Logix 5000 Controllers
Ladder Diagram

1756 ControlLogix, 1756 GuardLogix, 1769 CompactLogix,
1769 Compact GuardLogix, 1789 SoftLogix, 5069
CompactLogix, 5069 Compact GuardLogix, Studio 5000
Logix Emulate
Important User Information

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

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice.

If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

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

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

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

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

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

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

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

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

Grammatical and editorial style changes are not included in this summary.

**Global changes**

This table identifies changes that apply to all information about a subject in the manual and the reason for the change. For example, the addition of new supported hardware, a software design change, or additional reference material would result in changes to all of the topics that deal with that subject.

<table>
<thead>
<tr>
<th>Change</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updated Legal notices.</td>
<td>Legal notices on page 8</td>
</tr>
<tr>
<td>Updated branding.</td>
<td>Throughout</td>
</tr>
</tbody>
</table>

**New or enhanced features**

None in this release.
## Table of Contents

### Summary of changes
- Studio 5000 environment ................................................................. 7
- Additional resources ............................................................................. 7
- Legal notices ............................................................................................. 8

### Preface

### Program ladder diagram
- Introduction ............................................................................................. 11
  - Instruction ............................................................................................... 11
  - Branch ..................................................................................................... 11
  - Rung condition ........................................................................................ 12
- Write ladder logic ...................................................................................... 12
  - Choose the required instructions ....................................................... 13
  - Arrange the input instructions ........................................................... 13
  - Arrange the output instructions ......................................................... 14
  - Choose a tag name for an operand ..................................................... 15
- Enter ladder logic ...................................................................................... 16
  - Append an element to the cursor location ......................................... 16
  - Drag and drop an element ................................................................... 17
- Assign instruction operands ..................................................................... 17
  - Create and assign a new tag ............................................................... 17
  - Choose a name or an existing tag ....................................................... 18
  - Drag and drop a tag from the Tags window ...................................... 18
  - Assign an immediate (constant) value ............................................... 19
- Enter a rung comment .............................................................................. 19
  - Language switching ............................................................................. 19
- Verify the routine ...................................................................................... 20

### Index
This manual shows how to program Logix 5000 controllers with the relay
ladder programming language. This manual is one of a set of related manuals
that show common procedures for programming and operating Logix 5000
controllers.

For a complete list of common procedures manuals, refer to the Logix 5000
Controllers Common Procedures Programming Manual, publication 1756-
PM001.

The term Logix 5000 controller refers to any controller based on the Logix
5000 operating system.

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

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

These documents contain additional information concerning related
Rockwell Automation products.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logix 5000 Controllers Program Parameters Programming Manual, publication 1756-PM021</td>
<td>Describes how to use program parameters when programming Logix 5000 controllers.</td>
</tr>
<tr>
<td>Logix 5000 Controllers General Instructions Reference Manual, publication 1756-RM003</td>
<td>Describes the available instructions for a Logix 5000 controller.</td>
</tr>
</tbody>
</table>
Logix 5000 Controllers Process and Drives Instructions Reference Manual, publication 1756-RM006
Describes how to program a Logix 5000 controller for process or drives applications.

Logix 5000 Controllers Motion Instruction Set Reference Manual, publication MOTION-RM002
Describes how to program a Logix 5000 controller for motion applications.

Product Certifications website, http://ab.rockwellautomation.com
Provides declarations of conformity, certificates, and other certification details.

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

Legal notices

Rockwell Automation publishes legal notices, such as privacy policies, license agreements, trademark disclosures, and other terms and conditions on the Legal Notices page of the Rockwell Automation website.

End User License Agreement (EULA)

You can view the Rockwell Automation End User License Agreement (EULA) by opening the license.rtf file located in your product’s install folder on your hard drive.

The default location of this file is:

C:\Program Files (x86)\Common Files\Rockwell\license.rtf.

Open Source Software Licenses

The software included in this product contains copyrighted software that is licensed under one or more open source licenses.

You can view a full list of all open source software used in this product and their corresponding licenses by opening the oss_license.txt file located your product’s OPENSOURCE folder on your hard drive. This file is divided into these sections:

- **Components**
  Includes the name of the open source component, its version number, and the type of license.

- **Copyright Text**
  Includes the name of the open source component, its version number, and the copyright declaration.
• Licenses
  Includes the name of the license, the list of open source components
citing the license, and the terms of the license.

