Logix5000 Controllers

Catalog Numbers 1756 ControlLogix, 1756 GuardLogix, 1768 CompactLogix, 1768 Compact GuardLogix, 1769 CompactLogix, 1789 SoftLogix, PowerFlex with DriveLogix

Quick Start
Important User Information

Solid state equipment has operational characteristics differing from those of electromechanical equipment. Safety Guidelines for the Application, Installation and Maintenance of Solid State Controls (publication SGI-1.1 available from your local Rockwell Automation sales office or online at http://www.rockwellautomation.com/literature/) describes some important differences between solid state equipment and hard-wired electromechanical devices. Because of this difference, and also because of the wide variety of uses for solid state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

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.

**IMPORTANT**

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

**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.

**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.

---

Allen-Bradley, Rockwell Automation, CompactLogix, ControlLogix, DriveLogix, FactoryTalk Batch, FactoryTalk View SE, PhoenixManager, PowerFlex, RSLogix, RSLogix Classic, RSLogix 5000, SoftLogix, and TechConnect are trademarks of Rockwell Automation, Inc.

Trademarks not belonging to Rockwell Automation are property of their respective companies.
This version of the quick start corresponds to revision 18 of the Logix5000 controller firmware.

<table>
<thead>
<tr>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descriptions of controller modes</td>
<td>32</td>
</tr>
<tr>
<td>Language switching</td>
<td>108</td>
</tr>
<tr>
<td>Additional information for finalizing edits in larger projects</td>
<td>124</td>
</tr>
</tbody>
</table>
Notes:
## Table of Contents

### Preface
- About This Publication .......................................................... 9
- Required Software ................................................................. 9
- Additional Resources .............................................................. 10

### Chapter 1
**Program and Test a Simple Project**
- What You Need ................................................................. 11
- Before You Begin .............................................................. 12
- Follow These Steps ............................................................ 13
- Create a Project for the Controller ....................................... 14
  - Conventions for Names .................................................... 15
- Add Your I/O Modules ........................................................ 15
- Look at Your I/O Data ......................................................... 17
- Ladder Logic ................................................................. 19
  - Enter Ladder Logic ...................................................... 20
- Enter a Function Block Diagram .......................................... 21
  - Create a Routine .......................................................... 21
  - Call the Routine .......................................................... 22
  - Enter a Function Block Diagram ...................................... 23
  - Configure a Function Block Instruction ............................. 24
- Assign Alias Tags for Your Devices ..................................... 25
  - Show or Hide Alias Information ..................................... 27
- Establish a Serial Connection to the Controller ..................... 28
- Download a Project to the Controller .................................. 30
- Select the Operating Mode of the Controller ......................... 32

### Chapter 2
**Organize a Project**
- What You Need ................................................................. 35
- Before You Begin .............................................................. 35
- Follow These Steps ............................................................ 36
- Configure the Task Execution ............................................. 36
- Create Additional Programs ............................................... 38
- Create User-defined Data Types ......................................... 40
  - Define Your Routines .................................................... 43
  - Define a Routine for Each Section of a Machine or Process .... 44
  - Identify the Programming Languages That Are Installed ....... 44
  - Assign a Programming Language to Each Routine .............. 45
  - Divide Each Routine Into More Meaningful Increments ....... 46
- Assign Main Routines ......................................................... 47
- Configure the Controller .................................................... 48
- Configure I/O Modules ....................................................... 49
# Table of Contents

## Chapter 3

**Program Add-On Instructions**
- What You Need ................................................. 51
- Follow These Steps ........................................... 51
- Insert an Add-On Instruction ............................ 52
- Copy an Add-On Instruction Definition ............. 53
- Import an Add-On Instruction Definition ........... 54
- Access a Parameter That Is Not Visible ............ 55
  - Function Block ........................................... 55
  - Ladder Logic and Structured Text ................. 55
- Monitor or Change the Value of a Parameter of an
  Add-On Instruction ........................................ 56
- View the Logic of an Add-On Instruction .......... 57
- Edit and Monitor an Add-On Instruction .......... 58
- Update an Add-On Instruction to a Newer Revision ................................. 58

## Chapter 4

**Program an Equipment Phase**
- What You Need ................................................. 59
- Follow These Steps ........................................... 59
- Create an Equipment Phase ............................. 60
- Create a State Routine .................................... 60
- Manually Step Through the States .................. 61
- Configure the Initial State for an Equipment Phase ................................. 63
- Open the Configuration for an Equipment Phase ................................. 64
- Configure an Equipment Phase ......................... 64

## Chapter 5

**Program a Project Offline**
- What You Need ................................................. 67
- Before You Begin ............................................. 67
- Follow These Steps ........................................... 67
- Enter Ladder Logic .......................................... 68
  - Add a Rung or an Instruction ......................... 69
  - Add a Branch ............................................. 69
  - Add a Level to a Branch .............................. 69
  - Delete an Element ..................................... 70
  - Use the Keyboard to Add an Element ............. 70
- Enter Logic Using ASCII Text ............................ 71
- Enable Quick Keys .......................................... 72
- Export/Import Ladder Logic .............................. 73
  - When You Import Rungs ................................. 73
  - Export Rungs ............................................. 74
  - Import Rungs ............................................. 75
- Check Alias Tags .............................................. 76
# Table of Contents

Enter a Function Block Diagram ........................................... 77
Use the Keyboard to Add an Element ................................. 78
Connect Elements .......................................................... 79
Resolve a Loop ............................................................... 80
Add Sheet ................................................................. 80
Use a Faceplate for a Function Block .................................. 81
Set Up a Topic .............................................................. 82
Add a Faceplate to Microsoft Excel Software ..................... 83
Enter Structured Text ...................................................... 84
Browse For an Instruction ............................................... 85
Assign Operands to an Instruction ....................................... 86
Enter a Sequential Function Chart ..................................... 87
Enter an SFC ............................................................... 88
Assign Operands ........................................................... 89
Create a Tag ............................................................... 90
Select an Existing Tag .................................................... 91
Verify a Project ............................................................ 92
Guidelines for Tags ....................................................... 94

## Chapter 6

Document a Project

What You Need ............................................................. 97
Follow These Steps ....................................................... 97
User-defined Data Type ................................................... 98
Turn Pass-Through and Append Descriptions On or Off ........ 99
Paste a Pass-Through Description ..................................... 100
Add Rung Comments ..................................................... 101
Rung Comments Using Microsoft Excel ............................ 102
Export the Existing Comments ......................................... 102
Edit the Export File ....................................................... 103
Import the New Comments ............................................. 104
Comments in a Function Block Diagram or SFC .................. 105
Set the Word Wrap Option .............................................. 105
Add a Text Box ........................................................... 106
Comments in Structured Text .......................................... 107
Language Switching ...................................................... 108

## Chapter 7

Go Online to the Controller

What You Need ............................................................. 109
Follow These Steps ....................................................... 109
Establish EtherNet/IP Communication with the Controller .... 110
Equipment and Information That You Need ....................... 111
Connect Your EtherNet/IP Device and Computer .................. 112
Assign an IP Address to the Controller or Communication Module .................................................. 112
Assign an IP Address to Your Computer ............................ 114
# Table of Contents

Configure an Ethernet Driver ........................................... 115
Online with a Controller ............................................... 116
  If Your Computer Has the Project For the Controller .......... 117
  If Your Computer Does Not Have the Project
  For the Controller .................................................. 117

## Chapter 8

**Program a Project Online**

What You Need .......................................................... 119
Follow These Steps .................................................... 119
Edit Logic While Online .............................................. 119
  Start a Pending Edit ............................................... 121
  Make and Accept Your Edits ....................................... 122
  Test the Edits ...................................................... 122
  Assemble and Save the Edits ....................................... 123
Finalize All Edits in a Program ..................................... 124

## Chapter 9

**Troubleshoot the Controller**

What You Need .......................................................... 125
Follow These Steps .................................................... 125
Troubleshoot I/O Communication ................................. 126
Clear a Major Fault .................................................... 127
Search Functions in a Project ....................................... 128
  Search for All Occurrences of a Element .................... 128
  Go to an Instruction ............................................... 129
Browse Logic ............................................................ 130
Forcing an I/O Value ................................................. 131
  Install an I/O Force (Force an I/O Value) ................. 133
  Remove an Individual Force .................................... 134
  Disable All I/O Forces ........................................... 134
Data Trend (Histogram) ............................................... 135
  Run a Trend for a Tag .......................................... 135
  Add More Tags to the Trend .................................... 136
  Save the Trend .................................................... 137
View Scan Time .......................................................... 138
  View Task Scan Time ............................................. 138
  View Program Scan Time ....................................... 139

Index

Publication 1756-QS001E-EN-P - October 2009
About This Publication

Use this manual to get started programming and maintaining Logix5000 controllers.

This manual describes the necessary tasks to do the following.
- establish communication with a Logix5000 controller
- program a Logix5000 controller
- perform online maintenance tasks such as search and edit logic, run a histogram, clear faults, and force I/O values.

Required Software

To complete this quick start, the following software is required:
- RSLogix 5000 software, version 18 or later
- RSLinx Classic software, version 2.51
## Additional Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logix5000 Controllers System Reference, publication 1756-QR107</td>
<td>Look up abbreviated information and procedures regarding programming languages, instructions, communications, and status</td>
</tr>
<tr>
<td>Logix5000 Controllers Design Considerations Reference, publication 1756-RM094</td>
<td>Design and optimize a controller application.</td>
</tr>
<tr>
<td>Logix5000 Controllers Common Procedures, publication 1756-PM001</td>
<td>Program a Logix5000 controller—detailed and comprehensive information</td>
</tr>
<tr>
<td>- Logix5000 Controllers General Instructions Reference Manual, publication 1756-RM003</td>
<td>Program a specific Logix5000 programming instruction</td>
</tr>
<tr>
<td>- Logix5000 Controllers Process and Drives Instructions Reference Manual, publication 1756-RM006</td>
<td></td>
</tr>
<tr>
<td>- Logix5000 Controllers Motion Instruction Set Reference Manual, publication MOTION-RM001</td>
<td></td>
</tr>
<tr>
<td>Logix5000 Controllers Import/Export Reference Manual, publication 1756-RM084</td>
<td>Import or export a Logix5000 project or tags from or to a text file</td>
</tr>
<tr>
<td>- 1768 CompactLogix Controller Quick Start and User Manual, publication 1768-UM001</td>
<td>Integrate a specific Logix5000 controller within a system of controllers, I/O modules, and other devices</td>
</tr>
<tr>
<td>- 1769 CompactLogix System User Manual, publication 1769-UM007</td>
<td></td>
</tr>
<tr>
<td>- ControlLogix System User Manual, publication 1756-UM001</td>
<td></td>
</tr>
<tr>
<td>- DriveLogix Controller User Manual, publication 20D-UM002</td>
<td></td>
</tr>
<tr>
<td>- GuardLogix Controllers User Manual, publication 1756-UM020</td>
<td></td>
</tr>
<tr>
<td>- SoftLogix5800 System User Manual, publication 1789-UM002</td>
<td></td>
</tr>
<tr>
<td>EtherNet/IP Modules in Logix5000 Control Systems User Manual, publication ENET-UM001</td>
<td>Control devices over an EtherNet/IP network</td>
</tr>
<tr>
<td>ControlNet Modules in Logix5000 Control Systems User Manual, publication CNET-UM001</td>
<td>Control devices over a ControlNet network</td>
</tr>
<tr>
<td>DeviceNet Modules in Logix5000 Control Systems User Manual, publication DNET-UM004</td>
<td>Control devices over a DeviceNet network</td>
</tr>
</tbody>
</table>

You can view or download publications at [http://www.rockwellautomation.com/literature](http://www.rockwellautomation.com/literature). To order paper copies of technical documentation, contact your local Rockwell Automation distributor or sales representative.
Program and Test a Simple Project

This chapter introduces the basic programming sequence for a Logix5000 controller.

- It covers the steps required to develop and test a ladder or function block diagram.
- The examples in the chapter show how to control a digital or analog output based on the state of a digital or analog input.

What You Need

You need these items to complete the tasks in this manual:

- Personal computer running RSLogix 5000 software, version 16 or later
- A layout of the system for which you are creating a project
Chapter 1  Program and Test a Simple Project

Before You Begin

To configure and program a Logix5000 controller, you use RSLogix 5000 software to create and manage a project for the controller. A project is the file on your workstation (or server) that stores the logic, configuration, data, and documentation for a controller.

- The file for the project has an .ACD extension.
- When you create a project, the project name is the same as the name of the controller.
- The controller name is independent of the project name. You can rename either the project name or the controller name.

