PhaseManager Software
1756 ControlLogix, 1769 CompactLogix, 1789 SoftLogix, 1794 FlexLogix, 20D PowerFlex 700S with DriveLogix
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.

**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.

**IMPORTANT** Identifies information that is critical for successful application and understanding of the product.
# Table of Contents

## Preface
- Summary of Changes .............................................. 5
- Additional Resources ............................................. 5
- Purpose of This Manual .......................................... 6
- Who Should Use This Manual ..................................... 7
- How To Use This Manual ........................................... 7

## Chapter 1
**Introduction**
- PhaseManager Overview .......................................... 9
- State Model Overview ............................................. 12
- Equipment States .................................................. 13
- State Transitions .................................................. 14
- Manually Change State ............................................ 15
- Ownership ........................................................... 15
- Comparison of Other State Models ............................... 16

## Chapter 2
**PhaseManager Quick Start**
- Purpose of This Chapter .......................................... 17
- Equipment .......................................................... 17
- Create an Equipment Phase ....................................... 18
- Create a State Routine ............................................. 18
- Manually Step Through the States ............................... 19
- Configure the Initial State for an Equipment Phase ....... 22

## Chapter 3
**Guidelines**
- Purpose of This Chapter .......................................... 23
- Equipment Model Guidelines ..................................... 24
  - Example 1: Tank .................................................. 25
  - Example 2: Smart Belt ........................................... 25
- State Model Guidelines ............................................ 26
  - State Model Worksheet ......................................... 28
  - Example 1: Add Water .......................................... 29
  - Example 2: Space Parts ......................................... 30
- Equipment Code Guidelines ...................................... 31
  - Example 1: Add Water to a Tank .............................. 32
  - Example 2: Smart Belt .......................................... 33
- Execution Guidelines ............................................ 34
  - Example 1: Add Water to a Tank .............................. 39
  - Example 2: Smart Belt .......................................... 40
- Transition Guidelines ............................................ 41
  - Example 1: Tank .................................................. 45
  - Example 2: Smart Belt .......................................... 46
  - Example 3: Jam Detection ........................................ 47
- State Completion Guidelines .................................... 48
  - Example 1: Add Water to a Tank .............................. 50
  - Example 2: Smart Belt .......................................... 50
# Table of Contents

Equipment Interface Tag Guidelines ........................................... 51  
  Additional Resources .......................................................... 52  
  Example 1: Add Water to a Tank ............................................ 53  
  Example 2: Smart Belt .......................................................... 54  
  Example 2: Smart belt, Continued ......................................... 55  
Alias Tag Guidelines ............................................................... 56  
  Example ........................................................................... 56  
  Additional Resources ........................................................... 56  

## Appendix A

**PHASE Data Type**  
  Introduction ........................................................................... 57  
  Set and Clear Equipment Phase Tag Values ......................... 57  
  PHASE Data Type ................................................................. 58

## Appendix B

**Configure an Equipment Phase**  
  Introduction ........................................................................... 63  
  Open the Configuration for an Equipment Phase ................. 63  
  Equipment Phase Settings ...................................................... 64

**Glossary** .............................................................................. 67

**Index** .................................................................................. 69
Preface

Summary of Changes

This revised document removes Equipment Phase instructions and updates cross-references to the Logix5000™ Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006.

For the latest compatible software information, see the Product Compatibility and Download Center at http://www.rockwellautomation.com/rockwellautomation/support/pcdc.

Additional Resources

These documents contain additional information that concern-related products from Rockwell Automation.

Table 1 - Additional Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logix5000 Controllers Quick Start, publication 1756-QS001</td>
<td>Get started with a Logix5000 controller</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>PhaseManager™ User Manual, publication LOGIX-UM001</td>
<td>• Use equipment phases&lt;br&gt;• Configure a state model for your equipment&lt;br&gt;• Program in a way that is similar to S88 and PackML models</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 Instructions Reference Manual, publication MOTION-RM002</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>Converting PLC-5 or SLC 500 Logix to Logix5550® Logic Reference Manual, publication 1756-RM085</td>
<td>Convert a PLC-5® or SLC™ 500 application to a Logix5000 project</td>
</tr>
</tbody>
</table>
Purpose of This Manual

This manual shows you how to configure and program a Logix5000 controller to use equipment phases. It gives you guidance and examples to:

- Lay out your code in sections that include equipment phases.
- Configure a state model for your equipment.
- Program your equipment to run by the state model.
- Use equipment phase instructions to transition to another state, handle faults, create break points, and so forth.

A Logix5000 controller is any of the following:

- 1756 ControlLogix controllers
- 1769 CompactLogix controllers
- 1789 SoftLogix5800 controllers
- 1794 FlexLogix controllers
- 20D PowerFlex 700S with DriveLogix controllers

Table 1 - Additional Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompactLogix™ Controllers User manual, publication 1769-UM007</td>
<td>Use a specific Logix5000 controller</td>
</tr>
<tr>
<td>ControlLogix System User Manual, publication 1756-UM001</td>
<td></td>
</tr>
<tr>
<td>DriveLogix™ System 5720 User Manual, publication 20D-UM002</td>
<td></td>
</tr>
<tr>
<td>DriveLogix5730 Controller for PowerFlex® 700S Drives with Phasell Control User Manual, publication 20D-UM003</td>
<td></td>
</tr>
<tr>
<td>FlexLogix™ Controllers User Manual, publication 1794-UM001</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>
Who Should Use This Manual

This manual is for employees who program or maintain industrial automation systems.

To use this manual, you must already have experience with the following:

- Programmable controllers
- Industrial automation systems
- Personal computers

How To Use This Manual

As you use this manual, text that is courier identifies information that you must supply based on your application (a variable). For example, ‘Right-click name_of_program ...’ means that you must identify the specific program in your application. Typically, it is a name or variable that you have defined.
Notes:
PhaseManager Overview

PhaseManager™ software adds equipment phases to a controller. An equipment phase makes it easier to write, use, and manage the code for your machine or equipment.

An **equipment phase** directs one activity of your equipment. A **state model** divides the activity into a set of states that have specific transitions.

**Equipment phase instructions** control the transitions between states, handle faults, and so forth.

- PSC
- POVR
- PCLF
- PRNP
- PATT
- PCMD
- PFL
- PXRQ
- PPD
- PDET

**Other code** does the specific actions of your equipment.

**A PHASE tag** gives you the status of an equipment phase.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add_Water</td>
<td>PHASE</td>
</tr>
<tr>
<td>Add_Water.State</td>
<td>DINT</td>
</tr>
<tr>
<td>Add_Water.Running</td>
<td>BOOL</td>
</tr>
<tr>
<td>Add_Water.Holding</td>
<td>BOOL</td>
</tr>
<tr>
<td>Add_Water.Pumping</td>
<td>BOOL</td>
</tr>
</tbody>
</table>

**Controller Tags**

- Controller
- Controller Tags
- Tasks
- MainTask
- Add_Water_Phase
- Mix_Phase
- Drain_Phase
- Space_Parts_Phase
- MainProgram
- My_Equipment_Program
PhaseManager software helps you write the code for your equipment in a structured way. This structure results in the same behavior for all equipment across a plant.

### Table 2 - PhaseManager Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Equipment phase**      | An equipment phase is similar to a program:  
  - You run the equipment phase in a task.  
  - You give the equipment phase a set of routines and tags.  
  An equipment phase differs from a program in these ways:  
  - The equipment phase uses a state model.  
  - Use an equipment phase to do one activity of your equipment.                                                                                     |
| **State model**          | A state model divides the operating cycle of your equipment into a set of states. Each state is an instant in the operation of the equipment. It’s the actions or conditions of the equipment at a given time.  
  The state model of an equipment phase is similar to these state models:  
  - U.S. standard ISA S88.01-1995 and its IEC equivalent IEC 61512-1-1998, commonly referred to as S88  
  - PackML, which was previously under the supervision of OMAC but is now a working group within ISA                                                                 |
| **State machine**        | The controller has an embedded state machine for the equipment phase. This machine makes it a lot easier to use the state model. The state machine:  
  - Calls the main routine (state routine) for an acting state.  
  - Manages the transitions between states with minimal coding.  
  You code the transition conditions. When the conditions are true, the state machine transitions the equipment to the next required state.  
  - Makes sure that the equipment goes from state to state along an allowable path.  
  For example, if the equipment is in the Complete or Stopped state, the equipment phase makes sure that it goes only to the Resetting state. This functionality simplifies the amount of interlocking that you have to do. |
| **Equipment phase instructions** | Specific instructions that you use to control an equipment phase. See Logix5000™ Controllers Advanced Process Control and Drives Instructions Reference Manual, Publication 1756-RM006. |
| **PHASE tag**            | When you add an equipment phase, RSLogix 5000® software makes a tag for the equipment phase. The tag uses the PHASE data type. Use the tag to:  
  - See which state the equipment phase is in.  
  - Hold a failure code for the equipment phase.  
  - Hold an index for your steps.  
  - Hold the unit ID.  
  - See the status of an external request to FactoryTalk® Batch software.  
  - See if FactoryTalk Batch software has new parameters for the equipment phase.  
  - Create producing and standby states.  
  See Appendix A for more information about the PHASE data type.                                                                                         |
# PhaseManager Questions and Answers

