Programming Manual

Allen-Bradley

Logix5000 Controllers Function Block Diagram

Catalog Numbers 1756 ControlLogix, 1769 CompactLogix, 1789 SoftLogix, PowerFlex 700S with DriveLogix

Allen-Bradley • Rockwell Software

Rockwell Automation

Important user information

Read this document and the documents listed in the additional resources section about installation, configuration, and operation of this equipment before you install, configure, operate, or maintain this product. Users are required to familiarize themselves with installation and wiring instructions in addition to requirements of all applicable codes, laws, and standards.

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice. If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

	WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.
	ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence
Important:	Identifies information that is critical for successful application and understanding of the product.

Labels may also be on or inside the equipment to provide specific precautions.

	SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.
	BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.
	ARC FLASH HAZARD: Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

Allen-Bradley, Rockwell Software, Rockwell Automation, and TechConnect are trademarks of Rockwell Automation, Inc.

Trademarks not belonging to Rockwell Automation are property of their respective companies.

This manual contains new and updated information. This table contains the changes made to this revision.

Change	Topic
Updated the PIDE graphic to include the AutotuneTag parameter.	Define program/operator control on page 17
Updated the instructions on how to create a text box instructions.	Create a text box on page 21
Updated the steps and and graphic to match changes in software version 24.	Create and assign a new tag on page 25
Added Tag Browser graphic to the instructions.	Assign an existing tag on page 26
Added topic to rename a wire connector.	Rename a wire connector on page 29
Added topic to rename a connector group.	Rename a connector group on page 29

Table of contents

Preface

Studio 5000 environment 7
Additional resources 7

Chapter 1

Program a Function Block Diagram

Introduction 9
Choose the function block elements 10
Choose a tag name for an element 10
Define the order of execution 11
- Data latching 11
- Order of execution 13
- Resolve a loop 14
- Resolve data flow between two blocks 15
- Create a one scan delay 16
- Summary 16

Identify any connectors 17
Define program/operator control 17
Add a sheet 20
Add a function block element 20
Create a text box 21
- Language switching 22

Connect elements 23
- Show or hide a pin 23
- Wire elements together 24
- Mark a wire with the Assume Data Available indicator 24

Assign a tag 25
- Create and assign a new tag 25
- Assign an existing tag 26

Assign an immediate value (constant) 27
- Use an IREF 27
- Enter a value in the tag of a block 28

Connect blocks with an OCON and ICON 28
- Add an OCON 28
- Add an ICON 28
- Rename a wire connector 29
- Rename a connector group 29

Verify the routine 30

Index

This manual shows how to program Logix5000 controllers with the function block diagram (FBD) programming language. This manual is one of a set of related manuals that show common procedures for programming and operating Logix5000 controllers.

For a complete list of common procedures manuals, refer to the Logix5000 Controllers Common Procedures Programming Manual, publication 1756-PM001.

The term Logix5000 controller refers to any controller that is based on the Logix5000 operating system.

Studio 5000 environment

The Studio 5000 Automation Engineering & Design Environment™ combines engineering and design elements into a common environment. The first element is the Studio 5000 Logix Designer™ application. The Logix Designer application is the rebranding of RSLogix™ 5000 software and will continue to be the product to program Logix5000™ controllers for discrete, process, batch, motion, safety, and drive-based solutions.

The Studio 5000® environment is the foundation for the future of Rockwell Automation® engineering design tools and capabilities. The Studio 5000 environment is the one place for design engineers to develop all elements of their control system.

Additional resources

These documents contain additional information concerning related Rockwell Automation products.

Resource	Description
Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1	Provides general guidelines for installing a Rockwell Automation industrial system.

Resource	Description
Product Certifications webpage, available at http://ab.rockwellautomation.com	Provides declarations of conformity, certificates, and other certification details.

You can view or download publications at http://www.rockwellautomation.com/literature. To order paper copies of technical documentation, contact your local Rockwell Automation distributor or sales representative.

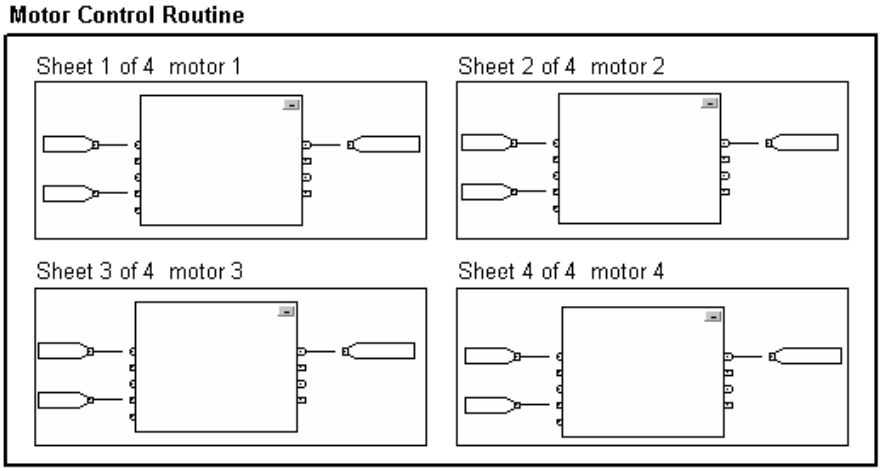
Program a Function Block Diagram

Introduction

To make it easier to navigate through a function block routine, divide the routine into a series of sheets.

