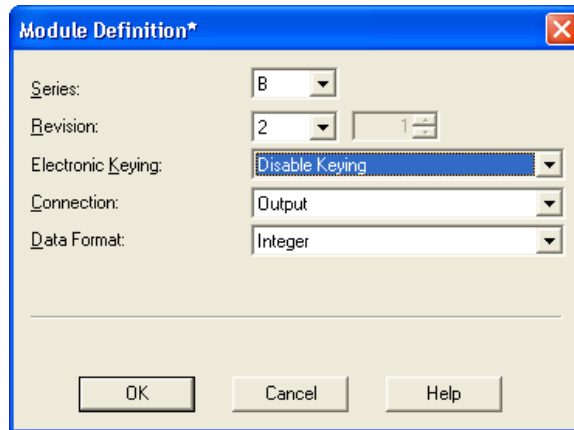
In Logix there are no more dip switches or jumpers needed to configure I/O modules. I/O modules are software configured. This saves time when setting up a system. The configuration for all modules is part of the controller's program and is downloaded to the module from the controller; this allows for ease of replacement if an I/O module fails.

5. Enter the **Name** and **Slot** parameters as shown below. Leave all other fields set to their default values.

The screenshot shows the 'New Module' dialog box with the following details:

- General** tab selected.
- Type: 1769-IQ6XOW4 6 Point 24V DC Sink/Source Input, 4 Point AC/DC Relay Output
- Vendor: Allen-Bradley
- Parent: Local
- Name: Discrete_IQ
- Slot: 1
- Description: (empty text area)
- Module Definition:
 - Series: B (with 'Change...' button)
 - Revision: 2.1
 - Electronic Keying: Compatible Module
 - Connection: Output
 - Data Format: Integer
- Status: Creating
- Buttons: OK, Cancel, Help

6. Click on the **Change** button and change the **Electronic Keying** information as shown below, and then click **OK** to accept your changes



FYI

Connection

Determines the data structure for the tags that are associated with the module. Many I/O modules support different formats. Each format uses a different data structure.

Electronic Keying

When you insert a module into a slot of a chassis, the controller compares the information read from the newly inserted module with what the user configured that particular slot to be in their project.

The following data is read and compared:

Vendor, Product Type, Catalog Number, Major Revision, Minor Revision.

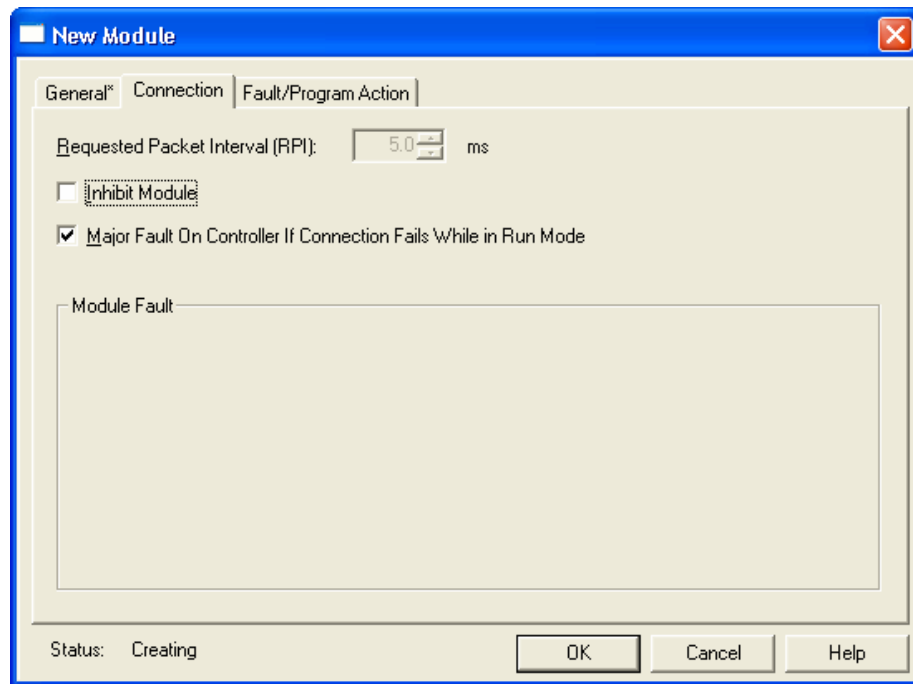
The user may select one of the following module keying options during the initial module configuration:

Exact Match – all of the parameters described above must match or the inserted module will reject the connection.

Compatible Module – The following criteria must be met, or else the inserted module will reject the connection: Module Types, Catalog Number, and Major Revision must match and the Minor Revision of the physical module must be equal to or greater than the one specified in the software

Disable Keying – No keying used at all.

7. Click on the **Connection** tab to view the **Requested Packet Interval** data.



FYI

Requested Packet Interval (RPI)

The Requested Packet Interval specifies the period at which data is updated to and from the module. RPIs are configured in milliseconds. The range is .2ms to 750ms.

ControlLogix and 1768-L43 processors allow individual RPI values to be configured whereas 1769-L35E CompactLogix processors treat I/O module connections as if they were rack optimized meaning all 1769 I/O modules must share the same RPI.

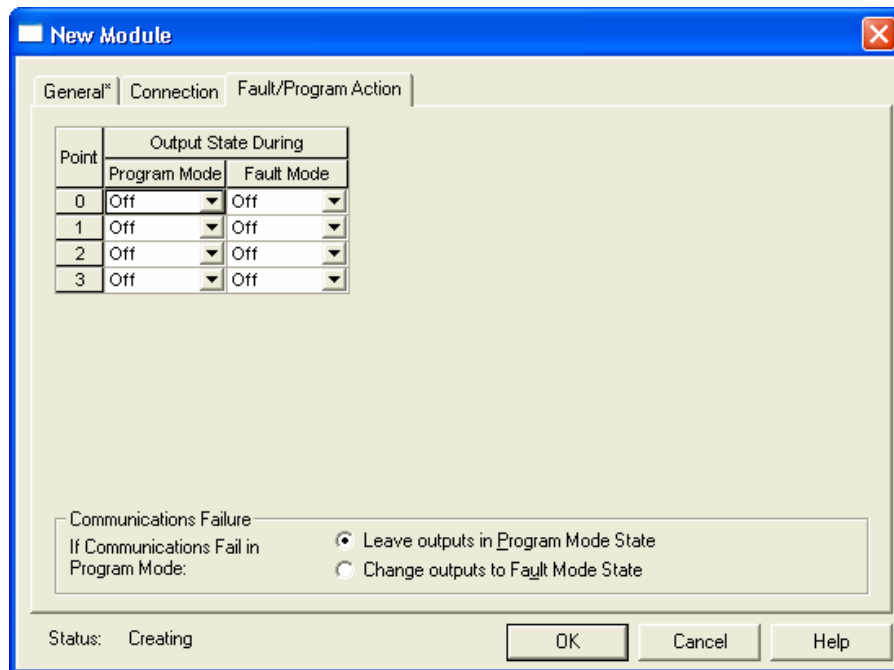
- Click on the **Fault/Program Action** tab.

The Fault/Program Action screen appears. This shows specific configuration parameters for the 1769-IQ6XOW4 module.

FYI

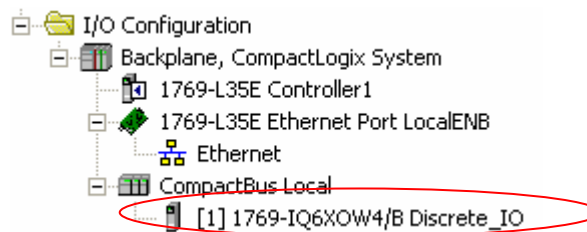
Fault/Program Action tab of 1769 I/O modules


CompactLogix controllers do not support Hold Last State (HLS) or User Defined Safe State (UDSS) for fault/program modes for outputs. Outputs are set to zero when the controller faults or enters Program mode. These features are supported when the module is connected to a 1769-ADN Compact I/O DeviceNet adapter module.



- Click on **OK** to close the wizard.

In the Controller Organizer, the **I/O Configuration** folder will show the digital I/O module in Slot 1:



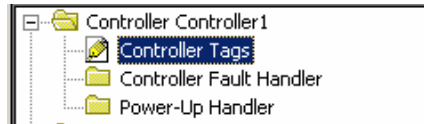
- Save** the program by clicking on the Save icon  on the toolbar.

Viewing the CompactLogix I/O Tags Just Created

Now that we have configured I/O modules in the project, let's take a look how that information is presented in RSLogix 5000.

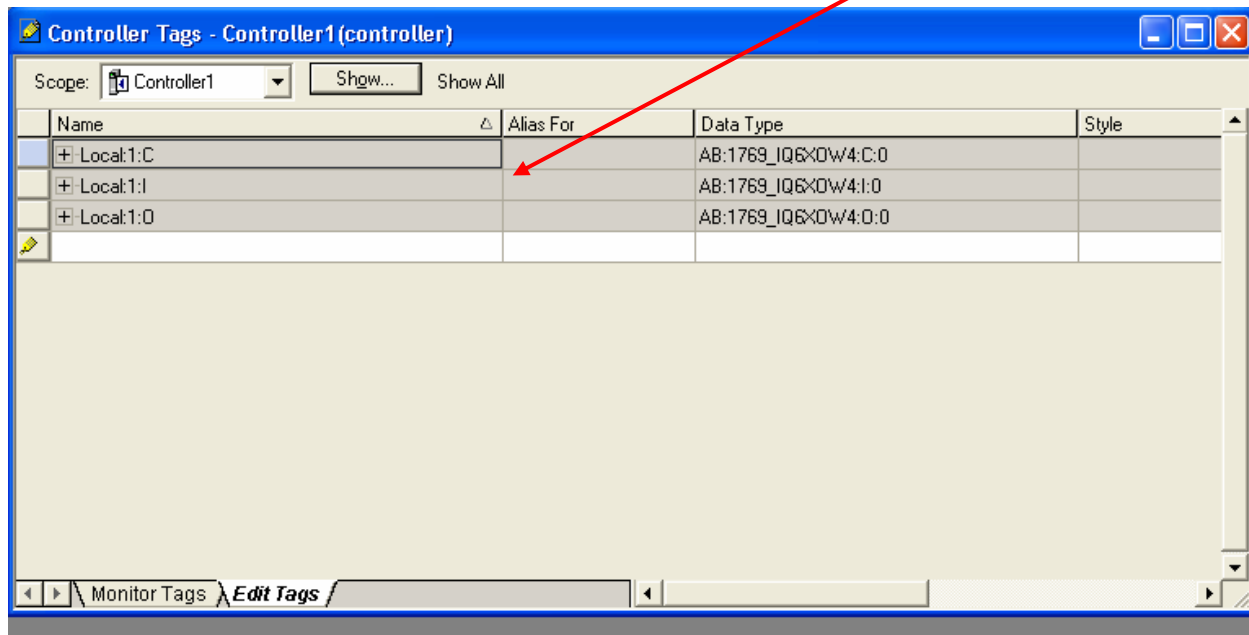
You will continue to use the project already opened.

1. From the Controller Organizer double click on Controller Tags.



Drag to the right to increase the size of the Tag Name field. This will allow you to view the entire Tag Name.

The tag editor window will appear.




FYI

I/O Address Format

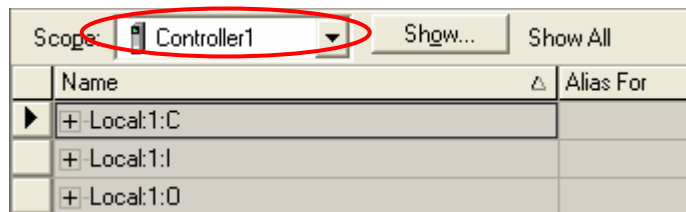
An I/O address follows this format:

Location	:Slot	:Type	.Member	.SubMember	.Bit
----------	-------	-------	---------	------------	------

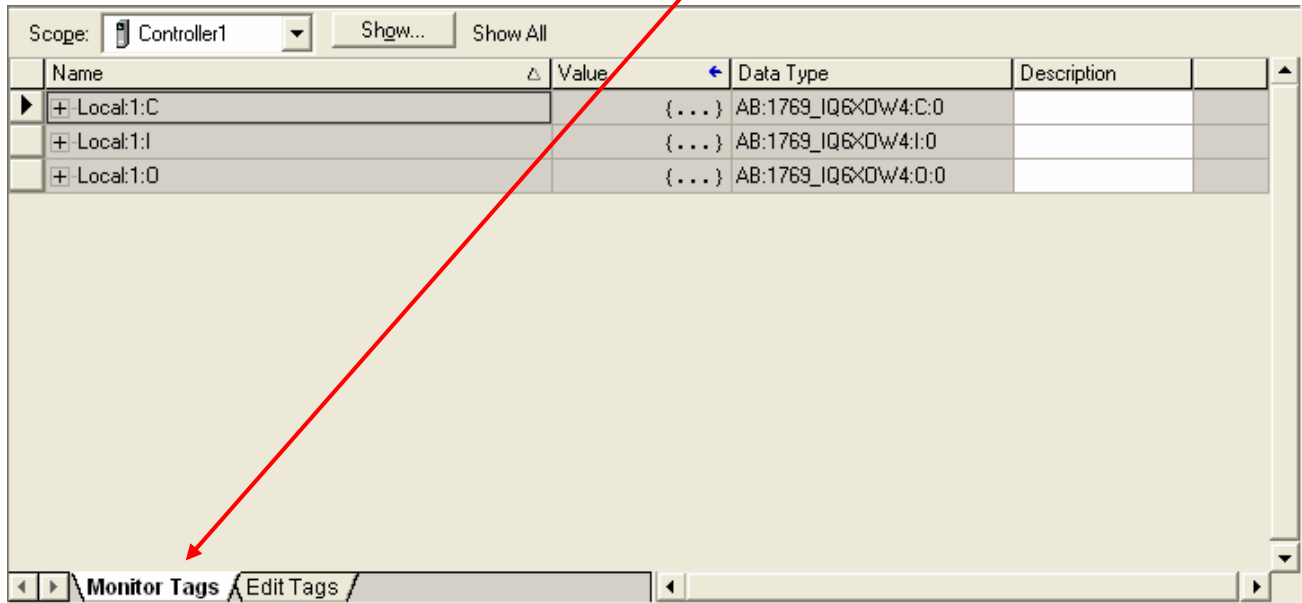
 = Optional

Where:	Is:
<i>Location</i>	Network location LOCAL = same chassis or DIN rail as the controller ADAPTER_NAME = identifies remote communication adapter or bridge module
<i>Slot</i>	Slot number of I/O module in its chassis or DIN rail
<i>Type</i>	Type of data I = input O = output C = configuration S = status
<i>Member</i>	Specific data from the I/O module; depends on what type of data the module can store. <ul style="list-style-type: none">• For a digital module, a Data member usually stores the input or output bit values.• For an analog module, a Channel member (CH#) usually stores the data for a channel.
<i>SubMember</i>	Specific data related to a Member.
<i>Bit</i>	Specific point on a digital I/O module; depends on the size of the I/O module (0-31 for a 32-point module)


You notice by looking in the upper left corner of the tag editor that you are in the Controller Scope. All I/O module tags are created in the Controller Scope.



2. Switch to Monitor Tags by Clicking on the **Monitor Tags** Tab.



The above entries are tag structures for the modules you added. They contain more tags than are actually displayed. Note the + sign next to the tag name, this indicates that you can expand the tag structure to see more information.