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>How can I get the highest performance possible from my equipment?</td>
<td>You have to measure equipment performance to improve it. The state model gives you a way to measure the status of your equipment. With that data, you can calculate the efficiency and performance measures that you want. If you use PhaseManager software across your plant, you have consistent data from equipment to equipment.</td>
</tr>
<tr>
<td>How can I cut the cost of integrating my equipment into the plant?</td>
<td>Clear structure and consistent tags make it a lot easier to plug the equipment into your plant and configure communication right away. Equipment up and down that line share data that uses the same tag names. And all equipment communicates with higher-level systems in the same way.</td>
</tr>
<tr>
<td>How can I make it easier to maintain the code?</td>
<td>A state model helps you lay out the general functions of your equipment. We found that programmers prefer a state model as the heart of their code. A state model serves as a map for the code. With a clear structure, you know just where to look for the piece of code that you want.</td>
</tr>
<tr>
<td>How can I give my operators a clean, intuitive HMI?</td>
<td>A state model lets you make all your equipment behave the same. Your HMIs can then show consistent equipment conditions across the plant. When an HMI says that the equipment is in an idle, run, or hold state, your operators know exactly what the message means.</td>
</tr>
</tbody>
</table>
State Model Overview

A state model divides the operating cycle of your equipment into a series of states. Each state is an instant in the operation of the equipment. It's the actions or conditions of the equipment at a given time.

In a state model, you define what your equipment does under different conditions, such as run, hold, and stop. You are not required to use all states for your equipment. Use only the states that you want.

There are two types of states.

<table>
<thead>
<tr>
<th>Type of State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acting</td>
<td>Does something or several things for a certain time or until certain conditions are met. An acting state runs one time or repeatedly.</td>
</tr>
<tr>
<td>Waiting</td>
<td>Shows that certain conditions are met and the equipment is waiting for the signal to go to the next state.</td>
</tr>
</tbody>
</table>

PhaseManager software uses the following states.

Your equipment can go from any state in the box to the stopping or aborting state.

- **Acting**: Acting states represent the things that your equipment does at a given time.
- **Waiting**: Waiting states represent the condition of your equipment when it is in-between acting states.
One common objection to a state model is that it doesn’t fit all equipment. You could hear or think: ‘My equipment is complex. There's much synchronization and many things happen in parallel.’

Keep in mind that a state model views your equipment at a general level. Different equipment does different things and needs specific code for everything it does. A state model simply gives you a higher-level framework for your code.

- The state model defines the general behavior, commands, and status of the equipment.
- You program the details of the equipment within that framework.

**Equipment States**

The use of a state model can sound like a significant change for programmers. But it simply represents another way to view the same control problem.

With a state model, you define the behavior of your equipment and put it into a brief functional specification. In this way, you show what happens and when it happens.

<table>
<thead>
<tr>
<th>For this State</th>
<th>Ask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stopped</td>
<td>What happens when you turn on power?</td>
</tr>
<tr>
<td>Resetting</td>
<td>How does the equipment get ready to run?</td>
</tr>
<tr>
<td>Idle</td>
<td>How do you tell that the equipment is ready to run?</td>
</tr>
<tr>
<td>Running</td>
<td>What does the equipment do to make product?</td>
</tr>
<tr>
<td>Holding</td>
<td>How does the equipment temporarily stop the production of product without making scrap?</td>
</tr>
<tr>
<td>Held</td>
<td>How do you tell if the equipment is safely holding?</td>
</tr>
<tr>
<td>Restarting</td>
<td>How does the equipment resume production after holding?</td>
</tr>
<tr>
<td>Complete</td>
<td>How do you tell when the equipment is done with what it had to do?</td>
</tr>
<tr>
<td>Stopping</td>
<td>What happens during a normal shutdown?</td>
</tr>
<tr>
<td>Aborting</td>
<td>How does the equipment shutdown if a fault or failure happens?</td>
</tr>
<tr>
<td>Aborted</td>
<td>How do you tell if the equipment is safely shut down?</td>
</tr>
</tbody>
</table>
## State Transitions

The arrows in the state model show to which states your equipment can go from the state it is in now.

- Each arrow is called a transition.
- A state model lets the equipment make only certain transitions. This functionality gives the equipment the same behavior as any other equipment that uses the same model.

PhaseManager software uses the following transitions.

### Type of Transition | Description
--- | ---
**Command** | A command tells the equipment to start doing something or do something different. For example, the operator pushes the start button to start production and the stop button to shut down. PhaseManager software uses these commands:
- Reset
- Stop
- Restart
- Start
- Hold
- Abort

**Done** | Equipment goes to a waiting state when it’s done with what it’s doing. You don’t give the equipment a command. Instead, you design your code to signal when the equipment is done. The waiting state shows that the equipment is done.

**Fault** | A fault tells you that something out of the ordinary has happened. You design your code to look for faults and act if it finds any. Suppose that you want your equipment to shut down as fast as possible if a certain fault happens. In that case, design your code to look for that fault and give the abort command if it finds it.
Manually Change State

RSLogix 5000 software has a window that lets you monitor and command an equipment phase.

State that the equipment phase is in right now

To change states manually.

1. Take ownership of the equipment phase.

2. Give a command.

Ownership

Ownership locks out programs or FactoryTalk Batch software from giving commands to an equipment phase.

<table>
<thead>
<tr>
<th>If this component owns the equipment phase</th>
<th>Then</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSLogix 5000 software</td>
<td>Sequencers can’t give commands to the equipment phase. These sequencers include:</td>
</tr>
<tr>
<td></td>
<td>• Internal sequencer — program in the controller.</td>
</tr>
<tr>
<td></td>
<td>• External sequencer — FactoryTalk Batch software.</td>
</tr>
<tr>
<td>Internal sequencer — program in the controller</td>
<td>Other sequencers can’t give commands to the equipment phase.</td>
</tr>
<tr>
<td>External sequencer — FactoryTalk Batch software</td>
<td>Other sequencers can’t give commands to the equipment phase.</td>
</tr>
</tbody>
</table>

Exception: Use an Equipment Phase Override Command (POVR) instruction to give a hold, stop, or abort command regardless of ownership.
Comparison of Other State Models

The following table compares the PhaseManager software state model to other common state models.

<table>
<thead>
<tr>
<th>S88</th>
<th>Pack ML</th>
<th>PhaseManager Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>Starting &gt; Ready</td>
<td>Resetting &gt; Idle</td>
</tr>
<tr>
<td>Running &gt; Complete</td>
<td>Producing</td>
<td>Running &gt; Complete</td>
</tr>
<tr>
<td>Pausing &gt; Paused</td>
<td>Standby</td>
<td>Subroutines, breakpoints, or both.</td>
</tr>
<tr>
<td>Holding &gt; Held</td>
<td>Holding &gt; Held</td>
<td>Holding &gt; Held</td>
</tr>
<tr>
<td>Restarting</td>
<td>None</td>
<td>Restarting</td>
</tr>
<tr>
<td>Stopping &gt; Stopped</td>
<td>Stopping &gt; Stopped</td>
<td>Stopping &gt; Stopped</td>
</tr>
<tr>
<td>Aborting &gt; Aborted</td>
<td>Aborting &gt; Aborted</td>
<td>Aborting &gt; Aborted</td>
</tr>
</tbody>
</table>
PhaseManager Quick Start

Purpose of This Chapter

Use this quick start to:

- Get an introduction to how an equipment phase runs.
- Monitor an equipment phase.
- Manually tell an equipment phase to go to another state.

Use this quick start when you want to:

- Try out PhaseManager™ software for the first time.
- Test an equipment phase by manually stepping through its states.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an Equipment Phase</td>
<td>18</td>
</tr>
<tr>
<td>Create a State Routine</td>
<td>18</td>
</tr>
<tr>
<td>Manually Step Through the States</td>
<td>19</td>
</tr>
<tr>
<td>Configure the Initial State for an Equipment Phase</td>
<td>22</td>
</tr>
</tbody>
</table>

Equipment

To use this quick start, you need:

- A Logix5000™ controller. See the preface if you aren’t sure which controllers are Logix5000 controllers.
- Firmware, revision 18.0 or later, for the controller
- A power supply for the controller
- A communication path to the controller:
  - Communication card or built-in port
  - Corresponding communication cable
- RSLogix 5000® software, version 18.0 or later
Create an Equipment Phase

1. 
2. 
3. 

