

PlantPAx Display and Library Guidelines

Allen-Bradley
by ROCKWELL AUTOMATION

Reference Manual

Original Instructions

Important User Information

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

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

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

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

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

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

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

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

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

ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

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

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

SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

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

The following icon may appear in the text of this document.

Identifies information that is useful and can help to make a process easier to do or easier to understand.

Chapter 3

Configure the Graphic Framework

Chapter 4

Organization, Ownership, Arbitration, and Propagation (OOAP)

Chapter 5

Ownership (raP_Opr_Owner)

Chapter 6

Arbitration (raP_Opr_ArbitrationQ)

Chapter 7

Organizational Scan (raP_Opr_OrgScan)

Chapter 8

Organizational View (raP_Opr_OrgView)

Chapter 9

Organizational Node Interface (raP_Opr_OrgDeviceCtrl)

Chapter 10

Process Area Module (raP_Opr_Area)

Chapter 11

Process Unit (raP_Opr_Unit)

Chapter 12

Generic Equipment Module (raP_Opr_EMGen)

Chapter 13

Variable State Machine (raP_Opr_VSM)

Chapter 14

Generic Equipment Phase (raP_Opr_EPGen)

Chapter 15

Parameter and Reports (raP_Tec_ParRpt)

Chapter 16

Operator Prompt (raP_Opr_Prompt)

Chapter 17

Logix Diagnostic Objects

Appendix A

FactoryTalk View Customization Tool

Appendix B

Command Sources and Device Virtualization

Appendix C

Tag Extended Properties and Default Alarm Settings

Appendix D

HMI Navigation

Appendix E

5094-IF8IH to PAH Configuration Example

1756-IF8IH with raP_Tec_HARTChanData_to_PAH Add-On Instruction Configuration Example

Appendix F

About This Publication

This publication Describes the PlantPAx® graphic framework, Add-On Instructions, and associated faceplates that are available to develop applications.

Download Firmware, AOP, EDS, and Other Files

Download firmware, associated files (such as AOP, EDS, and DTM), and access product release notes from the Product Compatibility and Download Center (PCDC) at rok.auto/pcdc.

When you update software or firmware revisions, we recommend that you verify the impact on performance and memory utilization before implementing the upgrade on the production system. For FactoryTalk® View or ControlLogix® platforms, we recommend that you review the release notes and verify the impact of the upgrade on performance and memory utilization.

You can also verify the compatibility of the upgrade with the installed software and operating systems in use on your PlantPAx system. See the Product Compatibility and Download Center.

Summary of Changes

This publication contains the following new or updated information. This list includes substantive updates only and is not intended to reflect all changes.

Topic	Page
Updated Organization, Ownership, Arbitration, and Propagation (OOAP) chapter	125
Updated Ownership (raP_Opr_Owner) chapter	177
Updated Arbitration (raP_Opr_ArbitrationQ) chapter	181
Updated Organizational Scan (raP_Opr_OrgScan) chapter	185
Updated Organizational View (raP_Opr_OrgView) chapter	187
Added Organizational Node Interface (raP_Opr_OrgDeviceCtrl) chapter	189
Added Variable State Machine (raP_Opr_VSM) chapter	257
Updated Generic Equipment Module (raP_Opr_EMGen) chapter for Pre-defined state faceplates	235, 246
Removed Studio 5000 View Designer support	Throughout

Additional Resources

These documents contain additional information concerning related products from Rockwell Automation. You can view or download publications at rok.auto/literature.

Resource	Description
Selection Guide, publication PROCES-SG001	Helps you understand the elements of the PlantPAx system to make sure that you buy the proper components.
Template User Manual, publication 9528-UM001	Provides direction on how to install and deploy PlantPAx virtual templates.
Configuration and Implementation User Manual, publication PROCES-UM100	Provides system guidelines and instructions to assist with the development of your PlantPAx system.
Rockwell Automation Sequencer Object, Publication PROCES-RM202	Provides an overview of how to use the Rockwell Automation Sequencer Object. The manual includes a Sequencer programming demonstration, example, and configuration instructions.
PlantPAx Faceplates for Process Controller Instructions, publication PROCES-RM203	Describes the PlantPAx Process instructions, and associated faceplates that are available to develop applications.
PlantPAx Process Control Instructions, publication PROCES-RM215	This manual provides a programmer with details about the available Process instruction set for a Logix-based Process controller.
Process Object parameters Spreadsheet, publication, PROCES-RD200	Describes the PlantPAx Process object parameters.
PlantPAx Visualization Files, publication, PROCES-RD201	Describes the visualization files that are required for the Library of Process Objects.
FactoryTalk Optix Solutions, OPTIX-AT001	Provides an overview of the system, application examples, and ordering guidelines to help you choose exactly what you need. It also guides you through the basics of creating and deploying your own application.
EtherNet/IP Network Devices User Manual, publication ENET-UM006	Describes how to configure and use EtherNet/IP™ devices to communicate on the EtherNet/IP network.
Ethernet Reference Manual, publication ENET-RM002	Describes basic Ethernet concepts, infrastructure components, and infrastructure features.
System Security Design Guidelines Reference Manual, publication SECURE-RM001	Provides guidance on how to conduct security assessments, implement Rockwell Automation products in a secure system, harden the control system, manage user access, and dispose of equipment.

Resource	Description
UL Standards Listing for Industrial Control Products, publication CMPNTS-SR002	Assists original equipment manufacturers (OEMs) with construction of panels, to help ensure that they conform to the requirements of Underwriters Laboratories.
American Standards, Configurations, and Ratings: Introduction to Motor Circuit Design, publication IC-AT001	Provides an overview of American motor circuit design based on methods that are outlined in the NEC.
Industrial Components Preventive Maintenance, Enclosures, and Contact Ratings Specifications, publication IC-TD002	Provides a quick reference tool for Allen-Bradley® industrial automation controls and assemblies.
Safety Guidelines for the Application, Installation, and Maintenance of Solid-state Control, publication SGI-1.1	Designed to harmonize with NEMA Standards Publication No. ICS 1.1-1987 and provides general guidelines for the application, installation, and maintenance of solid-state control in the form of individual devices or packaged assemblies incorporating solid-state components.
Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1	Provides general guidelines for installing a Rockwell Automation industrial system.
ProposalWorks™ configuration software, rok.auto/systemtools	Helps configure complete, valid catalog numbers and build complete quotes based on detailed product information.
Rockwell Automation Global SCCR tool, rok.auto/sccr	Provides coordinated high-fault branch circuit solutions for motor starters, soft starters, and component drives.
Product Certifications website, rok.auto/certifications	Provides declarations of conformity, certificates, and other certification details.