3. Expand and explore the tags for the I/O modules by clicking the +.
What you will find under the Configuration tags, for each module, is all the data, you entered and selected from the Module Configuration Wizard.
4. **Save** the program by clicking on the **Save** icon  on the toolbar.

Assigning Alias Tags

In this section of the lab you will learn about Alias Tags.

You will continue to use the project already opened.

FYI

Aliasing

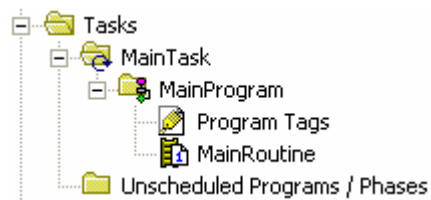
An Alias tag lets you create one tag that represents another tag.

- Both tags share the same value
- When the value of one of the tags changes, the other tag reflects the change

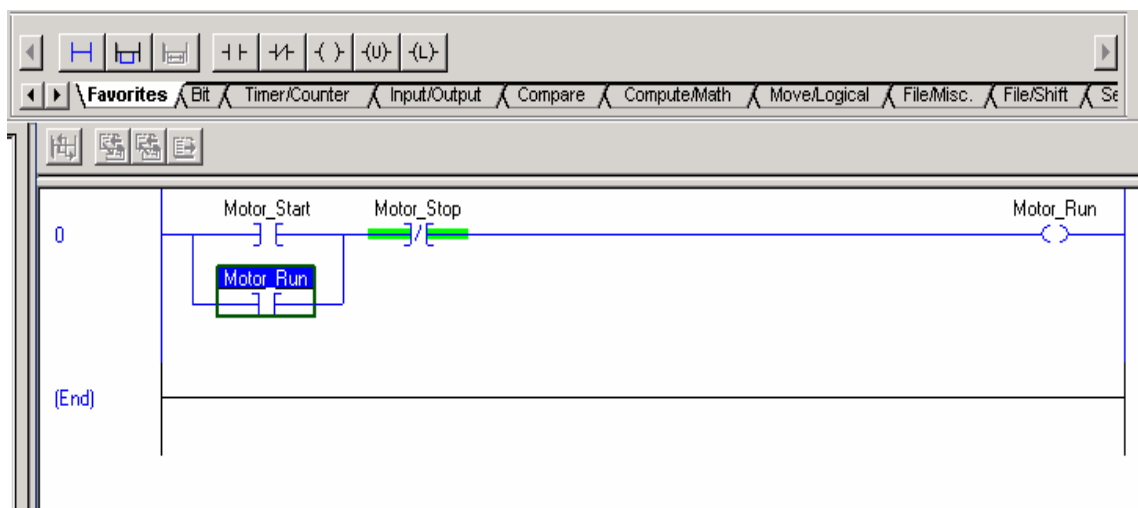
Use Aliases in the following situations:

- Program logic in advance of wiring diagrams
- Assign a descriptive name to an I/O device
- Provide a simpler name for a complex tag
- Use a descriptive name for an element of an array

1. From the Controller Organizer double click on **MainRoutine**.



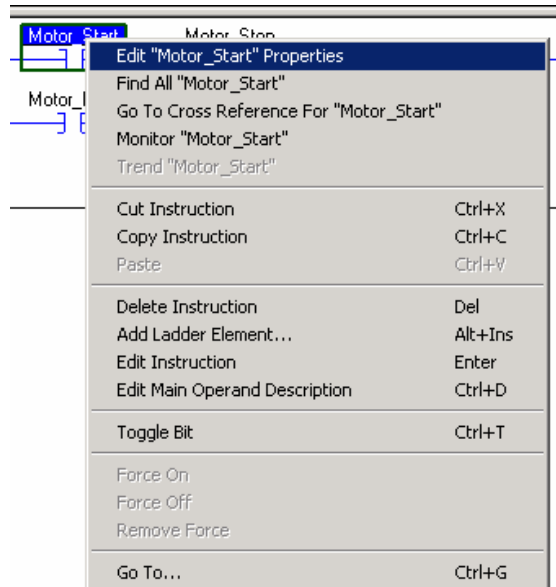
The ladder editor appears as shown below:



In the last part of the lab we added I/O modules to the project. Now lets Alias the tags in the program to the I/O Modules.

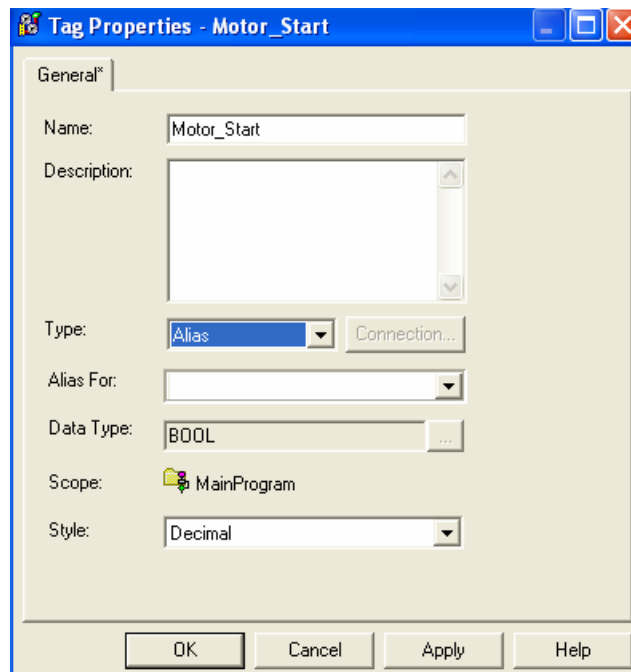
- Motor_Start will be Aliased to input point zero of the 1769-IQ6XOW4 in Slot one.
- Motor_Stop will be Aliased to input point one of the 1769-IQ6XOW4 in Slot one.
- Motor_Run will be Aliased to output point zero of the 1769-IQ6XOW4 in Slot one.

2. Right click on the tag **Motor_Start** and select **Edit 'Motor_Start' Properties**.



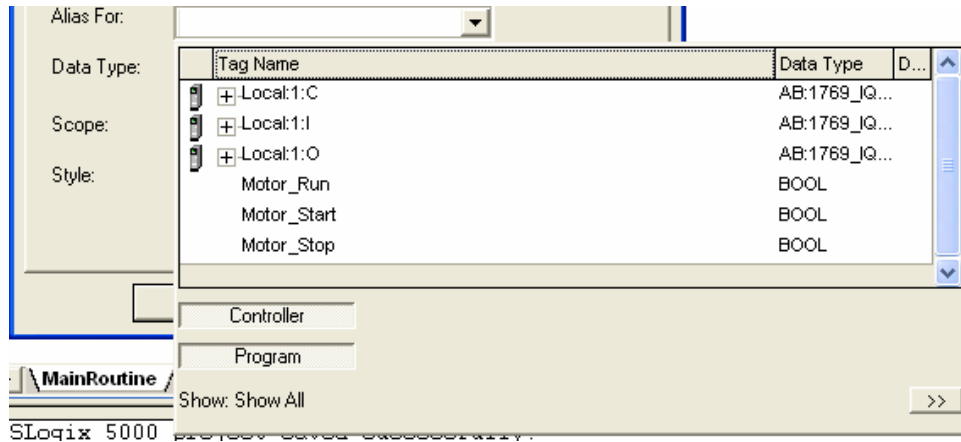
The **Tags Properties** window for Motor_Start will appear. Currently the tag is defined as a Base tag.

3. Select **Alias** as a type and notice that the **Tag Properties** window changed.



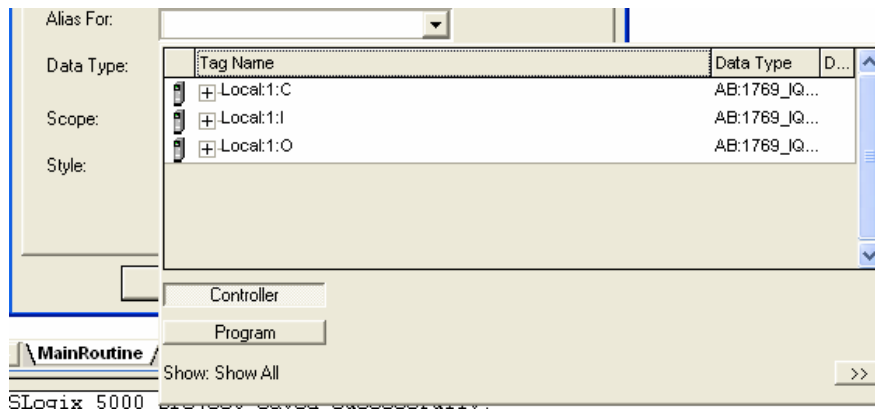
4. Click on the down arrow for **Alias For**.

The tag browser appears. The browser shows both Controller and Program Scope Tags. You will need to select your address from controller scoped tags.



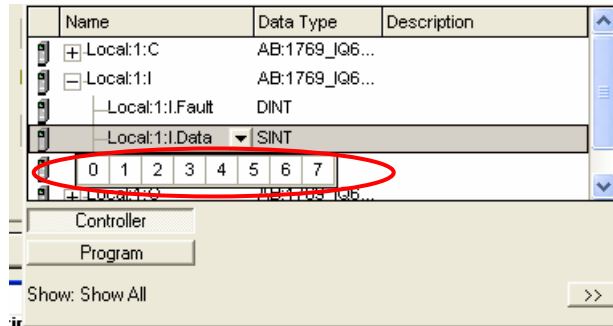
5. Click on the **Program Scoped Tags** button to deselect Program Scope Tags.

The view on the screen will change to view only your Controller Scoped Tags

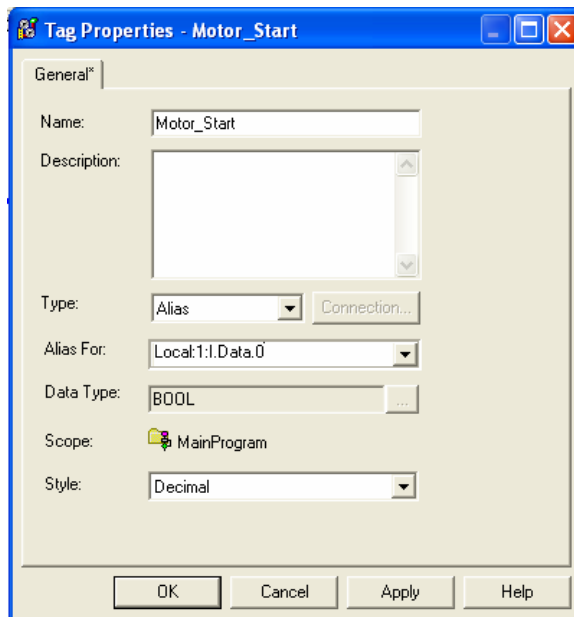


6. Expand Local:1:I by clicking on the + sign and select **Local:1:I.Data**.

- Click the down arrow for Local:1:I.Data as shown below.
This will open the table of data points for the 1769-IQ6XOW4 module.



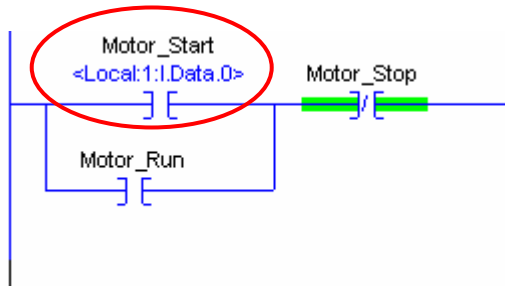
- Select 0 from the table.
When you select 0 from the tag browser the window will close. **Tag Properties** will now appear as follows:



Motor_Start will now be aliased to **Local:1:I.Data.0**. This is the 1769-IQ6XOW4 in Slot 1.

- Click **OK** to close and apply the changes to the tag Motor_Start.

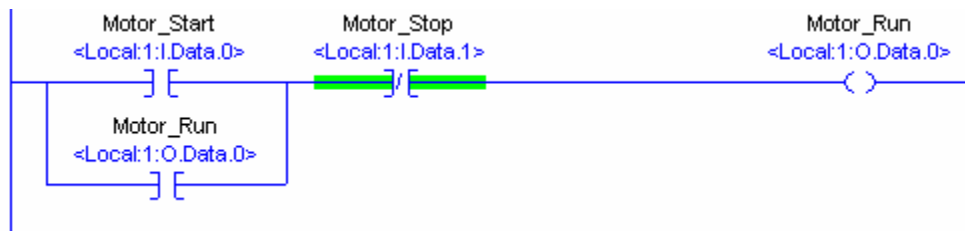
Motor_Start has been Aliased to Local:1:I.Data.0. This means that the tags are equivalent to one another in code. Is it easier to use Motor_Start than Local:1:I.Data.0.




10. Using the previous steps, alias the remaining two tags.

- Motor_Stop = Local:1:I.Data.1
- Motor_Run = Local:1:O.Data.0

11. When you are finished the ladder code should appear as follows:



12. **Save** the program by clicking on the **Save** icon  on the toolbar.

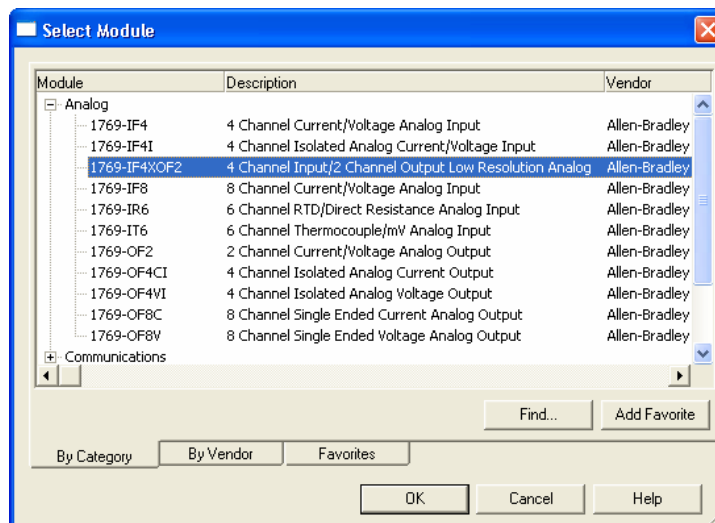
13. Minimize the RSLogix 5000 software.

Optional Exercise - Adding Analog I/O

If time permits, try adding the analog I/O module to your RSLogix 5000 project. Use the following steps (and what you have learned so far) to help you add the analog I/O module to your project. ***If you do not have time to do this, move on to lab 3 on pg 53.***

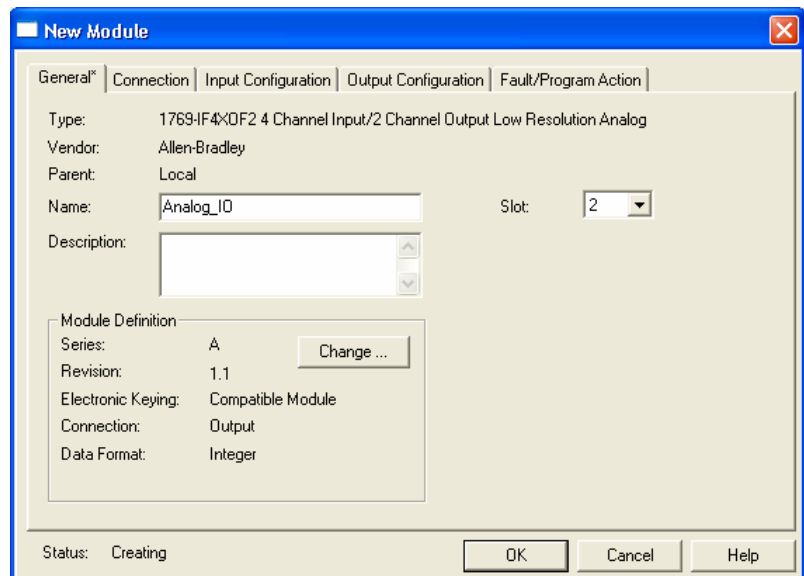
You will continue to use the program already opened.