Create a State Routine

1. 
2. 
3. 
4.
### Manually Step Through the States

Before you do this procedure, do the following:

- Download the project to the controller.
- Put the controller in Run or Remote Run mode.

<table>
<thead>
<tr>
<th>Step</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Right-click the equipment phase and choose Monitor Equipment Phase.</td>
<td></td>
</tr>
<tr>
<td>2. Click the ownership button and then Yes—take ownership.</td>
<td>This action lets you use this window to step through the states.</td>
</tr>
</tbody>
</table>
| 3. Click Start. | • The equipment phase goes to the Running state.  
• Any code in the Running state routine starts running. This routine is where you put the code for the normal production sequence of your equipment. |
### Step Notes

**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 the ownership button. This action 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 power-up.

1. Choose your initial state.

2. [Diagram showing the interface for setting initial state]

3. Choose your initial state.

4. [Diagram showing the interface for setting initial state]
Chapter 3

Guidelines

Purpose of This Chapter

This chapter guides your development and programming of a Logix5000™ project that uses equipment phases. Use the procedures for the following:

- Before you lay out the equipment phases for your Logix5000 project.
- As a reference while you program the project.

Review the following guidelines before you lay out your project. Refer to these guidelines as needed.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment Model Guidelines</td>
<td>24</td>
</tr>
<tr>
<td>State Model Guidelines</td>
<td>26</td>
</tr>
<tr>
<td>Equipment Code Guidelines</td>
<td>31</td>
</tr>
<tr>
<td>Execution Guidelines</td>
<td>34</td>
</tr>
<tr>
<td>Transition Guidelines</td>
<td>40</td>
</tr>
<tr>
<td>State Completion Guidelines</td>
<td>47</td>
</tr>
<tr>
<td>Equipment Interface Tag Guidelines</td>
<td>50</td>
</tr>
<tr>
<td>Alias Tag Guidelines</td>
<td>55</td>
</tr>
</tbody>
</table>
Equipment Model Guidelines

Each equipment phase is a specific activity that your equipment does. An equipment phase tells the equipment what to do and when to do it.

Follow these guidelines to decide how many equipment phases to use.

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make sure that each equipment phase does an independent activity.</td>
<td>Make sure that each equipment phase does an activity that is independent (relatively independent) from other equipment. The equipment phase commands all equipment that works together to do the specific activity.</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>This activity is probably an equipment phase</th>
<th>This activity is probably NOT an equipment phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Fill bottles with product.</td>
<td>• Accelerate filler axis (too small)</td>
</tr>
<tr>
<td>• Put bottles in carton.</td>
<td>• Run bottling line (too large)</td>
</tr>
<tr>
<td>• Add water to a tank.</td>
<td>• Open water valve (too small)</td>
</tr>
<tr>
<td>• Mix ingredients in tank</td>
<td>• Brew ingredients (too large)</td>
</tr>
</tbody>
</table>

Keep the number of equipment phases and programs within the following limits.

<table>
<thead>
<tr>
<th>If you have this controller</th>
<th>You can have up to</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlLogix®</td>
<td>100 programs and equipment phases per task</td>
</tr>
<tr>
<td>SoftLogix™</td>
<td>100 programs and equipment phases per task</td>
</tr>
<tr>
<td>FlexLogix™</td>
<td>32 programs and equipment phases per task</td>
</tr>
<tr>
<td>CompactLogix™</td>
<td>32 programs and equipment phases per task</td>
</tr>
</tbody>
</table>

List the equipment that goes along with each equipment phase.

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>This equipment phase</th>
<th>Relates this equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add_Water</td>
<td>Water pump</td>
</tr>
<tr>
<td></td>
<td>Water valve</td>
</tr>
<tr>
<td></td>
<td>Limit switch</td>
</tr>
<tr>
<td>Smart_Belt</td>
<td>Coarse belt axis</td>
</tr>
<tr>
<td></td>
<td>Fine belt axis</td>
</tr>
<tr>
<td></td>
<td>Exit belt axis</td>
</tr>
</tbody>
</table>
Example 1: Tank

This example shows the equipment phases for a tank that cooks ingredients.

To cook the ingredients, the tank completes these steps.

1. Adds water.
2. Heats the water.
3. Adds other ingredients.
4. Mixes all ingredients.
5. Dispenses the finished product.

Which become these phases

Which commands this equipment

Example 2: Smart Belt

This example shows a smart belt. The smart belt does only one activity. It spaces product evenly on an exit belt. Because it does only one activity, it needs only one equipment phase.
State Model Guidelines

A state model divides the operating cycle of your equipment into a series of states. Each state is an instant in the operation of the equipment. It's the actions or conditions of the equipment at a given time.

Follow these guidelines as you fill out the state model for an equipment phase.

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fill out one state model for each phase.</td>
<td>Each phase runs its own set of states. Fill out one state model worksheet for each phase.</td>
</tr>
<tr>
<td>Decide which state you want as your initial state after powerup.</td>
<td>Which state do you want the equipment phase to go to when you turn on power?</td>
</tr>
</tbody>
</table>

An equipment phase goes to its initial state when you turn on power. We recommend that you use one of these states as the initial state:
- Idle (default)
- Complete
- Stopped

Choose the initial state that shows what your equipment is waiting to do after powerup (reset, run, and so forth).
### Guidelines

**Chapter 3**

Start with the initial state and work through the model.

Start with the initial state. Then work forward from that point. Use the following questions to help you.

<table>
<thead>
<tr>
<th>For this State</th>
<th>Ask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stopped</td>
<td>What happens when you turn on power?</td>
</tr>
<tr>
<td>Resetting</td>
<td>How does the equipment get ready to run?</td>
</tr>
<tr>
<td>Idle</td>
<td>How do you tell that the equipment is ready to run?</td>
</tr>
<tr>
<td>Running</td>
<td>What does the equipment do to make product?</td>
</tr>
<tr>
<td>Holding</td>
<td>How does the equipment pause without making scrap?</td>
</tr>
<tr>
<td>Held</td>
<td>How do you tell if the equipment is safely paused?</td>
</tr>
<tr>
<td>Restarting</td>
<td>How does the equipment resume production after a pause?</td>
</tr>
<tr>
<td>Complete</td>
<td>How do you tell when the equipment is done with what it had to do?</td>
</tr>
<tr>
<td>Stopping</td>
<td>What happens during a normal shutdown?</td>
</tr>
<tr>
<td>Aborting</td>
<td>How does the equipment shutdown if a fault or failure happens?</td>
</tr>
<tr>
<td>Aborted</td>
<td>How do you tell if the equipment is safely shut down?</td>
</tr>
</tbody>
</table>

Use only the states that you want. Define only the states that are appropriate for your equipment. You are **not** required to use all states. The equipment phase just skips any states that you don’t add.

For the producing and standby states, use subroutines.

If you want to define producing and standby states for your equipment, use subroutines.

A. Create a routine for the producing state and another routine for the standby state.

B. In the running state, check for the produce versus standby conditions. Set either the Producing bit or the Standby bit of the equipment phase tag.

C. To call the corresponding routine, use the Producing and Standby bits as conditions.

See [Appendix A](#).
State Model Worksheet

Equipment Phase:

IDLE
- Start Command
- Done

RUNNING
- Hold Command
- Done

HOLDING
- Hold Command
- Done

RESTARTING
- Restart Command

HELD

RESETTING
- Restart Command
- Done

COMPLETE
- Reset Command

STOPPING
- Stop Command
- Abort Command
- Done

STOPPED
- Reset Command

ABORTING
- Abort Command
- Done

ABORTED

Key
- Waiting State
- Acting State

Publication LOGIX-UM001C-EN-P - June 2016
Example 1: Add Water

Equipment Phase: *Add Water*

**Key**
- **Waiting State**
- **Acting State**

**Equipment Phase**

- **Idle**
  - No water flow
  - Tank Not full
  - Operator can control equipment

- **Running**
  - Lock equipment in program control
  - Add water
  - Unlock equipment from program control

- **Holding**
  - Stop water
  - Unlock equipment from program control

- **Restarting**
  - Lock equipment in program control

- **Held**
  - No water flow
  - Operator can control equipment

- **Stopping**
  - Stop water
  - Unlock equipment from program control

- **Stopped**
  - No water flow
  - Operator can control equipment

- **Resetting**

- **Complete**
  - No water flow
  - Water at high limit
  - Operator can control equipment

- **Aborting**
  - Stop water
  - Unlock equipment from program control

- **Aborted**
  - No water flow
  - Operator can control equipment

**Waiting State**

- **Running**
- **Holding**
- **Complete**
- **Aborting**
- **Stopped**

**Acting State**

- **Idle**
- **Restarting**
- **Held**
- **Stopping**
- **Resetting**
- **Aborted**

