Allen-Bradley

CompactLogix™ System

(1769-L31, 1769-L32E, 1769-L35E)

User Manual

Rockwell Automation

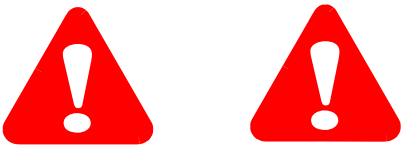
Important User Information

Because of the variety of uses for the products described in this publication, those responsible for the application and use of this control equipment must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards.

The illustrations, charts, sample programs and layout examples shown in this guide are intended solely for purposes of example. Since there are many variables and requirements associated with any particular installation, Allen-Bradley does not assume responsibility or liability (to include intellectual property liability) for actual use based upon the examples shown in this publication.

Allen-Bradley publication SGI-1.1, *Safety Guidelines for the Application, Installation and Maintenance of Solid-State Control* (available from your local Allen-Bradley office), describes some important differences between solid-state equipment and electromechanical devices that should be taken into consideration when applying products such as those described in this publication.

Throughout this manual we use notes to make you aware of safety considerations:

ATTENTION Or **WARNING** 	Identifies information about practices or circumstances that can lead to personal injury or death, property damage or economic loss

Attention and warning statements help you to:

- identify a hazard
- avoid a hazard
- recognize the consequences

IMPORTANT	Identifies information that is critical for successful application and understanding of the product.

Summary of Changes

This document describes how to use the CompactLogix controller. Changes for this version of the document include:

- addition of 1769-L31 and 1769-L32E controllers
- addition of 1769-IF8, 1769-OF8C, and 1769-OF8V I/O modules to the power calculation worksheet (see page 2-4)
- improved instructions and new examples for using MSG instructions to send email (see page 3-22)
- Modbus support (see page 5-2)
- use of an isolator for the non-isolated, channel 1 on a 1769-L31 controller (see 5-4)
- information on improving browse times for DH-485 networks (see page 6-10)

Notes:

Who Should Use This Manual

Read this preface to familiarize yourself with the rest of the manual. This preface covers the following topics:

- who should use this manual
- how to use this manual
- related publications
- conventions used in this manual
- Rockwell Automation support

Use this manual if you are responsible for designing, installing, programming, or troubleshooting control systems that use Allen-Bradley CompactLogix™ controllers.

How to Use This Manual

As much as possible, we organized this manual to explain, in a task-by-task manner, how to install, configure, program, operate and troubleshoot a CompactLogix control system.

Related Documentation

These core documents address the Logix5000 family of controllers:

If you are:	Use this publication:
a new user of a Logix5000 controller This quick start provides a visual, step-by-step overview of the basic steps you need to complete to get your controller configured and running.	Logix5000 Controllers Quick Start publication 1756-QS001
an experienced user of Logix5000 controllers This system reference provides a high-level listing of configuration information, controller features, and instructions (ladder relay, function block diagram, and structured text).	Logix5000 Controllers System Reference publication 1756-QR107
any user of a Logix5000 controller This common procedures manual explains the common features and functions of all Logix5000 controllers.	Logix5000 Controllers Common Procedures publication 1756-PM001

CompactLogix-specific information is also available:

For	Read this document	Document number
Information on installing a 1769-L31 CompactLogix controller	1769-L31 CompactLogix Controller Installation Instructions	1769-IN069
Information on installing a 1769-L32E, -L35E CompactLogix controller	1769-L32E, -L35E CompactLogix Controller Installation Instructions	1769-IN020
Information on the CompactLogix Instruction Set	Logix5000 Controllers General Instruction Set Reference Manual	1756-RM003
Information on function block programming Logix controllers.	Logix5000 Controllers Process Control/Drives Instruction Set Reference Manual	1756-RM006
Execution times and memory use for instructions	Logix5000 Controllers Execution Time and Memory Use Reference Manual	1756-RM087
Information on installing, configuring, and using Compact Analog I/O modules	Compact I/O Analog Modules User Manual	1769-UM002
Information on using the 1769-ADN DeviceNet adapter	Compact I/O 1769-ADN DeviceNet Adapter User Manual	1769-UM001
Information on using the 1769-SDN DeviceNet scanner	Compact I/O 1769-SDN DeviceNet Scanner Module User Manual	1769-UM009
Information on grounding and wiring Allen-Bradley programmable controllers.	Allen-Bradley Programmable Controller Grounding and Wiring Guidelines	1770-4.1

If you would like a manual, you can:

- download a free electronic version from the internet at **www.theautomationbookstore.com**
- purchase a printed manual by:
  - contacting your local distributor or Rockwell Automation representative
  - visiting **www.theautomationbookstore.com** and placing your order
  - calling 1.800.963.9548 (USA/Canada) or 001.330.725.1574 (Outside USA/Canada)

Conventions Used in This Manual

The following conventions are used throughout this manual:

- Bulleted lists (like this one) provide information not procedural steps.
- Numbered lists provide sequential steps or hierarchical information.
- *Italic* type is used for emphasis.

Chapter 1

What Is CompactLogix?

Chapter 2

Placing, Configuring, and Monitoring Local I/O

Communicating with Devices on an EtherNet/IP Network

Chapter 3

Chapter 4

Communicating with Devices on a DeviceNet link

Chapter 5

Communicating with Devices on a Serial Link

Chapter 6

Communicating with Devices on a DH-485 Link

Appendix A

CompactLogix System Specifications

Appendix B

EtherNet/IP Diagnostics

Dynamic Memory Allocation in CompactLogix Controllers

Appendix

Notes:

What Is CompactLogix?

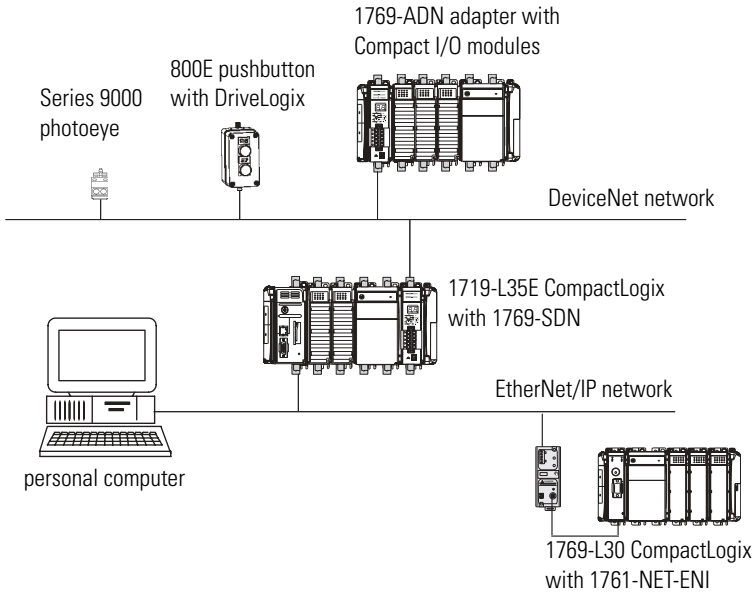
Using This Chapter

The CompactLogix controller, part of the Logix family of controllers, provides a small, powerful, cost-effective system built on these components:

- The CompactLogix controller is available in different combinations of communication options and user memory:

This controller:	Has this much memory:	With these communication options:
1769-L35E	1.5 Mbytes	1 port EtherNet/IP 1 port RS-232 serial (DF1, ASCII)
1769-L32E	750 Kbytes	1 port EtherNet/IP 1 port RS-232 serial (DF1, ASCII)
1769-L31	512 Kbytes	1 port RS-232 serial (DF1, ASCII) 1 port RS-232 serial (DF1 only)
1769-L30	256 Kbytes	1 port RS-232 serial (DF1, ASCII) 1 port RS-232 serial (DF1 only)
1769-L20	64 Kbytes	1 port RS-232 serial (DF1, ASCII)

- RSLogix 5000 programming software that supports every Logix controller.
- Compact I/O modules that provide a compact, DIN-rail or panel-mounted I/O system.
- The 1769-SDN communication interface module provides I/O control and remote device configuration over DeviceNet.



The newer 1769-L3xx controllers (1769-L35E, 1769-L32E, and 1769-L31) offer significant performance and capacity improvements over the 1769-L20 and 1769-L30 controllers. These 1769-L3xx controllers are designed for mid-range applications. They offer:

- increased user memory up to 1.5 Mbytes
- as many as 8 tasks (the 1769-L20, -L30 controllers support 4 tasks)
- CompactFlash for non-volatile memory storage
- extended I/O capacity up to 30 I/O modules
- increased backplane capacity and throughput resulting in the ability to mix and match any combination of digital, analog, and specialty I/O modules
- backplane messaging support
- integrated EtherNet/IP support, including control of distributed I/O
- increased I/O performance allows 1ms backplane RPI under certain conditions

For information about:	See page
loading controller firmware	1-2
developing programs	1-7
selecting a system overhead percentage	1-11

Loading Controller Firmware

The controller ships without working firmware. You must download the current firmware before you can use the controller. To load firmware, you can use:

- ControlFlash utility that ships with RSLogix 5000 programming software.
- AutoFlash that launches through RSLogix 5000 software when you download a project to a controller that does not have the current firmware.
- a 1784-CF64 CompactFlash card with valid memory already loaded.

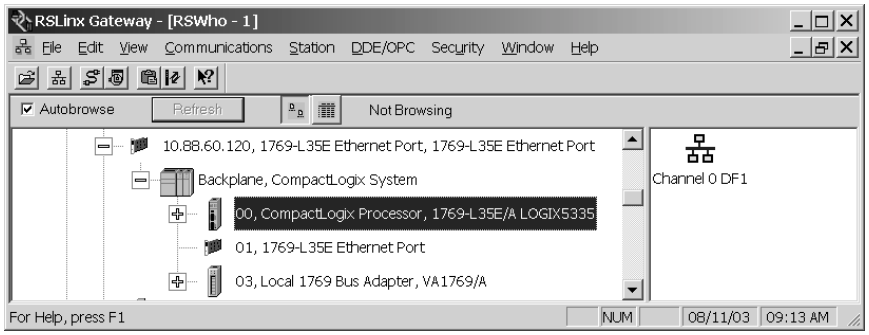
The firmware is available with RSLogix 5000 software or you can download it from the support website:

1. Go to http://support.rockwellautomation.com/
2. In the left column (frame), select Firmware Updates under Technical Support
3. Select the firmware revision.

   The download process will require you to enter the serial number of your RSLogix 5000 programming software.

If you load (flash) controller firmware via the ControlFlash or AutoFlash utilities, you need a serial or EtherNet/IP connection to the controller. Flashing via an EtherNet/IP connection is faster than the serial connection. The controller's EtherNet/IP configuration settings are maintained during a flash process.

If you load firmware via an EtherNet/IP connection, browse through the network port, across the virtual backplane, and select the appropriate controller.



Using ControlFlash to load firmware

You can use ControlFlash to load firmware through either an Ethernet connection (an IP address must already be assigned to the Ethernet port) or a serial connection.

1. Make sure the appropriate network connection is made before starting.
2. Start the ControlFlash utility. Click Next when the Welcome screen appears.
3. Select the catalog number of the controller and click Next.

4. Expand the network until you see the controller. If the required network is not shown, first configure a driver for the network in RSLinx software.

   If you use an Ethernet connection to load the firmware (which is much faster than the serial connection), the utility will require a valid IP address before connecting to the controller.

5. Select the controller and click OK

6. Select the revision level to which you want to update the controller and click Next.

7. To start the update of the controller, click Finish and then click Yes.

8. After the controller is updated, the status box displays *Update complete*. Click OK.

9. To close ControlFlash software, click Cancel and then click Yes.

Using AutoFlash to load firmware

You can use AutoFlash to load firmware through either an Ethernet connection (an IP address must already be assigned to the Ethernet port) or a serial connection.

1. Make sure the appropriate network connection is made before starting.

2. Use RSLogix 5000 programming software to download a controller project. If the processor firmware does not match that project revision, AutoFlash automatically launches.

3. Select the catalog number of the controller and click Next.

4. Expand the network until you see the controller. If the required network is not shown, first configure a driver for the network in RSLinx software.

   If you use an Ethernet connection to load the firmware (which is much faster than the serial connection), the utility will ask for a valid IP address before connecting to the controller.

5. Select the controller and click OK

6. Select the revision level to which you want to update the controller and click Next.

7. To start the update of the controller, click Finish and then click Yes.

8. After the controller is updated, the status box displays *Update complete*. Click OK.

9. To close AutoFlash software, click Cancel and then click Yes.

Using a CompactFlash card to load firmware

Only the 1769-L35E, 1769-L32E, and 1769-L31 controllers support CompactFlash.

If you have an existing 1769-L3xx controller that is already configured and has firmware loaded, you can store the current controller user program and firmware on CompactFlash and use that card to update other controllers.

1. Store the controller user program and firmware of a currently configured 1769-L3xx controller to the CompactFlash card.

   Make sure to select Load Image On Powerup when you save to the card.

2. Remove the card and insert it into a 1769-L3xx controller that you want to have the same firmware and controller user program.

3. When you power up the second 1769-L3xx controller, the image stored on the CompactFlash card is loaded into the controller.

Using CompactFlash

The 1784-CF64 CompactFlash card provides nonvolatile memory storage for the 1769-L3xx controller. The card stores the contents of the controller memory (program logic and tag values) and the controller firmware at the time that you store the project. Storing information to the CompactFlash card is like storing a snapshot of controller memory at a given time.

ATTENTION

If you configured the CompactFlash card to “restore on power up” and you make changes to a project, such as online edits or changes to tag values, you must store the project to the CompactFlash card again after you make changes. Otherwise, your changes are not saved and you will lose those changes on the next power cycle to the controller.

Tag values stored in flash are a snapshot at the time of the store. During a program restore the processor tag values will be equal to tag data stored on flash.

The locking tab on the front of the controller helps hold the CompactFlash card in its socket.

ATTENTION

Do not remove the CompactFlash card while the controller is reading from or writing to the card, as indicated by a flashing green CF LED. This could corrupt the data on the card or in the controller, as well as corrupt the latest firmware in the controller.

The CompactFlash card supports removal and insertion under power.

WARNING

When you insert or remove the card while backplane power is on, an electrical arc can occur. This could cause an explosion in hazardous location installations.

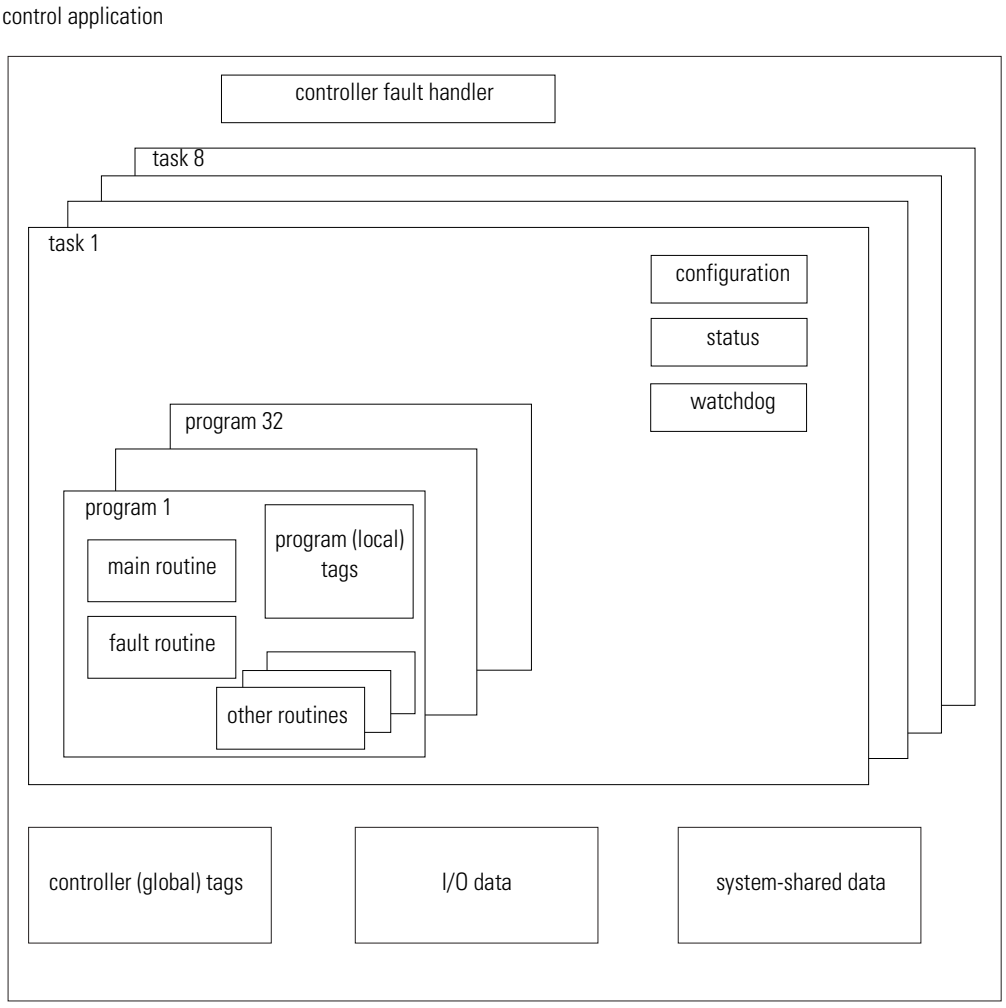
Be sure that power is removed or the area is nonhazardous before proceeding. Repeated electrical arcing causes excessive wear to contacts on both the module and its mating connector. Worn contacts may create electrical resistance that can affect module operation.

See the *Logix5000 Controllers Common Procedures Programming Manual*, publication 1756-PM001 for steps on storing an image on the CompactFlash card.

Developing Programs

The controller operating system is a preemptive multitasking system that is IEC 1131-3 compliant. This environment provides:

- tasks to configure controller execution
- programs to group data and logic
- routines to encapsulate executable code written in a single programming language



Defining tasks

A task provides scheduling and priority information for a set of one or more programs. You can configure tasks as continuous, periodic, or event. Only one task can be continuous.

This controller:	Supports this many tasks:
1769-L35E	8
1769-L32E	6
1769-L31 1769-L30 1769-L20	4

A task can have as many as 32 separate programs, each with its own executable routines and program-scoped tags. Once a task is triggered (activated), all the programs assigned to the task execute in the order in which they are grouped. Programs can only appear once in the Controller Organizer and cannot be shared by multiple tasks.

Specifying task priorities

Each task in the controller has a priority level. The operating system uses the priority level to determine which task to execute when multiple tasks are triggered. You can configure periodic tasks to execute from the lowest priority of 15 up to the highest priority of 1. A higher priority task will interrupt any lower priority task. The continuous task has the lowest priority and is always interrupted by a periodic task.

The CompactLogix controller uses a dedicated periodic task at priority 7 to process I/O data. This periodic task executes at the RPI you configure for the CompactBus, which can be as fast as once every 1 ms. Its total execution time is as long as it takes to scan the configured I/O modules.

How you configure your tasks affects how the controller receives I/O data. Tasks at priorities 1 to 6 take precedence over the dedicated I/O task. Tasks in this priority range can impact I/O processing time. If you configure the I/O RPI at 1ms and you configure a task of priority 1 to 6 that requires 500 μs to execute and is scheduled to run every millisecond. This leaves the dedicated I/O task 500 μs to complete its job of scanning the configured I/O.

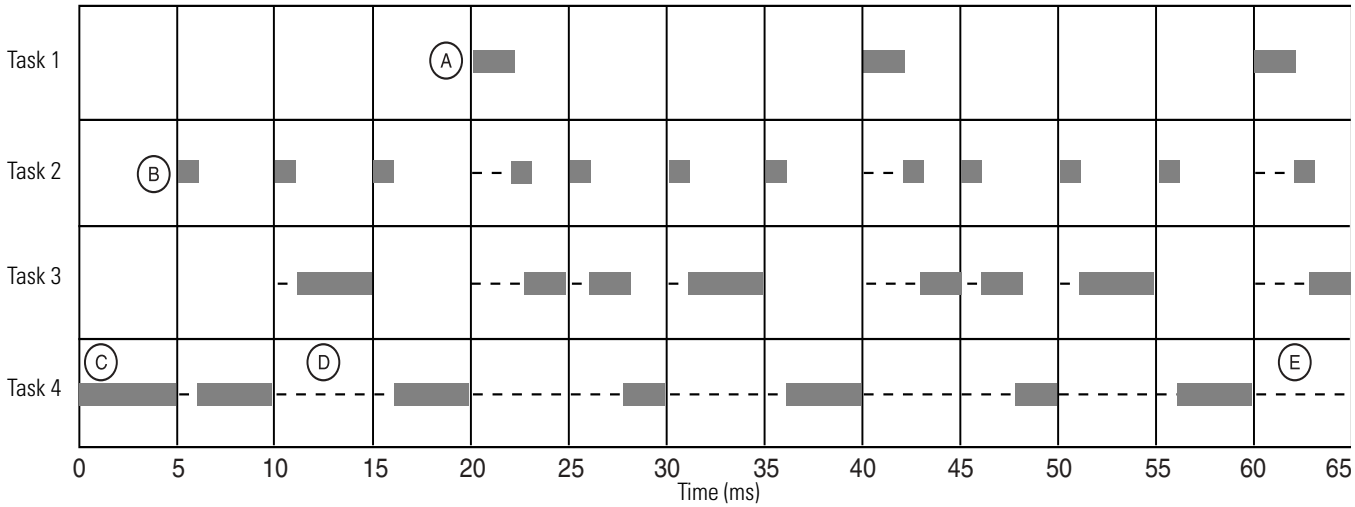
However, if you schedule two high priority tasks (1 to 6) to run every millisecond, and they both require 500 µs or more to execute, no CPU time would be left for the dedicated I/O task. Furthermore, if you have so much configured I/O that the execution time of the dedicated I/O task approaches 2 ms (or the combination of the high priority tasks and the dedicated I/O task approaches 2 ms) no CPU time is left for low priority tasks (8 to 15).

TIP

For example, if your program needs to react to inputs and control outputs at a deterministic rate, configure a periodic task with a priority higher than 7 (1 through 6). This keeps the dedicated I/O task from affecting the periodic rate of your program. However, if your program contains a lot of math and data manipulation, place this logic in a task with priority lower than 7 (8 through 15), such as the continuous task, so that the dedicated I/O task is not adversely affected by your program.

The following example shows the task execution order for an application with periodic tasks and a continuous task.

Task:	Priority Level:	Task Type:	Example Execution Time:	Worst Case Completion Time:
1	5	20 ms periodic task	2 ms	2 ms
2	7	dedicated I/O task 5 ms selected RPI	1 ms	3 ms
3	10	10 ms periodic task	4 ms	8 ms
4	none (lowest)	continuous task	25 ms	60 ms



Notes:

A. The highest priority task interrupts all lower priority tasks.

B. The dedicated I/O task can be interrupted by tasks with priority levels 1 to 6. The dedicated I/O task interrupts tasks with priority levels 8 to 15. This task runs at the selected RPI rate scheduled for the CompactLogix system (2ms in this example).

C. The continuous task runs at the lowest priority and is interrupted by all other tasks.

D. A lower priority task can be interrupted multiple times by a higher priority task.

E. When the continuous task completes a full scan it restarts immediately, unless a higher priority task is running.

Defining programs

Each program contains program tags, a main executable routine, other routines, and an optional fault routine. Each task can schedule as many as 32 programs.

The scheduled programs within a task execute to completion from first to last. Programs that are not attached to any task show up as unscheduled programs. You must specify (schedule) a program within a task before the controller can scan the program.

Defining routines

A routine is a set of logic instructions in a single programming language, such as ladder logic. Routines provide the executable code for the project in a controller. A routine is similar to a program file or subroutine in a PLC or SLC controller.

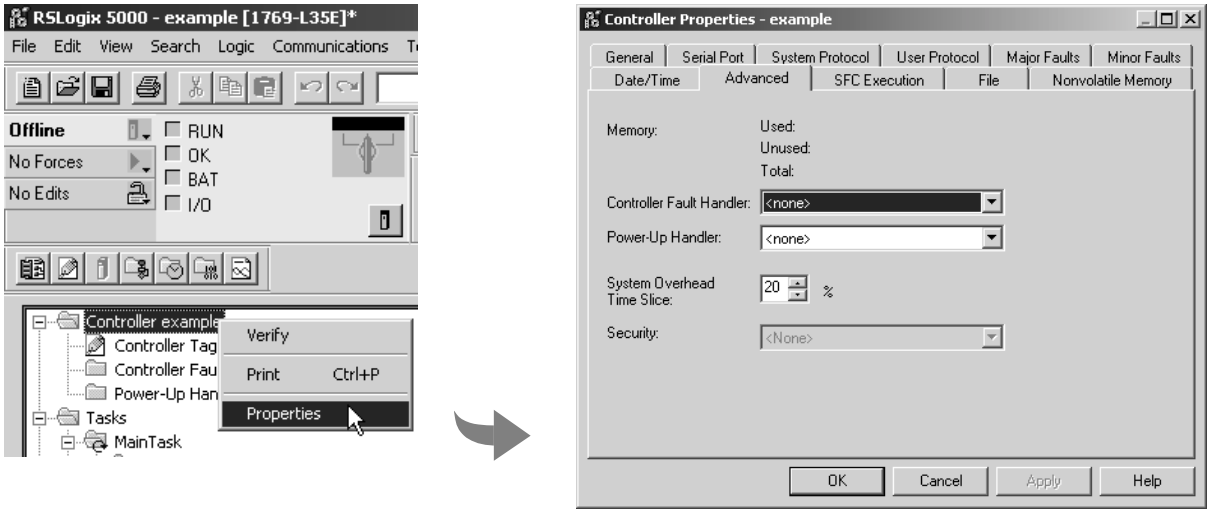
Each program has a main routine. This is the first routine to execute when the controller triggers the associated task and calls the associated program. Use logic, such as the Jump to Subroutine (JSR) instruction, to call other routines.

You can also specify an optional program fault routine. The controller executes this routine if it encounters an instruction-execution fault within any of the routines in the associated program.

Selecting a System Overhead Percentage

The Controller Properties dialog lets you specify a percentage for system overhead. This percentage specifies the percentage of controller time (excluding the time for periodic tasks) that is devoted to communication and background functions.

1. View properties for the controller and select the Advanced tab.



System overhead functions include:

- communicating with programming and HMI devices (such as RSLogix 5000 software)
- responding to messages
- sending messages

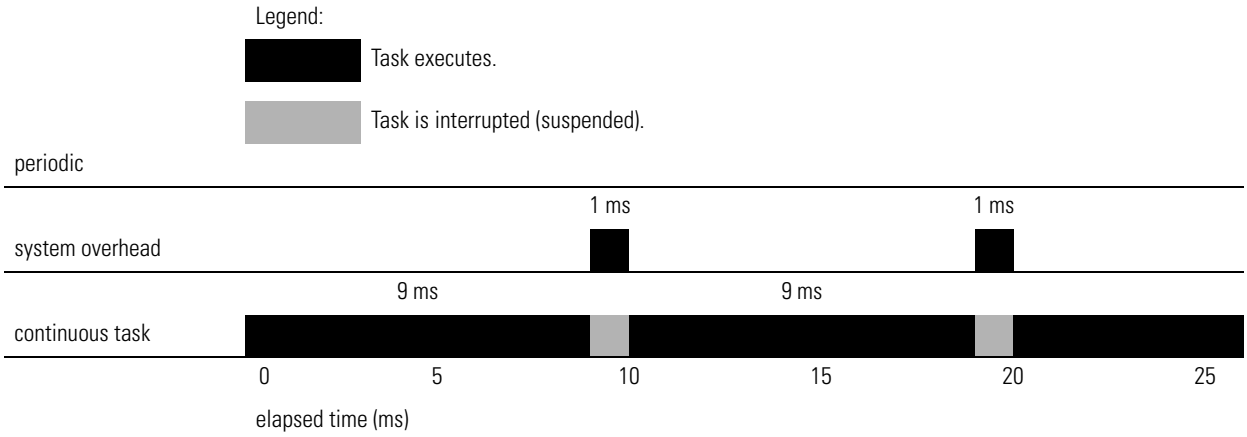
The controller performs system overhead functions for up to 1 ms at a time. If the controller completes the overhead functions in less than 1 ms, it resumes the continuous task.

As the system overhead percentage increases, time allocated to executing the continuous task decreases. If there are no communications for the controller to manage, the controller uses the communications time to execute the continuous task. While increasing the system overhead percentage decreases execution time for the continuous task, it does increase communications performance. However, increasing the system overhead percentage also increases the amount of time it takes to execute a continuous task - increasing overall scan time

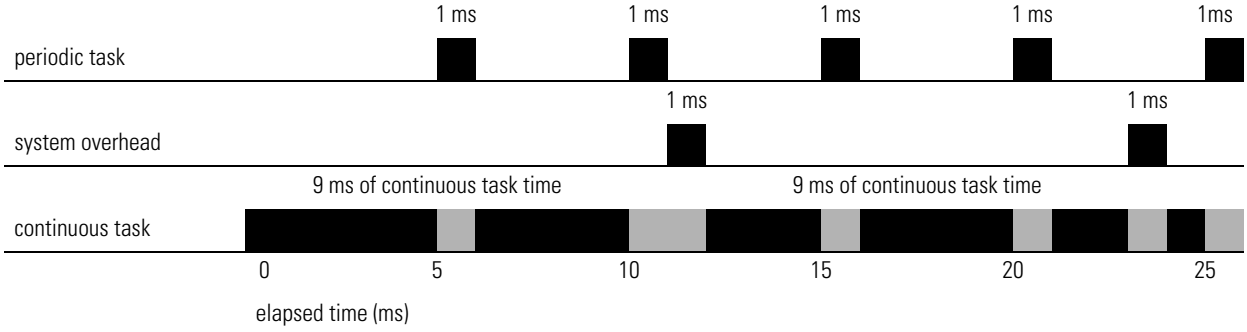
The following table shows the ratio between the continuous task and the system overhead functions:

At this time slice:	The continuous tasks runs for:	And then overhead occurs for up to:
10%	9 ms	1 ms
20%	4 ms	1 ms
33%	2 ms	1 ms
50%	1 ms	1 ms

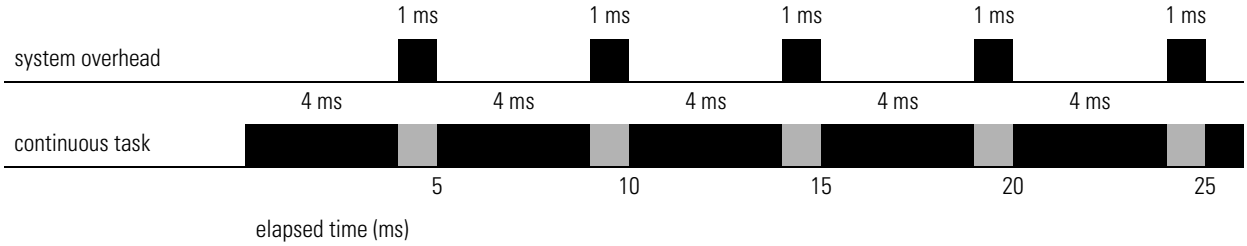
At the default time slice of 10%, system overhead interrupts the continuous task every 9 ms (of continuous task time), as illustrated below.



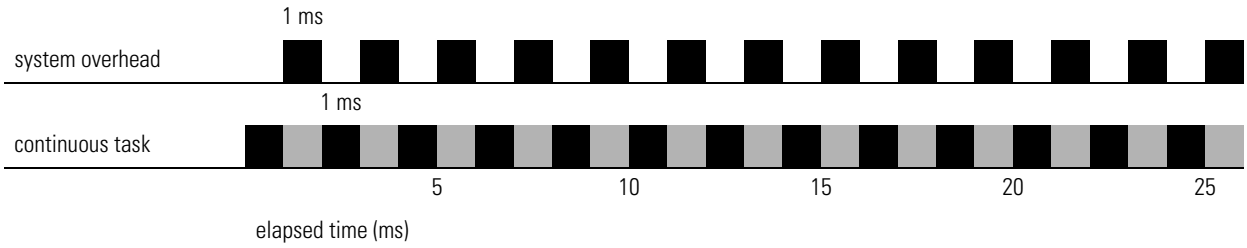
The interruption of a periodic task increases the elapsed time (clock time) between the execution of system overhead, as shown below.



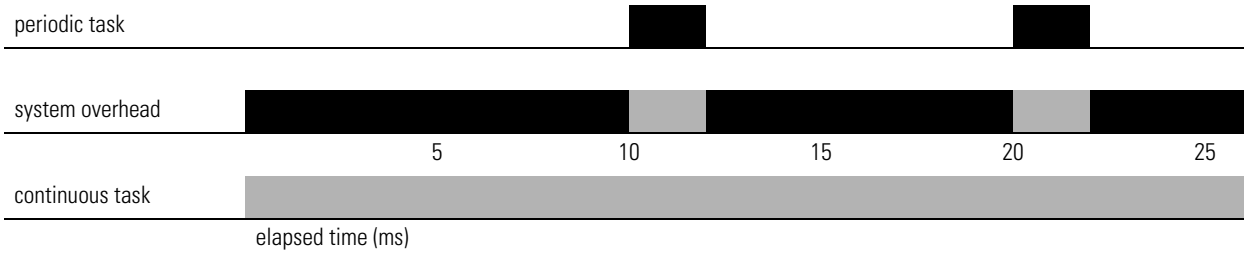
If you increase the time slice to 20%, the system overhead interrupts the continuous task every 4 ms (of continuous task time).



If you increase the time slice to 50%, the system overhead interrupts the continuous task every 1 ms (of continuous task time).



If the controller only contains a periodic task(s), the system overhead timeslice value has no effect. System overhead runs whenever a periodic task is not running.



Notes:

Placing, Configuring, and Monitoring Local I/O

Using This Chapter

Placing Local I/O Modules

The controller you use determines how many local I/O modules you can configure.

