Allen-Bradley

PhaseManager™

1756 ControlLogix®,
1769 CompactLogix™,
1789 SoftLogix™,
1794 FlexLogix™,
20D PowerFlex® 700S with DriveLogix™

User Manual

Rockwell Automation

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.ab.com/manuals/gi) 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.

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

WARNING

Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.

IMPORTANT

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

ATTENTION

Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you:

- identify a hazard
- avoid a hazard
- recognize the consequence

SHOCK HAZARD

Labels may be located on or inside the equipment (e.g., drive or motor) to alert people that dangerous voltage may be present.

BURN HAZARD

Labels may be located on or inside the equipment (e.g., drive or motor) to alert people that surfaces may be dangerous temperatures.

When to Use This Manual

The manual is one of various Logix5000 manuals.

To:	See:
get started with a Logix5000 controller	*Logix5000 Controllers Quick Start*, publication 1756-QS001
program a Logix5000 controller—detailed and comprehensive information	*Logix5000 Controllers Common Procedures*, publication 1756-PM001
You are here ⇨ • use equipment phases • set up a state model for your equipment • program in a way that is similar to S88 and PackML models	*PhaseManager User Manual*, publication LOGIX-UM001
program a specific Logix5000 programming instruction	• *Logix5000 Controllers General Instructions Reference Manual*, publication 1756-RM003 • *Logix5000 Controllers Process and Drives Instructions Reference Manual*, publication 1756-RM006 • *Logix5000 Controllers Motion Instructions Reference Manual*, publication 1756-RM007
import or export a Logix5000 project or tags from or to a text file	*Logix5000 Controllers Import/Export Reference Manual*, publication 1756-RM084
convert a PLC-5 or SLC 500 application to a Logix5000 project	*Logix5550 Controller Converting PLC-5 or SLC 500 Logic to Logix5550 Logic Reference Manual*, publication 1756-6.8.5
use a specific Logix5000 controller	• *CompactLogix Controllers User Manual*, publication1769-UM007 • *ControlLogix System User Manual*, publication 1756-UM001 • *DriveLogix System 5720 User Manual*, publication 20D-UM002 • *DriveLogix5730 Controller for PowerFlex 700S Drives with Phase II Control User Manual*, publication 20D-UM003 • *FlexLogix Controllers User Manual*, publication1794-UM001 • *SoftLogix5800 System User Manual*, publication 1789-UM002
control devices over an EtherNet/IP network	*EtherNet/IP Modules in Logix5000 Control Systems User Manual*, publication ENET-UM001
control devices over an ControlNet™ network	*ControlNet Modules in Logix5000 Control Systems User Manual*, publication CNET-UM001
control devices over an DeviceNet™ network	*DeviceNet Modules in Logix5000 Control Systems User Manual*, publication DNET-UM004

Purpose of This Manual

This manual shows you how to set up 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
- set up a state model for your equipment
- program your equipment to run by the state model
- use equipment phase instructions to transition to a different state, handle faults, set up break points, etc.

A Logix5000 controller is any of the following:

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

Who Should Use this Manual

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

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

- programmable controllers
- industrial automation systems
- personal computers

How to Use this Manual

As you use this manual, you will see some terms that are formatted differently from the rest of the text:

Text that is:	Identifies:	For example:	Means:
Italic	the actual name of an item that you see on your screen or in an example	Right-click *User-Defined* …	Right-click on the item that is named User-Defined.
`courier`	information that you must supply based on your application (a variable)	Right-click `name_of_program` …	You must identify the specific program in your application. Typically, it is a name or variable that you have defined.
enclosed in brackets	a keyboard key	Press [Enter].	Press the Enter key.

Appendix A

Equipment Phase Instructions (PSC, PCMD, POVR, PFL, PCLF, PX RQ, PRNP, PPD, PATT, PDET)

Appendix B

PHASE Data Type

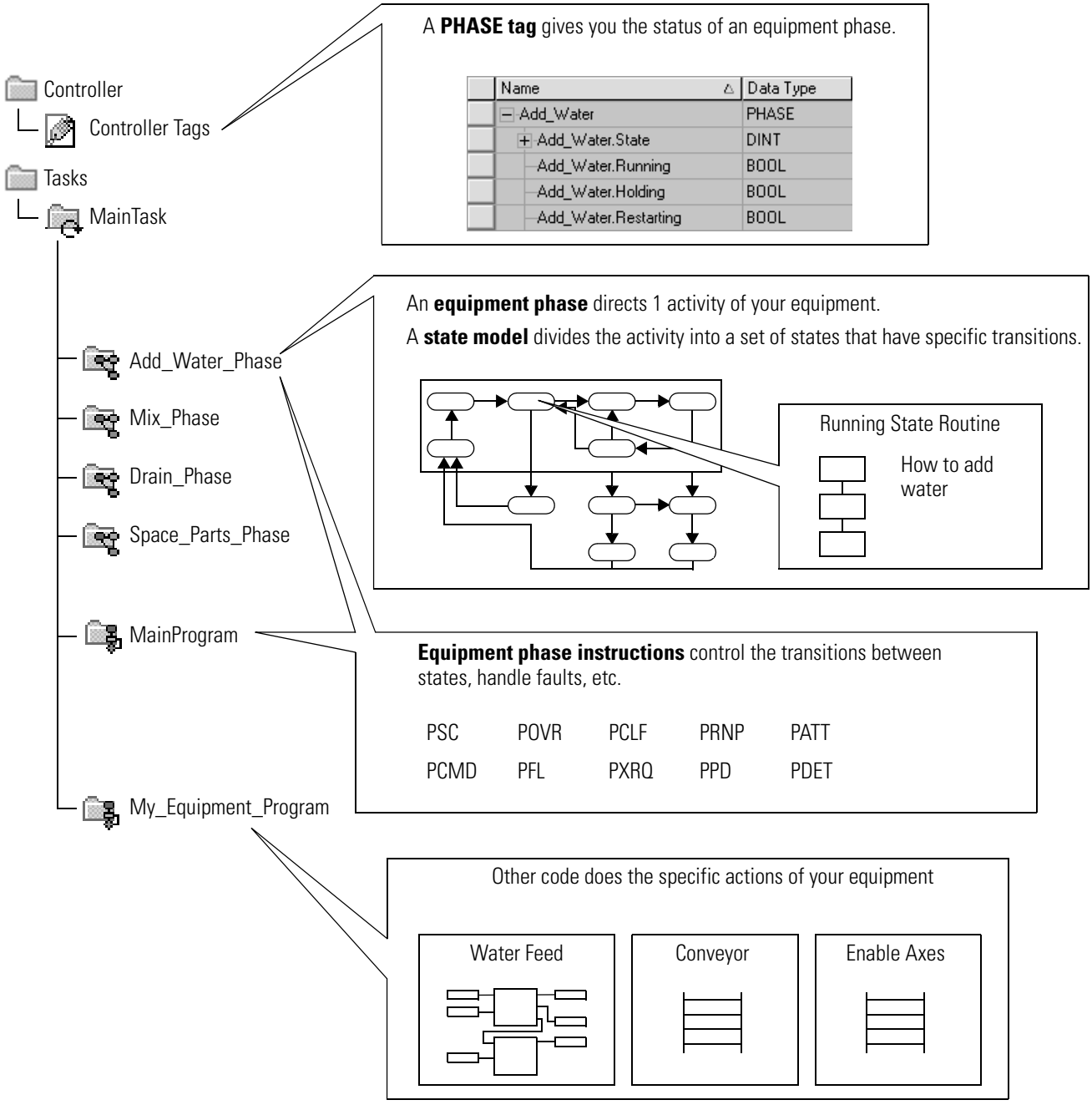
Appendix C

Configure an Equipment Phase

Introduction

What is PhaseManager?

PhaseManager™ lets you add equipment phases to your controller. An equipment phase makes it easier to write, use, and manage the code for your machine or equipment.



Here's some PhaseManager terms:

Term	Description
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 is different from a program in these ways: • The equipment phase uses a state model. • Use an equipment phase to do 1 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 • Pack*ML*, 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 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 simplifies the amount of interlocking that you have to do.
equipment phase instructions	Specific instructions that you use to control an equipment phase. See Appendix A.
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 RSBizWare Batch software • see if RSBizWare Batch software has new parameters for the equipment phase • set up producing and standby states See Appendix B for more information about the PHASE data type.

How does PhaseManager help me?

PhaseManager helps you write the code for your equipment in a structured way. This results in the same behavior for all the equipment across a plant.

Specifically, PhaseManager helps you with questions such as:

Question	Answer
How can I get the highest performance possible from my equipment?	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'll be able to calculate the efficiency and performance measures that you want. If you use PhaseManager across your plant, you have consistent data from equipment to equipment.
How can I cut the cost of integrating my equipment into the plant?	Clear structure and consistent tags make it a lot easier to plug the equipment into your plant and set up communication right away. Equipment up and down that line share data using the same tag names. And all equipment communicates with higher-level systems in the same way.
How can I make it easier to maintain the code?	A state model helps you lay out the general functions of your equipment. We found that the best programmers use a state model as the heart of their code. A state model serves as a map for the code. With a clear structure, you'll know just where to look for the piece of code that you want.
How can I give my operators a clean, intuitive HMI?	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 idle, running, or holding, your operators will know exactly what that means.

What is a state model?

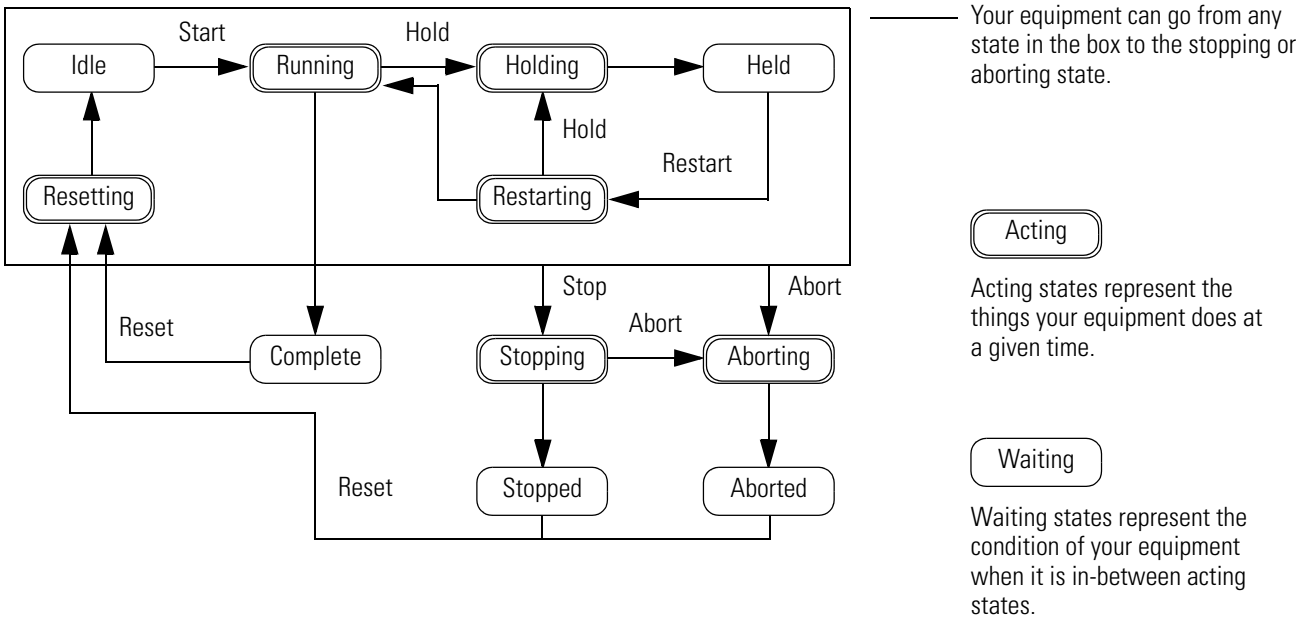
A state model divides the operating cycle of your equipment into a series of states. Each state is an instant in the operation of the equipment. It's the actions or conditions of the equipment at a given time.

In a state model, you define what your equipment does under different conditions, such as run, hold, stop, etc. You don't need to use all the states for your equipment. Use only the states that you want.

There are 2 types of states:

Type of state	Description
Acting	Does something or several things for a certain time or until certain conditions are met. An acting state runs one time or repeatedly.
Waiting	Shows that certain conditions are met and the equipment is waiting for the signal to go to the next state.

PhaseManager uses the following states:



How do I apply a state model to my equipment?

The use of a state model may sound like a big change for programmers. But it's simply a different way to look at 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.

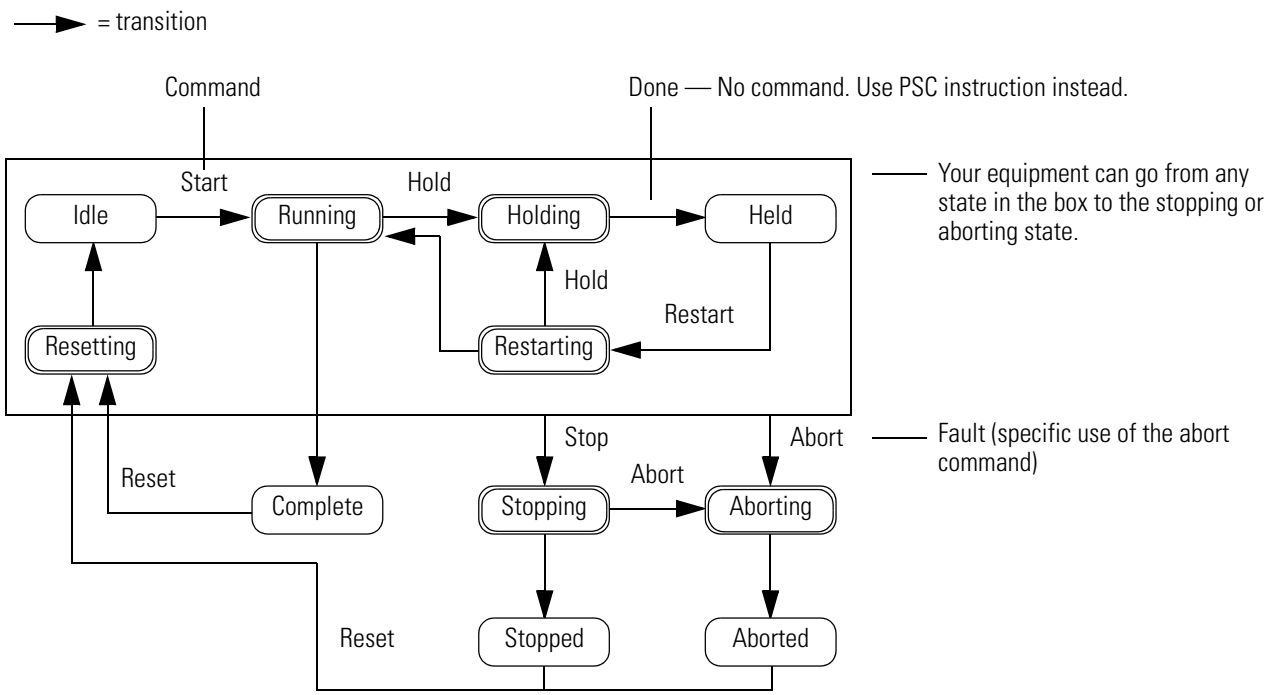
For this State:	Ask:
Stopped	What happens when you turn on power?
Resetting	How does the equipment get ready to run?
Idle	How do you tell that the equipment is ready to run?
Running	What does the equipment do to make product?
Holding	How does the equipment ***temporarily*** stop making product without making scrap?
Held	How do you tell if the equipment is safely holding?
Restarting	How does the equipment resume production after holding?
Complete	How do you tell when the equipment is done with what it had to do?
Stopping	What happens during an normal shutdown?
Aborting	How does the equipment shutdown if a fault or failure happens?
Aborted	How do you tell if the equipment is safely shutdown?

How does my equipment change states?

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 gives the equipment the same behavior as any other equipment that uses the same model.

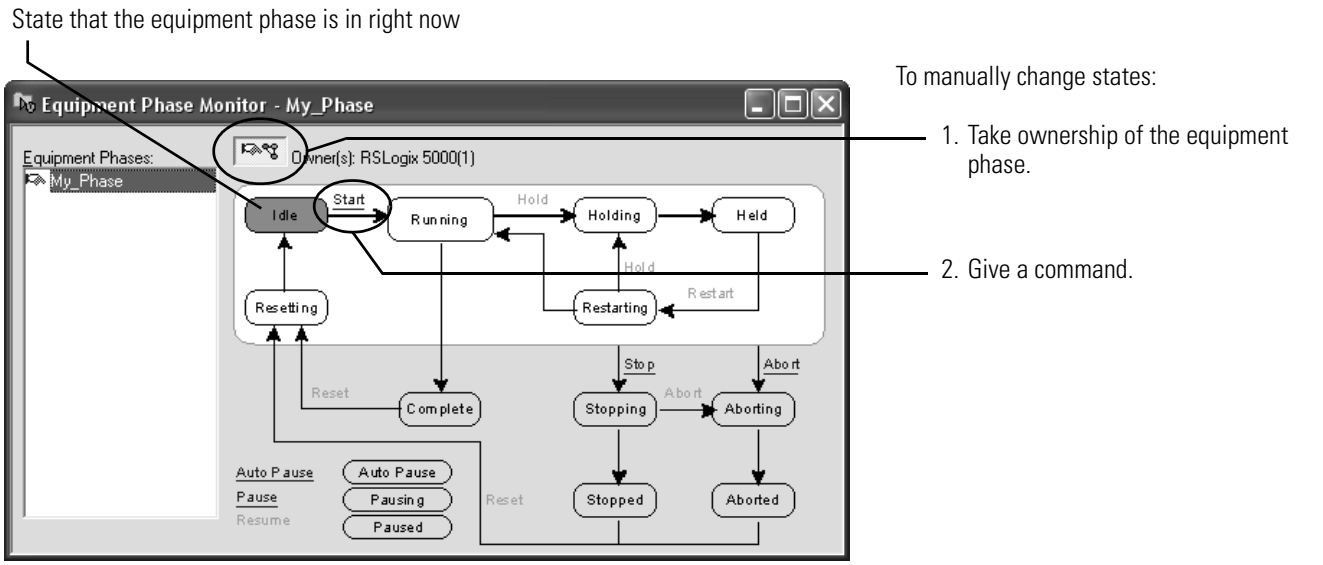
PhaseManager uses the following transitions:



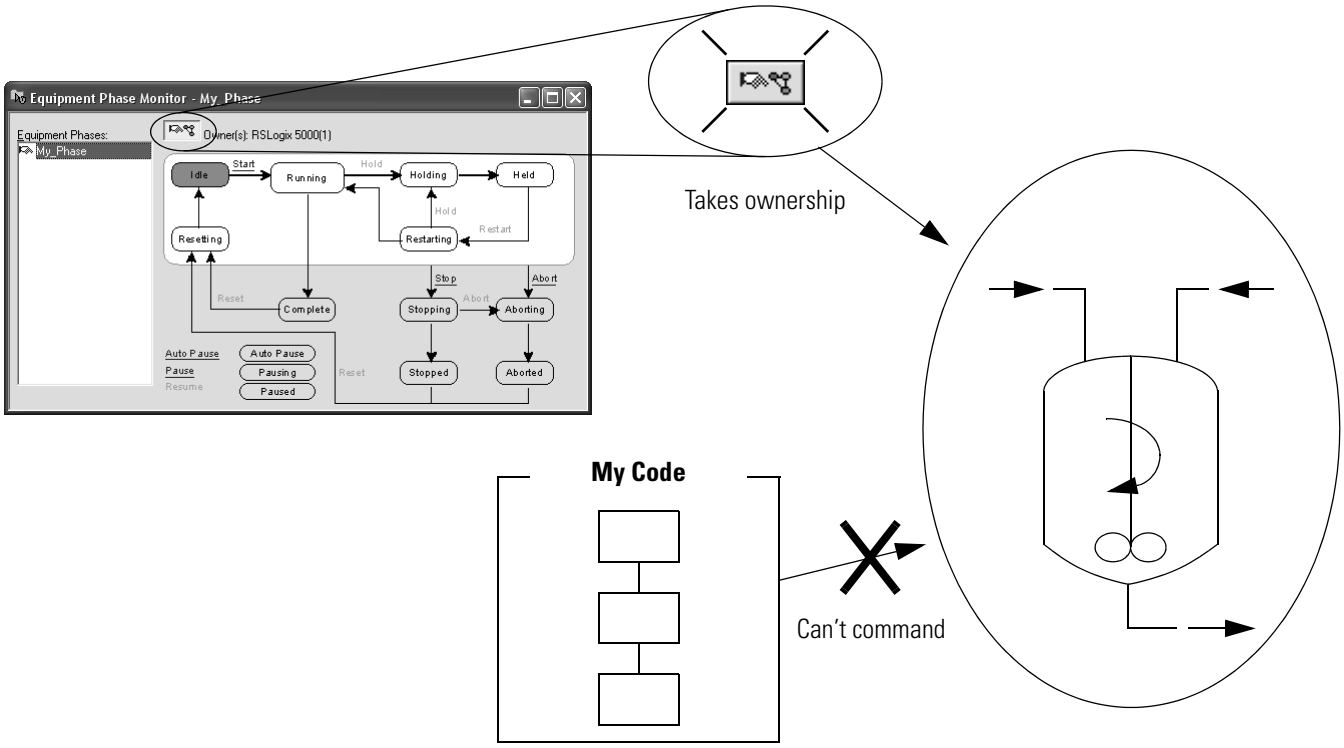
Type of transition	Description
Command	A command tells the equipment to start doing something or do something different. For example the operator pushes the start button to start production and the stop button to shutdown. PhaseManager uses these commands: reset, stop, restart start, hold, abort
Done	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 set up 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 set up your code to look for faults and take action if it finds any. Suppose you want your equipment to shut down as fast as possible if a certain fault happens. In that case, set up your code look for that fault and give the abort command if it finds it.

Can I manually change states?

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



What is ownership?



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

If this owns the equipment phase	Then
RSLogix 5000 software	Sequencers can't give commands to the equipment phase. This includes: • Internal sequencer — program in the controller • External sequencer — RSBizWare Batch software
Internal sequencer — program in the controller	Other sequencers can't give commands to the equipment phase.
External sequencer — RSBizWare Batch software	Other sequencers can't give commands to the equipment phase.

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

See these instructions:

- Equipment Phase Command (PCMD) instruction on page A-8.
- Equipment Phase Override Command (POVR) instruction on page A-13.
- Attach to Equipment Phase (PATT) instruction on page A-42.

What if my equipment doesn't fit the state model?

One common objection to a state model is that it doesn't fit all equipment. You may hear or think: "My equipment is very complex. There's a lot of synchronization and many things happen in parallel."

Keep in mind that a state model looks at your equipment at a very 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.

How does PhaseManager compare to other state models?

The following table compares PhaseManager's state model to other common state models:

S88	Pack*ML*	PhaseManager
Idle	Starting ⇒Ready	Resetting ⇒Idle
Running ⇒Complete	Producing	Running ⇒Complete
Pausing ⇒Paused	Standby	subroutines, breakpoints, or both.
Holding ⇒Held	Holding ⇒Held	Holding ⇒Held
Restarting	none	Restarting
Stopping ⇒Stopped	Stopping ⇒Stopped	Stopping ⇒Stopped
Aborting ⇒Aborted	Aborting ⇒Aborted	Aborting ⇒Aborted

How do I get started?

To get started with PhaseManager:

1. Install RSLogix 5000 software 15.0 or later.
2. See the rest of this manual.

For this information	See
try-out PhaseManager	chapter 2
guidelines for programming a PhaseManager project	chapter 3
detailed reference of equipment phase instructions	appendix A
description of the members of the PHASE data type	appendix B

Notes:

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 a different state

When to use this chapter

Use this quick start when you want to:

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

How to use this chapter

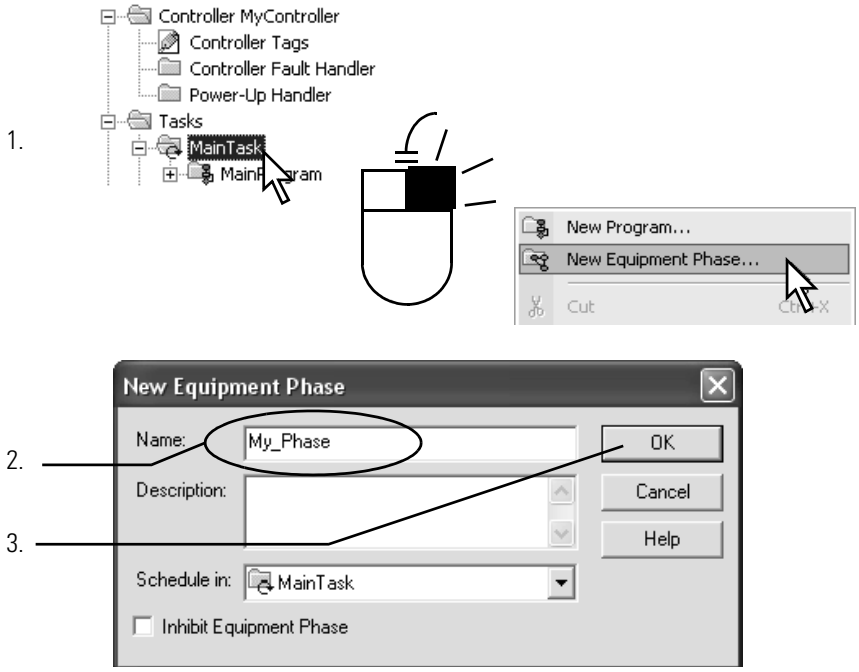
To use this quick start:

Equipment

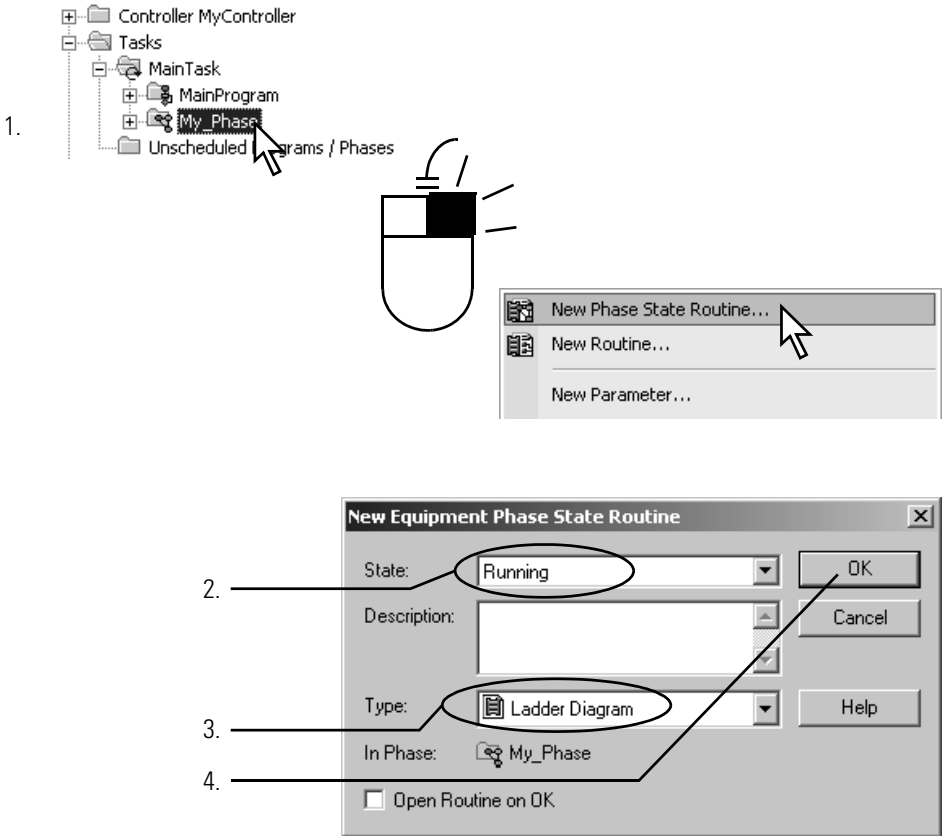
To use this quick start, you need:

- Logix5000 controller. See the preface if you aren't sure which controllers are Logix5000 controllers.
- Firmware 15.0 or later for the controller
- Power supply for the controller
- Communication path to the controller
  - Communication card or built-in port
  - Corresponding communication cable
- RSLogix 5000 software 15.0 or later