Rockwell Automation Library of Process Objects

The Rockwell Automation® Library of Process Objects, also referred to in this document as the PlantPAx® library, contains all the tools required to enable consistent deployment and faster product delivery.

The Library Includes the following:

- Graphics for built-in instructions
- HMI images and Help files
- Logix diagnostic objects
- Process objects
- Premier Integration objects
- Control strategies
- Sequencer objects
- Color Change tool
- Historian -- Asset Framework template and objects

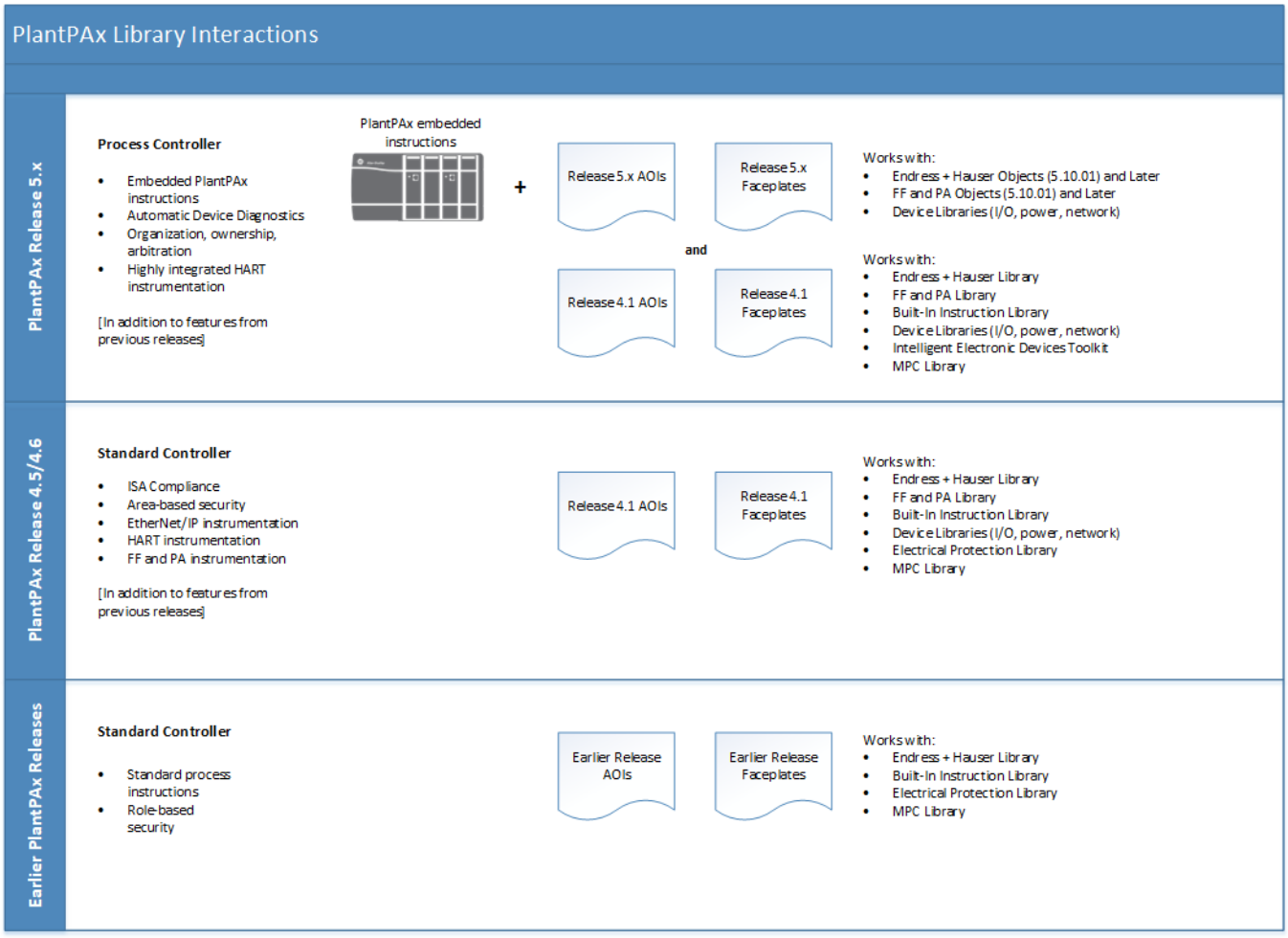
The Library of Process objects is designed to work in conjunction with the following libraries:

Item	Description
I/O Device Library	Provides objects for Rockwell Automation 1756, 1769, 1734, 1794, 1738, 1732E, 1719, 5069, 5094 I/O modules. Provides preconfigured status and diagnostic faceplates sets for Rockwell Automation digital and analog I/O devices. You can use these objects with Machine Builder, Process, and Packaged Libraries, or as standalone components.
IO-Link Device Library	Provides IO-Link master and sensor objects. Provides preconfigured status and diagnostic faceplates.
Machine Builder Libraries	Library objects for use with Application Code Manager. • Independent Cart Technology Libraries, includes ICT Libraries for iTRAK® and MagneMotion® • Studio 5000® Application Code Manager • Power Device Library, including objects for E300, ArmorStart®, PowerFlex®, and Kinetix®
Network Device Library	Provides objects for Stratix® switch and Device Level Ring network objects.
Power Device Library	Provides objects for discrete and velocity power devices.

In addition to the libraries, the Intelligent Electronic Devices Toolkit contains Add-On Instructions and visualization objects. The Intelligent Electronic Devices Toolkit supports communication via the ProSoft Technology IEC 61850 communication module, MVI56E-61850C, and the ProSoft Technology EtherNet/IP™ Server to IEC 61850 Dual Port Client Gateway, PLX82-EIP-61850. See Rockwell Automation Intelligent Electronic Devices Toolkit, publication PROCES-RM211 for more information.

When you deploy the process controller in PlantPAx 5.0 and later, you gain access to additional PlantPAx instructions. The PlantPAx instructions on the process controller provide objects that are embedded in the controller firmware. For more information on faceplates for these instructions, see PlantPAx Faceplates for Process Controller Instructions, publication PROCES-RM203.

See Logix 5000® Advanced Process Control and Drives Instructions, publication 1756-RM006 for more information on PlantPAx Instructions.



Rockwell Automation Services and Support

System Support offers technical assistance that is tailored for control systems. Some of the features include the following:

- Highly experienced team of engineers with training and systems experience
- Process support at a systems-level that is provided by process engineers
- Use of online remote diagnostic tools
- Access to otherwise restricted TechConnect^{SM} Knowledgebase content
- 24-hour, 7 days per week, 365 days per year of phone-support coverage upgrade option

For more information, contact your local distributor or Rockwell Automation representative or see https://www.rockwellautomation.com.