In an open project, there is this information:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Name of the project. If you rename the project or controller, both names are shown.</td>
</tr>
<tr>
<td>B</td>
<td>Name of the controller.</td>
</tr>
<tr>
<td>C</td>
<td>The controller organizer is a graphical overview of the project. Use the controller organizer to navigate to the various components of a project.</td>
</tr>
</tbody>
</table>

To open a folder and show its contents, either:
- double-click the folder.
- click the + sign.
Follow These Steps

1. Create a project for the controller (page 14).
2. Add I/O modules (page 15).
3. Look at I/O data (page 17).
4. Enter ladder logic (page 19).
5. Enter a function block diagram (page 21).
6. Assign alias tags for your devices (page 25).
7. Establish a serial connection to the controller (page 28).
8. Download a project to the controller (page 30).
9. Select the operating mode of the controller (page 32).
Create a Project for the Controller

1. Start RSLogix 5000 software.

```
<table>
<thead>
<tr>
<th>File</th>
<th>Edit</th>
<th>View</th>
<th>Search</th>
<th>Logic</th>
<th>Communications</th>
<th>Tools</th>
<th>Window</th>
</tr>
</thead>
</table>
```

2. Click New.

3. Specify the general configuration for the controller.

Specify these items (some items apply to only certain controllers):
- Type of controller.
- Major revision of firmware for the controller.
- Name for the controller.
- Chassis type for the controller.
- Slot number of the controller.
- The path where the project will be stored.

4. Click OK.
Conventions for Names

Throughout a Logix5000 project, you define names for the different elements of the project, such as the controller, data addresses (tags), routines, and I/O modules. As you enter names, follow these rules.

- Only letters, numbers, and underscores (_)
- Must start with a letter or an underscore
- ≤ 40 characters
- No consecutive or trailing underscores
- Not case sensitive

Add Your I/O Modules

To communicate with an I/O modules in your system, you add the modules to the I/O Configuration folder of the controller. The properties you select for each module defines the behavior of the module.

TIP

The screens shown are representative of three types of controllers; other types are available, but are not shown here.

1. Right-click the I/O Configuration folder and choose New Module.
2. Select the module and click OK.

3. From the Major Revision pull-down menu, choose the revision of the module.

4. Define the module and click OK.
Look at Your I/O Data

I/O information is presented as a set of tags.

An I/O address follows this format.

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

Where

<table>
<thead>
<tr>
<th>Where</th>
<th>Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>Network location</td>
</tr>
<tr>
<td></td>
<td>LOCAL = same chassis or DIN rail as the controller</td>
</tr>
<tr>
<td></td>
<td>ADAPTER_NAME = identifies remote communication adapter or bridge module</td>
</tr>
<tr>
<td>Slot</td>
<td>Slot number of I/O module in its chassis or DIN rail</td>
</tr>
<tr>
<td>Type</td>
<td>Type of data</td>
</tr>
<tr>
<td></td>
<td>I = input</td>
</tr>
<tr>
<td></td>
<td>O = output</td>
</tr>
<tr>
<td></td>
<td>C = configuration</td>
</tr>
<tr>
<td></td>
<td>S = status</td>
</tr>
<tr>
<td>Member</td>
<td>Specific data from the I/O module; depends on what type of data the module can store.</td>
</tr>
<tr>
<td></td>
<td>• For a digital module, a Data member usually stores the input or output bit values.</td>
</tr>
<tr>
<td></td>
<td>• For an analog module, a Channel member (CH#) usually stores the data for a channel.</td>
</tr>
<tr>
<td>SubMember</td>
<td>Specific data related to a Member.</td>
</tr>
<tr>
<td>Bit</td>
<td>Specific point on a digital I/O module; depends on the size of the I/O module (0-31 for a 32-point module)</td>
</tr>
</tbody>
</table>
1. Right-click Controller Tags and choose Monitor Tags.

The Tag Monitor displays the tags.

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Value</th>
<th>Force Mask</th>
<th>Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local:0:C</td>
<td>(...)</td>
<td>(...)</td>
<td></td>
</tr>
<tr>
<td>Local:0:1</td>
<td>(...)</td>
<td>(...)</td>
<td></td>
</tr>
<tr>
<td>Local:0:0</td>
<td>(...)</td>
<td>(...)</td>
<td></td>
</tr>
<tr>
<td>Local:0:0.Data</td>
<td>2#000...</td>
<td>Binary</td>
<td></td>
</tr>
<tr>
<td>- Local:0:0.Data.0</td>
<td>0</td>
<td>Decimal</td>
<td></td>
</tr>
<tr>
<td>- Local:0:0.Data.1</td>
<td>0</td>
<td>Decimal</td>
<td></td>
</tr>
<tr>
<td>- Local:0:0.Data.2</td>
<td>0</td>
<td>Decimal</td>
<td></td>
</tr>
</tbody>
</table>

Values are shown in the following styles.

<table>
<thead>
<tr>
<th>Style</th>
<th>Base</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>2</td>
<td>2#</td>
</tr>
<tr>
<td>Decimal</td>
<td>10</td>
<td>NA</td>
</tr>
<tr>
<td>Hexadecimal</td>
<td>16</td>
<td>16#</td>
</tr>
<tr>
<td>Octal</td>
<td>8</td>
<td>8#</td>
</tr>
<tr>
<td>Exponential</td>
<td>NA</td>
<td>0.00000000e+000</td>
</tr>
<tr>
<td>Float</td>
<td>NA</td>
<td>0.0</td>
</tr>
</tbody>
</table>

A blue arrow indicates that when you change the value, it immediately takes effect.

2. To see a value in a different style, select the desired style.

3. To change a value, click the Value cell, type the new value, and click Enter.

4. To expand a tag and show its members, click the + sign.
Ladder Logic

For a Logix5000 controller, you enter your logic in routines.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A routine provides the executable code (logic) for a program (similar to a program file in a PLC or SLC controller).</td>
</tr>
</tbody>
</table>
| B    | There is one main routine you assign for each program.  
- When the program executes, its main routine automatically executes.  
- Use the main routine to control the execution of the other routines in the program.  
- To call (execute) another routine (subroutine) within the program, use a Jump to Subroutine (JSR) instruction. |
| C    | A subroutine is any routine other than the main routine or fault routine. To execute a subroutine, use a Jump to Subroutine (JSR) instruction in another routine, such as the main routine. |

When you create a project, the software automatically creates a main routine that uses the ladder diagram programming language.
Enter Ladder Logic

One way to enter logic is to drag buttons from a toolbar to the desired location.

To add ladder logic, drag the button for the rung or instruction directly to the desired location. You can enter your logic and leave the operands undefined. After you enter a section of logic, go back and assign the operands.

**EXAMPLE**

In the following example, an Examine If Closed (XIC) instruction checks the on/off state of a pushbutton. If the pushbutton is on, the Output Energize (OTE) instruction turns on a light.

A green dot shows a valid placement location (drop point).
Enter a Function Block Diagram

Follow these steps to add a function block diagram to your project.

Create a Routine

Each routine in your project uses a specific programming language. To program in a different language, such as function block diagram, create a new routine.

1. Right-click MainProgram and choose New Routine.

2. Type a name for the routine.

3. Choose the programming language.

4. Click OK.
Call the Routine

To execute a routine other than the main routine, use a Jump to Subroutine (JSR) instruction to call the routine.

1. Add a rung.

![Image of a rung with JSR instruction]

2. On the Program Control tab, add a JSR instruction.

3. In the Routine Name field of the JSR instruction, type the name of the routine that you want to execute.

4. To simply call the routine, remove the rest of the parameters for the JSR instruction. To remove a parameter, right-click the parameter and choose Remove Instruction Parameter.

![Image of removing instruction parameter]
Enter a Function Block Diagram

Enter function block diagram instructions in a function block routine.

1. Click the tab for the desired instructions.

2. Drag elements from the toolbar to the sheet.

3. To connect elements, click corresponding pins (green dot = valid connection point).

**EXAMPLE** In the following example, an Input Reference (IREF) reads the value of an analog input and sends the value to a Scale (SCL) instruction. The SCL instruction converts the value to engineering uses and sends it to an Output Reference (OREF). The OREF writes the value to an analog output.
Configure a Function Block Instruction

Assign specific values (parameters) to configure a function block instruction.

1. Click the configuration button.

2. To change the value of a parameter, click the value cell, type the new value, and click Enter.
For example, in the SCL instruction, specify the following parameters:

- InRawMax – maximum input value
- InRawMin – minimum input value
- InEUMax – maximum engineering value
- InEUMin – minimum engineering value

3. Click OK.

Assign Alias Tags for Your Devices

While you can use the input and output tags of a module directly in your logic, it is easier to use alias tags. An alias tag is a tag that represents another tag.

- Both tags share the same data.
- When the data changes, both tags change.
- An alias tag provides a descriptive name for data, such as DeviceNet input or output data.
- If the location of the data changes, simply point the alias tag goes to the new location without editing your logic.

As an option, create tags that describe each device without pointing them to the actual addresses of the devices. Later, convert the tags to aliases for the data of the devices.

1. Enter the logic.

2. Type a descriptive tag name for the device.

3. Right-click the tag name and choose New…
4. Select Alias from the menu.

5. Select the tag that this alias tag represents.

6. Select the scope for the alias tag.

7. Click OK.

8. Select the address of the data.

To select a bit, click the button.
Show or Hide Alias Information

Show or hide alias information for a tag.

1. From the Tools menu, choose Options.

2. Select the Ladder Editor Display category.

3. Check or uncheck Show Tag Alias Information.

4. Click OK.
Establish a Serial Connection to the Controller

RSLinx Classic software handles communication between Logix5000 controllers and your software programs, such as RSLogix 5000 software. To communicate with a controller (for example, download, monitor data), configure RSLinx Classic software for the required communication.

Use a serial cable to establish a point-to-point connection between the serial ports on your computer and controller.

**WARNING**

If you connect or disconnect the serial cable with power applied to this module or the serial device on the other end of the cable, 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.

1. Connect a serial cable to your controller and computer.
2. Start RSLinx Classic software.

3. Click .

4. Select RS-232 DF1 devices and click .

5. From the Comm Port pull-down menu, choose the COM port of your computer.

6. From the Device pull-down menu, choose Logix 5550/CompactLogix.

7. Click .

8. When the auto-configuration completes, click OK.

The driver is successfully configured and running.
Download a Project to the Controller

To execute a project in a controller, download the project to the controller to transfer a project from your computer to the controller so you can run the project. When you download a project, you lose the project and data that is currently in the controller, if any. If the revision of the controller does not match the revision of the project, you are prompted to update the firmware of the controller. RSLogix 5000 software lets you update the firmware of the controller as part of the download sequence.

ATTENTION When you download a project or update firmware, all active servo axes are turned off. Before you download a project or update firmware, make sure that this will not cause any unexpected movement of an axis.

IMPORTANT To update the firmware of a controller, first install a firmware upgrade kit.
- An upgrade kit ships on a supplemental CD along with RSLogix 5000 software.

1. Turn the keyswitch of the controller to .
2. Open the RSLogix 5000 project that you want to download.
3. Click .
4. Browse to the controller.

5. To download the project, click [Download].

If the process failed to download to the controller. The revision of the offline project and controller’s firmware are not compatible.

1. Choose [Update Firmware...].

2. Select the revision for the controller.

3. Choose [Update...] and then [Yes].
Select the Operating Mode of the Controller

To execute or stop executing the logic in a controller, change the operating mode of the controller.

1. Determine which mode you want for the controller.
   - Do you want to execute the logic in the controller? No → Choose Program mode.
   - Yes → Do you want the logic to control the output devices? Yes → Choose Run mode.
   - No → Choose Test mode.

2. Turn the keyswitch to RUN REM PROG.

3. Go online with the controller.

4. Select the mode.
<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program</td>
<td>Logic is not executing, outputs are not controlled, and editing operations are available. If you are configuring an output module, the owner controller is in Program Mode. Newly received output values are ignored and all outputs will transition to their Program mode state (which you can configure on the Configuration tab). The output module’s health LED will flash green when in Program mode. Input modules are always in Run mode and always report back input data to the controller. It does not matter whether the owner controller is in Run or Program mode. The input module’s health indicator is always solid green if a connection exists to it.</td>
</tr>
<tr>
<td>Run</td>
<td>Logic is executing, inputs are read, logic is scanned, and outputs are controlled by the application program and changes made through the data monitor or the I/O force table. The actual I/O modules accept the output results of the application and set the outputs accordingly. The keyswitch must be in the Remote or Run position.</td>
</tr>
</tbody>
</table>
| Test  | Logic is executing, inputs are read, logic is scanned, and outputs are controlled by the application program and changes made through the data monitor or the I/O force table. The actual I/O modules will ignore the output results of the application. Some editing operations are restricted. The keyswitch must be in the Remote position. When going into Test mode:  
  - Input modules continue to update in Test mode.  
  - Produce/consume tags continue to update in Test mode.  
  - Test mode places all outputs in the project in the Program mode state (as configured in the Configuration tab for module properties). |
Notes:
Organize a Project

This chapter provides more detailed information on how to organize the program layout and data structures for the controller.