Create an Equipment Phase



Create a State Routine



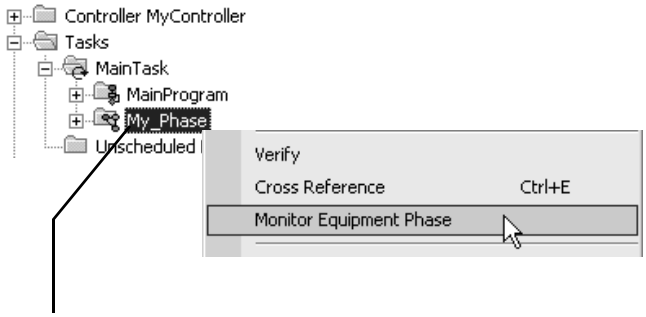
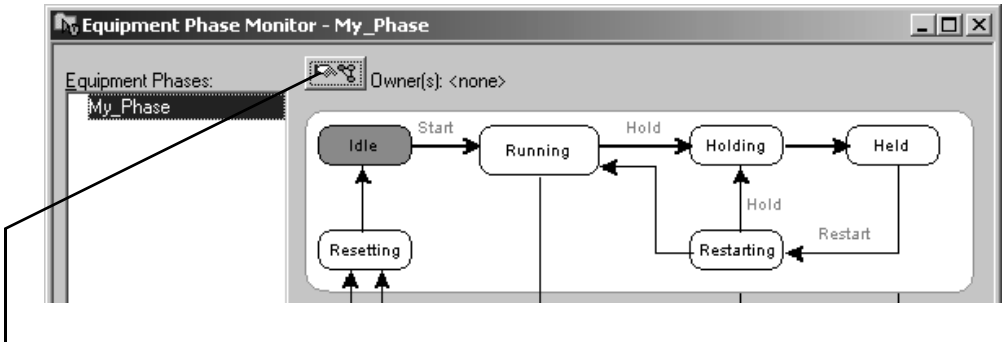
Manually Step Through the States

Before you begin

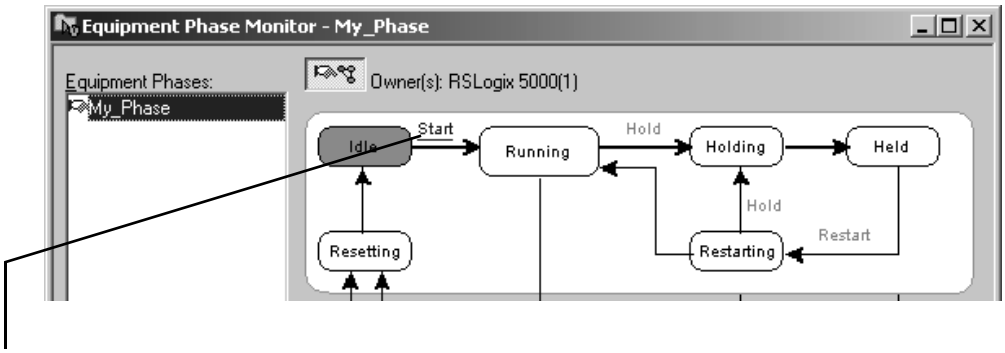
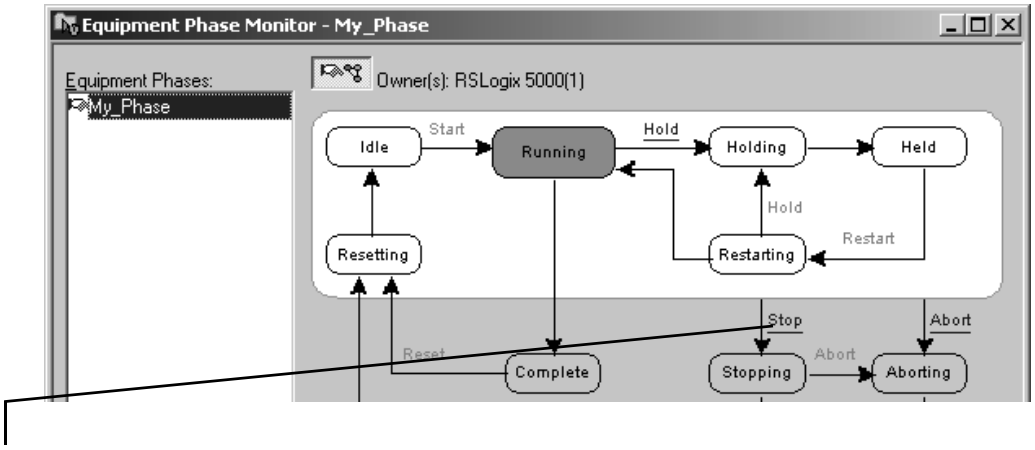
Before you do this procedure, do the following:

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

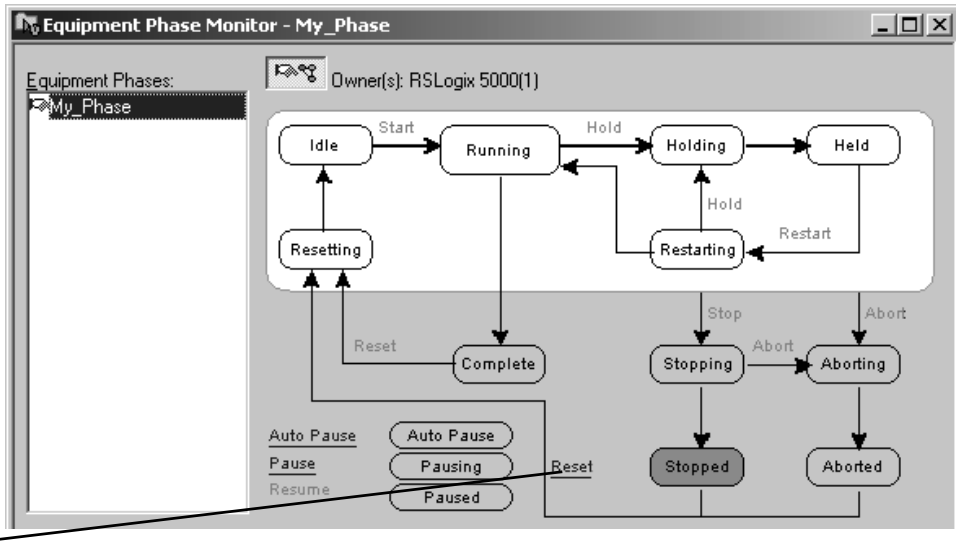
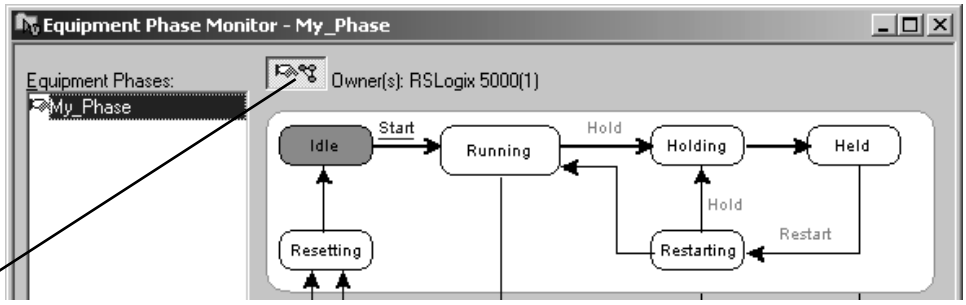
Actions

Step	Notes
 1. Right-click the equipment phase and choose *Monitor Equipment Phase*.	
 2. Click the ownership button and then *Yes—take ownership.*	This lets you use this window to step through the states.

Continued on next page

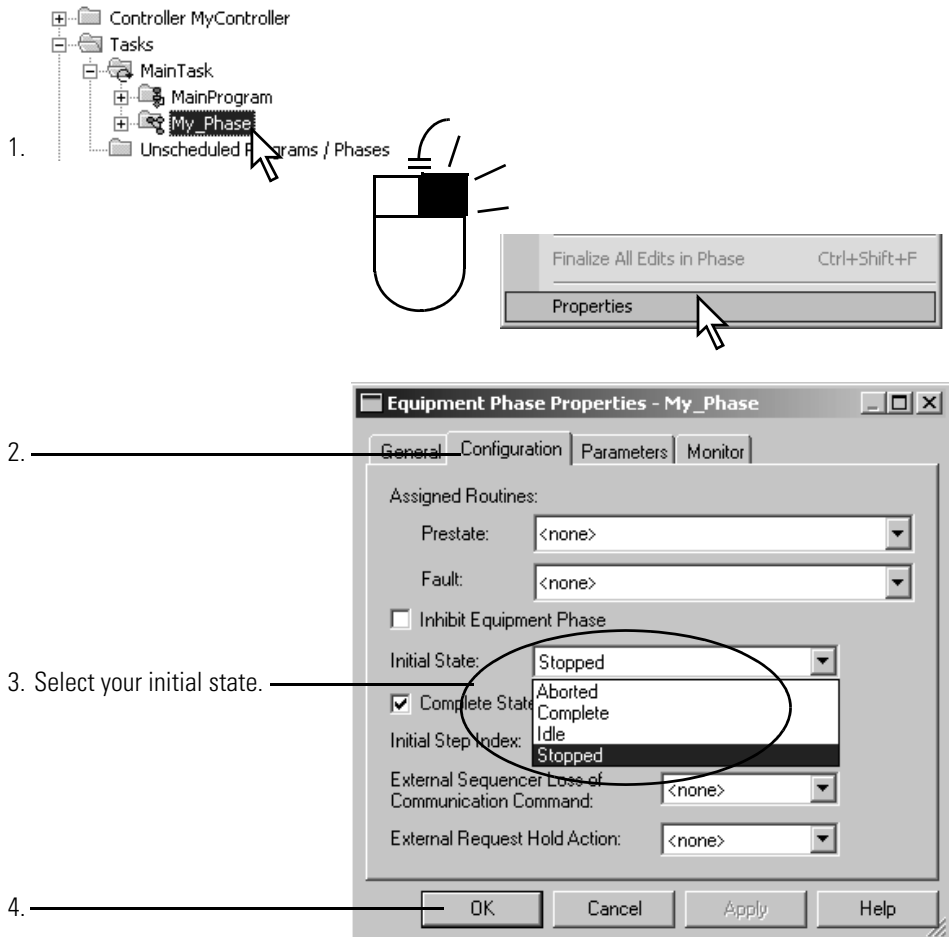
Step	Notes
 3. Click *Start.*	• The equipment phase goes to the Running state. • Any code in the Running state routine starts running. This is where you put the code for the normal production sequence of your equipment.
 4. Click *Stop.*	• The equipment phase goes to the Stopped state. • The Running state routine stops running. • The Stopping state routine is optional. Without it, the equipment phase goes directly to the Stopped state.

Continued on next page

Step	Notes
 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 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.



Guidelines

Purpose of this chapter

To guide your development and programming of a Logix5000 project that uses equipment phases

When to use this chapter

Use this chapter:

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

How to use this chapter

Review the following guidelines before you lay-out your project and refer back to them as needed:

Use a separate equipment phase for each activity

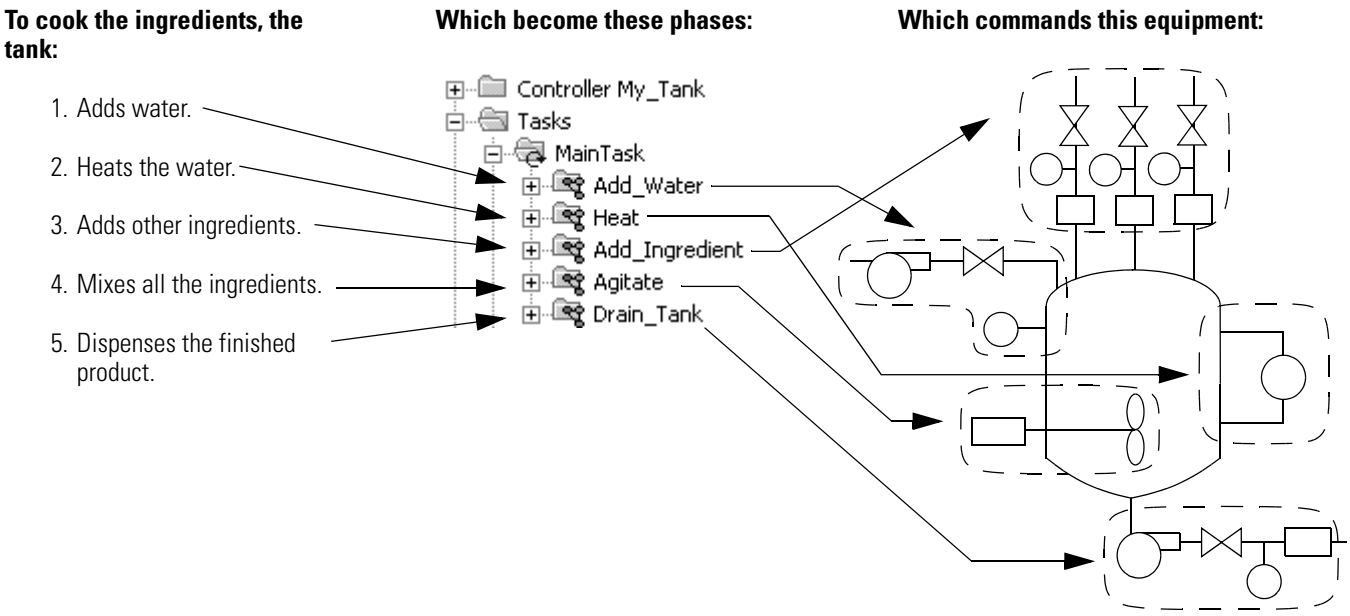
Each equipment phase is a specific activity that your equipment does. An equipment phase tells the equipment what to do and when to do it.

Follow these guidelines as you decide how many equipment phases to use:

Guideline	Details	
1. Make sure each equipment phase does an independent activity.	Make sure each equipment phase does an activity that is independent (relatively independent) from other equipment. The equipment phase commands all the equipment that works together to do the specific activity.	
	Example	
	This is probably an equipment phase	**This in probably NOT an equipment phase**
	• Fill bottles with product. • Put bottles in carton. • Add water to a tank. • Mix ingredients in tank	• Accelerate filler axis (too small) • Run bottling line (too big) • Open water valve (too small) • Brew ingredients (too big)
2. Keep the number of equipment phases and programs within the following limits.	**If you have this controller:**	**You can have up to:**
	ControlLogix	100 programs and equipment phases per task
	SoftLogix	100 programs and equipment phases per task
	FlexLogix	32 programs and equipment phases per task
	CompactLogix	32 programs and equipment phases per task
3. List the equipment that goes along with each equipment phase.	**Example**	
	This equipment phase	**Relates this equipment**
	Add_Water	water pump water valve limit switch
	Smart_Belt	Coarse belt axis Fine belt axis Exit belt axis

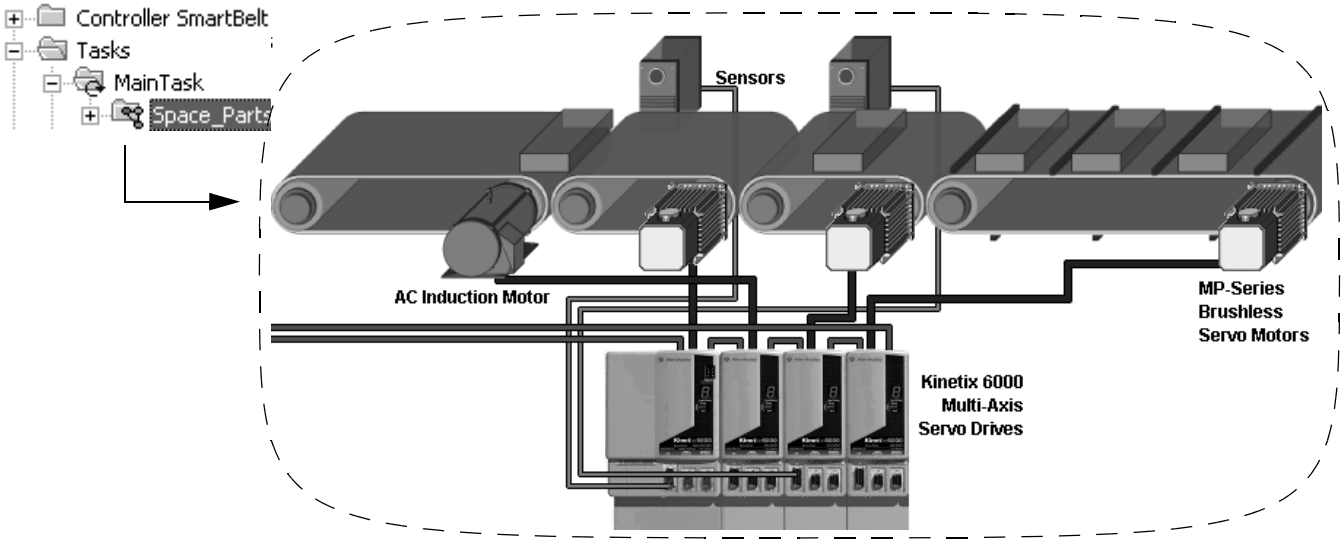
Example 1: Tank

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



Example 2: Smart belt

The following example shows a smart belt. The smart belt does only one activity. It spaces product evenly on an exit belt. Since it does only one activity, it needs only 1 equipment phase.

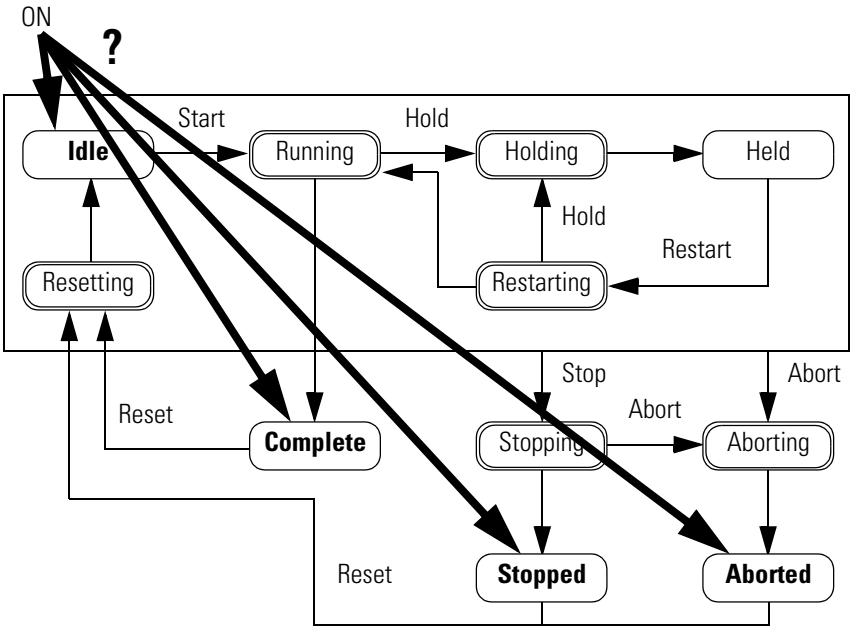


Fill out the state model for each equipment phase

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:

Guideline	Details
1. Fill out 1 state model for each phase.	Each phase runs its own set of states. Fill out 1 state model worksheet for each phase.
2. Decide which state you want as your initial state after power-up.	Which state do you want the equipment phase to go to when you turn on power?



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:

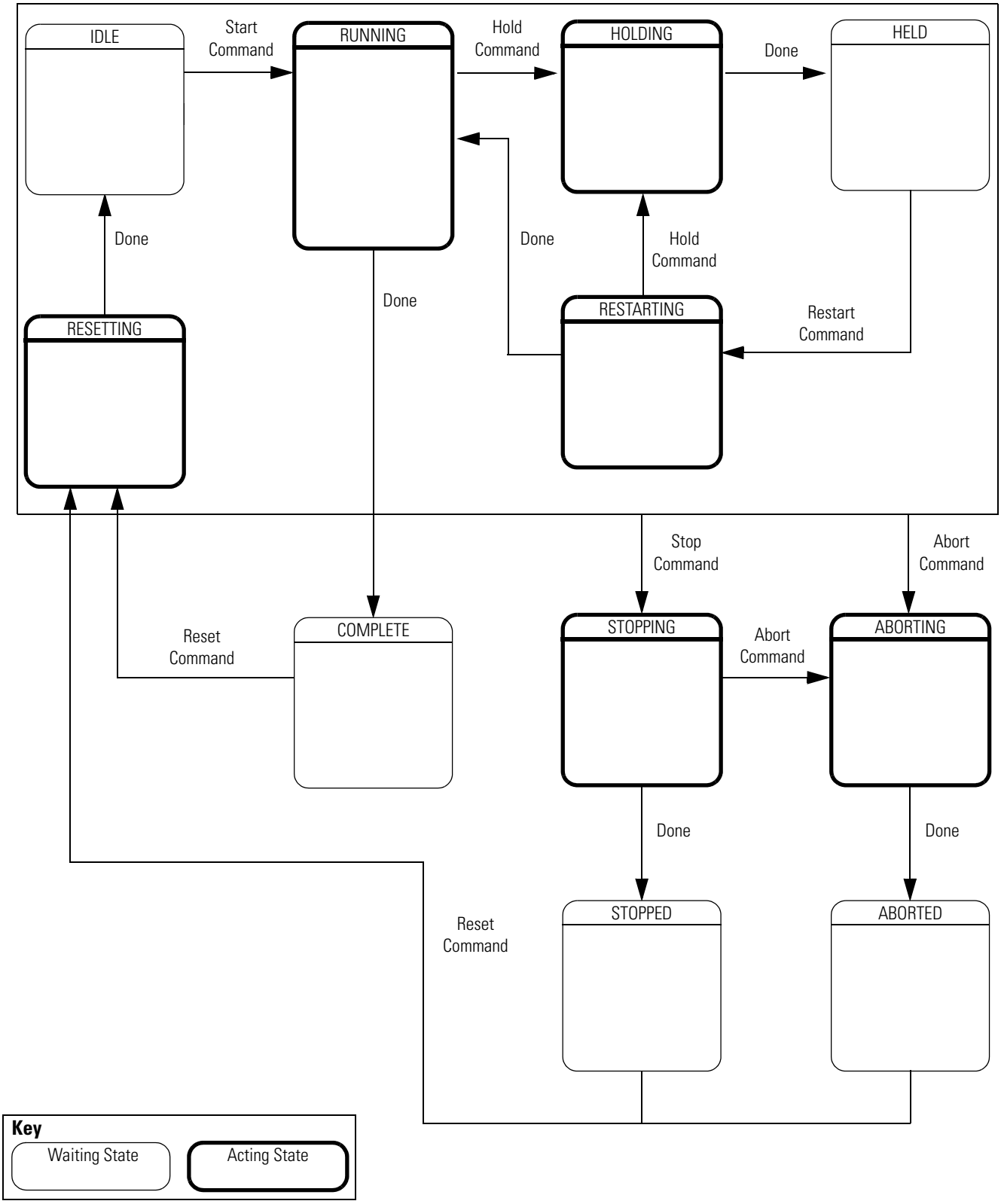
- idle (default)
- complete
- stopped

Choose the initial state that best shows what your equipment is waiting to do after power-up (reset, run, etc.).

Guideline	Details	
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:	
	For this State:	**Ask:**
	Stopped	What happens when you turn on power?
	Resetting	How does the equipment get ready to run?
	Idle	How do you tell that the equipment is ready to run?
	Running	What does the equipment do to make product?
	Holding	How does the equipment pause without making scrap?
	Held	How do you tell if the equipment is safely paused?
	Restarting	How does the equipment resume production after a pause?
	Complete	How do you tell when the equipment is done with what it had to do?
	Stopping	What happens during an normal shutdown?
	Aborting	How does the equipment shutdown if a fault or failure happens?
	Aborted	How do you tell if the equipment is safely shutdown?
4. Use only the states that you want.	Define only the states that are appropriate for your equipment. You *don't* need to use all the states. The equipment phase just skips any states that you don't add.	
5. 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 routines for the producing state and another routine for the standby state. B. In the running state, check for the produce verses standby conditions. Set either the Producing bit or the Standby bit of the equipment phase tag. C. Use the Producing and Standby bits as conditions to call the corresponding routine. See Appendix B.	

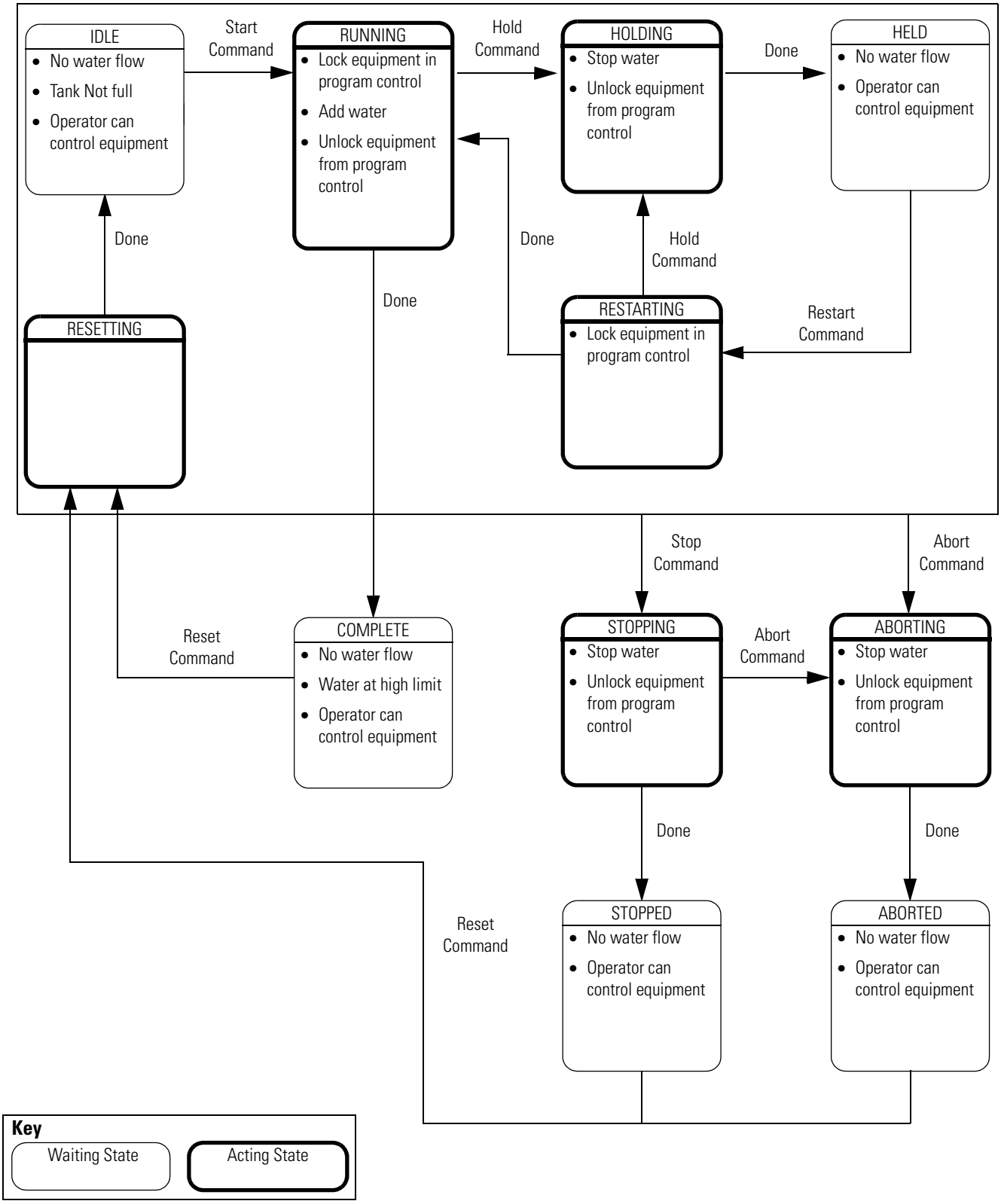
State Model Worksheet

Equipment Phase:



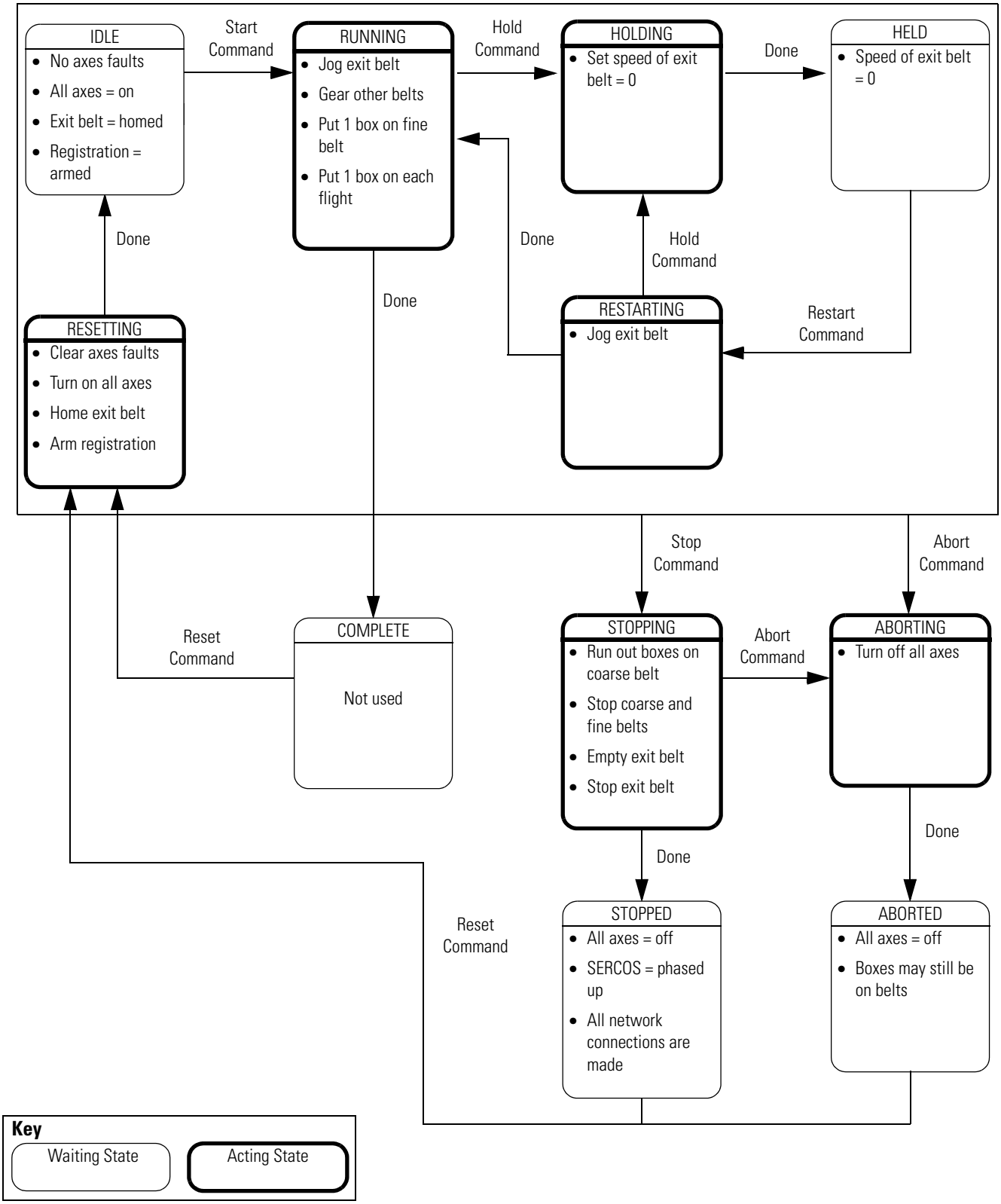
State Model Worksheet

Equipment Phase: *Add Water*



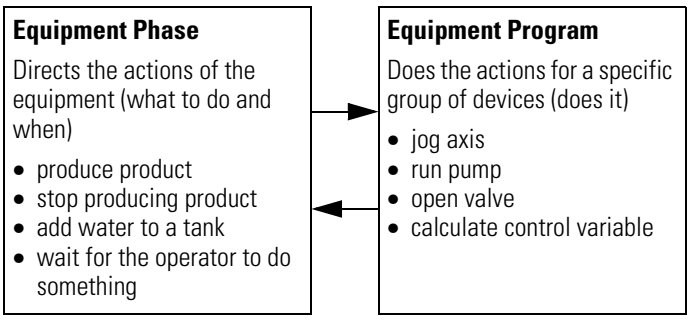
State Model Worksheet

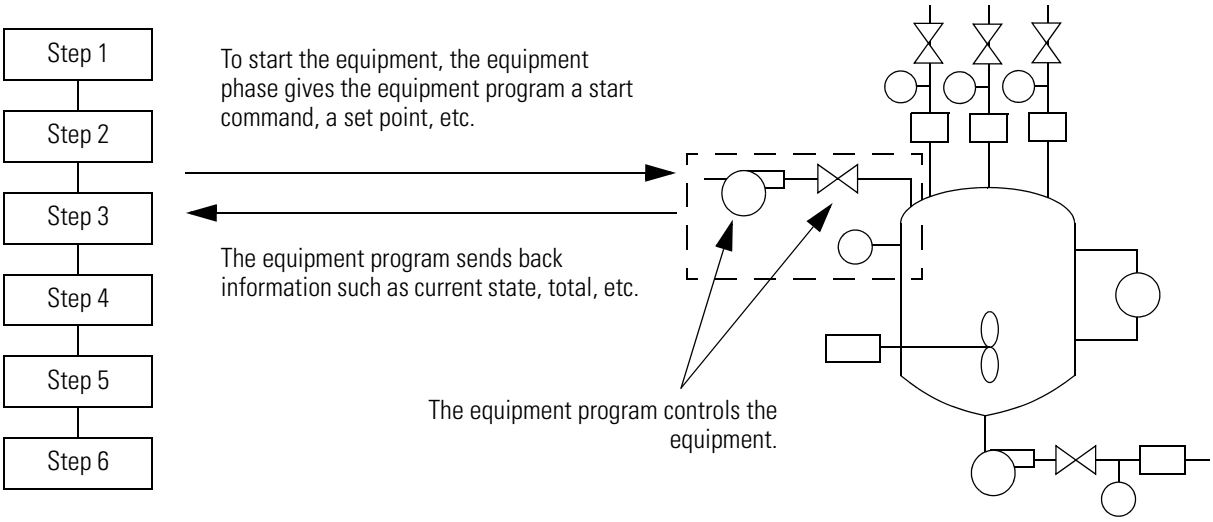
Equipment Phase: *Space Parts*



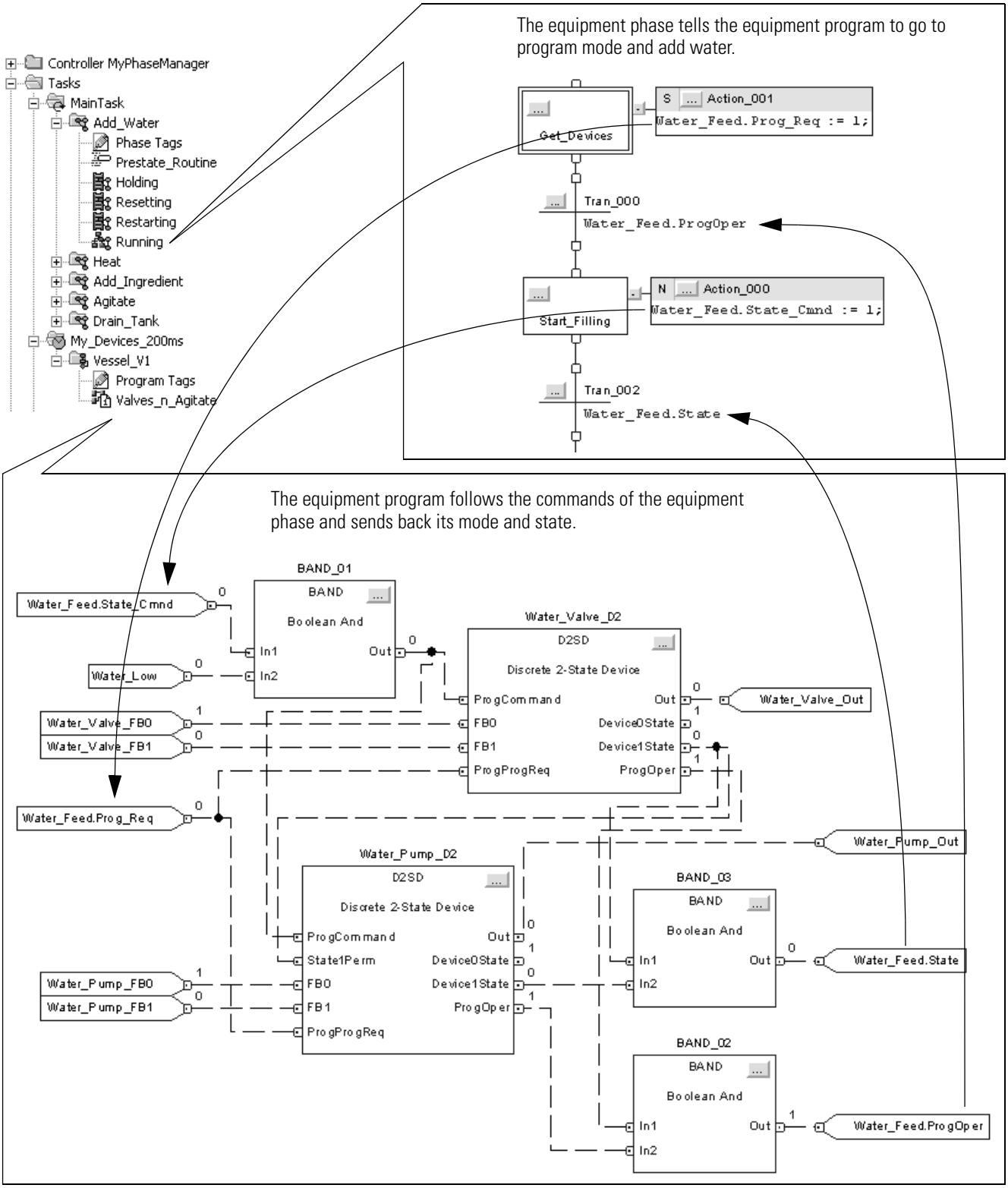
Separate equipment phase code from equipment code

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

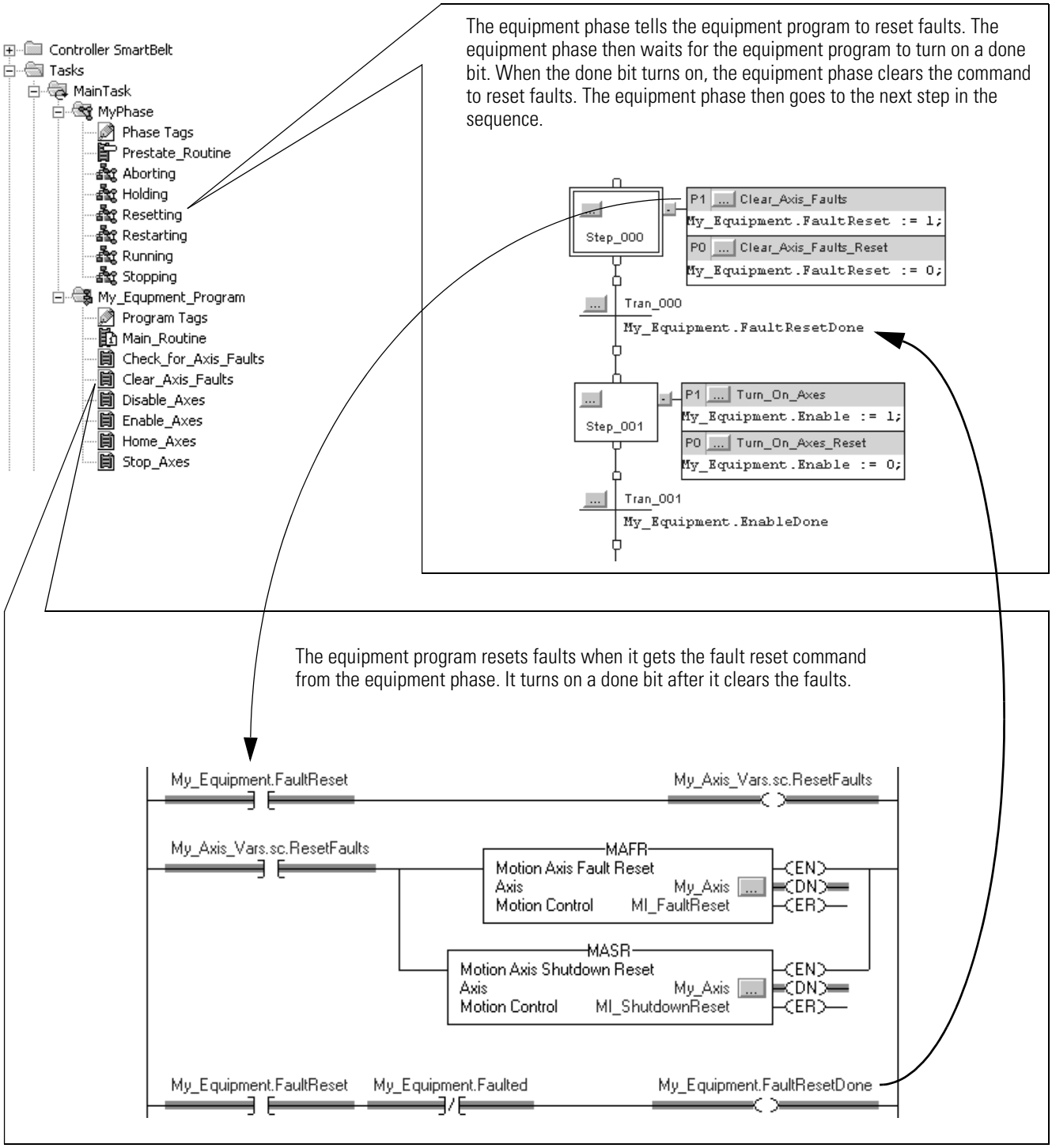




Example 1: Add water to a tank

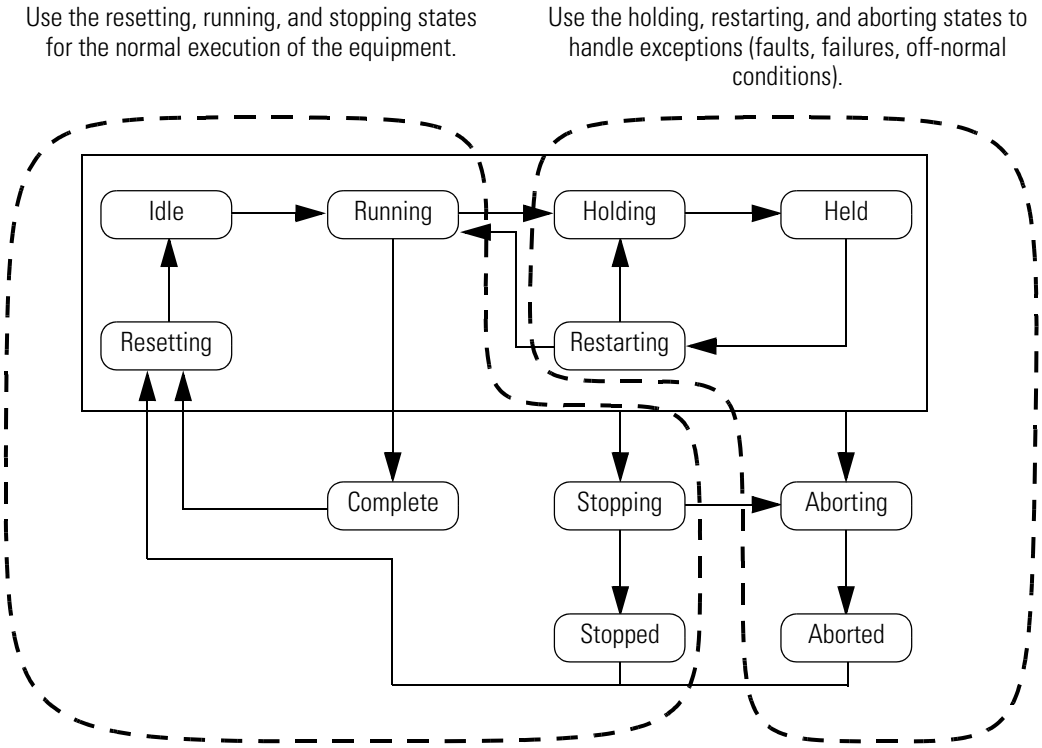


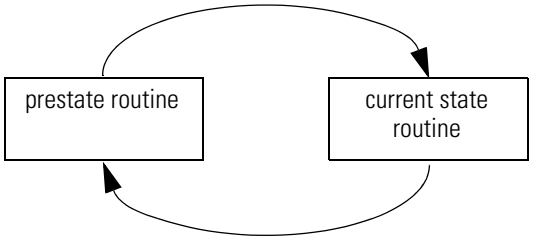
Example 2: Smart belt

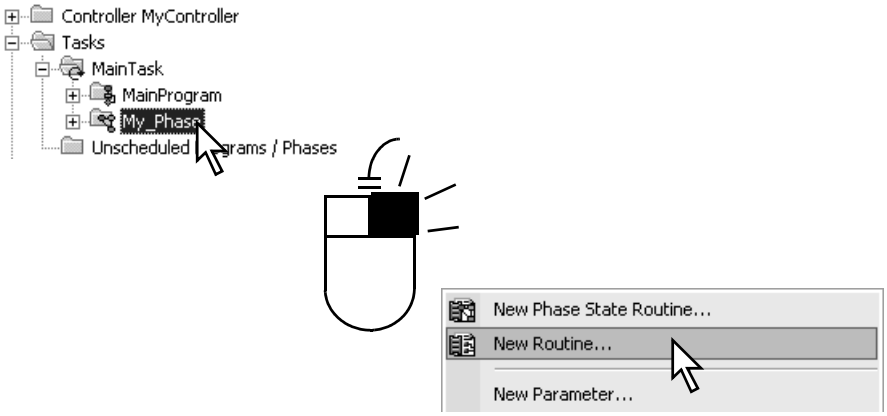
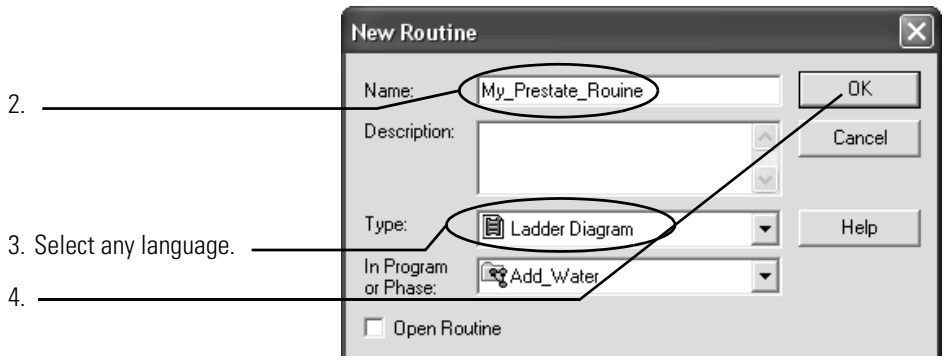


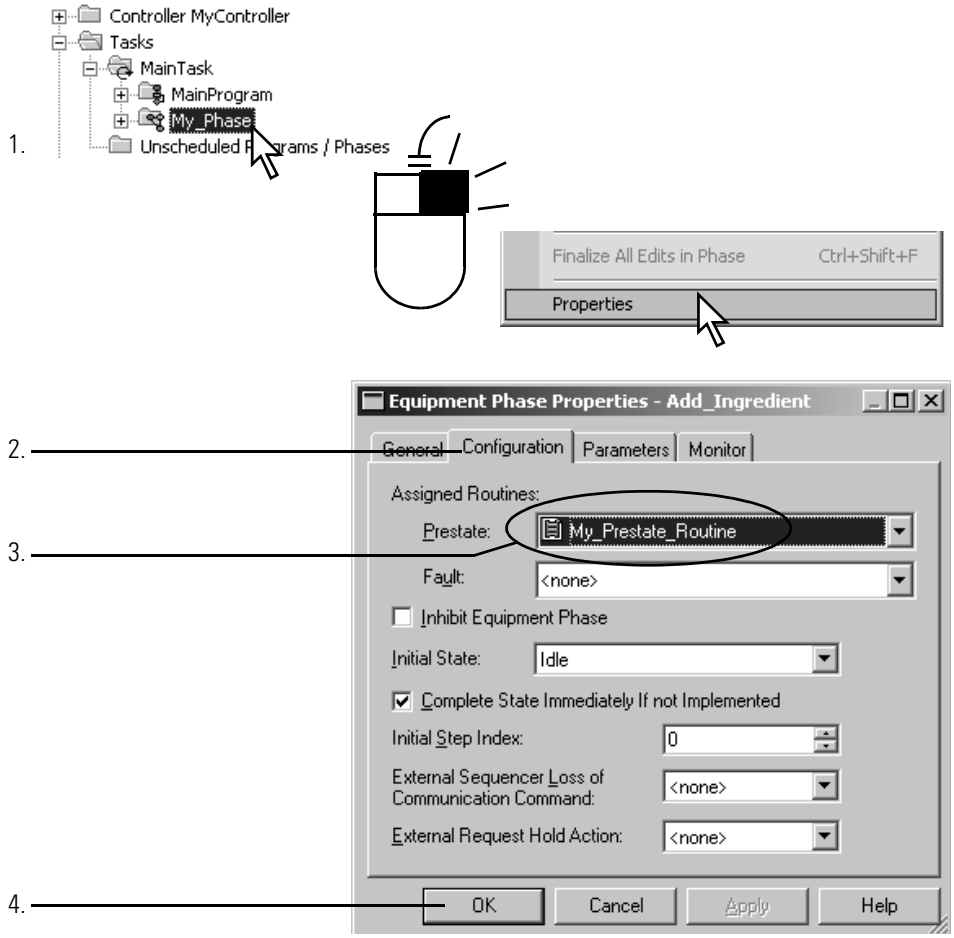
Separate normal execution from exceptions

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



Guideline	Details
1. Use the prestate routine to watch for faults	 Use the prestate routine for conditions that you want to watch all the time such as fault bits. The prestate routine: • runs all the time • runs before *each* scan of a state • runs even in the waiting states (idle, held, complete, stopped, or aborted)

Guideline	Details
2. Create a prestate routine just like the routine for a program. It's not a phase state routine.	1.  

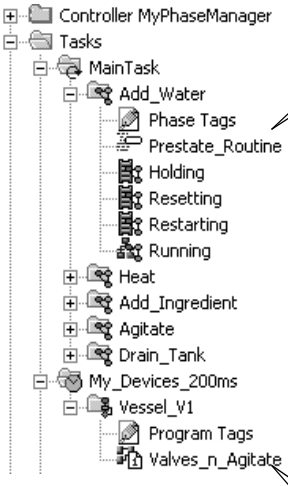
Guideline	Details
3. Assign a prestate routine.	
4. Use a state bit to limit code to a specific state	RSLogix 5000 software automatically makes a tag for each equipment phase. The tag has bits that tell you the state of the equipment phase. • The tag is at the controller scope. • The tag uses the PHASE data type. • Use bits of the tag for code that you want to limit to certain states. **Example** Suppose the name of your equipment phase is *My_Phase*. 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 *My_Phase.Running* bit for on (1): `If My_Phase.Running then…` See Appendix B for more information.

Guideline	Details

5. 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, look at 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 ⇒ idle state.
	  • A PCLF instruction clears the failure code.
	  • RSBizWare Batch software clears the failure code.
	
	See page A-17 for more information.

6. 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 page A-21 for more information.