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

Process Library 5.30 Add-On Instructions

In addition to the PlantPAx instructions listed previously, PlantPAx provides several Add-On Instructions.

Libraries noted in the following table:
GEMS - Global Engineering Modular Solutions
RAMS - Rockwell Automation Mining Solutions
PO - Process Objects

Input Control

PlantPAx 5.0 and later Add-On Instruction Bundled with 5.30 Library Download	Previous Process Library Add-On Instruction	Description
raP_Tec_HARTChanData_to_PAH	New Instruction	Transfers data from one Library 4.10 HART module Channel Data array member (for one input or output channel) to one (Highly Integrated HART) PAX_HART_DEVICE:I:0 data structure for use by PAH instruction.
raP_Tec_LgxEvent	New Instruction	Captures any of 16 event bit rising edge transitions and records the lowest-order rising edge bit as the reason for the event.

Controller Diagnostics

PlantPAx 5.0 and later Add-On Instruction Bundled with 5.30 Library Download	Previous Process Library Add-On Instruction	Description
Logix Change Detector (raP_Dvc_LgxChangeDet)	L_ChangeDet (PO)	The Logix Change Detector (raP_Dvc_LgxChangeDet) Add-On Instruction monitors another Logix controller on the network and checks for changes that impact operation. Changes that can be monitored include downloads, online edits, I/O forcing, and controller mode changes.
Logix Controller CPU Utilization (raP_Dvc_LgxCPU_5x80)	L_CPU_5x80 (PO)	The Logix Controller CPU Utilization (raP_Dvc_LgxCPU_5x80) Add-On Instruction monitors a Logix controller, and provides information on controller CPU utilization, communication usage, memory usage, task scan times, and other information. Data that is provided by the L_CPU instruction is useful to diagnose communication or control responsiveness issues and in tuning the performance of control tasks for optimum controller performance. The raP_Dvc_LgxCPU_5x80 instruction can be loaded as part of a control application and disabled (default) until needed. The instruction can also be enabled at a slow update rate for general controller monitoring. The update rate can be increased, if necessary, as directed by a Rockwell Automation Technical Support representative to help diagnose controller performance issues. ControlLogix® 5580 Controllers. This instruction supports ControlLogix 5580 and CompactLogix® 5380 controllers, firmware release 33 and later.

Controller Diagnostics

PlantPAx 5.0 and later Add-On Instruction Bundled with 5.30 Library Download	Previous Process Library Add-On Instruction	Description
Logix Module Status (raP_Dvc_LgxModuleSts)	L_ModuleSts (PO)	The Logix Module Status (raP_Dvc_LgxModuleSts) Add-On Instruction monitors the connection status of one module in the I/O configuration tree of the Logix controller. The instruction provides an I/O fault signal if the connection is not 'running'.
Logix Redundant Controller Monitor (raP_Dvc_LgxRedun)	L_Redun (PO)	The Logix Redundant Controller Monitor (raP_Dvc_LgxRedun) Add-On Instruction monitors one redundant pair of Logix controllers. The instruction checks primary and secondary controller status that can affect the ability of the system to switch to the back-up controller on a failure of the primary.
Logix Task Monitor (raP_Dvc_LgxTaskMon)	L_TaskMon (PO)	The Logix Task Monitor (raP_Dvc_LgxTaskMon) Add-On Instruction monitors one task running in a Logix controller to provide task statistics, such as task scan time and overlap count. The instruction also provides the following: • Task configuration settings, such as priority, rate, and watchdog timer setting • Task 'plan' execution time • Alarm if the planned execution time is exceeded Maintenance commands are provided for clearing the maximum execution time and the overlap count.

Equipment Control

PlantPAx 5.0 and later Add-On Instruction Bundled with 5.30 Library Download	Previous Process Library Add-On Instruction	Description
raP_Opr_Area	AREA (GEMS)	The raP_Opr_Area (Area Object) object groups Units together, and provides a propagation mechanism for aggregating status from Unit objects, and broadcasting commands to Unit Modules.
raP_Opr_Unit	UNIT (GEMS)	The UNIT (Unit Object) object controls a Unit in various command sources and monitors for fault conditions.
raP_Opr_EMGen	EM_GEN (GEMS)	The raP_Opr_EMGen (Generic Equipment Module) object controls an Equipment Module in various modes and monitors for fault conditions.
raP_Opr_EPGen	EP_GEN (GEMS)	The raP_Opr_EPGen (Generic Equipment Phase Module) object controls an Equipment Phase in various modes and monitors for fault conditions.
raP_Tec_ParRpt	I_ParameterEnum (GEMS) I_ParameterInteger (GEMS) I_ParameterReal (GEMS) I_ParameterString (GEMS)	The raP_Tec_ParRpt (Parameter \ Report)Add-On Instruction is used to implement parameter and report data items. The raP_Tec_ParRpt instruction may be used as follows: • For a read-only parameter /report • For a read/write parameter /report • For a parameter /report of type Integer, Real, String, or Enumeration • Equipment Module (raP_Opr_EMGen) and Equipment Phase (raP_Opr_EPGen) are designed to work with the raP_Tec_ParRpt instruction, which may be used for Parameter or Report data items
raP_Opr_Prompt	Prompt (GEMS) P_Prompt (PO)	The P_Prompt (Operator Prompt) Add-On Instruction is a universal mechanism for operator interaction that can be used within a control scheme. The instruction presents an operator with configurable message or data fields and accepts operator response data and confirmation.
Process Extended Alarms (raP_Opr_ExtddAlm)	Extended Alarms (GEMS)	Monitors one input condition and provides one configurable Alarm. The Alarm is provided as a Logix Tag Based Alarm. Use <InstanceTag>.@Alarms members for access.

Organization

Organization is a method by which parent / child relationships can be created and modified among PlantPAx Instructions. Organization provides a method to propagate a selected subset of commands (related to command source, alarms, and so on) from the parent down to its children or propagate the aggregate of a selected subset of status (related to command source, alarms, and so on) from the children up to the parent. For more information See Organization, Ownership, Arbitration, and Propagation (OOAP) on page 125.