What You Need

You need these items to complete the tasks in this manual:

- Personal computer running RSLogix 5000 software, version 16 or later
- A layout of the system for which you are creating a project

Before You Begin

A new project contains a default task for the execution of your logic. Before you can create programs, you must first configure the task execution. A task defines scheduling and priority information for the execution (scan) of your logic.
Follow These Steps

1. Configure the task execution (page 36).
2. Create additional programs (page 38).
3. Create user-defined data types (page 40).
4. Define your routines (page 43).
5. Assign main routines (page 47).
6. Configure the controller (page 48).
7. Configure I/O modules (page 49).

Configure the Task Execution

In this quick start, we limit the project to a single task with one of the following types of execution.

<table>
<thead>
<tr>
<th>If you want to execute your logic</th>
<th>Then configure the task for this type of execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>All of the time</td>
<td>Continuous [This is the default configuration of MainTask.]</td>
</tr>
<tr>
<td>Execution of Logic</td>
<td></td>
</tr>
<tr>
<td>task automatically restarts</td>
<td></td>
</tr>
<tr>
<td>task automatically restarts</td>
<td></td>
</tr>
<tr>
<td>task automatically restarts</td>
<td></td>
</tr>
<tr>
<td>task automatically restarts</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>At a specific period</th>
<th>Periodic [You define the period at which the task executes.]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution of Logic</td>
<td></td>
</tr>
<tr>
<td>task finishes</td>
<td></td>
</tr>
<tr>
<td>period expires</td>
<td></td>
</tr>
<tr>
<td>task finishes</td>
<td></td>
</tr>
<tr>
<td>period expires</td>
<td></td>
</tr>
<tr>
<td>task restarts</td>
<td></td>
</tr>
<tr>
<td>task restarts</td>
<td></td>
</tr>
</tbody>
</table>
1. Right-click MainTask and choose Properties.

2. Click the Configuration tab.

3. From the Type pull-down menu, choose Periodic.

4. Type the period for the task and click OK.

To use multiple tasks or execute a task when a specific event (trigger) occurs, see Logix5000 Controllers Common Procedures, publication 1756-PM001.
Create Additional Programs

A Logix5000 controller lets you divide your application into multiple programs, each with its own tags (data).

There is no need to manage conflicting tag names between the programs.

All programs have access to data that is at the controller scope. Data at the program scope is isolated from other programs.

- Routines cannot access data that is at the program scope of another program.
- You can re-use the tag name of a program-scoped tag in multiple programs.
- For example, both Program_A and Program_B can have a program tag named Tag_4.
If you have multiple machines, stations, or processes that use identical logic but different data, create a program for each machine, station, or process.

- You can re-use both code and tag names in the programs.
- There is no need to manage conflicting tag names between the programs.

**1.** Right-click MainTask and choose New Program.

**2.** Type a name for the program and click OK.

**TIP**

Names follow these conventions:
- Only letters, numbers, and underscores (_)
- Must start with a letter or an underscore
- ≤ 40 characters
- No consecutive or trailing underscores
- Not case-sensitive

Certain tags must be controller scope.

<table>
<thead>
<tr>
<th>If you want to use a tag</th>
<th>Use this scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>In more than one program in the project</td>
<td>Controller Tags</td>
</tr>
<tr>
<td>In a Message (MSG) instruction</td>
<td></td>
</tr>
<tr>
<td>To produce or consume data</td>
<td></td>
</tr>
<tr>
<td>To communicate with a PanelView terminal</td>
<td></td>
</tr>
<tr>
<td>In a single program only</td>
<td>Program Tags for the program</td>
</tr>
</tbody>
</table>
Create User-defined Data Types

User-defined data types let you organize your data to match your machine or process. This streamlines program development and creates self-documenting code that is easier to maintain.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Tag stores data. There is no fixed data table or numeric format for data addresses. The tag name is the address. You create the tags that you want to use.</td>
</tr>
<tr>
<td>B</td>
<td>Data type defines the type of data that a tag stores, such as a bit, integer, floating-point value, or string.</td>
</tr>
<tr>
<td>C</td>
<td>Array defines a block of data (file). The entire block uses the same data type. It can have 1, 2, or 3 dimensions.</td>
</tr>
<tr>
<td>D</td>
<td>Structure combines a group of data types into a re-usable format (template for tags). Use a structure as the basis for multiple tags with the same data layout.</td>
</tr>
<tr>
<td>E</td>
<td>Member describes an individual piece of data within a structure.</td>
</tr>
<tr>
<td>F</td>
<td>User-defined data type defines your own data structure. A user-defined data type stores all the data related to a specific aspect of your system. This keeps related data together and easy to locate, regardless of its data type.</td>
</tr>
</tbody>
</table>
As you create user-defined data types, follow these guidelines.

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consider the pass-through of descriptions.</td>
<td>See User-defined Data Type on page 98.</td>
</tr>
<tr>
<td>Data that represents an I/O device requires additional programming.</td>
<td>If you include members that represent I/O devices, you must use logic to copy the data between the members in the user-defined data type and the corresponding I/O tags.</td>
</tr>
<tr>
<td>If you include an array as a member, limit the array to a single dimension.</td>
<td>Multi-dimension arrays are not permitted in a user-defined data type.</td>
</tr>
<tr>
<td>When you use the BOOL, SINT, or INT data types, place members that use the same data type in sequence:</td>
<td>Logix5000 controllers allocate memory in 4-byte chunks. If you sequence smaller data types together, the controller packs as many as it can fit into a 4-byte chunk.</td>
</tr>
</tbody>
</table>

Follow these steps to create a user-defined data type.

1. Right-click Data Types and choose New Data Type.

2. Type a name for the data type (not the name of a tag that will use the data type).
3. Enter the members.

As an option, type a description for each member.

4. Click OK.

Follow these steps to create a tag that uses a user-defined data type.

1. Right-click the scope that you want for the tag and choose Edit Tags.

2. Type a name for the tag.

3. Type the name of the user-defined data type.
4. Do the following if you want the tag to be an array (multiple instances of the data type).
   a. Select the data type and click [button].
   b. Specify the array dimensions and click OK.

Define Your Routines

Once your project has the required programs, you have to define and create the routines for each program.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Routine provides the executable code (logic) for a program (similar to a program file in a PLC or SLC controller).</td>
</tr>
</tbody>
</table>
| B    | Main routine is required each program.  
  • When the program executes, its main routine automatically executes.  
  • Use the main routine to control the execution of the other routines in the program.  
  • To call (execute) another routine (subroutine) within the program, use a Jump to Subroutine (JSR) instruction. |
| C    | Subroutine is any routine other than the main routine or fault routine. To execute a subroutine, use a Jump to Subroutine (JSR) instruction in another routine, such as the main routine. |
Chapter 2  Organize a Project

Define a Routine for Each Section of Your Machine or Process

To make your project easier to develop, test, and troubleshoot, divide it into routines (subroutines).

1. Identify each physical section of your machine or process.

2. Assign a routine for each of those sections.

Description of Your Machine or Process

First Section = Routine 1
Second Section = Routine 2
Third Section = Routine 3

Identify the Programming Languages That Are Installed

Follow these steps to determine which programming languages are installed on your version of RSLogix 5000 software.

1. Start RSLogix 5000 software.

2. From the Help menu, choose About RSLogix 5000.
Assign a Programming Language to Each Routine

For each routine, choose a programming language.
- Logix5000 controllers let you use the following languages:
  - Ladder logic
  - Function block diagram
  - Sequential function chart
  - Structured text
- Use any combination of the languages in the same project.

<table>
<thead>
<tr>
<th>In general, if a routine represents</th>
<th>Use this language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous or parallel execution of multiple operations (not sequenced)</td>
<td>Ladder logic</td>
</tr>
<tr>
<td>Boolean or bit-based operations</td>
<td></td>
</tr>
<tr>
<td>Complex Logical operations</td>
<td></td>
</tr>
<tr>
<td>Message And Communication Processing</td>
<td></td>
</tr>
<tr>
<td>Machine interlocking</td>
<td></td>
</tr>
<tr>
<td>Operations that service or maintenance personnel may have to interpret in order to troubleshoot the machine or process.</td>
<td></td>
</tr>
<tr>
<td>Continuous process and drive control</td>
<td>Function block diagram (FBD)</td>
</tr>
<tr>
<td>Loop control</td>
<td></td>
</tr>
<tr>
<td>Calculations in circuit flow</td>
<td></td>
</tr>
<tr>
<td>High-level management of multiple operations</td>
<td>Sequential function chart (SFC)</td>
</tr>
<tr>
<td>Repetitive sequences of operations</td>
<td></td>
</tr>
<tr>
<td>Batch process</td>
<td></td>
</tr>
<tr>
<td>Motion control using structured text</td>
<td></td>
</tr>
<tr>
<td>State machine operations</td>
<td></td>
</tr>
<tr>
<td>Complex mathematical operations</td>
<td>Structured text</td>
</tr>
<tr>
<td>Specialized array or table loop processing</td>
<td></td>
</tr>
<tr>
<td>ASCII string handling or protocol processing</td>
<td></td>
</tr>
</tbody>
</table>
## Divide Each Routine Into More Meaningful Increments

<table>
<thead>
<tr>
<th>If a routine uses this language</th>
<th>Then</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ladder logic</td>
<td>Break up large routines into several smaller routines</td>
<td>To continuously execute several complex boolean operations… …create a separate routine for each operation.</td>
</tr>
<tr>
<td>Structured text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function block diagram (FBD)</td>
<td>Within the FBD routine, make a sheet for each functional loop for a device, such as a motor or valve.</td>
<td>To control 4 valves, where each valve requires feedback that it is in its commanded position… …make a separate sheet for each valve.</td>
</tr>
<tr>
<td>Sequential function chart (SFC)</td>
<td>Break the SFC into steps.</td>
<td>To perform the following sequence: 1. Fill a tank. 2. Mix the ingredients in the tank. 3. Empty the tank… …make each section (fill, mix, empty) a separate step.</td>
</tr>
</tbody>
</table>
Assign Main Routines

Each program requires a main routine. Once you create your routines, assign a main routine for each program.

**IMPORTANT** In the default project, MainProgram already has a main routine (MainRoutine). You have to assign a main routine only for each additional program that you create.

Follow these steps to assign a main routine.

1. Right-click the program folder and choose Properties.

2. Click the Configuration tab.

3. Select the main routine and click OK.
Chapter 2  Organize a Project

Configure the Controller

If you want to change the configuration of the controller, such as name, chassis size, or slot number, use the Controller Properties dialog box.

1. Click the Controller Properties button.

2. Change the required properties (some items apply only to certain controllers) and click OK.
Configure I/O Modules

To change the behavior of a module, use the Module Properties window for the module. The configuration options vary from module to module.

1. Right-click the module and choose Properties.

2. To change the name or slot number, use the General tab.

3. To change the configuration, click the Configuration tab.

Some modules have several configuration tabs.
Notes:
Program Add-On Instructions

With version 16 of RSLogix 5000 programming software, you can design and configure sets of commonly used instructions to increase project consistency. Similar to the built-in instructions contained in Logix5000 controllers, these instructions you create are called Add-On Instructions.

With Add-On Instructions, you can:
- insert your own instruction.
- copy an Add-On Instruction definition from another RSLogix 5000 project.
- import an Add-On Instruction definition from another RSLogix 5000 project.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 software, version 16 or later
- A layout of the system for which you are creating a project

Follow These Steps

4. Insert an Add-On Instruction (page 52).
5. Copy an Add-On Instruction definition (page 53).
6. Import an Add-On Instruction definition (page 54).
7. Access a parameter that is not visible (page 55).
8. Monitor or change the value of a parameter (page 56).
9. View the logic of an Add-On Instruction (page 57).
11. Update an Add-On Instruction to a newer revision (page 58).
## Insert an Add-On Instruction

You can create an Add-On Instruction in a ladder, function block, or structured text routine.