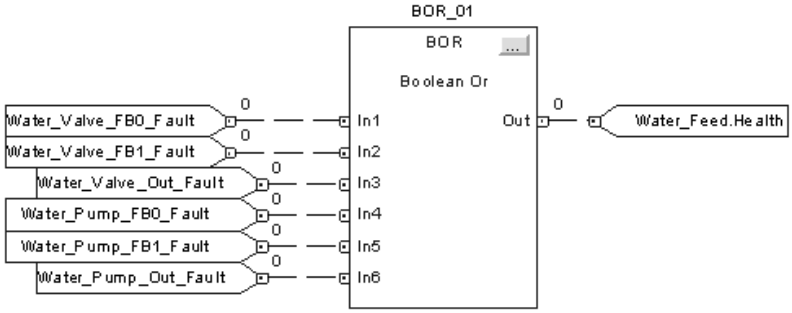
Example 1: Add water to a tank



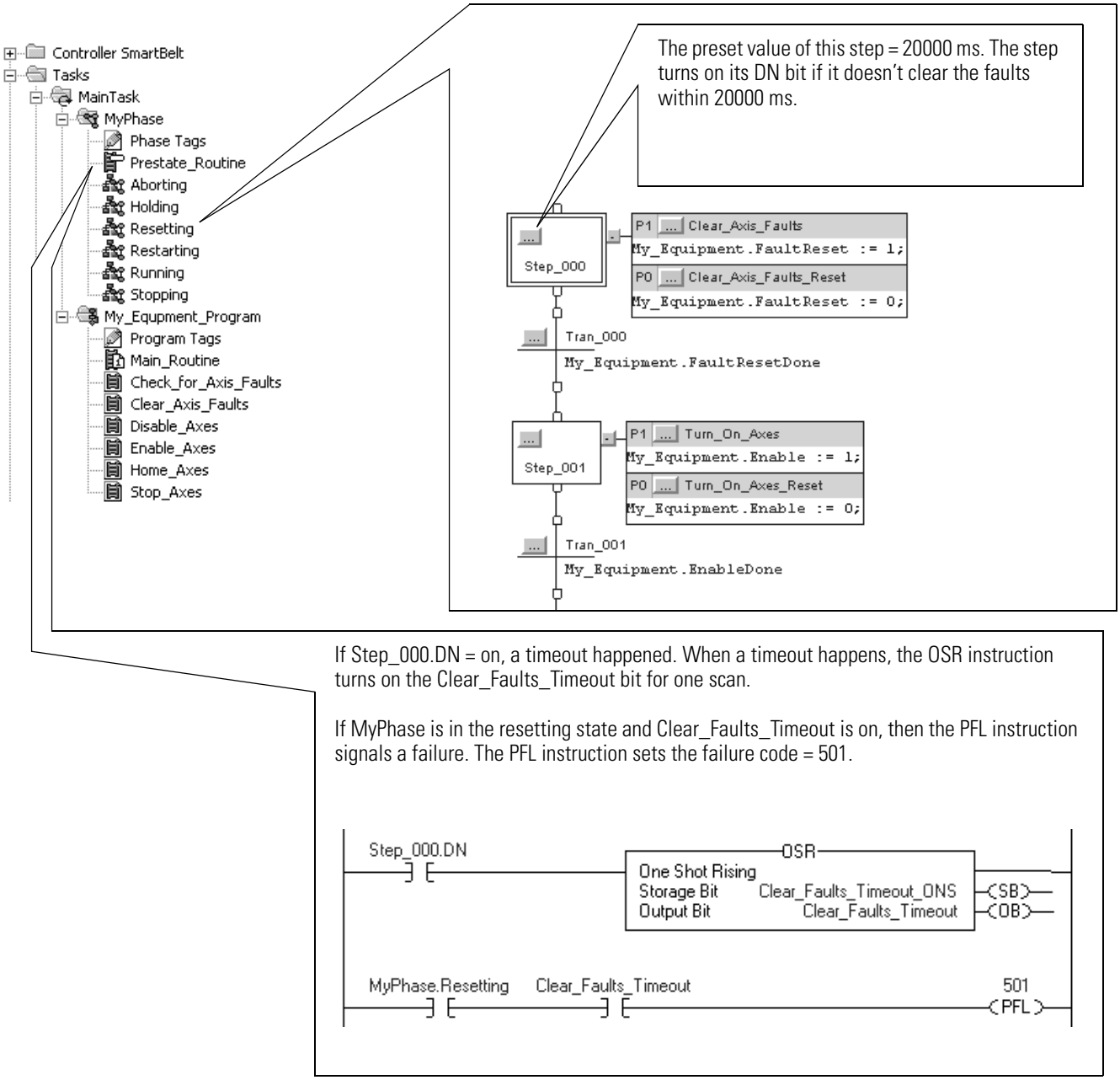
The prestate routine watches for equipment faults while the equipment phase is in the running state (*Add_Water.Running* = 1). If *Water_Feed.Health* = 1, then a fault happened. If a fault happens, the equipment phase sets a failure code of 202.

```
If Add_Water.Running And Water_Feed.Health Then

     PFL(202);

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 *Water_Feed.Health* bit.



Example 2: Smart belt

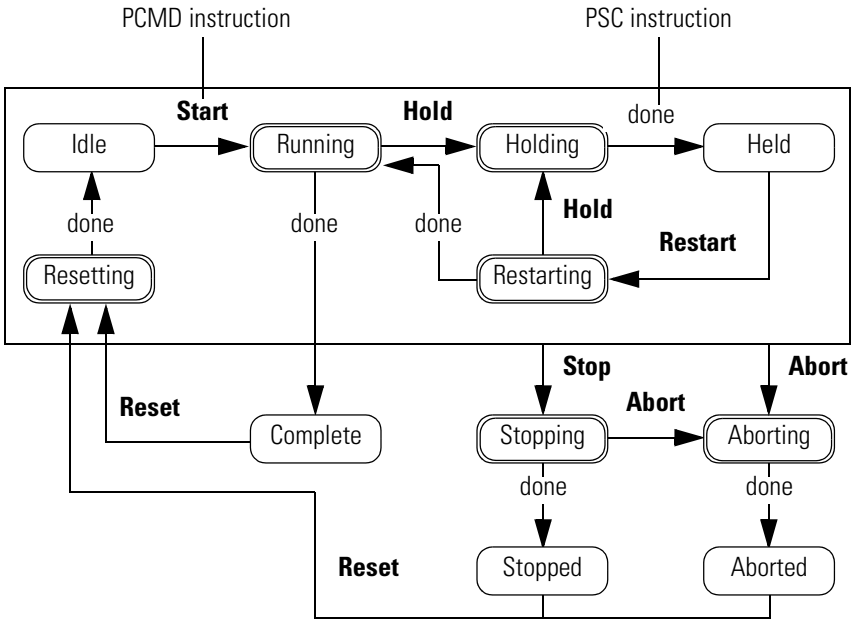


Use the PCMD instruction to transition to a different state

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 RSBizWare Batch software in place of a PCMD instruction to trigger transitions

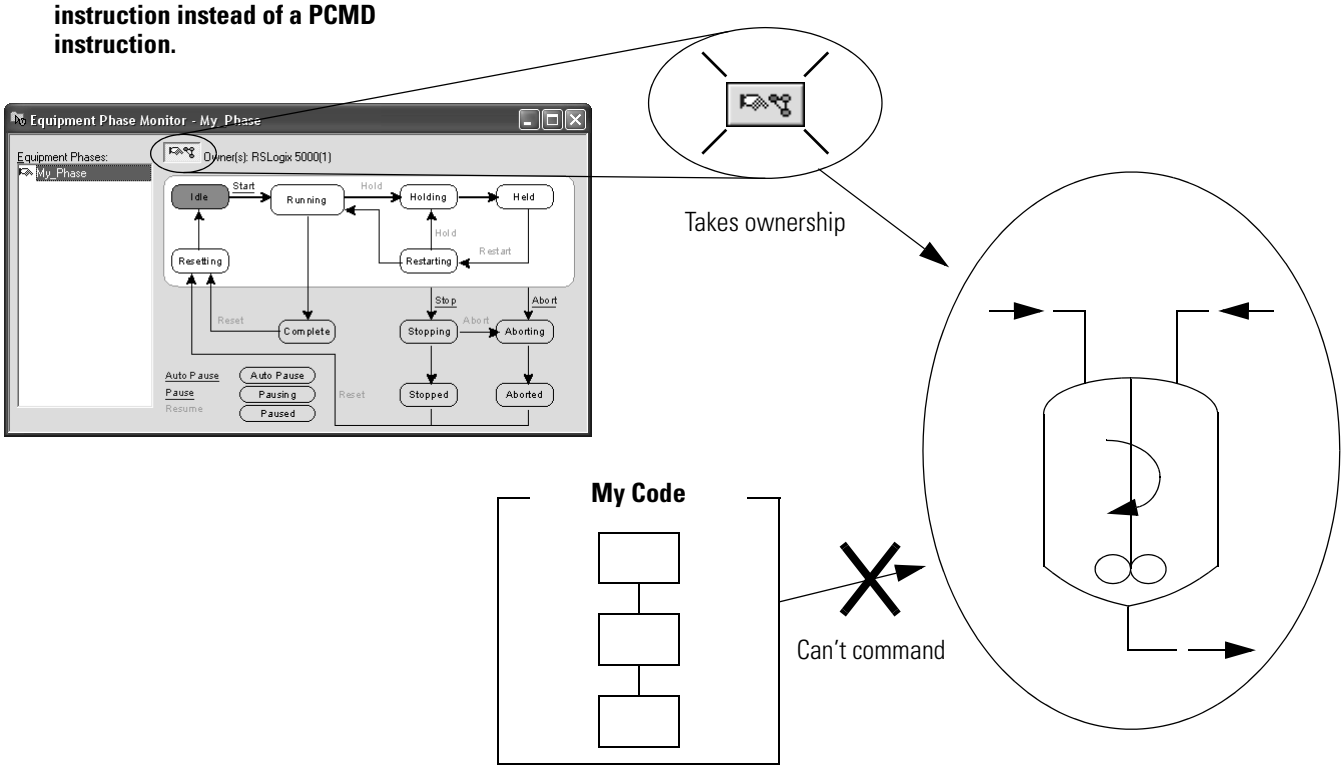
Use the state model to see which transitions need a PCMD instruction.



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 shutdown. PhaseManager uses these commands: reset stop restart start hold abort	PCMD Use an Equipment Phase Command (PCMD) instruction to give a command. Or use RSLogix 5000 software. See page A-8 for more information.
Done	Equipment goes to a waiting state when it's done with what it's doing. You set up 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 page A-5 for more information.

Guideline	Details
1. 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 happens even if the current state isn't done.
2. See if you need to reset the state that you've left.	Are you leaving an acting state (e.g., running, holding)? • YES — Consider resetting the code of the state that you've left. • NO — You probably don't need to reset anything. The equipment phase stops running the code of the current state when it goes to a different state. This leaves outputs at their last values unless the new state takes control of them. It also leaves an SFC at the step it was at when the equipment phase changed states. **Example 1: You don't need to reset** Suppose your equipment phase is in the idle state. In that case, it isn't running any state code. So you probably don't need to reset any state when you go to a different state like running, stopping, etc. **Example 2: You don't need to reset** Suppose your equipment phase is in the running state and you go to the holding state. When you go back to the running state, you probably want to pick up where you left off. In that case, you probably don't need to reset the code in the running state. **Example 3: You must reset** Suppose your equipment phase is half way through the resetting state and you give the stop command. And suppose you want to run the entire resetting sequence when you go back to it. In that case, you probably need to 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.
3. 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: **To reset the SFC of this state:** — **Place an SFR instruction in this state routine:** Running — Resetting Holding — Holding—Let the SFC reset itself at the last step. Restarting — Reset the restarting routine in both these routines: • Holding—In case you go back to holding before you finish restarting. • Restarting—Let the SFC reset itself at the last step.
4. Use the PCMD instruction to go to an allowed next state.	PhaseManager 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 your equipment phase is in the running state and you give it the hold command. In that case, the equipment phase goes to holding since that transition is allowed. **Example 2: A transition isn't allowed** Suppose 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.

Guideline	Details
5. See if you must use a POVR instruction instead of a PCMD instruction.	

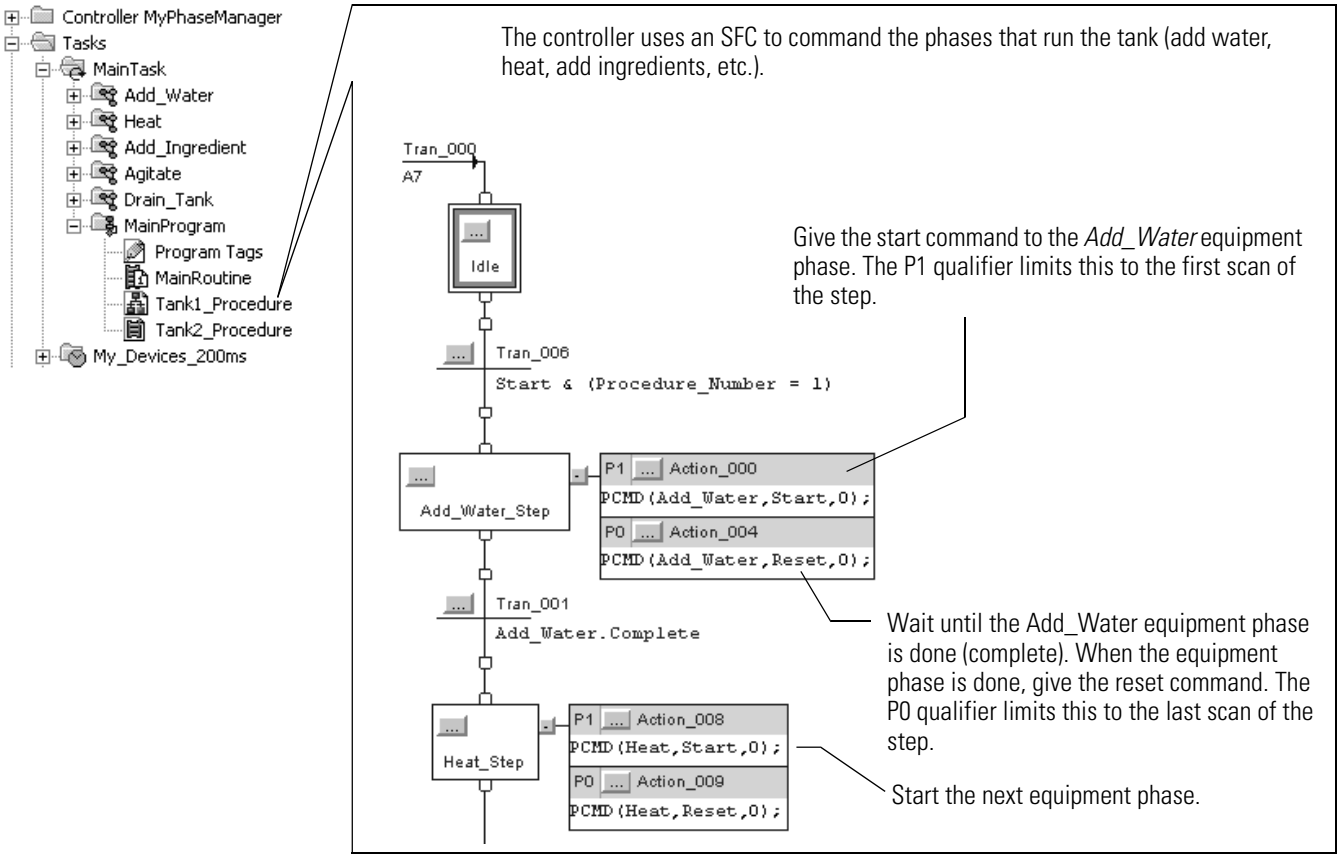


A. Are you giving the hold, stop, or abort command?
- NO — Use the PCMD instruction.
- YES — Go to step B.

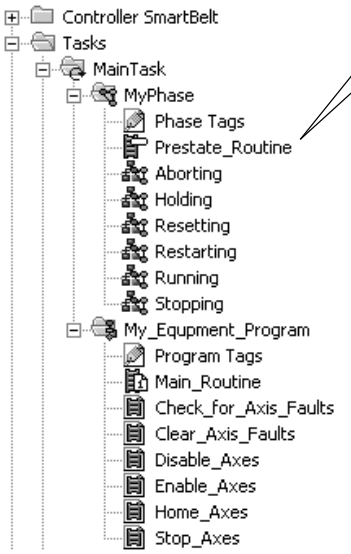
B. Must the command work even if you have manual control of the equipment phase via RSLogix 5000 software?
- YES — Use the POVR instruction instead. See page A-13.
- NO — Go to step C.

C. Must the command work even if RSBizWare Batch software or another program owns the equipment phase?
- YES — Use the POVR instruction instead. See page A-13.
- NO — Use the PCMD instruction.

Example 1: Tank



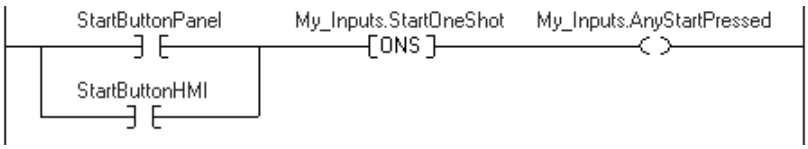
Example 2: Smart belt



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

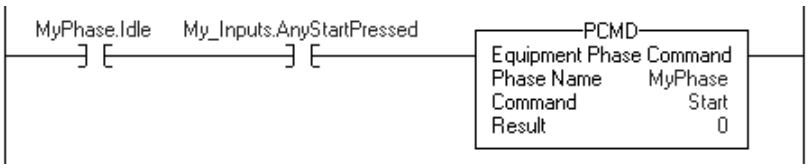
My_Inputs.AnyStartPressed = on for 1 scan.

The ONS instruction makes sure that *My_Inputs.AnyStartPressed* turns on only when a start button goes from off ⇒on.

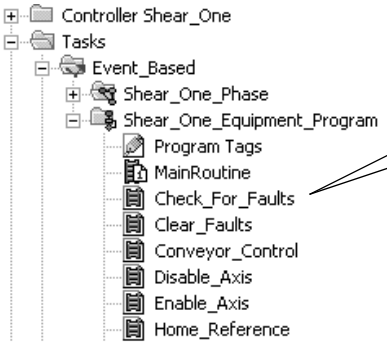


If the equipment phase is in the idle state and *My_Inputs.AnyStartPressed* = on, then

The PCMD instruction gives *MyPhase* the start command.



Example 3: Jam Detection

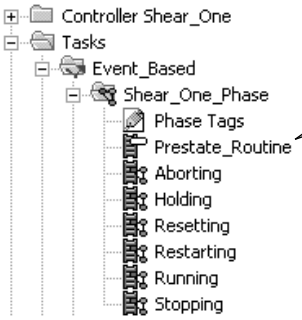


The equipment program watches for the following faults:

- faulted axis
- jammed material

If there is a fault, then

Local_Interface.Equipment_Faults_Cleared = 0. This tag is an alias for the controller-scoped tag *Shear_1*.



The prestate routine of the equipment phase watches for the equipment program to signal a fault.

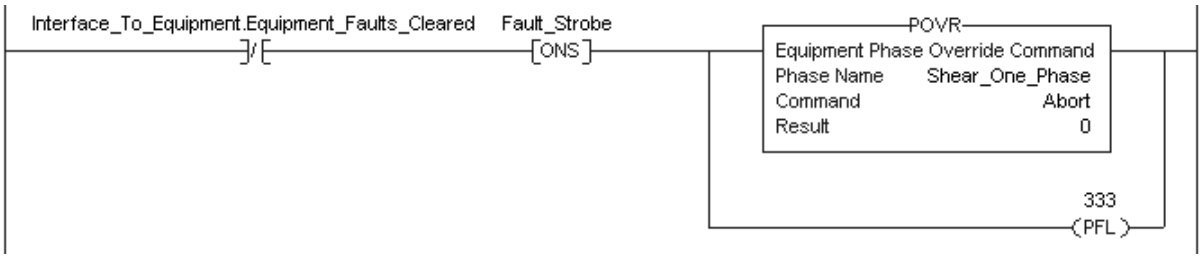
- If *Interface_To_Equipment.Equipment_Faults_Cleared* = 0 then there is a fault.
- Both *Interface_To_Equipment* and *Local_Interface* are aliases for *Shear_1*, so they have the same values.

If there is a fault Then

Give the *Shear_One_Phase* equipment phase the abort command. The POVR instruction makes sure 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 *Shear_One_Phase* = 333.

The *Fault_Strobe* keeps these actions to a single scan.



Use a PSC instruction to signal when a state is done

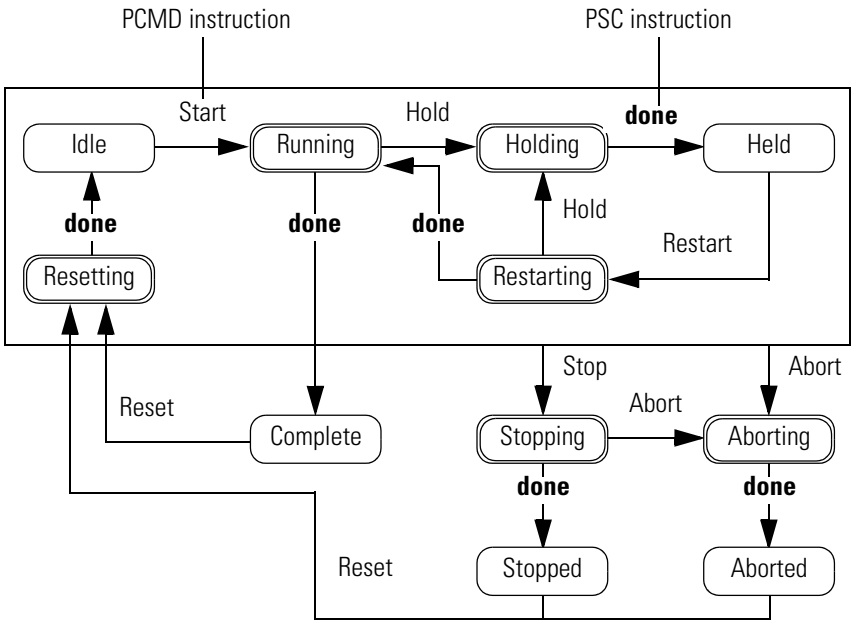
To leave an acting state, you usually signal that the state is done doing what it had to do. Use the Phase State Complete (PSC) instruction to signal when a state is done.

IMPORTANT

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

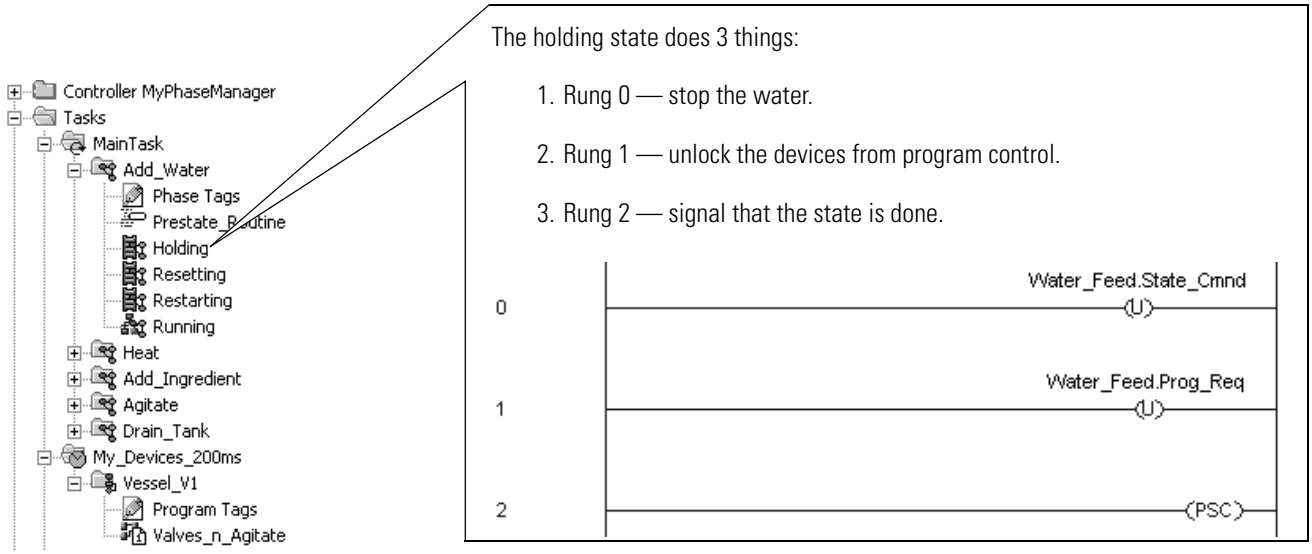
When the PSC instruction executes, the controller scans the rest of the routine and then transitions the equipment phase to the next state. The PSC instruction *does not* terminate the execution of the routine.

Use the state model to see which transitions need a PSC instruction.

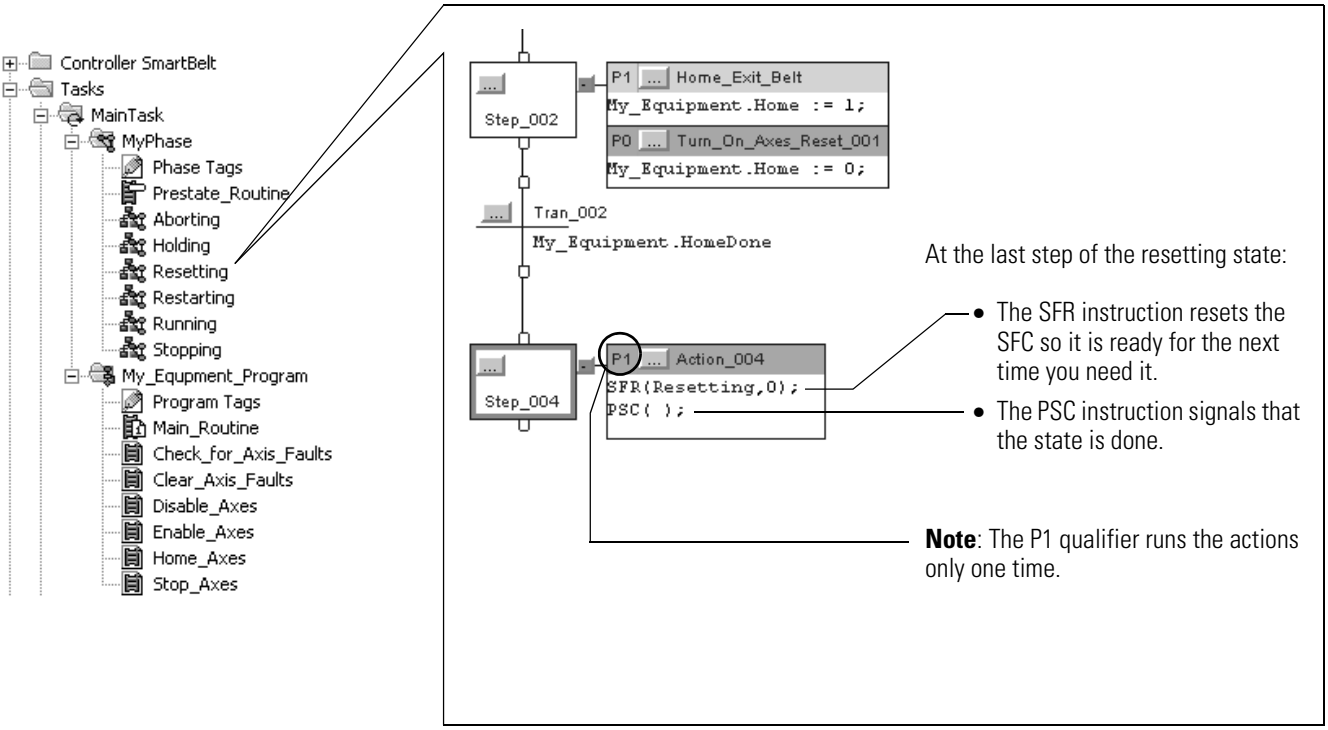


Type of transition	Description	Instruction
Command	A command tells the equipment to start doing something or do something different. For example the operator pushes the start button to start production and the stop button to shutdown. PhaseManager uses these commands: reset, stop, restart start, hold, abort	PCMD Use an Equipment Phase Command (PCMD) instruction to give a command. Or use RSLogix 5000 software.
Done	Equipment goes to a waiting state when it's done with what it's doing. You set up 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 page A-5 for more information.

Example 1: Add water to a tank



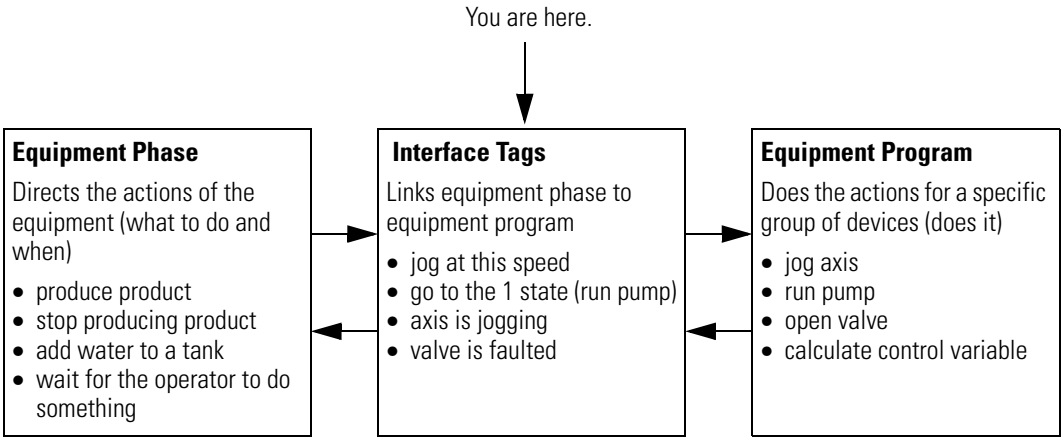
Example 2: Smart belt



Create equipment interface tags

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.



Guideline	Details:	
1. 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. Do *not* include I/O data.	
	Inputs to the equipment program	**Outputs from the equipment program**
	• mode requests • set points • commands such as on, off, start, stop, reset • permissives • overrides	• mode status • control values • done or completion • alarms • faults • health indication • totals or accumulated values

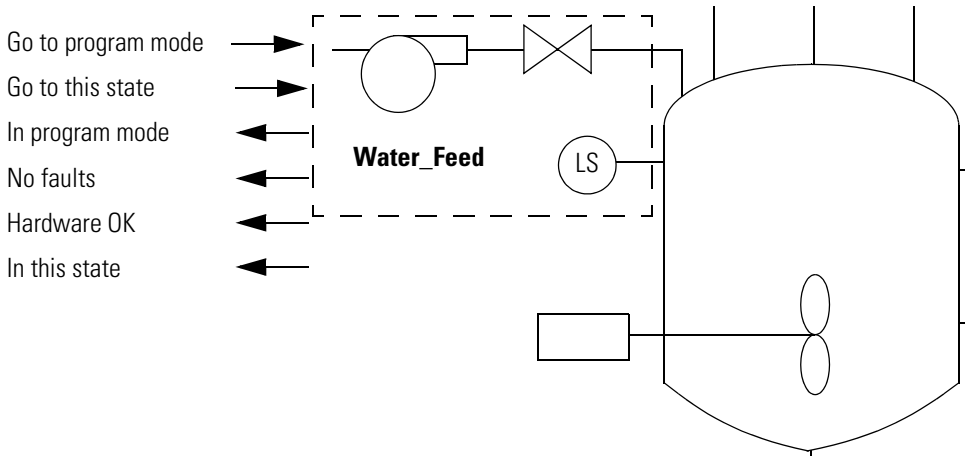
Guideline	Details:
2. Create a user-defined data type	A user-defined data type lets you make a template for your data. It lets you group related data into a single data type. You then use the data type to make tags with the same data lay-out. If you have more than one equipment phase, lay out the data type so that it's easy to use with more than one equipment phase. 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_Cmnd* lets you use it for any equipment that runs in 2 states like on/off, running/not running, pumping/not pumping. It is easier to re-use than names such as *Open* or *Close*. Those names apply to valves but not pumps or motors.
3. 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 2. • 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 *Use alias tags* on page 3-31.

For more information…

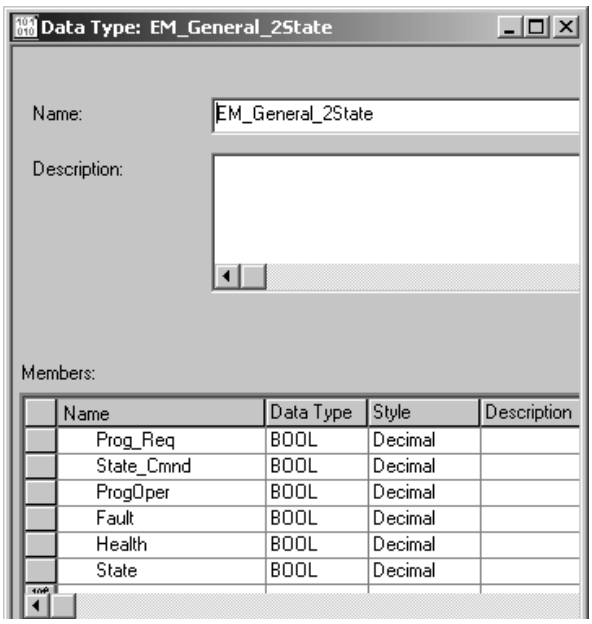
For this information:	See this publication:
guidelines and considerations regarding: • user-defined data types • alias tags	*Logix5000 Controllers Design Considerationsl*, 1756-RM094
step-by-step procedures on how to: • create user-defined data types • assign alias tags	*Logix5000 Controllers Common Procedures*, publication 1756-PM001

Example 1: Add water to a tank

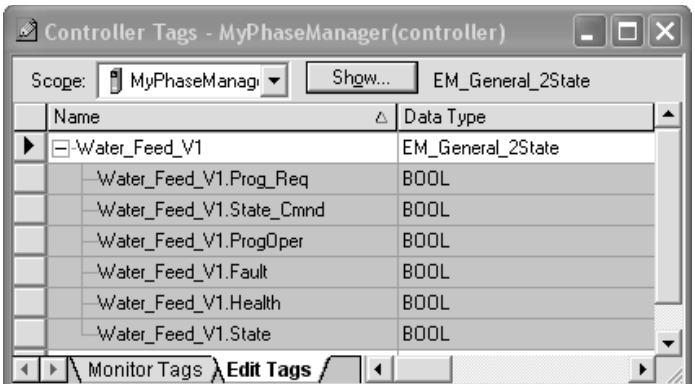
1. The equipment phase and equipment program share this data.



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



3. A tag stores the data that is shared by the equipment phase and equipment program. The tag uses the user-defined data type from step 2.



Example 2: Smart belt

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

Equipment program interface

Commands		Conditions or status	
Enable	Abort	FaultScroll	EnableCyclingDone
Disable	FaultReset	Faulted	DisableCyclingDone
Home	Stop	EnableDone	AbortingDone
ActivateRun	ArmRegistration	DisableDone	FaultResetDone
EnableProduct		HomeDone	StoppingDone
DisableProduct		ActivateRunDone	Selected
EnableCycling		EnableProductDone	RegistrationArmed
DisableCycling		DisableProductDone	

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

Axis interface

Commands		Conditions or status		
Enable	Abort	State	NoMotion	MoveActive
Disable	Stop	On	Homed	HomeDone
Home	ActivateRun	Ok	AxisSelected	RunDone
AutoRun		Auto	GearActive	
ResetFaults		Jogging	CamActive	

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

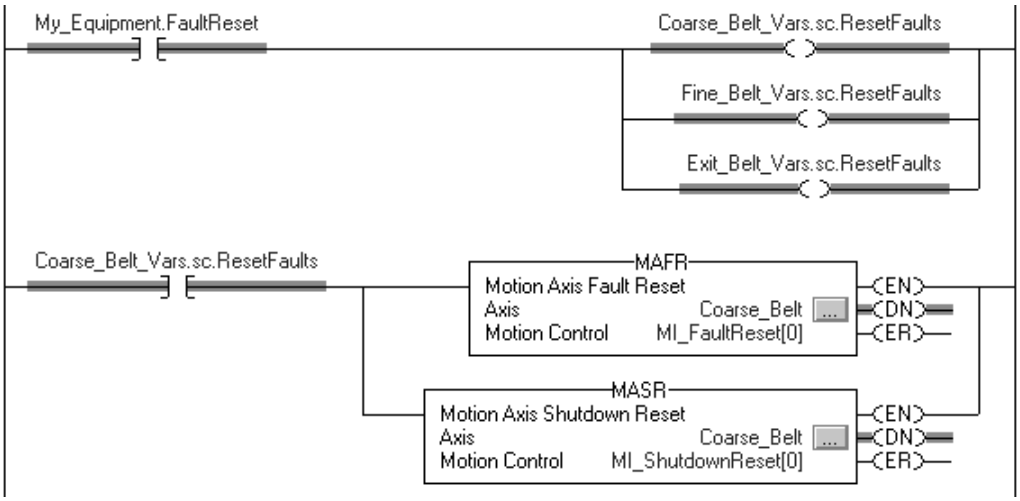
One tag stores the data that is shared by the equipment phase and equipment program. Other tags store the data for each individual axis.

Example 2: Smart belt, Continued

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

Routine of the equipment program

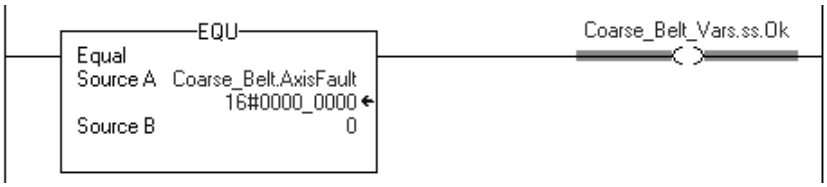
This tag	Is the interface between
My_Equipment	equipment phase and equipment program
Coarse_Belt_Vars	equipment program and an axis



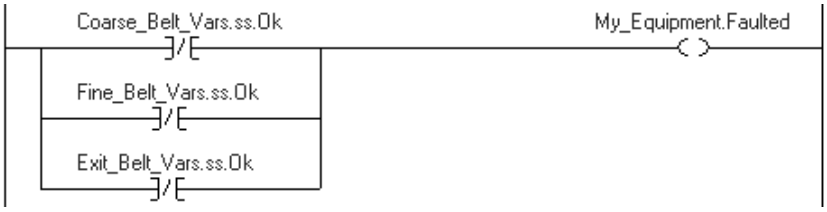
5. The equipment program collects the fault status of each axis and passes it back to the equipment phase.

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.



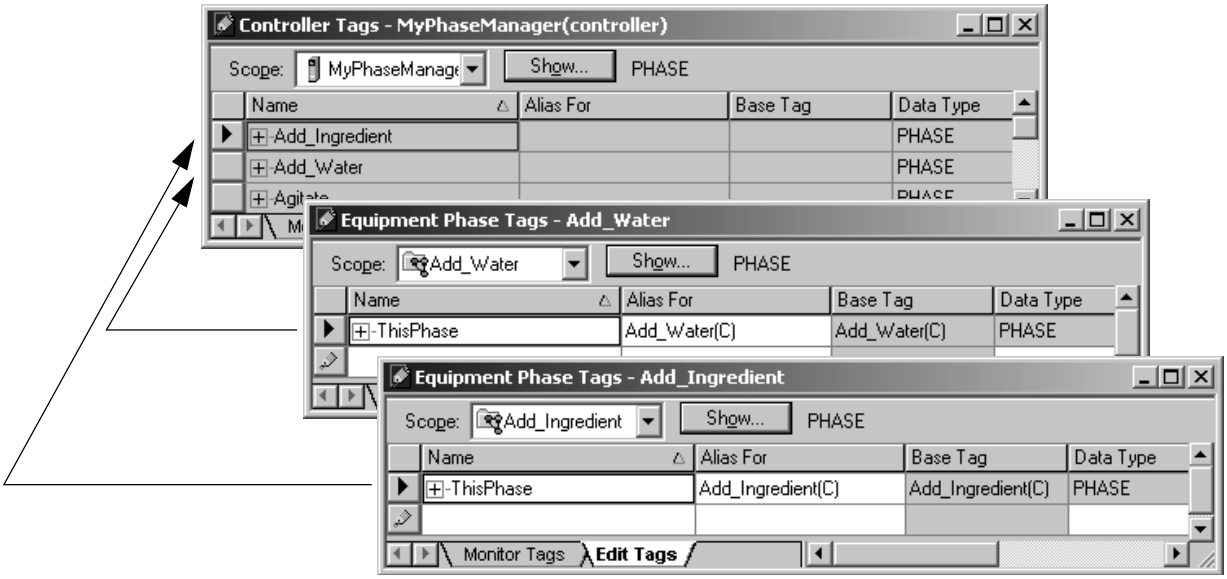
The equipment program collects the OK status of each axis. If the OK bit of each axis = on, then *My_Equipment.Faulted* = off (no faults).



Use alias tags

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 (e.g., copy/paste), simply point the phase-scoped tags to new tags at the controller scope. This 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 you plan to reuse an equipment phase for a different part of your tank. In that case:

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 (ThisPhase).
3. Make a copy of the equipment phase.
4. Point the alias tag of the copy to its controller tag.

For more information…

For this information:	See this publication:
guidelines and considerations for alias tags	*Logix5000 Controllers Design Considerationsl*, 1756-RM094
steps to assign alias tags	*Logix5000 Controllers Common Procedures*, publication 1756-PM001

Notes:

Equipment Phase Instructions
(PSC, PCMD, POVR, PFL, PCLF, PXRQ, PRNP, PPD, PATT, PDET)

Purpose of This Appendix

This appendix provides a description of each equipment phase instruction in this format.

This section:	Provides this type of information:
Instruction name	identifies the instruction
Operands	lists all the operands of the instruction
Instruction structure	lists control status bits and values, if any, of the instruction
Description	describes the instruction's use defines any differences when the instruction is enabled and disabled, if appropriate
Arithmetic status flags	defines whether or not the instruction affects arithmetic status flags
Fault conditions	defines whether or not the instruction generates minor or major faults if so, defines the fault type and code
Execution	defines the specifics of how the instruction operates
Example	provides at least one programming example in each available programming language includes a description explaining each example

The following icons help identify language specific information:

This icon:	Indicates this programming language:
	relay ladder
	structured text

Conventions and Related Terms

Set and Clear

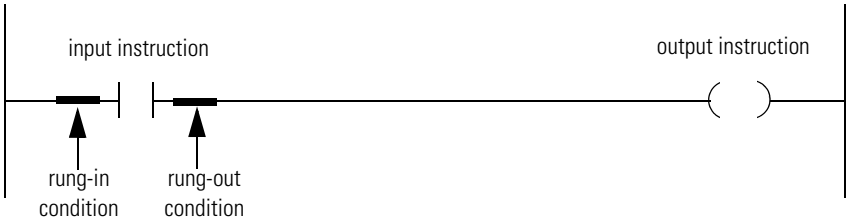
This manual uses set and clear to define the status of bits (booleans) and values (non-booleans):

This term:	Means:
set	the bit is set to 1 (ON) a value is set to any non-zero number
clear	the bit is cleared to 0 (OFF) all the bits in a value are cleared to 0

If an operand or parameter support more than one data type, the **bold** data types indicate optimal data types. An instruction executes faster and requires less memory if all the operands of the instruction use the same optimal data type, typically DINT or REAL.

Relay Ladder Rung Condition

The controller evaluates ladder instructions based on the rung condition preceding the instruction (rung-condition-in). Based on the rung-condition-in and the instruction, the controller sets the rung condition following the instruction (rung-condition-out), which in turn, affects any subsequent instruction.



If the rung-in condition to an input instruction is true, the controller evaluates the instruction and sets the rung-out condition based on the results of the instruction. If the instruction evaluates to true, the rung-out condition is true; if the instruction evaluates to false, the rung-out condition is false.

Prescan of Routines

The controller also prescans routines. Prescan is a special scan of all routines in the controller. During prescan, the controller:

- scans all main routines
- scans all state routines of equipment phases
- scans all prestate routines of equipment phases
- scans all subroutines of programs and equipment phases one time

  Once the controller prescans a subroutine, it *does not* prescan the subroutine again during that prescan.
- scans all routines called by FOR instructions of a ladder diagram routine
- ignores jumps that could skip the execution of instructions
- executes all instructions in prescan mode

  For details on how a specific instruction operates during prescan, see the *Execution* section for the instruction.
- resets to 0 all non-retentive assignments ([:=])
- *does not* update input values
- *does not* write output values

The following conditions generate prescan:

- Toggle from Program to Run mode
- Automatically enter Run mode from a power-up condition

Prescan does not occur for a program or equipment phase when:

- The program or equipment phase becomes scheduled while the controller is running.
- The program or equipment phase is unscheduled when the controller enters Run mode.

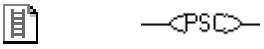
Choose an Equipment Phase Instruction

If you want to:	Use this instruction:	Available in these languages:	See page:
signal an equipment phase that the state routine is complete so go to the next state	Phase State Complete (PSC)	relay ladder structured text	A-5
change the state or substate of an equipment phase	Equipment Phase Command (PCMD)	relay ladder structured text	A-8
give a hold, stop, or abort command to an equipment phase regardless of ownership	Equipment Phase Override Command (POVR)	relay ladder structured text	A-13
signal a failure for an equipment phase	Equipment Phase Failure (PFL)	relay ladder structured text	A-17
clear the failure code of an equipment phase	Equipment Phase Clear Failure (PCLF)	relay ladder structured text	A-21
initiate communication with RSBizWare Batch software	Equipment Phase External Request (PXRQ)	relay ladder structured text	A-23
clear the NewInputParameters bit of an equipment phase	Equipment Phase New Parameters (PRNP)	relay ladder structured text	A-34
set up breakpoints within the logic of an equipment phase	Equipment Phase Paused (PPD)	relay ladder structured text	A-37
take ownership of an equipment phase to either: • prevent another program or RSBizWare Batch software from commanding an equipment phase • make sure another program or RSBizWare Batch software does *not* already own an equipment phase	Attach to Equipment Phase (PATT)	relay ladder structured text	A-42
relinquish ownership of an equipment phase	Detach from Equipment Phase (PDET)	relay ladder structured text	A-47

Phase State Complete (PSC)

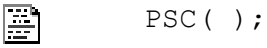
Use the PSC instruction to signal an equipment phase that the state routine is complete so go to the next state.

Operands:



Relay Ladder

none

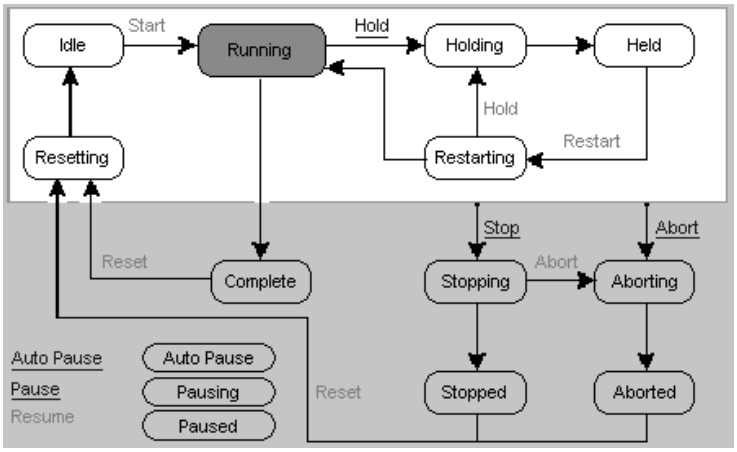


Structured Text

none

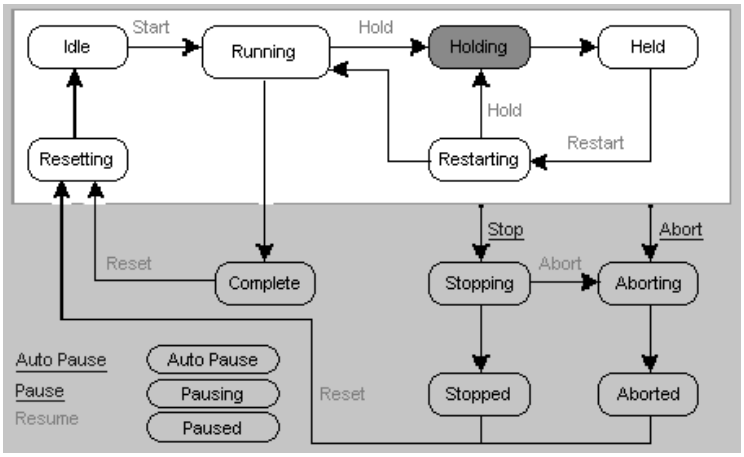
You must enter the parentheses ( ) after the instruction mnemonic, even though there are no operands.

Description: The PSC instruction signals the completion of a phase state routine.



In the running state routine, use the PSC instruction to transition the equipment phase to the complete state.

Guidelines for Using the PSC Instruction

Guideline:	Details:
❑ **Use the PSC instruction in *each* phase state routine that you add to an equipment phase.**	Without a PSC instruction, the equipment phase remains in the state and *does not* go to the next state. • Place the PSC instruction as the last step in your phase state routine. • When the state is done (complete), execute the PSC instruction.
	 In the holding state routine, use the PSC instruction to let the equipment phase go to the Held state.
❑ **Remember that the PSC instruction *does not* 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.
❑ ***Do not* use a PSC instruction in a prestate routine.**	Use the PSC instruction only to signal the transition from one state to another.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction signals that the state is complete.	The instruction signals that the state is complete.
postscan	The rung-condition-out is set to false.	No action taken.

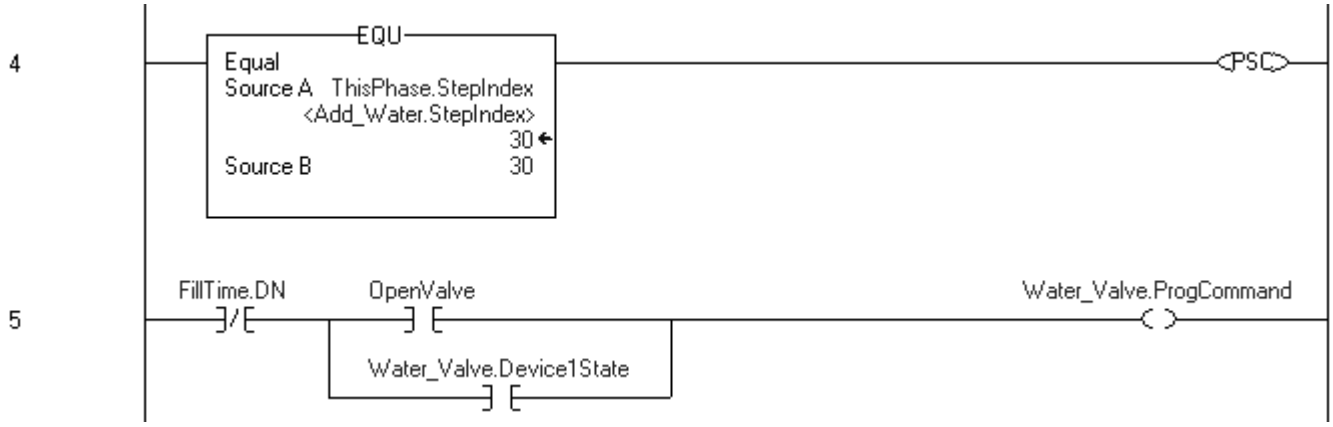
Example:

Relay Ladder

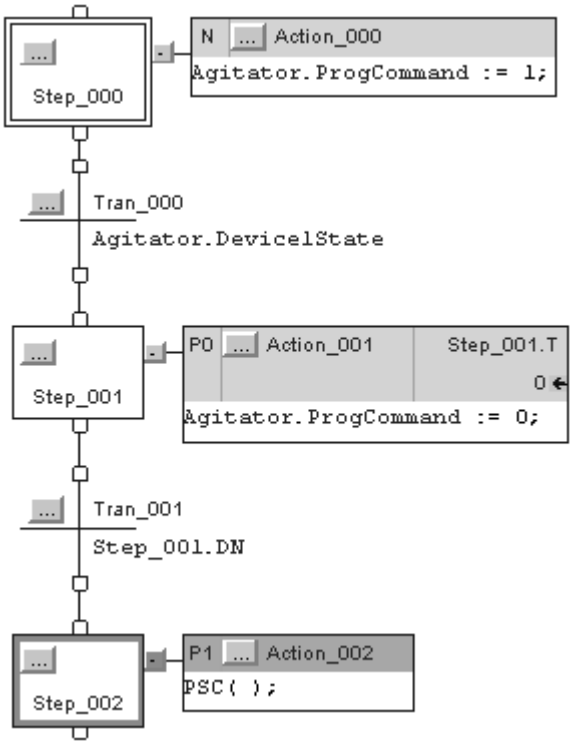
If ThisPhase.StepIndex = 30 (The routine is at step 30.)

Then the PSC instruction signals that the state is done (complete).

After the controller scans the rest of the routine (rung 5, rung 6, etc.), the equipment phase goes to the next state.



Structured Text

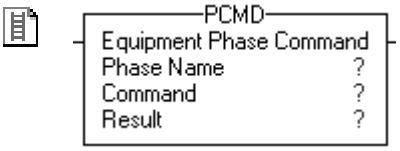


When the SFC reaches Step_002, the PSC instruction signals that the state is done (complete).

Equipment Phase Command (PCMD)

Use the PCMD instruction to change the state or substate of an equipment phase.

Operands:



Relay Ladder

Operand:	Type:	Format:	Description:
Phase Name	phase	name of the equipment phase	Equipment phase that you want to change to a different state
Command	command	name of the command	Command that you want to send to the equipment phase to change its state For available commands, see Figure A.1.
Result	DINT	immediate tag	To let the instruction return a code for its success/failure, enter a DINT tag in which to store the result code. Otherwise, enter 0.

Structured Text

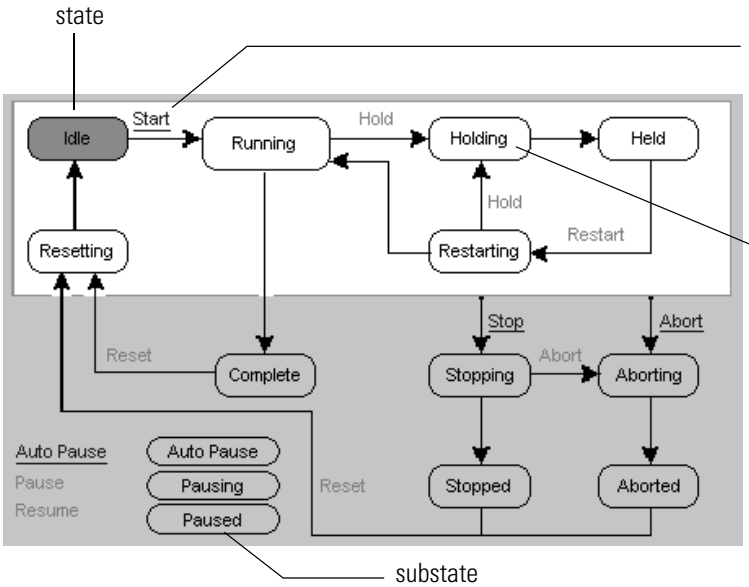
```
PCMD(PhaseName,
Command, Result);
```

The operands are the same as those for the relay ladder PCMD instruction.

Description:

The PCMD instruction transitions an equipment phase to the next state or substate.

Figure A.1 States, Substates, and Commands of an Equipment Phase



command

Some states need a command to go to the next state. If the equipment phase is in the idle state, a start command transitions the equipment phase to the running state. Once in the running state, the equipment phase executes its running state routine.

Other states use a Phase State Complete (PSC) instruction to go to the next state. If the equipment phase is in the holding state, a PSC instruction transitions the equipment phase to the held state. Once in the held state, the equipment phase needs a restart command to go to the restarting state.

Use the auto pause, pausing, and paused substates to test and debug a state routine. The substates require the Equipment Phase Paused (PPD) instruction to create breakpoints in your logic. The auto pause, pause, and resume commands let you step through the breakpoints.

Guidelines for Using the PCMD Instruction

Guideline:	Details:
❑ **Limit execution of a PCMD instruction to a single scan.**	Limit the execution of the PCMD instruction to a single scan. Each command applies to only a specific state or states. Once the equipment phase changes state, the command is *no longer* valid. To limit execution, use methods such as: • Execute the PCMD instruction within a P1 Pulse (Rising Edge) or P0 Pulse (Falling Edge) action • Place a one shot instruction before the PCMD instruction • Execute the PCMD instruction and then advance to the next step
❑ **Determine if you need to take ownership of the equipment phase.**	As an option, a program can own an equipment phase. This prevents another program or RSBizWare Batch software from also commanding the equipment phase.

If you use:	Then:
RSBizWare Batch software to also run procedures (recipes) within this controller	Before you use a PCMD instruction, use an Attach to Equipment Phase (PATT) instruction to take ownership of the equipment phase. See page A-42.
multiple programs to command the same equipment phase	
none of the above	There is no need to own 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 page A-13.
- NO — Go to step C.

C. Must the command work even if RSBizWare Batch software or another program owns the equipment phase?
- YES — Use the POVR instruction instead. See page A-13.
- NO — Use the PCMD instruction.

Example: Suppose your equipment checks for jammed material. And if there is a jam, you always want the equipment to abort. In that case, use the POVR instruction. This way, the equipment aborts even if you have manual control via RSLogix 5000 software.

❑ **Decide if you want a result code.**

Use the Result operand to get a code that shows the success/failure of the PCMD instruction.

If you:	Then in the Result operand, enter a:
anticipate ownership conflicts or other possible errors	DINT tag in which to store a code for the result of the execution of the instruction
do not anticipate ownership conflicts or other errors	0

To interpret the result code, see "PCMD Result Codes" on page A-10.

PCMD Result Codes

If you assign a tag to store the result of a PCMD instruction, the instruction returns one of the following codes when it executes:

Code (Dec):	Description:
0	The command was successful.
24577	The command is *not* valid.
24578	The command is *not* valid for the current state of the equipment phase. For example, if the equipment phase is in the running state, then a start command is *not* valid.
24579	You *cannot* command the equipment phase. One of the following already owns the equipment phase. • RSLogix 5000 software • external sequencer (RSBizWare Batch software) • another program in the controller
24594	The equipment phase is unscheduled, inhibited, or in a task that is inhibited.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

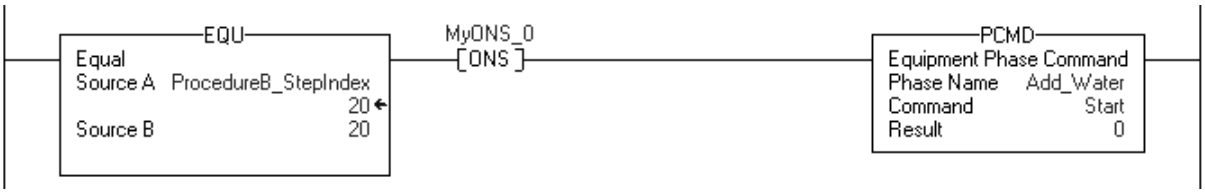
Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction commands the equipment phase to the specified state.	The instruction commands the equipment phase to the specified state.
postscan	The rung-condition-out is set to false.	No action taken.

Example 1:

Relay Ladder

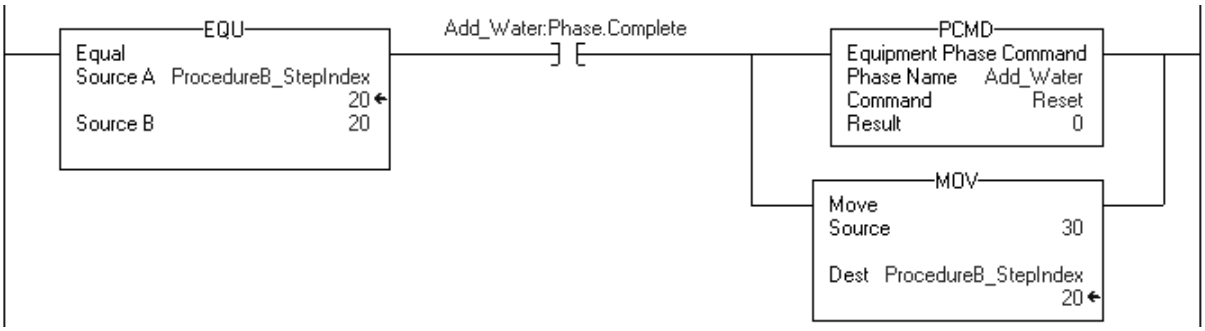
If *ProcedureB_StepIndex* = 20 (The routine is at step 20.)
And this is the transition to step 20 (The ONS instruction signals that the EQU instruction went from false to true.)
Then

Change the state of the *Add_Water* equipment phase to running via the start command.

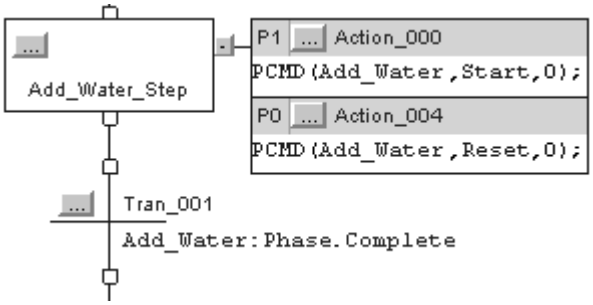


If *ProcedureB_StepIndex* = 20 (The routine is at step 20.)
And the *Add_Water* equipment phase is complete (*Add_Water:Phase.Complete* = 1)
Then

Change the state of the *Add_Water* equipment phase to resetting via the reset command.
Advance to step 30.



Structured Text



When the SFC enters *Add_Water_Step*, change *Add_Water* equipment phase to running via the start command. The P1 qualifier limits this to the first scan of the step.

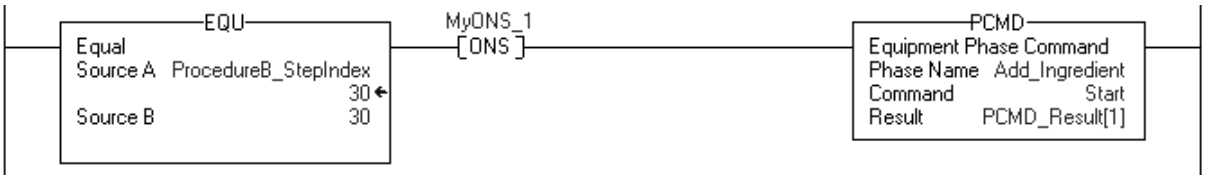
Before the SFC leaves *Add_Water_Step* (*Add_Water:Phase.Complete* = 1), change *Add_Water* equipment phase to resetting via the reset command. The P0 qualifier limits this to the last scan of the step.

Example 2:

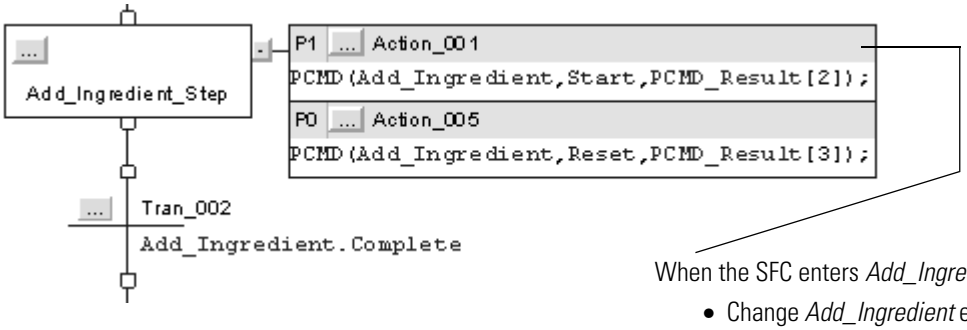
Relay Ladder

If *ProcedureB_StepIndex* = 30 (The routine is at step 30.)
And this is the transition to step 30 (The ONS instruction signals that the EQU instruction went from false to true.)
Then

Change the *Add_Ingredient* equipment phase to running via the start command.
Verify that the command was successful and store the result code in *PCMD_Result[1]* (DINT tag).



Structured Text



When the SFC enters *Add_Ingredient_Step*:

- Change *Add_Ingredient* equipment phase to running via the start command.
- Verify that the command was successful and store the result code in *PCMD_Result[2]* (DINT tag).

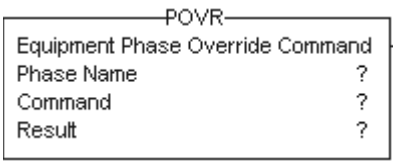
The P1 qualifier limits this to the first scan of the step.

Equipment Phase Override Command (POVR)

Use the POVR instruction to give a hold, stop, or abort command to an equipment phase regardless of ownership.

Operands:

Relay Ladder



Operand:	Type:	Format:	Description:
Phase Name	phase	name of the equipment phase	Equipment phase that you want to change to a different state
Command	command	name of the command	One of these commands for the equipment phase: • hold • stop • abort
Result	DINT	immediate tag	To let the instruction return a code for its success/failure, enter a DINT tag in which to store the result code. Otherwise, enter 0.

Structured Text