The default location of this file is:

C:\Program Files (x86)\Common Files\Rockwell\Help\<product
name>\Release Notes\OPENSOURCE\oss_licenses.txt.

You may obtain Corresponding Source code for open source packages
included in this product from their respective project web site(s).
Alternatively, you may obtain complete Corresponding Source code by
contacting Rockwell Automation via the Contact form on the Rockwell
Automation website: http://www.rockwellautomation.com/global/about-
us/contact/contact.page. Please include "Open Source" as part of the request
text.
Program ladder diagram

Introduction

You organize ladder diagram as rungs on a ladder and put instructions on each rung. There are two basic types of instructions:

• **Input instruction**: An instruction that checks, compares, or examines specific conditions in your machine or process.
• **Output instruction**: An instruction that takes some action, such as turn on a device, turn off a device, copy data, or calculate a value.

Instruction

A branch is two or more instructions in parallel.

Branch

There is no limit to the number of parallel branch levels that you can enter. This example shows a parallel branch with five levels. The main rung is the first branch level, followed by four additional branches.
You can nest branches to as many as 6 levels. This example shows a nested branch. The bottom output instruction is on a nested branch that is three levels deep.

Large rungs with complex, nested branches result in having to scroll through the ladder editor and may end up spanning multiple pages when you print the logic. To make it easier to maintain, divide the logic into multiple smaller rungs.

Rung condition

The controller evaluates ladder instructions based on the rung condition preceding the instruction (rung-condition-in).

Only create instructions that affect the rung-condition-in of subsequent instructions on the rung.

- If the rung-condition-in to an input instruction is true, the controller evaluates the instruction and sets the rung-condition-out to match the results of the evaluation.
- If the instruction evaluates to true, the rung-condition-out is true.
- If the instruction evaluates to false, the rung-condition-out is false.
- An output instruction does not change the rung-condition-out.
  - If the rung-condition-in to an output instruction is true, the rung-condition-out is set to true.
  - If the rung-condition-in to an output instruction is false, the rung-condition-out is set to false.

Write ladder logic

Writing ladder logic requires that you choose the input and output instructions, and choose the tag names for operands.
Choose the required instructions

Use these steps to choose the required instructions.

To choose the required instructions

1. Identify the conditions to check and separate them from the action to take for the rung.
2. Choose the appropriate input instruction for each condition and the appropriate output instruction for each action.

For more information on specific instructions, see:

- Logix 5000 Controllers General Instructions Reference Manual, publication 1756-RM003
- Logix 5000 Controllers Process and Drives Instructions Reference Manual, publication 1756-RM006
- Logix 5000 Controllers Motion Instruction Set Reference Manual, publication MOTION-RM002

Tip: I/O module data updates asynchronously to the execution of logic. If you reference an input multiple times in your logic, the input could change state between separate references. If you need the input to have the same state for each reference, buffer the input value and reference that buffer tag.

You can also use Input and Output program parameters which automatically buffer the data during the Logix Designer application execution. See the Logix 5000 Controllers Program Parameters Programming Manual, publication 1756-PM021.

The examples in this chapter use two simple instructions to help you learn how to write ladder diagram logic. The rules that you learn for these instructions apply to all other instructions.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Examine If Closed" /></td>
<td>Examine If Closed</td>
<td>XIC</td>
<td>An input instruction that looks at one bit of data. If the bit is On (1) Then the instruction (rung-condition-out) is True</td>
</tr>
<tr>
<td><img src="image" alt="Output Energize" /></td>
<td>Output Energize</td>
<td>OTE</td>
<td>An output instruction that controls one bit of data. If the instructions to the left (rung-condition-in) are True Then the instruction turns the bit On (1)</td>
</tr>
</tbody>
</table>

Arrange the input instructions

Determine how to arrange the input instructions on the rung, as shown below.
To check multiple input conditions when:

- All conditions must be met in order to take action.
  For example, if condition_1 AND condition_2 AND condition_3...

Arrange the input instructions:

In series:

```
  condition_1  condition_2  condition_3
```

- Any one of several conditions must be met in order to take action.
  For example, if condition_1 OR condition_2 OR condition_3...

In parallel:

```
condition_1

  condition_2

  condition_3
```

- There is a combination of the above.
  For example:
  If condition_1 AND condition_2...
  OR
  If condition_3 AND condition_2...