1. Open the routine that will use the instruction.

2. Click the Add-On tab of the Language Element toolbar.

3. Drag the instruction from the toolbar to the routine.

4. Enter the parameters.

<table>
<thead>
<tr>
<th>Routine Type</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ladder</strong></td>
<td>Single question mark — Required parameter. Enter a tag.</td>
</tr>
<tr>
<td></td>
<td>Single and double question marks — Required parameter. Enter a tag.</td>
</tr>
<tr>
<td></td>
<td>Only double question marks — Not a required parameter. You can either:</td>
</tr>
<tr>
<td></td>
<td>• leave it blank and use the default value.</td>
</tr>
<tr>
<td></td>
<td>• if it's an input value, enter a different value.</td>
</tr>
<tr>
<td></td>
<td>Nub on the end of a pin — Required parameter. Wire the pin to an IREF, OREF, connector, or another block.</td>
</tr>
<tr>
<td></td>
<td>Single question mark — Required parameter. Enter a tag.</td>
</tr>
<tr>
<td></td>
<td>No nub on the end of a pin — Not a required parameter.</td>
</tr>
<tr>
<td><strong>Structured text</strong></td>
<td>The instruction takes only the required parameters. Enter a tag for each parameter.</td>
</tr>
</tbody>
</table>

**TIP** For help with an instruction, select the instruction and then press [F1]. In structured text, make sure the cursor is in the blue instruction name.
Copy an Add-On Instruction Definition

You can copy an Add-On Instruction from within another RSLogix 5000 project.

1. Open the RSLogix 5000 project that has the Add-On Instruction definition.

2. Find the definition in the Add-On Instructions folder.

3. Right-click the definition and choose Copy.

4. Go to the project that gets the definition.

5. Right-click the Add-On Instructions folder and choose Paste.
**Import an Add-On Instruction Definition**

You can add the definition of an Add-On Instruction that was exported from another RSLogix 5000 project.

Does the RSLogix 5000 project already have a revision of this Add-On Instruction?
- No — use this procedure to import the instruction.
- Yes — see Update an Add-On Instruction to a Newer Revision on page 58

1. Right-click the Add-On Instructions folder and choose Import Add-On Instruction.

2. Find the instruction.

3. Select the instruction and click Import.
Access a Parameter That Is Not Visible

How you read or write to a parameter of an Add-On Instruction that is not visible depends on the programming language.

**Function Block**

1. Click the Properties button for the instruction.

2. Check the Vis box for the parameter and click OK.

3. Wire to the pin for the parameter.

**Ladder Logic and Structured Text**

Use another instruction, an assignment, or an expression to read or write to the tag name of the parameter. Use this format for the tag name of the parameter.

\[ \text{Add}_\text{On}_\text{Tag} . \text{Parameter} \]

<table>
<thead>
<tr>
<th>Where</th>
<th>Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add_On_Tag</td>
<td>Add-on-defined tag for the add-on instruction</td>
</tr>
<tr>
<td>Parameter</td>
<td>Name of the parameter</td>
</tr>
</tbody>
</table>
Monitor or Change the Value of a Parameter of an Add-On Instruction

Use the Properties dialog to monitor or change a parameter value of an Add-On Instruction.

<table>
<thead>
<tr>
<th>Routine Type</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ladder logic or function block</td>
<td>Click the Properties button for the instruction.</td>
</tr>
<tr>
<td>Structured text</td>
<td>Right-click the instruction and choose Properties.</td>
</tr>
</tbody>
</table>

1. Click and type the new value.

2. Click Apply.

3. Click OK.
**View the Logic of an Add-On Instruction**

It’s possible to protect an Add-On Instruction so that you can’t see its logic. You can determine whether an Add-On Instruction is protected.

1. Select the add-on instruction.

2. Look in the Quick View pane for Source Protection.

   If it isn’t listed, then the routine isn’t protected.

To view the logic, right-click the instruction and choose Open Instruction Logic.

```plaintext
Motor_Starter(Motor_Starter_ST, Stop_PB, Start_PB, Motor_Out_ST);
```
Edit and Monitor an Add-On Instruction

You can:

- see the logic as it executes.
- see tag values.
- change tag and parameter values.

You cannot:

- edit logic online.
- edit logic for just this instruction.

To edit the logic, you must edit the definition.

Update an Add-On Instruction to a Newer Revision

You can update the definition of an Add-On Instruction to a newer revision.

IMPORTANT Before you change the definition of an add-on instruction, make sure the change won’t cause problems with existing instances of that instruction. When you change the definition of an add-on instruction, the change affects all the instances of that instruction in your project.

For example, if a project uses a certain Add-On Instruction 5 times, update the definition so that all 5 instances change when you change the definition.

1. Right-click the Add-On Instructions folder and choose Import Add-On Instruction.
2. Find the instruction and choose Import.
3. Decide how to handle the conflict with the existing revision (probably overwrite).
4. Use a cross-reference list to check each use of the instruction in your logic.
Program an Equipment Phase

Use PhaseManager software to create an equipment phase and change the default settings for the equipment phase.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 software, version 16 or later
- A layout of the system for which you are creating a project

Follow These Steps

5. Create an equipment phase (page 60).
6. Create a state routine (page 60).
7. Manually step through the states (page 61).
8. Configure the initial state for an equipment phase (page 63).
9. Open the configuration for an equipment phase (page 64).
10. Configure an equipment phase (page 64).
Create an Equipment Phase

1. Right-click Main Task and choose New Equipment Phase.

![New Equipment Phase](image)

2. Type a name for the equipment phase and click OK.

Create a State Routine

1. Right-click Main Task and choose the equipment phase.

2. Choose New Phase State Routine.

3. Type a name for the state routine.

4. Select the programming language and click OK.

![New Equipment Phase State Routine](image)
Manually Step Through the States

Before you step through states, do the following:

- Download the project to the controller.
- Put the controller in run or remote run mode.

1. Right-click the equipment phase and choose Monitor Equipment Phase.

2. Click the Ownership and then Yes.

Use this window to step through the states.

3. Click Start.

The equipment phase goes to the Running state. Any code in the Running state routine starts running. This is where you put the code for the normal production sequence of your equipment.
4. Click Stop.

The equipment phase goes to the Stopped state. The Running state routine stops running. The Stopping state routine is optional. Without it, the equipment phase goes directly to the Stopped state.

5. Click Reset.

The equipment phase goes to the Idle state. The Resetting state routine is optional. Without it, the equipment phase goes directly to the Idle state.
6. Click Ownership.

This releases the equipment phase from control by this window.

**Configure the Initial State for an Equipment Phase**

The initial state is the first state to which the equipment phase goes after you apply power.

1. Right-click the equipment phase and choose Properties.

2. Choose the Configuration tab.

3. Choose the Initial State and click OK.
Open the Configuration for an Equipment Phase

1. Right-click the equipment phase and choose Properties.

2. Click the Configuration tab.

Configure an Equipment Phase

Use the following settings to configure an equipment phase.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Choices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prestate</td>
<td>prestate routine</td>
</tr>
</tbody>
</table>

The prestate routine runs all the time, even when the equipment phase is in the idle state. It runs before each scan of a state.

Do you want to run a prestate routine?

- Yes — Select the routine that you want to run.
- No — Leave this box set to <none>
### Setting | Choices
--- | ---
Fault | A fault routine lets you clear a major fault made by an instruction.  
Do you want to set up a fault routine for the instructions in this equipment phase?  
- Yes — Select the routine that you want as your fault routine.  
- No — Leave this box set to <none>
Inhibit Equipment Phase | Do you want the controller to inhibit this equipment phase?  
- Yes — Check this box.  
- No — Leave this box unchecked or uncheck it.
Initial State | Which state do you want the equipment phase to go to when you turn on the controller?  
- Idle  
- Complete  
- Stopped  
- Aborted
Complete State Immediately If not Implemented | Do you want the equipment phase to skip any states that you aren’t using?  
- Yes — Leave this box checked or check it.  
- No — Uncheck this box.
Initial Step Index | Are any of the state routines in ladder diagram or structured text?  
- No — Skip this box.  
- Yes — Go to the next question.  
Do any of those state routines use step numbers?  
- Yes — Type the number for the first step of each state.  
- No — Skip this box.  
The tag for the equipment phase has a SteplIndex number. The controller resets the SteplIndex each time the equipment phase changes states. The controller resets the SteplIndex to the number you put in the Initial Step Index box.
External Sequencer Loss of Communication Command | Are you using FactoryTalk Batch software to command this equipment phase?  
- No — Skip this box.  
- Yes — Go to the next question.  
If the controller loses communication with FactoryTalk Batch software, what do you want the equipment phase to do?  
- Continue in its current state — Select None.  
- Go to aborting — Select Abort.  
- Go to holding — Select Hold.  
- Go to stopping — Select Stop.  
The equipment phase must still follow the state model. For example, it goes to holding only if it is in running or restarting when communication fails.
### Setting: External Request Hold Action

<table>
<thead>
<tr>
<th>Choices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are you using any PXRQ instructions?</td>
</tr>
<tr>
<td>• No — Skip this box.</td>
</tr>
<tr>
<td>• Yes — Go to the next question.</td>
</tr>
<tr>
<td>What do you want to do if an equipment phase goes to holding while a PXRQ instruction is in process?</td>
</tr>
<tr>
<td>• Nothing — Select None.</td>
</tr>
<tr>
<td>• Stop the request — Select Clear.</td>
</tr>
</tbody>
</table>
Program a Project Offline

This chapter provides more detailed information on how to program the logic for a routine and create tags for the logic.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 Software, version 16
- A plan for the project you are programming

Before You Begin

In this chapter, you program the project while offline. Online programming requires additional steps. See chapter Chapter 8, Program a Project Online.

Follow These Steps

1. Enter ladder logic (page 68).
2. Export/import ladder logic (page 73).
3. Enter a function block diagram (page 77).
4. Use a faceplate for a function block (page 81).
5. Enter structured text (page 84).
6. Enter a sequential function chart (page 87).
7. Assign operands (page 89).
8. Verify a project (page 92).
Enter Ladder Logic

To enter ladder logic, you have the following options:

- **Quick keys** – Assign a logic element (rung, branch, instruction) to a keyboard key. To add an element to the right or below the cursor, press the designated key for the element.

- **ASCII text** – Use ASCII text to enter or edit logic. A tool tip helps you enter the required operands. ASCII text typically uses the following format:
  
  \[\text{mnemonic operand}_1 \text{ operand}_2\]

- **Drag and drop logic elements** – Use the Language Element toolbar to drag and drop a rung, branch, or instruction to your routine.

- **ASCII text** – Use ASCII text to enter or edit logic. A tool tip helps you enter the required operands. ASCII text typically uses the following format:
  
  \[\text{mnemonic operand}_1 \text{ operand}_2\]

- **Quick keys** – Assign a logic element (rung, branch, instruction) to a keyboard key. To add an element to the right or below the cursor, press the designated key for the element.

- **Outputs in series** – Place multiple output instructions in sequence (serial) on a rung.

- **Interface input and output instructions** – The last instruction on the rung must be an output instruction.

- **Parallel branches** – No limit to the number of parallel branches on a rung (nest up to 6 levels).

- **Leave operands undefined** – enter logic without defining operands. RSLogix 5000 software lets you enter and save logic without assigning operands. This lets you develop your logic in iterations and save libraries of code for re-use.
Add a Rung or an Instruction

Drag the button for the rung or instruction directly to the desired location. A green dot shows a valid placement location (drop point).

Add a Branch

1. Drag the branch button to where the branch starts.

A green dot shows a valid placement location (drop point).

2. Drag a branch rail to the desired location.

Add a Level to a Branch

Right-click the branch and choose Add Branch Level.
Delete an Element

1. Click the element.
2. Click Delete.

Use the Keyboard to Add an Element

You can add elements by using the computer keyboard.

1. Press Insert.
2. Type the mnemonic for the instruction or type Rung, Branch, or Branch Level.
3. Press Enter.
4. To move an instruction, branch, or rung to a different location, use the mouse to drag it there. A green dot shows a valid placement location (drop point).

Enter Logic Using ASCII Text

You can add elements by using the typing their ASCII equivalents.

1. Double-click the rung.

2. Enter the ASCII text for the rung.
Enable Quick Keys

Enable quick keys to use shortcuts when editing.

1. From the Tools menu, choose Options.
2. Click Ladder Editor.
3. Check these checkboxes.

To assign a key to an element:

1. Click Configure....
2. For the desired key, select the element.
3. When you have assigned the desired keys, click Close.
Export/Import Ladder Logic

If you want to re-use ladder logic from another project, simply export the logic to an L5X file and import it into the required project. The L5X file contains all that you need for the logic except I/O modules.

### When You Import Rungs

When you import rungs, RSLogix 5000 software shows a list of the tags and user-defined data types that go along with the rungs. Use the list to manage the tags and data types that are created during the import operation.

If desired, you can rename a tag to make it fit the project better.

If a tag already exists in the project, you can either:
- use the existing tag, which discards the tag in the library file and binds the logic to the existing tag.
- rename the tag, which creates a new one.

No new I/O tags are created.