**Examples**

1. **Add Water**

---

Publication LOGIX-UM001C-EN-P - June 2016
Example 2: Space Parts

**Equipment Phase: Space Parts**

<table>
<thead>
<tr>
<th>IDLE</th>
<th>Start Command</th>
<th>RUNNING</th>
<th>Hold Command</th>
<th>HOLDING</th>
<th>Done</th>
<th>HELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>· No axes faults</td>
<td></td>
<td>· Jog exit belt</td>
<td></td>
<td>· Set speed of exit belt = 0</td>
<td></td>
<td>· Speed of exit belt = 0</td>
</tr>
<tr>
<td>· All axes = on</td>
<td></td>
<td>· Gear other belts</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Exit belt = homed</td>
<td></td>
<td>· Put one box on fine belt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Registration = armed</td>
<td></td>
<td>· Put one box on each flight</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESETTING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Clear axes faults</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Turn on all axes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Home exit belt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· Arm registration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOPPING</td>
<td>Stop Command</td>
<td>STOPPED</td>
<td>Abort Command</td>
<td>ABORTED</td>
<td>Done</td>
<td></td>
</tr>
<tr>
<td>· All axes = off</td>
<td></td>
<td>· All axes = off</td>
<td></td>
<td>· All axes = off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>· SERCOS = phased up</td>
<td></td>
<td>· SERCOS = phased up</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>· All network connections are made</td>
<td></td>
<td>· All network connections are made</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key**

- Waiting State
- Acting State
Equipment Code Guidelines

An equipment phase lets you separate the procedures (recipes) for how to make the product from the control of the equipment that makes the product. This advantage makes it much easier to execute different procedures for different products using the same equipment.

**Equipment Phase**

Directs the actions of the equipment (what to do and when)
- Produce product
- Stop producing product
- Add water to a tank
- Wait for the operator to do

**Equipment Program**

Does the actions for a specific group of devices (does it)
- Jog axis
- Run pump
- Open valve
- Calculate control variable

To start the equipment, the equipment phase gives the equipment program a start command, a setpoint, and so forth.

The equipment program sends back information such as current state, total.

The equipment program controls the equipment.
Example 1: Add Water to a Tank

The equipment phase tells the equipment program to go to program mode and add water.

The equipment program follows the commands of the equipment phase and sends back its mode and state.
Example 2: Smart Belt

The equipment phase tells the equipment program to reset faults. The equipment phase then waits for the equipment program to enable a done bit. When the done bit turns on, the equipment phase clears the command to reset faults. The equipment phase then goes to the next step in the sequence.

The equipment program resets faults when it gets the fault reset command from the equipment phase. It turns on a done bit after it clears the faults.
Execution Guidelines

A state model makes it much easier to separate the normal execution of your equipment from any exceptions (faults, failures, off-normal conditions).

Use the resetting, running, and stopping states for the normal execution of the equipment. Use the holding, restarting, and aborting states to handle exceptions (faults, failures, off-normal conditions).

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
</table>
| Use the prestate routine to watch for faults. | ![Diagram of state transitions] Use the prestate routine for conditions that you want to watch all time such as fault bits. The prestate routine:  
- Runs constantly.  
- Runs before each scan of a state.  
- Runs even in the waiting states (idle, held, complete, stopped, or aborted). |
Create a prestate routine just like the routine for a program. It's not a phase state routine.

1. Choose any language.

2. Open Routine

3. Choose any language.

4. New Parameter...
## Chapter 3 Guidelines

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign a prestate routine.</td>
<td></td>
</tr>
<tr>
<td>Use a state bit to limit code to a specific state.</td>
<td>RSLogix 5000® software automatically makes a tag for each equipment phase. The tag has bits that tell you the state of the equipment phase.</td>
</tr>
<tr>
<td>• The tag is at the controller scope.</td>
<td></td>
</tr>
<tr>
<td>• The tag uses the PHASE data type.</td>
<td></td>
</tr>
<tr>
<td>• Use bits of the tag for code that you want to limit to certain states.</td>
<td></td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
<tr>
<td>Suppose that the name of your equipment phase is <em>My_Phase</em>. And you have some code that you want to run only when the equipment phase is in the running state. In that case, check the <em>My_Phase.Running</em> bit for on (1):</td>
<td></td>
</tr>
<tr>
<td><strong>If</strong> <em>My_Phase.Running</em> <strong>then...</strong></td>
<td></td>
</tr>
<tr>
<td>See Appendix A for more information.</td>
<td></td>
</tr>
<tr>
<td>Guideline</td>
<td>Details</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Use the empty phase state routine to complete phase execution.            | Unlike normal program routines, phase state routines are called by the batch manager (not other program routines), so they always have the potential of being called. In the configuration for a phase state routine, if the Complete State Immediately if not Implemented option is checked in RSLogix 5000 programming software, version 18 or later, an implemented, but empty (no logic), phase state routine behaves the same as an implemented phase state routine. The state immediately completes and execution of the phase continues. The phase then enters the next state in the state machine. In RSLogix 5000 programming software, version 16 and earlier, if a phase enters a state for which a state routine exists, but contains no logic, execution of the phase stops. The routine does complete, but there is no logic to execute. Choose from the following if you import a new state routine and in the Online Options dialog box.  
  - Import Logic Edits as Pending, an empty routine is created in the controller and the pending edits exist in the offline project.  
  - Accepts Program Edits, an empty routine is created in the controller, and the logic is placed in a test edits container in the routine. If you are not actively testing edits, then the routine appears as empty when running.  
  - Finalize All Edits in Program, the routine is created with the new logic and does not appear empty. In the first two cases, if the Complete State Immediately if not Implemented option is checked, the empty routine completes immediately and allows phase execution to continue. |
| Use the PFL instruction to signal a fault.                               | The Equipment Phase Failure (PFL) instruction sets a failure code for an equipment phase. Use the code to signal a specific failure such as the fault of a specific device.  
  - The PFL instruction writes a code to the failure member for the equipment phase.  
  - To see the failure code of an equipment phase, review the `phase_name.Failure` tag.  
  - The failure code stays until any of the following happens:  
    - A PFL instruction sets the failure code to a larger number.  
    - The equipment phase transitions from the resetting state to idle state.  
    - A PCLF instruction clears the failure code.  
    - FactoryTalk® Batch software clears the failure code. See publication 1756-RM006 for more information. |
| Use a PCLF instruction to clear a failure code.                         | The Equipment Phase Clear Failure (PCLF) instruction clears the failure code for an equipment phase.  
  - A CLR instruction, MOV instruction, or assignment (`:=`) doesn’t change the failure code of an equipment phase.  
  - If you are testing a PCLF instruction, make sure RSLogix 5000 software doesn’t own the equipment phase. The PCLF instruction doesn’t work if RSLogix 5000 software owns the equipment phase. See publication 1756-RM006 for more information. |
Example 1: Add Water to a Tank

The prestate routine watches for equipment faults while the equipment phase is in the running state \((\text{Add\_Water.Running} = 1)\). If \(\text{Water\_Feed.Health} = 1\), then a fault happened. If a fault happens, the equipment phase sets a failure code of 202.

\[
\text{If Add\_Water.Running And Water\_Feed.Health Then}
\]
\[
PFL(202);
\]
\[
\text{End\_If;}
\]

The equipment program watches the fault bits of the valve, pump, and their feedback devices. If any of that equipment faults, the equipment program turns on the \(\text{Water\_Feed.Health}\) bit.
Example 2: Smart Belt

The preset value of this step = 20000 ms. The step turns on its DN bit if it doesn’t clear the faults within 20000 ms.

If Step_000.DN = on, a timeout happened. When a timeout happens, the OSR instruction turns on the Clear_Faults_Timeout bit for one scan.

If MyPhase is in the resetting state and Clear_Faults_Timeout is on, then the PFL instruction signals a failure. The PFL instruction sets the failure code = 501.
Transition Guidelines

To start an acting state, you usually have to give the equipment phase a command. The command tells the equipment phase and its equipment to start doing something or do something different. Use the Equipment Phase Command (PCMD) instruction to give a command to an equipment phase.

**Optional:** You can also use FactoryTalk Batch software in place of a PCMD instruction to trigger transitions.

Use the state model to see which transitions need a PCMD instruction.
**Type of Transition** | **Description** | **Instruction**
--- | --- | ---
Command | A command tells the equipment to start doing something or do something different. For example, the operator pushes the start button to start production and the stop button to shut down. | PCMD
 PhaseManager software uses these commands:
 Reset | Stop | Restart
 Start | Hold | Abort
 | | Use an Equipment Phase Command (PCMD) instruction to give a command. Or use RSLogix 5000 software. See the Logix5000 Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006 for more information.