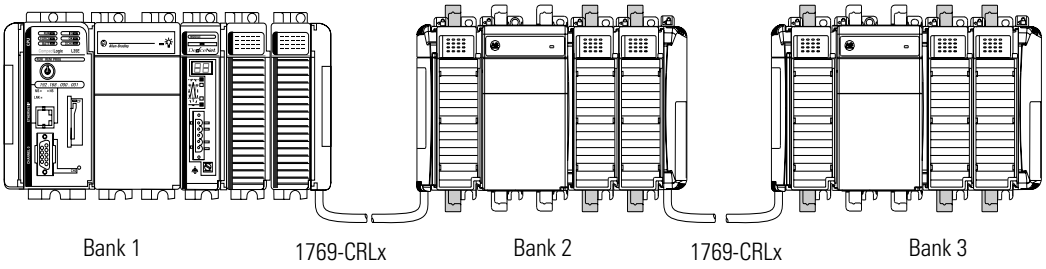
This controller:	Supports this many local I/O modules:	In this many I/O banks:
1769-L35E	30	3
1769-L32E 1769-L31 1769-L30	16	3
1769-L20	8	2

Use the 1769-CRR1/-CRR3 or 1769-CRL1/-CRL3 expansion cable to connect banks of I/O modules. You can split a bank right after the power supply or after any I/O module. Each bank must contain one power supply. An end cap/terminator must be used on the last I/O bank opposite of the expansion cable.

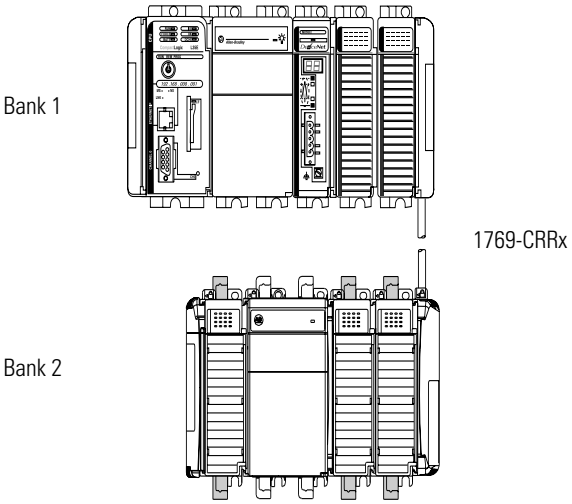
The first bank includes the CompactLogix controller in the far left position. The controller must be located within 4 positions of the bank's power supply. Only one controller can be used in a CompactLogix system.

Each I/O module also has a power supply distance rating (the number of modules from the power supply). The distance rating is printed on each module's label. Each module must be located within its distance rating.

Horizontal Orientation



Vertical Orientation



ATTENTION

The CompactLogix system does not support Removal and Insertion Under Power (RIUP). While the CompactLogix system is under power:

- any break in the connection between the power supply and the controller (i.e. removing the power supply, controller, or an I/O module) may subject the logic circuitry to transient conditions above the normal design thresholds and may result in damage to system components or unexpected behavior.
- removing an end cap or an I/O module faults the controller and may also result in damage to system components.

Validating I/O Layout

To validate your planned I/O layout, consider these requirements:

- Each module in a CompactLogix system uses a set amount of backplane memory, in addition to the data that the module stores or transfers. As you add modules, the minimum backplane RPI increases.
- The I/O modules must be distributed such that the current consumed from the left or right side of the power supply never exceeds 2.0A at 5V dc and 1.0A at 24V dc.

Estimating RPI

As you install modules, the minimum backplane RPI increases. The RPI (request packet interval) defines the frequency at which the controller sends and receives all I/O data on the backplane. There is one RPI for the entire 1769 backplane. Consider these guidelines when installing modules:

Type of Module:	Considerations:
digital and analog (any mix)	• 1-4 modules can be scanned in 1.0 ms • 5-16 modules can be scanned in 1.5 ms • 17-30 modules can be scanned in 2.0 ms • some input modules have a fixed 8.0 ms filter, so selecting a faster RPI has no affect
specialty	• "full-sized" 1769-SDN modules add 1.5 ms per module • 1769-HSC modules add 0.5 ms per module

You can always select an RPI that is slower than listed above. These considerations show how fast modules can be scanned - not how fast an application can use the data. The RPI is asynchronous to the program scan. Other factors, such as program execution duration, affect I/O throughput.

System power budget calculation

To validate your system, the total 5V dc current and 24V dc current consumed must be considered. The I/O modules must be distributed such that the current consumed from the left or right side of the power supply never exceeds 2.0A at 5V dc and 1.0A at 24V dc.

Catalog Number	Number of Modules	Module Current Requirements		Calculated Current = (Number of Modules) x (Module Current Requirements)	
		at 5V dc (in mA)	at 24V dc (in mA)	at 5V dc (in mA)	at 24V dc (in mA)
1769-HSC		425	0		
1769-IA8I		90	0		
1769-IA16		115	0		
1769-IM12		100	0		
1769-IF4		120	60		
1769-IF8		120	70		
1769-IF4XOF2		120	160		
1769-IQ16		115	0		
1769-IQ16F		110	0		
1769-IQ32		170	0		
1769-IQ6XOW4		105	50		
1769-IR6		100	45		
1769-IT6		100	40		
1769-OA8		145	0		
1769-OA16		225	0		
1769-OB8		145	0		
1769-OB16		200	0		
1769-OB16P		160	0		
1769-OB32		300	0		
1769-OF2		120	120		
1769-OF8C		145	160		
1769-OF8V		145	160		
1769-OV16		200	0		
1769-OW8		125	100		
1769-OW8I		125	100		
1769-OW16		205	180		
1769-L31		330	40		
1769-L32E		660	90		
1769-L35E		660	90		
1769-ADN		500	0		
1769-SDN		440	0		
1769-ECR(1)		5	0		
1769-ECL(1)		5	0		
			Total Current Required(2):		

(1) One 1769-ECR or 1769-ECL end cap/terminator is required in the system. The end cap/terminator used is dependent on your configuration.

(2) This number must not exceed the Power Supply Current Capacity listed below.

Power supply current capacity

Specification	1769-PA2	1769-PB2	1769-PA4	1769-PB4
Output Bus Current Capacity (0°C to +55°C)	2A at 5V dc and 0.8A at 24V dc		4A at 5V dc and 2A at 24V dc	
24V dc User Power Capacity (0°C to +55°C)	250 mA (maximum)	not applicable		

Determining When the Controller Updates I/O

The controller continually scans the control logic. One scan is the time it takes the controller to execute the logic once. Input data transfers to the controller and output data transfers to output modules are asynchronous to the logic scan.

TIP

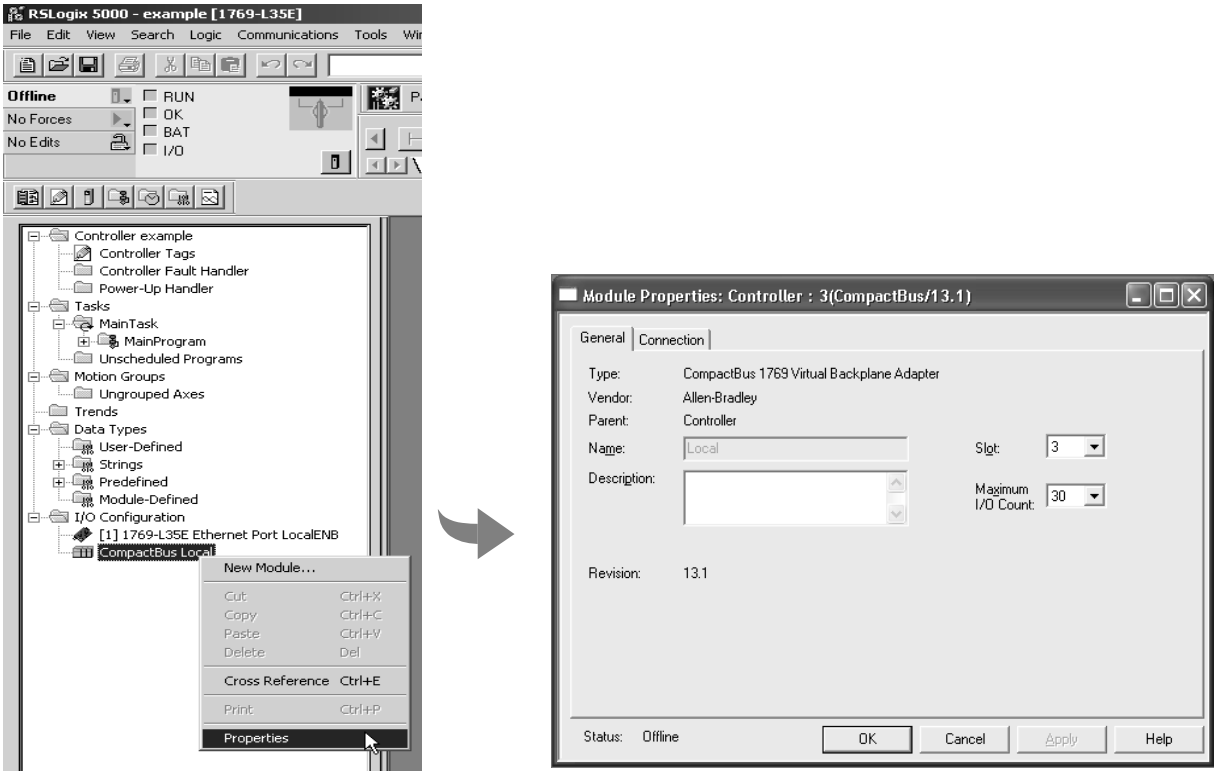
If you need to ensure that the I/O values being used during logic execution are from one moment in time (such as at the beginning of a ladder program), use the Synchronous Copy instruction (CPS) to buffer I/O data.

Refer to the *Logix5000 Controllers Common Procedures Programming Manual*, publication number 1756-PM001 for examples of I/O buffering or to the *Logix5000 Controllers General Instruction Set Reference Manual,* publication number 1756-RM003 for information on the CPS instruction.

Configuring the CompactBus

When you create a CompactLogix project, the programming software automatically creates the local CompactBus. You must configure the CompactBus.

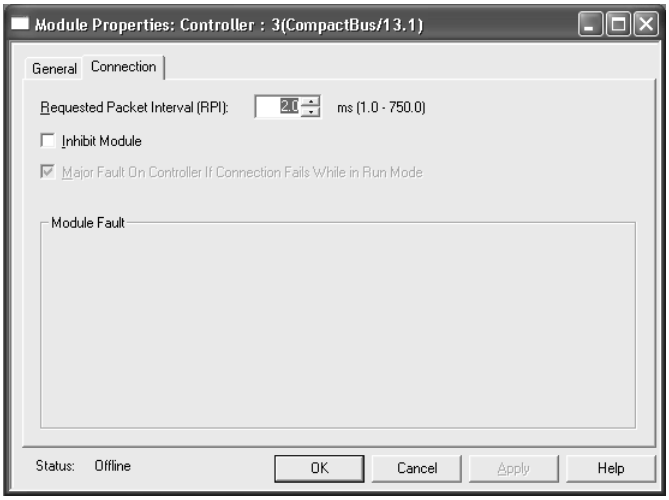
1. In the Controller Organizer, select either the CompactBus. Right-click and select Properties.



On the General tab, specify the size of the chassis. Enter the number of modules you plan to install. Include the CompactLogix controller in this total, along with a maximum of 30 I/O modules, not including the power supply.

The Comm Format for the CompactBus is automatically set to Rack Optimized and cannot be changed.

Using the Connection tab, you can specify the RPI for the systems and choose to inhibit or uninhibit the CompactBus.



The RPI you specify here is the RPI for every 1769 module on this controller's local CompactBus. Specify an RPI from 1-750ms for the system. You do not specify individual RPI values for each module.

By inhibiting and uninhibiting the CompactBus, you can write new configuration data to the entire system at once.

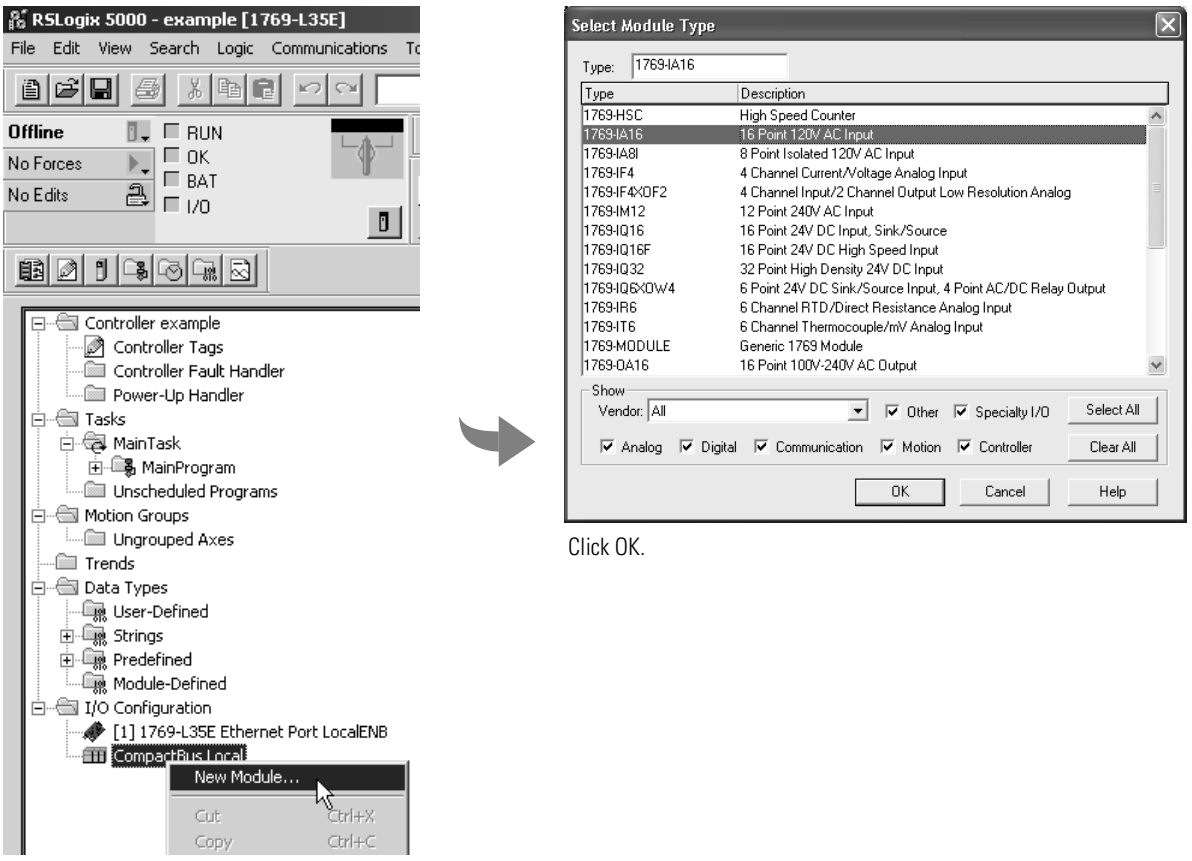
The controller's response to a CompactBus connection failure is fixed to always fault the controller. It is not configurable.

Configuring Local I/O Modules

Use your programming software to configure the I/O modules for the controller.

1. In the Controller Organizer, select the CompactBus. Right-click the selected rail and select New Module.

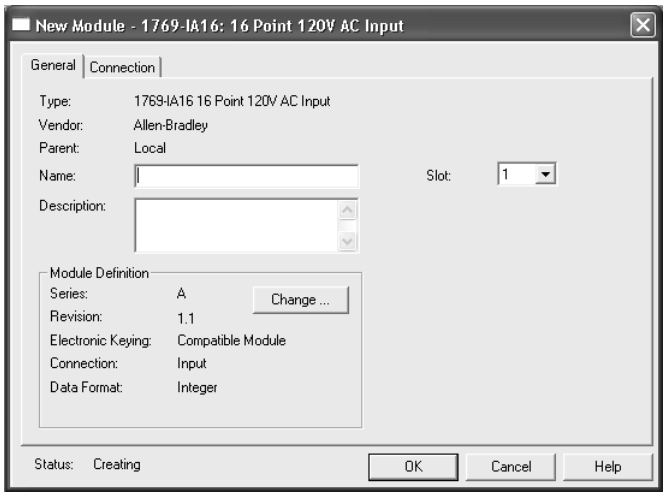
2. Select the module (1769-IA16 in this example).



Click OK.

3. Configure the module. Use the module wizard to specify characteristics for the module. Click Next to continue through the wizard.

Click Finish when you are done. The completed module appears in the Controller Organizer.



Communication formats

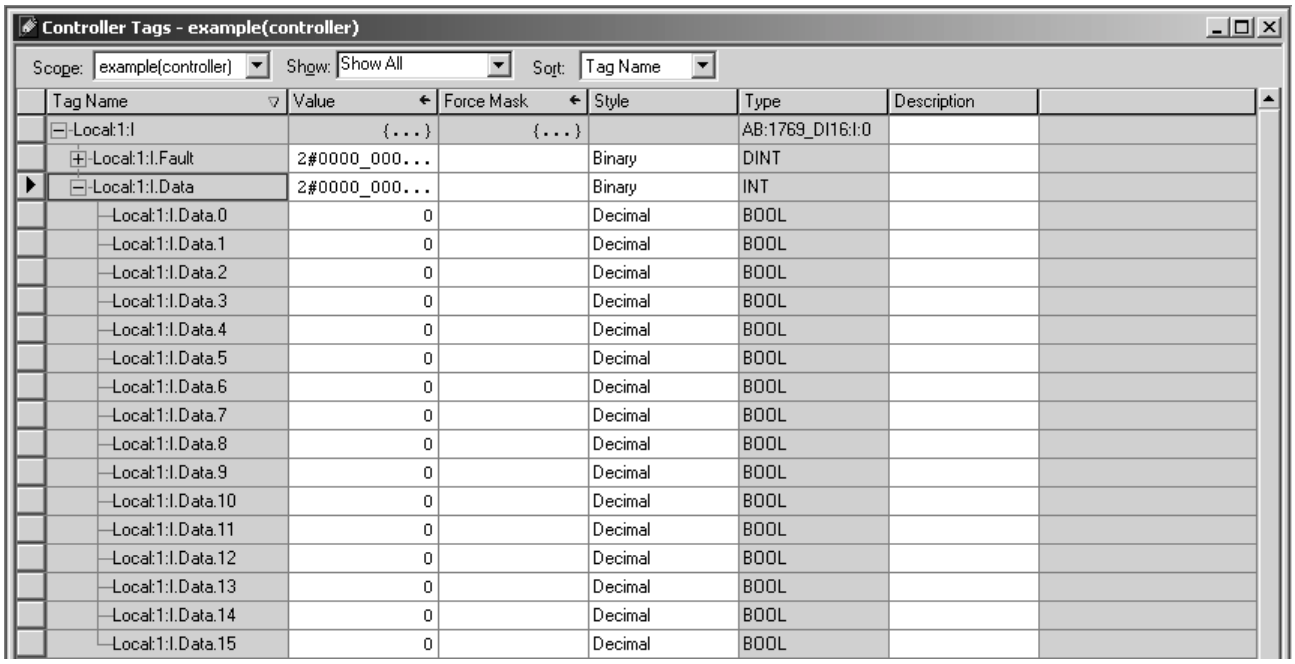
The communication format determines the data structure the I/O module uses. Each format supports a different data structure. Presently, the CompactLogix system supports two data formats:

- Input Data – INT (for 1769 input modules)
- Data – INT (for 1769 output modules)

TIP

The CompactLogix controller must own its local I/O modules. No other Logix-based controller can own the local CompactLogix I/O.

The communication format determines the tag structure that is created for the module. Assume that a 1769-IA16 Input module is in slot 1. The software creates the appropriate tags using the slot number to differentiate the tags for this example module from any other module.



Tag Name	Value	Force Mask	Style	Type	Description
Local:1:I	{...}	{...}		AB:1769_DI16:I:0	
Local:1:I.Fault	2#0000_000...		Binary	DINT	
Local:1:I.Data	2#0000_000...		Binary	INT	
Local:1:I.Data.0	0		Decimal	BOOL	
Local:1:I.Data.1	0		Decimal	BOOL	
Local:1:I.Data.2	0		Decimal	BOOL	
Local:1:I.Data.3	0		Decimal	BOOL	
Local:1:I.Data.4	0		Decimal	BOOL	
Local:1:I.Data.5	0		Decimal	BOOL	
Local:1:I.Data.6	0		Decimal	BOOL	
Local:1:I.Data.7	0		Decimal	BOOL	
Local:1:I.Data.8	0		Decimal	BOOL	
Local:1:I.Data.9	0		Decimal	BOOL	
Local:1:I.Data.10	0		Decimal	BOOL	
Local:1:I.Data.11	0		Decimal	BOOL	
Local:1:I.Data.12	0		Decimal	BOOL	
Local:1:I.Data.13	0		Decimal	BOOL	
Local:1:I.Data.14	0		Decimal	BOOL	
Local:1:I.Data.15	0		Decimal	BOOL	

Hold Last State and User-Defined Safe State not supported

When 1769 Compact I/O modules are used as local I/O modules in a CompactLogix system, the local I/O modules do not support the Hold Last State or User-Defined Safe State features, even though you can configure these options in the programming software.

- If a local I/O module fails such that its communication to the controller is lost, or if any module is disconnected from the system bus while under power, the controller will go into the fault mode. All outputs turn off when the system bus or any module faults.
- RSLogix 5000 software creates tags for modules when you add them to the I/O configuration. The 1769 module tags define configuration (C) data type members which may include attributes for alternate outputs. CompactLogix does not enable local modules to use the alternate outputs. Do not configure the attributes listed below:

For digital output modules:	For analog output modules:
• ProgToFaultEn • ProgMode • ProgValue • FaultMode • FaultValue	• CH*x*ProgToFaultEn • CH*x*ProgMode • CH*x*FaultMode • where CHx = the channel number

Any 1769 Compact I/O modules used as remote I/O modules in a CompactLogix system do support the Hold Last State and User-Defined Safe State features.

Inhibiting I/O module operation

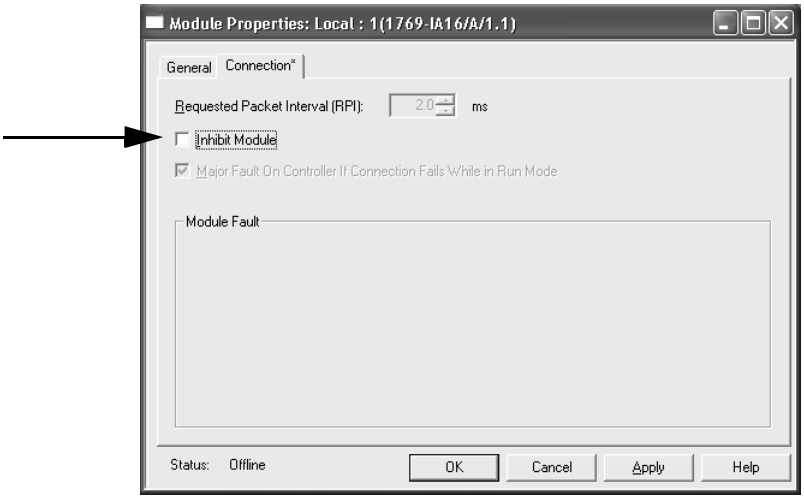
In some situations, such as when initially commissioning a system, it is useful to disable portions of a control system and enable them as you wire up the control system. The controller lets you inhibit individual modules or groups of modules, which prevents the controller from trying to communicate with these modules. Inhibiting a module shuts down the connection from the controller to that module.

When you create an I/O module, it defaults to being not inhibited. You can change an individual module's properties to inhibit a module.

ATTENTION

Inhibiting a module causes the connection to the module to be broken and prevents communication of I/O data. The controller and other I/O modules continue to operate based on old data from that module. To avoid potential injury and damage to machinery, make sure this does not create unsafe operation.

On the Connection tab of the Module Properties dialog, you can select to inhibit that specific module.



TIP

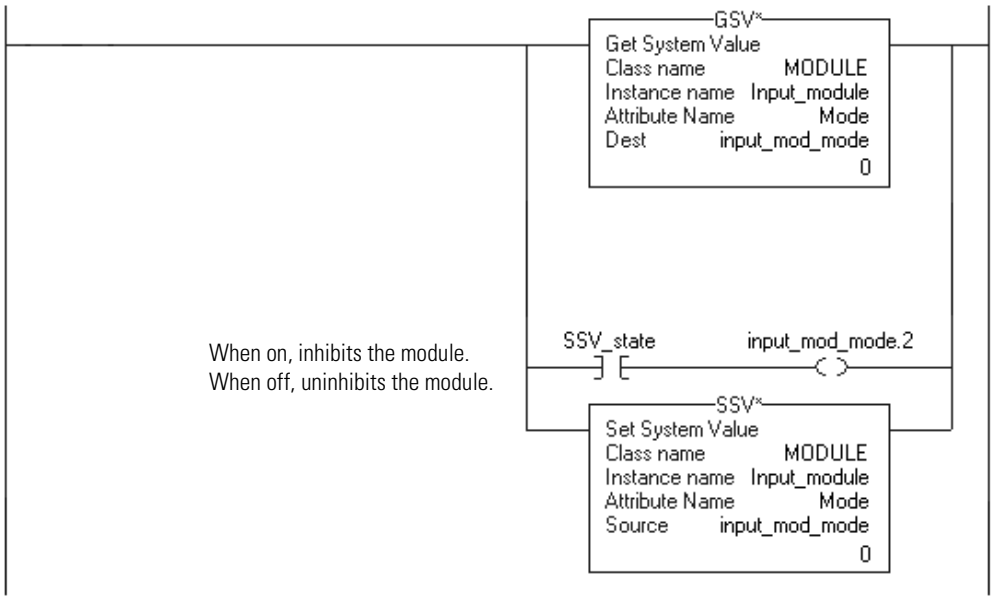
To easily inhibit all local I/O modules, you can inhibit the CompactBus, which in turn inhibits all the modules on that bus. See Configuring the CompactBus on page 2-6.

When you select to inhibit a module, the controller organizer displays a yellow circle symbol over the module.

If you are:	Inhibit a module to:
offline	put a place holder for a module you are configuring. The inhibit status is stored in the project. When you download the project, the module is still inhibited.
online	stop communication to a module. If you inhibit a module while you are connected to the module, the connection to the module is closed. The module's outputs turn off. If you inhibit a module but a connection to the module was not established (perhaps due to an error condition or fault), the module is inhibited. The module status information changes to indicate that the module is inhibited and not faulted. If you uninhibit a module (clear the check box), and no fault condition occurs, a connection is made to the module and the module is dynamically reconfigured with the configuration you created for that module. If you uninhibit the module and a fault condition occurs, a connection is not made to the module. The module status information changes to indicate the fault condition.

To inhibit a module from logic, you must first read the Mode attribute for the module using a GSV instruction. Set bit 2 to the inhibit status (1 to inhibit or 0 to uninhibit). Use a SSV instruction to write the Mode attribute back to the module. For example:

The GSV instruction gets the current status of the module named "input_module." The SSV instruction sets the state of "input_module" as either inhibited or uninhibited.



Sending module configuration information

The controller sends module configuration information once module connections are established.

ATTENTION

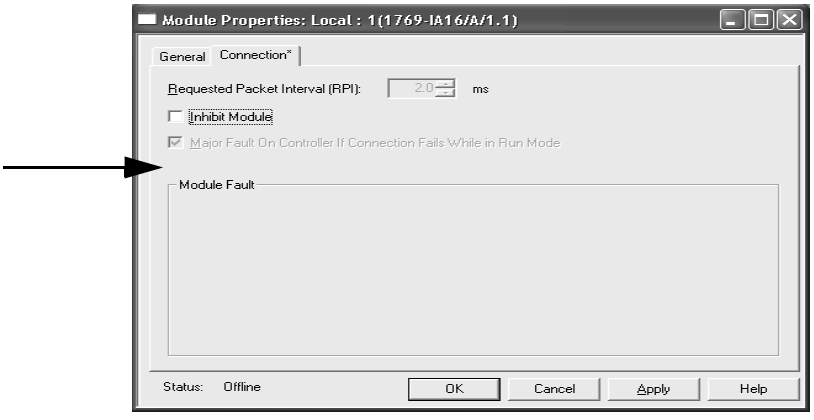
If you make a configuration change to any module in the system do one of the following to resend module configuration data:

- cycle power to the controller
- inhibit and then uninhibit the bus
- inhibit and then uninhibit the individual module
- send a MSG instruction of type Module Reconfigure (for information on configuring a MSG to send configuration data, see the *Logix5000 Controllers General Instructions Reference Manual*, publication 1756-RM003)

Configuring the controller's response to a connection failure

In a CompactLogix system, the controller's response to a CompactBus connection failure is fixed to always fault the controller. The CompactBus setting supersedes the individual module's setting.

IMPORTANT The controller's response to a connection failure of any I/O module is fixed to always fault the controller.



The I/O modules respond to a connection failure by turning off output.

Accessing I/O Data

The programming software displays I/O data as structures of multiple tags that depend on the specific features of the I/O module. The names of the data structures are based on the location of the I/O module. The programming software automatically creates the necessary structures and tags when you configure the module. Each tag name follows this format:

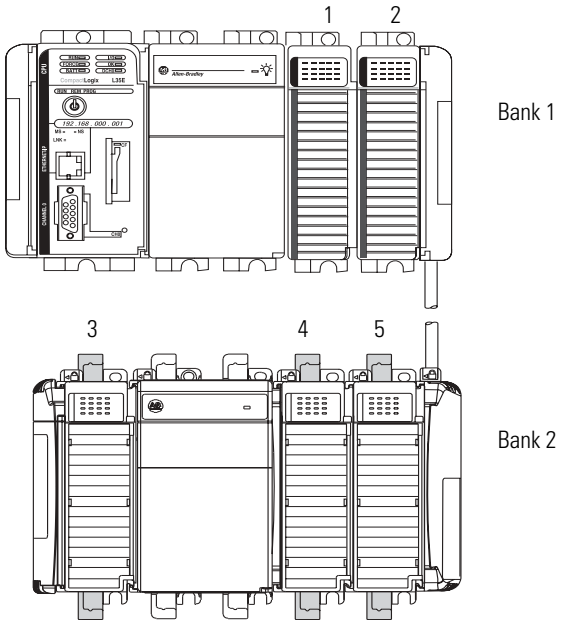
Location:SlotNumber:Type.MemberName.SubMemberName.Bit

where:

This address variable:	Is:
Location	Identifies network location LOCAL = local chassis
SlotNumber	Slot number of I/O module in its chassis
Type	Type of data I = input O = output C = configuration
MemberName	Specific data from the I/O module; depends on the type of data the module can store For example, Data and Fault are possible fields of data for an I/O module. Data is the common name for values that are sent to or received from I/O points.
SubMemberName	Specific data related to a MemberName.
Bit (optional)	Specific point on the I/O module; depends on the size of the I/O module (0-31 for a 32-point module)

The following examples show addresses for data in a CompactLogix system.

EXAMPLE I/O module on the local CompactBus utilizing two banks



Sample tag names for this example:

Location:	Example Tag Name:
input module in slot 1, LOCAL Bank 1	Local:1:C Local:1:I
output module in slot 2, LOCAL Bank 1	Local:2:C Local:2:I Local:2:O
analog input module in slot 3, LOCAL Bank 2	Local:3:C Local:3:I
analog output module in slot 4, LOCAL Bank 2	Local:4:C Local:4:I Local:4:O
analog input module in slot 5, LOCAL Bank 2	Local:5:C Local:5:I

Using aliases to simplify tag names

An alias lets you create a tag that represents another tag. This is useful for defining descriptive tag names for I/O values. For example:

Example:		Description:
I/O structure	Local:1:I:Data[0].0 Local:1:I:Fault.0	The aliases describe the specific I/O points.
alias	light_on = Local:1:I:Data[0].0 module_failed = Local:1:I:Fault.0	

Direct Connections for I/O Modules

Each local I/O module uses a direct connection to the CompactLogix controller. A direct connection is a real-time, data transfer link between the controller and an I/O module. The controller maintains and monitors the connection between the controller and the I/O module. Any break in the connection, such as a module fault, causes the controller to set fault status bits in the input data area associated with the module.

ATTENTION

The CompactLogix system does not support Removal and Insertion Under Power (RIUP). While the CompactLogix system is under power:

- any break in the connection between the power supply and the controller (i.e. removing the power supply, controller, or an I/O module) may subject the logic circuitry to transient conditions above the normal design thresholds and may result in damage to system components or unexpected behavior.
- removing an end cap or an I/O module faults the controller and may also result in damage to system components.

Monitoring I/O Modules

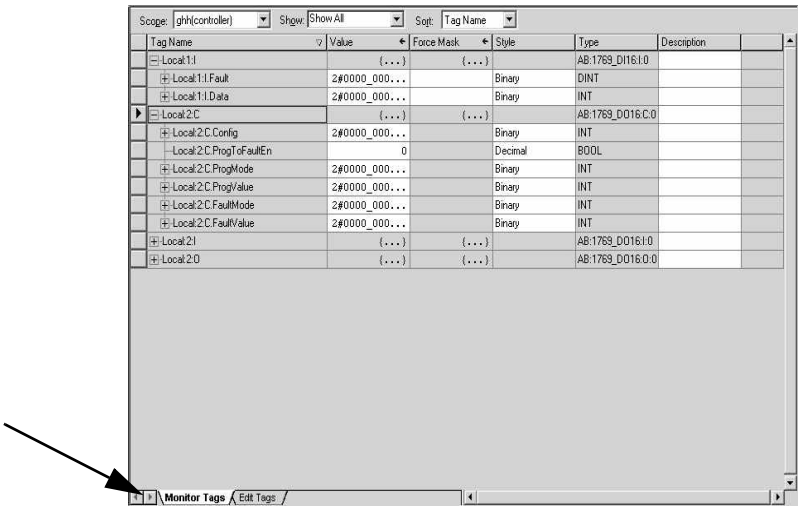
The CompactLogix controller offers different levels at which you can monitor I/O modules. You can:

- use the programming software to display fault data (See Displaying fault data on page 2-17)
- program logic to monitor fault data so you can take appropriate action (Refer to *Logix5000 Controllers Common Procedures Programming Manual*, publication number 1756-PM001, for examples.)