1. Maximize the RSLogix 5000 software window.
2. In the Controller Organizer, right click on CompactBus Local and select **New Module**.
3. Click and Expand **Analog** from the Select Module window.
4. Scroll through the list until you locate the **1769-IF4XOF2 Analog Module**.. Select the module and click **OK**.

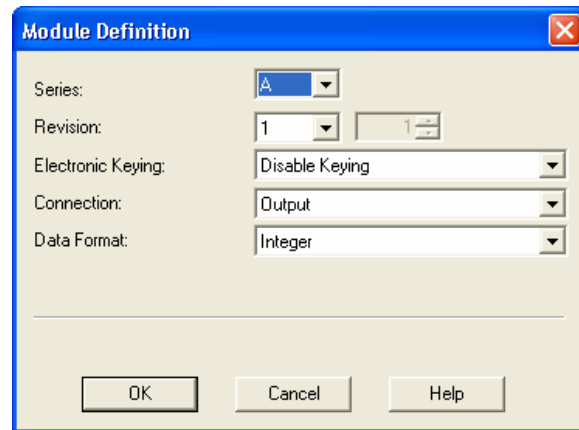


The Module Configuration Wizard will appear for the 1769-IF4XOF2.

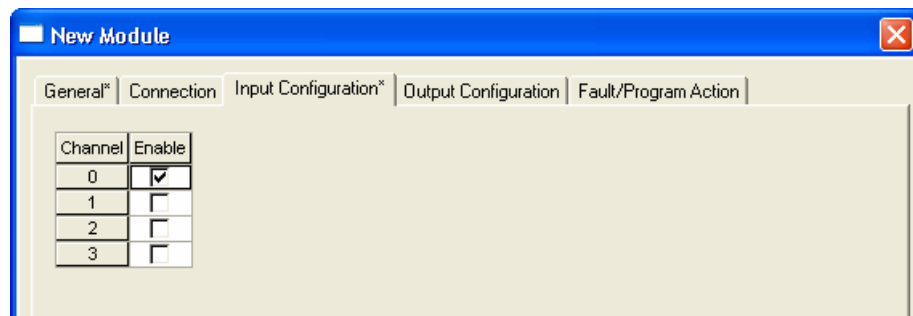
5. Enter the **Name** and **Slot** parameters as shown below. Leave all other fields set to their default values.
6. Click on the **Change** button and change the **Electronic Keying** information as shown below.



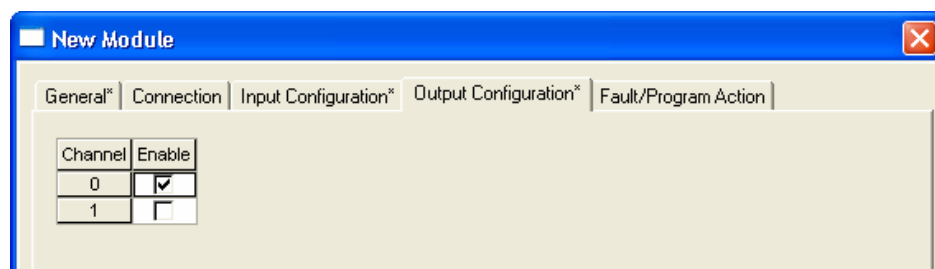
7. After changing the electronic keying, Click **OK** to accept the changes.



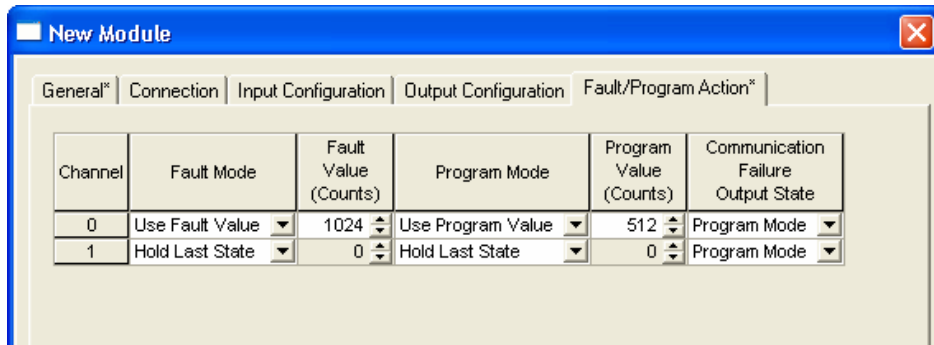
8. Click the **Input Configuration** tab and click on the checkbox to **Enable Input Channel 0**.



9. Click the **Output Configuration** tab and click on the checkbox to **Enable Output Channel 0**.



10. Click the **Fault/Program Action** tab and configure it as shown below:

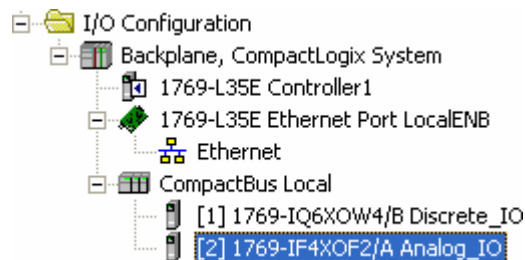



FYI

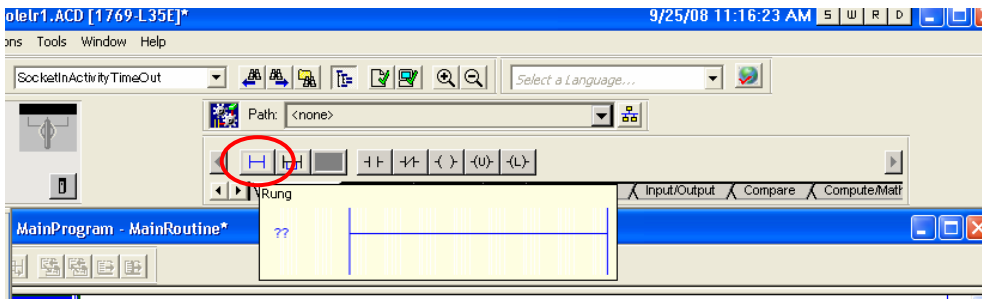
Fault/Program Action tab of 1769 I/O modules

CompactLogix controllers do not support Hold Last State (HLS) or User Defined Safe State (UDSS) for fault/program modes for outputs. Outputs are set to zero when the controller faults or enters Program mode. These features are supported when the module is connected to a 1769-ADN Compact I/O DeviceNet adapter module.

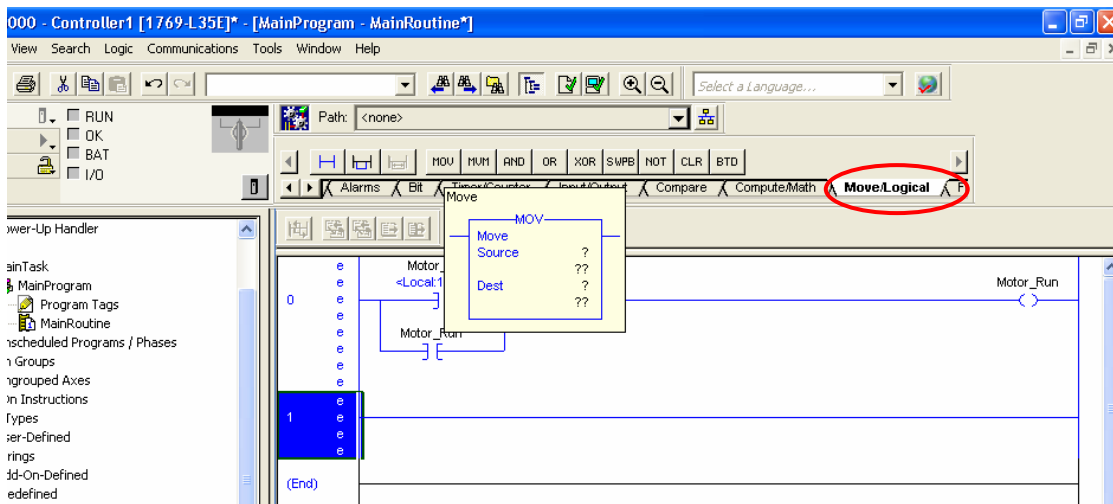
11. Click **OK** to close the wizard. The analog module should now appear in the I/O Configuration of your Controller Organizer.



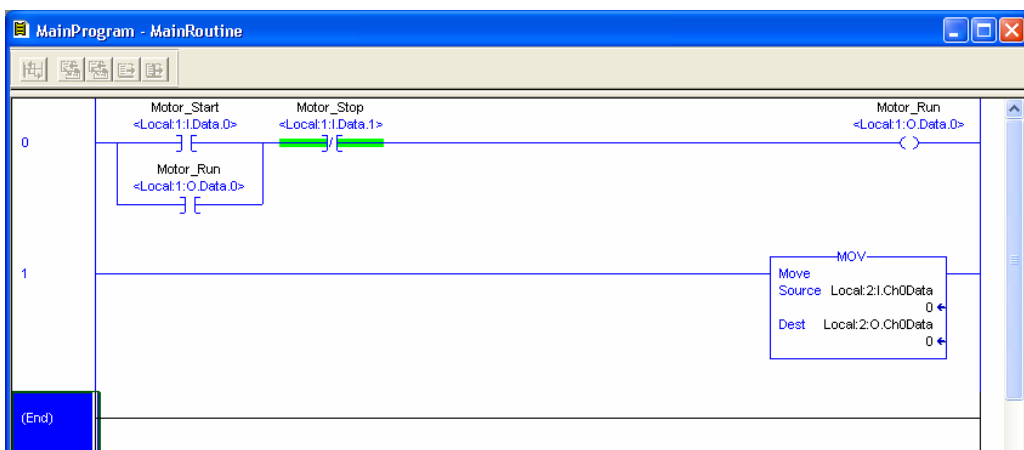
- Double click to open the Main Routine and add a rung by clicking the rung button  on the toolbar.



- Again, using the toolbar, under the **Move/Logical** category tab, click and drag an **MOV** instruction to the new rung.




- Configure the MOV instruction as shown in rung 1 below.



17. Verify the program by clicking on the Verify Routine or Verify Controller  icon on the toolbar.

You will see if there are any errors in the status window.

18. **Save** the program by clicking on the Save icon  on the toolbar.

Congratulations! You have Completed Lab 2. Please move on to Lab 3.

Lab 3: Connecting Your Computer to the Controller (5 Minutes)


About This Lab


In this lab, we will introduce you to the online operations that you will complete with the RSLogix5000 software. In this lab, you will:

- Launch RSLinx communications software
- Configure your communications driver

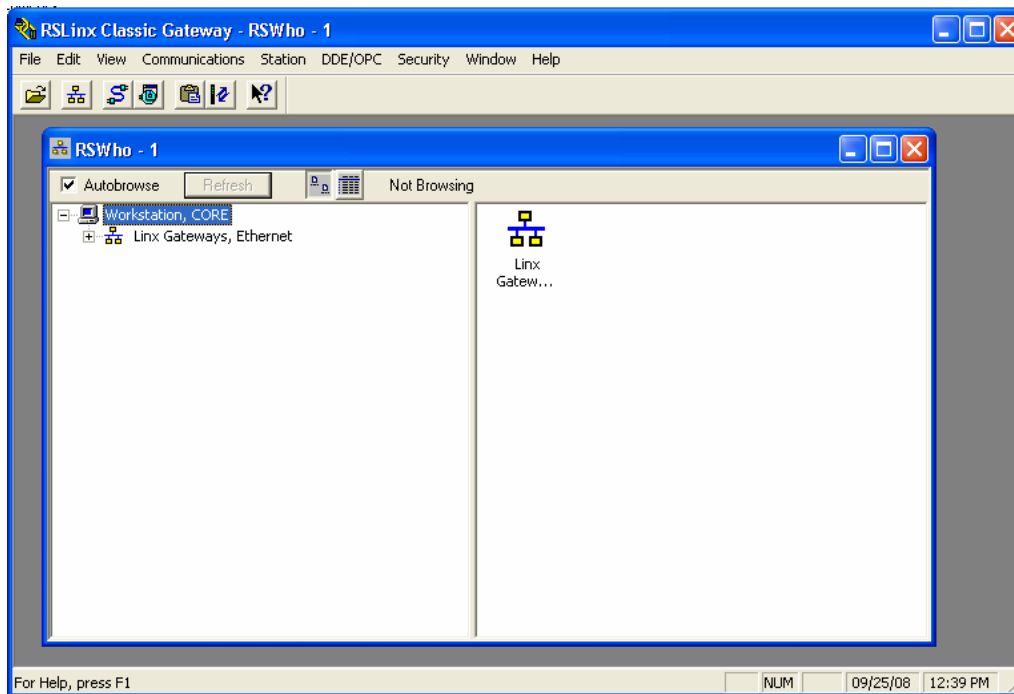
Launching RSLinx Software

In this section of the lab, you will launch the RSLinx software, which will enable you to configure the driver you will use to communicate with the Logix processor in the Demo Box.

1. Double click on the **RSLinx**  icon on the Desktop to launch RSLinx software.

2. Click the **RSWho** icon  in the toolbar.

The Rockwell Software RSLinx Gateway - [RSWho - 1] screen appears.



FYI

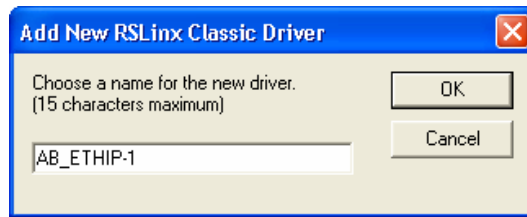
RSWho

The RSWho screen is actually RSLinx's network browser interface, which allows you to view all of your active network connections.

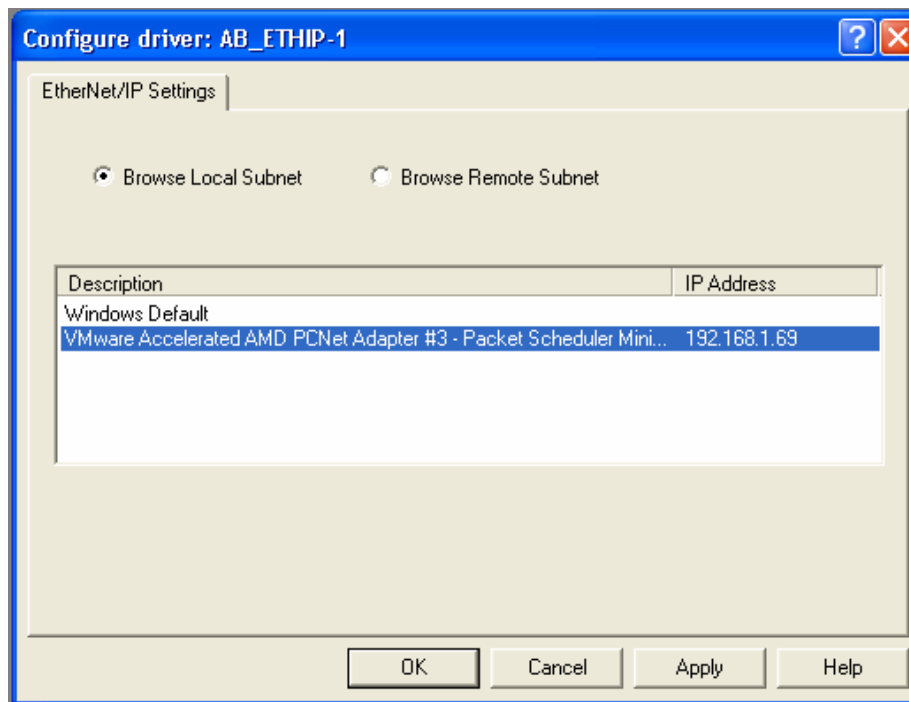
The left pane of this display is the Tree Control, which shows networks and devices in a hierarchical view. When a network or device is collapsed, as indicated by the + sign, you can click on the + sign or double click on the network or device icon to expand the view and begin browsing. When a network or device is expanded, as indicated by the - sign, you can click on the - sign or double click on the network or device icon to collapse the view.