```
POVR(PhaseName,
Command, Result);
```

The operands are the same as those for the relay ladder POVR instruction.

Description:

The POVR instruction:

- Gives the hold, stop, or abort command to an equipment phase.
- Overrides all owners of the equipment phase. The command works even if RSLogix 5000 software, RSBizWare Batch software, or another program already own the equipment phase.

Guidelines for Using the POVR Instruction

Guideline:	Details:
❑ **Make sure you want to override other owners.**	Do you want the equipment to hold, stop, or abort even if you have manual control of the equipment phase via RSLogix 5000 software? • YES — Use the POVR instruction. • NO — Use the PCMD instruction. This also applies to RSBizWare Batch software or other programs. Use the POVR only when you must hold, stop, or abort regardless of ownership. Example: Suppose your equipment checks for jammed material. And if there is a jam, you always want the equipment to abort. In that case, use the POVR instruction. This way, the equipment aborts even if you have manual control via RSLogix 5000 software.
❑ **Limit execution of a POVR instruction to a single scan.**	Limit the execution of the POVR instruction to a single scan. Each command applies to only a specific state or states. Once the equipment phase changes state, the command is *no longer* valid. To limit execution, use methods such as: • Execute the POVR instruction within a P1 Pulse (Rising Edge) or P0 Pulse (Falling Edge) action • Place a one shot instruction before the POVR instruction • Execute the POVR instruction and then advance to the next step

POVR Result Codes

If you assign a tag to store the result of a POVR instruction, the instruction returns one of the following codes when it executes:

Code (Dec):	Description:
0	The command was successful.
24577	The command is *not* valid.
24578	The command is *not* valid for the current state of the equipment phase. For example, if the equipment phase is in the stopping state, then a hold command is *not* valid.
24594	The equipment phase is unscheduled, inhibited, or in a task that is inhibited.

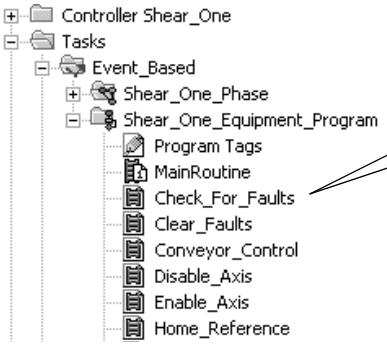
Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction commands the equipment phase to the specified state.	The instruction commands the equipment phase to the specified state.
postscan	The rung-condition-out is set to false.	No action taken.

Example:

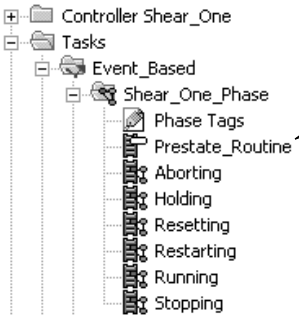


The equipment program watches for the following faults:

- faulted axis
- jammed material

If there is a fault, then

Local_Interface.Equipment_Faults_Cleared = 0. This tag is an alias for the controller-scoped tag *Shear_1*.



The prestate routine of the equipment phase watches for the equipment program to signal a fault.

- If *Interface_To_Equipment.Equipment_Faults_Cleared* = 0 then there is a fault.
- Both *Interface_To_Equipment* and *Local_Interface* are aliases for *Shear_1*, so they have the same values.

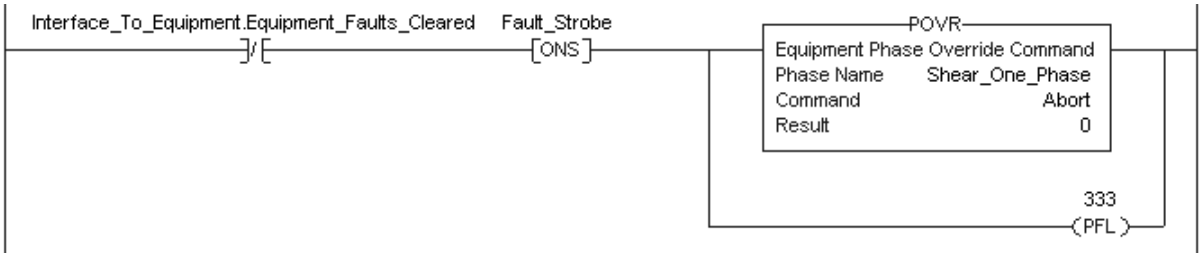
If there is a fault Then

Give the *Shear_One_Phase* equipment phase the abort command. The POVR instruction makes sure 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 *Shear_One_Phase* = 333.

The *Fault_Strobe* keeps these actions to a single scan.

Relay Ladder



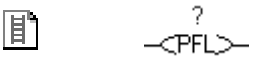
Structured Text