Displaying fault data

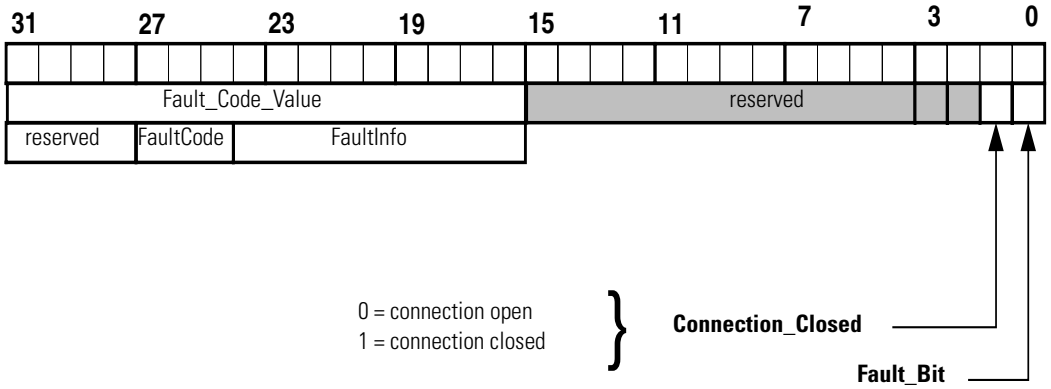
Fault data for certain types of module faults can be viewed through the programming software.

To view this data, select Controller Tags in the Controller Organizer. Right-click to select Monitor Tags.



The display for the fault data defaults to decimal. Change it to Hex to read the fault code.

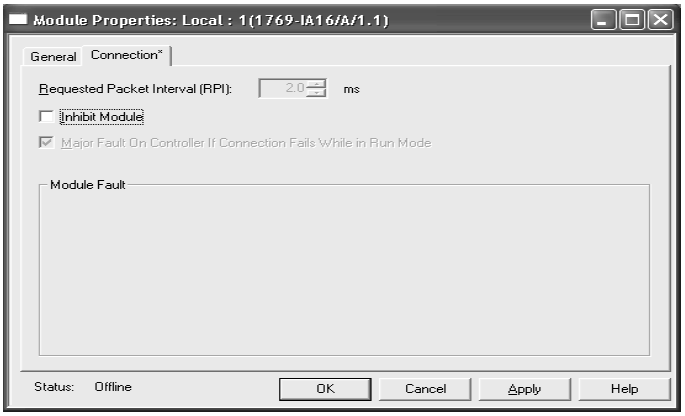
If the module faults, but the connection to the controller remains open, the controller tags database displays the fault value 16#0E01_0001. The fault word uses this format:



Where:

Bit	Description
Fault_Bit	This bit indicates that at least one bit in the fault word is set (1). If all the bits in the fault word are cleared (0), this bit is cleared (0).
Connection_Closed	This bit indicates whether the connection to the module is open (0) or closed (1). If the connection is closed (1), the Fault_Bit it set (1).

You can also view module fault data on the Connection tab of the Module Properties screen.



See your 1769 module's user documentation for a description of module faults. To recover from module faults, correct the module fault condition and send new data to the module by downloading the user program with configuration data, inhibiting and then uninhibiting the module, or cycling power.

End-cap detection and module faults

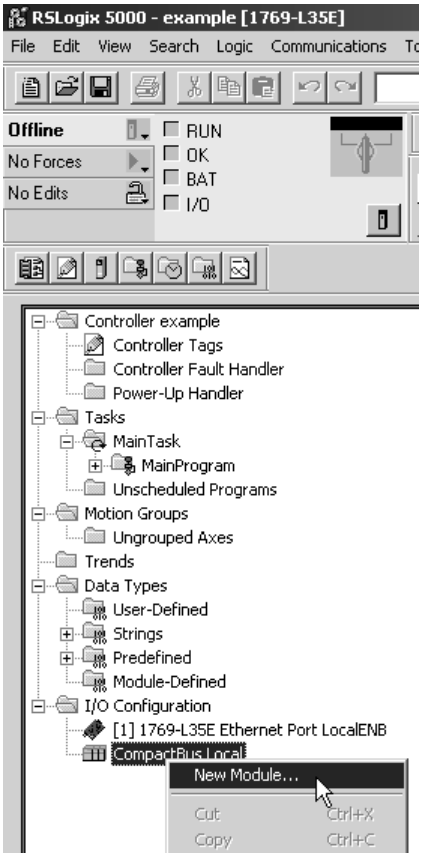
If a module that is not adjacent to an end cap experiences a fault and the connection to the controller is not broken, only the module enters the fault state.

If a module that is adjacent to an end cap experiences a fault, both the module and the controller transition to the fault state.

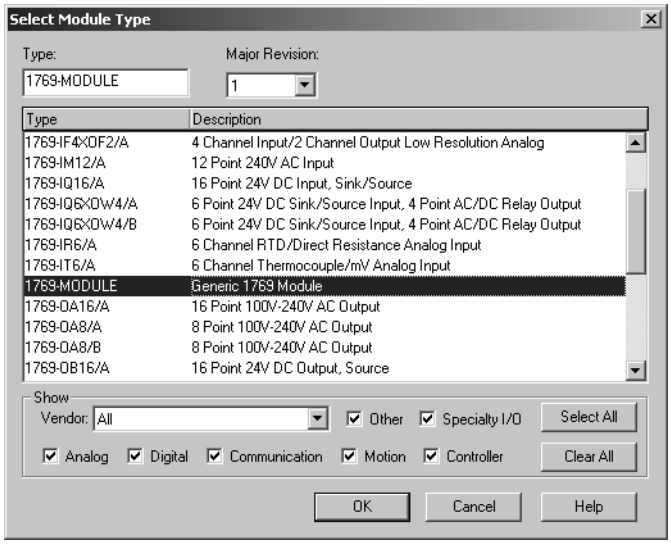
Configuring I/O Modules Using the Generic 1769-MODULE

Use the Generic 1769 Module only when a 1769 I/O module does not appear in the list of modules to add to the Controller Organizer. To configure a 1769 I/O module for a CompactLogix controller using the generic 1769-MODULE:

1. In the Controller Organizer, select the CompactBus. Right-click the selected rail and select New Module.
2. Select the 1769-MODULE (Generic 1769 Module).

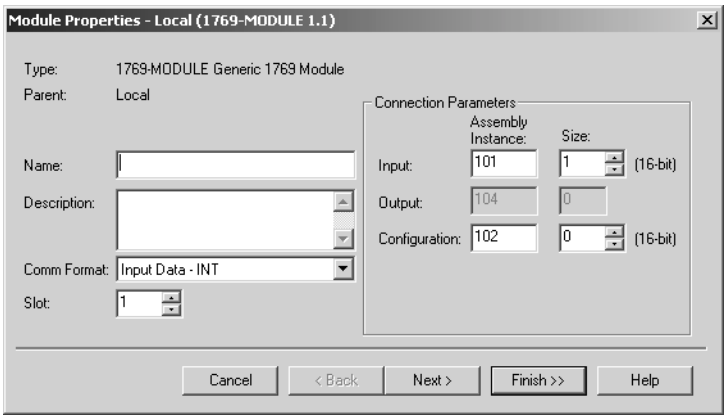


Click OK.



3. Configure the module. Use the module wizard to specify characteristics for the module. Click Next to continue through the wizard.

Click Finish when you are done. The completed module appears in the Controller Organizer.



The generic module requires you to specify more parameters of the module.

Important: The values you enter for these parameters are device specific. See the documentation for the device to determine which values to enter.

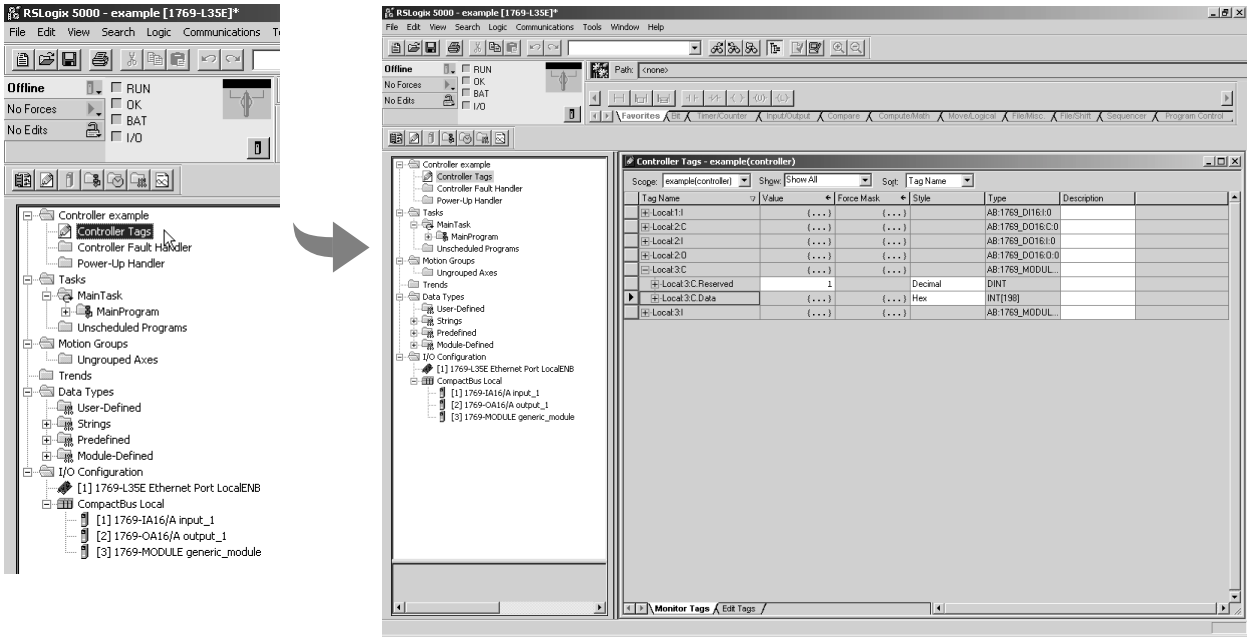
On the generic module screen, you define the parameters of the module.

In this field:	**Specify:**
Name	name of the module
Description	(optional) provide more details about the module
Comm Format	communication format 1769 analog output modules, digital output modules, analog combination modules, and digital combination modules, use Data – INT. 1769 analog input modules and digital input modules use Input Data – INT.
Slot	slot placement of the module on the CompactBus
Connection Parameters Input Output Configuration	connection information unique to the module The documentation for module should list the assembly instance and size numbers for the input, output, and configuration parameters.

Entering the configuration information for the module

Once you configure a module using the generic 1769-MODULE, you must enter the configuration information for the module into the tag database. The configuration information is downloaded to the module at program download, power up, and whenever a module is inhibited and then uninhibited.

1. In the Controller Organizer, double-click on Controller Tags.
2. Edit the tags for the module so that the tags contain the appropriate configuration information.



The generic module was added to slot 3, so you want to enter configuration data into the Local:3:C tags.

RSLogix 5000 programming software automatically create tags for configured I/O modules. All local I/O addresses are preceded by the word Local. These addresses have the following format:

- Input Data: Local:*s*:I
- Output Data: Local:*s*:O
- Configuration Data: Local:*s*:C

Where *s* is the slot number assigned the I/O module.

Open the configuration tag for that module by clicking on the plus sign to the left of its configuration tag in the tag database. The configuration information depends on the module. See the documentation on the I/O module for the appropriate configuration information.

Notes:

Communicating with Devices on an EtherNet/IP Network

Using This Chapter

The 1769-L32E and 1769-L35E controllers have a built-in EtherNet/IP port that supports program upload/download, messaging, and distributed I/O over an EthetNet/IP network.

For the CompactLogix controller to operate on an Ethernet network, you need:

- a 1769-L32E or 1769-L35E CompactLogix controller with valid firmware loaded (see the controller release notes for information on loading firmware)
- RSLinx software to configure the EtherNet/IP communication driver
- RSLogix5000 programming software

Connect the RJ-45 connector of the Ethernet cable to the Ethernet port (top port, CH1) on the controller.

ATTENTION

Do not plug a DH-485 network cable or a NAP port cable into the Ethernet port. Undesirable behavior and/or damage to the port may result.

Configuring Your System for an EtherNet/IP Network

The 1769-L32E and 1769-L35E controller ships with BOOTP enabled. You must assign an IP address to the Ethernet port in order for the controller to communicate over an EtherNet/IP network.

Step 1: Assigning network parameters

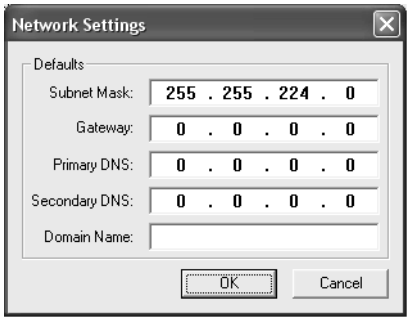
The BOOTP/DHCP utility is a stand alone program that is located in the:

- BOOTP-DHCP Server folder in the Rockwell Software program folder on the Start menu (the utility is automatically installed when you install RSLinx software)
- Tools directory on the RSLogix 5000 installation CD.

To use the BOOTP/DHCP utility:

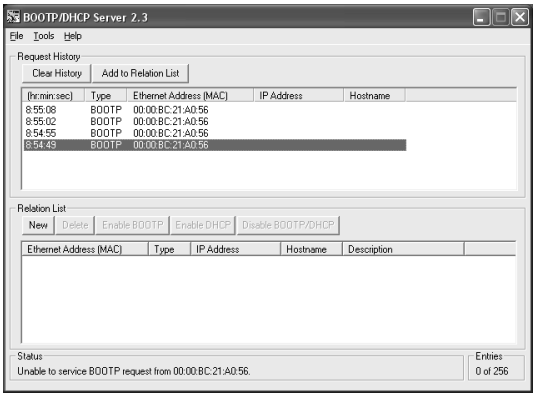
1. Start the BOOTP/DHCP software.

2. Select Tool → Network Settings. Enter the Ethernet mask and gateway. Click OK

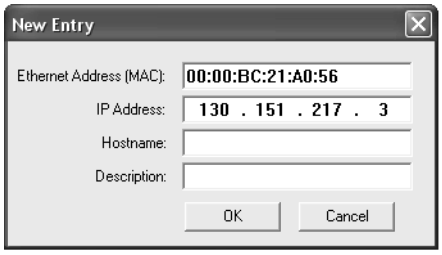


3. In the Request History panel you see the hardware addresses of devices issuing BOOTP requests. Double-click on the hardware address of the device you want to configure.

The hardware address is on the sticker located on the left-side circuit board of the controller next to the battery. The hardware address will be in this format: 00-0b-db-14-55-35.



4. The New Entry window appears with the device's Ethernet Address (MAC). Enter the Ethernet address, IP address, subnet mask, and gateway. Click OK



5. To permanently assign this configuration to the device, highlight the device and click on the Disable BOOTP/DHCP button. When power is recycled, the device uses the configuration you assigned and not issue a BOOTP request.

If you do not select the Disable BOOTP/DHCP button, on a power cycle, the controller clears the current IP configuration and will again begin sending BOOTP requests.

Other methods to assign network parameters include:

If you are working in these conditions:	Use this method for assigning network parameters:	See page:
• a BOOTP server is not available • the EtherNet/IP module is connected to another NetLinx network	RSLinx software	3-4
• the RSLogix 5000 project is online with the controller that communicates to or through the EtherNet/IP module	RSLogix 5000 software	3-5

If you use the Rockwell Automation BOOTP or DHCP server in an uplinked subnet where an enterprise DHCP server exists, a module may get an address from the enterprise server before the Rockwell Automation utility even sees the module. You might have to disconnect from the uplink to set the address and have the module remember its static address before reconnecting to the uplink. This is not a problem if you have node names configured in the module and leave DHCP enabled.

Using RSLinx software to set the IP address

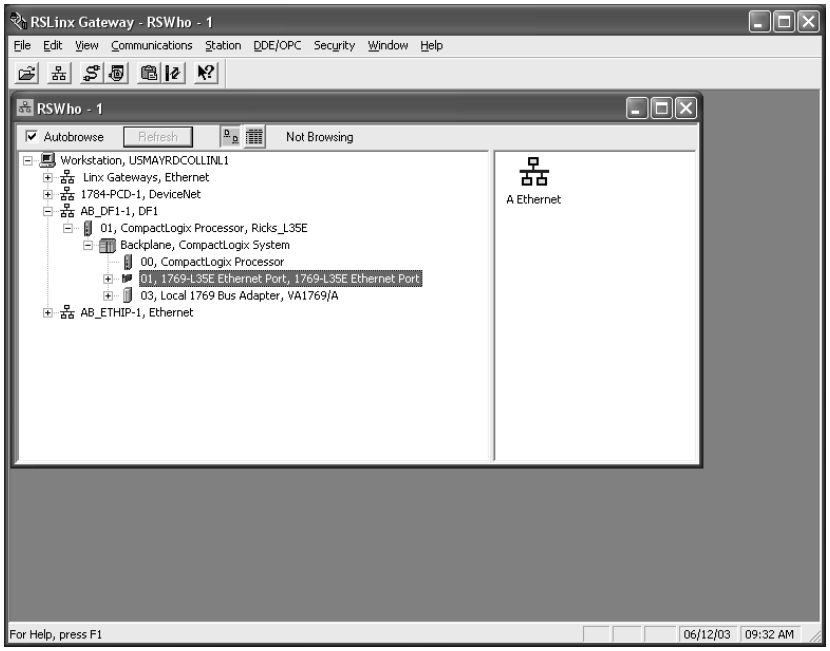
You need RSLinx software, version 2.41 or higher.

1. Make sure the controller that uses the IP address is installed and running.

2. Make a serial connection to the controller via the CH0 serial connector.
You might also need to use RSLinx software to create a DF1 driver for the workstation. See chapter 5 for more information.

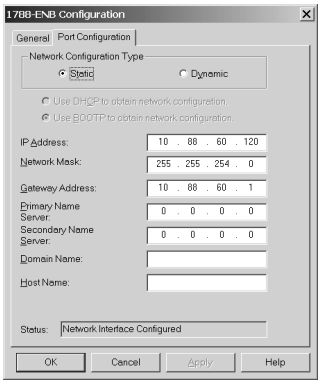
3. Start RSLinx. The RSWho window opens. Navigate in RSWho to the Ethernet network.

4. Right-click on the Ethernet port (not the controller) and select Module Configuration



5. Select the Port Configuration tab, choose Status Network Configuration type, and enter the IP address, network (subnet) mask, and gateway address (if needed).

Also, select the Static radio button to permanently assign this configuration to the port. If you select Dynamic, on a power cycle, the controller clears the current IP configuration and will again begin sending BOOTP requests.



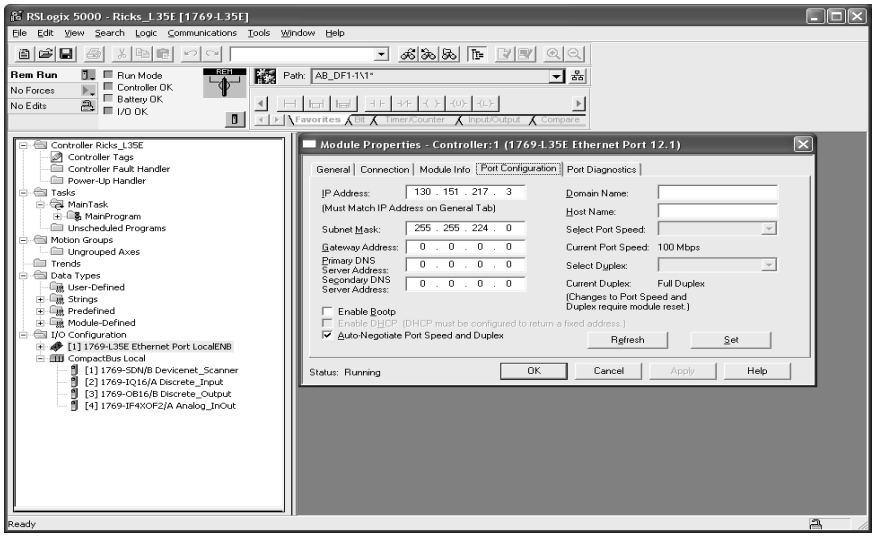
Using RSLogix 5000 software to set the IP address

IMPORTANT Before you can us RSLogix 5000 software to assign an IP address, the controller must have valid firmware loaded. See the *1769-L3xx Controller Release Notes*, publication 1769-RN006 for information on locating and loading firmware.

1. Make sure the controller that uses the IP address is installed and running.

2. Make a serial connection to the controller via the CH0 serial connector.
You might also need to use RSLinx software to create a DF1 driver for the workstation. See chapter 5 for more information.

3. Start RSLogix 5000 software. In the Controller Organizer, select properties for the Ethernet port.



4. Select the Port Configuration tab and specify the IP address and click Apply. Then click OK.

This sets the IP address in the hardware. This IP address should be the same IP address you assigned under the General tab.

From the Module Properties for the Ethernet port, you can also set a permanent port speed and duplex setting.

Step 2: Configuring the Ethernet communications driver

You need to load an Ethernet communications driver for a personal computer to communicate with other devices on an EtherNet/IP network. A personal computer only needs this driver if you use the personal computer to:

- upload and download controller projects over EtherNet/IP via RSLogix 5000 programming software
- configure EtherNet/IP network parameters for devices on the network via RSNetWorx for EtherNet/IP software

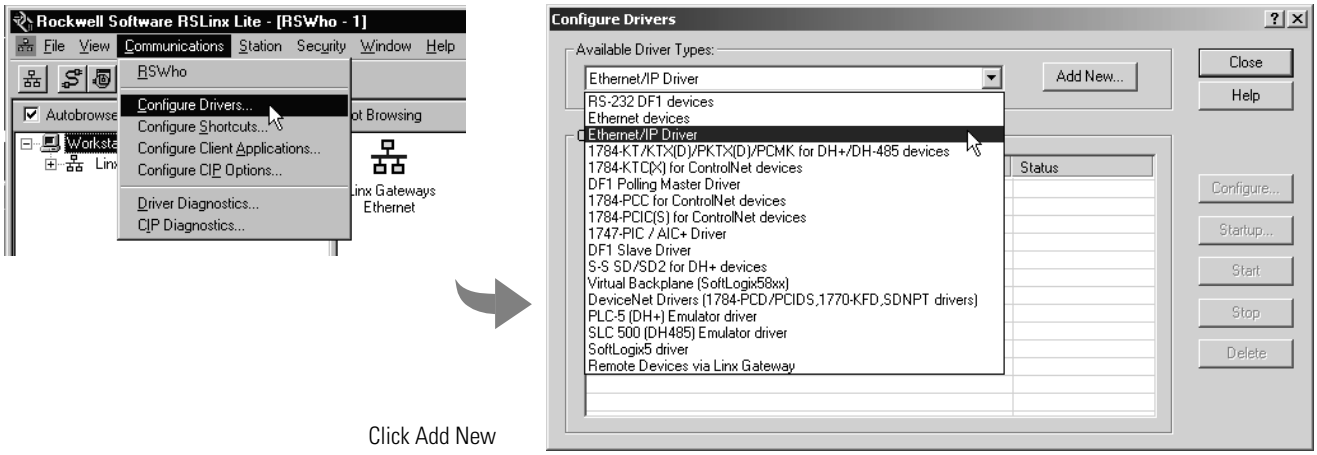
Before you load a communication driver, make sure the:

- Ethernet communication card has already installed in the personal computer
- IP address and other network parameters have been correctly configured for the personal computer
- personal computer is properly connected to the EtherNet/IP network

See the documentation for the Ethernet communications card for information on installing and configuring the card.

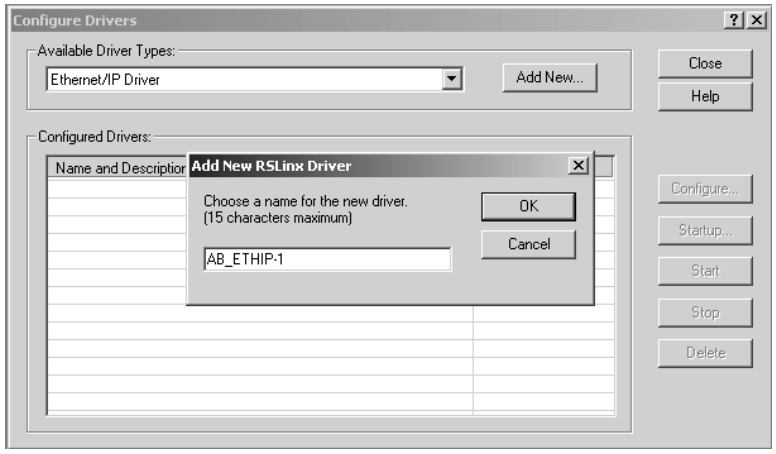
To configure the Ethernet communication driver:

1. In RSLinx software, select Configure Driver. Select “Ethernet/IP Driver” or “Ethernet Devices”.



Click Add New

2. Click Add New to add the driver.



Enter a name for the driver.

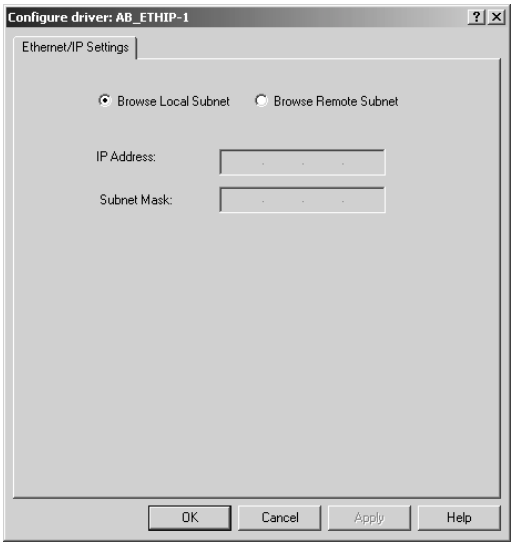
Click OK

continued

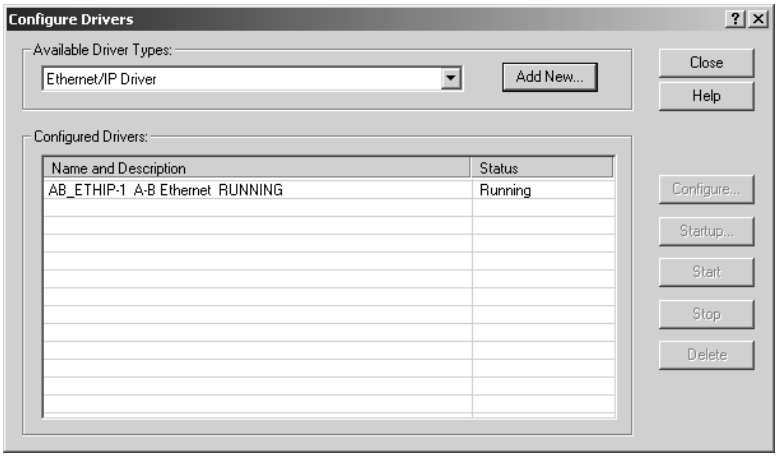
After you create the driver, configure it to correspond to the Ethernet port on the controller.

3. Select where the EtherNet/IP devices reside. The software locates valid IP addresses.

Click OK



4. The driver is now available and you can select the Ethernet port from Who Active in RSLogix 5000 programming software.



Controller Connections Over EtherNet/IP

A Logix system uses a connection to establish a communication link between two devices. Connections can be:

- controller to distributed I/O or remote communication modules
- produced and consumed tags
- messages

You indirectly determine the number of connections the controller uses by configuring the controller to communicate with other devices in the system. Connections are allocations of resources that provide more reliable communications between devices than unconnected messages.

All EtherNet/IP connections are unscheduled. An unscheduled connection is a message transfer between controllers that is triggered by the requested packet interval (RPI) or the program (such as a MSG instruction). Unscheduled messaging lets you send and receive data when needed.

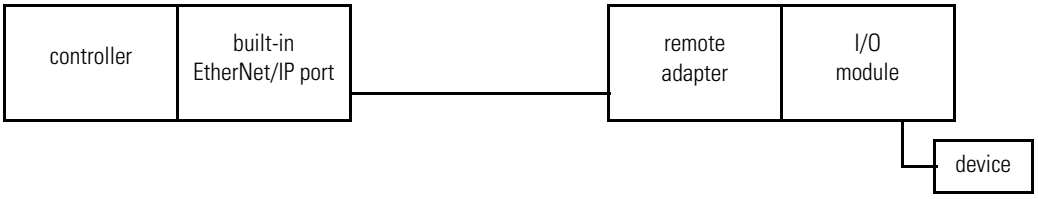
The 1769-L32E and 1769-L35E controller each supports 32 CIP connections over an EtherNet/IP network.

Configuring Distributed I/O

The CompactLogix controller supports distributed I/O over a EtherNet/IP link. Configuring I/O in a remote chassis is similar to configuring local I/O. You create the remote communication module and distributed I/O modules on the local Ethernet port.

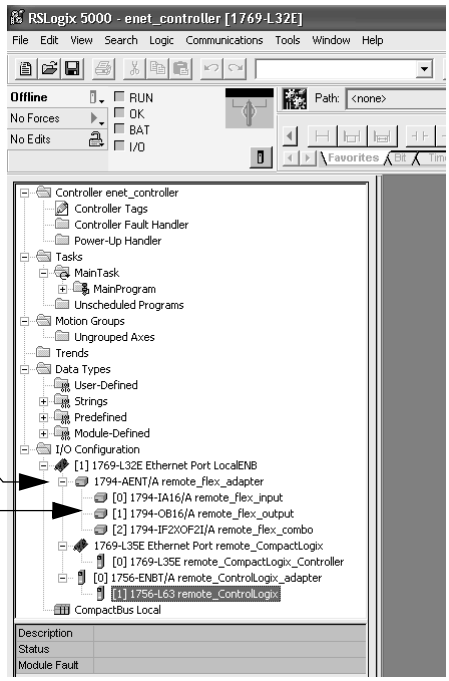
To communicate with distributed I/O modules, you add a remote adapter and I/O modules to the I/O Configuration folder of the controller.

For a typical CompactLogix distributed I/O network…



…you build the I/O configuration in this order

1. Add the remote adapter to the EtherNet/IP port of the controller.
2. Add the I/O modules to the remote adapter.



Accessing distributed I/O

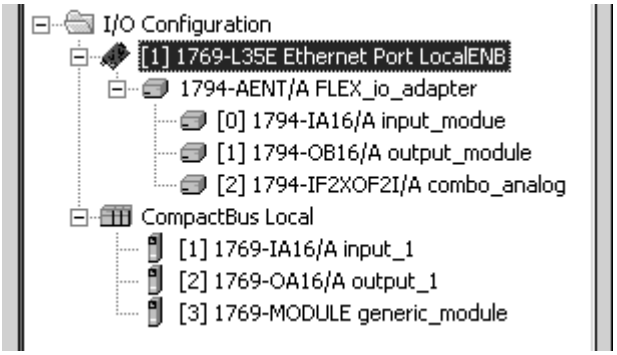
I/O information is presented as a structure of multiple fields, which depend on the specific features of the I/O module. The name of the structure is based on the location of the I/O module in the system. Each I/O tag is automatically created when you configure the I/O module through the programming software. Each tag name follows this format:

Location:SlotNumber:Type.MemberName.SubMemberName.Bit

where:

This address variable:	Is:
Location	Identifies network location LOCAL = local DIN rail or chassis ADAPTER_NAME = identifies remote adapter or bridge
SlotNumber	Slot number of I/O module in its chassis
Type	Type of data I = input O = output C = configuration S = status
MemberName	Specific data from the I/O module; depends on the type of data the module can store For example, Data and Fault are possible fields of data for an I/O module. Data is the common name for values the are sent to or received from I/O points.
SubMemberName	Specific data related to a MemberName.
Bit (optional)	Specific point on the I/O module; depends on the size of the I/O module (0-31 for a 32-point module)

EXAMPLE



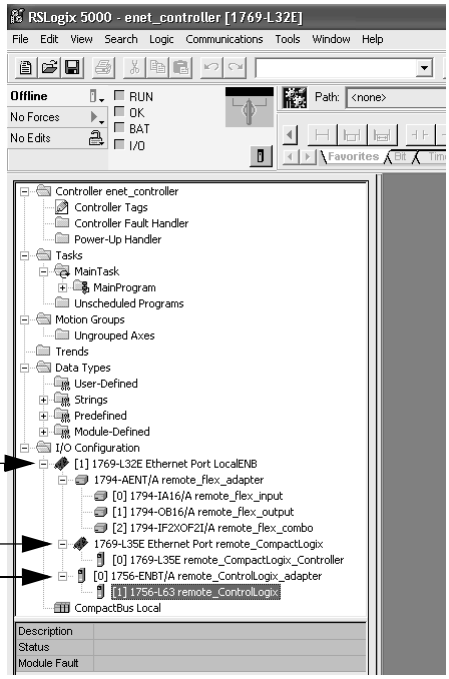
Device:	Example Tag Names (automatically created by the software):
remote adapter “FLEX_io_adapter”	FLEX_io_adapter:I FLEX_io_adapter:I.SlotStatusBits FLEX_io_adapter:I.Data FLEX_io_adapter:O FLEX_io_adapter:O.Data
remote “input_module” in slot 0 rack-optimized connection	FLEX_io_adapter:0:C FLEX_io_adapter:0:C.Config FLEX_io_adapter:0:C.DelayTime_0 FLEX_io_adapter:0:C.DelayTime_1 FLEX_io_adapter:0:C.DelayTime_2 FLEX_io_adapter:0:C.DelayTime_3 FLEX_io_adapter:0:C.DelayTime_4 FLEX_io_adapter:0:C.DelayTime_5 FLEX_io_adapter:0:I
remote “output_module” in slot 1 rack-optimized connection	FLEX_io_adapter:1:C FLEX_io_adapter:1:C.SSData FLEX_io_adapter:1:O FLEX_io_adapter:1:O.Data
remote “combo_analog” in slot 2 direct connection	FLEX_io_adapter:2:C FLEX_io_adapter:2:C.InputFIlter FLEX_io_adapter:2:C.InputConfiguration FLEX_io_adapter:2:C.OutputConfiguration FLEX_io_adapter:2:C.RTSInterval FLEX_io_adapter:2:C.SSCh0OuputData FLEX_io_adapter:2:C.SSCH1OutputData FLEX_io_adapter:2:I

Adding a Remote Controller

If you want to add the controller as a remote consumed controller to the I/O configuration, you first add the EtherNet/IP port and then the controller.