PlantPAx 5.0 and later Add-On Instruction Bundled with 5.30 Library Download	Previous Process Library Add-On Instruction	Description
Ownership (raP_Opr_Owner)	Ownership, Command, and Status Propagation (GEMS)	The Add-On Instruction Function to allow ownership of a Bus element.
Organizational View (raP_Opr_OrgView)	Ownership, Command, and Status Propagation (GEMS)	The Add-On Instruction Function to create a tree view of the nodal organization in FactoryTalk® View.
Organizational Scan (raP_Opr_OrgScan)	Ownership, Command, and Status Propagation (GEMS)	The Add-On Instruction Function to scan and update all Bus elements and tree nodes.
Arbitration Queue (raP_Opr_ArbitrationQ)	Ownership, Command, and Status Propagation (GEMS)	The Arbitration Queue (raP_Opr_ArbitrationQ) Add-On Instruction Function to add a FIFO to each class of owner in the ownership function.

Libraries can be accessed from the
Product Compatibility and Download Center.

Visualization Files

FactoryTalk View SE

Each Add-On Instruction has associated FactoryTalk View SE visualization files that provide a common user interface. You must import these files in the following order:

- Images (.png files)
- Global Objects (.ggfx file type)
- HMI faceplates (.gfx file type)
- Tags (.csv file type)
- Macros (FactoryTalk View SE software only) (.mcr file type)
- Local Message files (.loc file type)

File Type Abbreviations	FactoryTalk View SE	Description
Images (.png)	All .png files in the images folder. **IMPORTANT:** FactoryTalk View application renames PNG files when they are imported with a .bmp file extension, but the files retain a .png format.	Common icons that are used in the Global Objects and standard displays for all Process Objects.
Global objects (.ggfx)	(raP-5_30-SE) precedes name of the Global Objects.	Examples: (raP-5_30-SE) Common Objects
Standard displays (.gfx)	(raP-5_30-SE) precedes name of the display.	Examples: (raP-5_30-SE) PAI-Faceplate

File Type Abbreviations	FactoryTalk View SE	Description
HMI tags (.csv)	FTViewSE_ProcessLibrary_Tags_5_0_**xx**.csv where **xx** = the service release number.	HMI tags are created in a FactoryTalk View SE application to support security and other features on Process Library faceplates. HMI tags can be imported via the comma-separated values file (.csv file type).
Macros (.mcr file)	Macros used for the general library: • NavToDisplay • ToggleWithRemark Macro that is used for the PLLS object displays: • NavToPLLS_Motor Macros that are used for the Organization TreeView and navigation: • DefineShowHWTreeCmd.mcr • DefineShowTreeCmd.mcr • NavToBusDevice • NavToBusDeviceWithSC • NavToBusDisplay • ShowTreeForObject • NavToDisplay with line of sight • NavToDisplay_with_4_x • NavToDisplayIndirect • NavToFaceplate_with_ETP • NavToVSM	In a FactoryTalk View SE application, a macro is a series of commands that are stored in a text file.
Local Message Files	• SystemMaterialNames • SystemStepDescriptions • SystemSummary	Local message files used by raP_Opr_EMGen, raP_Opr_EPGen, and raP_Opr_Unit.

Images are external graphic files that can be used in displays. They must be downloaded from PCDC to be used by FactoryTalk View software.

Global object files contain Graphic Symbols that are created once and referenced multiple times on multiple displays in an application. When changes are made to a global object, all instances in the application are automatically updated.

Global objects serve two purposes:

- Toolbox files contain common elements that are used to build faceplate displays.
- Graphic Symbols files contain device symbols that you can use to build your application displays. Select the symbol to open the corresponding faceplate display.

Standard display files, commonly called faceplates, provide a common user interface.

FactoryTalk Optix

The following instructions also have associated FactoryTalk® Optix™ visualization types: PDI, PDO, PD4SD, PAI, PAID, PAO, PDBC, PDOSE, PHLS, PPID, PMTR, PRT, PVSD, PVLV, PVLVS, PINTLK, PAH, PAIM, PFO, PLLS, PNPOS, PVLVMP, PRI, PPERM, raP_Opr_Area, raP_Opr_Unit, raP_Opr_EMGen, raP_Opr_EPGen, raP_Tec_ParRpt, raP_Opr_Prompt (Runtime only), raP_Opr_ExtddAlm, raP_Dvc_EH_Flowmeter, raP_Dvc_EH_Heartbeat, raP_Dvc_EH_Sensor, and all Logix Diagnostic instructions.

You must use the project provided with the library as the starting point for your project. FactoryTalk Optix content for other library instructions will be added in a future release.

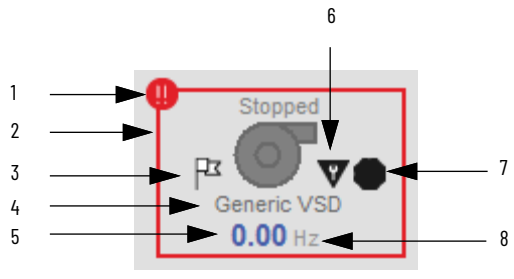
File Type Abbreviations	FactoryTalk Optix	Description
Window	UI\MainWindow Panels\MainLibraryObjects	The main window contains all graphical elements displayed at design time in FactoryTalk Optix Studio and at runtime in your FactoryTalk Optix Application.
Panels	UI\RockwellAutomationLibr aries	Panels, commonly called faceplates, provide a common user interface
Widgets	–	Widgets are pre-configured objects that allow you to interact with and configure the device at runtime. Widgets are analogous to Global Objects used in FactoryTalk View SE
Graphic symbols	Graphic symbols may be found in the “Graphic Symbols” folder for an object.	Reusable and repeatable graphic content to be used on panels.
Images (.svg)	All image files are in the ProjectFiles\res folder in the template application.	Common icons that are used in the Widgets and Panels for all Process Objects.

Basic Attributes and Indicators

This section shows examples of visual indicators that are common for graphic symbols in the Rockwell Automation Library of Process Objects. Visual indicators are critical to the daily operation of a plant. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

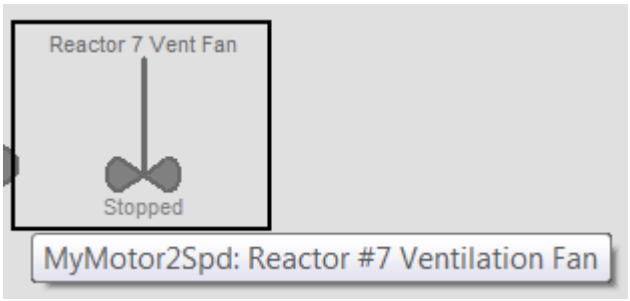
Common attributes of graphic symbols typically include:

- Status/quality/threshold indicator
- Maintenance bypass indicator
- Engineering units
- Label
- Command Source indicator (only for non-analog inputs)
- Alarm border that changes color and blinks on unacknowledged alarm
- Alarm indicator symbol that changes with the severity of an alarm