If an I/O tag already exists in the project, the import operation uses this tag for any aliases to that tag name. Once you import a project, make sure you check the alias tags for accuracy.
Export Rungs

1. Select the rungs to export.

2. Right-click the selection and choose Export Rung.

3. Choose a location and name for the file.

4. Create the file.
Import Rungs

1. Right-click the location for the rungs and choose Import Rung.

![Import Rung dialog box]

2. Select the file to import and click Import.

![Import Configuration dialog box]

3. Check for conflicts in names.

4. Click OK.
Check Alias Tags

Check the alias tags in the rungs you import.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>If you import an alias tag, make sure it points to the correct base tag. When a tag is an alias for a tag that already exists in the project, the software sets up the relationship between the alias and base tags.</td>
</tr>
<tr>
<td>B</td>
<td>If the project does not have the base tag, you have to either create the base tag or point the alias to a different base tag.</td>
</tr>
</tbody>
</table>
Enter a Function Block Diagram

A function block diagram lets you visually define the flow of data between instructions. The data flow then drives the execution order of the instructions.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Sheet divides the diagram into sections without affecting execution. When the routine executes, all sheets execute.</td>
</tr>
<tr>
<td>B</td>
<td>Input reference (IREF) reads a value from a tag or set a constant value</td>
</tr>
<tr>
<td>C</td>
<td>Wire transfers a value between elements</td>
</tr>
<tr>
<td>D</td>
<td>Instruction produces output values based on input values</td>
</tr>
<tr>
<td>E</td>
<td>Output reference (OREF) writes a value to a tag</td>
</tr>
<tr>
<td>F</td>
<td>Output wire connector (OCON) writes a value to one or more ICONs</td>
</tr>
<tr>
<td>G</td>
<td>Input wire connector (ICON) reads a value from an OCON on the same sheet or a different sheet in the routine. To read a value from another routine, use an OREF/IREF combination.</td>
</tr>
</tbody>
</table>
Chapter 5  Program a Project Offline

Use the Keyboard to Add an Element

You can add function block elements by using the computer keyboard.

1. Press Insert.

2. Type the mnemonic for the element and press Enter.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>If a group of blocks are in a loop, you have to identify which block to execute first. Use the Assume Data Available indicator to mark the input wire that creates the loop (the feedback wire).</td>
</tr>
<tr>
<td></td>
<td>This input pin uses the output that block 3 produced on the previous scan.</td>
</tr>
</tbody>
</table>
3. Drag the element to the desired location.

**Connect Elements**

To connect elements, click corresponding pins (green dot = valid connection point).
**Resolve a Loop**

To resolve a loop (define a wire as an input), right-click the wire and choose Assume Data Available.

**Add Sheet**

Add sheets as needed to a function block diagram.

1. Click New Sheet.

2. Type a name for the sheet.
Use a Faceplate for a Function Block

RSLogix 5000 software includes faceplates (controls) for some of the function block instructions. A faceplate is an Active-X control that lets you interact with a function block instruction.

- Your RSLogix 5000 software package includes the faceplates but does not automatically install them. To use the faceplates, locate them on your software CD and install them separately.
- Use faceplates in an Active-X container, such as the following software:
  - FactoryTalk View SE
  - Microsoft Excel
- RSLogix 5000 software is not a valid Active-X container.
- Faceplates communicate with the controller via DDE/OPC topics in RSLinx Classic software. To use RSLinx Classic software for DDE/OPC topics, purchase either:
  - RSLinx Classic software as a separate package
  - RSLogix 5000 professional edition software, which includes RSLinx Classic professional edition software

RSLinx Classic Lite software, which comes with the other RSLogix 5000 software packages, does not provide DDE/OPC communication.
In RSLinx Classic software, a topic represents a specific path to a controller. RSLogix 5000 software, revision 10.0 or later, automatically creates an RSLinx topic whenever you:

- create a project.
- save a project.
- change the revision of a project to 10.0 or later.

In some cases, you have to update the data source for the topic in RSLinx software.

**Set Up a Topic**

1. Use RSLogix 5000 software to create the topic.
   
   a. Set the project path (communication route to the controller).
   
   b. Save the project.

2. In RSLinx Classic software, check the topic.
   
   a. From the DDE/OPC menu, choose Topic Configuration.
   
   b. Select your project.
   
   c. Make sure the data source points to your controller.
   
   d. Click Done.
Add a Faceplate to Microsoft Excel Software

1. Start Microsoft Excel software.

2. Choose View > Toolbars > Control Toolbox.

3. Click and select the Logix 5000…Faceplate Control that you want.

4. In the location for the faceplate, drag the pointer to the desired size of the faceplate.

5. Right-click the faceplate and from the Logix 5000…Faceplate Control Object menu, choose Properties.

6. Click and browse to the tag that the faceplate controls.

7. Select the update period for the control and click OK.

8. To exit design mode and use the control, click here.
Enter Structured Text

Structured text is a textual programming language that uses statements to define what to execute. Structured text can contain these components.

As you enter structured text, follow these guidelines:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Construct defines logical conditions for the execution of other structured text code (other statements). In this example, the construct is If...Then...Else...End_if.</td>
</tr>
<tr>
<td>B</td>
<td>BOOL expression checks if a tag or equation is true or false. A BOOL expression typically serves as the condition for an action (the if, while, or until of a construct).</td>
</tr>
<tr>
<td>C</td>
<td>Assignment writes a value to a tag. The value moves from the right side of the := to the left side.</td>
</tr>
<tr>
<td>D</td>
<td>Numeric expression calculates a value.</td>
</tr>
<tr>
<td>E</td>
<td>Semicolon ';' terminates an assignment, instruction, or end of a construct.</td>
</tr>
</tbody>
</table>

As you enter structured text, follow these guidelines:

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structured text is not case sensitive.</td>
<td>Use any combination of upper-case and lower-case letters that makes your text easiest to read. For example, these three variations of “IF” are the same: IF, If, if.</td>
</tr>
<tr>
<td>Use tabs, spaces, and carriage returns (separate lines) to make your structured text easier to read.</td>
<td>Tabs, spaces, and carriage returns have no effect on the execution of the structured text.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>This</th>
<th>Executes the same as this</th>
</tr>
</thead>
</table>
| If Bool1 then  
  Bool2 := 1;  
End_if; | If Bool1 then Bool2 := 1; End_if; |
| Bool2 := 1; | Bool2:=1; |
Program a Project Offline  Chapter 5

Guideline                  Description
Write BOOL expressions as either true or false

Use a BOOL expression to determine if specific conditions are true (1) or false (0).
- A BOOL tag is already true (1) or false (0). Do not use an “=” sign to check its state.

<table>
<thead>
<tr>
<th>This is OK</th>
<th>This is NOT OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>If Bool1 …</td>
<td>If Bool1 = 1 …</td>
</tr>
<tr>
<td>If Not(Bool2) …</td>
<td>If Bool2 = 0 …</td>
</tr>
</tbody>
</table>
- To check an integer, REAL, or string, make a comparison (=, <, <=, >, >=, <>).

<table>
<thead>
<tr>
<th>This is OK</th>
<th>This is NOT OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>If Dint1 &gt; 5 …</td>
<td>If Dint1 …</td>
</tr>
</tbody>
</table>

For an assignment, start with the destination.

Write an assignment as follows:
Destination := Source;

Browse For an Instruction

1. Press Alt + Insert.

2. Type the mnemonic for the instruction and press Enter.
Assign Operands to an Instruction

1. Right-click the instruction and choose Argument List.

2. For each parameter, select a tag or type an immediate value.

3. Click OK.
Enter a Sequential Function Chart

A sequential function chart (SFC) lets you define a sequence of states (steps) through which your machine or process progresses. The steps can execute structured text, call subroutines, or simply serve as signals for other logic.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Step is major function of your process. It contains the actions that occur at a particular time, phase, or station.</td>
</tr>
<tr>
<td>B</td>
<td>Action is one of the functions that a step performs. To program the action, either enter structured text or call a subroutine.</td>
</tr>
<tr>
<td>C</td>
<td>Transition is a true or false condition that tells the SFC when to go to the next step. To specify the condition, either enter a BOOL expression in structured text or call a subroutine.</td>
</tr>
<tr>
<td>D</td>
<td>Branch executes more than 1 step at the same time (simultaneous) or choose between different steps (selective).</td>
</tr>
<tr>
<td>E</td>
<td>Wire connects one element to another anywhere on the chart.</td>
</tr>
</tbody>
</table>
Enter an SFC

1. Drag elements from the toolbar to the chart.
   - A green dot shows a point to which the element will automatically connect if you release the mouse button.
   - Some toolbar buttons are active only after you select a corresponding element on the SFC. For example, to add an action, first select a step.
   - Drag an action until it is on top of the required step and then release the mouse button.

2. To manually connect elements, click corresponding pins. A green dot shows a valid connection point.

3. To enter structured text, double-click a ? symbol. Then type the structured text and press Ctrl + Enter.
Assign Operands

RSLogix 5000 software lets you program according to your workflow. You can enter logic without assigning operands or defining tags. Later, you can go back and assign or define the operands to complete the logic.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>This instruction is missing an operand. You can enter and save logic without assigning operands. This lets you develop your logic in iterations and save libraries of code for re-use.</td>
</tr>
<tr>
<td>B</td>
<td>This is an undefined tag. You can enter and save logic without defining all the tags. This lets you develop your logic in iterations.</td>
</tr>
</tbody>
</table>

A tag name follows this format:

```plaintext
Name [Element] .Member [Element] .Bit
or
.[Index]
```

= Optional
## Chapter 5: Program a Project Offline

### Create a Tag

1. Double-click the tag area.

   ![Diagram](image)

2. Type a name for the tag and press Enter.

   Use underscores '_' in place of spaces.

3. Right-click the tag name and choose New.

4. Type the data type.

   To browse for a data type or assign array dimensions, click 

---

<table>
<thead>
<tr>
<th>Where</th>
<th>Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name that identifies this specific tag.</td>
</tr>
<tr>
<td>Element</td>
<td>Subscript or subscripts that point to a specific element within an array.</td>
</tr>
<tr>
<td></td>
<td>- Use the element identifier only if the tag or member is an array.</td>
</tr>
<tr>
<td></td>
<td>- Use one subscript for each dimension of the array. For example: [5], [2,8], [3,2,7].</td>
</tr>
<tr>
<td></td>
<td>To indirectly (dynamically) reference an element, use a tag or numeric expression that provides the element number. For example, MyArray[Tag_1], MyArray[Tag_2-1], MyArray[ABS(Tag_3)].</td>
</tr>
<tr>
<td>Member</td>
<td>Specific member of a structure.</td>
</tr>
<tr>
<td></td>
<td>- Use the member identifier only if the tag is a structure.</td>
</tr>
<tr>
<td></td>
<td>- If the structure contains another structure as one of its members, use additional levels of the <code>Member</code> format to identify the required member.</td>
</tr>
<tr>
<td>Bit</td>
<td>Specific bit of an integer data type (SINT, INT, or DINT).</td>
</tr>
<tr>
<td>Index</td>
<td>To indirectly (dynamically) reference a bit of an integer, use a tag or numeric expression that provides the bit number. For example, MyTag.[Tag_1], MyTag.[Tag_2-1], MyTag.[ABS(Tag_4)].</td>
</tr>
</tbody>
</table>
5. Choose the scope for the tag.

![New Tag dialog box](image)

6. Click OK.

**Select an Existing Tag**

1. Double-click the tag area.

2. Click the ✉.

![MainProgram - Routine B*](image)

3. Select the desired tag.
To select a bit, click the ◻.

4. To change the scope of tags in which to look, click the appropriate button.

Verify a Project

As you program your project, periodically verify your work.
Program a Project Offline  Chapter 5

Follow these steps to verify a routine or project.

1. Choose a verify option.

<table>
<thead>
<tr>
<th>To go to</th>
<th>Click this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify routine in view</td>
<td></td>
</tr>
<tr>
<td>Verify entire project</td>
<td></td>
</tr>
</tbody>
</table>

2. Go to an error or warning.

<table>
<thead>
<tr>
<th>To go to</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific error or warning</td>
<td>Double-click the error or warning.</td>
</tr>
<tr>
<td>Cycle through the list of errors and warnings</td>
<td>Press [F4].</td>
</tr>
</tbody>
</table>

3. Close the Errors tab.
4. To turn off duplicate destructive bit detection (it’s on by default), from the Tools menu, choose Options.

Guidelines for Tags

Use the following guidelines to create tags for a Logix5000 project.