To add a remote controller, you build the I/O configuration in this order

1. You add devices to the EtherNet/IP port of the controller.
2. Add a CompactLogix controller. The software adds the EtherNet/IP port.
3. For a controller that requires a communication module, add the module first and then add the controller.



Producing and Consuming Data

The 1769-L32E and 1769-L35E controller supports the ability to produce (broadcast) and consume (receive) system-shared tags over an EtherNet/IP link. Produced and consumed data is accessible by multiple controllers over an Ethernet network. The controller sends or receives data at a predetermined RPI rate. This is the recommended method of communication between Logix controllers.

Produced and consumed tags must be controller-scoped tags of DINT or REAL data type, or in an array or structure.

Tag type:	Description:	Specify:
produced	These are tags that the controller produced for other controllers to consume.	• Enabled for producing • How many consumers allowed
consumed	These are tags whose values are produced by another controller.	• Controller name that owns the tag that the local controller wants to consume • Tag name or instance that the controller wants to consume • Data type of the tag to consume • Update interval of how often the local controller consumes the tag

The producer and consumer must be configured correctly for the specified data to be shared. A produced tag in the producer must be specified exactly the same as a consumed tag in the consumer.

If any produced/consumed tag between a producer and consumer is not specified correctly, none of the produced/consumed tags for that producer and consumer will be transferred. For example, if a CompactLogix controller is consuming three tags that another controller consumes but the first tag is specified incorrectly, none of the tags are transferred to the consuming CompactLogix controller.

However, one consumer failing to access shared data does not affect other consumers accessing the same data. For example, if the producing CompactLogix controller from the previous example also produced tags for other consuming controllers but did so correctly, those tags are still transferred to the additional consuming controllers.

Maximum number of produced and consumed tags

The maximum number of produced/consumed tags that you can configure depends on the connection limits of the Ethernet port on the controller. You can have a maximum of 32 connections through the Ethernet port.

Each produced tag uses one connection for the tag and the first configured consumer of the tag. Each consumer thereafter uses an additional connection.

If you have a lot of data to produce or consume, organize that data into an array. An array is treated as one tag, so it uses only one connection. An array can be as large as 488 bytes, which is also the limit of a produced or consumed tag.

Size limit of a produced or consumed tag

A produced or consumed tag can be as large as 488 bytes, but it must also fit within the bandwidth of the EtherNet/IP network.

Producing a tag

Produced data must be of DINT or REAL data type or a structure. You can use a user-defined structure to group BOOL, SINT, and INT data to be produced. To create a produced tag:

1. You must be programming offline.
2. In the controller organizer, double-click the Controller Tags folder and then click the Edit Tags tab.
3. Select the tag that you want to produce, or enter a new tag, and display the Tag Properties dialog box.
4. Make sure the tag is controller scope.
5. Select the “Produce this tag” check box. Specify how many controllers can consume the tag.

You can produce a base, alias, or consumed tag.

The consumed tag in a receiving controller must have the same data type as the produced tag in the originating controller. The controller performs type checking to ensure proper data is being received.

Produced tags require connections. The number of connections depends on how many controllers are consuming the tags. The controller requires one connection for the produced tag and the first consumer. Then, the controller requires an additional connection for each subsequent consumer.

Consuming a tag

A consumed tag represents data that is produced (broadcast) by one controller and received and stored by the consuming controller. To create a consumed tag:

1. You must be programming offline.
2. In the controller organizer, double-click the Controller Tags folder and then click the Edit Tags tab.
3. Select the tag that you want to consume, or enter a new tag, and display the Tag Properties dialog box.
4. Specify:

In this field:	Type or select:
Tag Type	Select Consumed.
Controller	Select the name of the other controller. You must have already created the controller in the controller organizer for the controller name to be available.
Remote Tag Name Remote Instance	Type a name for the tag in the other controller you want to consume. **Important:** The name must match the name in the remote controller exactly, or the connection faults.
RPI (requested packet interval)	Type the amount of time in msec between updates of the data from the remote controller. The local controller will receive data at least this fast. Virtual-backplane controllers, such as CompactLogix and FlexLogix controllers, only produce data at RPIs in powers of two milliseconds (such as 2, 4, 8, 16, 32, 64, etc.), or when triggered by an IOT instruction.
Display Style	If you are creating a consumed tag that refers to a tag whose data type is BOOL, SINT, INT, DINT, or REAL, you can select a display style. This display style defines how the tag value will be displayed in the data monitor and ladder editor. The display style does not have to match the display style of the tag in the remote controller.

All consumed tags are automatically controller-scope.

The produced tag in the originating CompactLogix controller must have the same data type as the consumed tag in the consuming controller. The CompactLogix controller performs type checking to make sure proper data is being received.

IMPORTANT If a consumed-tag connection fails, none of the tags are transferred from the producing controller to the consuming controller.

Sending Messages

The CompactLogix controller can send MSG instructions to other controllers and devices over an EtherNet/IP link. Each MSG instruction requires you to specify a target and an address within the target.

MSG instructions are unscheduled. The type of MSG determines whether or not it requires a connection. If the MSG instruction requires a connection, it opens the needed connection when it is executed. You can configure the MSG instruction to keep the connection open (cache) or to close it after sending the message.

This type of MSG:	Using this communication method:	Uses a connection:	Which you can cache:
CIP data table read or write	CIP	X	X
PLC-2, PLC-3, PLC-5, or SLC (all types)	CIP	X	X
	CIP with Source ID	X	X
	DH+	X	
CIP generic	CIP	X(1)	X
block-transfer read or write	na	X	X

(1) You can connect CIP generic messages, but for most applications, we recommend you leave CIP generic messages unconnected.

IMPORTANT The update time of local I/O modules may increase when the controller is bridging messages.

Bridging over the CompactLogix controller should be targeted toward applications that are not real time dependent, such as RSLogix 5000 program downloads and ControlFlash updates.

Communicating with another Logix-based controller

All Logix-based controllers can use MSG instructions to communicate with each other. The following examples show how to use tags in MSG instructions between Logix-based controllers.

Type of MSG Instruction:	Example Source and Destination:	
Logix-based controller writes to Logix-based controller (CIP Data Table Write)	source tag	*array_1*
	destination tag	*array_2*
Logix-based controller reads from Logix-based controller (CIP Data Table Read)	source tag	*array_1*
	destination tag	*array_2*

The source and destination tags:

- must be controller-scoped tags.
- can be of any data type, except for AXIS, MESSAGE, or MOTION_GROUP.

Communicating with other controllers over EtherNet/IP

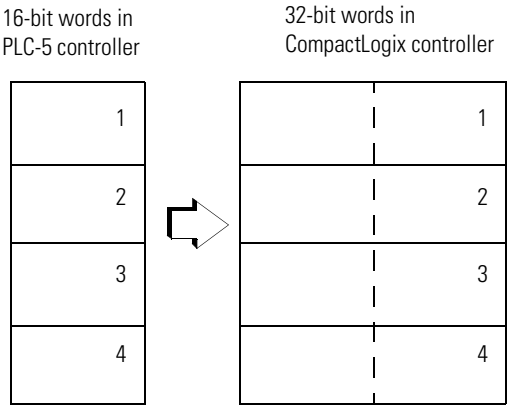
The CompactLogix controller also uses MSG instructions to communicate with PLC and SLC controllers. The MSG instructions differ depending on which controller initiates the instruction.

For MSG instructions originating from a CompactLogix controller to a PLC or SLC controller:

Type of MSG Instruction:	Supported Source File Types:	Supported Destination File Types:
CompactLogix writes to PLC-5 or SLC	In the CompactLogix controller, specify the source data type based on the destination device: PLC-5: SINT, INT, DINT, or REAL SLC: INT, REAL Example source element: *array_1*	Specify the destination file type based on the destination device: PLC-5 typed write: S, B, N, or F PLC-5 word-range write: S, B, N, F, I, O, A, or D SLC: B, N or F Example destination tag: *N7:10*
CompactLogix writes to PLC-2	In the CompactLogix controller, select one of these data types: SINT, INT, DINT, or REAL Example source element: *array_1*	Use the PLC-2 compatibility file. Example destination tag: *010*
CompactLogix reads from PLC-5 or SLC	Specify the destination file type based on the destination device: PLC-5 typed read: S, B, N, or F PLC-5 word-range read: S, B, N, F, I, O, A, or D SLC: B, N or F Example source element: *N7:10*	In the CompactLogix controller, specify the destination data type based on the destination device: PLC-5: SINT, INT, DINT, or REAL SLC: INT, REAL Example destination tag: *array_1*
CompactLogix reads from PLC-2	Use the PLC-2 compatibility file. Example source element: *010*	In the CompactLogix controller, select one of these data types: SINT, INT, DINT, or REAL Example destination tag: *array_1*

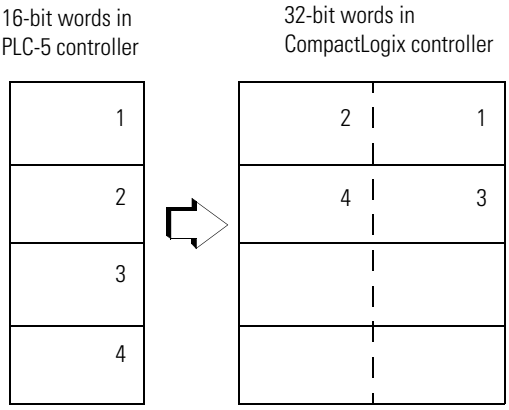
The CompactLogix controller can send typed or word-range commands to PLC-5 controllers. These commands read and write data differently. The following diagrams show how the typed and word-range commands differ.

Typed read command



The typed commands maintain data structure and value.

Word-range read command



The word-range commands fill the destination tag contiguously. Data structure and value change depending on the destination data type.

The CompactLogix controller can process messages initiated from PLC or SLC controllers. These messages use data table addresses. In order for these controllers to access tags within the CompactLogix controller, you map tags to data table addresses.

Mapping addresses

The programming software includes a PLC/SLC mapping tool which allows you to make an existing controller array tag in the local controller available to PLC-2, PLC-3, PLC-5, or SLC controllers.

To map addresses:

1. From the Logic menu, select Map PLC/SLC Messages.

2. Specify this information:

For:	In this field:	Specify:	For example:
PLC-3, PLC-5, and SLC controllers	File Number	Type the file number of the data table in the PLC/SLC controller.	*10*
	Tag Name	Type the array tag name the local controller uses to refer to the PLC/SLC data table address. The tag must be an integer array (SINT, INT, or DINT) that is large enough for the message data.	*array_1*
PLC-2 controllers	Tag Name	Type the tag name to be the PLC-2 compatibility file.	*200*

TIP

You can map as many tags as you want to a PLC-3, PLC-5, or SLC controller. You can map only one tag to a PLC-2 controller.

The following table shows example source and destination tags and elements for different controller combinations.

Type of MSG Instruction:	Example Source and Destination:	
PLC-5 writes to CompactLogix SLC writes to CompactLogix SLC 5/05 SLC 5/04 OS402 and above SLC 5/03 OS303 and above	source element	*N7:10*
	destination tag	*"array_1"*
	The PLC-5, PLC-3, and SLC controllers support logical ASCII addressing so you do not have to map a compatibility file for MSG instructions initiated by a PLC-5, PLC-3, or SLC controller. Place the CompactLogix tag name in double quotes (").	
	You could optionally map a compatibility file. For example, if you enter *10* for the compatibility file, you enter *N10:0* for the destination tag.	
PLC-2 writes to CompactLogix	source element	*010*
	destination tag	*200*
	The destination tag is the three-digit PLC-2 address you specified for PLC-2 mapping.	
PLC-5 reads from CompactLogix SLC reads from CompactLogix SLC 5/05 SLC 5/04 OS402 and above SLC 5/03 OS303 and above	source tag	*"array_1"*
	destination element	*N7:10*
	The PLC-5, PLC-3, and SLC controllers support logical ASCII addressing so you do not have to map a compatibility file for MSG instructions initiated by a PLC-5, PLC-3, or SLC controller. Place the CompactLogix tag name in double quotes (").	
	You could optionally map a compatibility file. For example, if you enter *10* for the compatibility file, you enter *N10:0* for the source tag.	
PLC-2 reads from CompactLogix	source tag	*200*
	destination element	*010*
	The source tag is the three-digit PLC-2 address you specified for PLC-2 mapping.	

When the CompactLogix controller initiates messages to PLC or SLC controllers, you do not have to map compatibility files. You enter the data table address of the target device just as you would a tag name.

SLC 5/05 controllers, SLC 5/04 controllers (OS402 and above), and SLC 5/03 controllers (OS303 and above) support logical ASCII addressing and support PLC/SLC mapping (see the examples above). For all other SLC or MicroLogix1000 controllers, you must map a PLC-2 compatibility file (see the PLC-2 examples above).

Using a MSG Instruction to Send an Email

The controller is an email client that uses a mail relay server to send email. The CompactLogix controller can execute a generic CIP message that sends an email message to a SMTP mail relay server using the standard SMTP protocol.

Some mail relay servers require a domain name be provided during the initial handshake of the SMTP session. For these mail relay servers, make sure you specify a domain name when you configure the network settings. See page 3-2 for information on configuring the network settings of the controller and specifying a domain name.

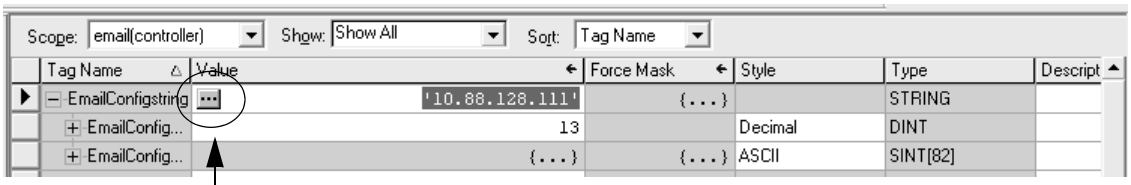
IMPORTANT Be careful to write the ladder logic to ensure the MSG instructions are not continuously triggered to send email messages.

Step 1: Create string tags

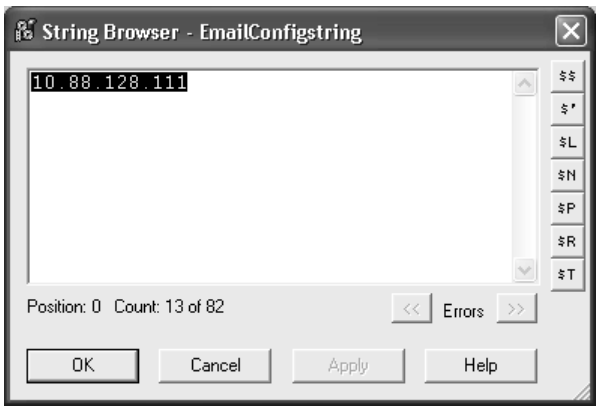
You need three string tags:

- one to identify the mail server
- one to contain the email text.
- one to contain the status of the email transmission

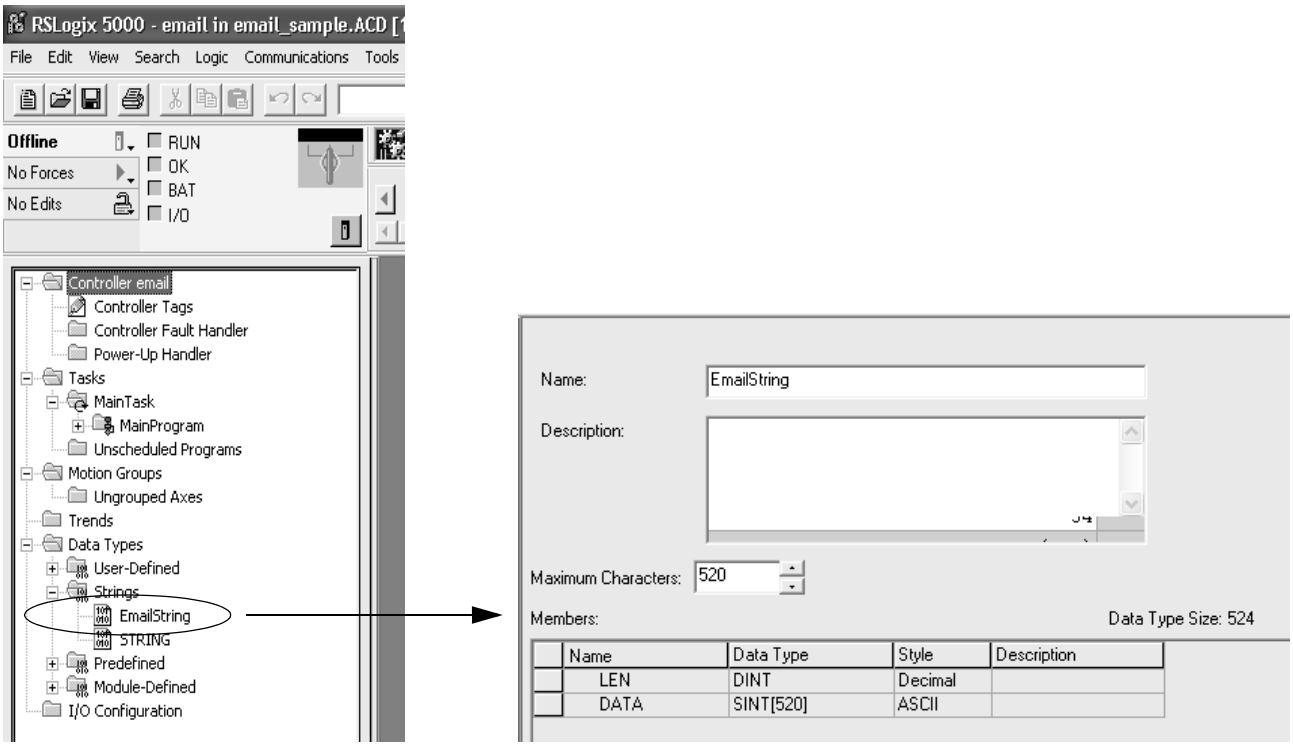
The default STRING data type supports as many as 82 characters. In most cases, this is sufficient to contain the address of the mail server. For example, create tag EmailConfigstring of type STRING:



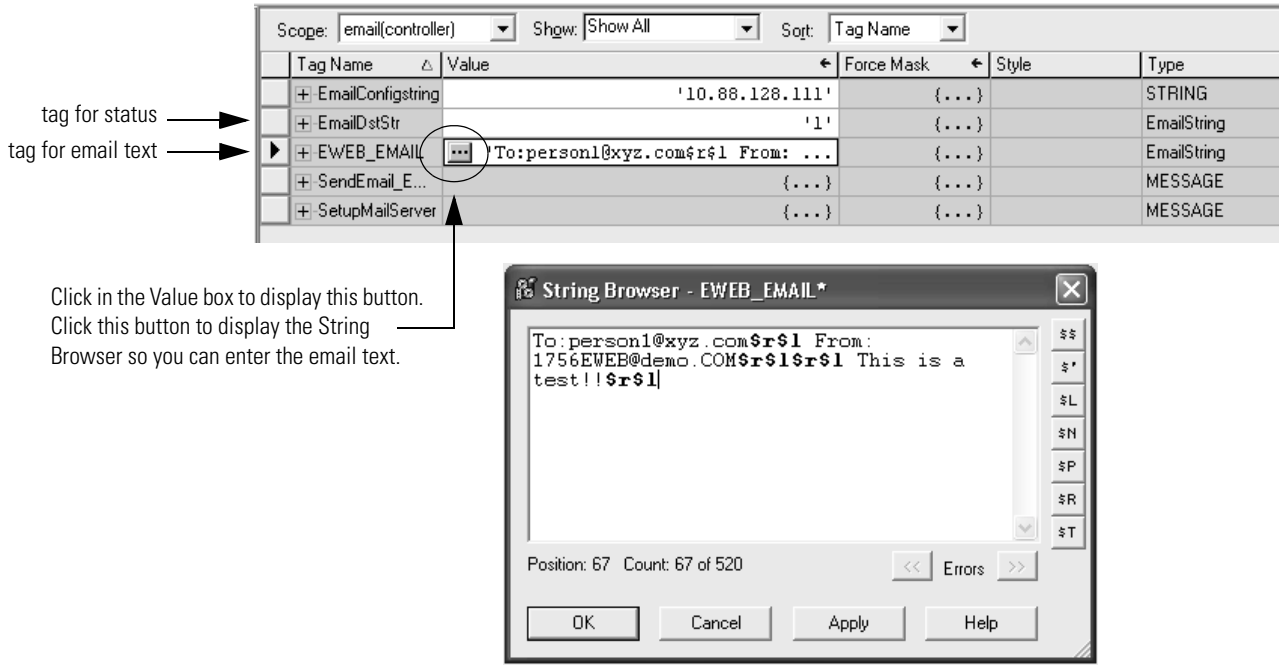
Click in the Value box to display this button. Click this button to display the String Browser so you can enter the IP address or host name of the mail server.



The tags for the email text and transmission status can contain as many as 474 characters. For these tags, you must create a user-defined STRING data type that is larger than the default. For example, create a STRING data type named EmailString.



Create one tag of this new data type to contain the email text. Create a second tag of this new data type to contain the transmission status. For example, create tag EWEB_EMAIL (to contain the email text) and EmailDstStr (to contain the transmission status). Both of these tags are of type EmailString.

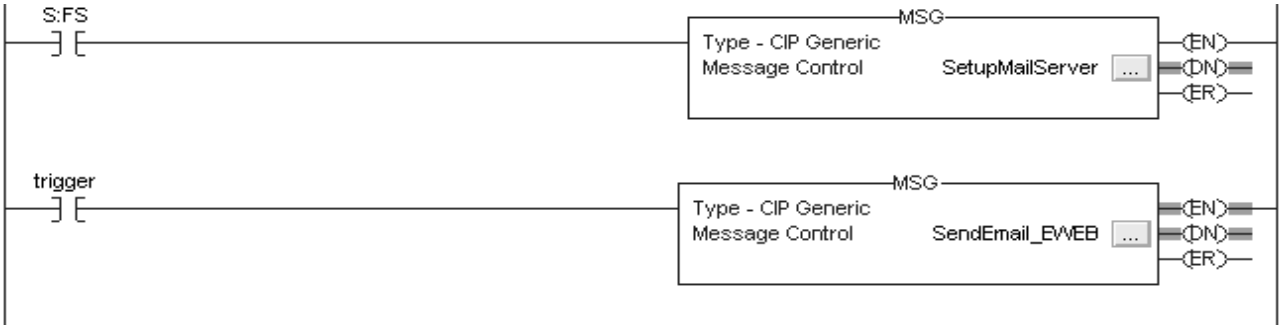


The text of the email does not have to be static. You can program a controller project to collect specific data to be sent in an email. For more information on using ladder logic to manipulate string data, see the *Logix5000 Controllers Common Procedures Programming Manual*, publication 1756-PM001.

See page 3-29 for details on entering email text.

Step 2: Enter the ladder logic

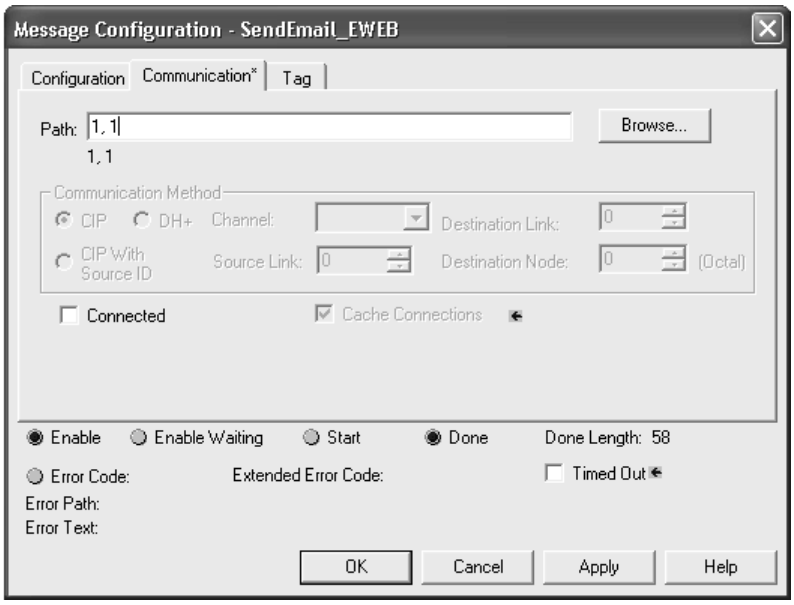
You need two MSG instructions. One MSG instruction configures the mail server. This only needs to be executed once. The next MSG instruction triggers the email. Execute this email MSG instruction as often as needed.



The first rung configures the mail server. The second rung sends the email text.

Step 3: Configure the MSG instruction that identifies the mail relay server

On the Communication tab of the MSG instruction, configure the path for the MSG instruction.



The path starts with the controller and specifies the Ethernet port of the 1769-L32E or 1769-L35E controller. In this example, the path is: 1, 1.

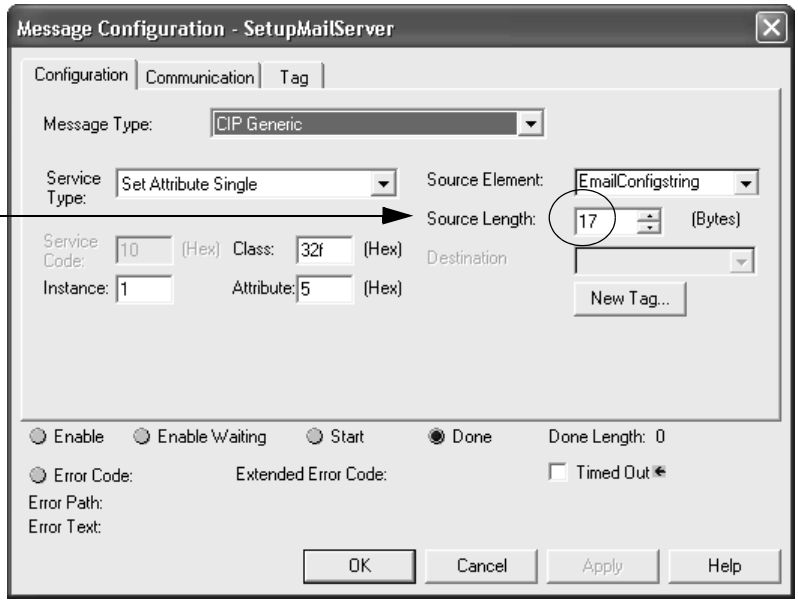
For more information on configuring the path of a MSG instruction, see the *Logix5000 Controllers General Instructions Reference Manual*, publication 1756-RM003.

On the Communication tab of the MSG instruction, configure the MSG parameters for identifying the mail relay server.

Some mail relay servers require a domain name be provided during the initial handshake of the SMTP session. For these mail relay servers, make sure you specify a domain name when you configure the network settings. See page 3-2 for information on configuring the network settings for the controller and specifying a domain name.

The Source Length is the number of characters in the STRING tag that identifies the mail relay server plus 4 characters.

In this example, the tag contains 13 characters.



where:

In this field:	Enter:
Service Type	Set Attribute Single
Instance	1
Class	32f
Attribute	5
Source Element	the STRING tag that contains the IP address or host name of the mail relay server In this example, enter EmailConfigstring
Source Length	the number of characters in the IP address or host name of the mail server plus 4 In this example, enter 17 (13 characters in the IP address 10.88.128.111 + 4)

After the MSG instruction that configures the mail relay server executes successfully, the controller stores the mail relay server information in non-volatile memory. The controller retains this information, even through power cycles, until another MSG instruction changes the information.

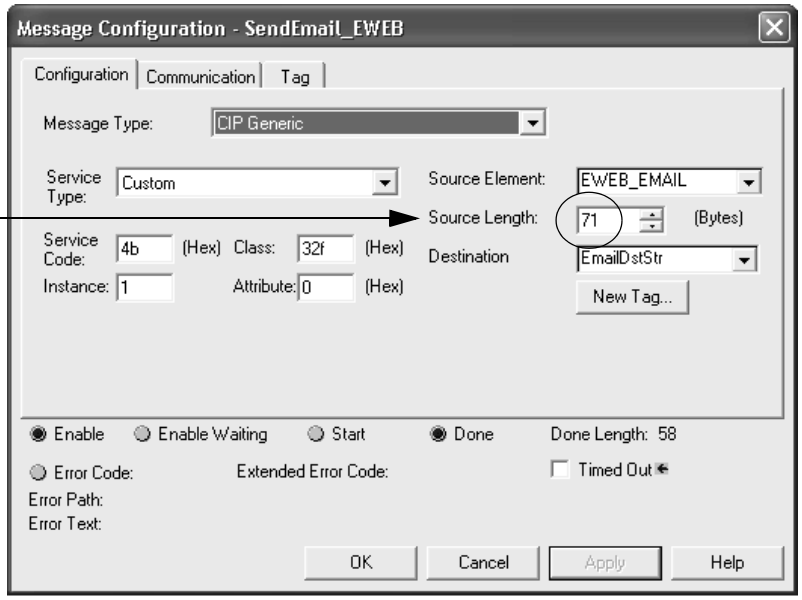
Step 4: Configure the MSG instruction that contains the email text

On the Communication tab of the MSG instruction, configure the path for the MSG instruction. This is the same as for the MSG instruction that identifies the mail relay server (see page 3-26).

On the Configuration tab of the MSG instruction, configure the MSG parameters for sending the email.

The Source Length is the number of characters in the email tag plus 4 characters.

In this example, the email text contains 67 characters, so the Source Length is 71.



where:

In this field:	Enter:
Service Type	Custom
Service Code	4b
Instance	1
Class	32f
Attribute	0
Source Element	the tag that contains the email text This tag is of the STRING data type you created to contain the email text. In this example, enter EWEB_EMAIL which is of type EmailString
Source Length	the number of characters in the email text plus 4 In this example, enter 71 (67 characters in the email + 4)
Destination	a tag to contain the status of the email transmission This tag is also of the STRING data type you created to contain the email text. In this example, enter EmailDstStr which is of type EmailString

Entering the text of the email

Use the string browser to enter the text of the email. In the example above, you enter the email text into the EWEB_EMAIL tag. To include "To:", "From:", and "Subject:" fields in the email, use <CR><LF> symbols to separate each of these fields. The "To:" and "From"" fields are required; the "Subject:" field is optional. Use a second set of <CR><LF> symbols after the last one of these fields you enter. For example:

To: *email address of recipient* rl
From: *email address of sender*rl
Subject: *subject of message* rlrl
body of email message

Use the "From" address to specify where the mail relay server can send an undeliverable email message.

The maximum length of an email message is 474 characters. An additional 4-byte string-length value is added to the tag. As a result, the maximum source length is 478 characters.

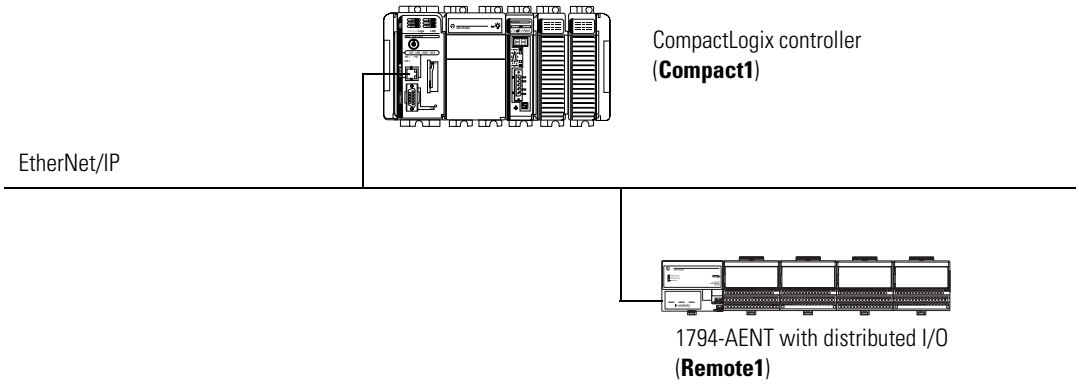
Possible email status codes

Examine the destination element of the email MSG to see whether the email was successfully delivered to the mail relay server. This indicates that the mail relay server placed the email message in a queue for delivery. It does not mean the intended recipient successfully received the email message. Possible codes that could be in this destination element are:

Error Code (hex):	Extended-Error Code (hex):	Description:
0x00	none	Delivery successful to the mail relay server.
0x02	none	Resource unavailable. The email object was unable to obtain memory resources to initiate the SMTP session.
0x08	none	Unsupported Service Request. Make sure the service code is 0x4B and the Class is 0x32F.
0x11	none	Reply data too large. The Destination string must reserve space for the SMTP server reply message. The maximum reply can be 470 bytes.
0x13	none	Configuration data size too short. The Source Length is less than the Source Element string size plus the 4-byte length. The Source Length must equal the Source Element string size + 4.
0x15	none	Configuration data size too large. The Source Length is greater than the Source Element string size plus the 4-byte length. The Source Length must equal the Source Element string size + 4.
0x19	none	Data write failure. An error occurred when attempting to write the SMTP server address (attribute 4) to non-volatile memory.
0xFF	0x0100	Error returned by email server; check the Destination string for reason. The email message was not queued for delivery.
	0x0101	SMTP mail server not configured. Attribute 5 was not set with a SMTP server address.
	0x0102	"To:" address not specified. Attribute 1 was not set with a "To:" address AND there is not a "To:" field header in the email body.
	0x0103	"From:" address not specified. Attribute 2 was not set with a "From:" address AND there is not a "From:" field header in the email body.
	0x0104	Unable to connect to SMTP mail server set in Attribute 5. If the mail server address is a hostname, make sure that the device supports DNS, and that a Name Server is configured. If the hostname is not fully qualified, i.e., "mailhost" and not "mailhost.xx.yy.com" then the domain must be configured as "xx.yy.com". Try "ping <*mail server address*>" to insure the mail server is reachable from your network. Also try "telnet <*mail server address*> 25" which attempts to initiate a SMTP session with the mail server via telnet over port 25. (If you connect then enter "QUIT").
	0x0105	Communication error with SMTP mail server. An error occurred after the initial connection with the SMTP mail server. See the ASCII text following the error code for more details as to the type of error.
	0x0106	SMTP mail server host name DNS query did not complete. A previous send service request with a host name as the SMTP mail server address did not yet complete. Note that a timeout for a DNS lookup with an invalid host name can take up to 3 minutes. Long timeouts can also occur if a domain name or name server is not configured correctly.