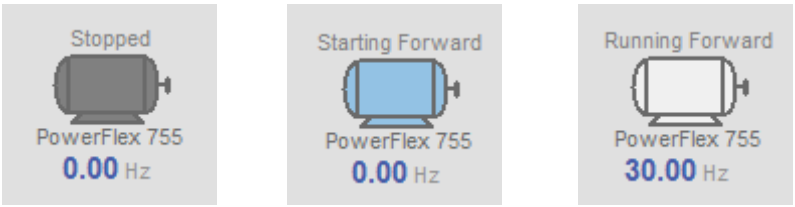


Item	Description
1	Alarm Indicator
2	Alarm Border
3	Command source indicator (In the example the flag indicates not in normal command source)
4	Label
5	Process Variable
6	Maintenance bypass indicator
7	Not Ready indicator
8	Engineering units

Each graphic object includes a touch field over it that opens the faceplate. In addition, there is a tooltip (for FactoryTalk View SE only) on the graphic symbol that displays the configured tag and description.



State Indicators



The State Indicator text and the color change depending on the state of the drive. The indicators and colors are common across all Add-On Instructions. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Color	State
Dark Gray	Stopped, De-energized, Closed
Light Blue	Transitioning. examples: Starting, Jogging, Stopping, Opening, Closing, Moving
Light Blue	Horn
White	Running, Energized, Open

Status Quality Indicators

One of these images appears on the graphic symbol when the described condition is true. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Image	Description	Image	Description
No symbol displayed	I/O communication and quality good, configuration valid		Accelerating
	Invalid Configuration		Decelerating
	Invalid Configuration **FactoryTalk Optix Only**		Value is being initialized
	Data quality bad / failure		Value has not changed (stuck)
	Data Quality degraded: uncertain, test, virtual, substitution, or out of specification		Value is being replaced
	Device not ready to operate		Input matches target
	The input or device has been disabled		input does not match target

Image	Description	Image	Description
	Alarm Inhibit (Suppressed or Bypassed)		Auto loop mode
	Device in loopback test		Manual loop mode
	At target speed		Cascade loop mode
	Speed ref limited to the minimum / maximum		Motor not controllable
	Value infinite or not a number		Process Variable within setpoint deadband (no control action occurs)
	value is being held at last good value		Raise Process Variable output that is energized
	Input Controlled Variable that is clamped to minimum / maximum		Lower Process Variable output that is energized
	Output Controlled Variable that is clamped to minimum / maximum	–	–

When the Invalid Configuration indicator appears, you can find what configuration setting is invalid by following the indicators. Select the graphic symbol to open the faceplate. The Invalid Configuration indicator appears next to the appropriate tab at the top of the faceplate to guide you to the configuration error. Once you navigate to the tab, the misconfiguration is flagged with this indicator.

Threshold Indicators

These indicators show that the process variable has exceeded a threshold. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Image	Description
	High-high threshold exceeded
	High threshold exceeded
	Low threshold exceeded
	Low-low threshold exceeded

Deviation Indicators

These indicators warn of exceeding the deviation limits. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Image	Description
	High-high deviation exceeded
	High deviation exceeded
	Low deviation exceeded
	Low-low deviation exceeded

Command Source Indicators

The command source indicator displays by exception only. For example, if the device is operating normally, there is not an indicator. If the device is out of service (OoS), then the OoS indicator is displayed. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Command source indicators are not used for analog inputs.

Image	Description
No symbol displayed	Device is in normal command source operation
	Device is out of service
	Device is not in normal command source operation
	Device is in program command source operation
	Device is in program locked command source
	Device is in maintenance command source operation
	Device is in operator command source operation
	Device is in external command source operation

Image	Description
	Device is in operator locked command source operation
	Device is in override command source operation
	Device is in hand command source operation

Maintenance Bypass Indicator

The maintenance bypass indicator appears to the right of the label to indicate that a maintenance bypass has been activated. The Maintenance bypass indicator also appears when the Substitute PV function is enabled. A Maintenance-entered value supersedes the 'live' process variable. These indicators are applicable FactoryTalk View SE and FactoryTalk Optix applications.

Image	Description
	A maintenance bypass is active
No symbol displayed	No maintenance bypass is active

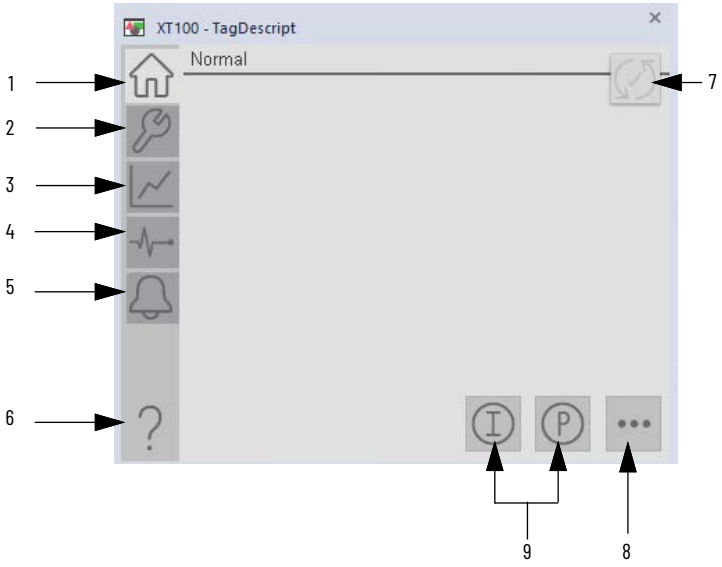
When the Maintenance bypass indicator appears, you can find what condition was bypassed by following the indicators. Select the graphic symbol to open the faceplate. The Maintenance bypass indicator appears next to the appropriate tab at the top of the faceplate to guide you to the bypass. Once you navigate to the tab, the bypassed item is flagged with this indicator.

Basic Faceplate Attributes

Faceplates consist of tabs, and each tab consists of one or more pages (or accordions in FactoryTalk Optix). The Operator (Home) tab is displayed when the faceplate is initially opened. The faceplate provides the means for operators, maintenance personnel, engineers, and others to interact with an instruction instance, which includes a view of its status and values. Faceplates also manipulate an instruction through its commands and settings. Select the appropriate icon on the left of the faceplate to access a specific tab. This section provides an overview of the faceplate attributes that are common across the objects. More details are supplied in the individual section for each object.