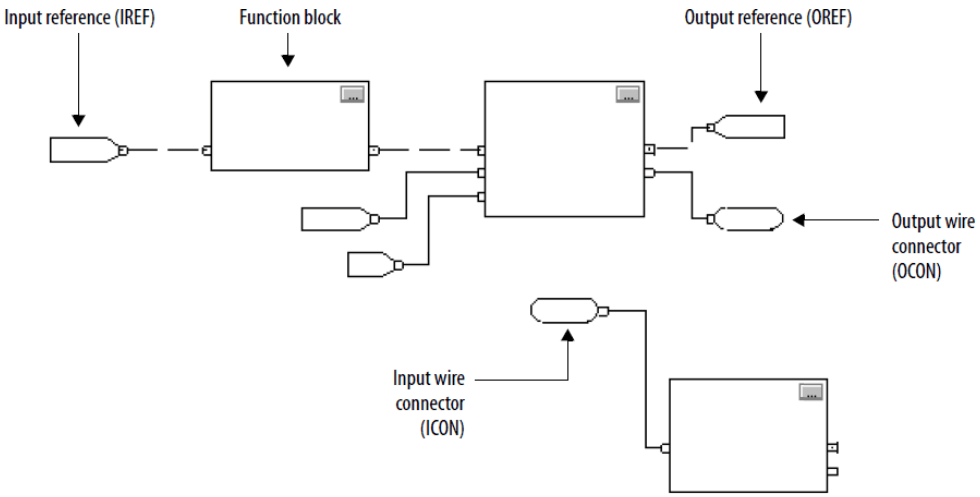
- Sheets help you organize and find your function blocks. They do not affect the order in which the function blocks execute.
- When the routine executes, all the sheets execute.
- In general, use one sheet for each device, such as a motor or valve

Example: **Function block routine divided into sheets**



Choose the function block elements

To control a device, use these elements.



To choose the function block elements, use these guidelines.

If you want to:	Then use this element:
Supply a value from an input device or tag	Input reference (IREF)
Send a value to an output device or tag	Output reference (OREF)
Perform an operation on an input value or values and produce an output value or values	Function block
Transfer data between function blocks when they are: • Far apart on the same sheet. • On different sheets within the same routine.	Output wire connector (OCON) and an input wire connector (ICON)
Disperse data to several points in the routine	Single output wire connector (OCON) and multiple input wire connectors (ICON)

Choose a tag name for an element

Each function block uses a tag to store configuration and status information about the instruction.

- When you add function block instruction, the Logix Designer application automatically creates a tag for the block. Use this tag, rename the tag, or assign a different tag.
- For IREFs and OREFs, create a tag or assign an existing tag.

For a:	Specify:
Tag	*tag_name*
Bit number of a larger data type	*tag_name.bit_number*
Member of a structure	*tag_name.member_name*
Element of a one dimension array	*tag_name[x]*

For a:	Specify:
Element of a two dimension array	*tag_name[x,y]*
Element of a three dimension array	*tag_name[x,y,z]*
Element of an array within a structure	*tag_name.member_name[x]*
Member of an element of an array	*tag_name[x,y,z].member_name*

where:

x is the location of the element in the first dimension.

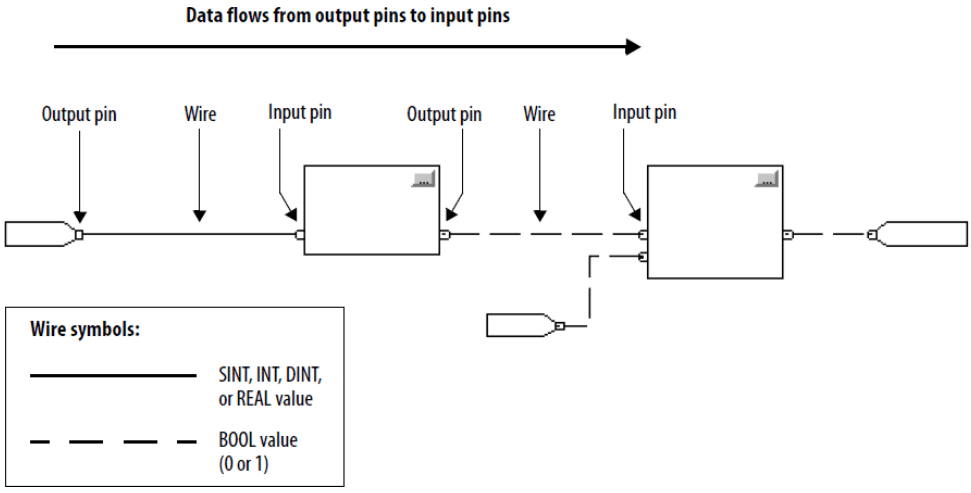
y is the location of the element in the second dimension.

z is the location of the element in the third dimension.

For a structure within a structure, add an additional *member_name*.

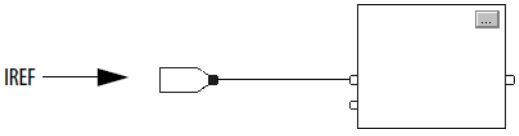
Define the order of execution

Define execution order (flow of data) by wiring elements together and indicating any input (feedback) wires, if necessary. The location of a block does not affect the order in which the blocks execute.

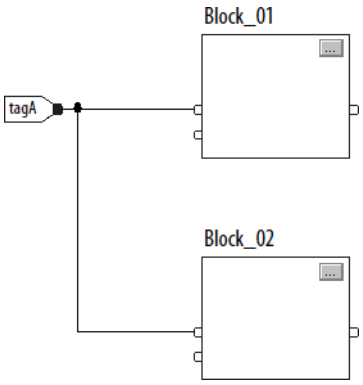


Data latching

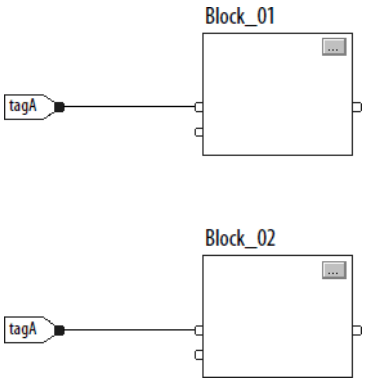
If you use an IREF to specify input data for a function block instruction, the data in that IREF is latched for the scan of the function block routine. The IREF latches data from program-scoped and controller-scoped tags. The controller updates all IREF data at the beginning of each scan as shown in this diagram.



In this example, the value of tagA is stored at the beginning of the execution of the routine. The stored value is used when Block_01 executes. The same stored value is also used when Block_02 executes. If the value of tagA changes during execution of the routine, the stored value of tagA in the IREF does not change until the next execution of the routine.

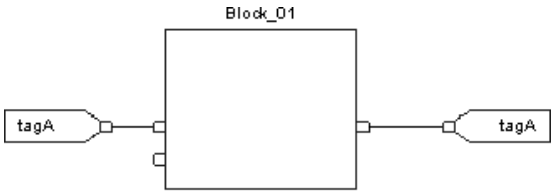


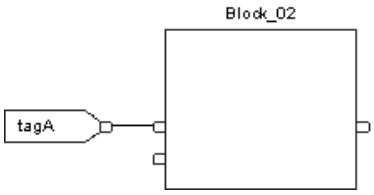
This example is the same as the previous example. The value of tagA is stored only once at the beginning of the execution of the routine. The routine uses this stored value throughout the routine.