Example 1: CompactLogix Controller and Distributed I/O

In the following example, one CompactLogix controller controls distributed I/O through a 1794-AENT module.



Controlling distributed I/O

This example has Compact1 controlling the I/O connected to the remote 1794-AENT module. The data the CompactLogix controller receives from the distributed I/O modules depends on how you configure the I/O modules. You can configure each module as a direct connection or as rack optimized. One chassis can have a combination of some modules configured as a direct connection and others as rack optimized.

All analog modules require direct connections. Diagnostic modules support rack-optimized connections, but require direct connections to take full advantage of their diagnostic features.

Total connections required by Compact1

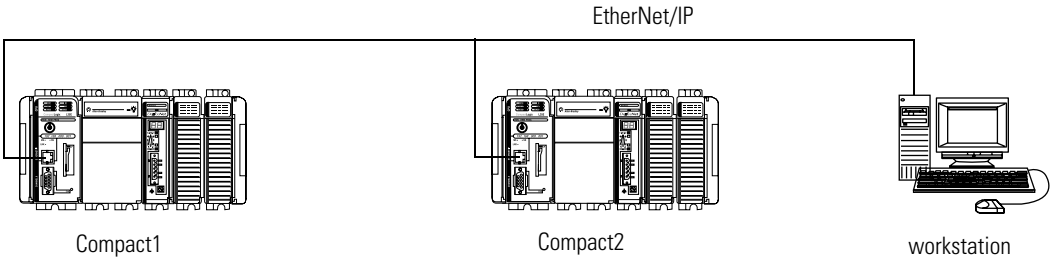
The following table calculates the connections used in this example.

Connection:	Amount:
Compact1 to 4 distributed I/O modules (through 1794-AENT) • all I/O modules configured as direct connection • no connection to the 1794-AENT	4
total connections used:	4

If you configured the distributed I/O modules as rack-optimized, you would only need a rack-optimized connection to the 1794-AENT, reducing the above example by 3 connections.

Example 2: Controller to Controller

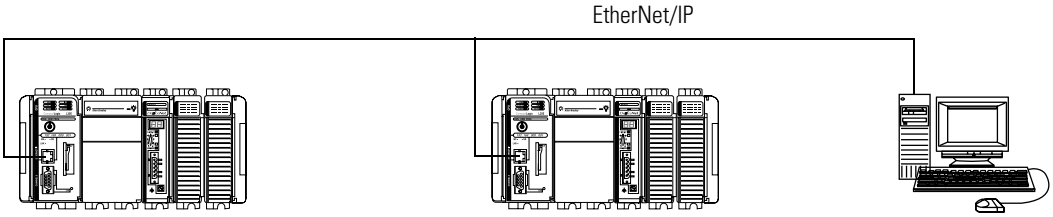
In the following example, one EtherNet/IP CompactLogix controller communicates with another EtherNet/IP CompactLogix controller over EtherNet/IP. Each controller has its own local I/O



Producing and consuming tags

Produced data must be of DINT or REAL data type or an array or structure. You can use a user-defined structure to group BOOL, SINT, and INT data to be produced. You can produce a base, alias, or consumed tag.

The consumed tag must have the same data type as the produced tag in the originating controller. The controller performs type checking to ensure proper data is being received.



Compact1
TagA DINT
TagB REAL

Compact2 (controllerb)
TagA DINT
TagB REAL

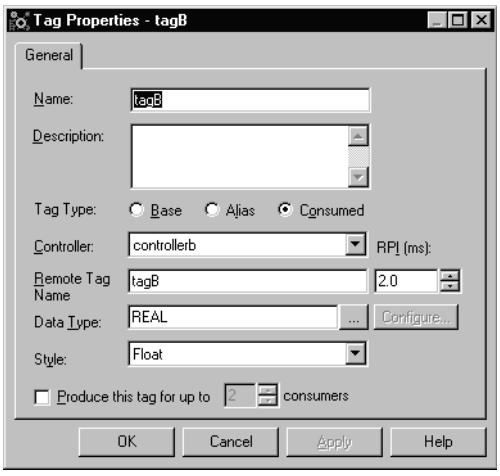
workstation

This example shows Compact1 as producing TagA and consuming TagB:

	☑	⊞-tagA			DINT	Decimal
	☐	tagB		controllerb:tagB	REAL	Float
✳	☐					

TagA

TagB



Each produced tags requires one connection for the producing controller and an additional connection for each consuming controller. Each consumed tag requires one connection.

Sending a MSG instruction

To send a MSG from Compact1 to Compact2:

1. For Compact1, create a controller-scoped tag and select the MESSAGE data type.

2. Enter a MSG instruction.

   In this example logic, a message is sent when a specific condition is met. When count_send is set, send count_msg.

3. Configure the MSG instruction. On the Configuration tab:

For this item:	Specify:
Message Type	CIP Data Table Read or CIP Data Table Write
Source Tag	Tag containing the data to be transferred
Number of Elements	Number of array elements to transfer
Destination Tag	Tag to which the data will be transferred

4. On the Communication tab, specify the communication path.

   Use the Browse button to select the device that will receive the MSG instruction. The communication path in this example is:

For this item:	Specify:
Communication Path	1,1,2,xxx.xxx.xxx.xxx,1,0 where: 1 is the virtual backplane of Compact 1 1 is the slot of the Ethernet port in the controller (note, the 1,1 displays as LocalENB) 2 is the EtherNet/IP network 100.100.115.11 is the IP address of Compact2 1 is the virtual backplane of Compact2 0 is the controller slot of Compact2

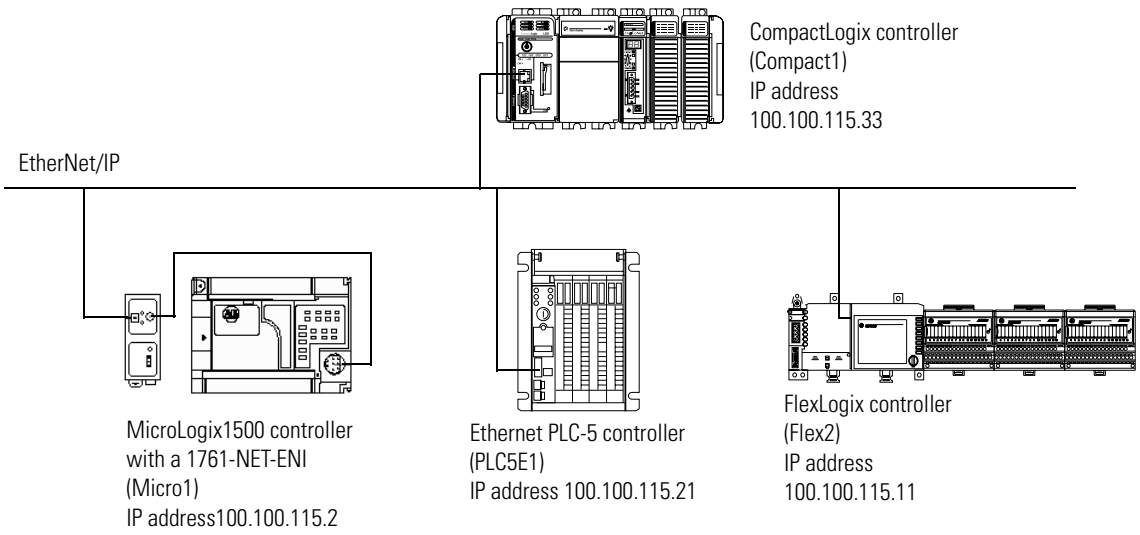
Total connections required by Compact1

The following table calculates the connections used in this example.

Connection:	Amount:
connected, cached MSG from Compact1 to Compact2	1
produced TagA produced from Compact1 to Compact2 other consumer (2 are configured)	 1 1
consumed TagB	1
total connections used:	4

Example 3: CompactLogix Controller to Other Devices

In the following example, one CompactLogix controller communicates with a MicroLogix 1500 controller, an Ethernet PLC-5 controller, and a FlexLogix controller over an EtherNet/IP network.



Sending a MSG instruction to another Logix-based controller

You configure a MSG instruction to other Logix-based controllers the same as you do for a CompactLogix controller. All Logix-based controllers follow the same MSG configuration requirements.

1. In the CompactLogix controller, create a controller-scoped tag and select the MESSAGE data type. Enter a MSG instruction. See Example 2 above for an example.

2. Configure the MSG instruction. On the Configuration tab:

For this item:	Specify:
Message Type	CIP Data Table Read or CIP Data Table Write
Source Tag	Tag containing the data to be transferred
Number of Elements	Number of array elements to transfer
Destination Tag	Tag to which the data will be transferred

3. On the Communication tab, specify the communication path.

   Use the Browse button to select the device that will receive the MSG instruction. The communication path in this example is:

For this item:	Specify:
Communication Path	1,1,2,100.100.115.11,1,0 where: 1 is the virtual backplane of Compact 1 1 is the slot of the Ethernet port in the controller (note, the 1,1 displays as LocalENB) 2 is the EtherNet/IP network 100.100.115.11 is the IP address of Flex2 1 is the virtual backplane of Flex2 0 is the controller slot of Flex2

Sending a MSG instruction to a PLC-5E processor

Configuring a MSG instruction for a PLC-5 processor requires different MSG configuration and PLC/SLC mapping.

1. In the CompactLogix controller, create a controller-scoped tag and select the MESSAGE data type. Enter a MSG instruction. See Example 2 above for an example.

2. Configure the MSG instruction. On the Configuration tab:

For this item:	Specify:
Message Type	PLC-5 Typed Read or PLC-5 Typed Write or PLC-5 Word Range Read or PLC-5 Word Range Write
Source Tag	Tag containing the data to be transferred
Number of Elements	Number of array elements to transfer
Destination Tag	Tag to which the data will be transferred

The source and destination data types depend on the message type you select:

Type of Logix MSG instruction:	Source:	Destination:
PLC-5 Typed Read	any integer element (such as B3:0, T4:0.ACC, C5:0.ACC, N7:0, etc.)	SINT, INT, or DINT tag
	any floating point element (such as F8:0, PD10:0.SP, etc.)	REAL tag
PLC-5 Typed Write	SINT or INT tag	any integer element (such as B3:0, T4:0.ACC, C5:0.ACC, N7:0, etc.)
	REAL tag	any floating point element (such as F8:0, PD10:0.SP, etc.)
PLC-5 Word Range Read	any data type (such as B3:0, T4:0, C5:0, R6:0, N7:0, F8:0, etc.)	SINT, INT, DINT, or REAL
PLC-5 Word Range Write	SINT, INT, DINT, or REAL	any data type (such as B3:0, T4:0, C5:0, R6:0, N7:0, F8:0, etc.)

3. On the Communication tab, specify the communication path.

   Use the Browse button to select the device that will receive the MSG instruction. The communication path in this example is:

For this item:	Specify:
Communication Path	1,1,2,100.100.115.21 where: 1 is the virtual backplane of Compact 1 1 is the slot of the Ethernet port in the controller (note, the 1,1 displays as LocalENB) 2 is the EtherNet/IP network 100.100.115.21 is the IP address of PLC5E1

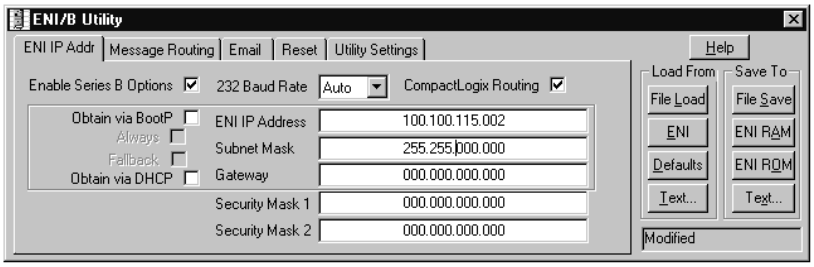
Sending a MSG instruction from a PLC-5E processor to a CompactLogix controller

The PLC-5E processor supports logical ASCII addressing so you do not have to map a compatibility file for MSG instructions initiated by a PLC-5 processor. Place the CompactLogix tag name in double quotes (“).

Type of MSG Instruction:	Example Source and Destination:	
PLC-5 writes to CompactLogix	source element	*N7:10*
	destination tag	*“array_1”*
PLC-5 reads from CompactLogix	source tag	*“array_1”*
	destination element	*N7:10*

Sending a MSG instruction to a MicroLogix 1500 controller with a 1761-NET-ENI module

1. Use the ENI utility to make sure the configuration for the 1761-NET-ENI module has the Enable Series B Options and CompactLogix Routing features enabled.



2. In the CompactLogix controller, create a controller-scoped tag and select the MESSAGE data type. Enter a MSG instruction. See Example 2 above for an example.

3. Configure the MSG instruction. On the Configuration tab:

For this item:	Specify:
Message Type	SLC Typed Read or SLC Typed Write
Source Tag	Tag containing the data to be transferred Make sure this tag is an INT
Number of Elements	Number of array elements to transfer
Destination Tag	Tag to which the data will be transferred

4. On the Communication tab, specify the communication path.

   Use the Browse button to select the device that will receive the MSG instruction. The communication path in this example is:

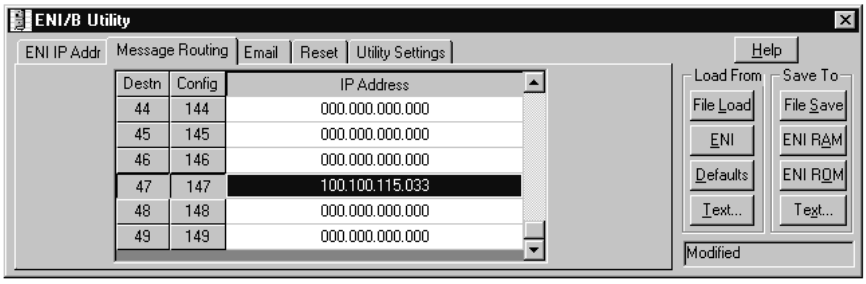
For this item:	Specify:
Communication Path	1,1,2,100.100.115.2 where: 1 is the virtual backplane of Compact 1 1 is the slot of the Ethernet port in the controller (note, the 1,1 displays as LocalENB) 2 is the EtherNet/IP network 100.100.115.2 is the IP address of Micro1

Sending a MSG instruction from a MicroLogix 1500 controller with a 1761-NET-ENI module to a CompactLogix controller

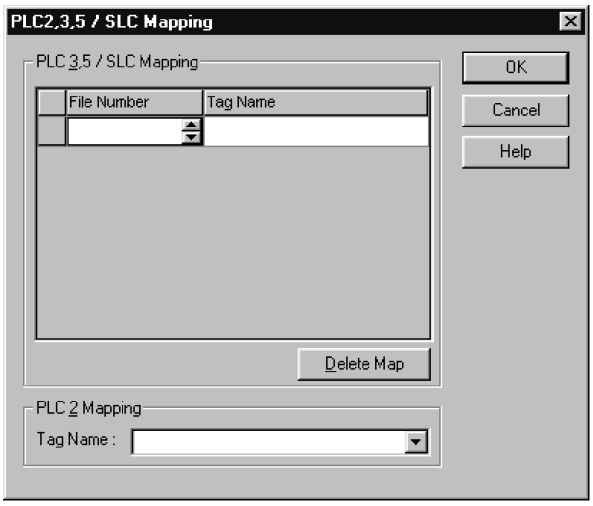
If the MSG instruction originates from the MicroLogix 1500 controller, make sure the configuration for the 1761-NET-ENI module can recognize the CompactLogix controller:

1. Use the ENI utility to make sure the configuration for the 1761-NET-ENI module has the Enable Series B Options and CompactLogix Routing features enabled.

2. Use the ENI utility to add the IP address of the CompactLogix controller to the configuration for the 1761-NET-ENI module. Assign the IP address of the CompactLogix controller to one of the Logix destination locations (45-49) on the Message Routing tab.



You must also map the logical address of the MicroLogix tag (i.e., N16) to a value (tag) in the CompactLogix controller.



- Type the file number of the logical address in the MicroLogix controller.
- Type or select the controller-scoped (global) tag in the CompactLogix controller that supplies or receives data for the file number. (You can map multiple files to the same tag.) This tag must be an INT tag.

Total connections required by Compact1

The following table calculates the connections used in this example.

Connection:	Amount:
connected, cached MSG from Compact1 to Flex2	1
connected, cached MSG from Compact1 to PLC-5E1	1
connected, cached MSG from Compact1 to Micro1	1
total connections used:	3

Example 4: Receiving Messages from Other Devices

When other devices send messages to the CompactLogix controller, the path for the message must identify the controller. Configure a CIP-type message in the originating device. Specify the path the CompactLogix controller as:

xxx.xxx.xxx.xxx,1,0

where:

xxx.xxx.xxx.xxx is the IP address of the controller

1 is the virtual backplane of controller

0 is the controller slot of the controller

Communicating with Devices on a DeviceNet link

Using This Chapter

Configuring Your System for a DeviceNet Link

Select the appropriate DeviceNet interface depending on the application and how the controller interacts with the devices:

If your application:	Select this interface:	Description:
• communicates with other DeviceNet devices • uses the controller as a master or slave on DeviceNet • uses the controller Ethernet port or serial port for other communications	1769-SDN DeviceNet scanner module	The scanner acts as an interface between DeviceNet devices and the CompactLogix controller. The scanner lets the controller: • read inputs from slave devices • write outputs to slave devices
• accesses remote Compact I/O over a DeviceNet network • sends remote I/O data for as many as 30 modules back to scanner or controller	1769-ADN DeviceNet adapter module	The adapter: • interfaces with as many as 30 Compact I/O modules • communicates to other network system components (typically a controller or scanner and/or programming terminals) over the DeviceNet network

You can also bridge from EtherNet/IP to DeviceNet through a CompactLogix controller with a 1769-SDN. This bridging lets you:

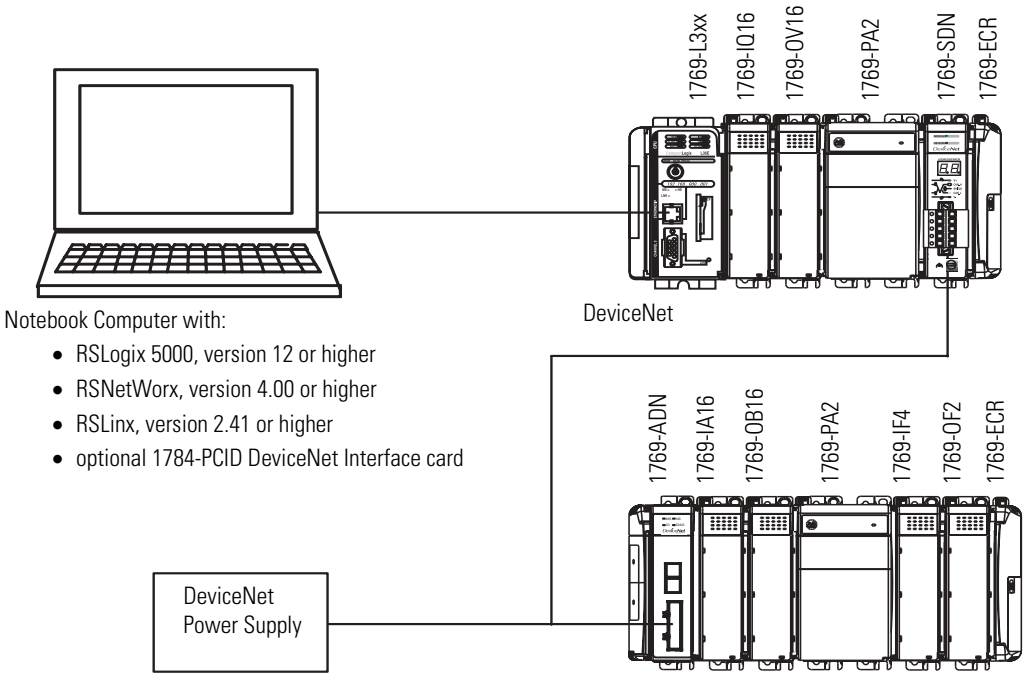
- configure the 1769-SDN scanner and its DeviceNet devices using RSNetWorx connected via an EtherNet/IP connection.
- flash 1769-SDN firmware via an EtherNet/IP connection.

To bridge from EtherNet/IP to DeviceNet, you need:

- 1769-SDN with firmware revision 2.2 or greater
- most current EDS files for both the controller and the 1769-SDN

Example 1: Controlling DeviceNet Devices

This example uses a 1769-SDN scanner module in the local CompactLogix system to control the I/O attached to a 1769-ADN adapter module.



This example describes:

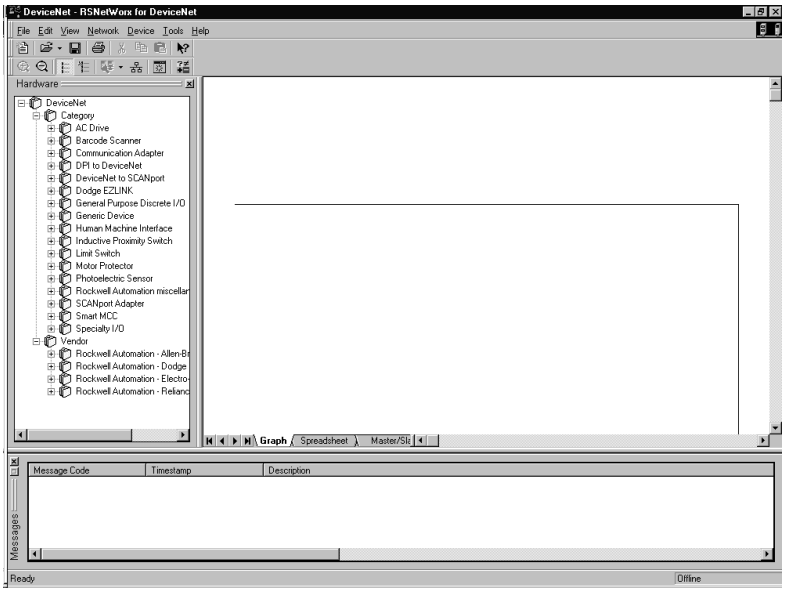
- using RSNetWorx for DeviceNet to assign node addresses to the 1769-SDN and the 1769-ADN and map the adapter's image into the scanner
- creating a CompactLogix project including the necessary configuration for the 1769-SDN DeviceNet scanner module
- controlling outputs and reading inputs with the distributed I/O via DeviceNet

The computer does not have to be connected to the DeviceNet network. The connection path in this example is through the controller. If you have a 1769-SDN module with firmware revision 2.2 or greater, you can bridge through the EtherNet/IP port of a 1769-L32E or 1769-L35E controller to the 1769-SDN module.

Step 1: Configuring the 1769-ADN adapter

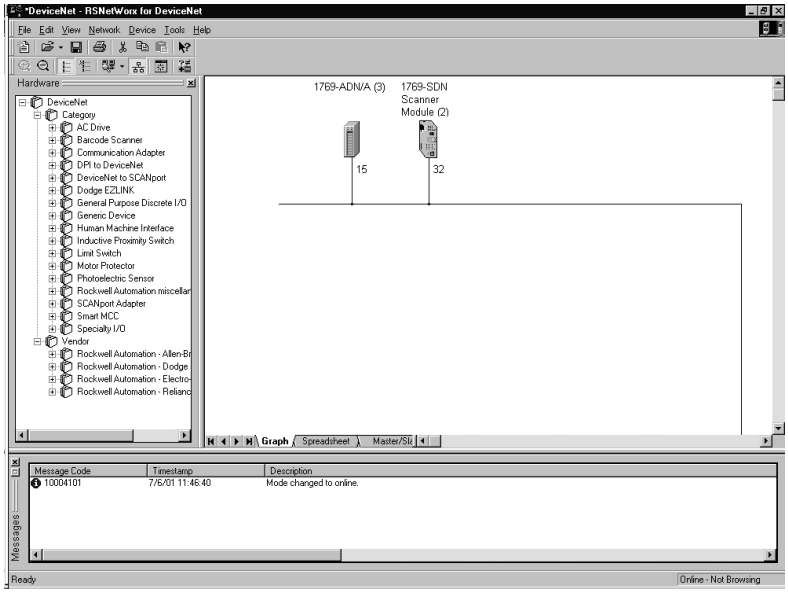
1. Start RSNetWorx.

2. Select Network → Online. The RSLinx communication driver screen appears. Choose the appropriate driver depending on whether the computer is directly connected to DeviceNet or you are bridging through the controller's EtherNet/IP or ControlNet port.



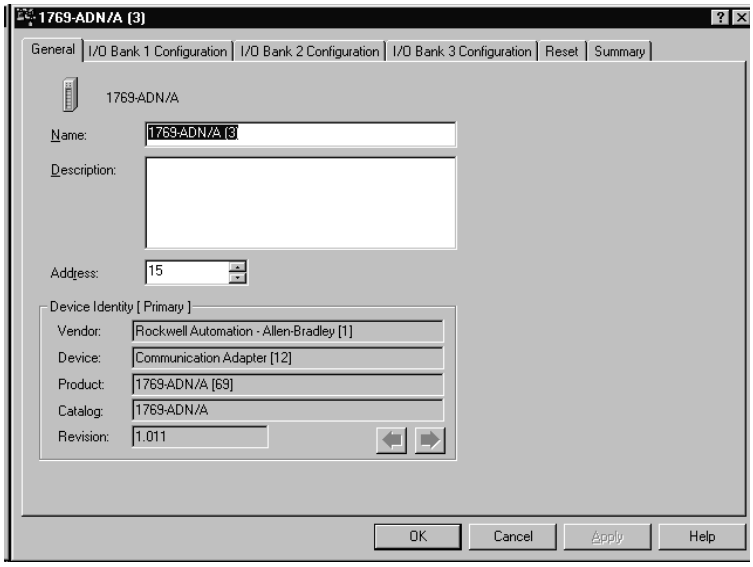
3. The software then prompts you to either upload or download. Choose upload. RSNetWorx browses the network for valid devices. The online screen should look something like the following, where the 1769-ADN is node 15, and the 1769-SDN is node 32 for this example.

If you had connected via DeviceNet, such as through a 1784-PCID card, the communication card would also appear as a node on the DeviceNet network.



continued

4. Right click on the 1769-ADN and choose Properties



5. Click on the I/O Bank 1 Configuration tab, then choose upload when prompted. The actual 1769-ADN I/O layout appears. From this screen you can configure the I/O modules in the 1769-ADN system by simply clicking on the slot number box associated with each I/O module.

6. When the I/O modules are configured, click on the Summary tab. Note the number of bytes of input and output data. This will be used later when adding the adapter to the 1769-SDN's scanlist.

7. Click Apply, then OK to save the configuration and download it to the adapter.

For this example, you only configure the two analog modules. For more information about analog modules, see the *Compact I/O Analog Modules User Manual*, publication 1769-UM002. Only analog and specialty modules are configurable. Discrete I/O modules, power supplies, and end caps are not configurable.

TIP

Configuration changes made to the adapter or any of its I/O modules with RSNetWorx will not be saved or downloaded to the adapter once the adapter is configured in a scanner's scanlist.

To make configuration changes, the controller must be placed into the Program mode and the adapter must be temporarily removed from the scanner's scanlist.

Step 2: Setting up the 1769-SDN scanlist

The 1769-SDN series B scanner supports automatic device recovery (ADR). An ADR tab appears in the scanlist window in RSNetWorx for DeviceNet for series B scanners so you can enable the ADR feature. This feature:

- automates the replacement of a failed slave device on a DeviceNet network by returning the device to the prior level of operation
- includes automatic address recovery which allows a slave device to be removed from the network and replaced with another identical slave device that is residing on the network at node 63 and is not in the scanlist.
- includes configuration recovery which allows a slave device to be removed from the network and replaced with an identical device with the same configuration

IMPORTANT To maintain proper mapping between the controller tags and the 1769-SDN scanlist, make sure you are using version 4.12 or greater of RSNetWorx for DeviceNet software and the most current 1769-SDN EDS files. This updated software lets you select the CompactLogix controller as a mapping configuration, which ensures that the scanlist and controller tags properly coincide.

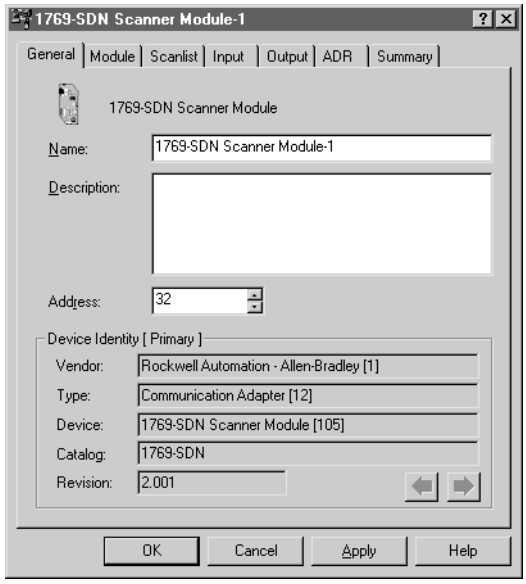
RSLogix 5000 software, version 12, includes a 1769-SDN profile. This profile provides two modifications to the previous method of using the generic 1769 profile to configure the 1769-SDN:

- The new profile separates the module status and the configuration information from the I/O data. The profile automatically creates one set of tags for module status and configuration and another set of tags for I/O data.
- The 1769-SDN profile uses DINT tags for I/O data. The generic profile used INT tags.

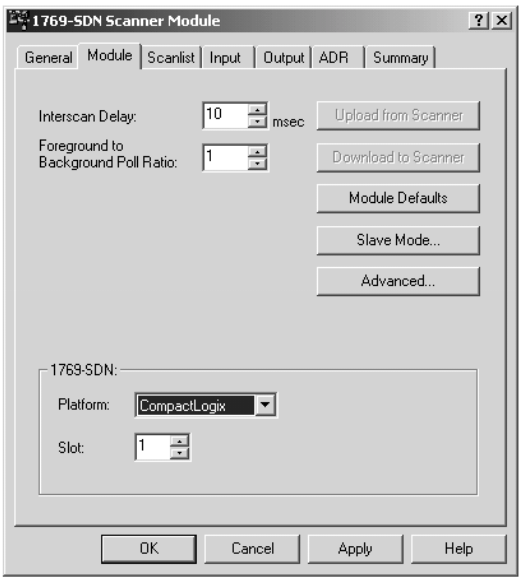
IMPORTANT If you are converting a project from a 1769-L20, -L30 controller to a 1769-L3xx controller and the project contains a 1769-SDN, you might want to leave the generic profile for the 1769-SDN in the project rather than converting it to the new 1769-SDN profile. The new 1769-SDN profile uses DINTs instead of INTs for data and the scanlist is configured differently than for the generic profile.

Use RSNetWorx for DeviceNet software to create the scanlist.

1. Right click on the 1769-SDN and choose Properties.



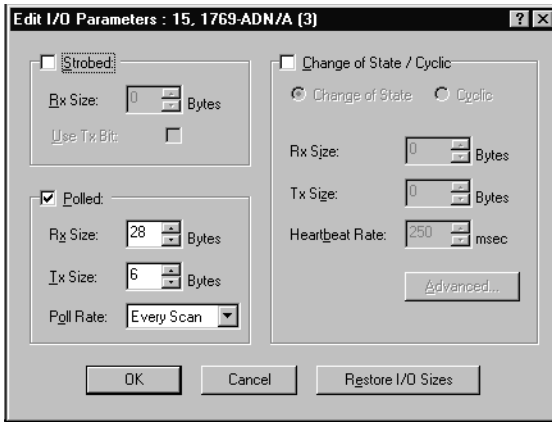
2. Select the Module tab. Pull down the platform combination box and select CompactLogix. This lets the software know that the scanner is being used with a CompactLogix controller. The scanlist will be mapped to coincide with the 1769-SDN profile in RSLogix 5000 software. Also, select the slot number of the 1769-SDN module.



3. Click the Scanlist tab, then click Upload when prompted. The area on the left is called "Available Devices" and the area on the right is called "Scanlist". The 1769-ADN adapter should be on the left.

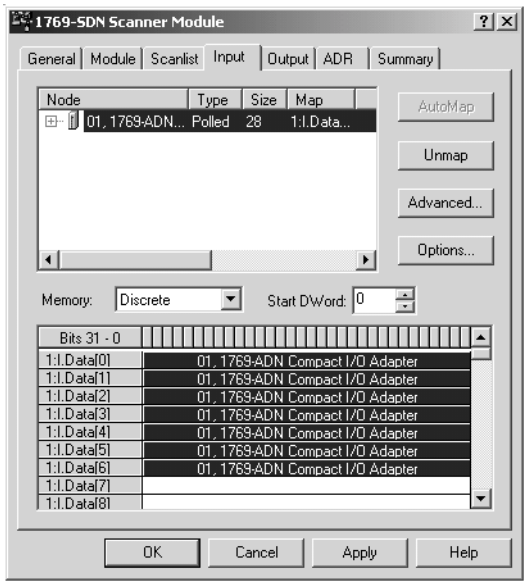
4. Click on the adapter, then click on the single arrow pointing to the right. This moves the adapter from Available Devices to the scanner's scanlist.

5. Click on the Edit I/O Parameters button



6. Verify that the Rx Size and Tx Size are correct. The Tx (Transmit) and Rx (Receive) sizes correspond to the total number of output and input bytes noted from the adapter's summary page. In this example, the scanner transmits 6 bytes to the adapter (output data) and receives 28 bytes from the adapter (input data). Click OK when finished with this screen.