Done | Equipment goes to a waiting state when it's done with what it's doing. You configure your code to signal when the equipment is done. The waiting state shows that the equipment is done. **Exception:** The restarting state goes to the running state when it’s done. | PSC
 | | Use the Phase State Complete (PSC) instruction to signal when a state is done. See the Logix5000 Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006 for more information.

**Guideline** | **Details**
--- | ---
A PCMD instruction causes a transition right away. | A PCMD instruction makes an equipment phase go to the commanded state. The equipment phase changes states as soon as it finishes its current scan. This state change happens even if the current state isn’t done.
### Chapter 3  Guidelines

#### Guideline

See if you must reset the state that you’ve left.

#### Details

Are you leaving an acting state (for example, running, holding)?
- **YES** — Consider resetting the code of the state that you’ve left.
- **NO** — You are probably not required to reset anything.

The equipment phase stops running the code of the current state when it goes to another state. Outputs remain at their last values unless the new state takes control of them. The stop also leaves an SFC at the step it was at when the equipment phase changed states.

**Example 1: No reset required**

Suppose that your equipment phase is in the idle state. In that case, it isn’t running any state code. So you are probably not required to reset any state when you go to another state like running or stopping.

**Example 2: No reset required**

Suppose that your equipment phase is in the running state and you go to the holding state. When you return to the running state, you probably want to pick up where you left off. In that case, you are probably not required to reset the code in the running state.

**Example 3: Reset required**

Suppose that your equipment phase is half way through the resetting state and you give the stop command. And suppose that you want to run the entire resetting sequence when you return to it. In that case, you probably must reset the code of the resetting state. If the resetting state uses an SFC, then use the SFR instruction to reset it to the first step.

#### Use an SFR instruction to reset the SFC of a state routine.

An SFC Reset (SFR) instruction is one way to reset an SFC. In some cases, reset an SFC from several other state routines.

<table>
<thead>
<tr>
<th>To reset the SFC of this state</th>
<th>Place an SFR instruction in this state routine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>Resetting</td>
</tr>
<tr>
<td>Holding</td>
<td>Holding—Let the SFC reset itself at the last step.</td>
</tr>
<tr>
<td>Restarting</td>
<td>Reset the restarting routine in both these routines:</td>
</tr>
<tr>
<td></td>
<td>• Holding—in case you return to holding before you finish restarting.</td>
</tr>
<tr>
<td></td>
<td>• Restarting—Let the SFC reset itself at the last step.</td>
</tr>
</tbody>
</table>

#### Use the PCMD instruction to go to an allowed next state.

PhaseManager software makes sure that an equipment phase follows the state model. So the equipment phase goes only to certain states from the state that it is in right now.

**Example 1: A transition is allowed**

Suppose that your equipment phase is in the running state and you give it the hold command. In that case, the equipment phase goes to holding because that transition is allowed.

**Example 2: A transition isn’t allowed**

Suppose that your equipment phase is in the running state and you give it the reset command. In that case, the equipment phase stays in the running state. To go to the resetting state, you first have to stop or abort the equipment phase.
See if you must use a POVR instruction instead of a PCMD instruction.

A. Are you giving the hold, stop, or abort command?
   - NO — Use the PCMD instruction.
   - YES — Go to step B.

B. Must the command work even if you have manual control of the equipment phase via RSLogix 5000 software?
   - YES — Use the POVR instruction instead. See the Logix5000 Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006.
   - NO — Go to step C.

C. Must the command work even if FactoryTalk Batch software or another program owns the equipment phase?
   - YES — Use the POVR instruction instead. See the Logix5000 Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006.
   - NO — Use the PCMD instruction.
Example 1: Tank

The controller uses an SFC to command the phases that run the tank (add water, heat, add ingredients, and so forth).

Give the start command to the \textit{Add\_Water} equipment phase. The P1 qualifier limits this command to the first scan of the step.

Wait until the \textit{Add\_Water} equipment phase is done (complete). When the equipment phase is done, give the reset command. The P0 qualifier limits this action to the last scan of the step.

Start the next equipment phase.
Example 2: Smart Belt

If the operator presses the start button on the machine or HMI, then

\[ My\_Inputs.AnyStartPressed = \text{on} \] for one scan.

The ONS instruction makes sure that \[ My\_Inputs.AnyStartPressed \] turns on only when a start button goes from off to on.

<table>
<thead>
<tr>
<th>StartButtonPanel</th>
<th>My_Inputs.StartOneShot</th>
<th>My_Inputs.AnyStartPressed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StartButtonHMI</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If the equipment phase is in the idle state and \[ My\_Inputs.AnyStartPressed = \text{on} \], then

The PCMD instruction gives \[ MyPhase \] the start command.

<table>
<thead>
<tr>
<th>MyPhase.Idle</th>
<th>My_Inputs.AnyStartPressed</th>
<th>PCMD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Equipment Phase Command</th>
<th>Phase Name</th>
<th>MyPhase Command</th>
<th>Start Result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Example 3: Jam Detection

The equipment program watches for the following faults:
- Faulted axis
- Jammed material

If there is a fault, then

\[ \text{Local\_Interface\_Equipment\_Faults\_Cleared} = 0 \]

This tag is an alias for the controller-scoped tag \text{Shear\_1}.

The prestate routine of the equipment phase watches for the equipment program to signal a fault.
- If \text{Interface\_To\_Equipment\_Equipment\_Faults\_Cleared} = 0 then there is a fault.
- Both \text{Interface\_To\_Equipment} and \text{Local\_Interface} are aliases for \text{Shear\_1}, so they have the same values.

If there is a fault Then

Give the \text{Shear\_One\_Phase} equipment phase the abort command. The POVR instruction makes sure that the command works even if someone has manual control of the equipment phase through RSLogix 5000 software.

The PFL instruction sets the failure code for \text{Shear\_One\_Phase} = 333.

The Fault Strobe keeps these actions to one scan.

<table>
<thead>
<tr>
<th>Interface_To_Equipment_Equipment_Faults_Cleared</th>
<th>Fault Strobe</th>
</tr>
</thead>
<tbody>
<tr>
<td>[]</td>
<td>[ONS]</td>
</tr>
</tbody>
</table>

POVR

<table>
<thead>
<tr>
<th>Equipment Phase Override Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase Name</td>
</tr>
<tr>
<td>Command</td>
</tr>
<tr>
<td>Result</td>
</tr>
</tbody>
</table>

333

(FPL)
State Completion Guidelines

To leave an acting state, you usually signal that the state is done doing what it had to do. Use the Phase State Complete (PSC) instruction to signal when a state is done.

**IMPORTANT**

The PSC instruction *doesn’t* stop the current scan of a routine.

When the PSC instruction executes, the controller scans the rest of the routine and then transitions the equipment phase to the next state. The PSC instruction *does not* terminate the execution of the routine.

Use the state model to see which transitions need a PSC instruction.
<table>
<thead>
<tr>
<th>Type of Transition</th>
<th>Description</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>A command tells the equipment to start doing something or do something</td>
<td>PCMD</td>
</tr>
<tr>
<td></td>
<td>different. For example, the operator pushes the start button to start</td>
<td>Use an Equipment Phase</td>
</tr>
<tr>
<td></td>
<td>production and the stop button to shut down.</td>
<td>Command (PCMD) instruction to give a command. Or use RSLogix 5000 software.</td>
</tr>
<tr>
<td></td>
<td>PhaseManager software Software uses these commands:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reset Stop Restart</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start Hold Abort</td>
<td></td>
</tr>
<tr>
<td>Done</td>
<td>Equipment goes to a waiting state when it’s done with what it’s doing. You</td>
<td>PSC</td>
</tr>
<tr>
<td></td>
<td>configure your code to signal when the equipment is done. The waiting state</td>
<td>Use the Phase State Complete (PSC) instruction to signal when a state is</td>
</tr>
<tr>
<td></td>
<td>shows that the equipment is done.</td>
<td>done. See the Logix5000 Controllers Advanced Process Control and Drives</td>
</tr>
<tr>
<td></td>
<td><strong>Exception.</strong> The restarting state goes to the running state when it's</td>
<td>Instructions Reference Manual, publication 1756-RM006 for more information.</td>
</tr>
</tbody>
</table>
Example 1: Add Water to a Tank

The holding state does three things.

1. Rung 0 — stop the water.
2. Rung 1 — unlock the devices from program control.
3. Rung 2 — signal that the state is done.

Example 2: Smart Belt

At the last step of the resetting state:

- The SFR instruction resets the SFC so it is ready for the next time you need it.
- The PSC instruction signals that the state is done.