With version 11 and later of the application, use the same tag in multiple IREFs and an OREF in the same routine. Because the values of tags in IREFs are latched every scan through the routine, all IREFs use the same value, even if an OREF obtains a different tag value during execution of the routine. In this example, if tagA has a value of 25.4 when the routine starts executing this scan, and Block_01 changes the value of tagA to 50.9, the second IREF wired into Block_02 still uses a value of 25.4 when Block_02 executes this scan.

The new tagA value of 50.9 is not used by any IREFs in this routine until the start of the next scan.





Order of execution

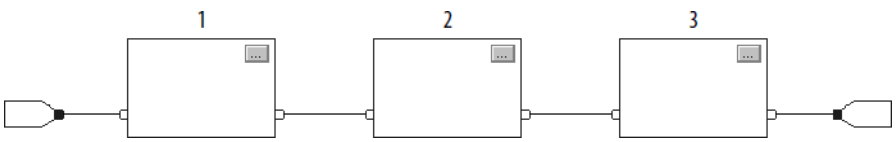
The Logix Designer application automatically determines the order of execution for the function blocks in a routine.

- When you verify a function block routine.
- When you verify a project that contains a function block routine.
- When you download a project that contains a function block routine.

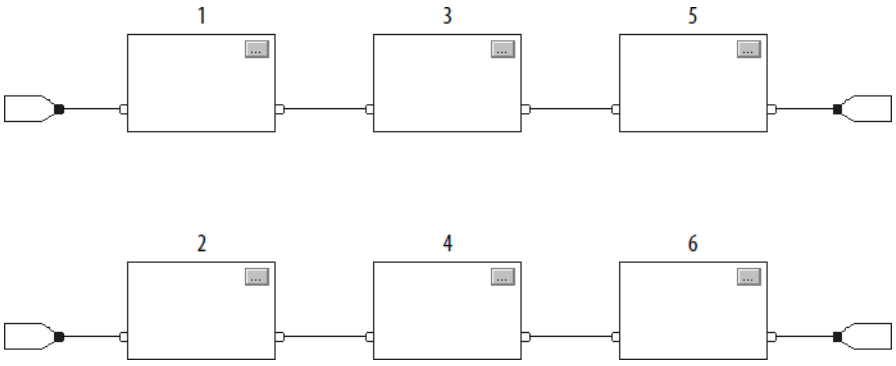
Define the order of execution by wiring function blocks together and indicating the data flow of any feedback wires, if necessary.

If function blocks are not wired together, it does not matter which block executes first. There is no data flow between the blocks.

If the blocks are wired sequentially, the order of execution moves from input to output. The inputs of a block require data to be available before the controller can execute that block. In this example, block 2 has to execute before block 3 because the outputs of block 2 feed the inputs of block 3.

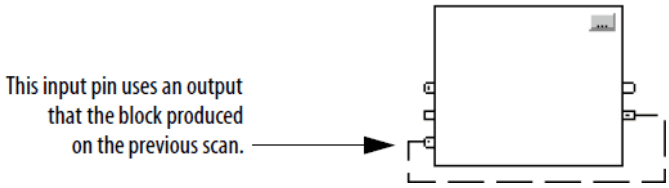


The order of execution is only relative to the blocks that are wired together. The two groups of blocks in this example are not wired together. The blocks within a specific group execute in the appropriate order in relation to the blocks in that group.

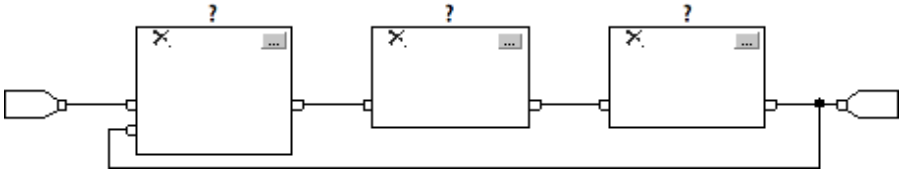


Resolve a loop

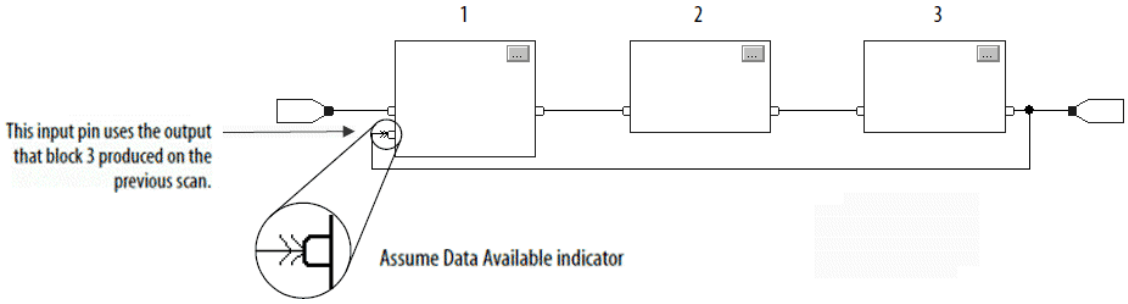
To create a feedback loop around a block, wire an output pin of the block to an input pin of the same block. In this example, the loop contains only a single block, so execution order does not matter.



If a group of blocks are in a loop, the controller cannot determine which block to execute first, and it cannot resolve the loop, as illustrated in this example.

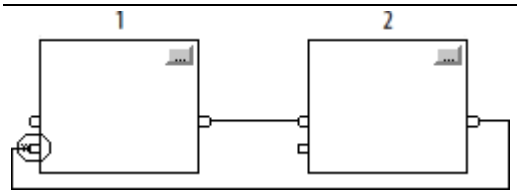
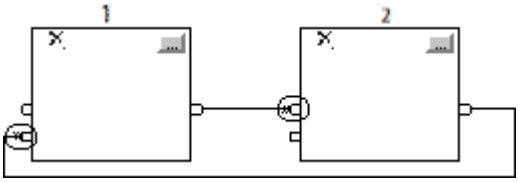


To identify which block to execute first, mark the input wire that creates the loop (the feedback wire) with the **Assume Data Available** indicator. In this example, block 1 uses the output from block 3 that was produced in the previous execution of the routine.