IMPORTANT The faceplates that are shown are FactoryTalk View SE faceplates. The FactoryTalk Optix faceplates contain the same basic attributes, but use accordions rather than multiple pages. In FactoryTalk Optix, the trend, help, and more information buttons are located in the faceplate title bar.

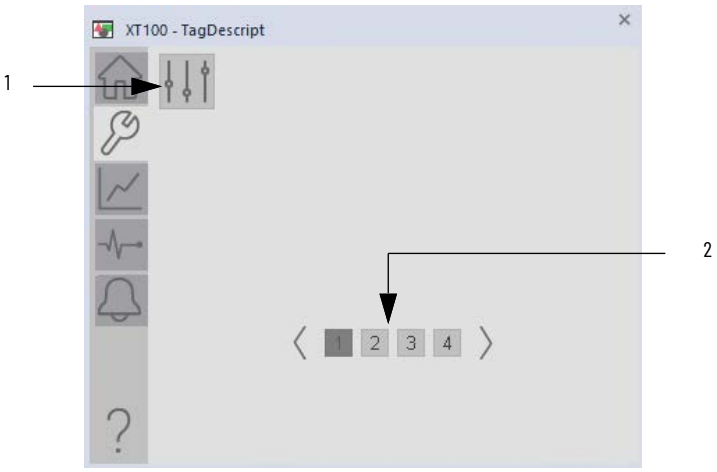
Operator (Home) Tab



Item	Action
1	Select to open the operator tab.
2	Select to open the maintenance tab.
3	Select to open the trends tab.
4	Select to open the diagnostics tab.
5	Select to open the alarm tab.
6	Select to open the help file.
7	Select to reset and acknowledge all alarms.
8	Select to enable navigation to an object with more information (Cfg_HasMoreObj is set to true.) You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the object's faceplate.
9	If the object is configured to have permissive and interlock objects (for example, Cfg_HasPermObj (Fast or Slow) or Cfg_HasIntlkObj is true), the permissive and interlock indication become buttons. These buttons open the faceplates of the source objects that are used as a permissive or interlock. Often this is a PPERM or PINTLK instruction. If the object is not configured in this way, the permissive or interlock symbols are indicators only.

Maintenance Tab

In the maintenance tab, there is a button for Advanced properties. There are also page identifiers at the bottom if there are multiple configuration pages. See the following diagram for common attributes of the maintenance tab.

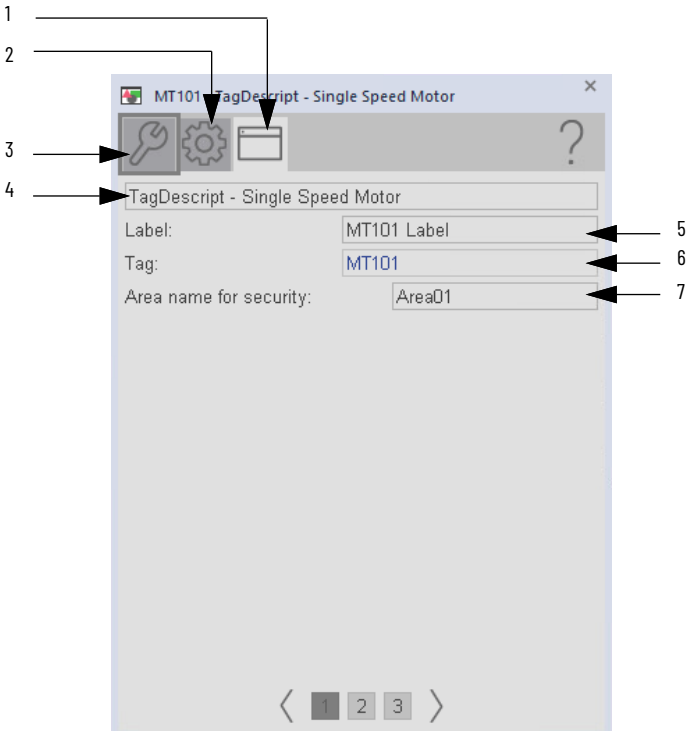


Item	Action
1	Select to open the Advanced Properties.
2	Page identifiers

Advanced Properties

The advanced maintenance, engineering, HMI configuration, Diagnostics, and Faults tabs for the objects are available in the advanced properties faceplate. The advanced maintenance and engineering tabs have object-specific configurations that are detailed for each object.

The HMI configuration tab has settings that are common to the objects. See the following diagram for common attributes of the HMI configuration tab.



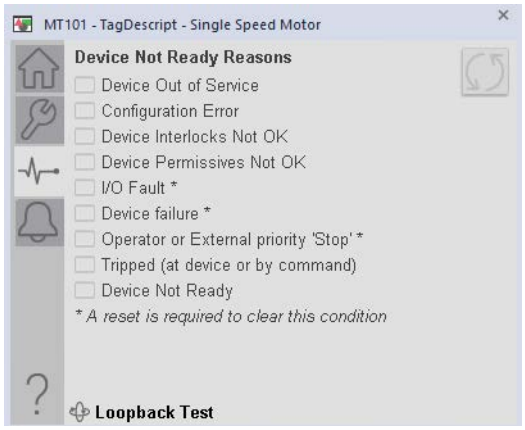
Item	Action
1	Select to open the HMI Configuration tab.
2	Select to open the engineering tab.
3	Select to open the Advanced Maintenance tab.
4	Device description that shows on the faceplate title bar.
5	Label to show on the graphic symbol.
6	Tag name that shows on the faceplate and on the tooltip for graphic symbols.
7	Area name for security.

Hover the cursor over the tag name to see the actual network path and tag name that is associated with the object.

Diagnostics Tab

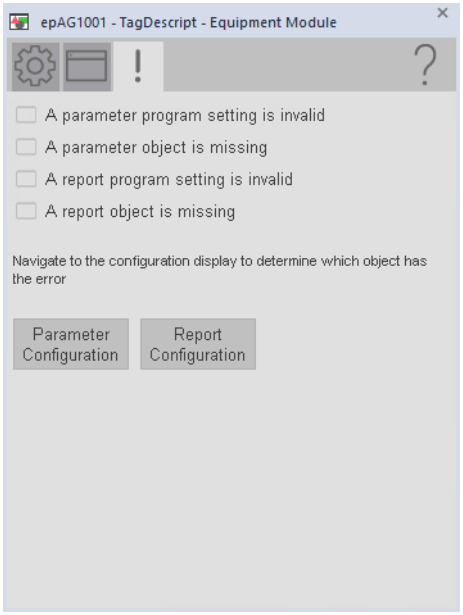
The Diagnostic tab provides indications that are helpful to diagnose or help prevent device problems. These problems can include specific reasons a device is 'Not Ready', device warnings and faults, warning and fault history, and predictive/preventive maintenance data.