**Note:** The P1 qualifier runs the actions only one time.
Equipment Interface Tag Guidelines

An equipment interface tag links an equipment phase to an equipment program.

- The equipment phase uses the tag to configure and command the equipment program.
- The equipment program uses the tag to report its status or condition.

### Equipment Phase
Directs the actions of the equipment (what to do and when)
- Produce product
- Stop producing product
- Add water to a tank
- Wait for the operator to do

### Interface Tags
Links equipment phase to equipment program
- Jog at this speed
- Go to the one state (run pump)
- Axis is jogging
- Valve is faulted

### Equipment Program
Does the actions for a specific group of devices (does it)
- Jog axis
- Run pump
- Open valve
- Calculate control variable

### Guideline Details
List the values that your equipment phase must give to the equipment program or get back from it.

Think of these values as a faceplate to the equipment program. It is the values that your equipment phase uses to control and monitor the equipment program. Exclude I/O data.

#### Inputs to the equipment program
- Mode requests
- Setpoints
- Commands such as on, off, start, stop, reset
- Permissives
- Overrides

#### Outputs from the equipment program
- Mode status
- Control values
- Done or completion
- Alarms
- Faults
- Health indication
- Totals or accumulated values

Create a user-defined data type.

A user-defined data type lets you make a template for your data. It lets you group related data into one data type. You then use the data type to make tags with the same data lay-out.

If you have multiple equipment phases, lay out the data type so that it’s easy to use with multiple equipment phases. Consider the following:

- Include a range of data that makes the data type more versatile.
- Use names that are as general as possible.

**Example:** The name `State_Cmd` lets you use it for any equipment that runs in two states like on/off, running/not running, pumping/not pumping. It is easier to reuse than names such as `Open` or `Close`. Those names apply to valves but not pumps or motors.
Guidelines

Chapter 3

Guideline | Details
--- | ---
Create a tag for each equipment phase | Create tag for the interface data of each equipment phase.

- Make a tag for each equipment phase.
- Use the data type from guideline.
- Make the tag at the controller scope. Both the equipment phase and the equipment program must get to the tag.
- Consider using alias tags. See Alias Tag Guidelines on page 56.

Additional Resources

For this information | See this publication
--- | ---
Guidelines and considerations regarding:  
- User-defined data types  
- Alias tags | Logix5000 Controllers Design Considerations, publication 1756-RM094

Step-by-step procedures on how to:  
- Create user-defined data types  
- Assign alias tags | Logix5000 Controllers Common Procedures Programming Manual, publication 1756-PM001

Provides a description of each instruction in the Logix5000 format | Logix5000 Controllers Advanced Process Control and Drives Instructions Reference Manual, publication 1756-RM006
Example 1: Add Water to a Tank

1. The equipment phase and equipment program share this data.

   - Go to program mode
   - Go to this state
   - In program mode
   - No faults
   - Hardware OK
   - In this state

2. A user-defined data type creates a template for the data.

3. A tag stores the data that the equipment phase and equipment program share. The tag uses the user-defined data type from step 2.
Example 2: Smart Belt

The equipment phase and equipment program share this data.

A separate user-defined data type holds data for each axis.

Equipment program interface

<table>
<thead>
<tr>
<th>Commands</th>
<th>Conditions or status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>Abort</td>
</tr>
<tr>
<td>Disable</td>
<td>FaultReset</td>
</tr>
<tr>
<td>Home</td>
<td>Stop</td>
</tr>
<tr>
<td>ActivateRun</td>
<td>ArmRegistration</td>
</tr>
<tr>
<td>EnableProduct</td>
<td></td>
</tr>
<tr>
<td>DisableProduct</td>
<td></td>
</tr>
<tr>
<td>EnableCycling</td>
<td></td>
</tr>
<tr>
<td>DisableCycling</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Commands</th>
<th>Conditions or status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>FaultScroll</td>
</tr>
<tr>
<td>Disable</td>
<td>Faulted</td>
</tr>
<tr>
<td>Home</td>
<td>EnableDone</td>
</tr>
<tr>
<td>ActivateRun</td>
<td>DisableDone</td>
</tr>
<tr>
<td>EnableProduct</td>
<td>HomeDone</td>
</tr>
<tr>
<td>DisableProduct</td>
<td>StoppingDone</td>
</tr>
<tr>
<td>EnableCycling</td>
<td>EnableProductDone</td>
</tr>
<tr>
<td>DisableCycling</td>
<td>RegistrationArmed</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Commands</th>
<th>Conditions or status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>Abort</td>
</tr>
<tr>
<td>Disable</td>
<td>Stop</td>
</tr>
<tr>
<td>Home</td>
<td>ActivateRun</td>
</tr>
<tr>
<td>AutoRun</td>
<td></td>
</tr>
<tr>
<td>ResetFaults</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Commands</th>
<th>Conditions or status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>State</td>
</tr>
<tr>
<td>Disable</td>
<td>On</td>
</tr>
<tr>
<td>Home</td>
<td>Ok</td>
</tr>
<tr>
<td>AutoRun</td>
<td>Auto</td>
</tr>
<tr>
<td>ResetFaults</td>
<td>Jogging</td>
</tr>
</tbody>
</table>

Axis interface

There is an interface tag for each axis and one for the entire machine.

One tag stores the data that the equipment phase and equipment program share. Other tags store the data for each individual axis.
Example 2: Smart belt, Continued

The equipment program gets the command from the equipment phase and passes it to each axis.

**Routine of the equipment program**

<table>
<thead>
<tr>
<th>This tag</th>
<th>Is the interface between</th>
</tr>
</thead>
<tbody>
<tr>
<td>My_Equipment</td>
<td>Equipment phase and equipment program</td>
</tr>
<tr>
<td>Coarse_Belt_Vars</td>
<td>Equipment program and an axis</td>
</tr>
</tbody>
</table>

The equipment program collects the fault status of each axis and passes it back to the equipment phase.

**Routine of the equipment program**

The equipment program checks the fault code of each axis. If an axis isn’t faulted, the OK bit for the axis turns on.

The equipment program collects the OK status of each axis. If the OK bit of each axis = on, then My_Equipment.Faulted = off (no faults).
**Alias Tag Guidelines**

Program-scoped tags and phase-scoped tags make your code easier to reuse. Make the tags aliases for tags at the controller scope. If you reuse the equipment phase (for example, copy/paste), simply point the phase-scoped tags to new tags at the controller scope. This practice reduces address fixes within the code.

**Example**

The controller automatically makes a tag for an equipment phase. The tag is at the controller scope (controller tag). Suppose that you plan to reuse an equipment phase for another part of your tank.

1. Make an alias tag for the first equipment phase. Make the tag at the phase scope and point it to the controller tag for that equipment phase.

2. Use the alias tag throughout the code of the equipment phase (This Phase).

3. Make a copy of the equipment phase.

4. Point the alias tag of the copy to its controller tag.

**Additional Resources**

<table>
<thead>
<tr>
<th><strong>For this information</strong></th>
<th><strong>See this publication</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Guidelines and considerations for alias tags</td>
<td>Logix5000 Controllers Design Considerations, publication 1756-RM094</td>
</tr>
<tr>
<td>Steps to assign alias tags</td>
<td>Logix5000 Controllers Common Procedures Programming Manual, publication 1756-PM001</td>
</tr>
</tbody>
</table>
PHASE Data Type

Introduction

The PHASE data type gives you status information about an equipment phase.

When you create an equipment phase, RSLogix 5000® software creates a tag for the status of the equipment phase. Controller scope
Name = phase_name
PHASE data type

Set and Clear Equipment Phase Tag Values

For most of the members of the PHASE data type, you can only monitor its value. You can control only the following members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Control Method</th>
</tr>
</thead>
</table>
| StepIndex   | If you program an equipment phase as a sequence of steps in ladder diagram or structured text, use the StepIndex value as the step number or bit value. (SFCs automatically sequence through steps.)
|             | • To initialize the StepIndex value, use the configuration properties for the equipment phase. |
| Failure     | To Use this instruction                                                        |
|             | Set the Failure value                                                         |
|             | Equipment Phase Failure (PFL)                                                 |
|             | Clear the Failure value                                                       |
|             | Equipment Phase Clear Failure (PCLF)                                          |

When the equipment phase goes from idle → running, StepIndex = Initial Step Index.