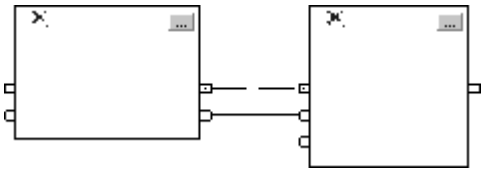
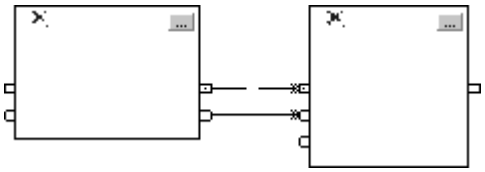
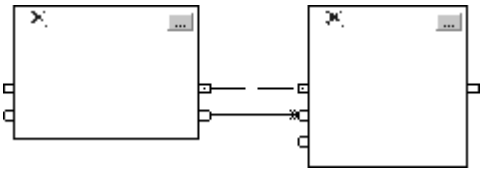
The **Assume Data Available** indicator defines the data flow within the loop. The arrow indicates that the data serves as input to the first block in the loop.

Do not mark all the wires of a loop with the **Assume Data Available** indicator.

This example is OK.	This example is not OK.
 The **Assume Data Available** indicator defines the data flow within the loop.	 The controller cannot resolve the loop because all the wires use the **Assume Data Available** indicator.

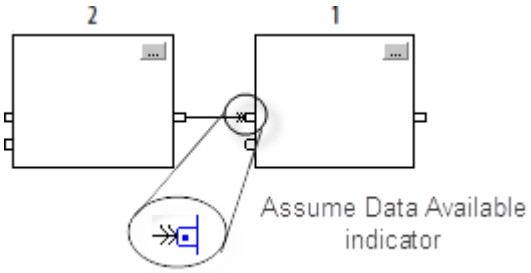
Resolve data flow between two blocks

If you use two or more wires to connect two blocks, use the same data flow indicators for all of the wires between the two blocks.

This is OK	This is NOT OK
In this example, neither wire uses the **Assume Data Available** indicator.  In this example, both wires use the **Assume Data Available** indicator. 	In this example, only one wire uses the **Assume Data Available** indicator. 

Create a one scan delay

Use the **Assume Data Available** indicator to produce a one scan delay between blocks. In this example, block 1 executes first. It uses the output from block 2 that was produced in the previous scan of the routine.



Summary

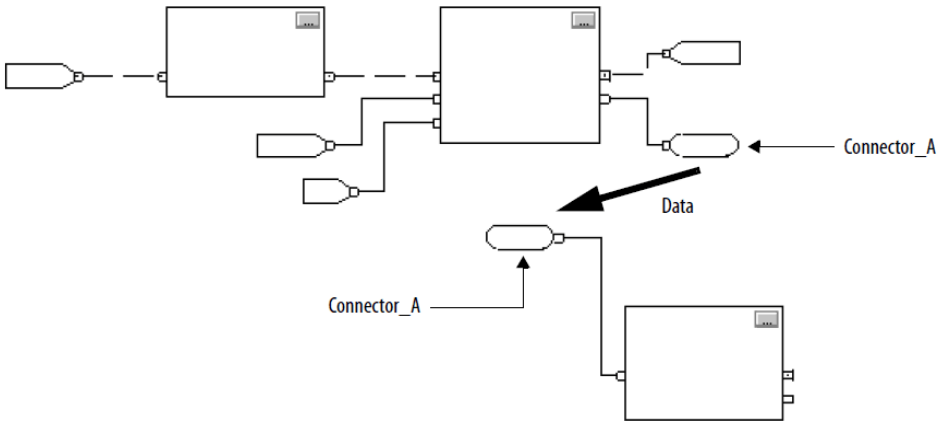
A function block routine executes in this order.

1. The controller latches all data values in IREFs.
2. The controller executes the other function blocks in the order determined by how they are wired.
3. The controller writes outputs in OREFs.

Identify any connectors

Like wires, connectors transfer data from output pins to input pins. Use connectors when:

- The elements that you want to connect are on different sheets within the same routine.
- A wire is difficult to route around other wires or elements.
- You want to disperse data to several points in the routine.



To use connectors, use these rules.

- Each OCON requires a unique name.
- For each OCON, you must have at least one corresponding ICON, such as an ICON with the same name as the OCON.
- Multiple ICONs can reference the same OCON. This lets you disperse data to several points in your routine.

Define program/operator control

These instructions support the concept of Program/Operator control.

- Enhanced Select (ESEL)
- Totalizer (TOT)
- Enhanced PID (PIDE)
- Ramp/Soak (RMPS)
- Discrete 2-State Device (D2SD)
- Discrete 3-State Device (D3SD)

Program/Operator control lets you control these instructions simultaneously from the user program and from an operator interface device. When in Program control, the instruction is controlled by the Program inputs to the instruction. When in Operator control, the instruction is controlled by the Operator inputs to the instruction.

Program or Operator control is determined by using these inputs:

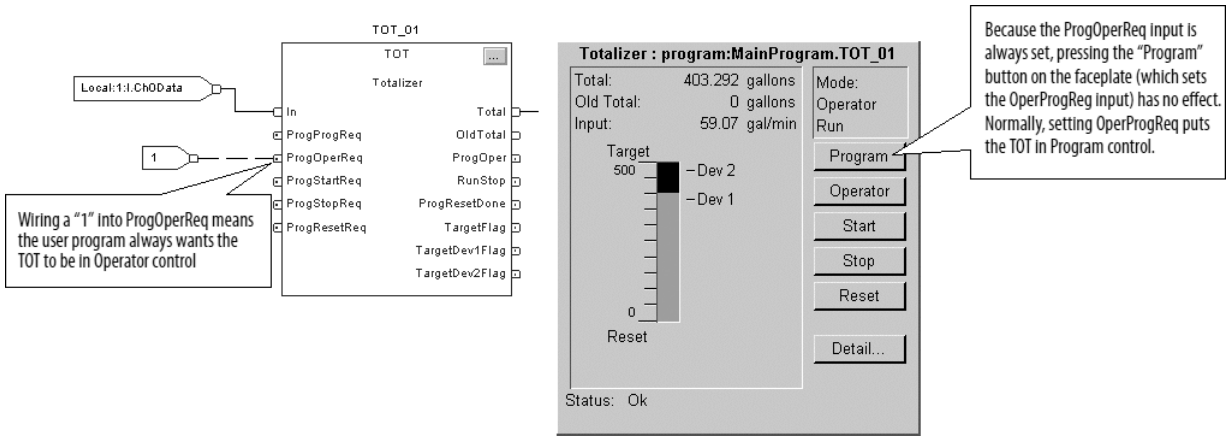
Input	Description
.ProgProgReq	A program request to go to Program control.
.ProgOperReq	A program request to go to Operator control.
.OperProgReq	An operator request to go to Program control.
.OperOperReq	An operator request to go to Operator control.