```
If NOT
Interface_To_Equipment.Equipment_Faults_Cleared
And NOT Fault_Strobe Then
    POVR(Shear_One_Phase,Abort,0);
    PFL(333);
End_If;

Fault_Strobe := NOT
Interface_To_Equipment.Equipment_Faults_Cleared;
```

Equipment Phase Failure (PFL)

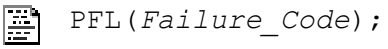
Use the PFL instruction as an optional method to signal a failure for an equipment phase.

Operands:



Relay Ladder

Operand:	Type:	Format:	Description:
Failure_Code	DINT	immediate tag	value to which you want to set the failure code for the equipment phase

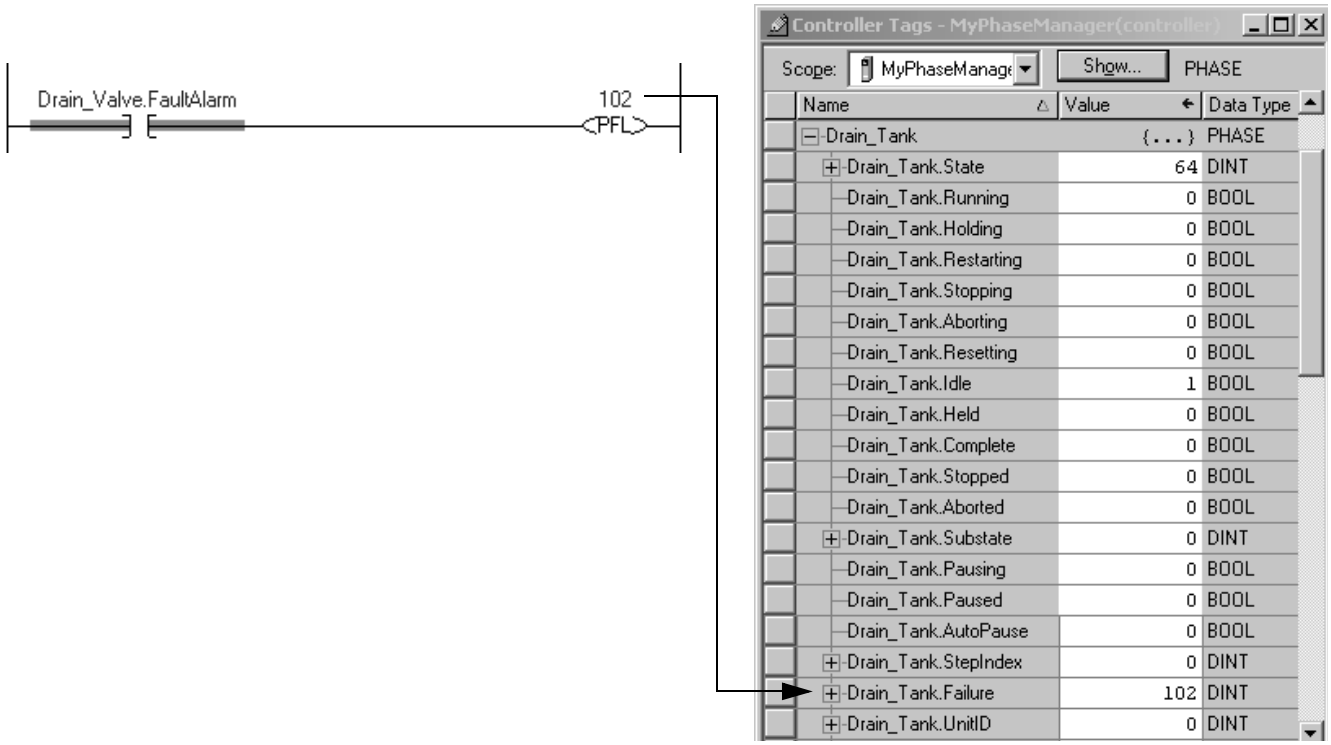


Structured Text

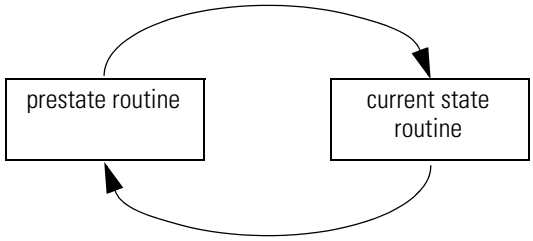
The operands are the same as those for the relay ladder PFL instruction.

Description:

The PFL instruction sets the value of the failure code for an equipment phase. Use the instruction to signal a specific failure for an equipment phase, such as a specific device has faulted.



Guidelines for Using the PFL Instruction

Guideline:	Details:
❑ **Put the PFL instruction in the equipment phase.**	The PFL instruction sets the failure code for the equipment phase in which you put the instruction. There is *no* operand to identify a specific equipment phase. Typically, put the PFL instruction in a prestate routine for the equipment phase. • The controller always scans the prestate routine, even when an equipment phase is in the idle state. • The controller scans the prestate routine before *each* scan of a state.  • Use the prestate routine to continuously monitor the health of an equipment phase as you progress through its states.
❑ **Prioritize your failure codes.**	The PFL instruction sets the failure code only to a value greater than its current value. • For example, if a PFL instruction sets the failure code = 102, another PFL instruction can only set the failure code > 102. • Make sure that you assign higher values to exceptions that require higher priority in their handling. Otherwise, a lower priority exception may overwrite a more critical exception.
❑ **To take action when a failure occurs, monitor the Failure member of the PHASE tag.**	The PFL instruction writes its value to the Failure member of the PHASE tag for the 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

The PFL instruction writes its value to the failure member for the equipment phase.

Controller Tags - MyPhaseManager(controller)

Scope: MyPhaseManage | Show... | PHASE

Name	Value	Data Ty
Drain_Tank	{...}	PHASE
Drain_Tank.State	64	DINT
Drain_Tank.Running	0	BOOL
Drain_Tank.Holding	0	BOOL
Drain_Tank.Restarting	0	BOOL
Drain_Tank.Stopping	0	BOOL
Drain_Tank.Aborting	0	BOOL
Drain_Tank.Resetting	0	BOOL
Drain_Tank.Idle	1	BOOL
Drain_Tank.Held	0	BOOL
Drain_Tank.Complete	0	BOOL
Drain_Tank.Stopped	0	BOOL
Drain_Tank.Aborted	0	BOOL
Drain_Tank.Substate	0	DINT
Drain_Tank.Pausing	0	BOOL
Drain_Tank.Paused	0	BOOL
Drain_Tank.AutoPause	0	BOOL
Drain_Tank.StepIndex	0	DINT
Drain_Tank.Failure	102	DINT

Guideline:	Details:
❑ **To clear the failure code, use a PCLF instruction.**	You must use a PCLF instruction to clear the failure code of an equipment phase. Instructions such as a CLR or MOV *won't* change the failure code. See Equipment Phase Clear Failure (PCLF) instruction on page A-21.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction sets the value of the failure code for the equipment phase.	The instruction sets the value of the failure code for the equipment phase.
postscan	The rung-condition-out is set to false.	No action taken.

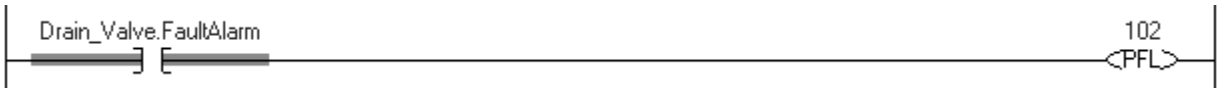
Example:

Relay Ladder

In the prestate routine of an equipment phase…

If *Drain_Valve.FaultAlarm* = 1 (The valve did not go to the commanded state.) then

Failure code for the equipment phase = 102.

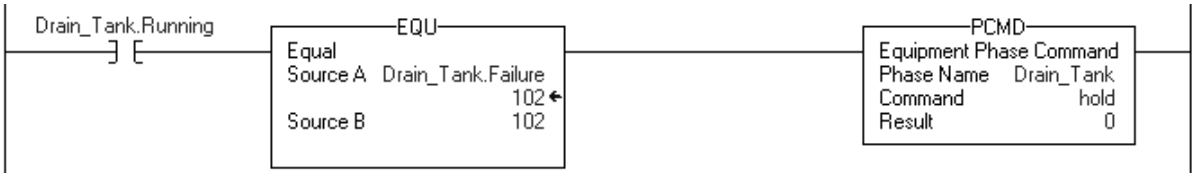


If *Drain_Tank.Running* = 1 (The *Drain_Tank* equipment phase is in the running state.)

And *Drain_Tank.Failure* = 102 (failure code for the equipment phase)

Then

Change the state of the *Drain_Tank* equipment phase to holding via the hold command.



Structured Text

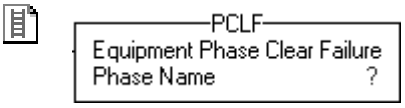
In the prestate routine of an equipment phase…

```
(*If the drain valve does not go to the commanded
state, then set the failure code of this equipment
phase = 102.*)
If Drain_Valve.FaultAlarm Then
    PFL(102);
End_If;

(*If the Drain_Tank equipment phase = running and
its failure code = 102, issue the hold command and
send the equipment phase to the holding state.*)
If Drain_Tank.Running And (Drain_Tank.Failure =
102) Then
    PCMD(Drain_Tank,hold,0);
End_IF;
```

Equipment Phase Clear Failure (PCLF)

Use the PCLF instruction to clear the failure code of an equipment phase.

Operands:



Relay Ladder

Operand:	Type:	Format:	Description:
Phase Name	phase	name of the equipment phase	Equipment phase whose failure code you want to clear

```
PCLF(Phase_Name);
```

Structured Text

The operands are the same as those for the relay ladder PCLF instruction.

Description: The PCLF instruction clears the failure code for an equipment phase.

- You must use a PCLF instruction to clear the failure code of an equipment phase.
- A CLR instruction, MOV instruction, or assignment (:=) *doesn't* change the failure code of an equipment phase.
- Make sure the equipment phase *doesn't* have other owners when you use the PCLF instruction. The PCLF instruction *won't* clear the failure code if RSLogix 5000 software, RSBizWare Batch software, or another program owns the equipment phase.

Arithmetic Status Flags: not affected

Fault Conditions: none

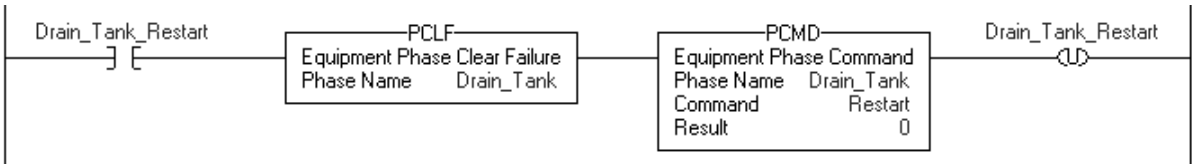
Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction clears the value of the failure code for the specified equipment phase.	The instruction clears the value of the failure code for the specified equipment phase.
postscan	The rung-condition-out is set to false.	No action taken.

Example:

Relay Ladder

If *Drain_Tank_Restart* = 1 (restart the *Drain_Tank* equipment phase) then
Clear the failure code of the *Drain_Tank* equipment phase
Change the state of the *Drain_Tank* equipment phase to restarting via the restart command.
Drain_Tank_Restart = 0.



Structured Text

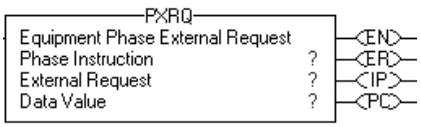
```
(*If Drain_Tank_Restart = on, then:
    Clear the failure code for the Drain_Tank
    equipment phase.
    Restart the Drain_Tank equipment phase.
    Turn off Drain_Tank_Restart.*)

If Drain_Tank_Restart Then
    PCLF(Drain_Tank);
    PCMD(Drain_Tank,Restart,0);
    Drain_Tank_Restart := 0;
End_If;
```

Equipment Phase External Request (PXRQ)

Use the PXRQ instruction to initiate communication with RSBizWare Batch software.

Operands:

Relay Ladder



Operand:	Type:	Format:	Description:
Phase Instruction	PHASE_INSTRUCTION	tag	tag that controls the operation
External Request	request	name	type of request For available requests, see page A-26.
Data Value	DINT	array tag	parameters of the request For array size and data values, see page A-26.

Structured Text

```
PXRQ(Phase_Instruction,
External_Request,
Data_Value);
```

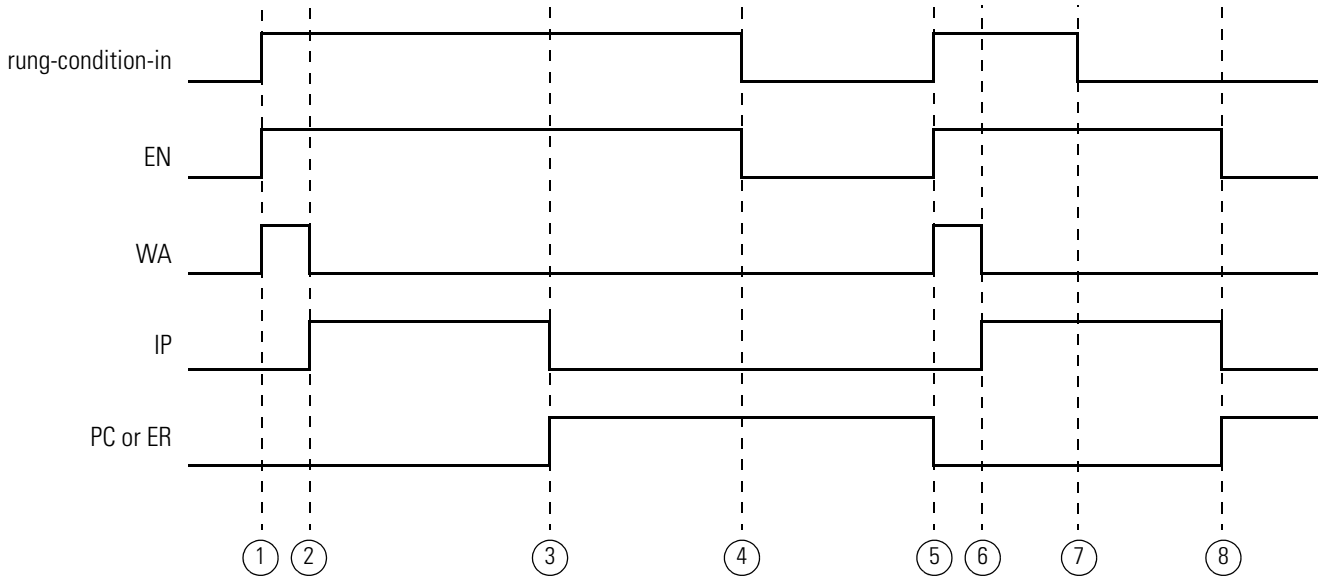
The operands are the same as those for the relay ladder PXRQ instruction.

PHASE_INSTRUCTION Data Type

If you want to:	Then check or set this member:	Data type:	Notes:
determine if a false-to-true transition caused the instruction to execute	EN	BOOL	See Figure A.2 on page A-25.
determine if the request failed	ER	BOOL	See Figure A.2 on page A-25. To diagnose the error, see the ERR and EXERR values.
determine if the RSBizWare Batch software has completed its processing of the request	PC	BOOL	See Figure A.2 on page A-25.
determine if the RSBizWare Batch software is processing the request	IP	BOOL	See Figure A.2 on page A-25.
determine if the instruction sent the request but RSBizWare Batch software has not yet acknowledged it	WA	BOOL	See Figure A.2 on page A-25. WA also = 0 if: • connection times out • network error occurs • ABORT = 1
cancel the request	ABORT	BOOL	To abort (cancel) the request, set the ABORT bit = 1. When the controller aborts the instruction: • ER = 1 • ERR shows the result of the abort

If you want to:	Then check or set this member:	Data type:	Notes:
• diagnose the cause of an error • write logic to respond to specific errors	ERR	INT	If ER = 1, the error code gives diagnostic information. To interpret the error code, see "PXRQ Error Codes" on page A-31.
	EXERR	INT	If ER = 1, the extended error code gives additional diagnostic information for some errors. To interpret the extended error code, see "PXRQ Error Codes" on page A-31.
use one member for the various status bits of the tag	Status	DINT	**For this member:** / **Use this bit:** EN: 31 ER: 28 PC: 27 IP: 26 WA: 25 ABORT: 24

Figure A.2 Timing Diagram of a PXRQ Instruction



Description:		Description:	
1	The rung-condition-in goes true. The instruction sends the request to RSBizWare Batch software.	5	The rung-condition-in goes true. The instruction sends the request to RSBizWare Batch software.
2	RSBizWare Batch software starts processing the request.	6	RSBizWare Batch software starts processing the request.
3	Either of the following occur: • RSBizWare Batch software completes its processing of the request. (PC = 1). • An error occurs. (ER = 1).	7	The rung-condition-in goes false. • EN remains = 1. • RSBizWare Batch software continues processing the request. (IP = 1).
4	The rung-condition-in goes false.	8	Either of the following occur: • RSBizWare Batch software completes its processing of the request. (PC = 1). • An error occurs. (ER = 1). Since the rung-condition-in is false, EN = 0.

Description: The PXRQ instruction sends a request to RSBizWare Batch software.

Guidelines for Using the PXRQ Instruction

Guideline:	Details:
❑ **Make sure to use an array for the Data Values operand.**	The Data Values operand requires a DINT array, even if the array contains only 1 element (i.e., the data type is DINT[1]).
❑ **In relay ladder, condition the instruction to execute on a transition.**	This is a transitional instruction. Each time you want to execute the instruction, toggle the rung-condition-in from false to true.
❑ **In structured text, use a construct to condition the execution of the instruction.**	When you program a PXRQ instruction in structured text, consider the following: • In structured text, instructions execute *each time* they are scanned. • The PXRQ instruction updates its status bits *only* when it is scanned. To keep the instruction from repeatedly executing but ensure that the status bits update, enclose the instruction in a construct that: • initiates the execution of the instruction *only* on a transition (change in conditions) • remains true until either PC = 1 or ER = 1

Configure the PXRQ Instruction

For more information about PXRQ requests, see *RSBizWare Batch PhaseManager User's Guide*, publication BATCHXUMA008-EN-P.

If you want to:	Then configure the PXRQ instruction as follows:		
	External Request:	**Data Value Array Element:**	**Value:**
download all input parameters	Download Input Parameters	DINT[0]	0
download a single input parameter	Download Input Parameters	DINT[0]	parameter ID
download a range of input parameter	Download Input Parameters	DINT[0]	parameter ID of the first parameter
		DINT[1]	number of values
download the input parameters configured for automatic download on start or transfer of control	Download Input Parameters Subset	DINT[0]	start = 1 transfer of control = 2
download all output parameter limits	Download Output Parameter Limits	DINT[0]	0
download a single output parameter limit	Download Output Parameter Limits	DINT[0]	parameter ID
upload all reports	Upload Output Parameters	DINT[0]	0
upload a single report	Upload Output Parameters	DINT[0]	report ID
upload a range of reports	Upload Output Parameters	DINT[0]	report ID of the first report
		DINT[1]	number of values

If you want to:	**Then configure the PXRQ instruction as follows:**		
	External Request:	**Data Value Array Element:**	**Value:**
upload the output parameters configured for automatic upload on terminal state or transfer of control	Upload Output Parameters Subset	DINT[0]	terminal = 1 transfer of control = 2
send a message to an operator	Send Message to Operator	DINT[0]	message ID
clear a message from an operator	Clear Message to Operator	DINT[0]	0
acquire a resource	Acquire Resources	DINT[0]	equipment ID
acquire multiple resources	Acquire Resources	DINT[0]	equipment ID
		DINT[1]	equipment ID
		…	…
release a single resource	Release Resources	DINT[0]	equipment ID
release multiple resources	Release Resources	DINT[0]	equipment ID
		DINT[1]	equipment ID
		…	…
release all resources	Release Resources	DINT[0]	0
send a message (and optional data) to another phase	Send Message to Linked Phase	DINT[0]	message ID
		DINT[1]	number of receivers
		DINT[2]	value 1
		DINT[3]	value 2
		…	…
send a message (and optional data) to another phase and wait for the phase to receive the message	Send Message to Linked Phase and Wait	DINT[0]	message ID
		DINT[1]	number of receivers
		DINT[2]	value 1
		DINT[3]	value 2
		…	…
wait to receive a message from another phase	Receive Message From Linked Phase	DINT[0]	message ID
		DINT[1]	value 1
		DINT[2]	value 2
		…	…
cancel a message to another phase	Cancel Message to Linked Phase	DINT[0]	message ID
cancel all messages to another phase	Cancel Message to Linked Phase	DINT[0]	0
download customer's batch ID	Download Batch Data	DINT[0]	1
		DINT[1]	parameter ID in which to store the value
download unique batch ID	Download Batch Data	DINT[0]	2
		DINT[1]	parameter ID in which to store the value

If you want to:	**Then configure the PXRQ instruction as follows:**		
	External Request:	**Data Value Array Element:**	**Value:**
download phase ID	Download Batch Data	DINT[0]	3
		DINT[1]	parameter ID in which to store the value
download recipe control verses manual phase control	Download Batch Data	DINT[0]	4
		DINT[1]	parameter ID in which to store the value In the result value: 0 = recipe control 1 = manual phase control
download current mode of the phase	Download Batch Data	DINT[0]	5
		DINT[1]	parameter ID in which to store the value In the result value: 0 = P - Auto mode 1 = O - Auto mode
download the low limit of an input parameter	Download Batch Data	DINT[0]	6 The input parameter tag stores the low limit.
download the high limit of an input parameter	Download Batch Data	DINT[0]	7 The input parameter tag stores the high limit.
download data about the container currently in use.	Download Material Track Data Container In Use	DINT[0]	1
		DINT[1]	attribute ID
		DINT[2]	parameter ID in which to store the value
download data about the current material inside the container currently in use.	Download Material Track Data Container In Use	DINT[0]	2
		DINT[1]	attribute ID
		DINT[2]	parameter ID in which to store the value
download data about the current lot inside the container currently in use.	Download Material Track Data Container In Use	DINT[0]	3
		DINT[1]	attribute ID
		DINT[2]	parameter ID in which to store the value
upload data about the container currently in use	Upload Material Track Data Container In Use	DINT[0]	1
		DINT[1]	attribute ID
		DINT[2]	parameter ID that has the value
upload data about the current material inside the container currently in use.	Upload Material Track Data Container In Use	DINT[0]	2
		DINT[1]	attribute ID
		DINT[2]	parameter ID that has the value
upload data about the current lot inside the container currently in use.	Upload Material Track Data Container In Use	DINT[0]	3
		DINT[1]	attribute ID
		DINT[2]	parameter ID that has the value

If you want to:	**Then configure the PXRQ instruction as follows:**		
	External Request:	**Data Value Array Element:**	**Value:**
download the current binding's container priority	Download Container Binding Priority	DINT[0]	parameter ID in which to store the value
upload a new container priority for the current binding	Upload Container Binding Priority	DINT[0]	parameter ID that has the value
download information regarding the availability of sufficient material	Download Sufficient Material	DINT[0]	parameter ID in which to store the value In the result value: 0 = insufficient material 1 = sufficient material
generate a signature	Generate E Signature	DINT[0]	ID of the signature template
		DINT[1]	define if the signature is cancelable: no = 0 yes = nonzero
download material attribute	Download Material Track Database Data	DINT[0]	0
		DINT[1]	parameter ID in which to store the value
		DINT[2]	controller ID
		DINT[3]	attribute ID
download lot attribute	Download Material Track Database Data	DINT[0]	1
		DINT[1]	parameter ID in which to store the value
		DINT[2]	controller ID
		DINT[3]	attribute ID
download container attribute	Download Material Track Database Data	DINT[0]	3
		DINT[1]	parameter ID in which to store the value
		DINT[2]	controller ID
		DINT[3]	attribute ID
download container priority assignment	Download Material Track Database Data	DINT[0]	4
		DINT[1]	parameter ID in which to store the value
		DINT[2]	material ID
		DINT[3]	container ID
		DINT[4]	feed type: 1 = add to container 2 = distribute from container
upload material attribute	Upload Material Track Database Data	DINT[0]	5
		DINT[1]	report ID that has the value
		DINT[2]	controller ID
		DINT[3]	attribute ID

If you want to:	**Then configure the PXRQ instruction as follows:**		
	External Request:	**Data Value Array Element:**	**Value:**
upload lot attribute	Upload Material Track Database Data	DINT[0]	6
		DINT[1]	report ID that has the value
		DINT[2]	controller ID
		DINT[3]	attribute ID
upload container attribute	Upload Material Track Database Data	DINT[0]	8
		DINT[1]	report ID that has the value
		DINT[2]	controller ID
		DINT[3]	attribute ID
upload container priority assignment	Upload Material Track Database Data	DINT[0]	9
		DINT[1]	report ID that has the value
		DINT[2]	material ID
		DINT[3]	container ID
		DINT[4]	feed type: add to container = 1 distribute from container = 2

PXRQ Error Codes

ERR (hex)	EXERR (hex)	Description	Recommended Action
00	0000	The PXRQ instruction was aborted before it sent the request to RSBizWare Batch software.	None
01	0000	The PXRQ instruction was aborted after it sent the request to RSBizWare Batch software.	None
02	0000	2 or more PXRQ instructions executed at the same time using the same request type	Limit execution to 1 PXRQ instruction at a time.
03	0110	Communication error. The request was not delivered because there is no subscriber subscribed to the phase.	Check that RSBizWare Batch software is connected and running.
	0210	Communication error. The request was not delivered because there is no connection to the Notify object.	Check that RSBizWare Batch software is connected and running.
	0410	Communication error. Delivery failed.	Check the connection and communication path to RSBizWare Batch software.
	1010	Communication error. The request was not delivered because RSBizWare Batch software does not subscribe to receive the external request.	Check that RSBizWare Batch software is connected and running.
	1020	RSBizWare Batch software isn't attached to the phase.	Check that RSBizWare Batch software is attached to the phase.
04	0002	The RSBizWare Batch software encountered an error while processing the request.	Check the connection and communication path to RSBizWare Batch software.
	0003	The PXRQ instruction contains an invalid value.	Check the connection and communication path to RSBizWare Batch software.
	0004	RSBizWare Batch software is not in the proper state to process the request.	Check the connection and communication path to RSBizWare Batch software.
	0005	2 or more PXRQ instructions executed at the same time using different request types	Limit execution to 1 PXRQ instruction at a time.
	0006	Error storing to parameter tags at end of request processing.	Check the connection and communication path to RSBizWare Batch software.
05	0000	RSBizWare Batch software received the request but passed back an invalid cookie.	Check the connection and communication path to RSBizWare Batch software.
06	0000	PXRQ instruction sent an invalid parameter to RSBizWare Batch software.	Check the connection and communication path to RSBizWare Batch software.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• When the rung-condition-in goes from false to true, the instruction executes one time. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction sends the specified request to RSBizWare Batch software.	The instruction sends the specified request to RSBizWare Batch software.
postscan	The rung-condition-out is set to false.	No action taken.

Example:

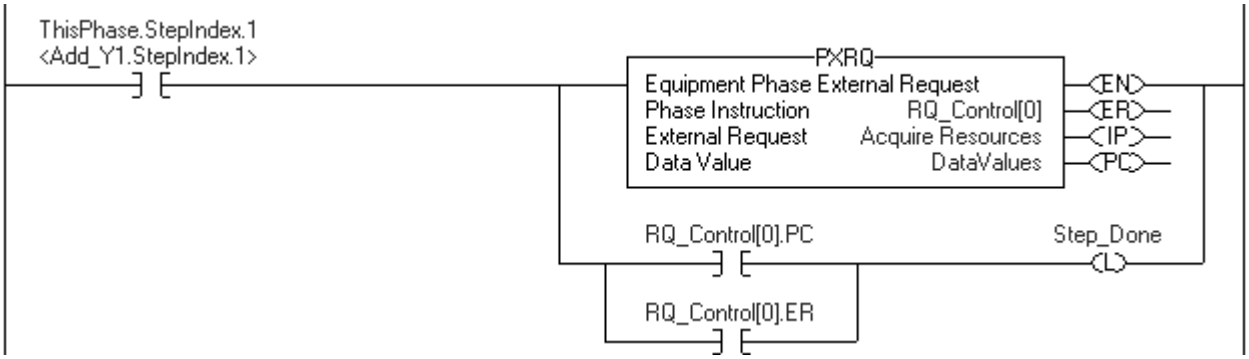
Relay Ladder

If *ThisPhase.StepIndex.1* = 1 (The routine is at step 1.), then

Send the Acquire Resources request to RSBizWare Batch software. The *DataValues* tag is a DINT array that contains the equipment IDs of the resources.

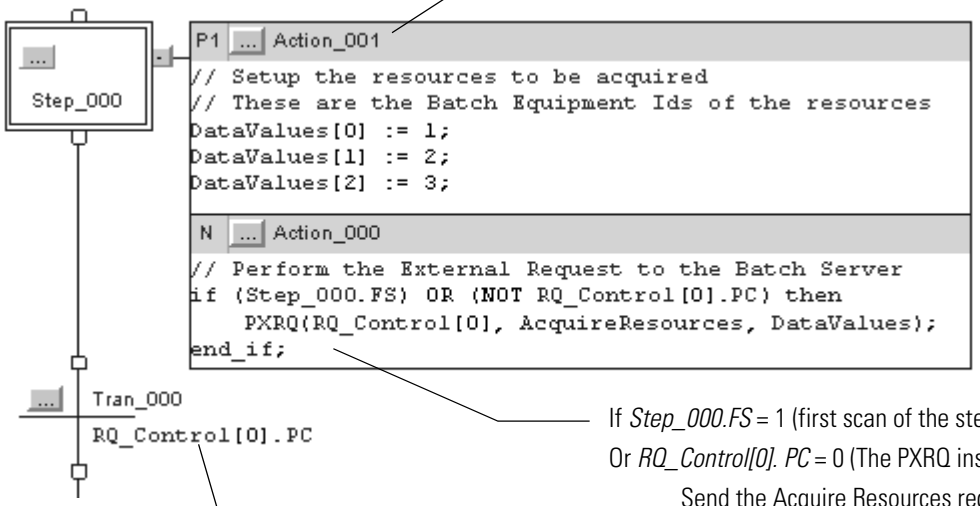
When *RQ_Control[0].PC* = 1 or *RQ_Control[0].ER* = 1 (The request is complete or it failed.), then

Done = 1. (This signals the sequence to go to the next step.)



Structured Text

This action executes one time to initialize the data values of the PXRQ instruction to the equipment IDs of the resources it will acquire.



If *Step_000.FS* = 1 (first scan of the step)

Or *RQ_Control[0]. PC* = 0 (The PXRQ instruction is in process.), then

Send the Acquire Resources request to RSBizWare Batch software. The *DataValues* tag is a DINT array that contains the equipment IDs of the resources.

If *RQ_Control[0]. PC* = 1 (The request is complete.), then

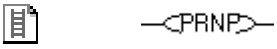
Go to the next step.

Equipment Phase New Parameters (PRNP)

Use the PRNP instruction to clear the NewInputParameters bit of an equipment phase.

Operands:

Relay Ladder



none

Structured Text