7. Click on the Input tab.



Click Apply and then click OK.

Mapping starts at word 0 for both the input and the output data image. The input status and output configuration words are no longer included with the I/O data scanlist. Use the status and configuration tags created in RSLogix 5000 software to read status or set configuration bits.

TIP

The input and output data being exchanged by the scanner and adapter is packed data. This means that there is no special structure to it that makes it obvious which I/O module it is associated with.

To establish which data is from which module, you must list the number of input and output words each module has. Then, based on its position in the I/O bank, you can determine where any module's data is in the controller's I/O tags.

Transferring data

There are 28 bytes of input data and 6 bytes of output data for this example. The I/O modules in the adapter's system are:

Module	Input	Output
ADN Status Information (added by the 1769-ADN)	1 DINT word	0 words
1769-IA16	1/2 DINT word	0 words
1769-OB16	1/2 DINT word	1/2 DINT word
1769-IF4	3 DINT words	0 words
1769-OF2	2 DINT words	1 DINT word
Total Words	7 DINT words	1 1/2 DINT words
Total Bytes	28 bytes	6 bytes

The total is 7 DINT words or 28 input bytes. The first DINT word is adapter status, leaving 6 DINT words (24 bytes) for data. The input data maps to the controller's input data tag at the following word locations:

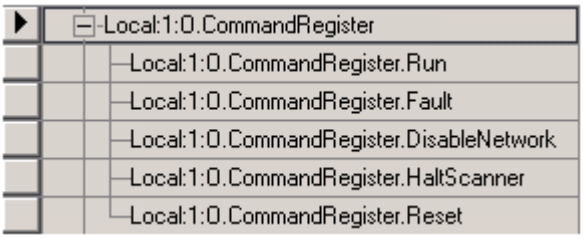
Location	Description
Word 0	1769-ADN status information
Word 1	1769-IA16 module's input word
Word 1	1769-OB16 module's input data (output data echo)
Words 2-4	1769-IF4 module's input data
Words 5-6	1769-OF2 module's input data

The output data can be determined in a similar manner. This data begins with word 0 of the output tag in the controller as follows:

Location	Description
Word 0	1769-OB16 module's output word
Words 0-1	1769-OF2 module's output words

Module command array

The module command array is the primary control interface between your control program and the module. In RSLogix 5000 software, the CommandRegister tag structure is as follows:



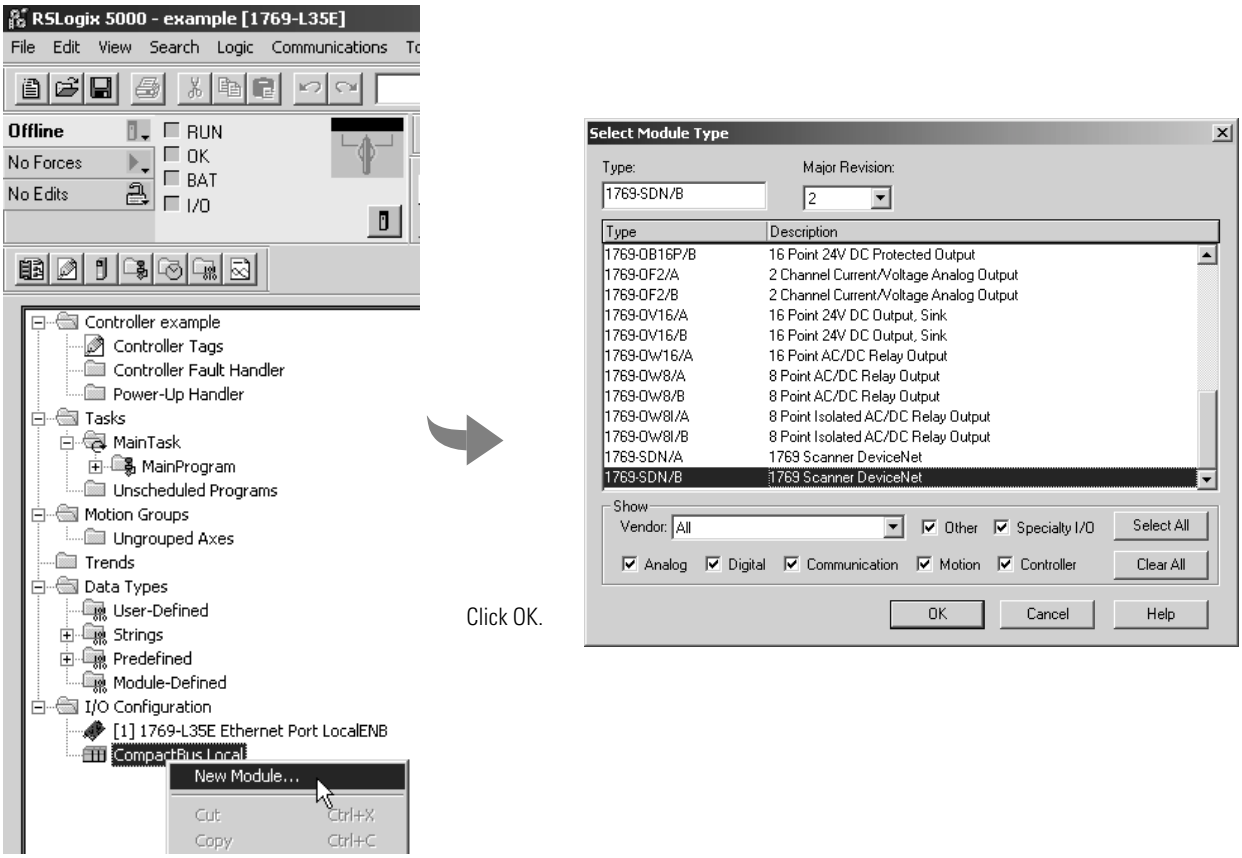
Output Word	Bit	Description	Behavior
0	0	Run	This bit controls when the module scans its mapped slave devices. When set (1), the scanner will process I/O data as defined by its scanlist. To actually scan the network the Fault and Disable Network command bits must be clear (0).
	1	Fault	When set, the scanner's I/O mode will be Halt; messaging will still operate. The fault bit is primarily used to artificially set the slave devices into a fault state due to some event or condition within the control program.
	2	Disable Network	When set, the scanner is functionally removed from the network.
	3	HaltScanner	When set, the scanner stops scanning its mapped slave devices.
	4	Reset	Restarts access to the DeviceNet network.

Download the scanner information to the 1769-SDN

After you configure the scanlist, you need to download that information to the 1769-SDN module.

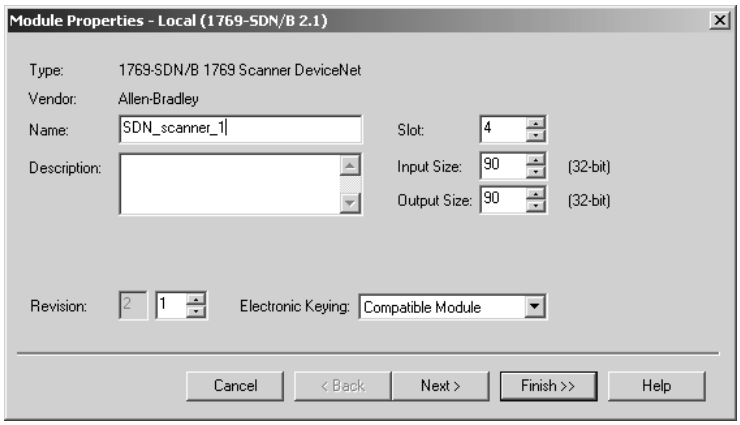
Step 3: Creating a project for the CompactLogix controller

1. In the Controller Organizer, select the CompactBus. Right-click the selected rail and select New Module.
2. Select the 1769-SDN/B module.



3. Configure the module. Use the module wizard to specify characteristics for the module. Click Next to continue through the wizard.

Click Finish when you are done. The completed module appears in the Controller Organizer.



IMPORTANT Version 12 of RSLogix 5000 software added a complete profile for configuring a 1769-SDN module in a CompactLogix system. To take advantage of this profile and the enhanced messaging capabilities of the 1769-SDN module:

- download and install new EDS files for the 1769-SDN module
- update the firmware of the 1769-SDN module

See the *CompactLogix Controller Release Notes*, publication 1769-RN006 for details on downloading and installing EDS files and firmware.

All tags for I/O modules are automatically created when the profiles for these modules are configured. Double click on Controller Tags in the controller organizer to view these tags. Each I/O module slot has Input, Output and Configuration tags created, if they apply. These tags are structured as:

Tag	Definition
Local:s:I	s is the slot number I represents Input Data
Local:s:O	O represents Output Data
Local:s:C	C represents Configuration Data

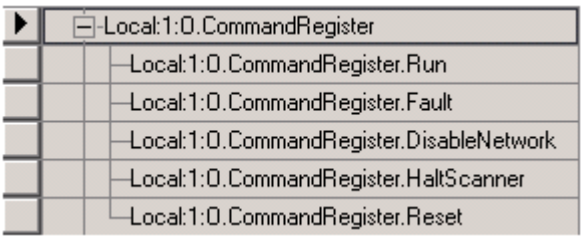
If the 1769-SDN is in slot 1, the input addresses for the scanner are:

Tag	Definition
Local:1:I.Data[0]	1769-ADN Status Information
Local:1:I.Data[1]	Input Data from 1769-IA16
Local:1:I.Data[1]	Input (output echo) Data from 1769-OB16
Local:1:I.Data[2] through Local:3:I.Data[4]	Input Data from 1769-IF4
Local:1:I.Data[5] through Local:3:I.Data[6]	Input Data from 1769-OF2

This output addresses for the scanner are:

Tag	Definition
Local:1:O.Data[0]	Output data for 1769-OB16
Local:1:O.Data[0] through Local:3:O.Data[1]	Output data for 1769-OF2

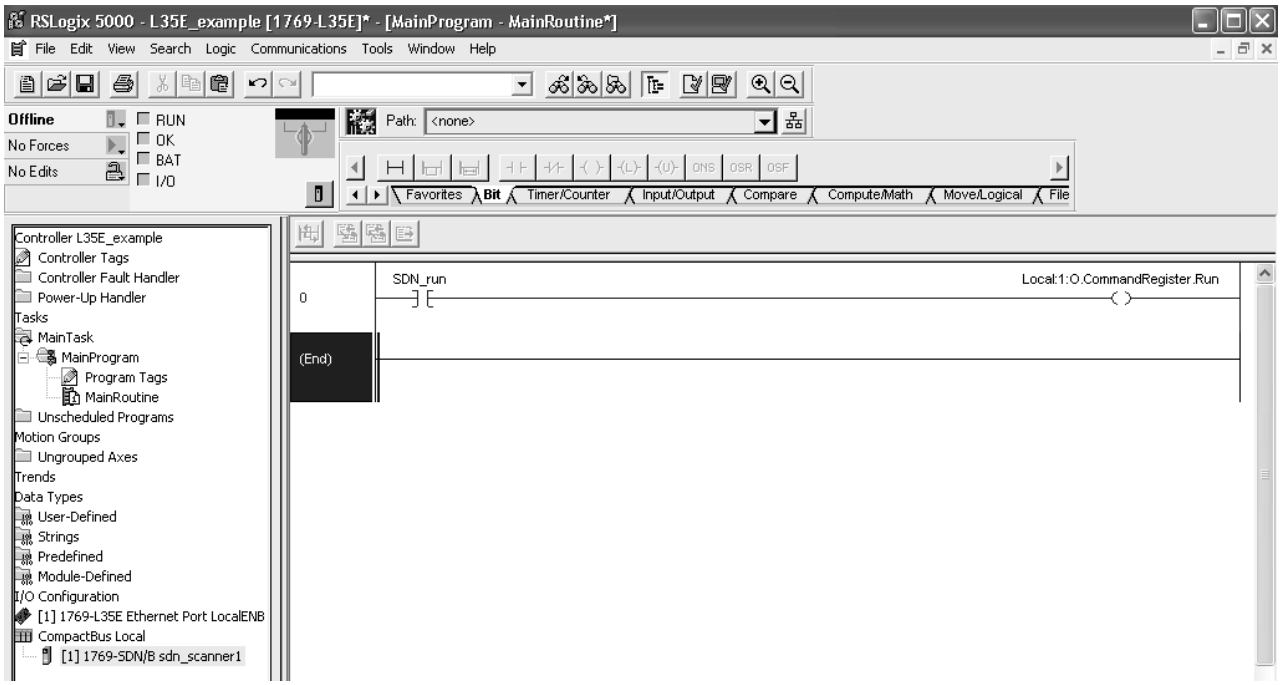
The controller uses the CommandRegister (Local:1:O.CommandRegister) to send commands to the scanner



Step 4: Enter program logic

The program for this example consists of a single rung that is used to place the scanner into the RUN mode.

To place the scanner in the Run mode when the CompactLogix controller is in the Run mode, either set “SDN_RUN” to a 1, or remove it from the program. When “SDN_RUN” is removed, the scanner’s Run bit is always in Run when the controller is in Run.



When your program is written, verify and save it, then download it to your controller to run and test your system.

Example 2: Bridging through Ethernet to DeviceNet

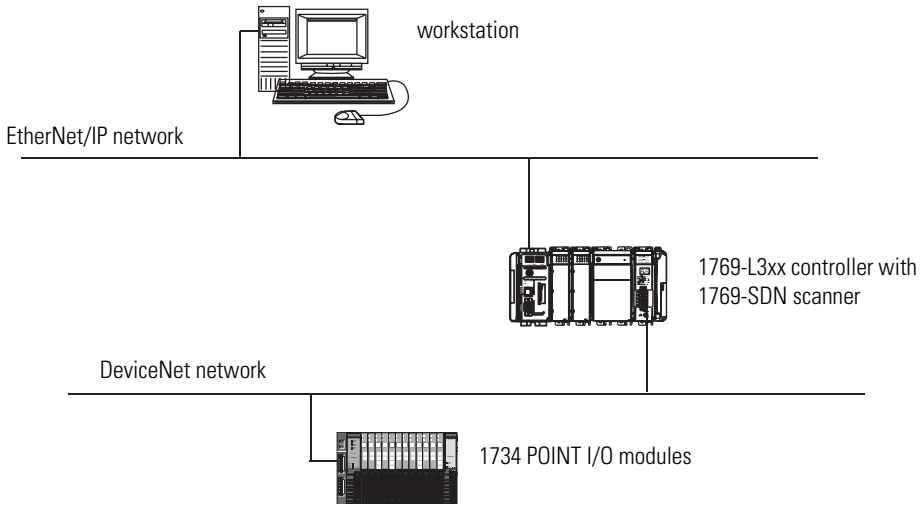
You can use the controller to bridge messages between devices; the controller supports one connected and one unconnected message between devices. The controller will only bridge messaging data (not I/O data), and there is limited buffering to store waiting messages that bridge networks.

IMPORTANT The update time of local I/O modules may increase when the controller is bridging messages.

Bridging over the CompactLogix controller should be targeted toward applications that are not real time dependent, such as RSLogix 5000 program downloads and ControlFlash updates.

The 1769-L32E, -L35E controller can bridge from the serial or EtherNet/IP port to DeviceNet. The 1769-L31 controller can bridge from either serial port to DeviceNet.

For example, a message originates at a workstation and bridges through a CompactLogix system to DeviceNet devices.



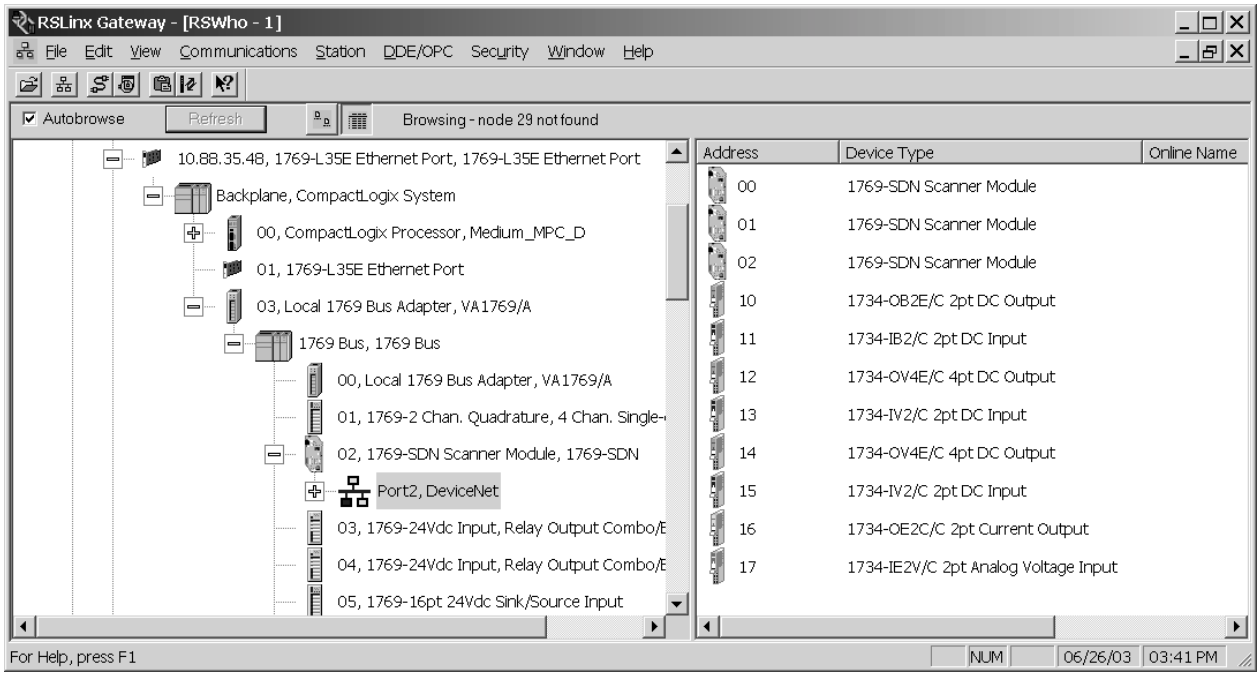
The CompactLogix controller can bridge these combinations of networks:

Messages that originate on this network:	And end on this network:
EtherNet/IP	DeviceNet RS-232 serial
RS-232 serial	EtherNet/IP DeviceNet

Bridging from Ethernet to DeviceNet lets you use one workstation to program the CompactLogix controller on Ethernet, as well as to maintain DeviceNet devices via RSNetWorx for DeviceNet software

Maintaining DeviceNet devices via a bridge

Use RSNetWorx for DeviceNet software to manage your DeviceNet network and devices. This screen shows how you would navigate through an Ethernet-to-DeviceNet bridge to select specific devices. You navigate through the 1769 bus to select the 1769-SDN module to get to DeviceNet devices.



Sending a MSG instruction from the controller to a DeviceNet device

1. For Compact1, create a controller-scoped tag and select the MESSAGE data type.

2. Enter a MSG instruction.

   In this example logic, a message is sent when a specific condition is met. When count_send is set, send count_msg.

3. Configure the MSG instruction. On the Configuration tab:

For this item:	Specify:
Message Type	CIP Generic Read or CIP Generic Write
Source Tag	Tag containing the data to be transferred
Number of Elements	Number of array elements to transfer
Destination Tag	Tag to which the data will be transferred

4. On the Communication tab, specify the communication path.

   You must enter the communication path. If you want to send a MSG instruction to a 1734-OB3E module at node 10 in this bridging example, the communication path is:

For this item:	Specify:
Communication Path	1,3,1,2,2,10 where: 1 is the virtual backplane of the CompactLogix controller 3 is slot number of the Local 1769 Bus Adapter 1 is the 1769 backplane 2 is the slot number of the 1769-SDN module 2 is the DeviceNet network 10 is the node number of the 1734-OB2E

If you send messages via DeviceNet, either local or through a bridge, program the MSG instructions sequentially. The 1769-SDN has limited buffering capability for MSG instructions.

Notes:

Communicating with Devices on a Serial Link

Using This Chapter

Default Communication Configuration

The CompactLogix controller has the following default serial configuration:

Parameter:	Channel 0 Default:	Channel 1 Default (1769-L31 only):
Baud Rate	19.2K	19.2K
Parity	none	none
Station Address	0	0
Control Lines	no handshaking	no handshaking
Error Detection	BCC	BCC
Embedded Responses	auto detect	auto detect
Duplicate Packet (Message) Detect	enabled	enabled
ACK Timeout	50 counts	50 counts
NAK Receive Limit	3 retries	3 retries
ENQ Transmit Limit	3 retries	3 retries
Data Bits	8	8
Stop Bits	1	1
Protocol	DF1 full-duplex	DF1 full-duplex

TIP

Node Address is part of the default configuration. Changing the node address will result in the DCH0 LED turning off.

System protocol options

The serial port supports:

- DF1 full-duplex
- DF1 master
- DF1 slave
- DH-485
- ASCII (user mode) channel 0 only
- Modbus (user mode protocol) via ladder logic routine

Modbus support

To use Logix5000 controllers on Modbus, you connect through the serial port and execute a specific ladder logic routine. The ladder logic routine is available on the CD for RSLogix 5000 Enterprise programming software. For more information, see Using Logix5000 Controllers as Masters or Slaves on Modbus Application Solution, publication CIG-AP129A-EN-P.

Using the Channel 0 default communication push button

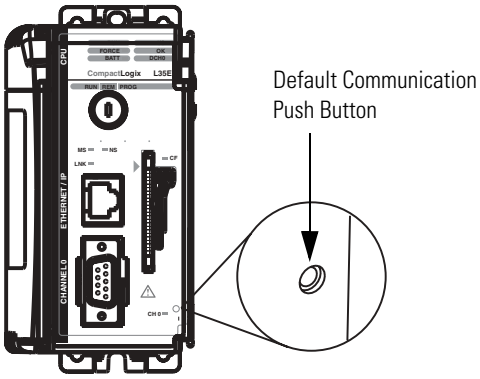
Use the Channel 0 Default Communication Push Button to change from the user-defined communication configuration to the default communications configuration. *Hold the button* until the Channel 0 Default Communications (DCH0) LED turns on (green, steady) showing that the default communication configuration is active.

TIP

Before pressing the Default Communication Push Button, be sure to note the present communication configuration. Pushing the Default Communication Push Button resets all configured parameters back to their default settings. To return the channel to its user-configured parameters, you must enter them manually while online with the controller or download them as part of a Logix Project file.

To accomplish this online, enter the Controller Properties screen under the Serial Port, System Protocol and User Protocol tabs.

The Default Communication Push Button is located on the front of the controller in the lower right corner.



Configuring Your System for a Serial Link

For the CompactLogix controller to operate on a serial network, you need:

- a workstation with a serial port
- RSLinx software to configure the serial communication driver
- RSLogix5000 programming software to configure the serial port of the controller

IMPORTANT Limit the length of serial (RS-232) cables to 15.2m (50 ft.).

ATTENTION

The CompactLogix controller is grounded through its DIN rail or mounting foot. It is important that you understand the workstation's grounding system before connecting it to the controller.

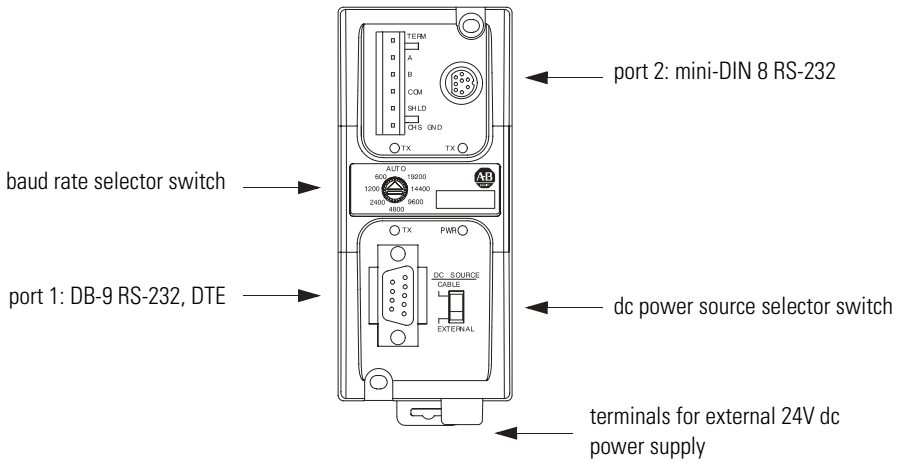
Step 1: Configure the hardware

Channel 0 on the CompactLogix controllers is fully isolated and does not need a separate isolation device. Channel 1 on the 1769-L31 is a non-isolated serial port.

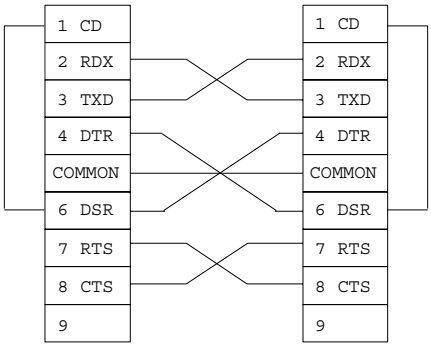
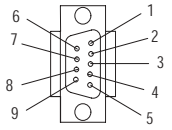
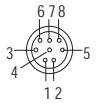
1. Determine whether you need an isolator

   If you connect channel 1 of the 1769-L31 controller to a modem or an ASCII device, consider installing an isolator between the controller and modem or ASCII device. An isolator is also recommended when connecting the controller directly to a programming workstation.

   One possible isolator is the 1761-NET-AIC interface converter.



2. Select the appropriate cable.

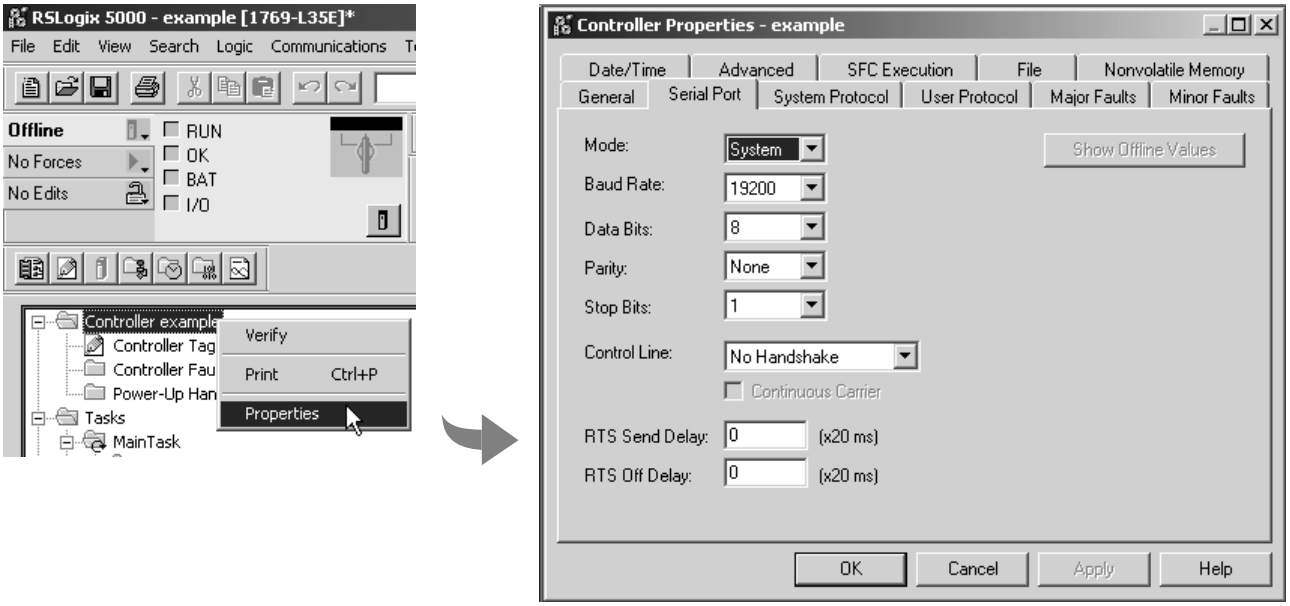
Are you using an isolator?	Use this cable:
no	The 1756-CP3 cable attaches the controller directly to the controller.  If you make your own cable, it must be shielded and the shields must be tied to the metal shell (that surrounds the pins) on both ends of the cable. You can also use a 1747-CP3 cable (from the SLC product family). This cable has a taller right-angle connector housing than the 1756-CP3 cable.
yes	The 1761-CBL-AP00 cable (right-angle connector to controller) or the 1761-CBL-PM02 cable (straight connector to the controller) attaches the controller to port 2 on the 1761-NET-AIC isolator. The mini-DIN connector is not commercially available, so you cannot make this cable.  DB-9 right-angle or straight cable end  8-pin, mini-DIN cable end **Pin:** / **DB-9 end:** / **Mini-DIN end:** 1 / DCD / DCD 2 / RxD / RxD 3 / TxD / TxD 4 / DTR / DTR 5 / ground / ground 6 / DSR / DSR 7 / RTS / RTS 8 / CTS / CTS 9 / na / na

3. Connect the appropriate cable to the serial port.

Step 2: Configure the serial port of the controller

1. In RSLogix 5000 programming software, select the Edit → Controller folder.

2. On the Serial Port tab, specify the appropriate serial communication configuration.



3. On the System Protocol tab, select the appropriate DF1 communication mode for point-to-point or master/slave communications. Or on the User Protocol tab, select ASCII to communicate with an ASCII device.

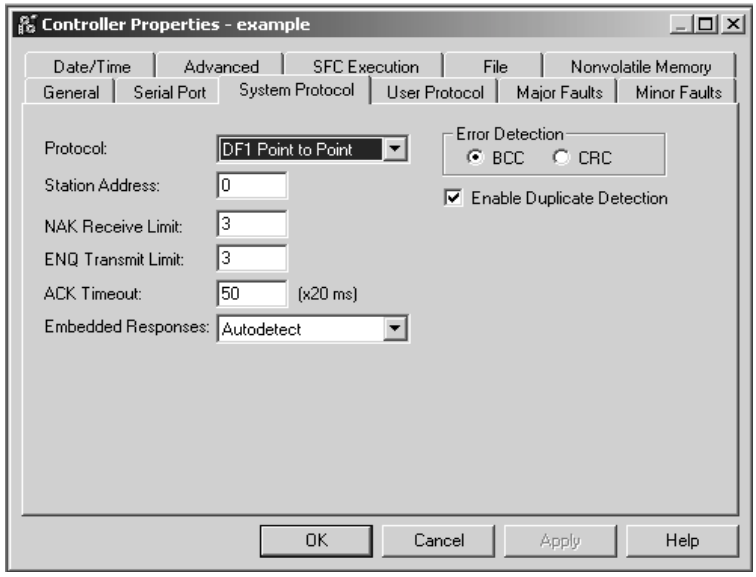
Specifying serial port characteristics

Specify these characteristics on the Serial Port tab (default values are shown in bold):

Characteristic:	Description (default is shown in bold):
Mode	Select **System** (for DF1 and DH485 communication) or User mode (for ASCII communication).
Baud rate	Specifies the communication rate for the serial port. Select a baud rate that all devices in your system support. Select 110, 300 600, 1200, 2400, 4800, 9600, **19200**, 38400 Kbps. Note: 38400 Kbps only in DF1 mode
Parity	Specifies the parity setting for the serial port. Parity provides additional message-packet error detection. Select **None** or Even.
Data bits	Specifies the number of bits per message packet. Select **8**.
Stop bits	Specifies the number of stop bits to the device with which the controller is communicating. Select **1** or 2.
Control line	Specifies the mode in which the serial driver operates. Select **No Handshake**, Full-Duplex, Half-Duplex with Continuous Carrier, or Half-Duplex without Continuous Carrier. If you are not using a modem, select No Handshake. If both modems in a point-to-point link are full-duplex, select Full-Duplex for both controllers. If the master modem is full-duplex and the slave modem is half-duplex, select Full-Duplex for the master controller and select Half-Duplex with Continuous Carrier for the slave controller. If all the modems in the system are half-duplex, select Half-Duplex without Continuous Carrier for the controller.
RTS send delay(1)	Enter a count that represents the number of 20 ms periods of time that elapse between the assertion of the RTS signal and the beginning of a message transmission. This time delay lets the modem prepare to transmit a message. The CTS signal must be high for the transmission to occur. The range is **0** to +32767 periods.
RTS off delay(1)	Enter a count that represents the number of 20 ms periods of time that elapse between the end of a message transmission and the de-assertion of the RTS signal. This time delay is a buffer to make sure the modem successfully transmits the entire message. The range is **0** to +32767 periods. Normally leave this setting at zero.

(1) This parameter is especially useful for communicating via radio modems.

Specifying system protocol characteristics

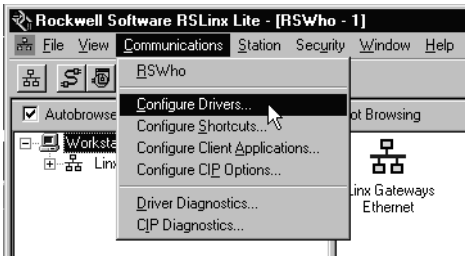


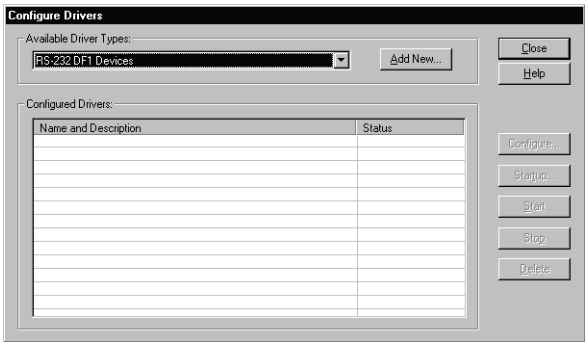
The available system modes are:

Use this mode:	For:	See page:
DF1 point-to-point	communication between the controller and one other DF1-protocol-compatible device. This is the default system mode. This mode is typically used to program the controller through its serial port.	5-10
DF1 master mode	control of polling and message transmission between the master and slave nodes. The master/slave network includes one controller configured as the master node and as many as 254 slave nodes. Link slave nodes using modems or line drivers. A master/slave network can have node numbers from 0 to 254. Each node must have a unique node address. Also, at least 2 nodes must exist to define your link as a network (1 master and 1 slave station are the two nodes).	5-13
DF1 slave mode	using a controller as a slave station in a master/slave serial communication network. When there are multiple slave stations on the network, link slave stations using modems or line drivers to the master. When you have a single slave station on the network, you do not need a modem to connect the slave station to the master. You can configure the control parameters for no handshaking. You can connect 2 to 255 nodes to a single link. In DF1 slave mode, a controller uses DF1 half-duplex protocol. One node is designated as the master and it controls who has access to the link. All the other nodes are slave stations and must wait for permission from the master before transmitting.	5-13
User mode (Channel 0 only)	communicating with ASCII devices. This requires your program logic to use the ASCII instructions to read and write data from and to an ASCII device.	5-15
DH-485	communicating with other DH-485 devices multi-master, token passing network allowing programming and peer-to-peer messaging.	6-1

Step 3: Configure the serial communication driver

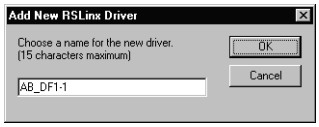
1. In RSLinx software, select Communication → Configure Driver. From the Available Driver Types list, select"RS-232 DF1 Devices".