The Diagnostics tab displays possible reasons for the device not being ready.



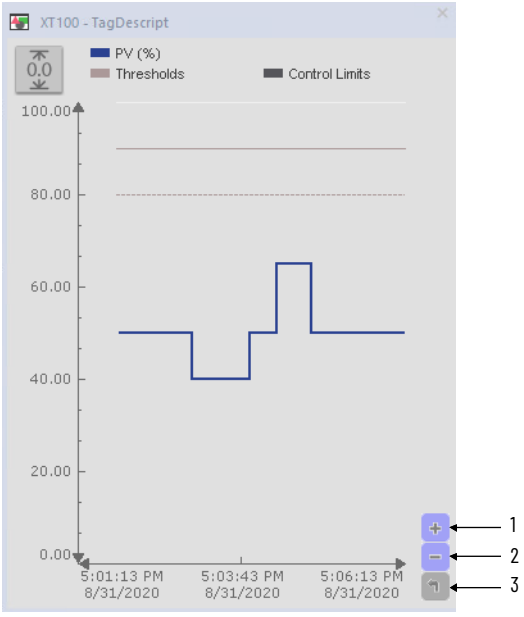
Faults Tab

The faults tab contains specific reasons that the device is not ready.



Trends Display

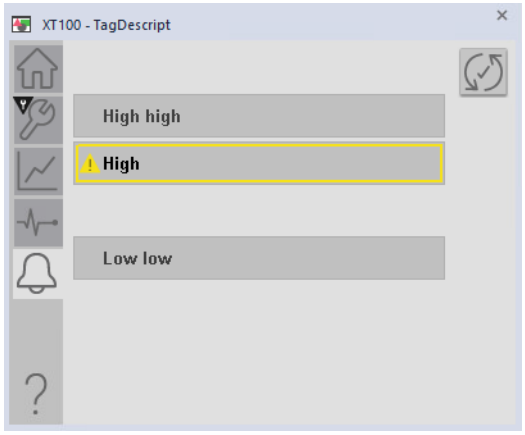
The Trends display shows trend charts of key device data over time. These faceplate trends provide a quick view of current device performance to supplement, but not replace, dedicated historical or live trend displays.



Item	Action
1	Select to zoom in
2	Select to zoom out
3	Select to reset view

Alarms Tab

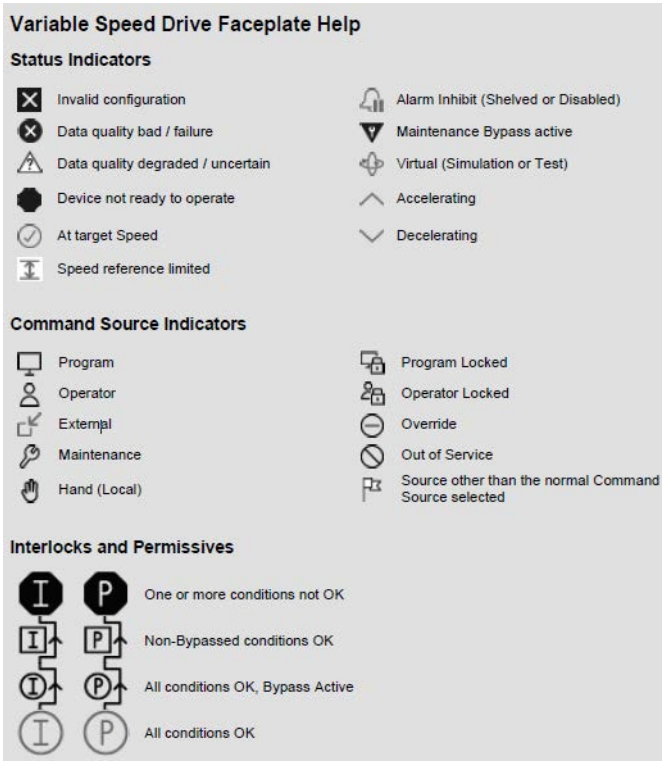
The Alarms tab displays each configured alarm. The icon on the tab for the alarms page has an outline that changes color to show the current active alarm status.

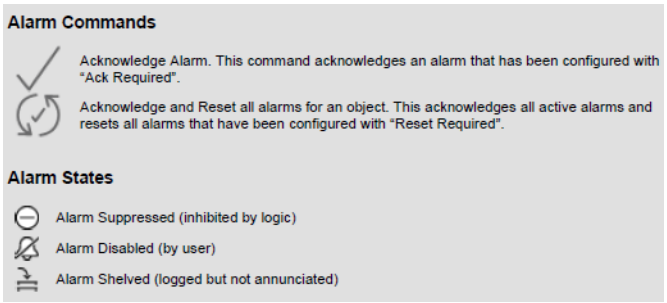


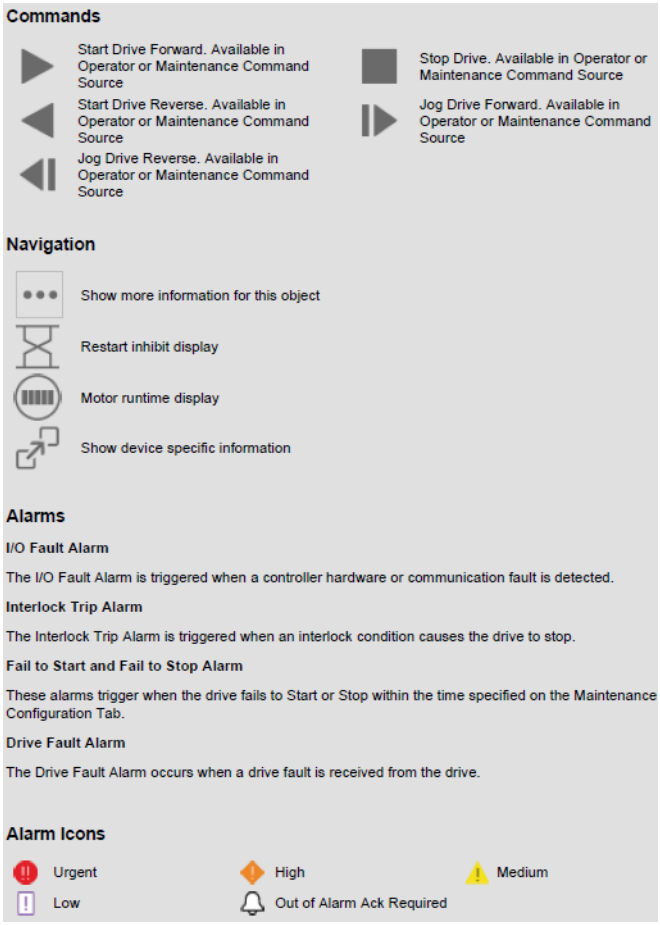
Help Button

Press the help button on the faceplates to access help specific to that faceplate. The help button can be used to access help files provided with the library download in a .pdf format (See Help Files on page 40) or any specified URL configured in the object.