The right pane of the RSWho display is the List Control, which is a graphical representation of all of the devices present on the network.

3. Click on **OK** to accept the default name (AB_ETHIP-1).



4. Ensure that the **Browse Local Subnet** radio button is enabled, click the VMWare address (this is the address of your VMWare image) and then click **OK**.



5. Exit the Configure Driver Dialog by clicking on **Ok**.

Congratulations! You have Completed Lab 3. Please move on to Lab 4.

Lab 4: Downloading the Project from the Computer to the Controller (10 minutes)

About This Lab

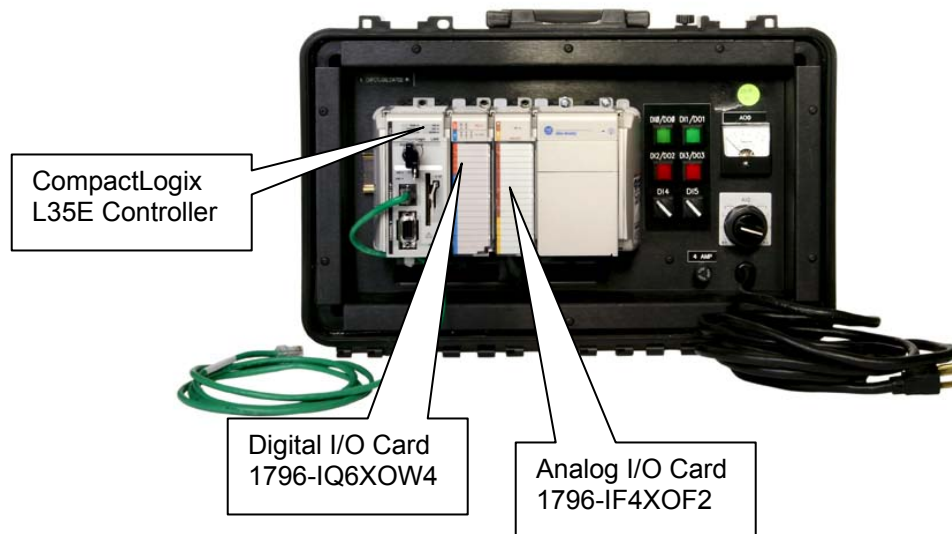
In this lab you will open a controller project based on the lab station at which you are seated.

You will:

7. Determine the type of controller you are using
8. Open the project that corresponds to the controller you are using
9. Download the program to the controller

You will be using the program that was created from the steps performed in Lab 1.

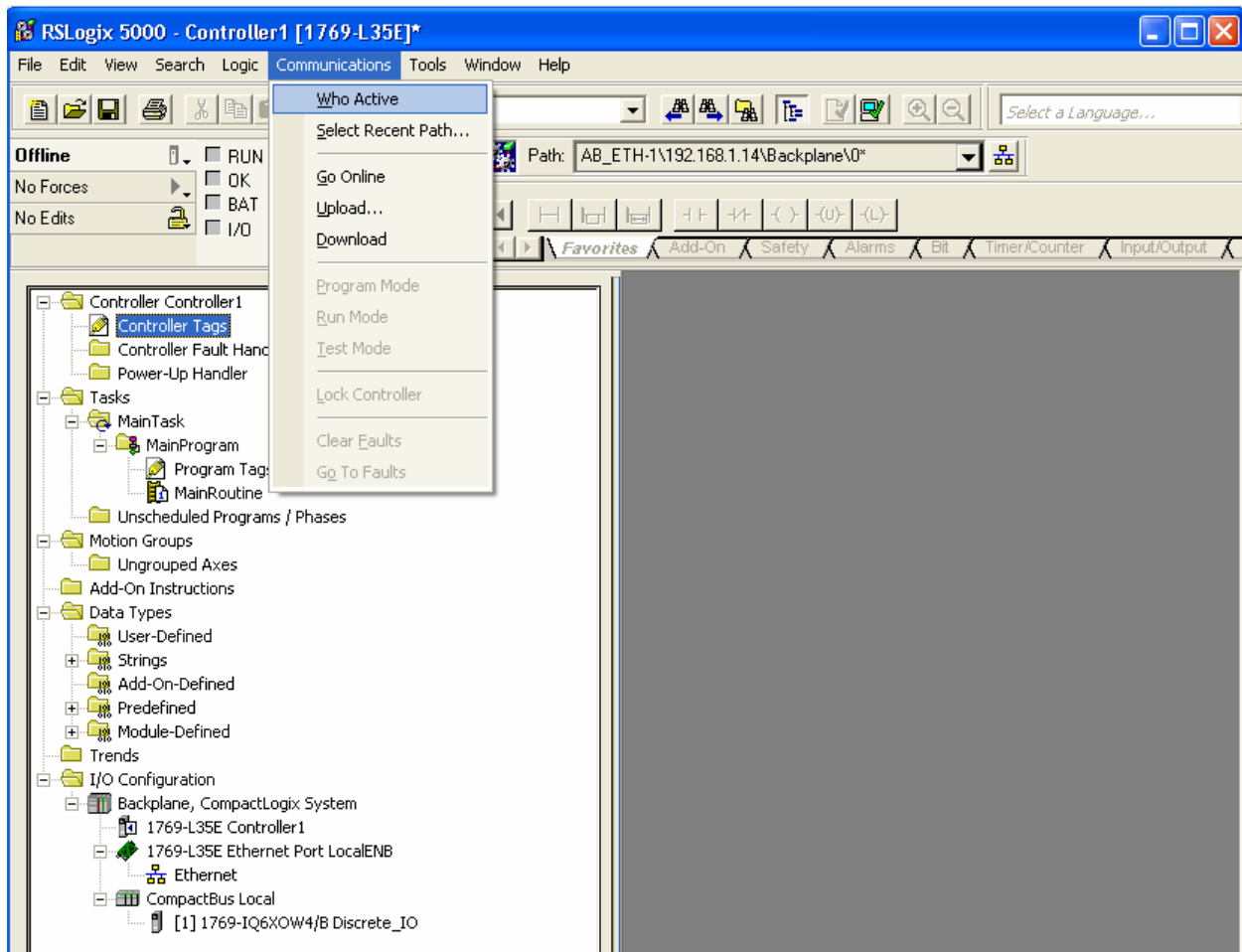
Look over the images below if you are unsure of the name associated with your lab station demo.



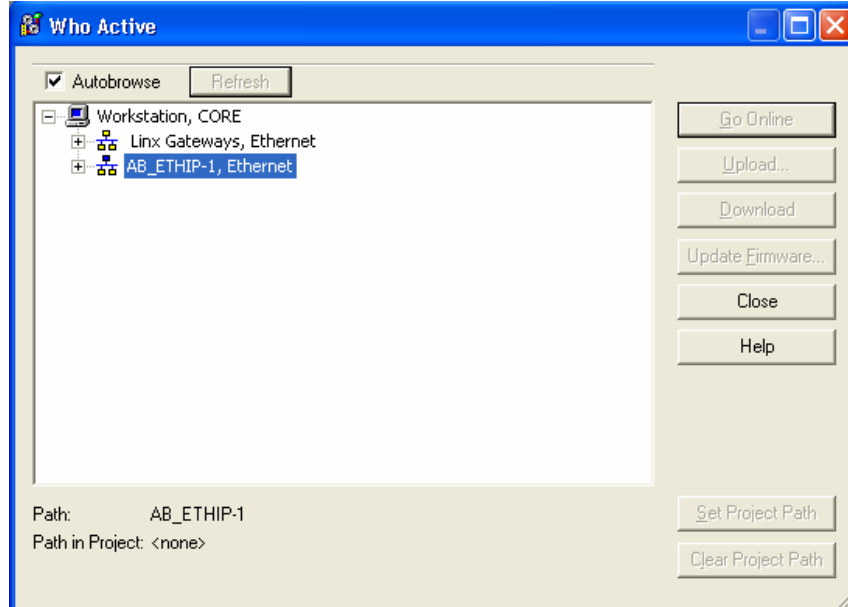
Downloading the Project to the Controller

In this section of the lab you will download the project.

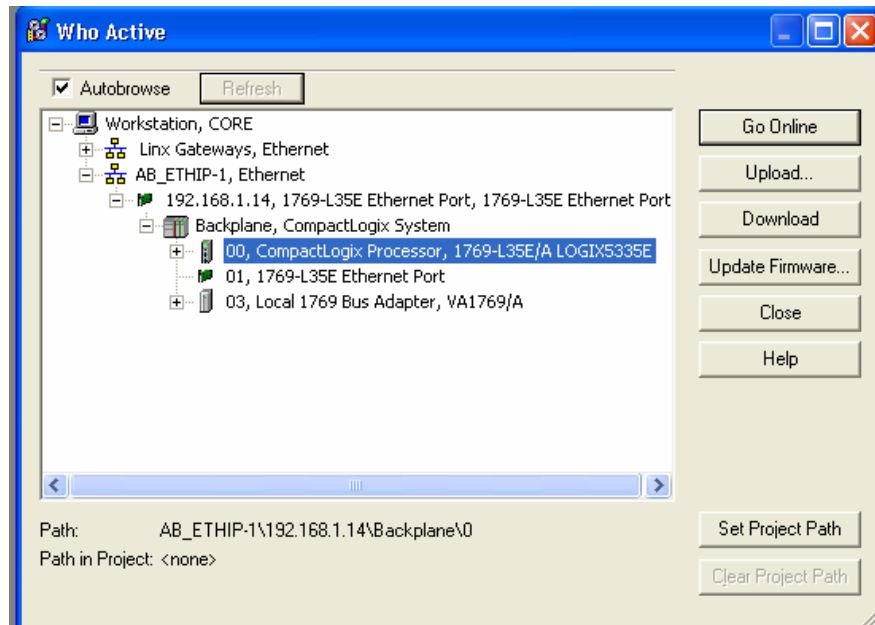
1. Maximize RSLogix 5000 and your Controller1.ACD project.
2. From the **Communications** menu, choose **Who Active**.



The **Who Active** Screen appears.



3. Expand the view by clicking on the '+'s until you see your controller.

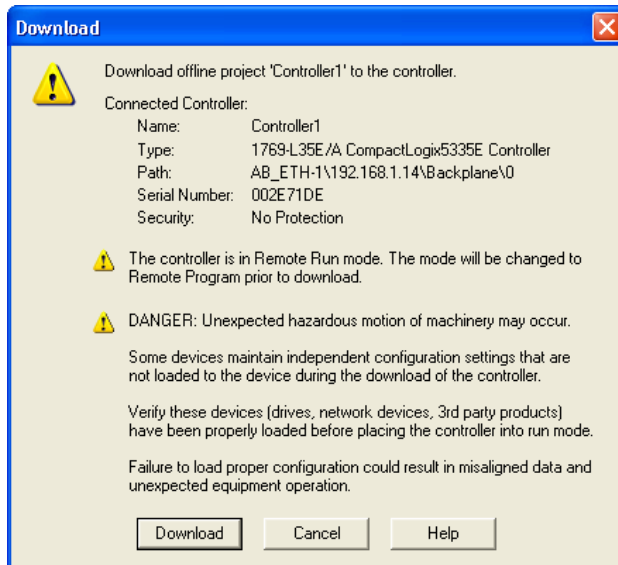


The Logix family of controllers all use, RSLogix 5000 software to configure the system. But each controller is set up slightly differently.

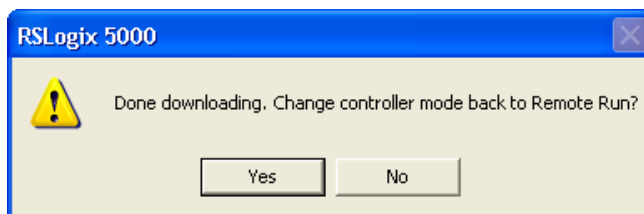
4. Click **Download**. You will be asked to verify the download.

The project will then begin to download to your controller.

If your controller was in the RUN mode prior to the download, you may be prompted to return to the RUN mode. If asked select **YES**.



5. When the following prompt appears, click **Yes** to change the controller mode to Remote Run.



At this point you will be online with the controller and the status LEDs in your project will mimic the LEDs on your controller.

Congratulations! You have Completed Lab 4. Please move on to Lab 5.

Lab 5: Testing Your Logic Program (5 Minutes)

About This Lab

In this lab you will verify the operation of your program.

FYI

I/O Mapping

For the lab there are a group of push buttons on the Demo Box. The push buttons are mapped as follows:

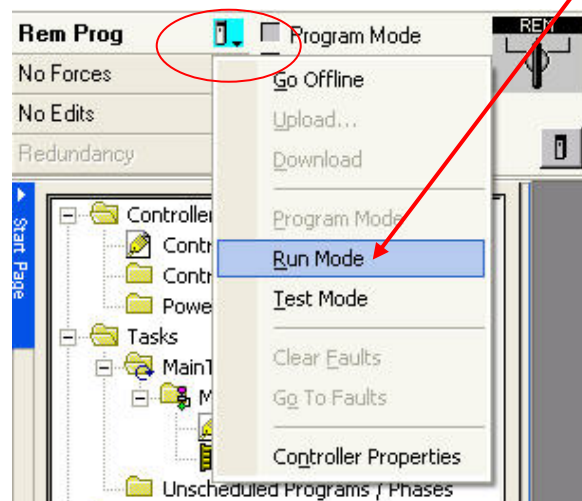
Motor_Start = DI0

Motor_Stop = DI1

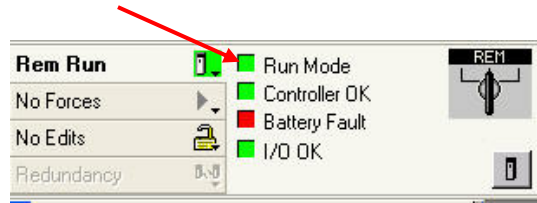
Motor_Run = DO0

Switching the Controller into Run Mode and Testing the Program

1. If not already in run mode, click the **Controller Faceplate** and select **Run Mode**.

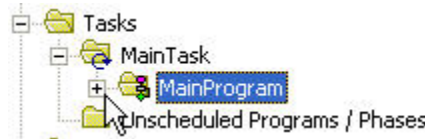


The controller will go into run mode. This can be verified by looking at the Run LED on the controller. It should now illuminate green. It can also be verified through RSLogix 5000 by viewing the controller faceplate.

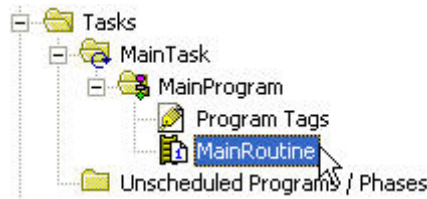


You notice that this is a replica of your controller's faceplate.