To determine whether an instruction is in Program or Operator control, examine the **ProgOper** output. If **ProgOper** bit is set, the instruction is in Program control. If **ProgOper** bit is cleared, the instruction is in Operator control.

Operator control takes precedence over Program control if both input request bits are set. For example, if **ProgProgReq** and **ProgOperReq** bits are set, the instruction goes to Operator control.

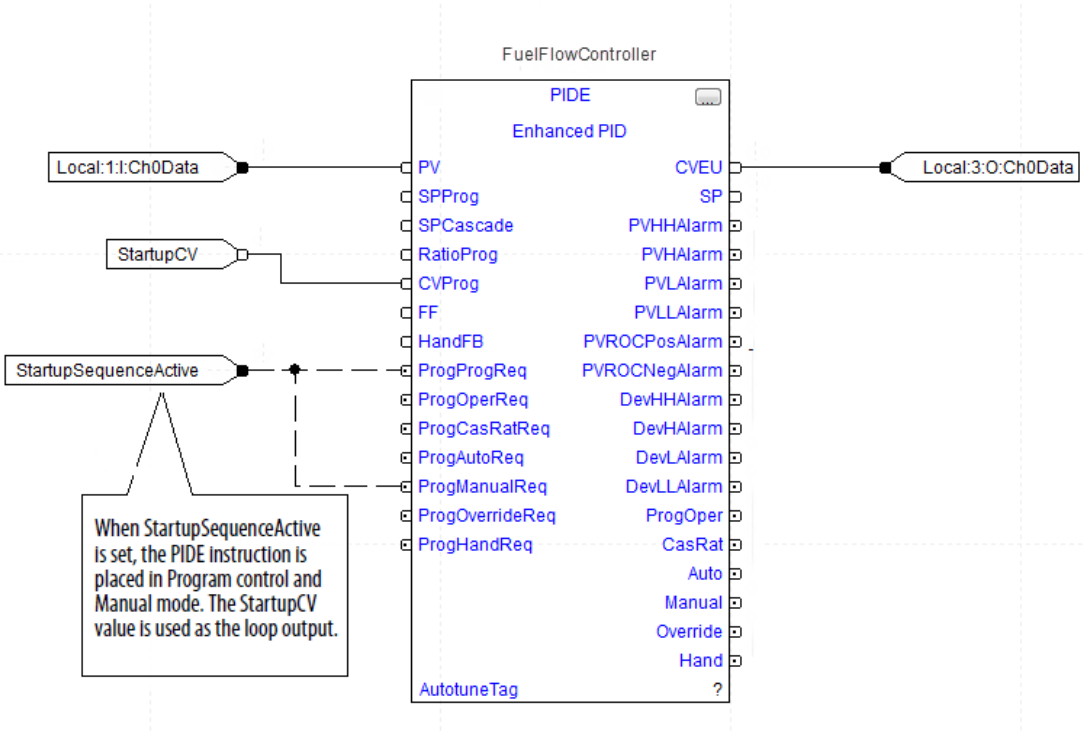
The Program request inputs take precedence over the Operator request inputs. This provides the capability to use the **ProgProgReq** and **ProgOperReq** inputs to *lock* an instruction in a desired control.

For example, assume that a Totalizer instruction is always used in Operator control, and the user program never controls the running or stopping of the Totalizer. In this case, wire a literal value of 1 into the **ProgOperReq** input. This prevents the operator from ever putting the Totalizer into Program control by setting the **OperProgReq** input from an operator interface device.



Likewise, constantly setting the **ProgProgReq** input can "lock" the instruction into Program control. This is useful for automatic startup sequences when you want the program to control the action of the instruction without an operator inadvertently taking control of the instruction. In this example, the program sets the **ProgProgReq** input during the startup, and clears the **ProgProgReq** input once the startup is complete. Once the **ProgProgReq** input is cleared, the instruction remains in Program control until it receives a request to change. For

example, the operator sets the **OperOperReq** input from a faceplate to take control of that instruction. This example shows how to lock an instruction into Program control.



Operator request inputs to an instruction are always cleared by the instruction when it executes. This lets operator interface work with these instructions by setting the desired mode request bit. You do not have to program the operator interface to reset the request bits. For example, if an operator interface sets the **OperAutoReq** input to a PIDE instruction, when the PIDE instruction executes, it determines the appropriate response and clears the **OperAutoReq**.

Program request inputs are not normally cleared by the instruction because these are normally wired as inputs into the instruction. If the instruction clears these inputs, the input would get set again by the wired input. There might be situations where you want to use other logic to set the Program requests to be cleared by the instruction. In this case, set the **ProgValueReset** input and the instruction always clears the Program mode request inputs when it executes.

In this example, a rung of ladder logic in another routine is used to one-shot latch a **ProgAutoReq** input to a PIDE instruction when a push button is pushed. Because the PIDE instruction automatically clears the Program mode requests, you do not have to write any ladder logic to clear the **ProgAutoReq** input after the routine executes. The PIDE instruction receives only one request to go to Auto every time the push button is pressed.

Example: When the TIC101AutoReq button is pressed, one-shot latch **ProgAutoReq** for the PIDE instruction TIC101.
TIC101 is configured with the **ProgValueReset** input set, so when the PIDE instruction executes, it automatically clears **ProgAutoReq**.