```
PRNP( );
```

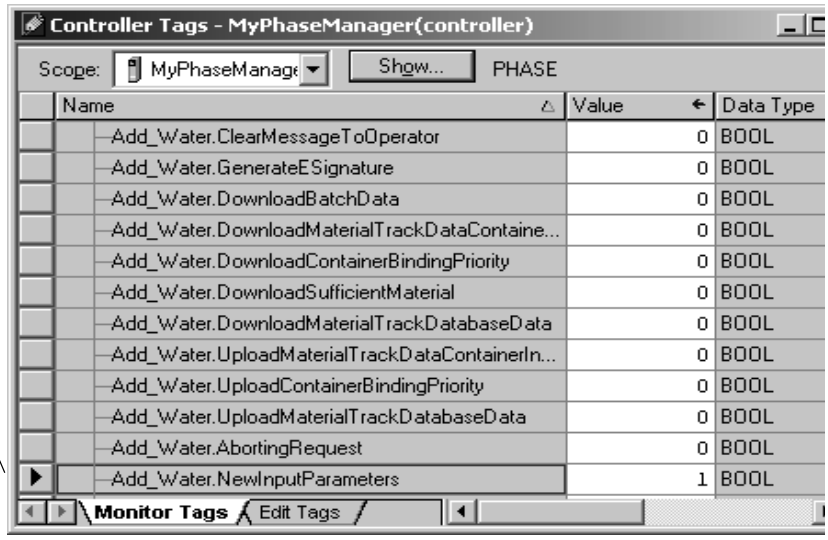
none

You must enter the parentheses ( ) after the instruction mnemonic, even though there are no operands.

Description: The PRNP instruction clears the NewInputParameters bit of the equipment phase.

When RSBizWare Batch software has new parameters for an equipment phase, it sets the NewInputParameters bit for the phase.

After you download the parameters, use the PRNP instruction to clear the bit.



Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction clears the NewInputParameters bit of the equipment phase.	The instruction clears the NewInputParameters bit of the equipment phase.
postscan	The rung-condition-out is set to false.	No action taken.

Example:

Relay Ladder

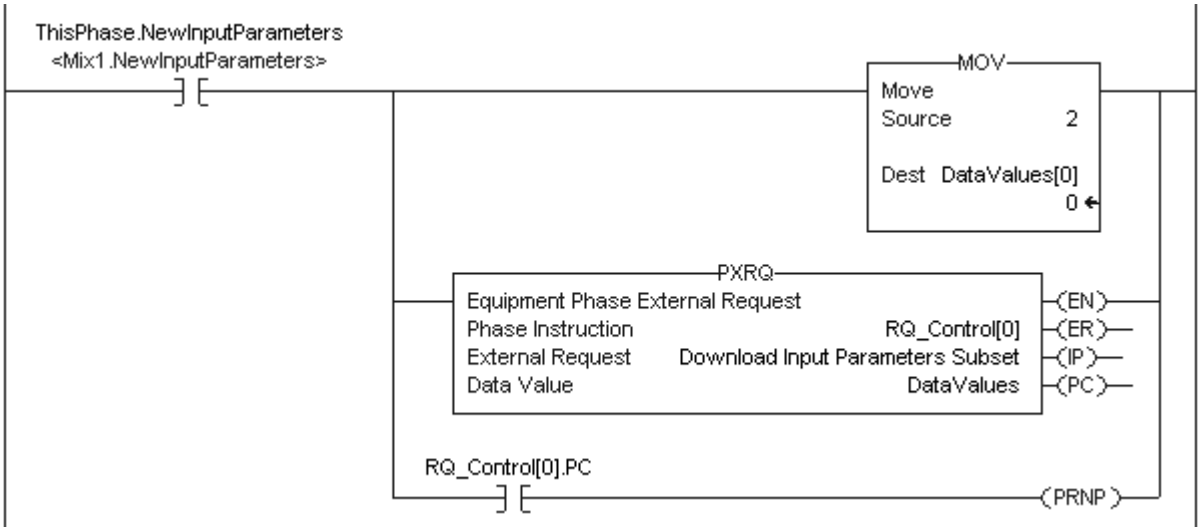
If *ThisPhase.NewInputParameters* = 1 (RSBizWare Batch software has new input parameters for the equipment phase), then

DataValues[0] = 2. This sets the PXRQ instruction for transfer of control.

Send the Download Input Parameters Subset request to RSBizWare Batch software. Since *DataValues[0]* = 2, the instruction is set for transfer of control.

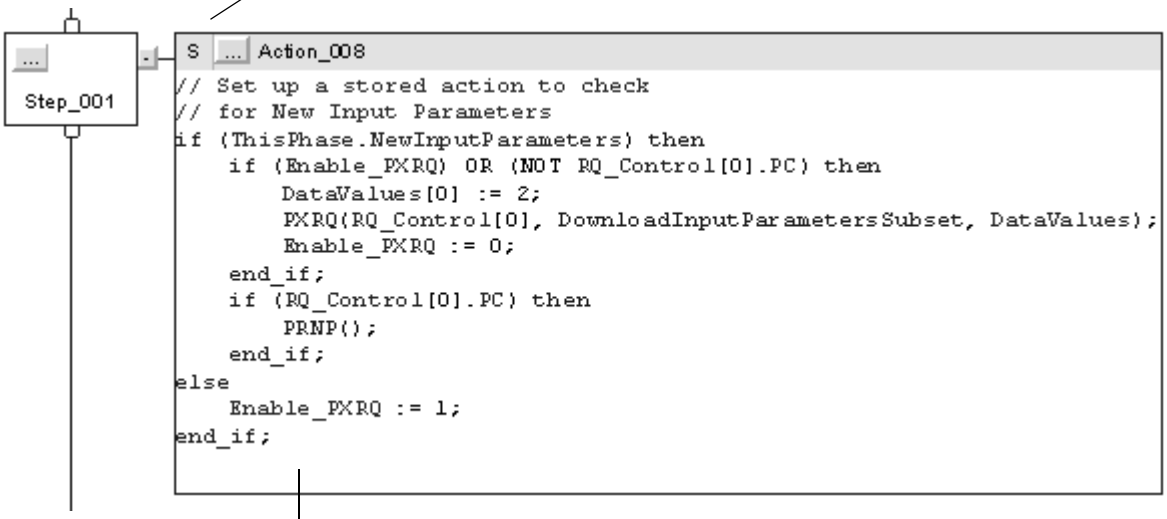
When *RQ_Control[0].PC* = 1 (The PXRQ instruction is done.), then

ThisPhase.NewInputParameters = 0 via the PRNP instruction.



Structured Text

This is a stored action. It continues to execute even after the SFC leaves Step_001.




```
// Set up a stored action to check
// for New Input Parameters
if (ThisPhase.NewInputParameters) then
    if (Enable_PXRQ) OR (NOT RQ_Control[0].PC) then
        DataValues[0] := 2;
        PXRQ(RQ_Control[0], DownloadInputParametersSubset, DataValues);
        Enable_PXRQ := 0;
    end_if;
    if (RQ_Control[0].PC) then
        PRNP();
    end_if;
else
    Enable_PXRQ := 1;
end_if;
```

If *ThisPhase.NewInputParameters* = 1 (RSBizWare Batch software has new input parameters for the equipment phase), then

If *Enable_PXRQ* = 1 (Let the PXRQ instruction execute.)

Or *RQ_Control[0]. PC* = 0 (The PXRQ instruction is in process.), then

DataValues[0] = 2. This sets the PXRQ instruction for transfer of control.

Send the Download Input Parameters Subset request to RSBizWare Batch software. Since *DataValues[0]* = 2, the instruction is set for transfer of control.

Enable_PXRQ = 0 (*Do not* let the PXRQ instruction restart after the request completes.)

If *RQ_Control[0]. PC* = 1(The request is complete.), then

ThisPhase.NewInputParameters = 0 via the PRNP instruction.

Otherwise

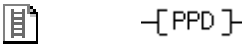
Enable_PXRQ = 1 (Let the PXRQ instruction execute the next time new input parameters are available.

Equipment Phase Paused (PPD)

Use the PPD instruction to set up breakpoints within the logic of an equipment phase.

Operands:

Relay Ladder



none

Structured Text

```
PPD( );
```

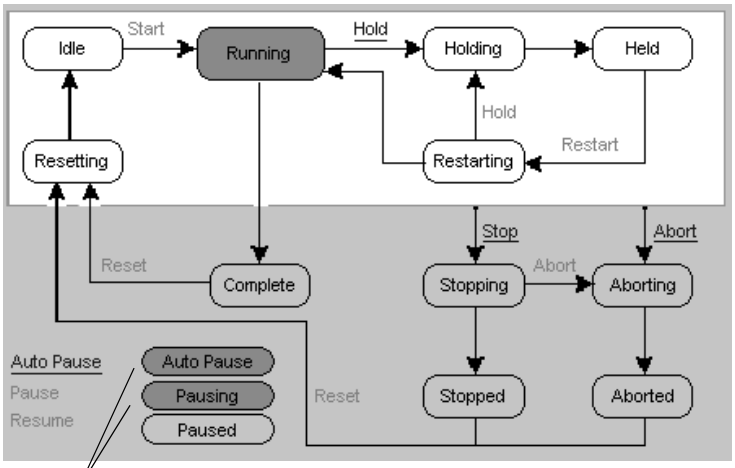
none

You must enter the parentheses ( ) after the instruction mnemonic, even though there are no operands.

Description: The PPD instruction lets you stop execution at a specific step (breakpoint) to test and troubleshoot your logic. When an equipment phase is in the pausing substate and the controller executes a PPD instruction, the controller:

- Sets the Paused bit of the PHASE tag = 1
- Makes the rest of the rung = false (RLL)

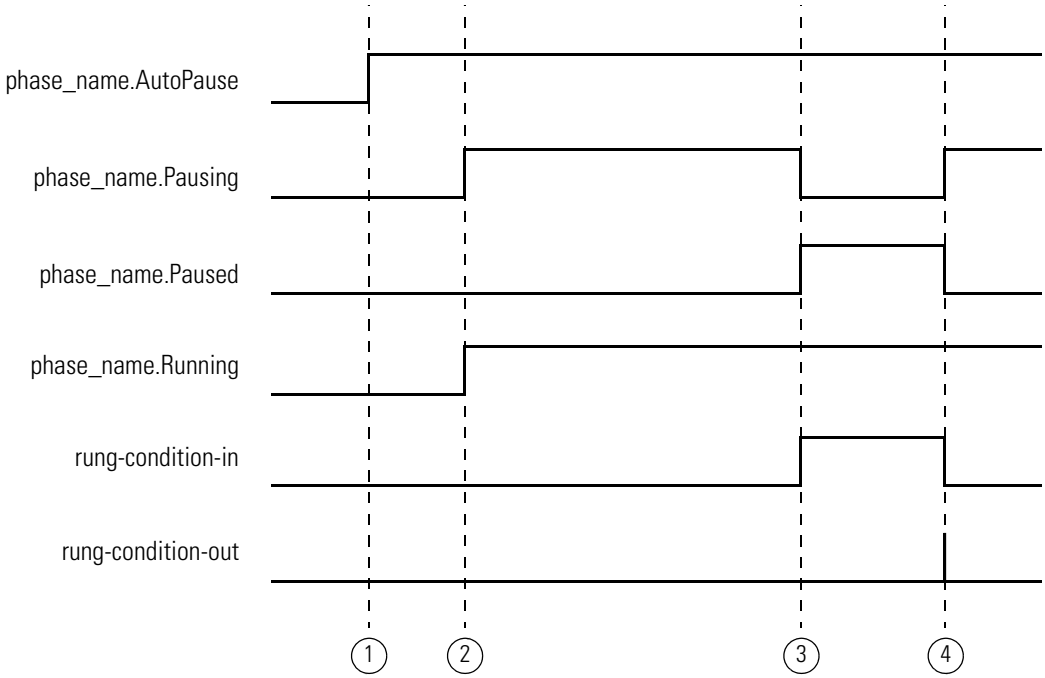
Once you place PPD instructions in your equipment phase, you can either use them or ignore them.



The auto pause and pausing substates let you control whether or not the equipment phase uses your breakpoints.

The auto pause, pause, and resume commands let you change the substate and step through your breakpoints.

The following timing diagram shows how the PPD instruction effects the substate bits of a PHASE tag.



Description:	
1	The equipment phase gets the auto pause command.
2	The equipment phase gets the start command. Once the equipment phase starts, it goes to the pausing substate.
3	Input conditions for the PPD instruction go true. The equipment phase changes from the pausing substate to the paused substate
4	The equipment phase gets the resume command. The logic goes to next step, so input conditions go false and the outputs no longer execute. Because auto pause is on, the equipment phase automatically goes back to the pausing substate.

Guidelines for Using Breakpoints

Guideline:	Details:
❑ **Organize your logic as a series of steps.**	PPD instructions (breakpoints) are easiest to use if your logic moves through defined steps, such as a state machine or SFC. • A breakpoint *only* signals that specific conditions are met. It *does not* stop the execution of the equipment phase. • To have your logic actually break (pause) at a breakpoint, organize your logic so that it stays at the step at which the breakpoint occurred until you give the resume command. See the examples on page A-41.
❑ ***Do not* use a PPD instruction as a temporary end of the routine.**	Even when an equipment phase is paused, it continues to execute all its logic. • When a PPD instruction executes, it only sets the Paused bit for the equipment phase. • If you program the PPD instruction in RLL, it disables only the rest of the logic on its rung. It *does not* terminate or suspend the execution of the routine. • Think of the PPD instruction as a condition that you can apply or ignore based on the auto pause and pause commands.
❑ **Limit the execution of a PPD instruction to a single scan.**	In the pausing substate, an equipment phase goes to paused at the *first* PPD instruction whose conditions are true. If the PPD instruction executes over several scans, the equipment phase may continually pause at the same breakpoint. (This is different than a One Shot (ONS) instruction, which executes only on a false-to-true transition.)
❑ **Make sure only 1 PPD instruction at a time is true.**	A PPD instruction *does not* have a control tag to remember whether it executed. • Anytime its conditions are true (and the equipment phase is in the pausing substate), the PPD instruction acts as a breakpoint (sets the equipment phase to paused). • Limiting your logic to one possible breakpoint at a time ensures that you pause at the required breakpoint.
❑ **Choose the correct substate.**	PPD instructions (breakpoints) work *only* when the equipment phase is in the pausing substate:

To pause at:	Give this command:	Notes:
each true breakpoint	auto pause	• When you start the equipment phase, it goes to the pausing substate. • When you give the resume command after a pause, the equipment phase remains in the pausing substate. • To turn off auto pause, give the auto pause command again.
first true breakpoint	pause	• Give the pause command after you start the equipment phase. • To pause at another breakpoint, give the resume command and then the pause command.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:		Structured Text Action:	
prescan	The rung-condition-out is set to false.		No action taken.	
rung-condition-in is false	The rung-condition-out is set to false.		na	
rung-condition-in is true	The instruction executes.		na	
scan of structured text	na		In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.	
instruction execution	**Substate:**	**Action:**	**Substate:**	**Action:**
	pausing	• The substate = paused. • The Paused bit of the PHASE tag = 1. • The rung-condition-out = false.	pausing	• The substate = paused. • The Paused bit of the PHASE tag = 1.
	not pausing	The rung-condition-out = true.	*not* pausing	No action taken.
postscan	The rung-condition-out is set to false.		No action taken.	

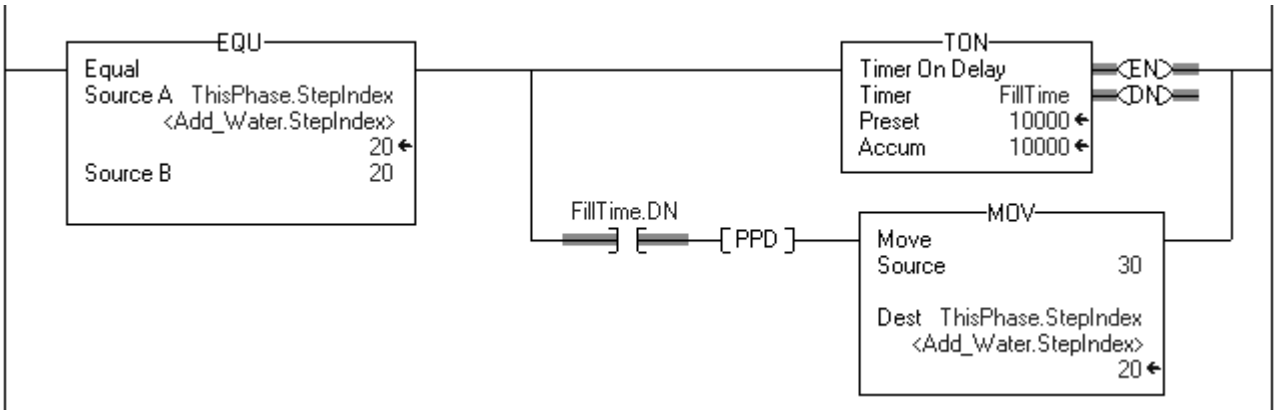
Example:

Relay Ladder

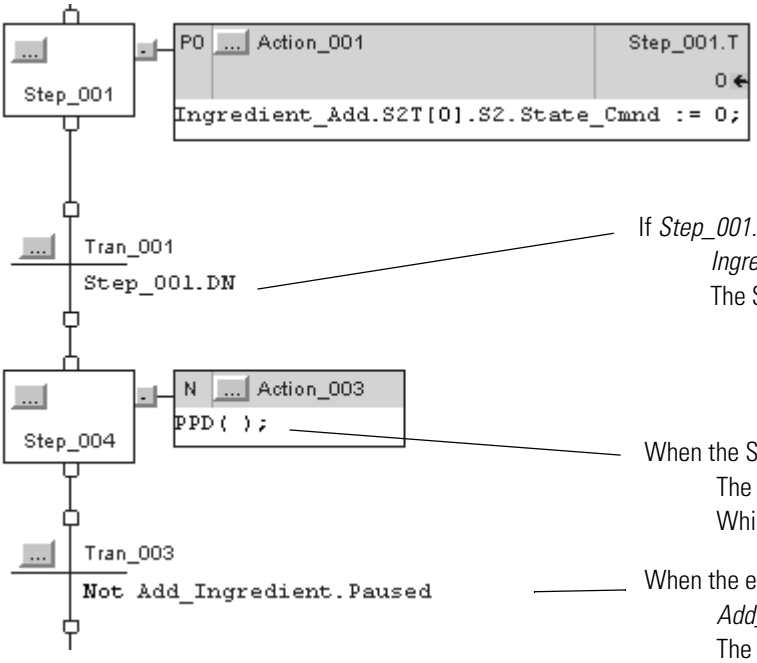
If the equipment phase is in the pausing substate
And *ThisPhase.StepIndex* = 20 (The routine is at step 20.)
And *FillTime.DN* = On

Then the PPD instruction prevents the MOV instruction from moving the routine to step 30 until the equipment phase gets the resume command. (The routine stays at step 20.)

When the equipment phase gets the resume command, the PPD instruction lets the MOV instruction execute, so the routine goes to step 30.



Structured Text



If *Step_001.DN* = 1 (The step ran for the preset time.), then
Ingredient_Add…State_Cmnd = 0 (stop adding ingredient)
The SFC goes to *Step_004*

When the SFC goes to *Step_004*
The PPD instruction sets *Add_Ingredient.Paused* = 1
While *Add_Ingredient.Paused* = 1, the SFC stays at *Step_004.*

When the equipment phase gets the resume command
Add_Ingredient.Paused = 0
The SFC goes to the next step

Attach to Equipment Phase (PATT)

Use the PATT instruction to take ownership of an equipment phase to either:

- prevent another program or RSBizWare Batch software from commanding an equipment phase
- make sure another program or RSBizWare Batch software does *not* already own an equipment phase

Operands:

```
PATT
Attach to Equipment Phase
Phase Name        ?
Result            ?
```

Relay Ladder

Operand:	Type:	Format:	Description:
Phase Name	phase	name of the equipment phase	Equipment phase that you want to own
Result	DINT	immediate tag	To let the instruction return a code for its success/failure, enter a DINT tag in which to store the result code. Otherwise, enter 0.