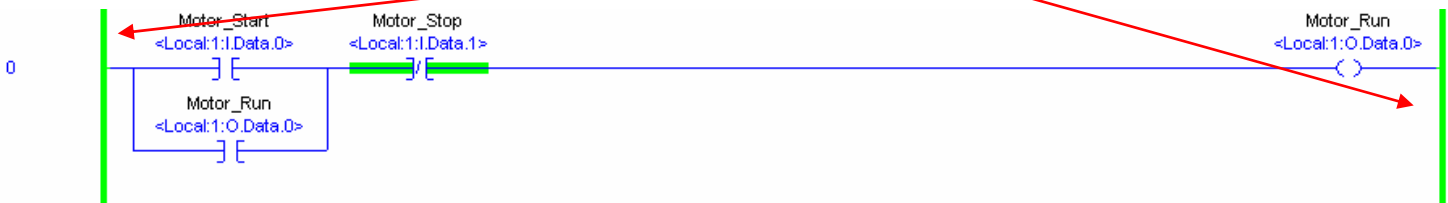
2. From the Controller Organizer expand the **MainProgram** by clicking on the "+".



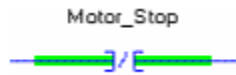
3. Double-click on the **MainRoutine** to open the ladder editor.



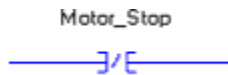
You will now see the ladder logic. Notice the green power rails on both sides of the ladder. This indicates you are online and the routine is executing.



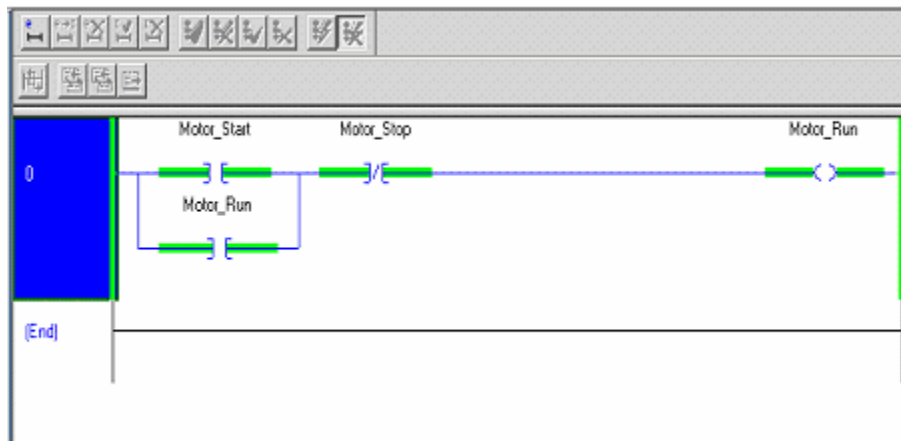
You notice that the XIO instruction **Motor_Stop** is green. This means that this instruction is in the 'true' or 'on' state. This is because the Motor_Stop Pushbutton is not pressed.



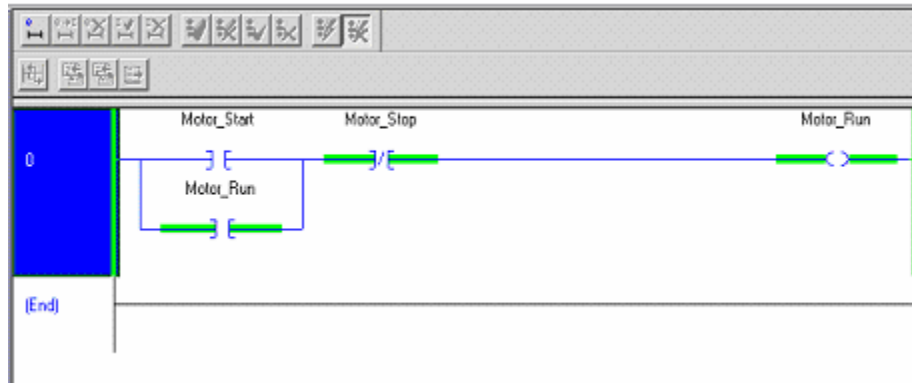
4. Press button **D11** button on the ControlLogix pushbutton panel.
This correlates to the XIO instruction for Motor_Stop. Notice it's no longer be green. This is because the instruction is no longer true.



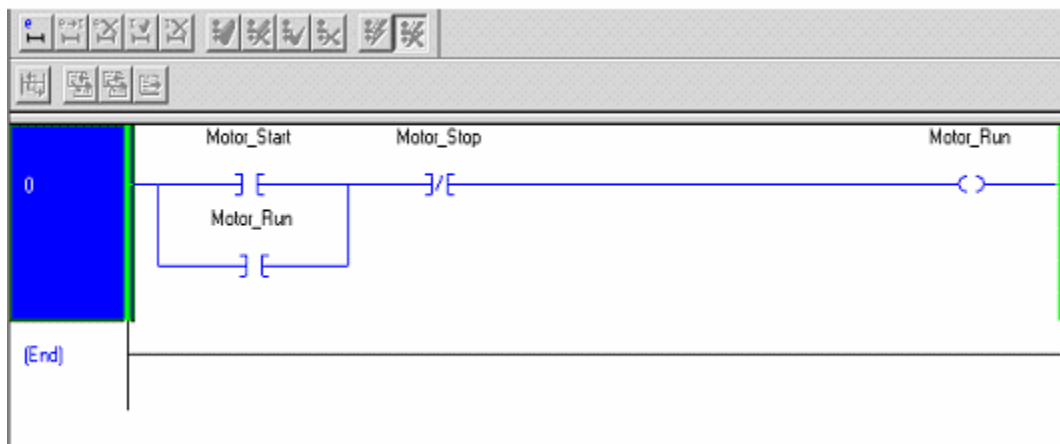
5. Press button **D10** (Motor_Start).
The XIC instruction will become true and turn green. Motor_Run will energize (turn green). And the pilot light DO0 on your lab station will illuminate.



6. Verify that output **DO0** (Motor_Run) stays illuminated when you release pushbutton **DI0** (Motor_Start).
The ladder logic you have just written is a simple 3-wire control or motor start/stop seal-in circuit.



7. Press pushbutton **DI1** (Motor_Stop) and verify that output **DO0** (Motor_Run) turns off.



Congratulations! You have Completed Lab 5. Please move on to Lab 6.

Lab 6: Adding Logic and Tags Online (15 Minutes)

About This Lab

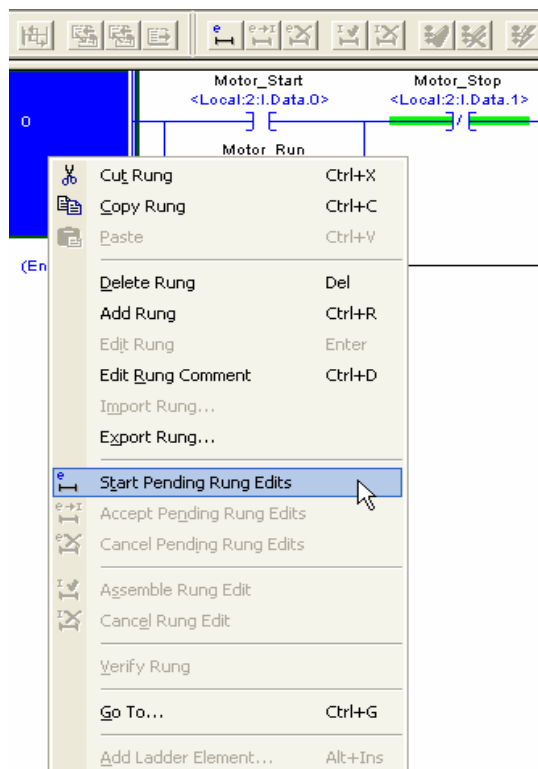
In this lab you will perform online editing. You will:

10. Add a timer to the logic and its execution will be based on the motor running
11. Add ladder logic to reset the timer when the motor is stopped.

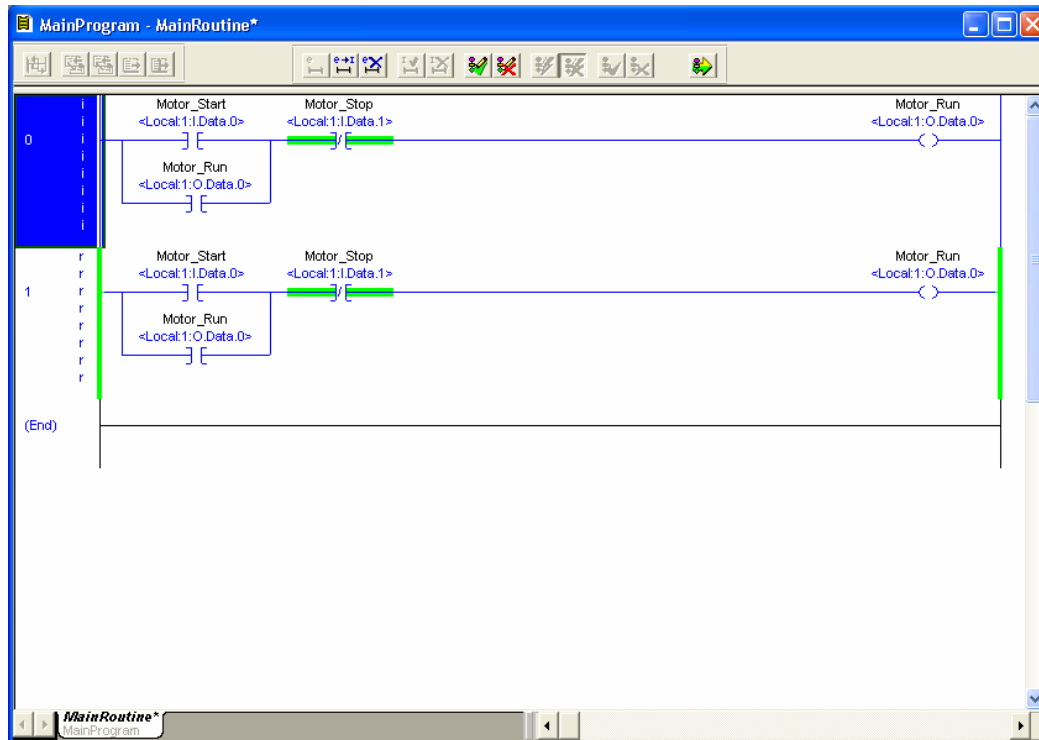
You will continue to use the project already opened.

Adding the Timer to the Logic

1. Right click in the **blue highlighted area** to the left of rung zero and select **Start Pending Rung Edits**.

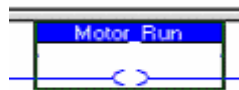


- The ladder editor will now look similar to the following:

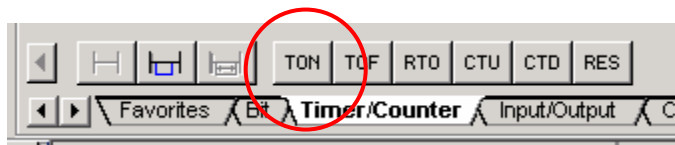


The rung with the '1's on the power rails is the rung you will perform the edits on.

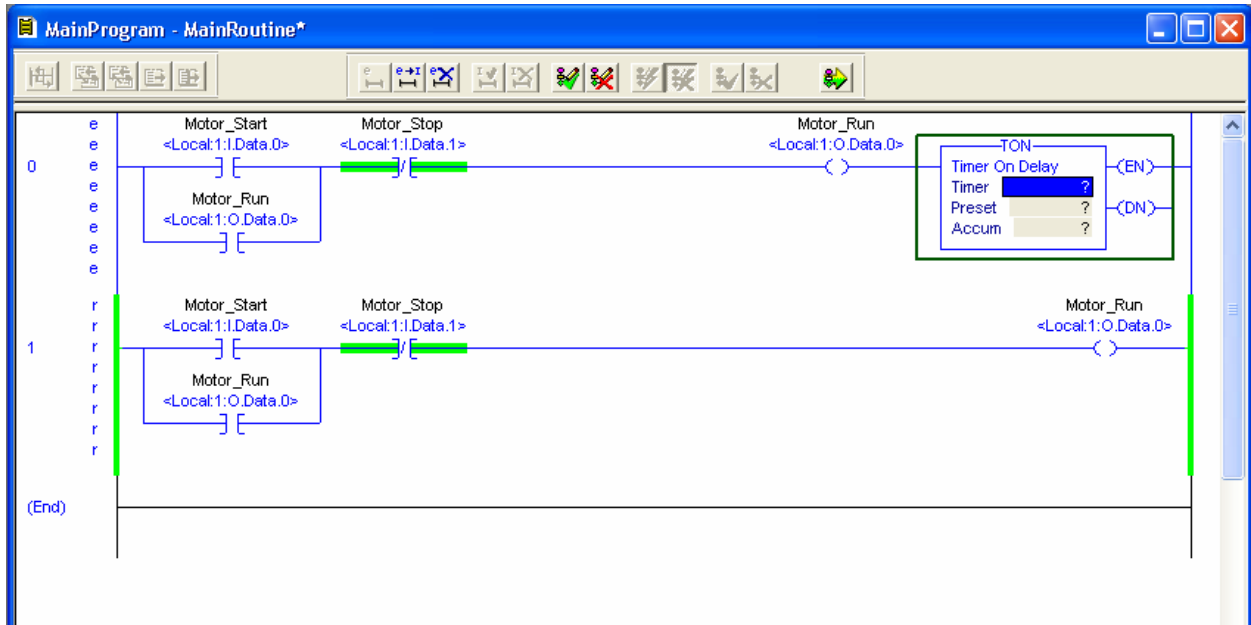
- Click the **OTE** instruction so it becomes highlighted.



- From the **Instruction Toolbar** click on the **Timer/Counter** tab, click the **Timer On (TON)** icon.

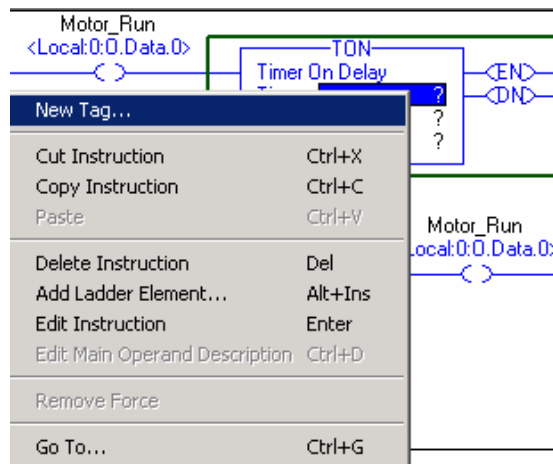


A timer is inserted into the code to the right of the OTE instruction.



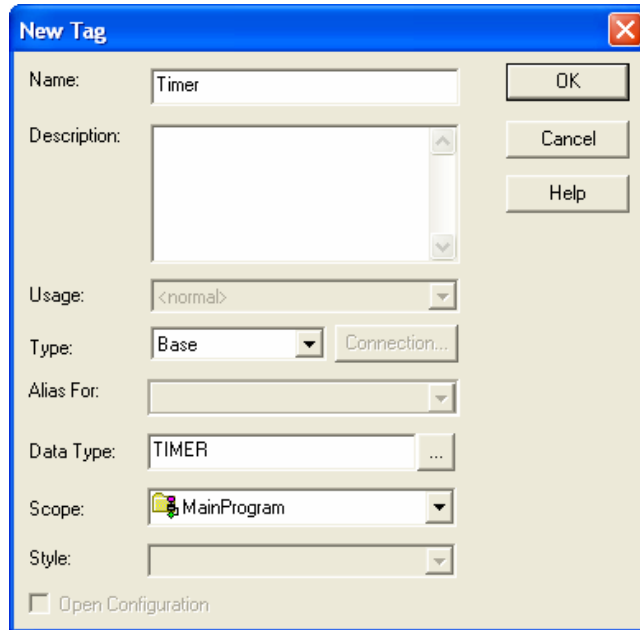
In **RSLogix 5000** you can string output instructions together. You do not have to create branches.