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
</table>
| Create user-defined data types. | User-defined data types (structures) let you organize your data to match your machine or process. A user-defined data type provides these advantages:  
  • One tag contains all the data related to a specific aspect of your system. This keeps related data together and easy to locate, regardless of its data type.  
  • Each individual piece of data (member) gets a descriptive name. This automatically creates an initial level of documentation for your logic.  
  • You can use the data type to create multiple tags with the same data lay-out.  
For example, use a user-defined data type to store all the parameters for a tank, including temperatures, pressures, valve positions, and preset values. Then create a tag for each of your tanks based on that data type. |
| Use arrays to quickly create a group of similar tags. | An array creates multiple instances of a data type under a common tag name.  
  • Arrays let you organize a block of tags that use the same data type and perform a similar function.  
  • You organize the data in 1, 2, or 3 dimensions to match what the data represents.  
For example, use a 2 dimension array to organize the data for a tank farm. Each element of the array represents a single tank. The location of the element within the array represents the geographic location of the tank.  
**Important:** Minimize the use of BOOL arrays. Many array instructions do not operate on BOOL arrays. This makes it more difficult to initialize and clear an array of BOOL data.  
  • Typically, use a BOOL array for the bit-level objects of a PanelView screen.  
  • Otherwise, use the individual bits of a DINT tag or an array of DINTs. |
<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take advantage of program-scoped tags.</td>
<td>If you want multiple tags with the same name, define each tag at the program scope (program tags) for a different program. This lets you re-use both logic and tag names in multiple programs. Avoid using the same name for both a controller tag and a program tag. Within a program, you cannot reference a controller tag if a tag of the same name exists as a program tag. Certain tags must be controller scope (controller tag).</td>
</tr>
<tr>
<td>If you want to use the tag</td>
<td>Assign this scope</td>
</tr>
<tr>
<td>In more than one program in the project</td>
<td>controller scope (controller tags)</td>
</tr>
<tr>
<td>In a Message (MSG) instruction</td>
<td>controller scope (controller tags)</td>
</tr>
<tr>
<td>To produce or consume data</td>
<td>program scope (program tags)</td>
</tr>
<tr>
<td>To communicate with a PanelView terminal</td>
<td>program scope (program tags)</td>
</tr>
<tr>
<td>None of the above</td>
<td>program scope (program tags)</td>
</tr>
<tr>
<td>For integers, use the DINT data type.</td>
<td>To increase the efficiency of your logic, minimize the use of SINT or INT data types. Whenever possible, use the DINT data type for integers.</td>
</tr>
<tr>
<td></td>
<td>• A Logix5000 controller typically compares or manipulates values as 32-bit values (DINTs or REALs).</td>
</tr>
<tr>
<td></td>
<td>• The controller typically converts a SINT or INT value to a DINT or REAL value before it uses the value.</td>
</tr>
<tr>
<td></td>
<td>• If the destination is a SINT or INT tag, the controller typically converts the value back to a SINT or INT value.</td>
</tr>
<tr>
<td></td>
<td>• The conversion to or from SINTs or INTs occurs automatically with no extra programming. But it takes extra execution time and memory.</td>
</tr>
</tbody>
</table>
### Guideline Details

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit a tag name to 40 characters.</td>
<td>Here are the rules for a tag name:</td>
</tr>
<tr>
<td></td>
<td>• Only alphabetic characters (A-Z or a-z), numeric characters (0-9), and underscores (_)</td>
</tr>
<tr>
<td></td>
<td>• Must start with an alphabetic character or an underscore</td>
</tr>
<tr>
<td></td>
<td>• No more than 40 characters</td>
</tr>
<tr>
<td></td>
<td>• No consecutive or trailing underscore characters (_).</td>
</tr>
<tr>
<td></td>
<td>• Not case sensitive</td>
</tr>
<tr>
<td>Use mixed case.</td>
<td>Although tags are not case sensitive (upper case A is the same as lower case a), mixed</td>
</tr>
<tr>
<td></td>
<td>case is easier to read</td>
</tr>
<tr>
<td></td>
<td>These tags are easier to read</td>
</tr>
<tr>
<td></td>
<td>Than these tags</td>
</tr>
<tr>
<td>Tank_1</td>
<td>TANK_1</td>
</tr>
<tr>
<td>Tank1</td>
<td>TANK1</td>
</tr>
<tr>
<td>tank_1</td>
<td>tank1</td>
</tr>
<tr>
<td>tank1</td>
<td></td>
</tr>
<tr>
<td>Consider the alphabetical order of tags.</td>
<td>RSLogix 5000 software displays tags of the same scope in alphabetical order. To make it</td>
</tr>
<tr>
<td></td>
<td>easier to monitor related tags, use similar starting characters for tags that you want to</td>
</tr>
<tr>
<td></td>
<td>keep together.</td>
</tr>
<tr>
<td></td>
<td>Starting each tag for a tank with Tank keeps the tags together.</td>
</tr>
<tr>
<td></td>
<td>Otherwise, the tags may end up separated from each other.</td>
</tr>
<tr>
<td><strong>Tag Name</strong></td>
<td></td>
</tr>
<tr>
<td>Tank_North</td>
<td></td>
</tr>
<tr>
<td>Tank_South</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
</tr>
<tr>
<td><strong>Tag Name</strong></td>
<td></td>
</tr>
<tr>
<td>North_Tank</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
</tr>
<tr>
<td>South_Tank</td>
<td>other tags that start with the letters o, p, q, and so on.</td>
</tr>
</tbody>
</table>
Document a Project

Use this chapter to document your RSLogix 5000 project. This makes the system easier to debug, maintain, and troubleshoot.

What You Need

You need these items to complete the tasks in this manual:

- Personal computer running RSLogix 5000 software, version 16
- The project you are documenting

Follow These Steps

1. Describe a user-defined data type (page 98).
2. Add rung comments (page 101).
3. Enter and edit rung comments using Microsoft Excel (page 102).
4. Add comments to a function block diagram or SFC (page 105).
5. Add comments to structured text (page 107).
User-defined Data Type

RSLogix 5000 software lets you automatically build descriptions out of the descriptions in your user-defined data types. This greatly reduces the amount of time you have to spend documenting your project. As you organize your user-defined data types, keep in mind the following features of RSLogix 5000 software:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| A    | Pass through of descriptions are automatically created when possible, RSLogix 5000 software looks for an available description for a tag, element, or member.  
  - Descriptions in user-defined data types ripple through to the tags that use that data type.  
  - Description of an array tag ripples through to the elements and members of the array. |
| B    | Append descriptions to base tags. RSLogix 5000 software automatically builds a description for each member of a tag that uses a user-defined data type. It starts with the description of the tag and then adds the description of the member from the data type. |
| C    | Edit pass-through descriptions so that you can use the data type and array description as a basis for more specific descriptions.  
  In this example, Tank became West Tank. |
RSLogix 5000 software uses different colors for descriptions:

<table>
<thead>
<tr>
<th>A description in this color</th>
<th>Is a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gray</td>
<td>Pass-through description</td>
</tr>
<tr>
<td>Black</td>
<td>Manually entered description</td>
</tr>
</tbody>
</table>

**Turn Pass-Through and Append Descriptions On or Off**

1. In RSLogix 5000 software, from the Tools menu, choose Options.

2. Select the Display.

3. Turn on (check) or turn off (uncheck) the desired options.
Paste a Pass-Through Description

To use a pass-through description as the starting point for a more specific description.

1. Right-click the pass-through description and choose Paste Pass-Through.

2. Edit the description and press Ctrl + Enter.
Add Rung Comments

Use a rung comment to describe the operation of a rung of ladder logic. You can also start the routine with a rung that contains only a No Operation (NOP) instruction. Add a comment to this initial rung that describes the routine in general.

1. Right-click the rung and choose Edit Rung Comment.

2. Type your comments.

3. Click the check to save the comment and close the entry window.
Rung Comments Using Microsoft Excel

You can also use spreadsheet software such as Microsoft Excel to create and edit rung comments. This lets you take advantage of the editing features in the spreadsheet software.

**IMPORTANT** Rung comments export in the CSV (comma delimited) format. Make sure you keep that format when you save and close the export file.

Export the Existing Comments

1. In RSLogix 5000 software, add at least one rung comment. This helps to format the export file.

2. From the Tools menu, choose Export.
3. Note the location and name of the export file.

![Export dialog box with file name and save as type selected.]

4. Choose what to export.

5. Click Export.

**Edit the Export File**

1. In Microsoft Excel software, open the export file.

2. Enter rung comments.

![Excel sheet with column headers: TYPE, SCOPE, ROUTINE, COMMENT, OWNING_ELEMENT, LOCATION. Data includes: 8. RCOMMMENT, MainProgram, MainRoutine, This routine calls the execution of all other routines. NCQF0, 8. RCOMMMENT, MainProgram, MainRoutine, If the conveyor is not turning on or off, check this routine. 10. RCOMMMENT, program that contains the rung, routine that contains the rung, comments for the rung, leave blank, rung number.]

4. Save and close the file (keep it in the CSV format).
Import the New Comments

1. In RSLogix 5000 software, from the Tools menu, choose Import.

   ![RSLogix 5000 screenshot](image1)

2. Select the file that has the comments you entered (the export file).

   ![Import file selection](image2)

3. Click Import.

4. Check the Errors tab for the results of the import operation. To refresh the view of the ladder logic and see the comments, close and open the routine.

   ![Import errors](image3)
Comments in a Function Block Diagram or SFC

Use Text boxes to add notes about the diagram or chart in general or a specific element. Or use a text box to capture information that you will use later on as you develop the project.

Set the Word Wrap Option

Use the word wrap option to control the width of the text box as you type. You set the option for function block diagrams and SFC independent of each other.

<table>
<thead>
<tr>
<th>If you want text boxes to</th>
<th>Choose this option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatically grow to the width of the longest line of text in the box.</td>
<td>☐ Word Wrap</td>
</tr>
</tbody>
</table>

| Turns conveyor on and off based on start and stop buttons. If both start and stop are on, the stop button overrides the start button. | ☑ Word Wrap |

Retain a fixed width and wrap the text. You can always manually resize the box.

1. In RSLogix 5000 software, from the Tools menu, choose Options.
2. Select the editor.

3. Select or clear the word wrap option.

**Add a Text Box**

1. Drag the text box button from the toolbar to the chart.

2. Type the comment and press Ctrl + Enter.

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

To make your structured text easier to interpret, add comments. Comments:
- let you use plain language to describe how your structured text works.
- download to the controller and upload from the controller.
- do not affect the execution of the structured text.

Follow these steps to add comments to your structured text.

<table>
<thead>
<tr>
<th>To add a comment</th>
<th>Use one of these formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>On a single line</td>
<td>//comment</td>
</tr>
</tbody>
</table>
| At the end of a line of structured text | (*comment*)
| | /*comment*/ |
| Within a line of structured text | (*comment*)
| | /*comment*/ |
| That spans more than one line | (*start of comment . . . end of comment*)
| | /*start of comment . . . end of comment*/ |

Here is an example.

<table>
<thead>
<tr>
<th>Format</th>
<th>Example</th>
</tr>
</thead>
</table>
| //comment | At the beginning of a line
| | //Check conveyor belt direction
| | IF conveyor_direction THEN...
| | At the end of a line
| | ELSE //If conveyor isn’t moving, set alarm light
| | light := 1;
| | END_IF; |
| (*comment*) | Sugar.Inlet[:=]1;(*open the inlet*)
| | IF Sugar.Low (*low level LS*)& Sugar.High (*high level LS*)THEN...
| | (*Controls the speed of the recirculation pump. The speed depends on the temperature in the tank.*)
| | IF tank.temp > 200 THEN...
| /*comment*/ | Sugar.Inlet:=0;/*close the inlet*/
| | IF bar_code=65 /*A*/ THEN...
| | /*Gets the number of elements in the Inventory array and stores the value in the Inventory_Items tag*/
| | SIZE(Inventory,0,Inventory_Items); |
Language Switching

With RSLogix 5000 software, version 17, 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. You define all the localized languages that the project will support and set the current, default, and optional custom localized language. The software uses the default language if the current language's content 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 your RSLogix 5000 project, either when programming in that language or by using the import/export utility to translate the documentation off-line and then import it back into the project. Once you enable language switching in RSLogix 5000 software, you can dynamically switch between languages as you use the software.