• To advance to the next step, write logic to increment the StepIndex value (for example, MOV, MUL, OTL, :=)
## PHASE Data Type

<table>
<thead>
<tr>
<th>Member</th>
<th>Control Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewInputParameters</td>
<td>To clear the NewInputParameters bit, use an Equipment Phase New Parameters (PRNP) instruction.</td>
</tr>
<tr>
<td>Producing</td>
<td>Use bit-level instructions or an assignment to set or clear this bit (for example, OTE, :=).</td>
</tr>
<tr>
<td>Standby</td>
<td>Use bit-level instructions or an assignment to set or clear this bit (for example, OTE, :=).</td>
</tr>
</tbody>
</table>

### PHASE Data Type

<table>
<thead>
<tr>
<th>If you want to</th>
<th>Then check this member</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use one member to monitor the state of an equipment phase</td>
<td>State</td>
<td>DINT</td>
<td>For this state: Use this bit:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Running 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Holding 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Restarting 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Stopping 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Aborting 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Resetting 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Idle 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Held 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Complete 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Stopped 9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Aborted 10</td>
</tr>
<tr>
<td>See if the equipment phase is in the running state</td>
<td>Running</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the holding state</td>
<td>Holding</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the restarting state</td>
<td>Restarting</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the stopping state</td>
<td>Stopping</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the aborting state</td>
<td>Aborting</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the resetting state</td>
<td>Resetting</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the idle state</td>
<td>Idle</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the held state</td>
<td>Held</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the complete state</td>
<td>Complete</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>If you want to</td>
<td>Then check this member</td>
<td>Data type</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>See if the equipment phase is in the stopped state</td>
<td>Stopped</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the aborted state</td>
<td>Aborted</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>Use one member to monitor the substate of an equipment phase</td>
<td>Substate</td>
<td>DINT</td>
<td>Read–only</td>
</tr>
<tr>
<td>For this substate Use this bit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pausing</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paused</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AutoPause</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>See if the equipment phase is in the pausing substate</td>
<td>Pausing</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the paused substate</td>
<td>Paused</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if the equipment phase is in the auto pause substate</td>
<td>AutoPause</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>Use an integer value or the bits of an integer to sequence through a series of steps</td>
<td>SteplIndex</td>
<td>DINT</td>
<td></td>
</tr>
<tr>
<td>• To initialize the SteplIndex value, use the configuration properties for the equipment phase.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• To advance to the next step, use logic such as an MOV, MUL, or := to increment the SteplIndex value.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flag a specific exception for an equipment phase (fault, failure, off-normal condition, and so forth.)</td>
<td>Failure</td>
<td>DINT</td>
<td>To</td>
</tr>
<tr>
<td>To</td>
<td>Use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set a Failure value</td>
<td>PFL instruction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clear the Failure value</td>
<td>PCLF instruction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Find the unit ID of an equipment phase</td>
<td>UnitID</td>
<td>DINT</td>
<td>Read–only</td>
</tr>
<tr>
<td>Monitor the ownership of an equipment phase</td>
<td>Owner</td>
<td>DINT</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an external request is in process via a PXRQ instruction</td>
<td>PendingRequest</td>
<td>DINT</td>
<td></td>
</tr>
<tr>
<td>• Read–only</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Each bit = the state of a specific request, starting with bit 0. The bits are in the order shown by the request-specific members.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>See if a Download Input Parameters request is in process via a PXRQ instruction</td>
<td>DownloadInputParameters</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Input Parameters Subset request is in process via a PXRQ instruction</td>
<td>DownloadInputParameters Subset</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Upload Output Parameters request is in process via a PXRQ instruction</td>
<td>UploadOutputParameters</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Upload Output Parameters Subset request is in process via a PXRQ instruction</td>
<td>UploadOutputParameters Subset</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Output Parameter Limits request is in process via a PXRQ instruction</td>
<td>DownloadOutput ParameterLimits</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Acquire Resources request is in process via a PXRQ instruction</td>
<td>AcquireResources</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
</tbody>
</table>
## Appendix A  PHASE Data Type

<table>
<thead>
<tr>
<th>If you want to</th>
<th>Then check this member</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>See if a Release Resources request is in process via a PXRQ instruction</td>
<td>ReleaseResources</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Send Message To Linked Phase request is in process via a PXRQ instruction</td>
<td>SendMessageToLinkedPhase</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Send Message To Linked Phase And Wait request is in process via a PXRQ instruction</td>
<td>SendMessageToLinkedPhaseAndWait</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Receive Message From Linked Phase request is in process via a PXRQ instruction</td>
<td>ReceiveMessageFromLinkedPhase</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Cancel Message To Linked Phase request is in process via a PXRQ instruction</td>
<td>CancelMessageToLinkedPhase</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Send Message To Operator request is in process via a PXRQ instruction</td>
<td>SendMessageToOperator</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Clear Message To Operator request is in process via a PXRQ instruction</td>
<td>ClearMessageToOperator</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Generate E Signature request is in process via a PXRQ instruction</td>
<td>GenerateESignature</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Batch Data request is in process via a PXRQ instruction</td>
<td>DownloadBatchData</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Material Track Data Container In Use request is in process via a PXRQ instruction</td>
<td>DownloadMaterialTrackDataContainerInUse</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Container Binding Priority request is in process via a PXRQ instruction</td>
<td>DownloadContainerBindingPriority</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Sufficient Material request is in process via a PXRQ instruction</td>
<td>DownloadSufficientMaterial</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if a Download Material Track Database Data request is in process via a PXRQ instruction</td>
<td>DownloadMaterialTrackDatabaseData</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Upload Material Track Data Container In Use request is in process via a PXRQ instruction</td>
<td>UploadMaterialTrackDataContainerInUse</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Upload Container Binding Priority request is in process via a PXRQ instruction</td>
<td>UploadContainerBindingPriority</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if an Upload Material Track Database Data request is in process via a PXRQ instruction</td>
<td>UploadMaterialTrackDatabaseData</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if your logic has aborted a PXRQ instruction</td>
<td>AbortingRequest</td>
<td>BOOL</td>
<td>Read–only</td>
</tr>
<tr>
<td>See if FactoryTalk Batch software has new parameters for an equipment phase</td>
<td>NewInputParameters</td>
<td>BOOL</td>
<td>• Read–only</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• FactoryTalk Batch software sets this bit when it has new parameters for an equipment phase.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• To clear the NewInputParameters bit, use a PRNP instruction.</td>
</tr>
<tr>
<td>If you want to</td>
<td>Then check this member</td>
<td>Data type</td>
<td>Notes</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------</td>
<td>-----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Initiate a producing state</td>
<td>Producing</td>
<td>BOOL</td>
<td>Logix5000 equipment phases don’t have a producing state. To create a producing state, use the Producing bit.</td>
</tr>
<tr>
<td>Initiate a standby state</td>
<td>Standby</td>
<td>BOOL</td>
<td>Logix5000 equipment phases don’t have a standby state. To create a standby state, use the Standby bit.</td>
</tr>
</tbody>
</table>
Notes:
Appendix B

Configure an Equipment Phase

Introduction

This appendix steps you through the configuration settings for an equipment phase.

Use this appendix when you want to change the default settings of an equipment phase.

Open the Configuration for an Equipment Phase

1. 

2. 

![Equipment Phase Properties - My_Phase](image)

- General
  - Configuration
  - Parameters
  - Monitor

  **Assigned Routines:**
  - Program: <none>
  - Fault: <none>

  - Inhibit Equipment Phase

  - Initial State: Idle

  - Complete State Immediately If not Implemented

  - Initial Step Index: 0

  - External Sequences Loss of Communication Command: <none>

  - External Request Hold Action: <none>

  - OK
  - Cancel
  - Apply
  - Help
## Equipment Phase Settings

Use the following settings to configure an equipment phase.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Choices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prestate</strong></td>
<td><strong>Choices</strong></td>
</tr>
<tr>
<td></td>
<td>Prestate routine</td>
</tr>
<tr>
<td></td>
<td>The prestate routine runs constantly, even when the equipment phase is in the idle state. It runs before each scan of a state.</td>
</tr>
<tr>
<td>Do you want to run a prestate routine?</td>
<td>• YES — Select the routine that you want to run.</td>
</tr>
<tr>
<td><strong>Fault</strong></td>
<td>A fault routine lets you clear a major fault made by an instruction.</td>
</tr>
<tr>
<td>Do you want to create a fault routine for the instructions in this equipment phase?</td>
<td>• YES — Select the routine that you want as your fault routine.</td>
</tr>
<tr>
<td><strong>Inhibit Equipment Phase</strong></td>
<td>Do you want the controller to run this equipment phase?</td>
</tr>
<tr>
<td>Do you want the controller to run this equipment phase?</td>
<td>• YES — Leave this box unchecked or uncheck it.</td>
</tr>
<tr>
<td><strong>Initial State</strong></td>
<td>Which state do you want the equipment phase to go to when you turn on the controller?</td>
</tr>
<tr>
<td></td>
<td>• Idle</td>
</tr>
<tr>
<td></td>
<td>• Stopped</td>
</tr>
<tr>
<td><strong>Complete State Immediately If not Implemented</strong></td>
<td>Do you want the equipment phase to skip any states that you aren’t using?</td>
</tr>
<tr>
<td>Do you want the equipment phase to skip any states that you aren’t using?</td>
<td>• YES — Leave this box checked or check it.</td>
</tr>
<tr>
<td><strong>Initial Step Index</strong></td>
<td>A. Are any of the state routines in ladder diagram or structured text?</td>
</tr>
<tr>
<td></td>
<td>• NO — Skip this box.</td>
</tr>
<tr>
<td></td>
<td>B. Do any of those state routines use step numbers?</td>
</tr>
<tr>
<td></td>
<td>• YES — Type the number for the first step of each state.</td>
</tr>
</tbody>
</table>