Add a sheet

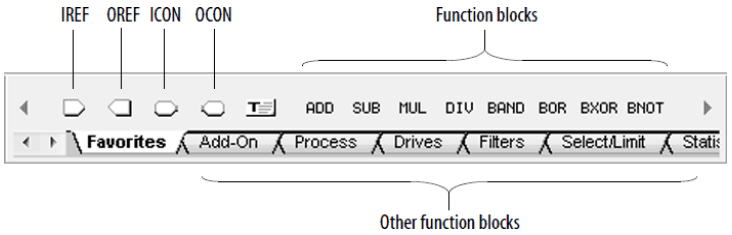
To add a sheet to a function block routine:

1. On the **Sheet** toolbar, click **Add Sheet** .

2. In the **Description** box, type a description of the sheet. Follow the IEC-1131 naming standard. The description must not be greater than 50 characters.

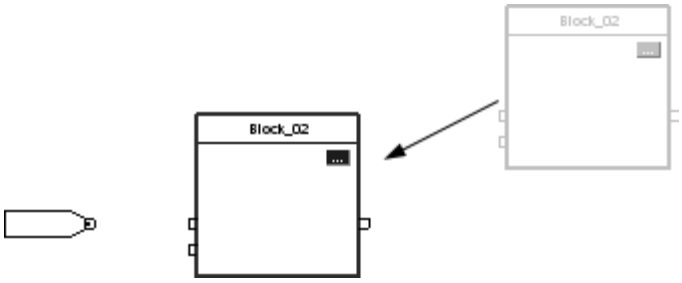
Add a function block element

Use the **Language Element** toolbar to add a function block element to a routine.

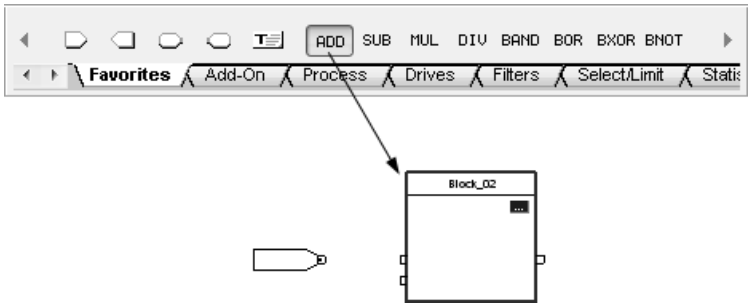


1. On the **Language Element** toolbar, click the button for the element that you want to add.

2. Drag the element to the desired location.



Tip: You can also drag the button for the element directly to the desired location.



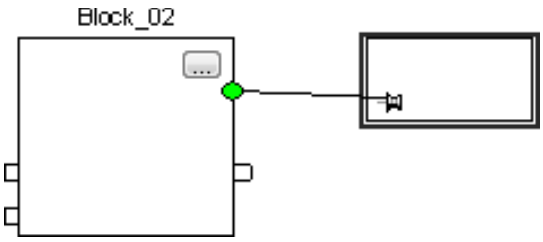
Important: Use caution when copying and pasting components between different versions of the Logix Designer application. The application only supports pasting to the same version or newer version. Pasting to a prior version of the application is not supported. When pasting to a prior version, the paste action may succeed but the results may not be as intended.

Create a text box

Create a text box to add notes that clarify the function of an FBD element, such as blocks, references, and connectors. Text boxes are only stored in the offline, ACD project file. Text boxes are not downloaded into controller memory.

1. On the **Language Element** toolbar, click **Text Box** .
2. In the the FBD editor, drag the text box to a location near the element to which it applies.
3. Double-click the **text box** and enter the desired text.
4. Press **Ctrl+Enter**.

5. To attach the text box to a specific element, click the pin symbol in the text box and click the corresponding element. A green dot shows a valid connection point.



Language switching

With version 17 and later of the application, you have the option to display project documentation, such as tag descriptions and rung comments for any supported localized language. You can store project documentation for multiple languages in a single project file rather than in language-specific project files. Define all the localized languages that the project will support and set the current, default, and optional custom localized language. The default language is used if the current content of the language is blank for a particular component of the project. However, you can use a custom language to tailor documentation to a specific type of project file user.

Enter the localized descriptions in a project by programming in that language, or by using the import/export utility to translate the documentation offline and import it back into the project. Once you enable language switching, you can dynamically switch between languages.

Project documentation that supports multiple translations within a project includes:

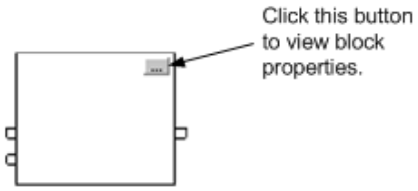
- Component descriptions in tags, routines, programs, user-defined data types, and Add-On Instructions.
- Equipment phases.
- Trends.
- Controllers.
- Alarm Messages (in ALARM_ANALOG and ALARM_DIGITAL configuration).
- Tasks.
- Property descriptions for modules in the Controller Organizer.
- Rung comments, SFC text boxes, and FBD text boxes.

For more information on enabling a project to support multiple translations of project documentation, see the online help.

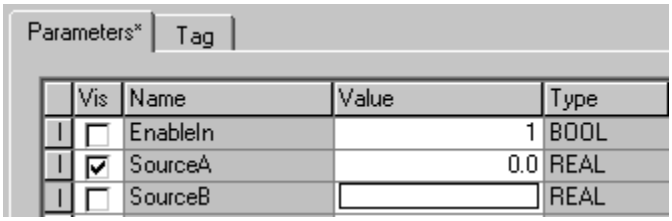
Connect elements

Show or hide a pin

When you add a Function Block instruction, the block appears with a set of pins for the default parameters. The rest of the pins are hidden. You can show or hide a pin on the **Parameters** tab in the **Properties** dialog box.



1. In the block, click **Properties** .

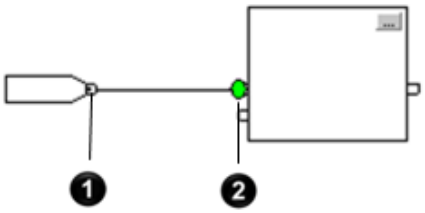


2. In the **Properties** dialog box, on the **Parameters** tab, clear the **Vis** check box to hide the pin. Select the **Vis** check box to show the pin.

3. Click **OK**.

Wire elements together

Wire (connect) two elements together by clicking the output pin of the first element and clicking the input pin of the other element. A green dot shows a valid connection point.