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.
Go Online to the Controller

Use this chapter to access the project in the controller so you can monitor, edit, or troubleshoot the controller.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 software, version 16 or later, and RSLinx software
- The physical system to which you are connecting
- EtherNet/IP cabling
- EtherNet/IP communication card(s) for the module(s) in our sample project
- The project you want to access

Follow These Steps

1. Establish EtherNet/IP communication with the controller (page 110)
2. Go online to a controller (page 116)
Establish EtherNet/IP Communication with the Controller

RSLinx Classic software handles communication between Logix5000 controllers and your software programs, such as RSLogix 5000 software. To communicate with a controller (download or monitor data), configure RSLinx Classic software for the required communication.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet address (MAC)</td>
<td>Address that is assigned to a module at the factory.</td>
</tr>
<tr>
<td></td>
<td>• The module always keeps its ethernet address.</td>
</tr>
<tr>
<td></td>
<td>• To determine the ethernet address of a device, look for a sticker on the device.</td>
</tr>
<tr>
<td></td>
<td>• An ethernet address uses this format:</td>
</tr>
<tr>
<td>IP address</td>
<td>Address that you assign to a module for communication over a specific ethernet network. An IP address uses this format:</td>
</tr>
<tr>
<td></td>
<td>xxx.xxx.xxx.xxx</td>
</tr>
<tr>
<td>BOOTP</td>
<td>Configure a device to request an IP address over an ethernet network from a BOOTP server. Out of the box, Allen-Bradley EtherNet/IP devices are configured for BOOTP.</td>
</tr>
<tr>
<td>BOOTP server</td>
<td>Software program that receives BOOTP requests from ethernet devices and assigns IP addresses. RSLinx software revision 2.40 and later includes BOOTP server software.</td>
</tr>
<tr>
<td>Driver</td>
<td>Establish communication over a specific network.</td>
</tr>
<tr>
<td>Path</td>
<td>Communication route to a device. To define a path, you expand a driver and select the device.</td>
</tr>
</tbody>
</table>
Equipment and Information That You Need

1. Depending on your controller, you may need a communication module or daughter card.

2. Determine if your EtherNet/IP network is connected to the Internet or if it is a standalone network that does not connect to the Internet?

3. For the EtherNet/IP device (controller, bridge module, or daughter card), obtain the following:

<table>
<thead>
<tr>
<th>Obtain this</th>
<th>If your network is connected to the Internet, from this source</th>
<th>If your network is a standalone network that does not connect to the Internet, from this source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet address</td>
<td>Sticker on the device</td>
<td>Sticker on the device</td>
</tr>
<tr>
<td>IP address</td>
<td>Network administrator</td>
<td>192.168.1.x, where x = any value between 1 and 254(^{(1)})</td>
</tr>
<tr>
<td>Subnet mask</td>
<td></td>
<td>255.255.255.0(^{(2)})</td>
</tr>
<tr>
<td>Gateway address (may not be required)</td>
<td></td>
<td>Not needed</td>
</tr>
</tbody>
</table>

\(^{(1)}\) In this case, your computer must use an IP address that is close to the EtherNet/IP device’s IP address. For example, if the EtherNet/IP device uses the 192.168.1.x addressing, the computer must also use that addressing but with a different x value.

\(^{(2)}\) In this case, your computer must use the same subnet mask value as the EtherNet/IP device.
Connect Your EtherNet/IP Device and Computer

Connect your EtherNet/IP device and computer via ethernet cable.

**ATTENTION** If you connect or disconnect the communications cable with power applied to this module or any device on the network, an electrical arc can occur. This could cause an explosion in hazardous location installations.

Assign an IP Address to the Controller or Communication Module

Follow these steps if you do not have a serial connection to the controller.

1. Start BOOTP server software by either of the following:
   - Start > Programs > Rockwell Software > BOOTP-DHCP Server > BOOTP-DHCP Server
   - Start > Programs > Rockwell Software > RSLinx Tools > BOOTP-DHCP Server.

2. If this is the first time you are using the software, type the subnet mask and gateway (if required) for your network and then click OK.
3. Double-click the Ethernet address of the controller/communication module.

4. Type the IP address and click OK.

5. In the Relation List (lower section), select the device and click Disable BOOTP.

   This lets the device keep the address even after a power cycle.

6. When you close the BOOTP server software, you are prompted to save your changes.
   • If you want a record of the IP address that you assigned to the device, save the changes.
   • Regardless of whether you save the changes, the device keeps the IP address.

Follow these steps if you have a serial connection to the controller.

1. Start RSLinx software.

2. Click .

3. Browse to the EtherNet/IP device.

4. Right-click the device and choose Module Configuration.
5. Click the Port Configuration tab.

6. Depending on your device, either:
   - Select the Static button.
   - Clear (uncheck) the Obtain IP Address from BOOTP Server check box.

7. Type the:
   - IP address.
   - subnet mask.
   - gateway address (if required).

8. Click OK and then click Yes.

**Assign an IP Address to Your Computer**

If your EtherNet/IP network is a standalone network and your EtherNet/IP device uses IP address and subnet mask values, you may need to change the IP address and subnet mask values for your computer.

1. Choose Start > Settings > Network and Dial-up Connections.

2. Right-click on Local Area Connection.

3. Choose Properties.

5. Choose Properties.

6. Select Use the following IP address.

7. Change the IP address and subnet mask.

8. Click OK.

**Configure an Ethernet Driver**

1. Start RSLinx software.

2. Click \[\text{Add New...}\].

3. Select Ethernet devices and choose \[\text{Add New...}\].
4. Accept the default name.

5. Type the IP address of the controller or communication module.

6. Click OK.

The driver is successfully configured and running.

Online with a Controller

To monitor a project that is executing in a controller, go online with the controller. The procedure that you use depends on whether you have a copy of the project on your computer.
If Your Computer Has the Project For the Controller

1. Open the RSLogix 5000 project for the controller.

![RSLogix 5000 screenshot]

2. Click to define a path to the controller.

![Path selection]

3. Select the controller.

![Controller selection]

4. Click .

If Your Computer Does Not Have the Project For the Controller

1. Open the RSLogix 5000 project for the controller.

2. Click to define a path to the controller.

![Path selection]
3. Select the controller.

4. Click **Upload**.

5. Click **Select File** to create the project file on your computer.

6. Click **Select** and then **Yes**.
Program a Project Online

Use this chapter to edit your logic while the controller continues to control your machine or process.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 software, version 16 or later, and RSLinx software
- The physical system to which you are connecting
- The project you want to access

Follow These Steps

1. Edit Logic While Online (page 119)
2. Finalize All Edits in a Program (page 124)

Edit Logic While Online

Online edits let you change your logic while your machine or process continues to run.

ATTENTION

Use extreme caution when you edit logic online. Mistakes can injure personnel and damage equipment. Before you edit online:
- assess how machinery will respond to the changes.
- notify all personnel of the changes.

IMPORTANT

When you edit an SFC online:
- the SFC resets to the initial step.
- stored actions turn off.
As you perform online edits, RSLogix 5000 software uses markers to show the state of your edits.

### Relay Ladder

<table>
<thead>
<tr>
<th>This marker</th>
<th>Means</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relay ladder</td>
<td><img src="image" alt="Original logic marker" /></td>
<td>When online, RSLogix 5000 software continues to show you the original logic while you edit a copy of the logic (pending edit). A green border or side rail shows which logic the controller is currently running. In function block, structured text, or SFC, use the buttons above the routine to switch between different views.</td>
</tr>
</tbody>
</table>

### Function Block, Structured Text, SFC

<table>
<thead>
<tr>
<th>This marker</th>
<th>Means</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relay ladder</td>
<td><img src="image" alt="Pending edits marker" /></td>
<td>This is a copy of the original logic for you to edit. Any changes remain on your computer until you accept the edits.</td>
</tr>
<tr>
<td>Function block</td>
<td><img src="image" alt="Pending edits marker" /></td>
<td>In relay ladder, you edit individual rungs within a routine.</td>
</tr>
<tr>
<td>Structured text</td>
<td><img src="image" alt="Pending edits marker" /></td>
<td>In function block, structured text, or SFC, you edit an entire routine.</td>
</tr>
</tbody>
</table>
### Program a Project Online

#### Chapter 8

#### Start a Pending Edit

1. For relay ladder, click (select) the rung that you want to edit.

2. Start a pending edit.

<table>
<thead>
<tr>
<th>This marker</th>
<th>Means</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relay ladder</td>
<td>Test edits</td>
<td>When you accept your pending edits, the software downloads them to the controller and marks them as test edits but the controller continues to execute the original logic. You then manually switch execution to the test edits or back to the original logic (test and untest the edits).</td>
</tr>
<tr>
<td>Function block Structured text SFC</td>
<td>- or -</td>
<td>In relay ladder, if you delete a rung the software immediately marks it as a test edit (upper-case ‘D’ character).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>If you</th>
<th>Then</th>
</tr>
</thead>
</table>
| Test the edits | • Execution switches to the test edits (all test edits execute).  
• Outputs in the original logic stay in their last state unless executed by the test edits (or other logic).  
• In an SFC, the chart resets to the initial step and stored actions turn off. |
| Untest the edits | • Execution switches back to the original logic.  
• Outputs in the test edits stay in their last state unless executed by the original logic (or other logic).  
• In an SFC, the chart resets to the initial step and stored actions turn off. |

Relay Ladder

![Relay Ladder](image1)

Function Block, Structured Text, SFC

![Function Block, Structured Text, SFC](image2)
Make and Accept Your Edits

1. Make your changes.

2. Accept your changes.

The changes download to the controller and become test edits.

Test the Edits

1. Test the edits to see if they execute as intended.
2. Click Yes to test the edits.

![Image of RsLogix 5000 interface showing a test edit]

3. If the edits are not correct, click ☐ to switch execution back to your original logic (untest the edits).

   To make changes, start another pending edit.

**Assemble and Save the Edits**

1. Assemble the edits.

   The edits become permanent and the original logic is removed.

2. Save the project.
Finalize All Edits in a Program

The Finalize All Edits in Program option lets you make an online change to your logic without testing the change.

Follow these steps to use the Finalize All Edits in Program option.

1. Start a pending edit.
2. Make your change.
3. Choose Finalize All Edits in Program.

Use extreme caution when you edit logic online. Mistakes can injure personnel and damage equipment. Before you edit online:

- assess how machinery will respond to the changes.
- notify all personnel of the changes.

When you choose Finalize All Edits in Program:

- all edits in the program (pending and test), immediately download to the controller and begin execution.
- the original logic is permanently removed from the controller.
- outputs that were in the original logic stay in their last state unless executed by the new logic (or other logic).

If your edits include an SFC:

- the SFC resets to the initial step.
- stored actions turn off.

IMPORTANT When editing online, if the program scan time is large, or the number of modified rungs is large, you might see HMI and RSLogix 5000 communication timeouts when edits are finalized.

The timeout is caused by the (scan time) x (number of changed rungs). You could have a large program with a very fast scan, or a lot of rungs (but you only modified a few), and you will not see a timeout.
Troubleshoot the Controller

Use this chapter to obtain basic diagnostic information about your system and perform basic tasks.

What You Need

You need these items to complete the tasks in this manual:
- Personal computer running RSLogix 5000 software, version 16 and RSLinx software
- The physical system you are troubleshooting
- The project you want to troubleshoot

Follow These Steps

4. Troubleshoot I/O communication (page 126).

5. Clear a major fault (page 127).

6. Search a project (page 128).

7. Browse logic (page 130).

8. Force an I/O value (page 131).

9. Create and run a trend (histogram) (page 135).

10. View scan time (page 138).
Troubleshoot I/O Communication

If there is a problem with several of the devices in your system, communication with an I/O module may have failed.

The I/O indicator on the front of the controller and in the programming software indicates status.

<table>
<thead>
<tr>
<th>If the indicator is</th>
<th>Then</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>Either:</td>
</tr>
<tr>
<td></td>
<td>• There are no modules in the I/O configuration of the controller.</td>
</tr>
<tr>
<td></td>
<td>• The controller does not contain a project (controller memory is empty).</td>
</tr>
<tr>
<td>Solid green</td>
<td>The controller is communicating with all the modules in its I/O configuration.</td>
</tr>
<tr>
<td>Flashing green</td>
<td>One or more modules in the I/O configuration of the controller are not responding.</td>
</tr>
</tbody>
</table>

The Controller Organizer also shows status.
Clear a Major Fault

If your entire process unexpectedly shuts down, the controller may have experienced a major fault. A major fault is a condition severe enough for the controller to shut down.

1. Go online with the controller.

2. Choose Go To Faults.

3. Use this information to correct the cause of the fault.

4. After you correct the cause of the fault, click **Clear Majors**.
Search Functions in a Project

You can find an element of your logic (such as a tag, instruction, or comment) based on the characters that you search for.