The tag for the equipment phase has a StepIndex number. The controller resets the StepIndex each time the equipment phase changes states. The controller resets the StepIndex to the number you put in the Initial Step Index box.
### External Sequencer Loss of Communication Command

<table>
<thead>
<tr>
<th>Setting</th>
<th>Choices</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Are you using RSBizWare™ Batch software to command this equipment phase?</td>
<td>• NO — Skip this box.</td>
</tr>
<tr>
<td></td>
<td>• YES — Go to step B.</td>
</tr>
<tr>
<td>B. If the controller loses communication with RSBizWare Batch software, what do you want the equipment phase to do?</td>
<td>• Continue in its current state — Choose None.</td>
</tr>
<tr>
<td></td>
<td>• Go to aborting — Choose Abort.</td>
</tr>
<tr>
<td></td>
<td>• Go to holding — Choose Hold.</td>
</tr>
<tr>
<td></td>
<td>• Go to stopping — Choose Stop.</td>
</tr>
<tr>
<td>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.</td>
<td></td>
</tr>
</tbody>
</table>

### External Request Hold Action

<table>
<thead>
<tr>
<th>A. Are you using any PXREQ instructions?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO — Skip this box.</td>
</tr>
<tr>
<td>• YES — Go to step B.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B. What do you want to do if an equipment phase goes to holding while a PXREQ instruction is in process?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Nothing — Choose none.</td>
</tr>
<tr>
<td>• Stop the request — Choose Clear.</td>
</tr>
</tbody>
</table>

---

**Publication LOGIX-UM001C-EN-P - June 2016**

**Appendix B**
Appendix B Configure an Equipment Phase

Notes:
This manual uses the following terms.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit</td>
<td>A group of equipment that works together to produce the product or interim product. The equipment of a unit operates independent (relatively independent) from other equipment.</td>
<td>• brew kettle&lt;br&gt;• mixing tank&lt;br&gt;• bottle filling machine&lt;br&gt;• bottle capping machine</td>
</tr>
<tr>
<td>Equipment module</td>
<td>A group of input devices, output devices, motors, drives, and soft controls (PID loops, totalizers, and so forth.) that go together to perform a specific activity (task, function) of a unit. The devices within an equipment module:&lt;br&gt;• operate as one entity. &lt;br&gt;• operate independent (relatively independent) from other equipment.</td>
<td>• fill a tank with water&lt;br&gt;• mix the contents of a tank&lt;br&gt;• drain a tank&lt;br&gt;• fill bottles&lt;br&gt;• cap bottles</td>
</tr>
<tr>
<td>Equipment module</td>
<td>Collection of data values that you supply to an equipment module or get from it to monitor and control it. An equipment module interface includes on/off/start/stop commands, mode requests, set points, and fault/health status. It acts as a faceplate for your logic to the equipment module.</td>
<td>• BOOL tag: Go_To_This_State&lt;br&gt;• BOOL tag: In_This_State&lt;br&gt;• BOOL tag: Go_To_This_Mode&lt;br&gt;• BOOL tag: In_This_Mode</td>
</tr>
<tr>
<td>Interface</td>
<td>Collection of data values that you supply to an equipment module or get from it to monitor and control it. An equipment module interface includes on/off/start/stop commands, mode requests, set points, and fault/health status. It acts as a faceplate for your logic to the equipment module.</td>
<td>• BOOL tag: Go_To_This_State&lt;br&gt;• BOOL tag: In_This_State&lt;br&gt;• BOOL tag: Go_To_This_Mode&lt;br&gt;• BOOL tag: In_This_Mode</td>
</tr>
<tr>
<td>Unit procedure</td>
<td>The sequence of processing activities that a unit performs to produce the product or interim product. A unit procedure directs the execution of phases. A unit procedure could be a hierarchy of SFCs that is subdivided into specific operations. Each operation directs the execution of a group of phases. A unit may have multiple unit procedures depending on how the sequence changes for different products.</td>
<td>• Charge&lt;br&gt;• Boil&lt;br&gt;• Drain</td>
</tr>
<tr>
<td>Phase</td>
<td>A specific task that your equipment does. A phase directs the actions of your equipment. It tells the equipment what to do and when to do it.</td>
<td>• Fill bottles with product.&lt;br&gt;• Put bottles in carton.&lt;br&gt;• Fill tank with water&lt;br&gt;• Mix ingredients in tank</td>
</tr>
<tr>
<td>State</td>
<td>The condition of your equipment in relation to normal production. A phase can have up to 11 different states, some of which are active and other are waiting. Active (…ing) states represent the things your equipment does at a given time (running, holding, restarting, stopping, aborting, resetting). Each state contains a separate blocks of code (routine) and can call other routines. Waiting states represent the condition of your equipment when it is in-between active states (stopped, complete, idle, held, aborted, stopped). Waiting states have no associated routines or logic. A phase transitions from one state to another only in a specific order. Your equipment can go from its current state to only certain other states.</td>
<td>• In the running state, fill the tank with water.&lt;br&gt;• In the holding state, temporarily stop filling the tank with water.&lt;br&gt;• In the resetting state, reset the logic and clear the totals.</td>
</tr>
</tbody>
</table>
Index

A
aborted state
use 13
aborting state
use 13, 34
add
equipment phase 18
phase state routine 18

C
clear
PHASE tag values 57
command
eexample 45, 46
give 14
give with PCMD instruction 41
give with RSLogix 5000 software 19
complete state
use 13
configure
equipment phase 63
create
equipment phase 18
phase state routine 18

E
equipment module
See equipment program
equipment phase
add prestate routine 35, 36
compared to PackML 16
compared to S88 16
configure 63
create 18
create a phase state routine 18
data type 58
define your states 26
faults 34
handle faults 34
inhibit 64
initial state 22
instructions 9
lay out 24
lay out the code 31
monitor 15, 19
number 24
override command 44
overview 9
phase state routine 18
set a failure code 38
set initial step index 64
set or clear tag values 57
set the initial state 22
set the prestate routine 37
set up 24
start 45, 46
states 12
test states 19
use 24
equipment phase instructions
overview 9
equipment program
interface tag 51
lay out the code 31
set up the data 51
use 31
example
equipment phases for a machine 25
equipment phases for a tank 25
handle a jam 47
handle fault of a device 39
handle timeout 40
interface tag for a machine 54
interface tags for a tank 53
machine is done resetting 50
procedure for a tank 45
separate code for a machine 33
separate code for a tank 32
sequence equipment phases 45
start a machine 46
state model for a machine 30
state model for a tank 29
tank is done adding water 50
exception
handle 34
external request
hold action 65
respond to lost communication 65

F
fault
eexample 39, 40, 47
handle 34
set a failure code 38

H
held state
use 13
holding state
use 13, 34

I
idle state
use 13
inhibit
equipment phase 64
initial state
choose 26
set 22
initial step index
set 64

M
monitor
equipment phase 19
override command
example 47

ownership
overview 15
take with RSLogix 5000 software 15

PFL instruction
use 38

phase
See equipment phase

PHASE data type
members 58
set or clear values 57
use a state bit 37

phase state routine
add 18

prestate routine
add 35, 36
assign 37
example 39, 40, 46, 47
overview 34
use 34

producing state
set up 26

program
equipment phase 31

PXRQ instruction
hold action 65
lost communication 65

resetting state
use 13

restarting state
use 13, 34

routine
add phase state routine 18

RSLogix 5000 software
give command 19
monitor an equipment phase 15, 19
ownership 15

running state
use 13

sequencer
example 45

set
hold action for a PXRQ instruction 65
initial step index 64
PHASE tag values 57

set up
equipment interface tag 51
equipment phase 24
states 26
transitions 41

standby state
set up 26

state model
See states

state routine
See phase state routine

states
compared to PackML 16
compared to S88 16
handle exceptions 34
mark as done 48
overview 12
set the initial state 22
set up transitions 41
step through 15, 19
transition when done 48
transitions between states 14, 41
use 13, 26
use a state bit 37

stopped state
use 13

stopping state
use 13

test
equipment phase 19

transition
step through 19
when done 48
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 is 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/.