In combination:

```
condition_1

  condition_2

  condition_3
```

Tip: The controller executes all instructions on a rung regardless of their rung-condition-in. For optimal performance of a series of instructions, sequence the instructions from most likely to be false on the left to least likely to be false on the right.

```
Instruction most likely to be FALSE

Instruction least likely to be FALSE
```

When the controller finds a false instruction, it executes the remaining instructions in the series with their rung-condition-in set to false. Typically, an instruction executes faster when its rung-condition-in (rung) is false rather than true.

### Arrange the output instructions

Place at least one output instruction to the right of the input instructions. You can enter multiple output instructions on a rung of logic:

<table>
<thead>
<tr>
<th>Option</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place the output instructions in sequence on the rung (serial).</td>
<td>![Diagram]</td>
</tr>
<tr>
<td>Place the output instructions in branches (parallel).</td>
<td>![Diagram]</td>
</tr>
</tbody>
</table>
### Choose a tag name for an operand

Tag names follow these formats:

<table>
<thead>
<tr>
<th>For a</th>
<th>Specify</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>tag_name</td>
</tr>
<tr>
<td>Bit number of a larger data type</td>
<td>tag_name.bit_number</td>
</tr>
<tr>
<td>Member of a structure</td>
<td>tag_name.member_name</td>
</tr>
<tr>
<td>Element of a one dimension array</td>
<td>tag_name[x]</td>
</tr>
<tr>
<td>Element of a two dimension array</td>
<td>tag_name[x,y]</td>
</tr>
<tr>
<td>Element of a three dimension array</td>
<td>tag_name[x,y,z]</td>
</tr>
<tr>
<td>Element of an array within a structure</td>
<td>tag_name.member_name[x]</td>
</tr>
<tr>
<td>Member of an element of an array</td>
<td>tag_name[x,y,z].member_name</td>
</tr>
</tbody>
</table>

where:

- \( x \) is the location of the element in the first dimension.
- \( y \) is the location of the element in the second dimension.
- \( z \) is the location of the element in the third dimension.

For a structure within a structure, add .member_name.

#### Example: Choose a Tag Name for an Operand

<table>
<thead>
<tr>
<th>To Access</th>
<th>The tag name looks like this</th>
</tr>
</thead>
<tbody>
<tr>
<td>machine_on tag</td>
<td>machine_on</td>
</tr>
<tr>
<td>bit number 1 of the one_shots tag</td>
<td>one_shots.1</td>
</tr>
<tr>
<td>DN member (bit) of the running_seconds timer</td>
<td>running_seconds.DN</td>
</tr>
<tr>
<td>mix member of the north_tank tag</td>
<td>north_tank.mix</td>
</tr>
<tr>
<td>element 2 in the recipe array and element 1,1 in the tanks array</td>
<td></td>
</tr>
<tr>
<td>element 2 in the preset array within the north_tank tag</td>
<td></td>
</tr>
<tr>
<td>part_advance member of element 1 in the drill array</td>
<td></td>
</tr>
</tbody>
</table>

---

Chapter 1  Program ladder diagram

Place the output instructions between input instructions. The last instruction on the rung must be an output instruction.
Enter ladder logic

A new routine contains a rung that is ready for instructions.

Use the Language Element toolbar to add a ladder diagram element.

Append an element to the cursor location

Use these steps to append an element to the cursor location.

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

To append an element to the cursor location

1. Click to select the instruction, branch, or rung that is above or to the left of where you want to add an element.
2. On the Language Element toolbar, click the button for the element that you want to add.
Drag and drop an element

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

Assign instruction operands

After you add an instruction to a ladder rung, you assign tags to the instruction operands. You can create a new tag, use an existing tag, or assign a constant value.

Create and assign a new tag

Use these steps to create and assign a new tag.

To create and assign a new tag

1. Click the operand area of the instruction.
   ![Operand area click]

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

3. Right-click the tag name and then click New "tag_name".

4. In the New Parameter or Tag dialog box, in the Usage box, choose the usage.
   ![New Parameter or Tag dialog box]

5. In the New Parameter or Tag dialog box, in the Data Type box, click the button.
6. In the **Select Data Type** dialog box, choose the data type for the tag.

   ![Select Data Type dialog box](image)

   If you want to define the tag as an array, in the **Array Dimensions** boxes, enter the number of elements in each dimension.