5. On the timer instruction right click in the **blue area** next to the word Timer and select **New Tag**.

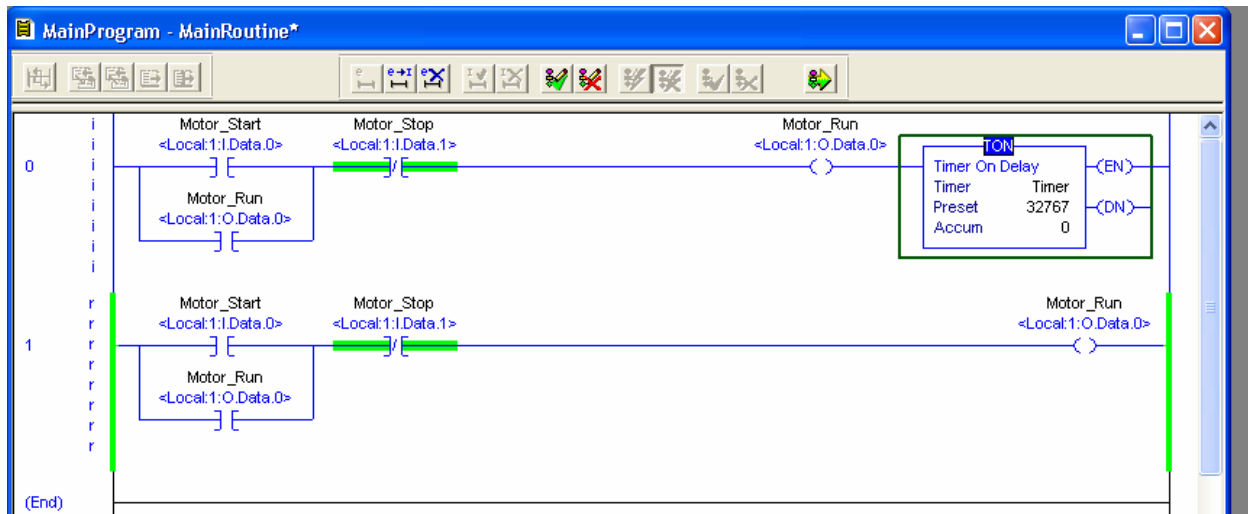


The New Tag window appears. You notice that the Data Type is already set to **TIMER**. This is because you are creating a tag in a timer instruction.

6. In the Name field enter '**Timer**' then click **OK**.



8. Verify that the tag has been created in the timer instruction as shown below:

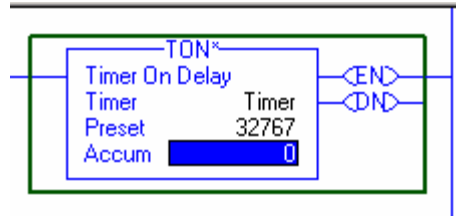


9. Double-click on the 0, in the timer instruction, next to the word **Preset**.

10. Enter a value of **32767**.

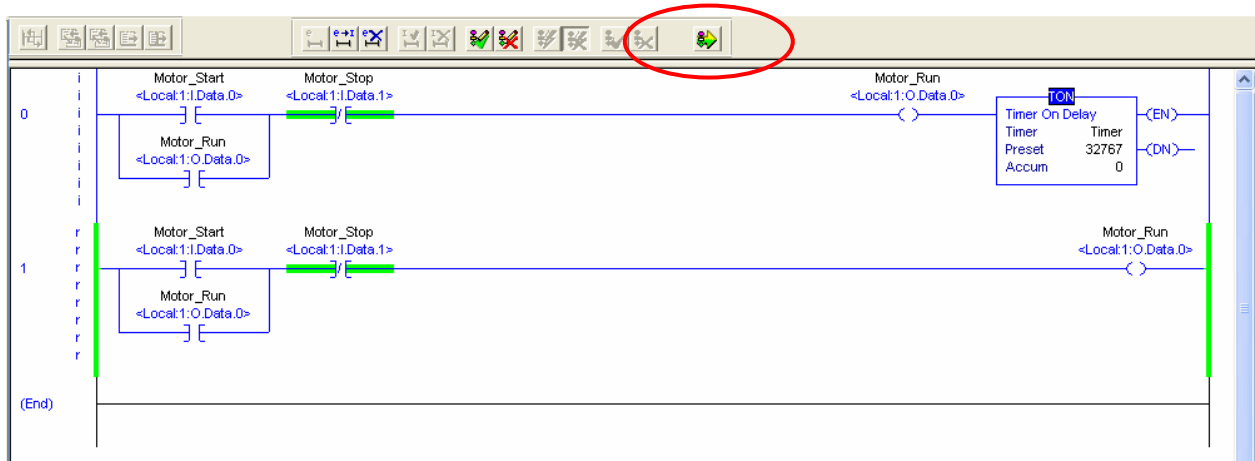
In Logix the Timer Preset is a 32-bit DINT which means the maximum value for your timers can be: 2,147,483,647

11. Press **Enter**. Your TON instruction should now appear as shown below.

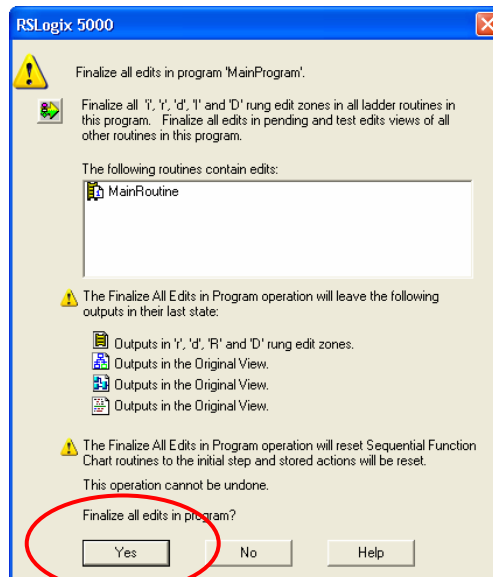


Your Preset value is now 32767 milliseconds. Leave the accumulated value set to zero. You are now ready to verify the edits you made.

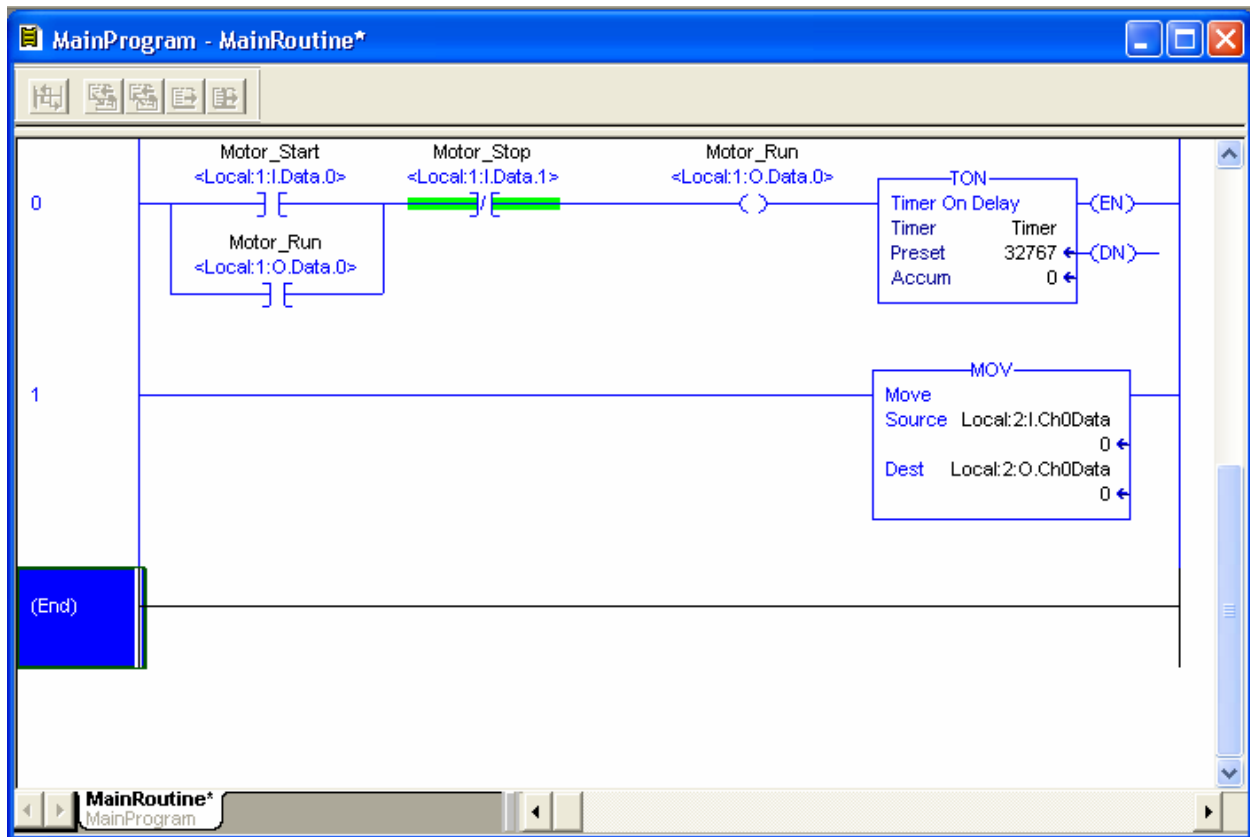
12. Click on the **Finalize All Edits** icon



13. When asked to finalize all edits click on **YES**.



The ladder editor will now appear as follows:



Testing Your Logic

1. Press the **DIO** (Motor_Start) pushbutton.
2. Verify that **DOO** (Motor_Run) illuminates and the Timer instruction starts incrementing.
3. Now, press push button **DI1** (Motor_Stop).
4. Verify that **DOO** turns off and the Timer resets.

Optional Step -

5. If you added an analog I/O module and the MOV instruction in lab 2, turn the analog dial (AI0) on your demobox and you will see the corresponding reading on the potentiometer (AO0)!

Congratulations! You have Completed Lab 6. Please move on to Lab 7.

Lab 7: Creating and Running a Trend (5 Minutes)

About This Lab

In this lab you will see the built-in trending capabilities of RSLogix 5000.

In this Lab you will:

12. Create a trend to watch the Timer instruction's accumulated value.

This will be done online with the program from the previous Lab.

FYI

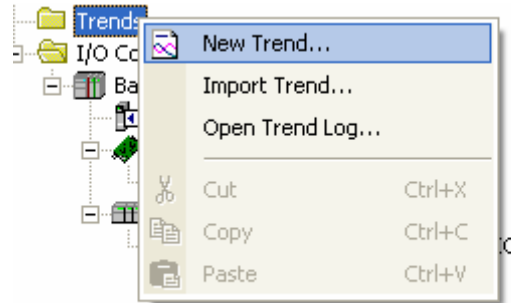
Trending

Basic Trending in RSLogix 5000 allows you to view data sampled over a time period in a graphical display. Data is sampled at a periodic rate that is configurable from 10 milliseconds to 30 minutes. RSLogix 5000 will allow you to create a trend and save it as part of your project file.

Basic Trending has these constraints: you can trend data elements of type BOOL, SINT, INT, DINT, and REAL, you are limited to sampling eight unique data elements, and you will be limited to one active trend at a time.

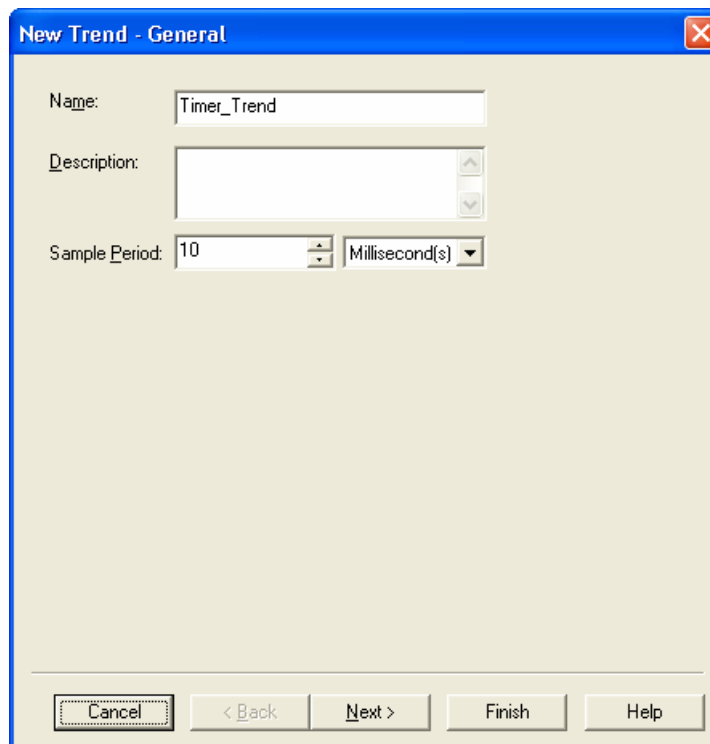
Creating and Running a Trend

1. From the Controller Organizer, right click on **Trends** and select **New Trend**.



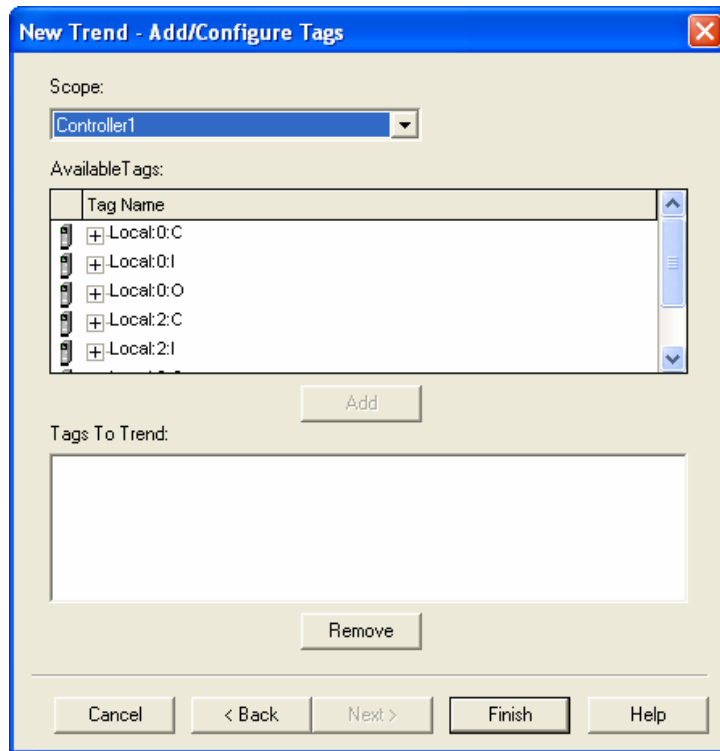
The **New Trend** window appears.