Click Add New.

2. Specify a name for the driver.

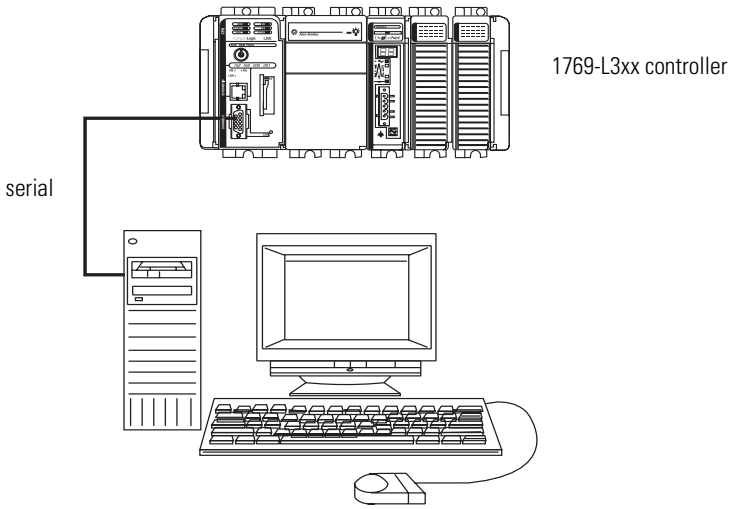


3. Specify the appropriate communication settings.
Select the "Logix/CompactLogix" and specify the COM port. Click Autoconfigure to have the software determine the remaining serial settings.

Click OK.

Example 1: Workstation Directly Connected to a CompactLogix Controller

In the following example, a workstation directly connects to a CompactLogix controller over a serial link. This is useful for downloading a controller project directly to the controller.



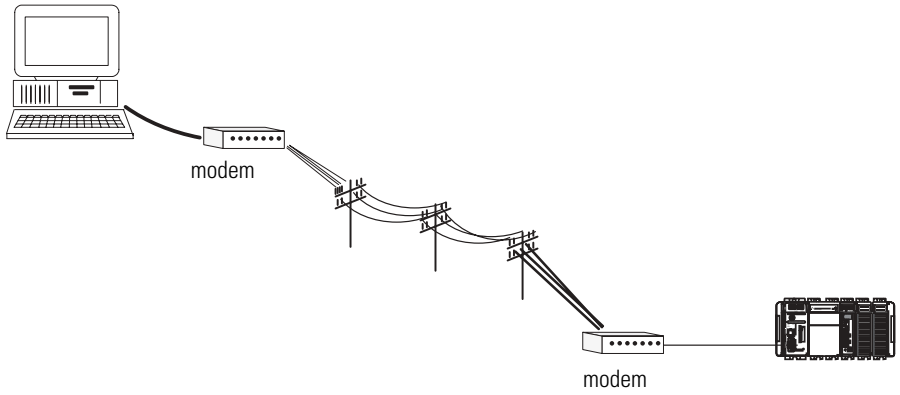
This type of protocol supports simultaneous transmission between two devices in both directions. The DF1 point-to-point protocol controls message flow, detects and signals errors, and retries if errors are detected.

Configuring a DF1 point-to-point station

This field:	Description:
Station address	The station address for the serial port on the DF1 point-to-point network. Enter a valid DF1 address (0 to 254). Address 255 is reserved for broadcast messages. The default is 0.
NAK receive limit	Specifies the number of NAKs the controller can receive in response to a message transmission. Enter a value 0 to 127. The default is 3.
ENQ transmit limit	Specifies the number of inquiries (ENQs) you want the controller to send after an ACK timeout. Enter a value 0 to 127. The default is 3.
ACK timeout	Specifies the amount of time you want the controller to wait for an acknowledgment to its message transmission. Enter a value 0 to 32767. Limits are defined in 20 ms intervals. The default is 50 (1000 ms).
Embedded response	Specifies how to enable embedded responses. Select Autodetect (enabled only after receiving one embedded response) or Enabled. The default is Autodetect.
Error detection	Select BCC or CRC error detection. Configure both stations to use the same type of error checking. **BCC:** the controller sends and accepts messages that end with a BCC byte for error checking. BCC is quicker and easier to implement in a computer driver. This is the default. **CRC:** the controller sends and accepts messages with a 2-byte CRC for error checking. CRC is a more complete method.
Enable duplicate detection	Select whether or not the controller should detect duplicate messages. The default is duplicate detection enabled.

Example 2: Workstation Remotely Connected to a CompactLogix Controller

In the following example, a workstation remotely connects to a CompactLogix controller over a serial link. A modem is connected to the controller to provide remote access.



If you use a modem to remotely connect the controller to one workstation, use DF1 point-to-point (full-duplex) protocol, as in the previous example.

Master/Slave communication methods

Half-duplex DF1 protocol

Half-duplex master/slave protocol is a SCADA protocol, consisting of 1 master and up to 254 slaves. Typically, the master polls all of the slaves for data in a round-robin fashion, using RF modems, leased-line modems, or any similar media.

A master station can communicate with a slave station in two ways:

Name:	This method:	Benefits:
standard communication mode	initiates polling packets to slave stations according to their position in the polling array(s). Polling packets are formed based on the contents of the normal poll array and the priority poll array.	This communication method is most often used for point-to-multipoint configurations. This method provides these capabilities: • slave stations can send messages to the master station (polled report-by-exception) • slave stations can send messages to each other via the master (slave-to-slave transfers) • master maintains an active station array The poll array resides in a user-designated data file. You can configure the master: • to send messages during its turn in the poll array *or* • for between-station polls (master transmits any message that it needs to send before polling the next slave station) In either case, configure the master to receive multiple messages or a single message per scan from each slave station.
message-based communication mode	initiates communication to slave stations using only user-programmed message (MSG) instructions. Each request for data from a slave station must be programmed via a MSG instruction. The master polls the slave station for a reply to the message after waiting a user-configured period of time. The waiting period gives the slave station time to formulate a reply and prepare the reply for transmission. After all of the messages in the master's message-out queue are transmitted, the slave-to-slave queue is checked for messages to send.	If your application uses satellite transmission or public switched-telephone-network transmission, consider choosing message-based communication. Communication to a slave station can be initiated on an as-needed basis. Also choose this method if you need to communicate with non-intelligent remote terminal units (RTUs).

Configuring a DF1 slave station

This field:	Description:
Station address	The station address for the serial port on the DF1 slave. Enter a valid DF1 address (0 to 254). Address 255 is reserved for broadcast messages. The default is 0.
Transmit retries	The number of times the remote station retries a message after the first attempt before the station declares the message undeliverable. Enter a value 0 to 127. The default is 3.
Slave poll timeout	Specifies the amount of time the slave station waits to be polled by a master before indicating a fault. Enter a value 0 to 32767. Limits are defined in 20 ms intervals. The default is 3000 (60,000 ms).
EOT suppression	Select whether or not to suppress sending EOT packets in response to a poll. The default is *not* to suppress sending EOT packets.
Error detection	Select BCC or CRC error detection. Configure both stations to use the same type of error checking. **BCC:** the controller sends and accepts messages that end with a BCC byte for error checking. BCC is quicker and easier to implement in a computer driver. This is the default. **CRC:** the controller sends and accepts messages with a 2-byte CRC for error checking. CRC is a more complete method.
Enable duplicate detection	Select whether or not the controller should detect duplicate messages. The default is duplicate detection enabled.

Configuring a DF1 master station

This field:	Description:
Station address	The station address for the serial port on the DF1 master. Enter a valid DF1 address (0 to 254). Address 255 is reserved for broadcast messages. The default is 0.
Transmit retries	Specifies the number of times a message is retried after the first attempt before being declared undeliverable. Enter a value 0 to 127. The default is 3.
ACK timeout	Specifies the amount of time you want the controller to wait for an acknowledgment to its message transmission. Enter a value 0 to 32767. Limits are defined in 20ms intervals. The default is 50 (1000 ms).
Reply message wait	**Message-based polling mode only** Specifies the amount of time the master station waits after receiving an ACK to a master-initiated message before polling the slave station for a reply. Enter a value 0 to 65535. Limits are defined in 20ms intervals. The default is 5 (100 ms).
Polling mode	Select one of these: • Message Based (slave cannot initiate messages) • Message Based (slave can initiate messages) - default • Standard (multiple message transfer per node scan) • Standard (single message transfer per node scan)
Master transmit	**Standard polling modes only** Select when the master station sends messages: • between station polls (default) • in polling sequence

This field:	Description:
Normal poll node tag	**Standard polling modes only** An integer tag array that contains the station addresses of the slave stations. Create a single-dimension array of data type INT that is large enough to hold all the normal station addresses. The minimum size is three elements. This tag must be controller-scoped. The format is: *list[0]* contains total number of stations to poll *list[1]* contains address of station currently being polled *list[2]* contains address of first slave station to poll *list[3]* contains address of second slave station to poll *list[n]* contains address of last slave station to poll
Normal poll group size	**Standard polling modes only** The number of stations the master station polls after polling all the stations in the priority poll array. Enter 0 (default) to poll the entire array.
Priority poll node tag	**Standard polling modes only** An integer tag array that contains the station addresses of the slave stations you need to poll more frequently. Create a single-dimension array of data type INT that is large enough to hold all the priority station addresses. The minimum size is three elements. This tag must be controller-scoped. The format is: *list[0]* contains total number of stations to be polled *list[1]* contains address of station currently being polled *list[2]* contains address of first slave station to poll *list[3]* contains address of second slave station to poll *list[n]* contains address of last slave station to poll
Active station tag	**Standard polling modes only** An array that stores a flag for each of the active stations on the DF1 link. Both the normal poll array and the priority poll array can have active and inactive stations. A station becomes inactive when it does not respond to the master's poll. Create a single-dimension array of data type SINT that has 32 elements (256 bits). This tag must be controller-scoped.
Error detection	Select BCC or CRC error detection. Configure both stations to use the same type of error checking. **BCC:** the controller sends and accepts messages that end with a BCC byte for error checking. BCC is quicker and easier to implement in a computer driver. This is the default. **CRC:** the controller sends and accepts messages with a 2-byte CRC for error checking. CRC is a more complete method.
Enable duplicate detection	Select whether or not the controller should detect duplicate messages. The default is duplicate detection enabled.

If You Choose One of the Standard Polling Modes

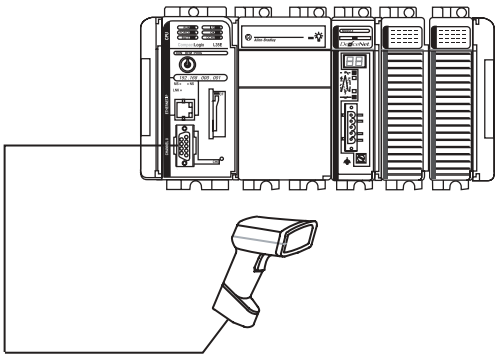
The master station polls the slave stations in this order:

1. all stations that are active in the priority poll array
2. one station that is inactive in the priority poll array
3. the specified number (normal poll group size) of active stations in the normal poll array
4. one inactive station, after all the active stations in the normal poll array have been polled

Use the programming software to change the display style of the active station array to binary so you can view which stations are active.

Example 3: CompactLogix Controller Connected to a Bar Code Reader

In the following example, a workstation connects to a bar code reader. Channel 0 of the CompactLogix controllers supports ASCII. A bar code reader is an ASCII device, so you configure the serial port differently than in the previous examples. Configure the serial port for User mode, rather than the system mode.

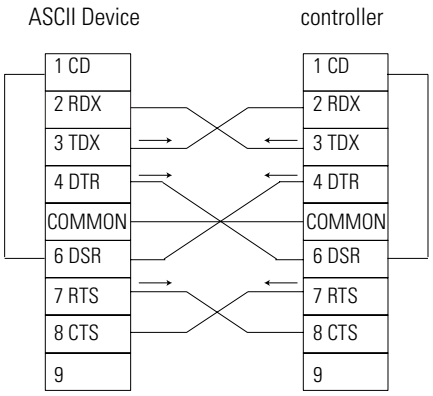
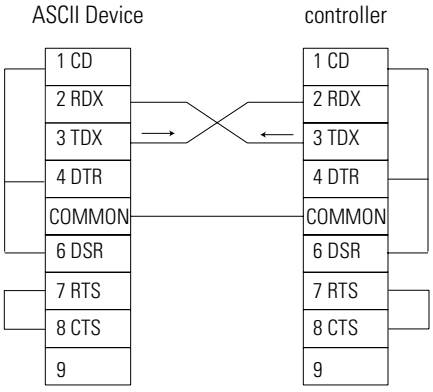


Connect the ASCII device to the controller

To connect the ASCII device to the Channel 0 serial port of the controller:

1. For the serial port of the ASCII device, determine which pins send signals and which pins receive signals.

2. Connect the sending pins to the corresponding receiving pins and attach jumpers:

If the communications hardware handshaking is:	Then wire the connectors as follows:
enabled	
disabled	

3. Attach the cable shield to both connectors and tie the cable to both connectors.

4. Connect the cable to the controller and the ASCII device.

The following table lists the default serial port configuration settings for the ASCII protocol. You specify these settings on the User Protocol tab under Controller Properties.

Configuring User mode

This field:	Description:
Buffer size	Specify the maximum size (in bytes) of the data array you plan to send and receive. The default is 82 bytes.
Termination characters	Specify the characters you will use to designate the end of a line. The default characters are '$r' and '$FF'.
Append characters	Specify the characters you will append to the end of a line. The default characters are '$r' and '$l'.(1)
XON/XOFF	Select whether or not to regulate the flow of incoming data. The default is disabled.
Echo mode	Select whether or not to echo data back to the device from which it was sent. The default is disabled.
Delete mode	Select Ignore, CTR, or Printer for the delete mode. The default is Ignore.

(1) IEC 1131-3 representation for carriage return and line feed.

Programming ASCII instructions

ASCII instructions are used to communicate with ASCII devices that you connect to channel 0. Your RSLogix5000 programming software CDROM includes programming examples using ASCII instructions.

For information about using these examples, see the *Logix5000 Controllers General Instruction Set Reference Manual*, publication 1756-RM003.

Example 4: Bridging through the Serial Port

You can use the controller to bridge between networks; the controller supports one connected and one unconnected message between devices. The controller will only bridge messaging data (not I/O data), and there is limited buffering to store waiting messages that bridge networks.

You can bridge from serial to Ethernet or from serial to DeviceNet.

IMPORTANT The update time of local I/O modules may increase when the controller is bridging messages.

Bridging over the CompactLogix controller should be targeted toward applications that are not real time dependent, such as RSLogix 5000 program downloads and ControlFlash updates.

IMPORTANT In the 1769-L31 controller, you cannot bridge from one serial port to the other serial port.

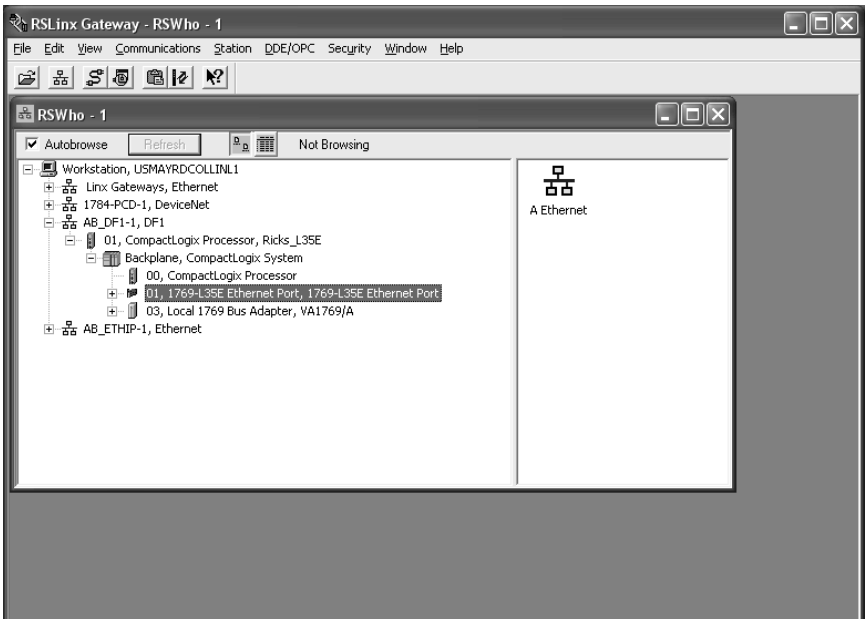
For example, you can use RSLogix 5000 software via a serial-to-Ethernet bridge to set the IP address of the EtherNet/IP port of the controller.

1. Make sure the controller is installed and running.
2. Connect to the controller via the serial connection.
3. Start RSLinx software. The RSWho window opens.

4. Navigate from the RSWho window to the EtherNet/IP port of the CompactLogix controller.

Starting with the serial driver (AB_DF1-1 in this example), you can locate the CompactLogix controller. From there, expand the backplane of the CompactLogix system and you can see the EthetNet/IP port.

Right-click on the Ethernet port (not the controller) and select Module Configuration



Notes:

Communicating with Devices on a DH-485 Link

Using This Chapter

When using a CompactLogix controller it is recommended that you use NetLinx networks (EtherNet/IP, ControlNet, or DeviceNet) because excessive traffic on a DH-485 network may make it impractical to connect to a CompactLogix controller with RSLogix 5000 programming software. CompactLogix processors fully support the DH-485 protocol, but using the recommended NetLinx networks is more practical.

The DH-485 protocol uses RS-485 half-duplex as its physical interface. (RS-485 is a definition of electrical characteristics; it is *not* a protocol.) You can configure the RS-232 port of the CompactLogix controller to act as a DH-485 interface. By using a 1761-NET-AIC and the appropriate RS232 cable (1756-CP3 or 1747-CP3), a CompactLogix controller can send and receive data on a DH-485 network.

IMPORTANT A DH-485 network consists of multiple cable segments. Limit the total length of all the segments to 1219m (4000 ft.).

Configuring Your System for a DH-485 Link

For the CompactLogix controller to operate on a DH-485 network, you need:

- a 1761-NET-AIC interface converter for each CompactLogix controller you want to put on the DH-485 network.

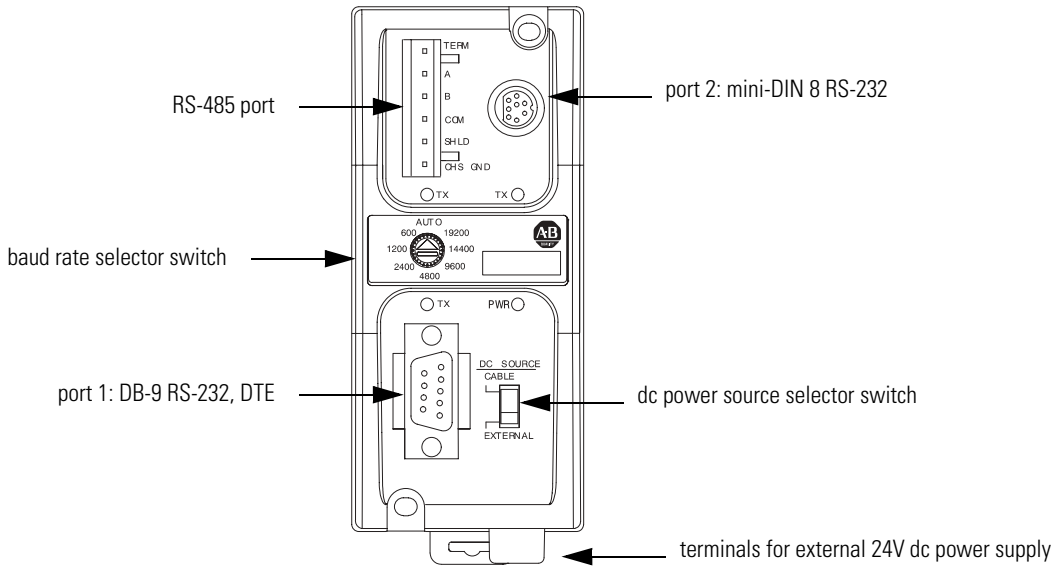
  You can have two controllers per one 1761-NET-AIC converter, but you need a different cable for each controller. Connect one controller to port 1 (9-pin connector) and one controller to port 2 (mini-DIN connector).

- RSLogix 5000 programming software to configure the serial port of the controller for DH-485 communications.

When attempting to go online or upload/download a program using the Communications/Who Active window in RSLogix 5000 software, disable the Autobrowse feature to minimize traffic from RSLogix 5000 software on the DH-485 network.

Step 1: Configure the hardware

The RS-232 port is built-in to the front of the CompactLogix controller. The 1769-L31 controller has two serial ports. Connect the serial port to an RS-232-to-RS-485 interface converter. One possible converter is the 1761-NET-AIC interface converter.



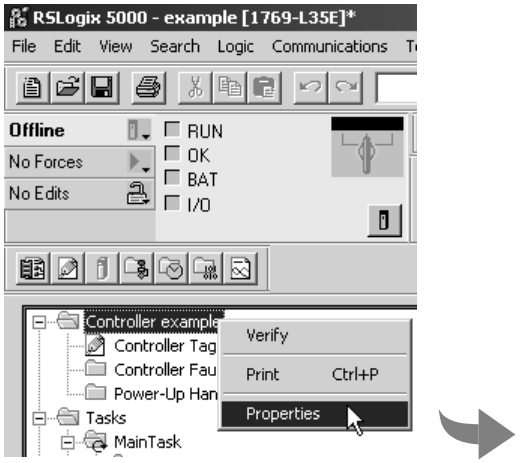
Connect the serial port of the CompactLogix controller to either port 1 or port 2 of the 1761-NET-AIC converter. Use the RS-485 port to connect the converter to the DH-485 network.

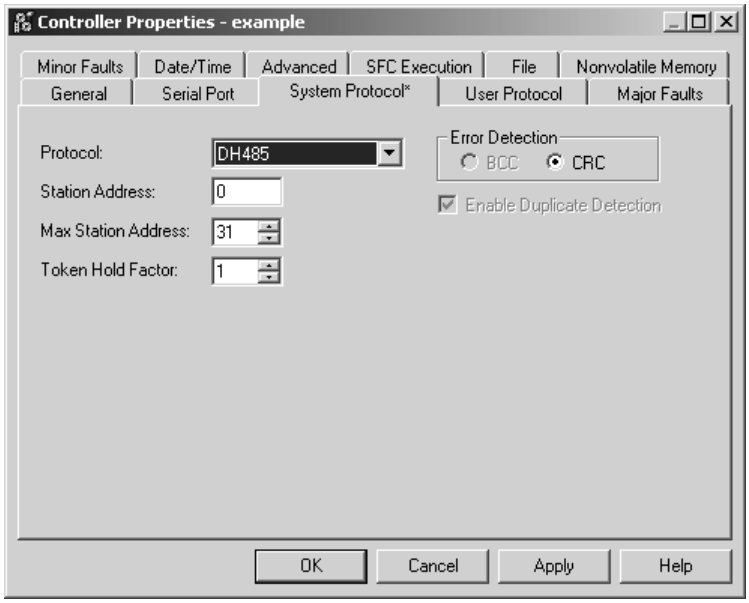
The cable you use to connect the controller depends on the port you use on the 1761-NET-AIC converter.

If you connect to this port:	Use this cable:
port 1 DB-9 RS-232, DTE connection	1747-CP3 or 1761-CBL-AC00
port 2 mini-DIN 8 RS-232 connection	1761-CBL-AP00 or 1761-CBL-PM02

Step 2: Configure the DH-485 port of the controller

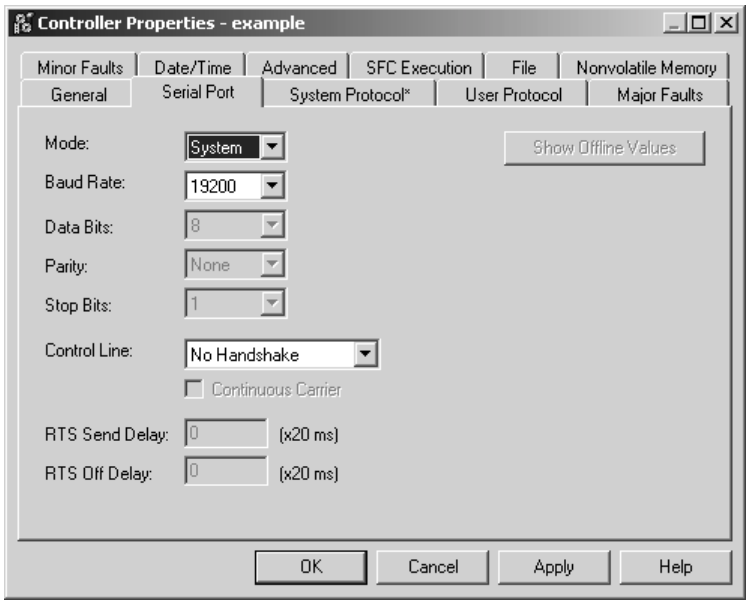
1. In RSLogix 5000 programming software, select the Controller folder. Right-click to select Properties.
2. On the System Protocol tab, specify the appropriate serial communication configuration.





3. On the Serial Port tab, specify the appropriate communication settings.

The grayed out settings are selections that do not apply to a DH-485 network.



Specify these characteristics on the Serial Port tab (default values are shown in bold):

Characteristic:	Description (default is shown in bold):
Baud Rate	Specifies the communication rate for the DH-485 port. All devices on the same DH-485 network must be configured for the same baud rate. Select 9600 or **19200** Kbps.
Node Address	Specifies the node address of the CompactLogix controller on the DH-485 network. Select a number **1**-31 decimal, inclusive. To optimize network performance, assign node addresses in sequential order. Initiators, such as personal computers, should be assigned the lowest address numbers to minimize the time required to initialize the network.
Token Hold Factor	Specifies the number of messages sent per token possession. Select a number **1**-4, inclusive.
Maximum Node Address	Specifies the maximum node address of all the devices on the DH-485 network. Select a number 1-**31** decimal, inclusive. To optimize network performance, make sure: • the maximum node address is the highest node number being used on the network • that all the devices on the same DH-485 network have the same selection for the maximum node address.

Planning a DH-485 Network

The DH-485 network offers:

- interconnection of 32 devices
- multi-master capability
- token passing access control
- the ability to add or remove nodes without disrupting the network
- maximum network length of 1219 m (4000 ft.)

The DH-485 protocol supports two classes of devices: initiators and responders. All initiators on the network get a chance to initiate message transfers. The DH-485 protocol uses a token-pass algorithm to determine which initiator has the right to transmit.

DH-485 Token Rotation

A node holding the token can send any valid packet onto the network. As a default, each node gets only one transmission (plus two retries) each time it receives the token. After a node sends one message packet, it attempts to give the token to its successor by sending a “token pass” packet to its successor.

If no network activity occurs, the initiator sends the token pass packet again. After two retries (a total of three tries) the initiator attempts to find a new successor.

IMPORTANT The maximum address that the initiator searches for before starting again with zero is the value in the configurable parameter "maximum node address." The default and maximum value for this parameter is 31 for all initiators and responders.

The allowable range of the node address of a initiator is 0 to 31. The allowable address range for all responders is 1 to 31. There must be at least one initiator on the network.

Network initialization

The network requires at least one initiator to initialize it. Network initialization begins when a initiator on the network detects a period of inactivity that exceeds the time of a link dead timeout. When the link dead timeout is exceeded, usually the initiator with the lowest address claims the token. When a initiator has the token it will begin to build the network.

Building a network begins when the initiator that claimed the token tries to pass the token to the successor node. If the attempt to pass the token fails, or if the initiator has no established successor (for example, when it powers up), it begins a linear search for a successor starting with the node above it in the addressing.

When the initiator finds another active node, it passes the token to that node, which repeats the process until the token is passed all the way around the network to the initial node. At this point, the network is in a state of normal operation.

Number of Nodes and Node Addresses

The number of nodes on the network directly affects the data transfer time between nodes. Unnecessary nodes (such as a second programming terminal that is not being used) slow the data transfer rate. The maximum number of nodes on the network is 32.

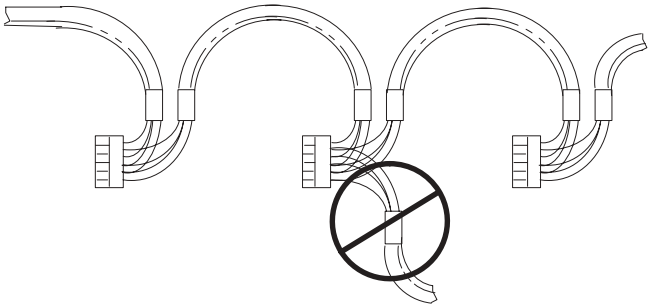
If the node addresses for controllers are assigned in sequence, starting at node 1 (with node 0 left for a programming terminal), it is as efficient to leave the maximum node address at 31 as it is to decrease it to the highest node address on the network. Then, adding devices to the network at a later time will not require modifying the maximum node address in every device on the network. The maximum node address should be the same for all devices on a DH-485 network for optimal operation.

The best network performance occurs when node addresses start at 0 and are assigned in sequential order. The controller defaults to node address 1. Initiators, such as personal computers, should be assigned the lowest numbered addresses to minimize the time required to initialize the network.

Installing a DH-485 Network

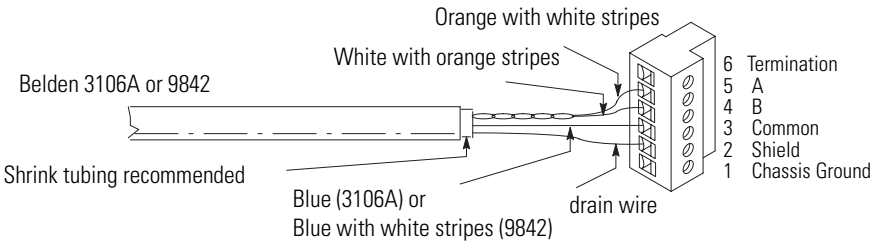
A DH-485 network consists of a number of cable segments daisy-chained together. The total length of the cable segments cannot exceed 1219 m (4000 ft).

IMPORTANT Use shielded, twisted-pair cable, either Belden 3106A or Belden 9842. A daisy-chained network is recommended.

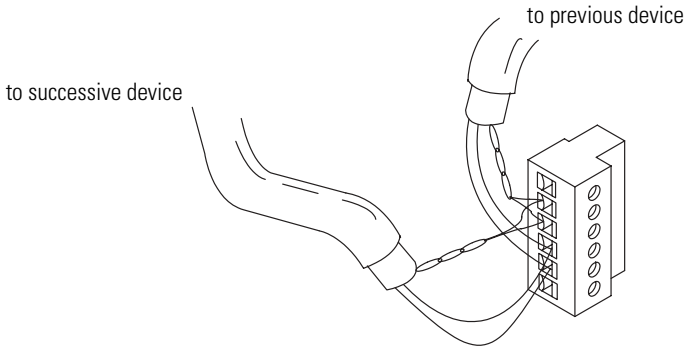


When cutting cable segments, make them long enough to route them from one link coupler to the next with sufficient slack to prevent strain on the connector. Allow enough extra cable to prevent chafing and kinking in the cable.

Single cable connection



Multiple cable connection



The table below shows wire/terminal connections for Belden 3106A.

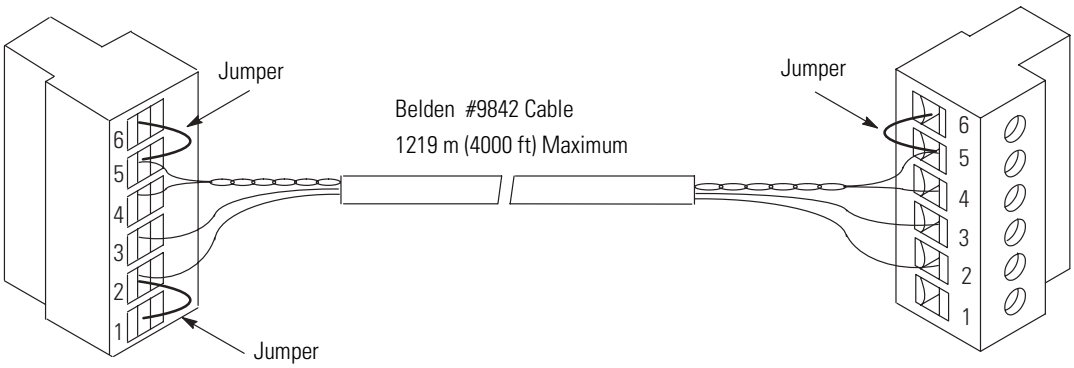
For this Wire/Pair	Connect this Wire	To this Terminal
shield/drain	non-jacketed	2 - Shield
blue	blue	3 - (Common)
white/orange	white with orange stripe	4 - (Data B)
	orange with white stripe	5 - (Data A)

The table below shows wire/terminal connections for Belden 9842.