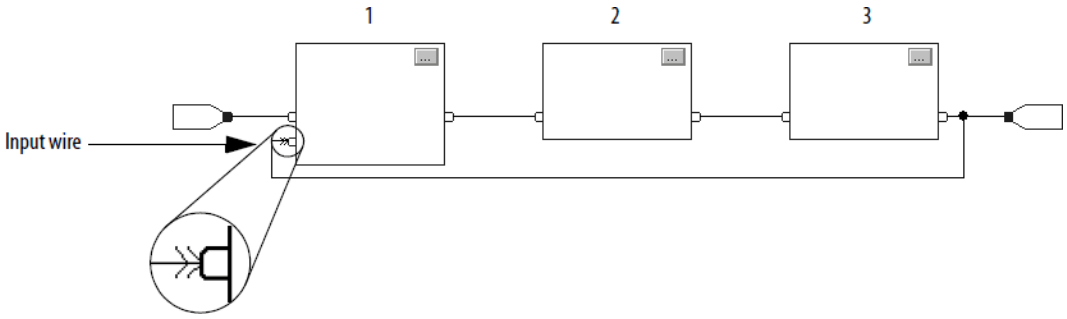


Item	Description
1	Output pin of the first element
2	Input pin of the second element

Mark a wire with the Assume Data Available indicator

When there are a group of blocks in a loop, identify which block executes first. The **Assume Data Available** indicator marks the input wire that creates the loop (the feedback wire). It defines the data flow within the loop.

- To define a wire as an input wire, right-click the wire and click **Assume Data Available**.



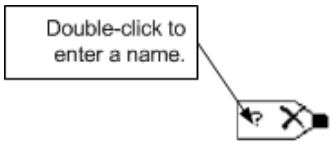
The arrow indicates that the data serves as input to the first block in the loop.

Assign a tag

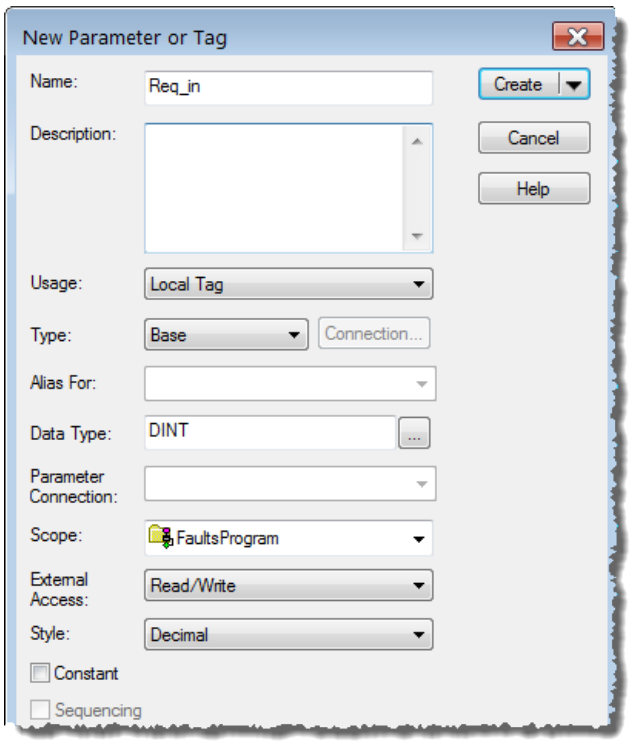
Create and assign a new tag to the connector or assign an existing tag to the connectors.

Create and assign a new tag

1. Double-click the operand area to enter a name.

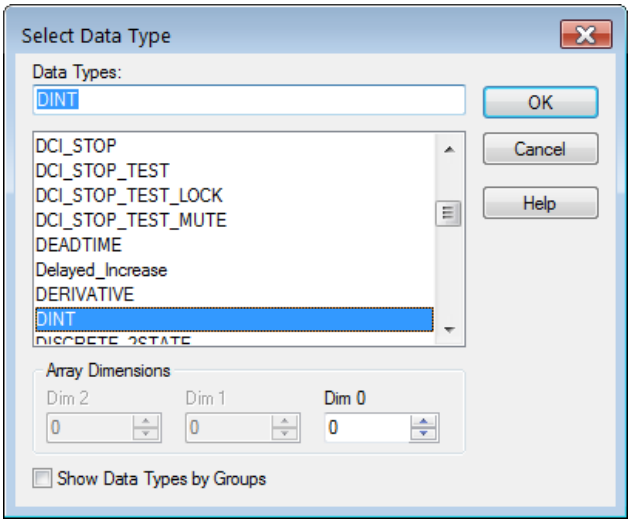


2. In the box, type a name for the tag and press **Enter**.
3. Right-click the tag name, and click **New <tag name>**.

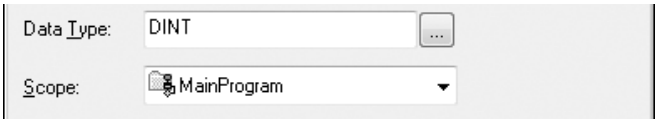


4. On the **New Parameter or Tag** dialog box, in the **Usage** list, click a usage value for the tag. The default is a local tag.
5. In the **Type** list, click the tag type.

6. In the **Data Type** box, click ....



7. On the **Select Data Type** dialog box, click the data type for the tag.
8. If the tag is an array, in the **Dim 0** box, type or select the number of elements in each dimension.
9. Click **OK**.
10. On the **New Parameter or Tag** dialog box, in the **Scope** list, click the scope for the tag.



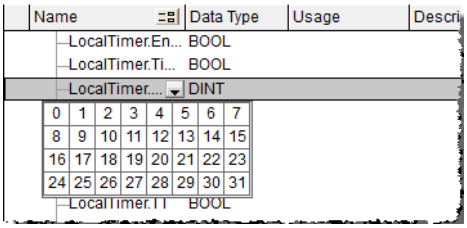
11. Click **Create**.

Assign an existing tag

1. Double-click the operand area.

2. In the box, click the down arrow to select the tag.

3. On the **Tag Browser**, click the tag, or select the bit by clicking the down arrow to the right of the tag and clicking the bit.