7. Click **OK**.

8. In the **New Parameter or Tag** dialog box, choose the scope for the tag.

9. Click **OK**.

### Choose a name or an existing tag

1. Double-click the operand area, and then click "." The Tag Browser window appears.

2. Select the name or tag:

<table>
<thead>
<tr>
<th>To select a:</th>
<th>Do this:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label, routine name, or similar type of name</td>
<td>Click the name.</td>
</tr>
<tr>
<td>Tag</td>
<td>Double-click the tag name.</td>
</tr>
<tr>
<td>Bit number</td>
<td>a. Click the tag name.</td>
</tr>
<tr>
<td></td>
<td>b. To the right of the tag name, click &quot;.&quot;</td>
</tr>
<tr>
<td></td>
<td>c. Click the required bit.</td>
</tr>
</tbody>
</table>

3. Press the **Enter** key or click a different spot on the ladder diagram to close the Tag Browser.

### Drag and drop a tag from the Tags window

1. Find the tag in the **Controller Tags** or the **Program Parameters and Local Tags** window.

2. Double-click the tag to select it.

3. Click and drag the tag to its location on the instruction. A green dot appears to show you where you can drop the tag.
Assign an immediate (constant) value

1. Click the operand area of the instruction.
2. Type the value and press the Enter key.

Enter a rung comment

When entering a rung of ladder logic, you can add comments that explain the purpose of your rung.

To enter a rung comment, perform this procedure.

1. Right-click the rung number of your ladder logic and then click Edit Rung Comment.

The Rung Comment dialog box appears.

2. Type your rung comment, and then click the green check to save your changes or click the red X to discard your changes.

Language switching

With version 17 and later of the application, you have the option to display project documentation, such as tag descriptions and rung comments for any supported localized language. You can store project documentation for multiple languages in a single project file rather than in language-specific project files. You define all the localized languages that the project supports.
and set the current, default, and optional custom localized language. The application 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 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, you can dynamically switch between languages.

Project documentation that supports multiple languages 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.

**Verify the routine**

As you program your routine(s), it is a good idea to periodically verify your work.

1. In the **Standard** toolbar click the **Verify** icon.
2. Errors are listed in the **Output** window on the **Errors** tab at the bottom of the application.
   a. To go to the first error or warning, press the **F4** key.
   b. Correct the error according to the description in the **Errors** tab.
   c. Repeat until you have corrected all errors.
3. To close the **Output** window, press the **Alt+1** keys.
Index

A
address 13

B
branch
  ladder logic 11

C
create
tag 17

I
immediate value
  ladder logic 19

L
ladder logic
  arrange input instructions 13
  arrange output instructions 14
  assign immediate value 19
  branch 11
  develop 12
  rung condition 12

N
name
tag name 15

R
routine
  verify 20
  rung condition 12

T
tag
  assign 17
  choose name 15
  create 17

enter 17
ladder logic 15, 17

V
verify
  routine 20
Rockwell Automation support

Use these resources to access support information.

<table>
<thead>
<tr>
<th>Technical Support Center</th>
<th>Find help with how-to videos, FAQs, chat, user forums, and product notification updates.</th>
<th>rok.auto/support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledgebase</td>
<td>Access Knowledgebase articles.</td>
<td>rok.auto/knowledgebase</td>
</tr>
<tr>
<td>Local Technical Support Phone Numbers</td>
<td>Locate the telephone number for your country.</td>
<td>rok.auto/phonesupport</td>
</tr>
<tr>
<td>Literature Library</td>
<td>Find installation instructions, manuals, brochures, and technical data publications.</td>
<td>rok.auto/literature</td>
</tr>
<tr>
<td>Product Compatibility and Download Center (PCDC)</td>
<td>Get help determining how products interact, check features and capabilities, and find associated firmware.</td>
<td>rok.auto/pcdc</td>
</tr>
</tbody>
</table>

Documentation feedback

Your comments help us serve your documentation needs better. If you have any suggestions on how to improve our content, complete the form at rok.auto/docfeedback.

Waste Electrical and Electronic Equipment (WEEE)

At the end of life, this equipment should be collected separately from any unsorted municipal waste.

Rockwell Automation maintains current product environmental information on its website at rok.auto/pec.

rockwellautomation.com —— expanding human possibility

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA. Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleielaan 12a. 1831 Diegem, Belgium. Tel: (32) 2 863 0600. Fax: (32) 2 863 0840
ASIA PACIFIC: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong. Tel: (852) 2887 4788. Fax: (852) 2808 1846

Copyright © 2020 Rockwell Automation Technologies, Inc. All Rights Reserved. Printed in the U.S.A.