For this Wire/Pair	Connect this Wire	To this Terminal
shield/drain	non-jacketed	2 - Shield
blue/white	white with blue stripe	cut back - no connection(1)
	blue with white stripe	3 - (Common)
white/orange	white with orange stripe	4 - (Data B)
	orange with white stripe	5 - (Data A)

(1) To prevent confusion when installing the communication cable, cut back the white with blue stripe wire immediately after the insulation jacket is removed. This wire is not used by DH-485.

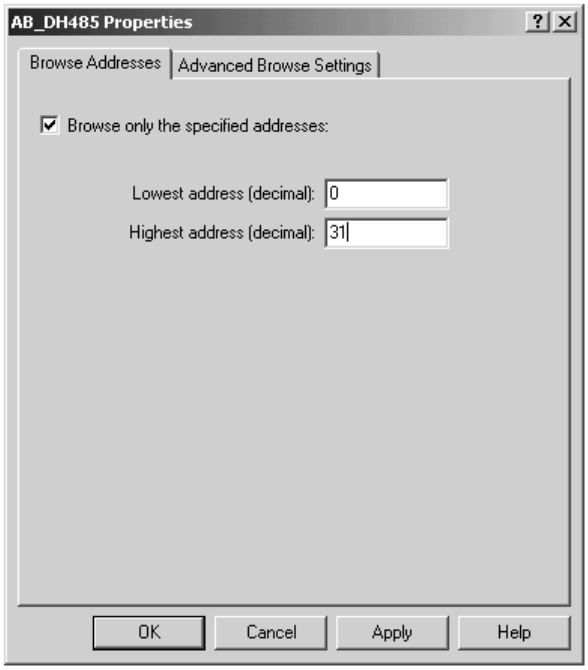
Grounding and terminating a DH-485 network



Browsing a DH-485 Network Remotely

To improve performance when browsing a DH-485 network, configure the DH-485 network properties in RSLinx software to display only those nodes that actually exist on the network.

1. In RSLinx software, right-click on the DH-485 network you plan to browse and select Properties.
2. On the Browse Addresses tab, specify the lowest and highest addresses that exist on the DH-485 network.



If you do not specify a specific range of addresses on the DH-485 network, the RSWho function in RSLinx software attempts to locate a device at every node address. Trying to locate devices that do not exist adds considerable time to displaying the RSWho window for the nework.

CompactLogix System Specifications

Using This Appendix

1769-L32E, 1769-L35E Controller Specifications

Description	1769-L32E	1769-L35E
Communication Ports	**CH0 - RS-232** RS-232, fully isolated DF1, DH-485, ASCII 38.4 Kbits/s maxmium **CH1 - EtherNet/IP** RJ-45 or 100BaseT EtherNet/IP 10/100 Mbytes/sec	
User Memory	750 Kbytes	1.5 Mbytes
Nonvolatile Memory	1784-CF64 CompactFlash	
Maximum Number of I/O Modules	16 I/O modules	30 I/O modules
Maximum Number of I/O Banks	3 banks	3 banks
Backplane Current	660 mA at 5V dc 90 mA at 24V dc	660 mA at 5V dc 90 mA at 24V dc
Maximum Power Dissipation	4.74 W	4.74 W
Power Supply Distance Rating	4 (The controller must be within four slot positions of the power supply.)	
Battery	1769-BA	
Weight	0.32 kg (0.70 lb.)	0.32 kg (0.70 lb.)
Programming Cable	1747-CP3 or 1756-CP3	
Panel Mounting Screw Torque (using M4 or #8 screws)	10 - 16 in-lb (1.1 - 1.8 Nm)	
Isolation Voltage	30V Tested to withstand 710 dc volts for 60 seconds	

1769-L31 Controller Specifications

Description	1769-L31	
Communication Ports	**CH0 - RS-232** DF1, DH-485, ASCII fully isolated 38.4 Kbits/s maxmium	**CH1 - RS-232** DF1, DH-485 non-isolated 38.4 Kbits/s maxmium
User Memory	512 Kbytes	
Nonvolatile Memory	1784-CF64 CompactFlash	
Maximum Number of I/O Modules	16 I/O modules	
Maximum Number of I/O Banks	3 banks	
Backplane Current	330 mA at 5V dc 40 mA at 24V dc	
Maximum Power Dissipation	2.61 W	
Power Supply Distance Rating	4 (The controller must be within four slot positions of the power supply.)	
Battery	1769-BA	
Weight	0.30 kg (0.66 lb.)	
Programming Cable	1747-CP3 or 1756-CP3	
Panel Mounting Screw Torque (using M4 or #8 screws)	10 - 16 in-lb (1.1 - 1.8 Nm)	
Isolation Voltage	30V Tested to withstand 710 dc volts for 60 seconds	

1769-L31, 1769-L32E, 1769-L35E Environmental Specifications

Description	1769-L31
Operating Temperature	0° to +60°C (+32° to +140°F)
Storage Temperature	-40° to +85°C (-40° to +185°F)
Relative Humidity	5% to 95% non-condensing
Vibration	Operating: 5G @ 10-500Hz
Shock DIN mount Panel mount	 Operating: 20G Non-operating: 30G Operating: 30G Non-operating: 40G
Emissions	CISPA11: Group 1, Class A
Electrical /EMC:	*The unit has passed testing at the following levels:*
ESD Immunity (IEC61000-4-2)	4 kV contact discharges, 8 kV air discharges
Radiated RF Immunity (IEC61000-4-3)	• 10V/M with 1kHz sine-wave 80%AM from 30MHz to 2000MHz • 10V/m with 200Hz 50% Pulse 100%AM at 900MHz
Surge Transient Immunity (IEC61000-4-5)	±1 kV line-line (DM) and ±2 kV line-earth (CM) on signal ports
Conducted RF Immunity (IEC61000-4-6)	10Vrms with 1kHz sine-wave 80%AM from 150kHz to 80MHz

1769-L31, 1769-L32E, 1769-L35E Certifications

Certification:	Description:
c-UL-us	UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada
CE(1)	European Union 89/336/EEC EMC Directive, compliant with: • EN 50082-2; Industrial Immunity • EN 61326; Meas./Control/Lab., Industrial Requirements • EN 61000-6-2; Industrial Immunity • EN 61000-6-4; Industrial Emissions
C-Tick(1)	Australian Radiocommunications Act, compliant with: • AS/NZS CISPR 11; Industrial Emissions
EtherNet/IP (1769-L32E, -L35E only)	ODVA conformance tested to EtherNet/IP specifications

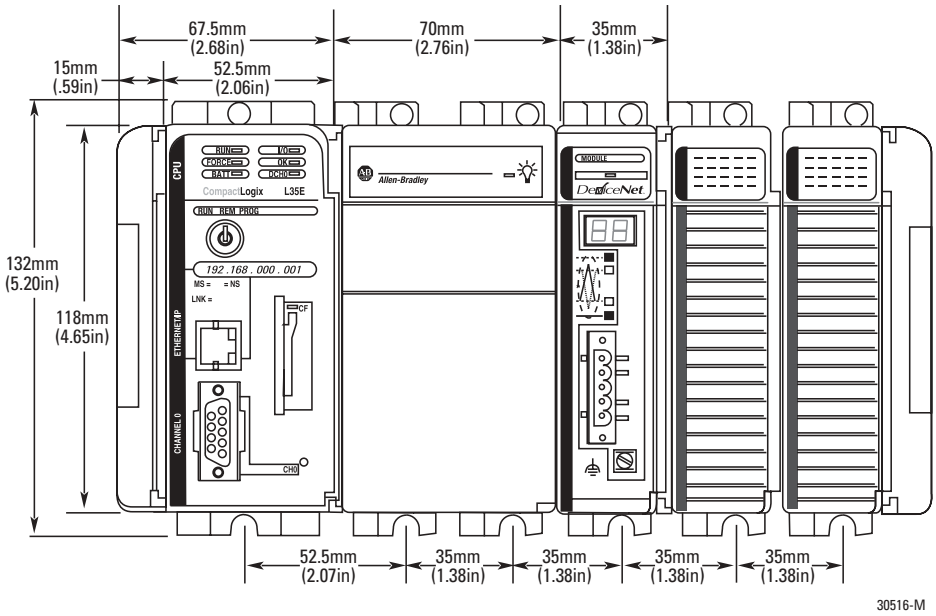
(1) See the Product Certification link at www.ab.com for Declarations of Conformity, Certificates, and other certification details.

Real-Time Clock Accuracy

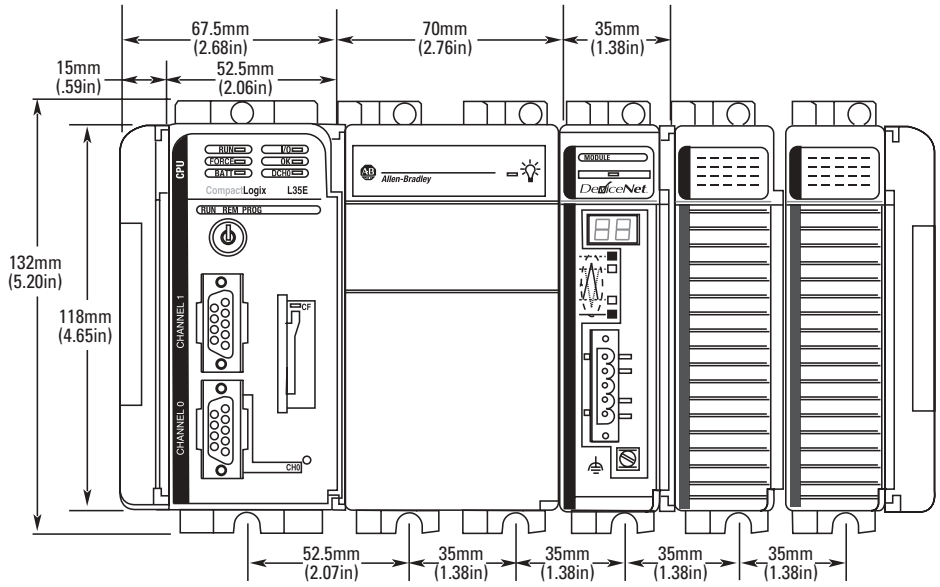
Ambient °C:	Accuracy:
0°C	+54 to -56 seconds/month
+25°C	+9 to -124 seconds/month
+40°C	-84 to -234 seconds/month
+55°C	-228 to -394 seconds/month
+60°C	-287 to -459 seconds/month

Dimensions

1769-L32E, 1769-L35E controller



1769-L31 controller



Controller LEDs

Indicator:	Color:	Description:
RUN	off	The controller is in Program or Test mode.
	solid green	The controller is in Run mode.
FORCE	off	No tags contain I/O force values. I/O forces are inactive (disabled).
	solid amber	I/O forces are active (enabled). I/O force values may or may not exist.
	flashing amber	One or more input or output addresses have been forced to an On or Off state, but the forces have not been enabled.
BAT	off	The battery supports memory.
	solid red	Either the battery is: • not installed. • 95% discharged and should be replaced.
I/O	off	Either: • There are *no* devices in the I/O configuration of the controller. • The controller does *not* contain a project (controller memory is empty).
	solid green	The controller is communicating with all the devices in its I/O configuration.
	flashing green	One or more devices in the I/O configuration of the controller are *not* responding.
	flashing red	The controller is not communicating to any devices. The controller is faulted.
OK	off	No power is applied.
	flashing red	**If the controller is:** / **Then:** a new controller — the controller requires a firmware update not a new controller — A major fault occurred. To clear the fault, either: - Turn the keyswitch from PROG to RUN to PROG - Go online with RSLogix 5000 software
	solid red	The controller detected a non-recoverable fault, so it cleared the project from memory. To recover: 1. Cycle power to the chassis. 2. Download the project. 3. Change to Run mode. If the OK LED remains solid red, contact your Rockwell Automation representative or local distributor.
	solid green	Controller is OK.
	flashing green	The controller is storing or loading a project to or from nonvolatile memory.

CompactFlash card LED

ATTENTION

Do not remove the CompactFlash card while the controller is reading from or writing to the card, as indicated by a flashing green CF LED. This could corrupt the data on the card or in the controller, as well as corrupt the latest firmware in the controller.

Indicator:	Color:	Description:
CF	off	No activity.
	flashing green	The controller is reading from or writing to the CompactFlash card.
	flashing red	CompactFlash card does not have a valid file system.

RS-232 Serial Port LEDs

Indicator:	Color:	Description:
DCH0	off	Channel 0 is configured differently than the default serial configuration.
	solid green	Channel 0 has the default serial configuration.
CH0	off	No RS-232 activity.
	flashing green	RS-232 activity.
CH1 (1769-L31 only)	off	No RS-232 activity.
	flashing green	RS-232 activity.

EtherNet/IP LEDs

These LEDS are only on 1769-L32E and 1769-L35E controllers.

Module Status (MS) indicator

Condition:	Status:	Indicates:	Recommended Action:
off	no power	The controller does not have power.	Check the controller power supply.
flashing green	standby	The port does not have an IP address and is operating in BOOTP mode.	Verify that the BOOTP server is running.
solid green	OK	The port is operating correctly.	Normal operation. No action required.
solid red	held in reset	The controller is holding the port in reset or the controller is faulted.	Clear the controller fault. Replace the controller.
	self-test	The port is performing its power-up self-test.	Normal operation during power-up. No action required.
	major fault	An unrecoverable fault has occurred.	Cycle power to the controller. Replace the controller.
flashing red	updating firmware	The port firmware is being updated.	Normal operation during firmware update. No action required.

Network Status (NS) indicator

Condition:	Status:	Indicates:	Recommended Action:
off	not initialized	The port does not have an IP address and is operating in BOOTP mode.	Verify that the BOOTP server is running.
flashing green	no CIP connections established	The port has an IP address, but no CIP connections are established.	Normal operation if no connections are configured. No action required. If connections are configured, check connection originator for connection error code.
solid green	CIP connections established	The port has an IP address and CIP connections (Class 1 or Class 3) are established.	Normal operation. No action required.
solid red	duplicate IP address	The port has detected that the assigned IP address is already in use.	Verify that all IP addresses are unique.
flashing red/green	self-test	The port is performing its power-up self-test.	Normal operation during powerup.

Link Status (LNK) indicator

Condition:	Status:	Indicates:	Recommended Action:
off	no link	The port is not connected to a powered Ethernet device. The port cannot communicate on Ethernet.	Verify that all Ethernet cables are connected. Verify that Ethernet switch is powered.
flashing green	self-test	The port is performing its power-up self-test.	Normal operation during powerup.
	data transmission and reception	The port is communicating on Ethernet.	Normal operation. No action required.
solid green	link OK	The port is connected to a powered Ethernet device. The port can communicate on Ethernet.	Normal operation. No action required.

Battery Life

Time ON/OFF	at 25°C (77°F)	at 40°C (104°F)	at 60°C (140°F)
Always OFF	14 months	12 months	9 months
ON 8 hours per day 5 days per week	18 months	15 months	12 months
ON 16 hours per day 5 days per week	26 months	22 months	16 months
Always ON	There is almost no drain on the battery when the controller is always ON.		

Battery duration after the LED turns on

The battery indicator (BAT) warns you when the battery is low. These durations are the amounts of time the battery will retain controller memory from the time the controller is powered down after the LED first turns on.

Temperature	Duration
60°C	8 days
25°C	25 days

Notes:

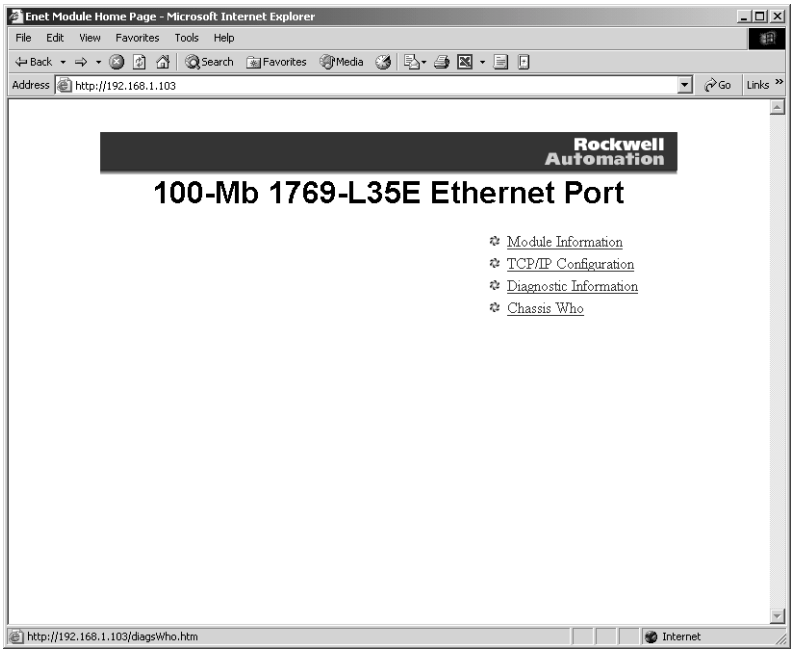
EtherNet/IP Diagnostics

Using This Appendix

The 1769-L32E and 1769-L35E controllers support web-based diagnostics.

For information about:	See page
Module information	B-2
TCP/IP configuration	B-2
Diagnostic information	B-3

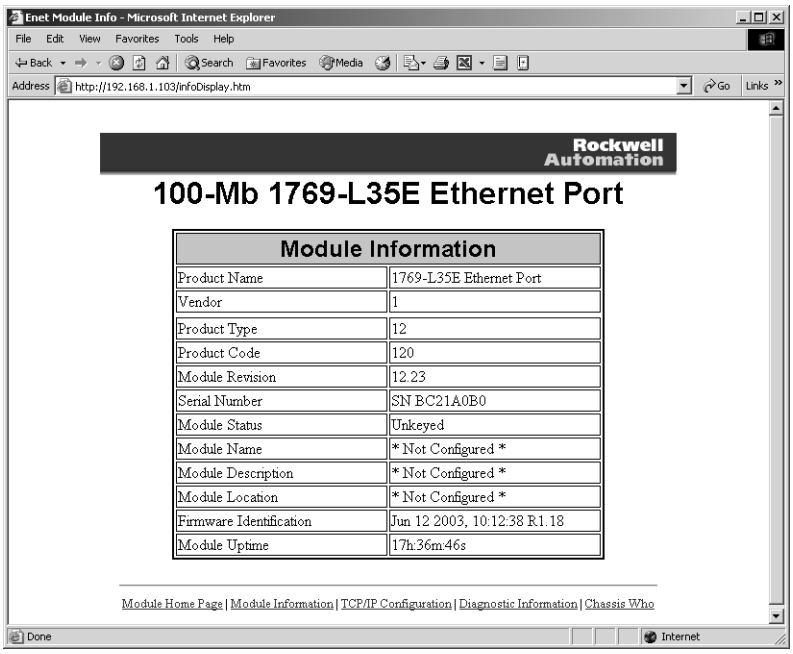
The EtherNet/IP controllers support web-based diagnostic pages that offer both internal and network diagnostics. To view the main web page, type the controller's IP address in your browser's address field.



From the main page, select links to display specific diagnostic information.

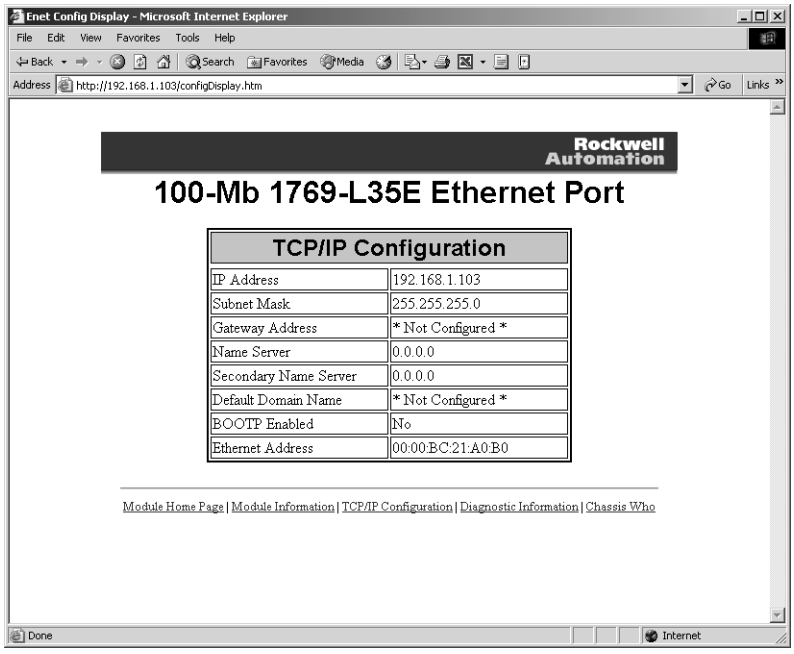
Module Information

Use the Module Information to display identification information about the controller.



TCP/IP Configuration

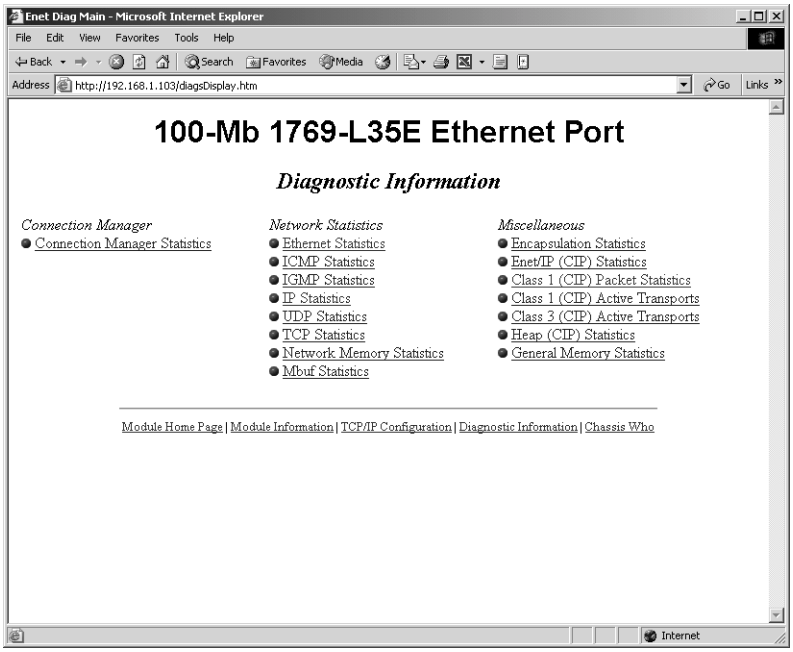
Use the TCP/IP Configuration page to display the current TCP/IP configuration settings for the controller.



Diagnostic Information

Use the Diagnostic Information page to display diagnostic information about:

- Class 1 connections - The most time critical connections, including I/O and produce/consume connections.
- Class 3 connections - The less time critical connections, such as those used for MMI and PLC programming or PLC to PLC messaging.



In the Miscellaneous section, you can get access to:

- Encapsulation statistics - General information about TCP connections, such as active incoming or outgoing connections and the total limit of TCP connections that can be made to the device.
- Class 1 (CIP) packet statistics - Information about the speed, duplex and user datagram (UDP) packet rates of CIP connections.
- Class 1 (CIP) transports - Specific information about any Class 1 (CIP) connections made to the device.
- Class 3 (CIP) transports - Specific information about any Class 3 (CIP) connections made to the device.

Encapsulation statistics

The Encapsulation Statistics offer general information about TCP connections coming into and going out of the device.

Field:	Definition:
Cumulative Encap (TCP) Connections	The total number of incoming and outgoing TCP module connections since powering up.
Active Total Encap (TCP) Connections	The number of incoming and outgoing TCP module connections currently active.
Total Encap (TCP) Connection Limit	The maximum number (64) of incoming or outgoing TCP connections the module can make at any single moment in time.
Active Incoming Encap (TCP) Connections	The number of TCP module connections coming in from the Ethernet media currently active.
Incoming Encap (TCP) Connection Limit	The maximum number (64) of incoming TCP connections the module can make at any single moment in time.
Active Outgoing Encap (TCP) Connections	The number of TCP module connections going out to the Ethernet media currently active.
Outgoing Encap (TCP) Connection Limit	The maximum number (64) of outgoing TCP connections the module can make at any single moment in time.

Class 1 (CIP) packet statistics

The Class 1 (CIP) Packet Statistics offer information about the speed, duplex and user datagram protocol (UDP) frame rate of TCP connections coming into and going out of the device.

Field:	Definition:
Link Status	Denotes whether the current link is active or inactive.
Speed	The speed that the module is passing data over the Ethernet network.
Mode	The module's communication mode, full-duplex or half-duplex.
Total Packet Capacity	Total number of Class 1 UDP packets your module can handle over the Ethernet network at any time.
Total Class 1 Packets/Second	Number of Class 1 UDP packets your module is currently receiving or transmitting over the Ethernet network.
Actual Reserved Class 1 Capacity	Number of Class 1 UDP packets your module can receive or transmit over the Ethernet network.

Class 1 (CIP) transports

The Class 1 (CIP) Transports offer specific information about Class 1 (CIP) connections coming into and going out of the device.

Field:	Definition:
Type	Type of connection. This field can be either consumer or producer.
Trigger	The mechanism by which the producer produces new data. The mechanism can be Cyclic, Change-of-State, or Application triggered
State	The state of the connection, either active or inactive.
Remote Address	The remote IP address of the connection's originator or destination.
Bridged	Denotes whether the connection is bridged across the controller or not.

Class 3 (CIP) transports

The Class 3 (CIP) Transports screen offers general information about TCP connections coming into and going out of the device

Field:	Definition:
Type	Type of connection. This field can be either consumer or producer. However, for class 3, this will be Client or Server.
State	The state of the connection, either active or inactive.
Remote Address	The IP address of the originator or destination.
Bridged	Denotes whether the connection is bridged across the controller or not.

Notes:

Dynamic Memory Allocation in CompactLogix Controllers

Certain operations cause the controller to dynamically allocate and de-allocate user-available memory, affecting the space available for program logic. As these functions become active, memory is allocated. Memory is then de-allocated when these functions become inactive.

Operations that dynamically allocate memory are:

- Messages
- Connection to a Processor with RSLogix 5000
- RSLinx Tag Optimization
- Trends
- DDE/OPC Topics

Although messages are the most likely to cause dynamic memory allocation on a CompactLogix system, all the above operations are discussed in the following sections, along with general guidelines for estimating the amount of memory allocated.

Messages

Messages can come in and go out of the controller via the Ethernet port or the serial port, causing memory allocation as described in the table below. The memory allocations for messages destined to I/O are accounted for in these allocations. One simple method to reduce the effect that message instructions have on user-available memory is to prevent messages from being sent simultaneously. In general, interlocking messages in this fashion is good practice for peer-to-peer communications.

Type		Connection Established	Dynamic Memory Allocated
Ethernet Port	Incoming	The message is connected (connection established)	1200 bytes
		The message is unconnected (no connection established)	1200 bytes
	Outgoing	All outgoing messages whether connected or unconnnected	1200 bytes
Serial Port	Incoming	All incoming messages whether connected or unconnected	1200 bytes
	Outgoing	All outgoing messages whether connected or unconnected	1200 bytes

RSLinx Tag Optimization

Tag optimization creates three items which allocate memory, a trend object, a trend driver, and a connection.

Item	Description	Memory Allocated
Trend Object	Created in the controller to group the requested tags. One trend object can handle approximately 100 tags (connection points)	80 bytes
Trend Driver	Created to communicate to the trend object	36 bytes/single point (some economy for multiple points in a driver)
Connection	Created between the controller and RSLinx	1200 bytes

EXAMPLE To monitor 100 points:

100 points x 36 bytes = 3600 bytes (Trend Driver)

3600 (Trend Driver) + 80 (Trend Object) + 1200 (Connection)
= approximately 4000 bytes(1)

(1) In general, we estimate that one tag takes about 40 bytes of memory.

Trends

Each trend created in a controller creates a trend object and allocates a buffer for logging as shown below.

Item	Memory Allocated
Trend Object	80 bytes
Log Buffer	4000 bytes

DDE/OPC Topics

A DDE/OPC Topic uses connections based on the following three variables:

- the number of "Maximum Messaging Connections per PLC" configured in RSLinx
- whether the "Use Connections for Writes to ControlLogix processor" is checked
- the number of connections needed to optimize throughput

IMPORTANT These variables are per path. For example, if you set up two different DDE/OPC topics, with different paths to the same controller, the variables limit the connections for each path. Therefore, if you have a limit of 5 connections, it is possible to have 10 connections, with 5 over each path.

Maximum Messaging Connections per PLC

This variable is configured in RSLinx under the "Communications" menu item "Configure CIP Options". This number limits the number of read connections made to Logix controllers from a particular workstation.

Checking "Use Connections for Writes to ControlLogix Controller"

This variable is configured in RSLinx under the "Communications" menu item "Configure CIP Options". This check box indicates whether you want RSLinx to open up additional connections for writing data to a Logix controller.

TIP There is no way to limit the number of write connections, once this box is checked.

Number of Connections Needed to Optimize Throughput

RSLinx only opens the number of connections required to optimize throughput. For example, if you have 1 tag on scan, but have configured RSLinx to allow five connections as the maximum number of connections, RSLinx only opens one connection for the tag. Conversely, if you have thousands of tags on scan and limit the maximum number of CIP connections to five, that is the maximum number of connections that RSLinx establishes to the CompactLogix controller. RSLinx then funnels all of the tags through those five available connections.

Viewing the Number of Open Connections

You can see how many connections are made from your workstation to the CompactLogix controller in RSLinx by selecting "CIP Diagnostics" from the "Connections" menu. The Dispatching tab contains various CIP information, including the number of connections open to the CompactLogix controller.

Numerics

A

B

C

D

P

R

S

T

W

Notes:

How Are We Doing?

Your comments on our technical publications will help us serve you better in the future. Thank you for taking the time to provide us feedback.

You can complete this form and mail it back to us, visit us online at www.ab.com/manuals, or email us at RADocumentComments@ra.rockwell.com

Pub. Title/Type CompactLogix System User Manual

Cat. No. 1769-L35E, -L32E, -L31 Pub. No. 1769-UM011B-EN-P Pub. Date March 2004 Part No. 957867-10

Please complete the sections below. Where applicable, rank the feature (1=needs improvement, 2=satisfactory, and 3=outstanding).

Overall Usefulness 1 2 3	How can we make this publication more useful for you?	
Completeness (all necessary information is provided) 1 2 3	Can we add more information to help you? ☐ procedure/step ☐ illustration ☐ feature ☐ example ☐ guideline ☐ other ☐ explanation ☐ definition	
Technical Accuracy (all provided information is correct) 1 2 3	Can we be more accurate? ☐ text ☐ illustration	
Clarity (all provided information is easy to understand) 1 2 3	How can we make things clearer?	
Other Comments	You can add additional comments on the back of this form.	

Your Name _____ Location/Phone _____

Your Title/Function _____ Would you like us to contact you regarding your comments?

___No, there is no need to contact me

___Yes, please call me

___Yes, please email me at _____

___Yes, please contact me via _____

Return this form to: Allen-Bradley Marketing Communications, 1 Allen-Bradley Dr., Mayfield Hts., OH 44124-9705

Phone: 440-646-3176 Fax: 440-646-3525 Email: RADocumentComments@ra.rockwell.com

PLEASE FASTEN HERE (DO NOT STAPLE)

Other Comments

PLEASE FOLD HERE

PLEASE REMOVE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS MAIL PERMIT NO. 18235 CLEVELAND OH

POSTAGE WILL BE PAID BY THE ADDRESSEE

Allen-Bradley
RELIANCE ELECTRIC
DODGE
ROCKWELL SOFTWARE
Rockwell Automation

1 ALLEN-BRADLEY DR
MAYFIELD HEIGHTS OH 44124-9705

Rockwell Automation Support

Rockwell Automation provides technical information on the web to assist you in using our products. At http://support.rockwellautomation.com, you can find technical manuals, a knowledge base of FAQs, technical and application notes, sample code and links to software service packs, and a MySupport feature that you can customize to make the best use of these tools.

For an additional level of technical phone support for installation, configuration and troubleshooting, we offer TechConnect Support programs. For more information, contact your local distributor or Rockwell Automation representative, or visit http://support.rockwellautomation.com.

Installation Assistance

If you experience a problem with a hardware module within the first 24 hours of installation, please review the information that's contained in this manual. You can also contact a special Customer Support number for initial help in getting your module up and running:

United States	1.440.646.3223 Monday – Friday, 8am – 5pm EST
Outside United States	Please contact your local Rockwell Automation representative for any technical support issues.

New Product Satisfaction Return

Rockwell tests all of our products to ensure that they are fully operational when shipped from the manufacturing facility. However, if your product is not functioning and needs to be returned:

United States	Contact your distributor. You must provide a Customer Support case number (see phone number above to obtain one) to your distributor in order to complete the return process.
Outside United States	Please contact your local Rockwell Automation representative for return procedure.

www.rockwellautomation.com

Corporate Headquarters
Rockwell Automation, 777 East Wisconsin Avenue, Suite 1400, Milwaukee, WI, 53202-5302 USA, Tel: (1) 414.212.5200, Fax: (1) 414.212.5201

Headquarters for Allen-Bradley Products, Rockwell Software Products and Global Manufacturing Solutions
Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe: Rockwell Automation SA/NV, Vorstlaan/Boulevard du Souverain 36-BP 3A/B, 1170 Brussels, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, 27/F Citicorp Centre, 18 Whitfield Road, Causeway Bay, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Headquarters for Dodge and Reliance Electric Products
Americas: Rockwell Automation, 6040 Ponders Court, Greenville, SC 29615-4617 USA, Tel: (1) 864.297.4800, Fax: (1) 864.281.2433
Europe: Rockwell Automation, Brühlstraße 22, D-74834 Elztal-Dallau, Germany, Tel: (49) 6261 9410, Fax: (49) 6261 17741
Asia Pacific: Rockwell Automation, 55 Newton Road, #11-01/02 Revenue House, Singapore 307987, Tel: (65) 351 6723, Fax: (65) 355 1733

Publication 1769-UM011B-EN-P - March 2004 PN 957867-10
Supersedes Publication 1769-UM011A-EN-P - August 2003