```
PATT(Phase_Name,
Result);
```

Structured Text

The operands are the same as those for the relay ladder PATT instruction.

Description:

The PATT instruction lets a program take ownership of an equipment phase.

- Ownership is optional. As long as an equipment phase has no owners, any sequencer (program in the controller, RSBizWare Batch software) can command an equipment phase.
- RSBizWare Batch software always takes ownership of an equipment phase.
- Once a sequencer owns an equipment phase, no other sequencer can command the equipment phase.

Guidelines for Using the PATT Instruction

Guideline:	Details:
❑ **Consider ownership if you have multiple sequencers that use a common equipment phase.**	Ownership makes sure that a program can command all the equipment phases it needs and locks out any other sequencers. **If you use:** / **Then:** RSBizWare Batch software to also run sequences within this controller — Before you execute your sequence (process), take ownership of all the equipment phases that the sequence uses. multiple programs to command the same equipment phase — Before you execute your sequence (process), take ownership of all the equipment phases that the sequence uses. none of the above — There is no need to own the equipment phases.
❑ **Remember that RSLogix 5000 software overrides the controller.**	Regardless of whether a program or RSBizWare Batch software owns an equipment phase, you can always use RSLogix 5000 software to override ownership and command the equipment phase to a different state. **This:** / **Overrides this:** RSLogix 5000 software — controller (internal sequencer); RSBizWare Batch software (external sequencer) controller (internal sequencer) — none RSBizWare Batch software (external sequencer) — none
❑ **Use the Result operand to validate ownership.**	Use the Result operand to get a code that shows the success/failure of the PATT instruction. To interpret the result code, see "PATT Result Codes" on page A-44
❑ **Avoid or plan for a result code = 24582.**	On each execution, the PATT instruction tries to take ownership of the equipment phase. Once a program owns an equipment phase, another execution of the PATT instruction produces a result code = 24582. When you use a PATT instruction, either: • Limit its execution to a single scan to avoid the 24582 result code. • Include in your conditions for ownership a result code = 24582. See the example on page A-45.
❑ **When the sequence is done, relinquish ownership.**	To relinquish ownership, use a Detach from Equipment Phase (PDET) instruction. See page A-47.

PATT Result Codes

If you assign a tag to store the result of a PATT instruction, the instruction returns one of the following codes when it executes:

Code (Dec):	Description:
0	The command was successful.
24579	RSLogix 5000 software already owns the equipment phase. • This program now also owns the equipment phase. • Since RSLogix5000 software is higher priority than a program, the program *cannot* command the equipment phase.
24582	The program already owns the equipment phase.
24593	One of the following already owns the equipment phase. • external sequencer (RSBizWare Batch software) • another program in the controller
24594	Equipment phase is inhibited, unscheduled, or in a task that is inhibited.

Arithmetic Status Flags: not affected

Fault Conditions: none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction tries to take ownership of the specified equipment phase.	The instruction tries to take ownership of the specified equipment phase.
postscan	The rung-condition-out is set to false.	No action taken.

Example: Relay Ladder

If *Step.1* = 1 (first step in the sequence) then

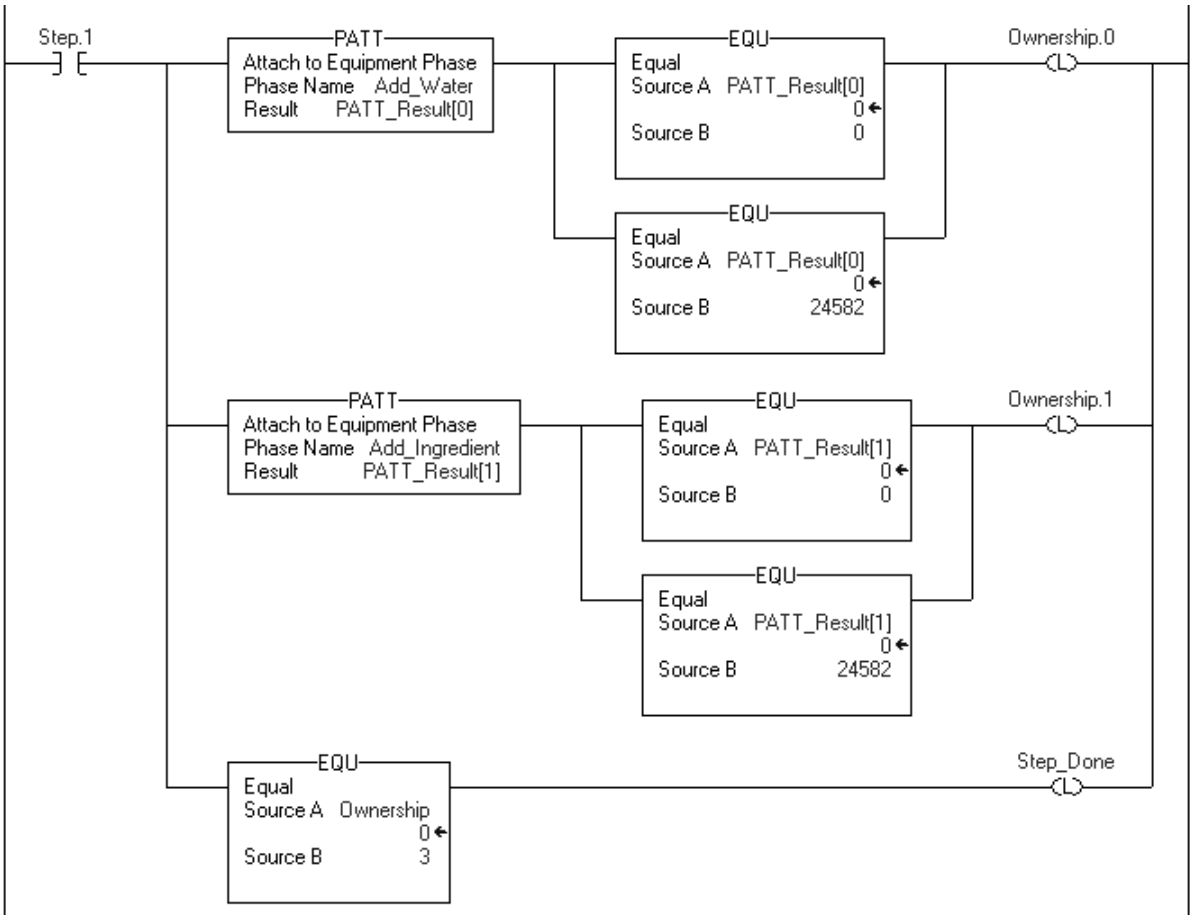
Each PATT instruction tries to take ownership of an equipment phase.

If the Result of a PATT instruction = 0 or 24582 (the program owns the equipment phase), then

A bit within the *Ownership* tag = 1. (In the *Ownership* tag, each equipment phase is assigned a bit.)

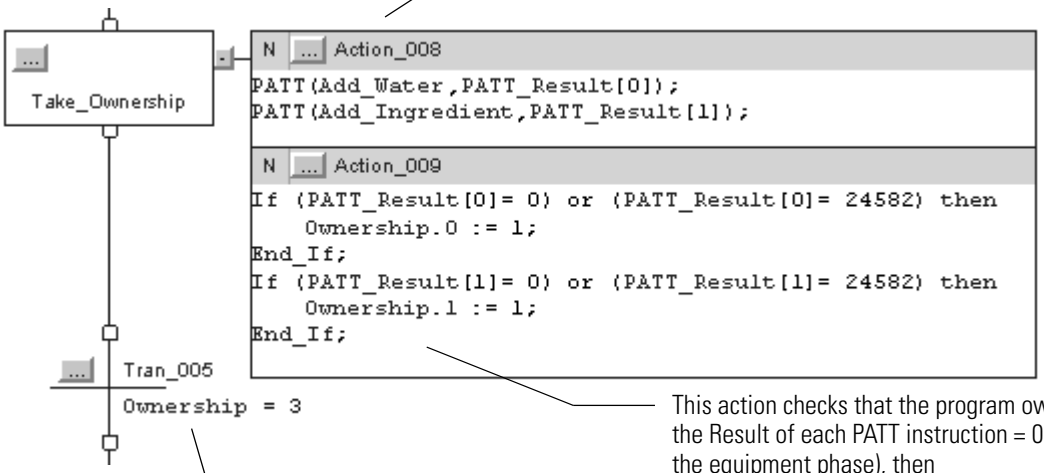
If *Ownership* = 3 (The program owns both equipment phases as shown by bits 0 and 1.), then:

Done = 1. (This signals the sequence to go to the next step.)



Structured Text

At the first step in the sequence, this action tries to take ownership of 2 equipment phases that the sequence uses.



This action checks that the program owns the equipment phases. If the Result of each PATT instruction = 0 or 24582 (the program owns the equipment phase), then

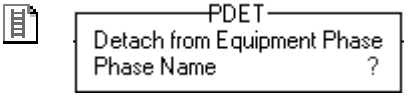
A bit within the *Ownership* tag = 1. (In the *Ownership* tag, each equipment phase is assigned a bit.)

If *Ownership* = 3 (The program owns both equipment phases as shown by bits 0 and 1.), then the SFC goes to the next step.

Detach from Equipment Phase (PDET)

Use the PDET instruction to relinquish ownership of an equipment phase.

Operands:



Relay Ladder

Operand:	Type:	Format:	Description:
Phase Name	phase	name of the equipment phase	Equipment phase that you *no longer* want to own

```
PDET(Phase_Name);
```

Structured Text

The operands are the same as those for the relay ladder PDET instruction.

Description:

After a program executes a PDET instruction, the program *no longer* owns the equipment phase. This frees the equipment phase for ownership by another program or by RSBizWare Batch software. Use the PDET instruction only if the program previously took ownership of an equipment phase via an Attach to Equipment Phase (PATT). instruction.

Arithmetic Status Flags:

not affected

Fault Conditions:

none

Execution:

Condition:	Relay Ladder Action:	Structured Text Action:
prescan	The rung-condition-out is set to false.	No action taken.
rung-condition-in is false	The rung-condition-out is set to false.	na
rung-condition-in is true	• The instruction executes. • The rung-condition-out is set to true.	na
scan of structured text	na	In structured text, instructions execute each time they are scanned. To limit the scan of an instruction, use a qualifier of an SFC action and/or a structured text construct.
instruction execution	The instruction relinquishes ownership of the specified equipment phase.	The instruction relinquishes ownership of the specified equipment phase.
postscan	The rung-condition-out is set to false.	No action taken.

Example:

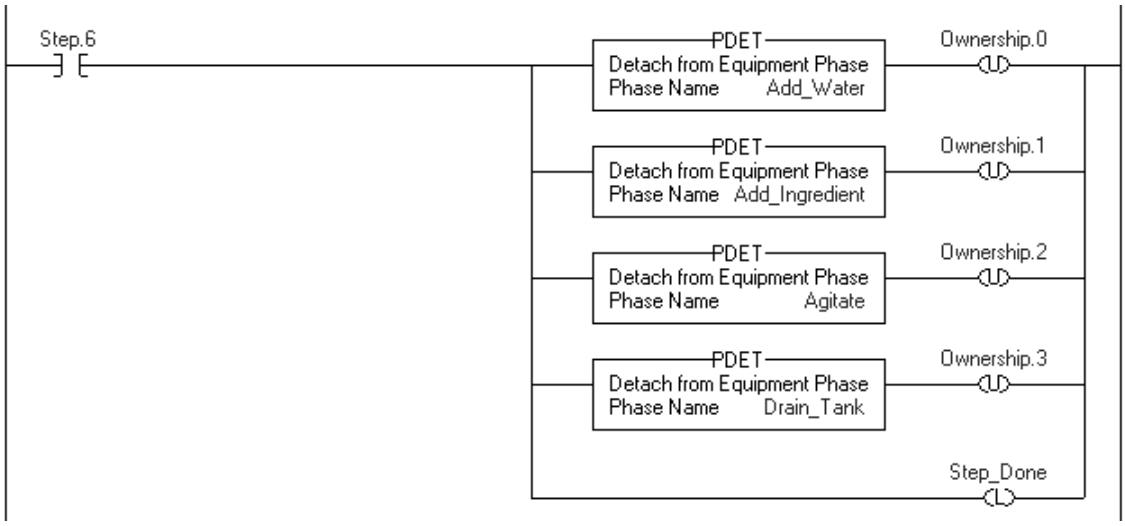
Relay Ladder

If *Step.6* = 1 (step 6 in the sequence) then

Each PDET instruction relinquishes ownership of the equipment phases that the sequence owned.

Each *Ownership* bit = 0. (In the *Ownership* tag, each equipment phase is assigned a bit.)

Done = 1. (This signals the sequence to go to the next step.)

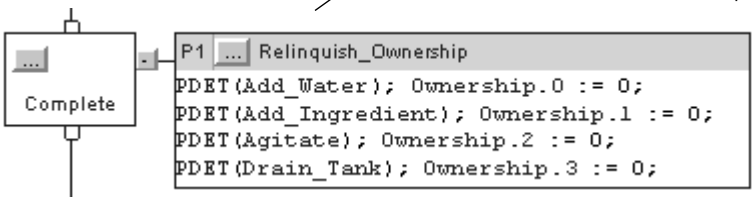


Structured Text

When the sequence is done, this action:

- relinquishes ownership of the equipment phases
- clears the ownership flags (bits that the SFC set when it took ownership of the equipment phases)

The P1 qualifier limits this to the first scan of the step.



PHASE Data Type

Using the PHASE Data Type

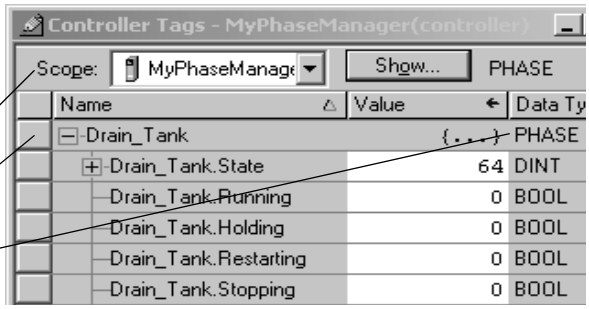
The PHASE data type gives you status information about an equipment phase.

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

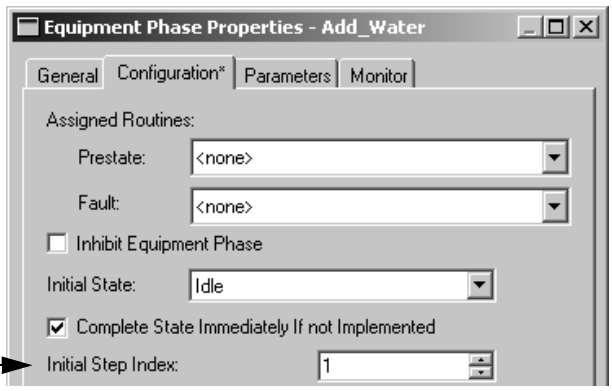


Set and Clear Equipment Phase Tag Values

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

Member	Control Method
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: • To advance to the next step, write logic to increment the StepIndex value (e.g., MOV, MUL, OTL, :=)

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



Member	Control Method	
Failure	**To:**	**Use this instruction:**
	set the Failure value	Equipment Phase Failure (PFL)
	clear the Failure value	Equipment Phase Clear Failure (PCLF)
NewInputParameters	To clear the NewInputParameters bit, use an Equipment Phase New Parameters (PRNP) instruction.	
Producing	Use bit-level instructions or an assignment to set or clear this bit (e.g., OTE, :=).	
Standby	Use bit-level instructions or an assignment to set or clear this bit (e.g., OTE, :=).	

PHASE Data Type

If you want to:	Then check this member:	Data type:	Notes:	
use one member to monitor the state of an equipment phase	State	DINT	Read–only	
			For this state:	**Use this bit:**
			Running	0
			Holding	1
			Restarting	2
			Stopping	3
			Aborting	4
			Resetting	5
			Idle	6
			Held	7
			Complete	8
			Stopped	9
			Aborted	10
see if the equipment phase is in the running state	Running	BOOL	Read–only	
see if the equipment phase is in the holding state	Holding	BOOL	Read–only	
see if the equipment phase is in the restarting state	Restarting	BOOL	Read–only	
see if the equipment phase is in the stopping state	Stopping	BOOL	Read–only	
see if the equipment phase is in the aborting state	Aborting	BOOL	Read–only	
see if the equipment phase is in the resetting state	Resetting	BOOL	Read–only	
see if the equipment phase is in the idle state	Idle	BOOL	Read–only	
see if the equipment phase is in the held state	Held	BOOL	Read–only	

If you want to:	Then check this member:	Data type:	Notes:
see if the equipment phase is in the complete state	Complete	BOOL	Read–only
see if the equipment phase is in the stopped state	Stopped	BOOL	Read–only
see if the equipment phase is in the aborted state	Aborted	BOOL	Read–only
use one member to monitor the substate of an equipment phase	Substate	DINT	Read–only **For this substate:** / **Use this bit:** Pausing / 0 Paused / 1 AutoPause / 2
see if the equipment phase is in the pausing substate	Pausing	BOOL	Read–only
see if the equipment phase is in the paused substate	Paused	BOOL	Read–only
see if the equipment phase is in the auto pause substate	AutoPause	BOOL	Read–only
use an integer value or the bits of an integer to sequence through a series of steps	StepIndex	DINT	• To initialize the StepIndex value, use the configuration properties for the equipment phase. • To advance to the next step, use logic such as an MOV, MUL, or := to increment the StepIndex value.
flag a specific exception for an equipment phase (fault, failure, off-normal condition, etc.)	Failure	DINT	**To:** / **Use:** set a Failure value / PFL instruction clear the Failure value / PCLF instruction
find the unit ID of an equipment phase	UnitID	DINT	RSBizWare Batch software sets this value.
monitor the ownership of an equipment phase	Owner	DINT	Read–only
see if an external request is in process via a PXRQ instruction	PendingRequest	DINT	• Read–only • Each bit = the state of a specific request, starting with bit 0. The bits are in the order shown by the request-specific members below.
see if a Download Input Parameters request is in process via a PXRQ instruction	DownloadInputParameters	BOOL	Read–only
see if a Download Input Parameters Subset request is in process via a PXRQ instruction	DownloadInputParameters Subset	BOOL	Read–only
see if a Upload Output Parameters request is in process via a PXRQ instruction	UploadOutputParameters	BOOL	Read–only
see if a Upload Output Parameters Subset request is in process via a PXRQ instruction	UploadOutputParameters Subset	BOOL	Read–only
see if a Download Output Parameter Limits request is in process via a PXRQ instruction	DownloadOutput ParameterLimits	BOOL	Read–only

If you want to:	Then check this member:	Data type:	Notes:
see if an Acquire Resources request is in process via a PXRQ instruction	AcquireResources	BOOL	Read–only
see if an Release Resources request is in process via a PXRQ instruction	ReleaseResources	BOOL	Read–only
see if a Send Message To Linked Phase request is in process via a PXRQ instruction	SendMessageToLinked Phase	BOOL	Read–only
see if a Send Message To Linked Phase And Wait request is in process via a PXRQ instruction	SendMessageToLinked PhaseAndWait	BOOL	Read–only
see if a Receive Message From Linked Phase request is in process via a PXRQ instruction	ReceiveMessageFrom LinkedPhase	BOOL	Read–only
see if a Cancel Message To Linked Phase request is in process via a PXRQ instruction	CancelMessageToLinked Phase	BOOL	Read–only
see if a Send Message To Operator request is in process via a PXRQ instruction	SendMessageToOperator	BOOL	Read–only
see if a Clear Message To Operator request is in process via a PXRQ instruction	ClearMessageToOperator	BOOL	Read–only
see if a Generate E Signature request is in process via a PXRQ instruction	GenerateESignature	BOOL	Read–only
see if a Download Batch Data request is in process via a PXRQ instruction	DownloadBatchData	BOOL	Read–only
see if a Download Material Track Data Container In Use request is in process via a PXRQ instruction	DownloadMaterialTrack DataContainerInUse	BOOL	Read–only
see if a Download Container Binding Priority request is in process via a PXRQ instruction	DownloadContainer BindingPriority	BOOL	Read–only
see if a Download Sufficient Material request is in process via a PXRQ instruction	DownloadSufficient Material	BOOL	Read–only
see if a Download Material Track Database Data request is in process via a PXRQ instruction	DownloadMaterialTrack DatabaseData	BOOL	Read–only
see if a Upload Material Track Data Container In Use request is in process via a PXRQ instruction	UploadMaterialTrackData ContainerInUse	BOOL	Read–only
see if a Upload Container Binding Priority request is in process via a PXRQ instruction	UploadContainderBinding Priority	BOOL	Read–only
see if a Upload Material Track Database Data request is in process via a PXRQ instruction	UploadMaterialTrackData baseData	BOOL	Read–only
see if your logic has aborted a PXRQ instruction	AbortingRequest	BOOL	Read–only

If you want to:	Then check this member:	Data type:	Notes:
see if RSBizWare Batch software has new parameters for an equipment phase	NewInputParameters	BOOL	• Read–only • RSBizWare Batch software sets this bit when it has new parameters for an equipment phase. • To clear the NewInputParameters bit, use a PRNP instruction.
initiate a producing state	Producing	BOOL	Logix5000 equipment phases don't have a producing state. To create a producing state, use the Producing bit.
initiate a standby state	Standby	BOOL	Logix5000 equipment phases don't have a standby state. To create a standby state, use the Standby bit.

Notes:

Configure an Equipment Phase

Purpose

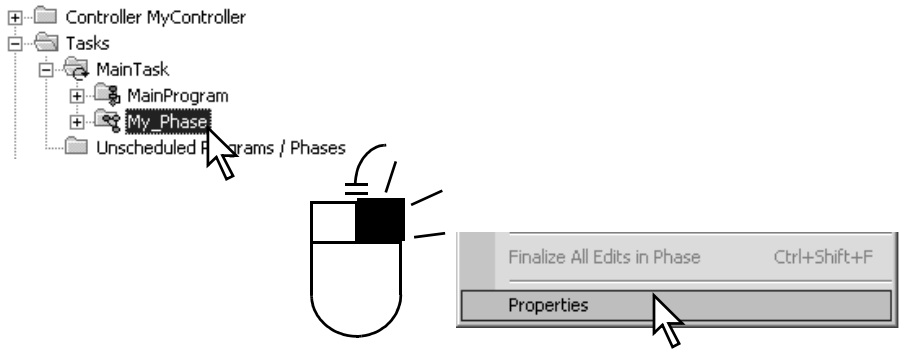
This appendix steps you through the configuration settings for an equipment phase.

When

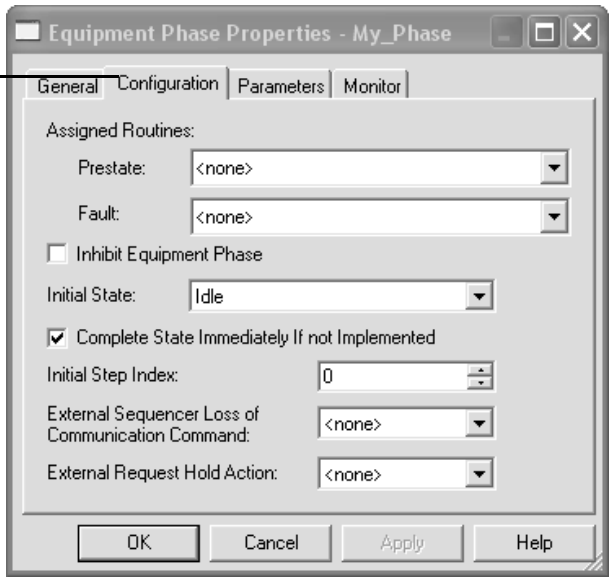
Use this appendix when you want to change the default settings of an equipment phase.

Open the Configuration for an Equipment Phase

1.

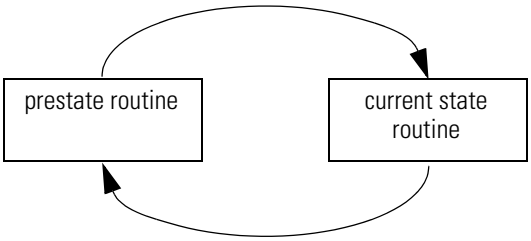


2.



Configure an Equipment Phase

Use the following settings to configure an equipment phase.

Setting	Choices
1. Prestate	 The prestate routine runs all the time, even when the equipment phase is in the idle state. It runs before *each* scan of a state. Do you want to run a prestate routine? • YES — Select the routine that you want to run. • NO — Leave this box set to <none>
2. Fault	A fault routine lets you clear a major fault made by an instruction. Do you want to set up a fault routine for the instructions in this equipment phase? • YES — Select the routine that you want as your fault routine. • NO — Leave this box set to <none>
3. Inhibit Equipment Phase	Do you want the controller to run this equipment phase? • YES — Leave this box unchecked or uncheck it. • NO — Check this box.
4. Initial State	Which state do you want the equipment phase to go to when you turn on the controller? • idle • complete • stopped • aborted
5. Complete State Immediately If not Implemented	Do you want the equipment phase to skip any states that you aren't using? • YES — Leave this box checked or check it. • NO — Uncheck this box.
6. Initial Step Index	A. Are any of the state routines in ladder diagram or structured text? • NO — Skip this box. • YES — Go to step B. B. Do any of those state routines use step numbers? • YES — Type the number for the first step of each state. • NO — Skip this box. The tag for the equipment phase has a 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.

Setting	Choices
7. External Sequencer Loss of Communication Command	A. Are you using RSBizWare Batch software to command this equipment phase? • NO — Skip this box. • YES — Go to step B. B. If the controller loses communication with RSBizWare Batch software, what do you want the equipment phase to do? • Continue in its current state — Select <none>. • Go to aborting — Select Abort. • Go to holding — Select Hold. • Go to stopping — Select Stop. The equipment phase must still follow the state model. For example, it goes to holding only if it is in running or restarting when communication fails.
8. External Request Hold Action	A. Are you using any PXRQ instructions? • NO — Skip this box. • YES — Go to step B. B. What do you want to do if an equipment phase goes to holding while a PXRQ instruction is in process? • Nothing — Select <none>. • Stop the request — Select Clear.

Notes:

R

S

T

How Are We Doing?

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

You can complete this form and mail (or fax) it back to us or email us at
RADocumentComments@ra.rockwell.com

Pub. Title/Type PhaseManager

Cat. No. _____ Pub. No. LOGIX-UM001A-EN-P Pub. Date May 2005 Part No. 957899-90

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

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

Your Name _____

Your Title/Function _____

Location/Phone _____

Would you like us to contact you regarding your comments?

____No, there is no need to contact me

____Yes, please call me

____Yes, please email me at _____

____Yes, please contact me via _____

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

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

PLEASE FASTEN HERE (DO NOT STAPLE)

Other Comments

PLEASE FOLD HERE

PLEASE REMOVE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 18235 CLEVELAND OH

POSTAGE WILL BE PAID BY THE ADDRESSEE

Allen-Bradley
RELIANCE ELECTRIC
DODGE
ROCKWELL SOFTWARE
Rockwell Automation

1 ALLEN-BRADLEY DR
MAYFIELD HEIGHTS OH 44124-9705

Rockwell Automation Support

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

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

Installation Assistance

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

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

New Product Satisfaction Return

Rockwell tests all of 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:

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

www.rockwellautomation.com

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

Headquarters for Allen-Bradley Products, Rockwell Software Products and Global Manufacturing Solutions
Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation SA/NV, Vorstlaan/Boulevard du Souverain 36, 1170 Brussels, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Headquarters for Dodge and Reliance Electric Products
Americas: Rockwell Automation, 6040 Ponders Court, Greenville, SC 29615-4617 USA, Tel: (1) 864.297.4800, Fax: (1) 864.281.2433
Europe/Middle East/Africa: Rockwell Automation, Herman-Heinrich-Gossen-Strasse 3, 50858 Köln, Germany, Tel: 49 (0) 2234 379410, Fax: 49 (0) 2234 3794164
Asia Pacific: Rockwell Automation, 55 Newton Road, #11-01/02 Revenue House, Singapore 307987, Tel: (65) 6356 9077, Fax: (65) 6356 9011

Publication LOGIX-UM001A-EN-P - May 2005 PN 957899-90
Supersedes Publication -

Allen-Bradley

PhaseManager

User Manual