2. In the **Name** field enter '**Timer_Trend**'.

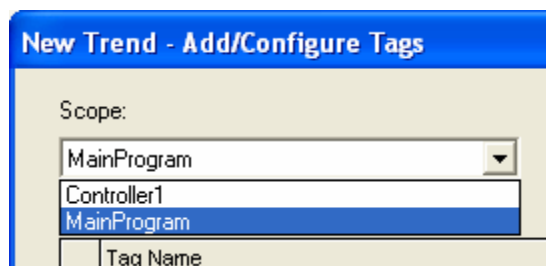


3. Click **Next**.

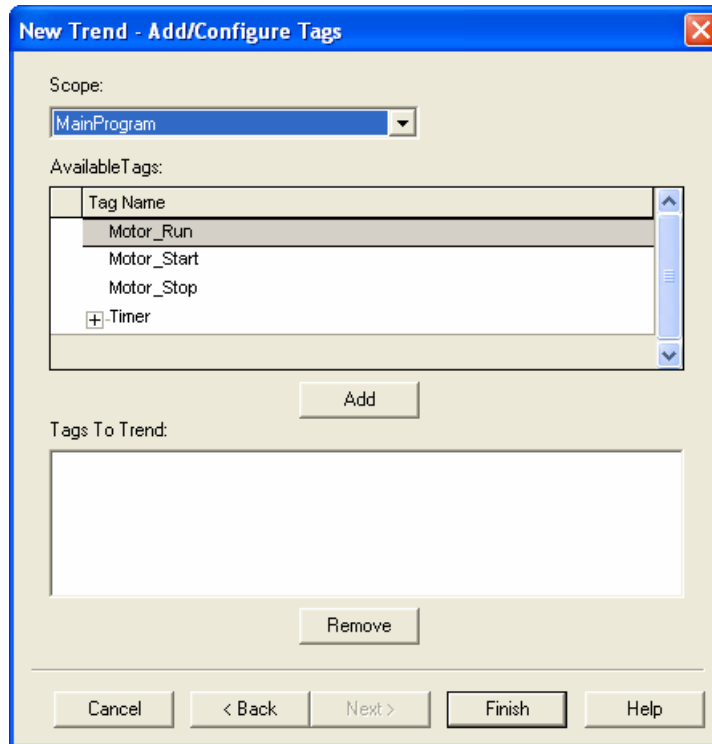
The **New Trend Add/Configure Tags** window appears.



We want to trend the timer accumulate value. When you added the timer the tag was created in the Program Scope, so we must select the **MainProgram** tags as shown below:



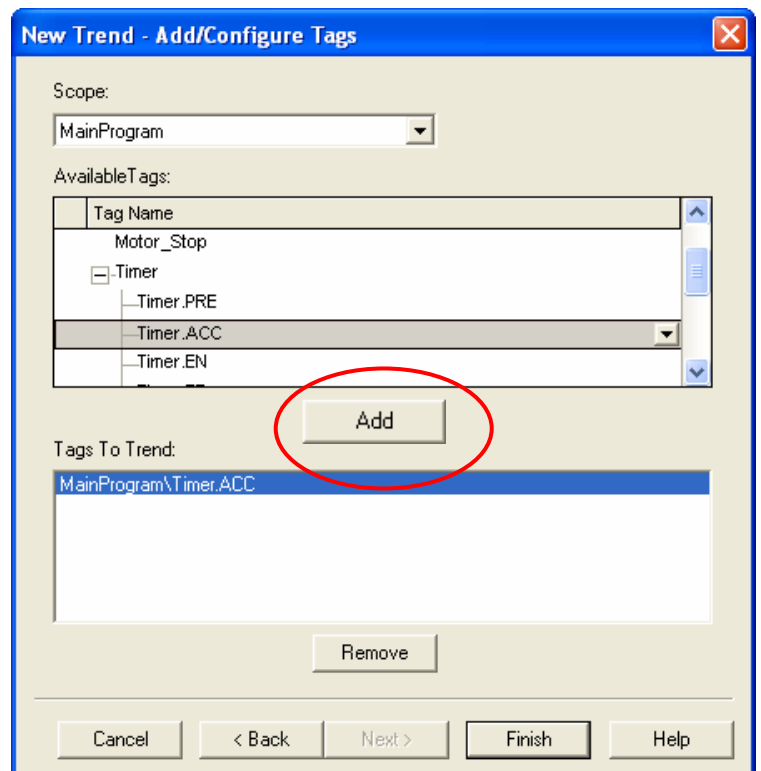
Now only the tags for the MainProgram are shown.



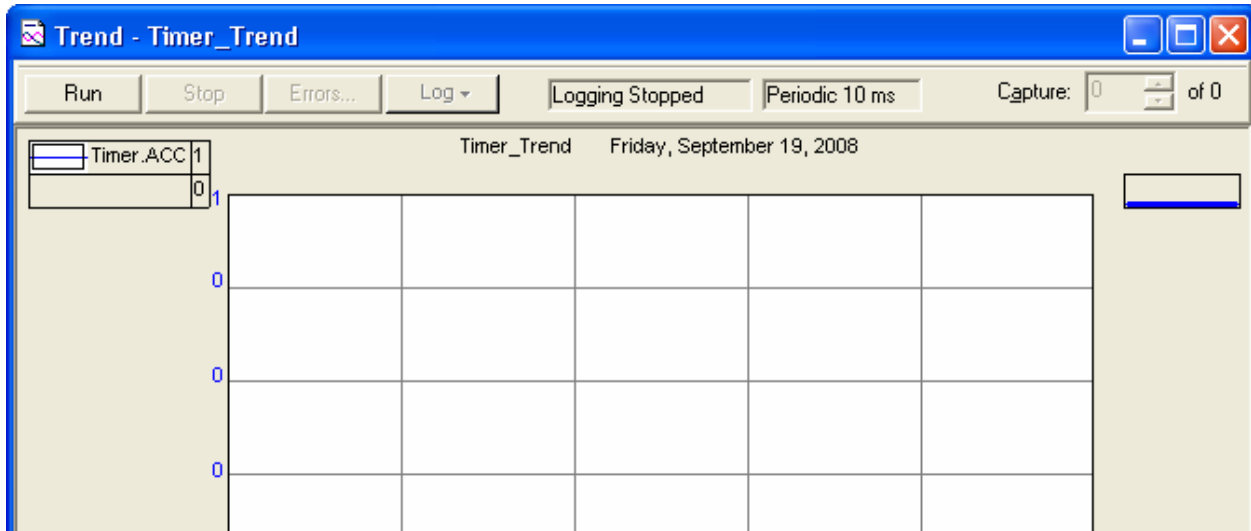
4. Expand the **Timer** tag by clicking on the +.

5. Select **Timer.ACC** and then click the Add button. This will add the tag Timer.ACC to the **Tags To Trend** list.

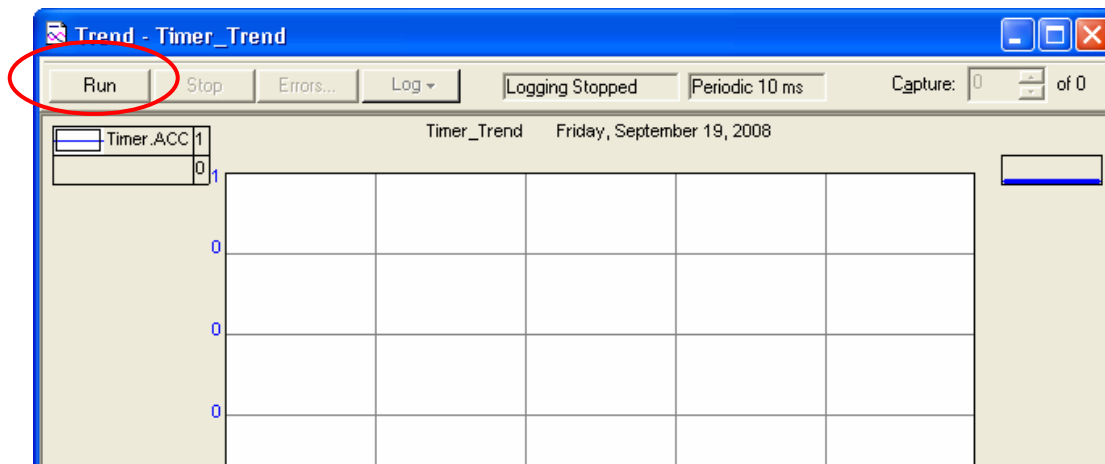
6. Click on **Finish**.



The Trend window will now appear.

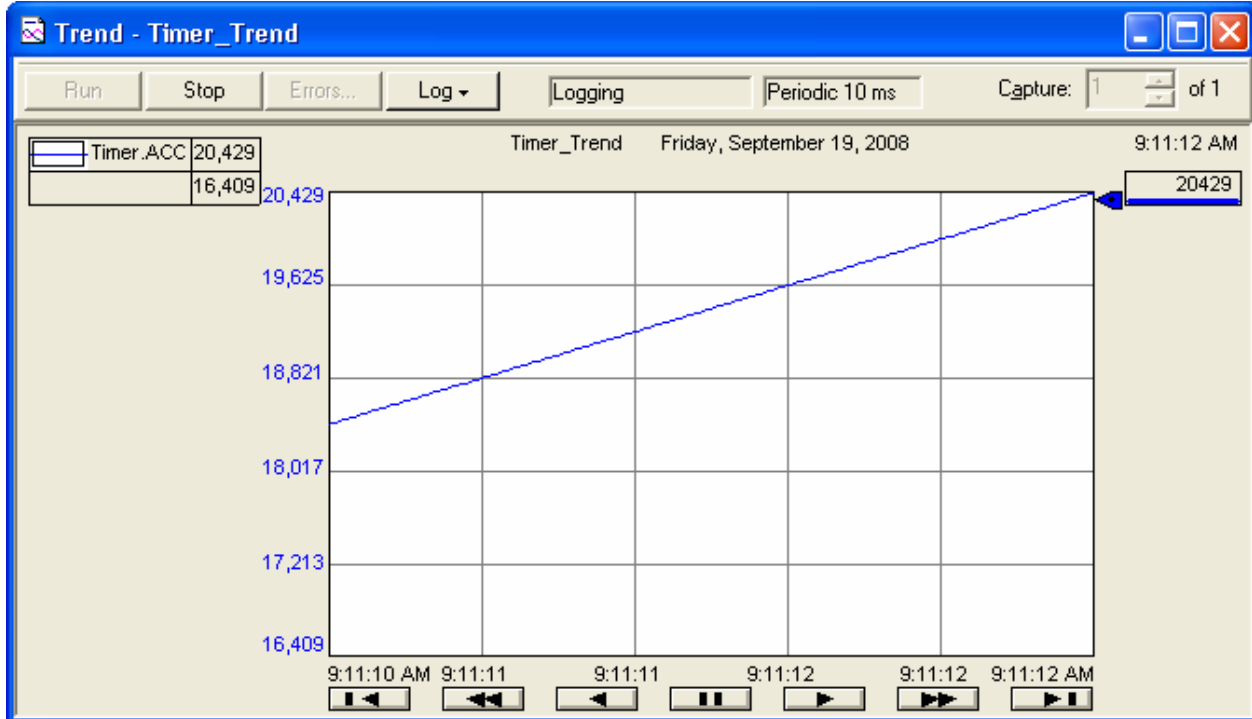


7. Start the trend by clicking on the RUN button located toward the upper left of the Trend dialog box.



8. Start the Timer in the program by pressing the **D10** pushbutton on your lab station.

9. Verify that you see the Trend begin capturing the data of the Timer.ACC as shown below:



10. Try pressing the **DI1** pushbutton and watch the trend.

11. When you are finished investigating the trend, click **Stop**.

Congratulations! You have Completed Lab 7. Please move on to Lab 8.

Lab 8: Using RSLogix 5000 Help (15 minutes)

About This Lab

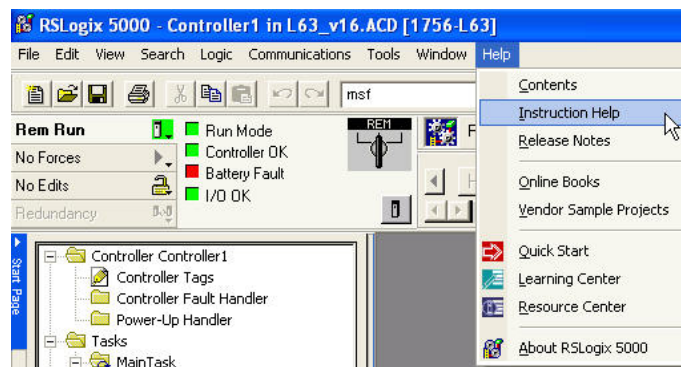
In this lab you will investigate the extensive online Help system in RSLogix 5000.

In this lab you will be viewing:

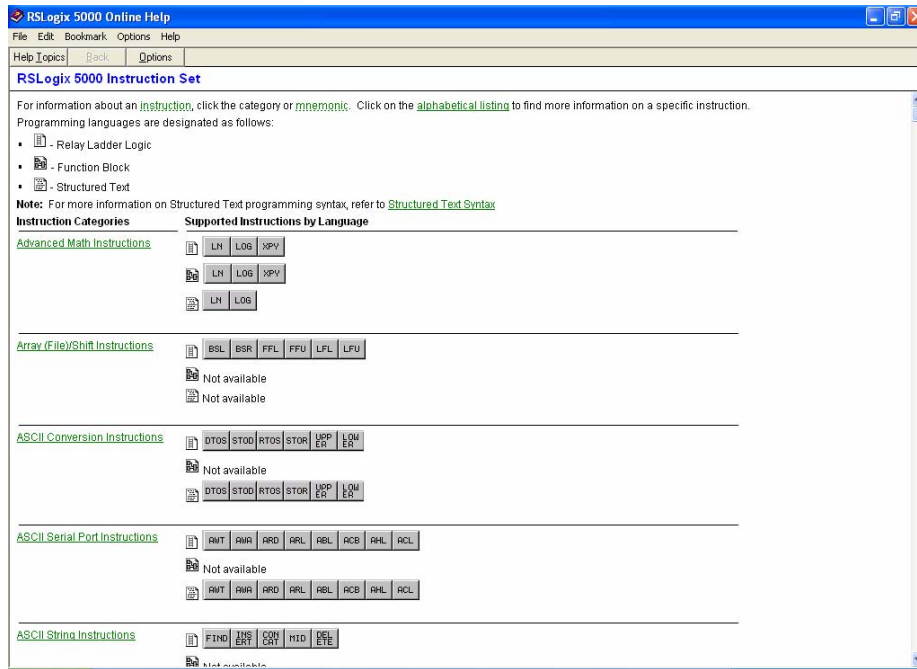
13. Instruction help
14. Module wiring diagrams
15. On-line reference materials
16. 3rd party vendor sample projects
17. The Start Page – Quick Start

Instruction Help

1. From the **Help** pull down menu select **Instruction Help**.



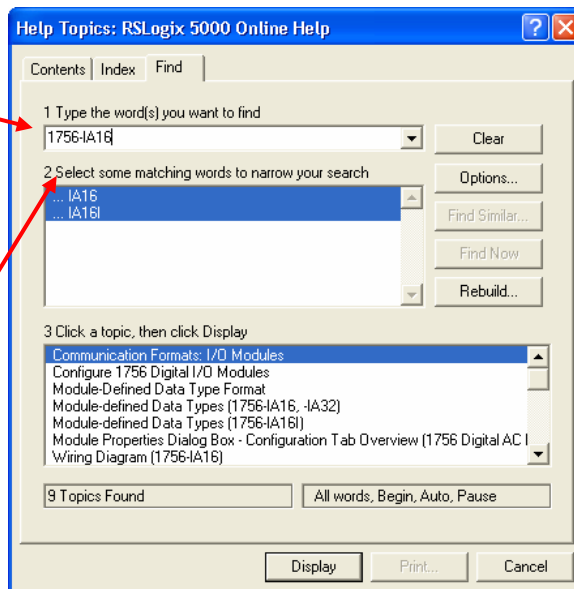
The following window will appear.