4. Press **Enter** or click a different spot on the diagram.

Assign an immediate value (constant)

You can assign a constant value instead of a tag value to an input parameter.

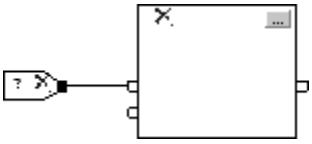
If you want to:	Then:
Make the value visible on the diagram and in reports	Use an IREF on page 27
Change the value online without editing the routine	Enter a Value in the Tag of a Block on page 28

Use an IREF

Complete these steps to assign a value to an IREF.

1. Add the IREF to the routine. For instructions on adding an element, see Add a function block element on page 20.

2. Wire the IREF to the input pin that gets the value. For instructions on wiring elements together, see Wire elements together on page 24.



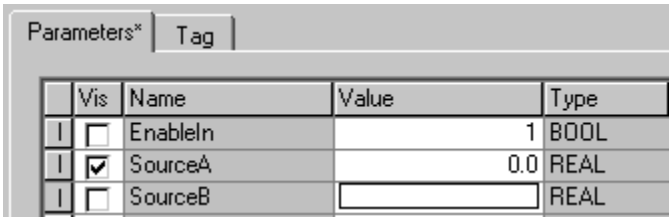
3. Double-click the operand area of the IREF.

4. In the box, type the value and press **Enter**.

Enter a value in the tag of a block

Complete these steps to assign a value to a parameter when on wire connects to its pin.

1. In the block, click **Properties** .

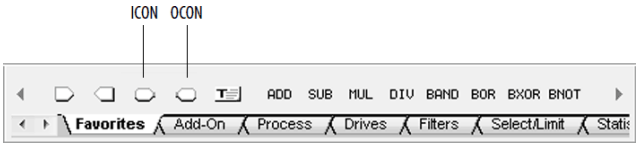


2. On the **Parameters** tab, in the **Value** box of the desired parameter, type the value.

3. Click **OK**.

Connect blocks with an OCON and ICON

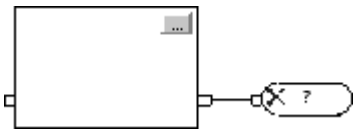
Use an output wire connector (OCON) or input wire connector (ICON) to transfer data between sheets or in complex wiring situations.



Add an OCON

1. Add an OCON and place it near the output pin that supplies the value.

2. Wire the OCON to the output pin.



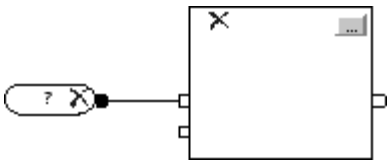
3. Double-click the operand area of the OCON.

4. In the box, type a name that identifies the connector and press **Enter**.

Add an ICON

1. Add an ICON and place it near the input pin that gets the value from the corresponding OCON.

2. Wire the ICON to the input pin.



3. Double-click the operand area of the ICON.
4. In the box, click the down arrow and click the name of the OCON that supplies the value to this connector.
5. Click **Enter** or click a blank spot on the diagram.

Rename a wire connector

Edit the name of an input wire connector or an output wire connector in a routine.

1. Right-click the operand area of the desired ICON or OCON, and click **Rename Element**.
2. In the box, type or select a new name and press **Enter**.

Rename a connector group

Wire connectors that share the same connector name can be changed in a routine.

Tip: If there are wire connectors with the new name, the renamed wire connectors merge with the existing connectors.

1. Right-click the operand area of the desired ICON or OCON, and click **Rename Connector Group**.
2. In the box, type or select the new name and press **Enter**.

   All instances in the group are changed to the new name.

Verify the routine

As you program your routine, periodically verify your work.

1. On the **Standard** toolbar, click .

   If there are errors, they are listed in the **Output** window on the **Errors** tab at the bottom of the Logix Designer application.

2. Press **F4** to go to the first error or warning.
3. Correct the error according to the description in the **Errors** tab.
4. Repeat steps 1...3 until all of the errors are corrected.
5. To close the **Output window,** press **Alt + 1.**

A

assume data available 14, 15, 16, 24

C

create
- tag
  - function block diagram 25

E

enter
- ICON 28
- OCON 28

execution order
- function block diagram 11

F

feedback loop
- function block diagram 14

function block diagram
- connect elements 24
- create a scan delay 16
- hide a pin 23
- latching data 11
- order of execution 11
- rename a connector group 29
- rename a wire connector 29
- resolve a loop 14
- resolve data flow between blocks 15
- resolve loop 24
- show a pin 23

I

ICON
- add 28
- rename a connector group 29
- rename a wire connector 29

IREF
- latching data 11
- to assign immediate value 27

L

latching data
- function block diagram 11

O

OCON
- add 28
- rename a connector group 29
- rename a wire connector 29

order of execution
- function block diagram 11

P

program/operator control
- overview 17

R

rename a connector group 29
rename a wire connector 29
routine
- verify 30

S

scan delay
- function block diagram 16

T

tag
- assign
  - function block diagram 25

U

unresolved loop
- function block diagram 14

V

verify

routine 30

W

wire
function block diagram 11, 14

Rockwell Automation support

Rockwell Automation provides technical information on the web to assist you in using its products. At http://www.rockwellautomation.com/support you can find technical and application notes, sample code, and links to software service packs. You can also visit our Support Center at https://rockwellautomation.custhelp.com for software updates, support chats and forums, technical information, FAQs, and to sign up for product notification updates.

In addition, we offer multiple support programs for installation, configuration, and troubleshooting. For more information, contact your local distributor or Rockwell Automation representative, or visit http://www.rockwellautomation.com/services/online-phone.

Installation assistance

If you experience a problem within the first 24 hours of installation, review the information that is contained in this manual. You can contact Customer Support for initial help in getting your product up and running.

United States or Canada	1.440.646.3434
Outside United States or Canada	Use the Worldwide Locator available at http://www.rockwellautomation.com/locations, or contact your local Rockwell Automation representative.

New product satisfaction return

Rockwell Automation tests all of its 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, follow these procedures.

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

Documentation feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete the feedback form, publication RA-DU002.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Rockwell Automation Publication 1756-PM009E-EN-P - October 2014

Supercedes Publication 1756-PM009D-EN-P - December 2012

Copyright © 2014 Rockwell Automation, Inc. All rights reserved. Printed in the U.S.A.