<table>
<thead>
<tr>
<th>To find a(n)</th>
<th>Specify</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>Full or partial tag name</td>
<td>MyTag_1</td>
</tr>
<tr>
<td>Comment/description</td>
<td>Text within the comment/description</td>
<td>fan</td>
</tr>
<tr>
<td>Instruction</td>
<td>Mnemonic of the instruction</td>
<td>OTE</td>
</tr>
<tr>
<td>Instruction and tag</td>
<td>Mnemonic and tag</td>
<td>OTE MyTag_1</td>
</tr>
</tbody>
</table>

Search for All Occurrences of a Element

1. Open the RSLogix 5000 project that you want to search.

2. From the Search menu, choose Find.
3. Specify the search criteria.
   a. Type the characters to find.
      To browse for a tag, click \( \mathbf{O} \), select the tag, and click OK.
      To select a bit number, click the \( \mathbf{\downarrow} \).

   b. Choose Text Only.
   c. Choose All Routines.
   d. Select each language and check the options in which to search.
      To display this section of the dialog box, click \( \text{Find Within} \).

4. Click \( \text{Find All} \).

Go to an Instruction

1. To go to an instruction, double-click it.

2. To show a list of cross-references to a tag, right-click and choose Go To Cross Reference.
3. To go to an instruction, double-click it.

A “Y” means this instruction changes the value of the tag.

**Browse Logic**

To browse the logic of a routine for a specific item (such as an instruction, element, tag, or comment), use the Browse Logic window.

1. In RSLogix 5000 software, from the Search menu, choose Browse Logic.
2. Expand and collapse entries to see or hide its contents.

3. To go to the location of a element in logic, select the element and click Go To.

**Forcing an I/O Value**

Use a force to override input data or logic when you need to:
- test and debug your logic.
- check wiring to an output device.
- temporarily keep your process functioning when an input device has failed.

**ATTENTION**

Forcing can cause unexpected machine motion that could injure personnel. Before you install, disable, or remove a force, determine how the change will effect your machine or process and keep personnel away from the machine area.

Enabling I/O forces causes input, output, produced, or consumed values to change.

If you remove an individual force, forces remain in the enabled state.

If forces are enabled and you install a force, the new force immediately takes effect.
Chapter 7  Troubleshoot the Controller

A force overrides a value from an input device or logic.

- Forcing an input tag overrides the value from the input device.
- Forcing an output tag overrides your logic and sends the force value to the output device.

When forces are in effect (enabled), a ▶ appears next to the forced element.

The force indicator on the front of the controller and in the programming software indicates status.

<table>
<thead>
<tr>
<th>If the indicator is</th>
<th>Then</th>
</tr>
</thead>
</table>
| Off                | • No tags contain I/O force values.  
                    | • I/O forces are inactive (disabled). |
| Flashing amber     | • One or more tags contain a force value.  
                    | • I/O forces are inactive (disabled).  
                    | • When you enable I/O forces, all existing I/O forces take effect. |
| Solid amber        | • I/O forces are active (enabled).  
                    | • Force values may or may not exist.  
                    | • When you install (add) a force, it immediately takes effect. |
Install an I/O Force (Force an I/O Value)

1. Go online with the controller and open the routine that contains the tag that you want to force.

2. Right-click the tag and choose Monitor.

3. If necessary, click the + sign of the tag to show the value that you want to force (for example, the BOOL value of a DINT tag).

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Value</th>
<th>Force Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local4:C</td>
<td>(...)</td>
<td>(...)</td>
</tr>
<tr>
<td>Local4:I</td>
<td>(...)</td>
<td>Forced</td>
</tr>
<tr>
<td>Local4:I.Fault</td>
<td>2#00...</td>
<td></td>
</tr>
<tr>
<td>Local4:I.Data</td>
<td>2#0...</td>
<td>2#...</td>
</tr>
<tr>
<td>Local4:I.Data0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Local4:I.Data1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

4. Install the force value:

<table>
<thead>
<tr>
<th>To force a</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOL value</td>
<td>Right-click the tag and choose Force ON or Force OFF.</td>
</tr>
<tr>
<td>Integer or REAL value</td>
<td>In the Force Mask column for the tag, type the value to which you want to force the tag and press [Enter].</td>
</tr>
</tbody>
</table>

5. From the Forces menu, choose I/O Forcing > Enable All I/O Forces and click Yes
Remove an Individual Force

1. Go online with the controller and open the routine that contains the tag that you want to force.

2. Right-click the tag and choose Monitor.

3. If necessary, click the + sign of the tag to show its members (for example, the BOOL value of a DINT tag).

4. Right-click the tag and choose Remove Force.

Disable All I/O Forces

1. Go online with the controller.

2. From the Forces menu, choose I/O Forcing > Disable All I/O Forces and click Yes.
Data Trend (Histogram)

Trends let you view sampled tag data over a period of time on a graphical display. Tag data is sampled by the controller and then displayed as point(s) on a trend chart.

![Trend Chart Example]

Run a Trend for a Tag

Right-click the first tag that you want to trend and choose Trend.

![Trend Setup Example]
Add More Tags to the Trend

1. Right-click the chart and choose Chart Properties.

2. Click the Pens tab.

3. Click Add/Configure Tag.

4. Select a tag to add and click Add. To change the scope, select a scope.

5. To select a bit number, click ♦.

6. When you have added the required tags, click OK.

7. Click the Y-Axis tab.
8. Choose the type of graphing and click OK.

9. To resume the trend, click Run.

**Save the Trend**

1. Close the trend.

You get the choice to save the trend for future use.
2. Type a name for the trend and click Finish.

View Scan Time

A Logix5000 controller provides two types of scan times. Each serves a different purpose.

<table>
<thead>
<tr>
<th>Scan Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed time (task scan time)</td>
<td>Time that has elapsed from the start of a task to the end of the task, in milliseconds. The elapsed time of a task includes the time that the task is interrupted to service communications or other tasks.</td>
</tr>
<tr>
<td>Execution time (program scan time)</td>
<td>Time to execute the logic of a program (its main routine and any subroutines that the main routine calls), in microseconds. The scan time of a program includes only the execution time of the logic. It does not include any interrupts.</td>
</tr>
</tbody>
</table>

View Task Scan Time

1. Right-click and choose Properties.
2. Click the Monitor tab.

![Task Properties - Main Task](image)

<table>
<thead>
<tr>
<th>Task Properties - Main Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
</tr>
<tr>
<td>Scan Times (Elapsed Time)</td>
</tr>
<tr>
<td>Max: 1.320000 ms</td>
</tr>
<tr>
<td>Last: 0.048000 ms</td>
</tr>
</tbody>
</table>

Elapsed Time of The Last Execution of This Task
Maximum Elapsed Time of the Task

### View Program Scan Time

1. Right-click and choose Properties.

![Controller - Controller Quick Start](image)

2. Click the Configuration tab.

![Program Properties - MainProgram](image)

<table>
<thead>
<tr>
<th>Program Properties - MainProgram</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
</tr>
<tr>
<td>Assigned Routines:</td>
</tr>
<tr>
<td>Main: MainRoutine</td>
</tr>
<tr>
<td>Fault: (none)</td>
</tr>
<tr>
<td>Scan Times (execution time):</td>
</tr>
<tr>
<td>Max: 84 us</td>
</tr>
<tr>
<td>Last: 10 us</td>
</tr>
</tbody>
</table>

Maximum Execution Time of This Program
Execution Time of the Last Execution of This Program
Notes:
Symbols

102

A
add
phase state routine 60
alias tags
use 25
array
create 40
organize 94
use of 40
ASCII text
enter logic using 68
assume data available indicator
use of 77

B
BOOTP
use of 110
browse
logic 130

C
clear
major fault 127
command
give with RSLogix 5000 software 61
comment
add to function block diagram 105
add to rung 101, 102
add to SFC 105
add to structured text 107
search for 128
communicate
with controller via EtherNet/IP network 110
with controller via serial cable 28
communication
fault 126
configure
controller 14, 48
driver for EtherNet/IP communication 110
driver for serial communication 28
I/O module 15, 49
task 36
trend 135
continuous task
execution 36
controller
communicate via EtherNet/IP network 110
communicate via serial cable 28
configure 14, 48
download project 30
faulted 127
go online with 116
mode 32
monitor 118
monitor execution 116
revision 30
controller organizer
add I/O module 15
navigate 14
controller-scope tags
when to use 38
create
phase state routine 60
program 38
routine 43
sheet 77
text box 105
trend 135

data
I/O module 17
trend 135
description
rung 101, 102
search for 128
tag 98
user-defined data type 98
document
function block diagram 105
rung 101, 102
SFC 105
structured text 107
tag 98
user-defined data type 98
download
project 30
driver
configure for EtherNet/IP communication 110
calculate for serial communication 28
duplicate destructive bit detection
use of 92
elapsed time
task 138
enter
function block diagram 77
ladder logic 68
logic while online 119, 124
rung comment 101
SFC 87
structured text 84
comments 107
equipment phase
create a phase state routine 60
inhibit 25, 64
initial state 63
monitor 61
phase state routine 60
set initial step index 25, 64
set the initial state 63
test states 61
errors
check routine for 92
EtherNet/IP network
assign IP address 110
communicate with controller 110
execution
choose controller mode 32
task 36
time 138
export
ladder logic 73
rung comment 102
external request
hold action 66
respond to lost communication 65

F
faceplate
add 81
fault
controller 127
I/O module 126
file
See array
finalize all edits in program 124
find
See search
firmware
update during download 30
force
I/O value 131
function block diagram
create sheet 77
document 105
edit online 119, 124
enter 77
resolve loop 77
use for 43
function block instruction
use of faceplate 81

H
histogram
See trend

I
I/O device
access data 17
I/O module
add to project 15
address format 17
communication failure 126
configure 15, 49
faulted 126
force value 131
import
ladder logic 73
rung comment 102
inhibit
equipment phase 25, 64
initial state
set 63
initial step index
set 25, 64
instruction
search for 128
IP address
assign to module 110

L
ladder logic
add rung comment 101, 102
edit online 119, 124
enter 68
export 73
import 73
use for 43
use of quick keys 68
library of logic
create and use 73
Index

logic
- check for errors 92
- edit online 119, 124

M
main routine
- assign 47
- use of 43
major fault
- clear 127
mode
- controller 32
monitor
- controller 116
- equipment phase 61
- project in controller 118

N
name
- guidelines for tag 94
- limitations 14

O
online
- edit logic 119, 124
- finalize all edits 124
- with controller 116
operand
- assign 89

P
pass-through description 98
period
- define for task 36
periodic task
- execution 36
phase state routine
- add 60
program
- assign main routine 47
- create 38
- finalize all edits 124
- scan time 138
program mode 32
programming language
- choose 43
- RSLogix 5000 software 44
program-scope tags
- when to use 38

project
- create 14
- download 30
- monitor in controller 116
- organize routines 43
- upload 118
- verify 92
PXRQ instruction
- hold action 66
- lost communication 65

Q
quick keys
- enter ladder logic 68

R
revision
- controller firmware 30
routine
- add phase state routine 60
- check for errors 92
- create 43
- create tag 89
- edit logic online 119, 124
- import ladder logic 73
- organize 43
- program using a function block diagram 77
- program using an SFC 87
- program using structured text 84
RSLogix 5000 software
- give command 61
- monitor an equipment phase 61
run mode 32
rung comment
- add 101, 102
- export/import 102

S
scan time
- view 138
scope
- choose for tag 38
- guidelines 94
search
- browse 130
- comments or descriptions 128
- instruction 128
- tag 128

Publication 1756-QS001E-EN-P - October 2009
sequential function chart
  See SFC
serial communication
  with controller 28
set
  hold action for a PXRQ instruction 66
  initial step index 25, 64
SFC
  document 105
  edit online 119, 124
  enter 87
  use for 43
sheet
  use of 77
state routine
  See phase state routine
states
  set the initial state 63
  step through 61
structure
  create 40
  organize 94
structured text
  document 107
  edit online 119, 124
  enter 84
  use for 43
subroutine
  See routine
text box
  add to function block diagram 105
  add to SFC 105
transition
  step through 61
trend
  create and run 135
troubleshoot
  check wiring to output device 131
  communication with I/O module 126
  entire system is shut down 127
  override logic 131
  see data history 135
  several devices not responding 126

U
update
  controller firmware 30
upload
  project 118
user-defined data type
  create 40
  use of 40

V
verify
  project 92
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 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://www.rockwellautomation.com/support/.

Installation Assistance

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

<table>
<thead>
<tr>
<th>United States or Canada</th>
<th>1.440.646.3434</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside United States or Canada</td>
<td>Use the Worldwide Locator at <a href="http://www.rockwellautomation.com/support/americas/phone_en.html">http://www.rockwellautomation.com/support/americas/phone_en.html</a> or contact your local Rockwell Automation representative.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>United States</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside United States</td>
<td>Please contact your local Rockwell Automation representative for the return procedure.</td>
</tr>
</tbody>
</table>

Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete this form, publication RA-DU002, available at http://www.rockwellautomation.com/literature/.