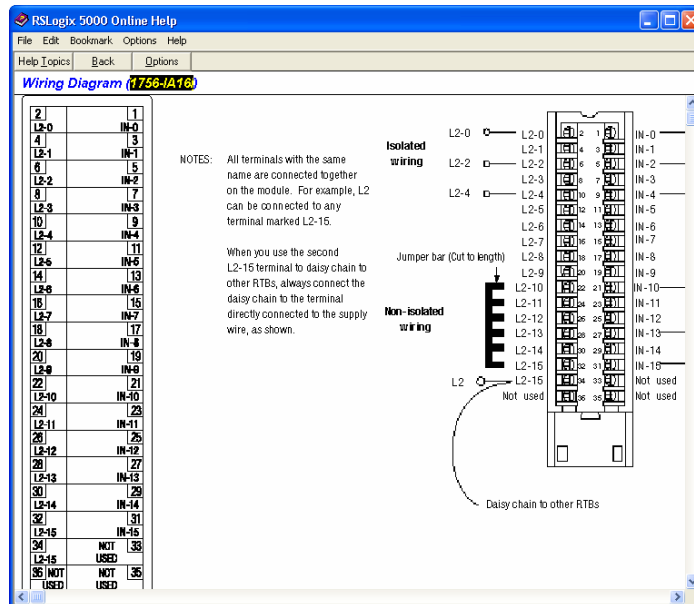
2. Click on an instruction to locate its description, details about its parameters, and related instructions along with examples on how to use the instruction.

Viewing I/O Module Wiring Diagrams

1. From the **Help** pull down menu select **Contents**.
2. Select the **Find** tab if it is not already selected.
3. Complete field 1 as shown below.



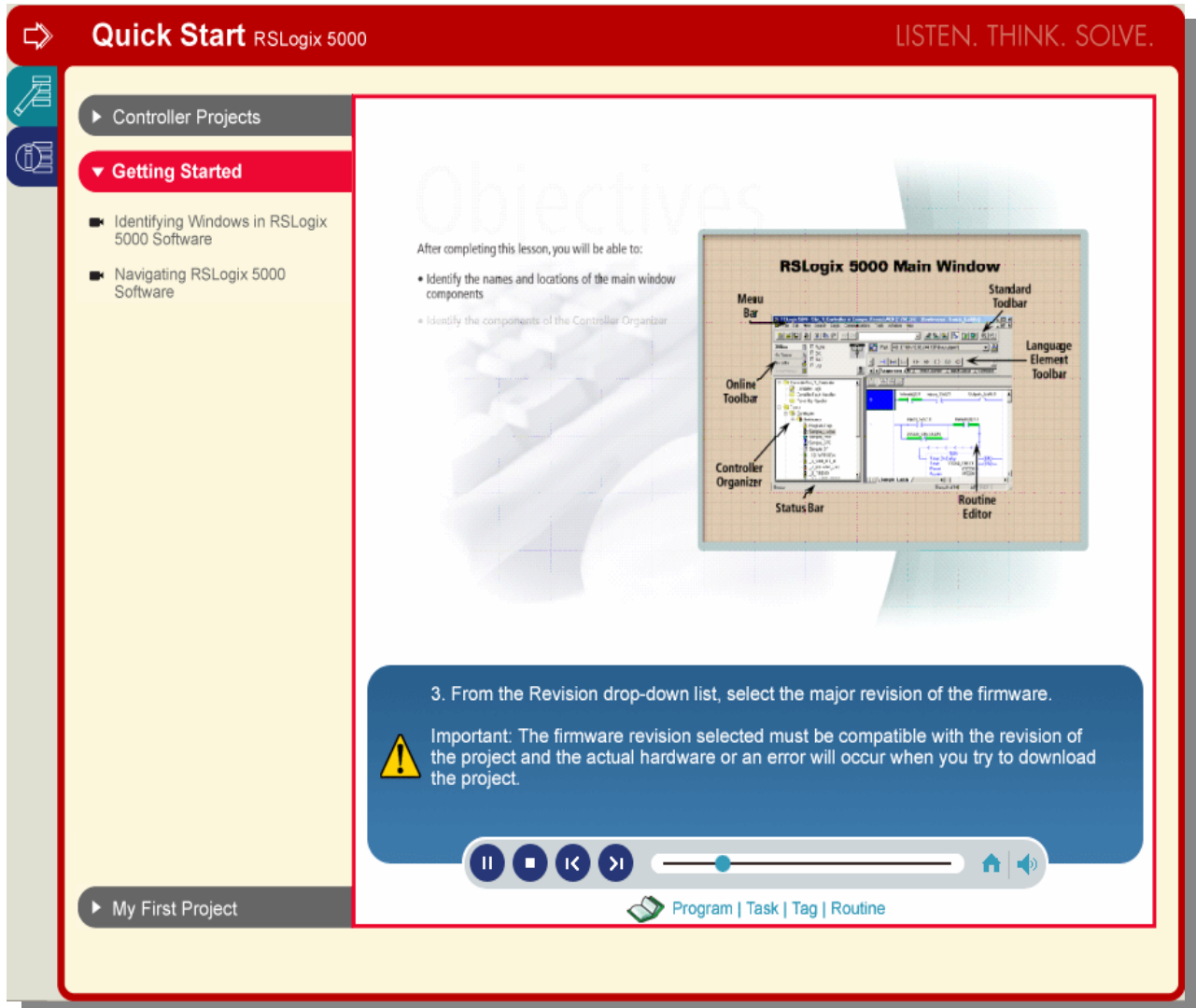
4. In field 2 select **IA16** as shown below.
5. In field 3 scroll down the list and locate **Wiring Diagram (1756-IA16)**.
6. Click **Display** to view the wiring diagram for this module. Note you may need to maximize your screen.



7. When you are finished viewing the wiring diagram close the display window.

Using Start Pages

From the **Help** pull down menu select **Quick Start**  which is one of the three tabs available from the **Start Page**.



Quick Start RSLogix 5000 LISTEN. THINK. SOLVE.

▶ Controller Projects

▼ Getting Started

- Identifying Windows in RSLogix 5000 Software
- Navigating RSLogix 5000 Software

▶ My First Project

Objectives


After completing this lesson, you will be able to:

- Identify the names and locations of the main window components
- Identify the components of the Controller Organizer

RSLogix 5000 Main Window

Labels in the screenshot: Menu Bar, Standard Toolbar, Language Element Toolbar, Online Toolbar, Controller Organizer, Status Bar, Routine Editor.

3. From the Revision drop-down list, select the major revision of the firmware.

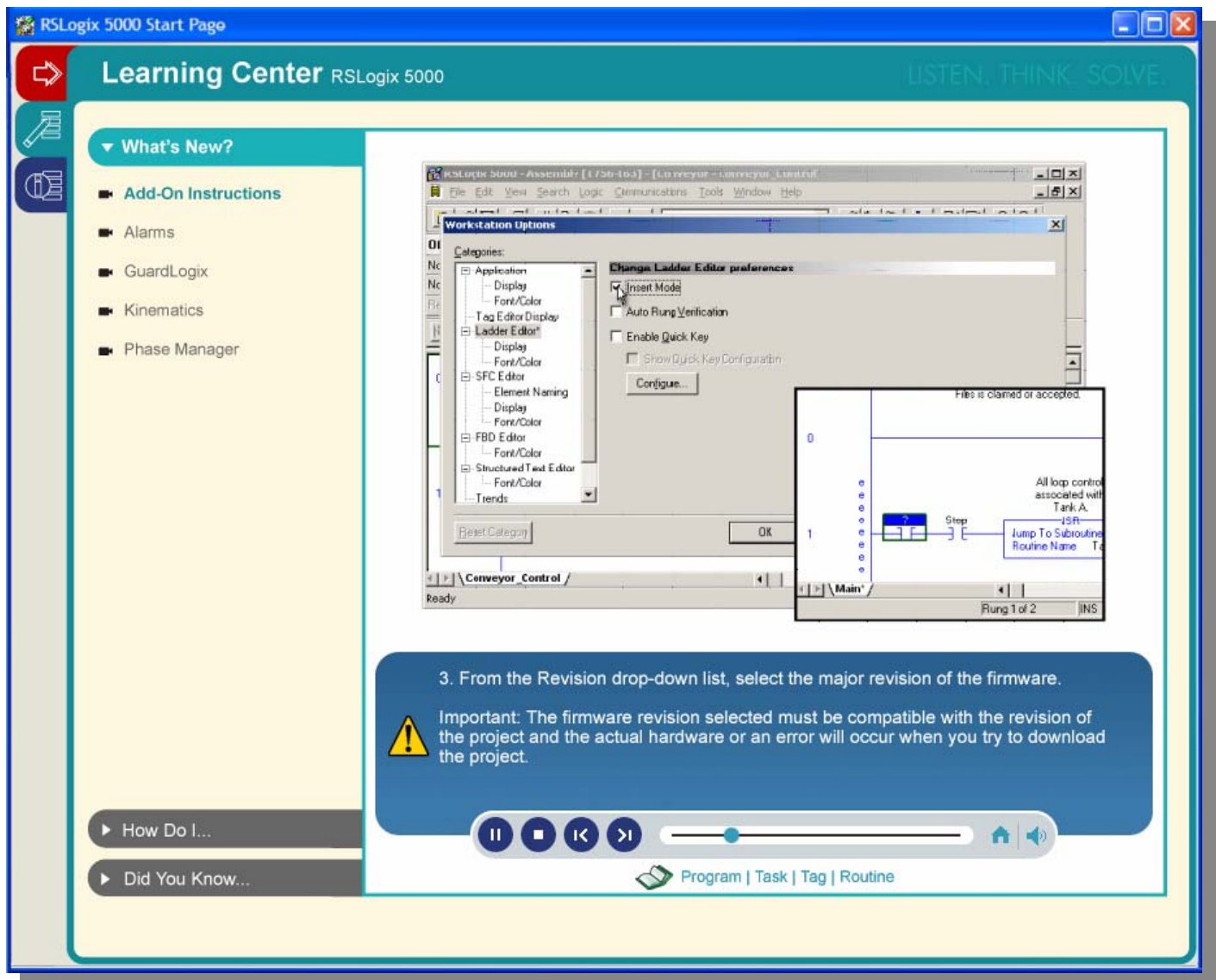
 Important: The firmware revision selected must be compatible with the revision of the project and the actual hardware or an error will occur when you try to download the project.

Media controls: Play, Stop, Back, Forward, Progress slider, Home, Speaker.


Program | Task | Tag | Routine

- Organizes various resources intended to accelerate the customer's ability to use the software and to locate relevant information
- Provides Getting Started and My First Project media clips and tutorials to assist new users
- Provides easy navigation to RSLogix 5000 sample projects Rockwell Automation specific and those involving other vendors

- Targets customers wanting to learn or explore how to use the software beyond just getting started reduces learning curve and helps increase productivity.



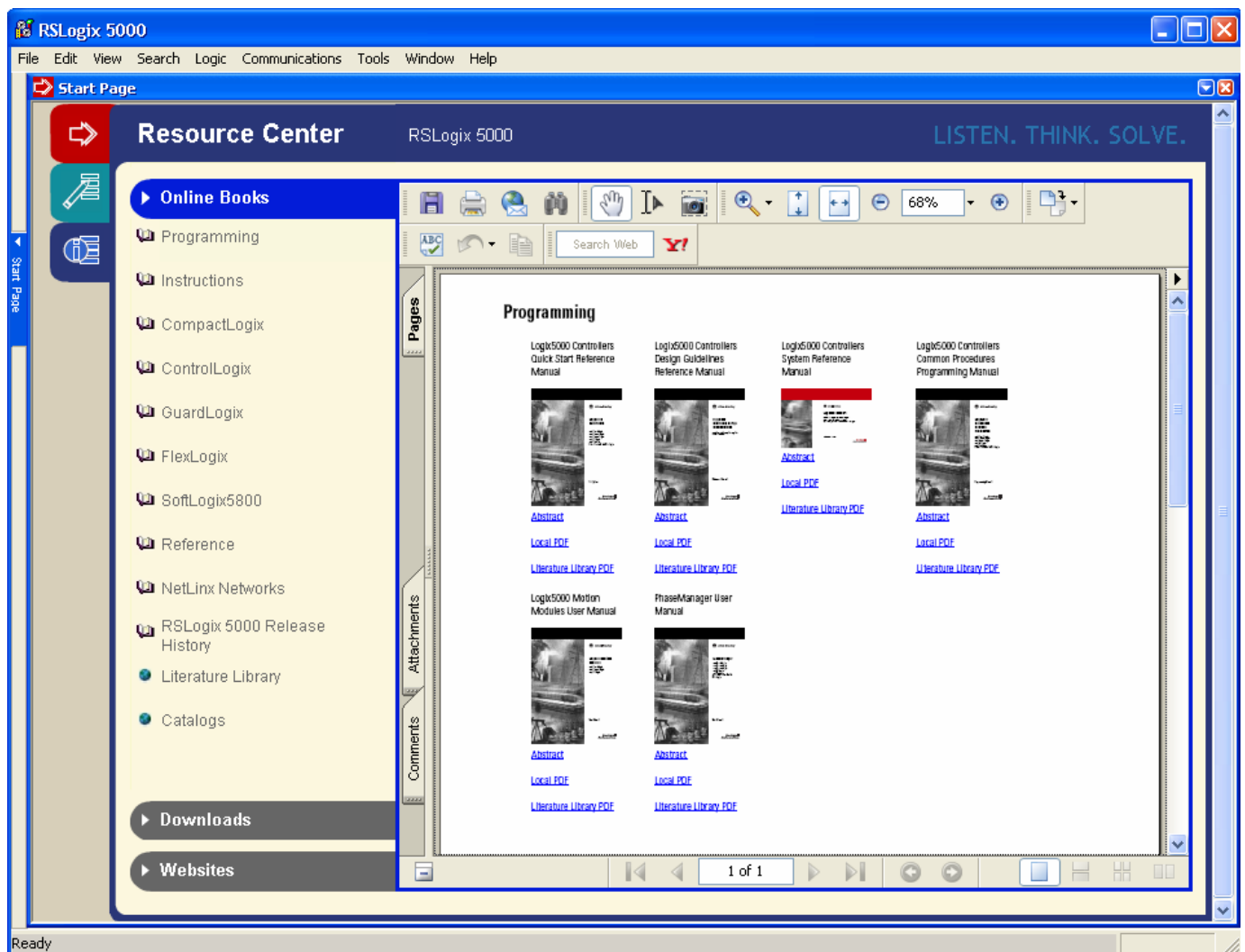
3. From the Revision drop-down list, select the major revision of the firmware.

 Important: The firmware revision selected must be compatible with the revision of the project and the actual hardware or an error will occur when you try to download the project.

- **What's New** media clips or tutorials previewing new features
- **How Do I** media clips or tutorials organized under various topics to show the user how to use the software to complete common tasks
- **Did You Know** tips / tricks for using the software, e.g. Keyboard Shortcuts

Resource Center

- Targets a customer looking for additional information or support
- Provides links to Download sites for software, firmware, EDS files, etc
- Provides links to Support sites Knowledgebase, Technical Bulletins, Sample Code
- Provides links to Online books installed to the PC with RSLogix 5000.



Thank you for attending Automation Fair 2009!

Notes :