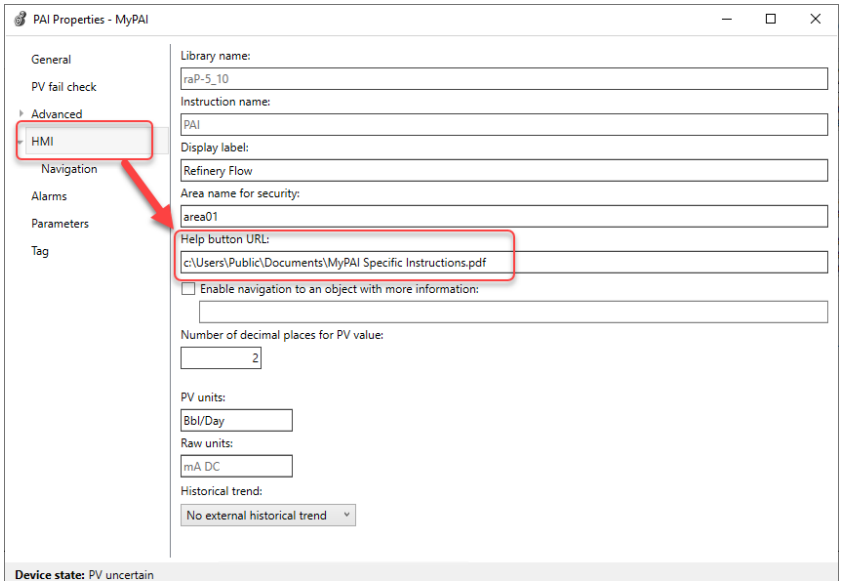
See the following example:



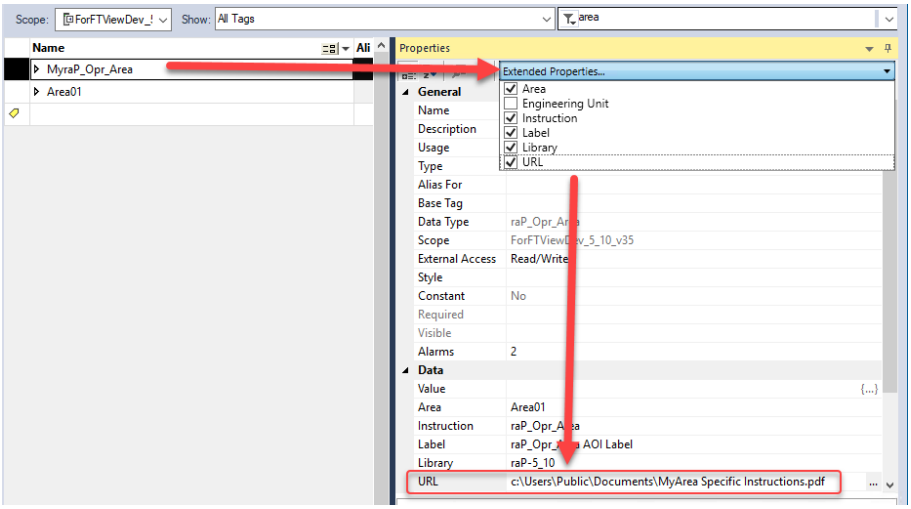




To use a specific URL with the help button for an object, open the object's dialog box in Logix Designer. Select the "HMI" tab and input the URL under "Help button URL:"



For a library object that is an AOI, find the object base tag in the controller tags. Select the tag and look at the properties panel. The extended tag property "URL" can be updated here.



The URL can be any file path or web URL. If the Help button URL is left blank, the button will default to using the Help files provided with the library download.

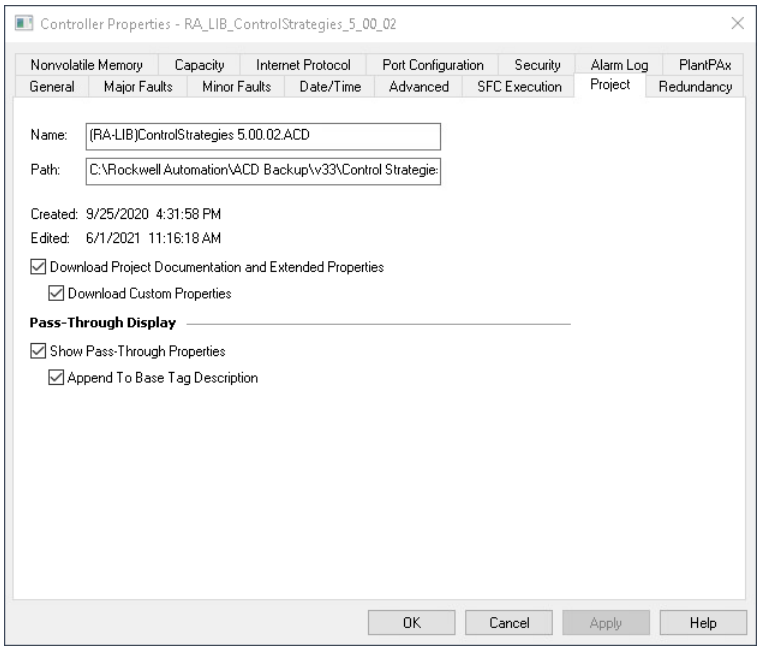
Studio 5000 Logix Designer Project Configuration

The Library of Process Objects 5.0 and later uses the Extended Tag Properties feature inside Studio 5000 Logix Designer®. When configuring your Studio 5000 Logix Designer project file, the following boxes must be checked (checked by default):

- Download Project Documentation and Extended Properties
- Download Custom Properties
- Show Pass-Through Properties
- Append To Base Tag Description

Note: Configuring the properties (.@Library, .@Instruction, .@Area, and .@Labels) incorrectly will result in empty values that cause an error when calling up the relevant HMI faceplate.

IMPORTANT	The pass-through feature of extended tag properties is currently only compatible with FactoryTalk View SE. The Pass-Through feature of Extended Tag Properties is not compatible with FactoryTalk Optix. If using FactoryTalk Optix faceplates, all extended tag properties MUST be entered directly or errors will be present.

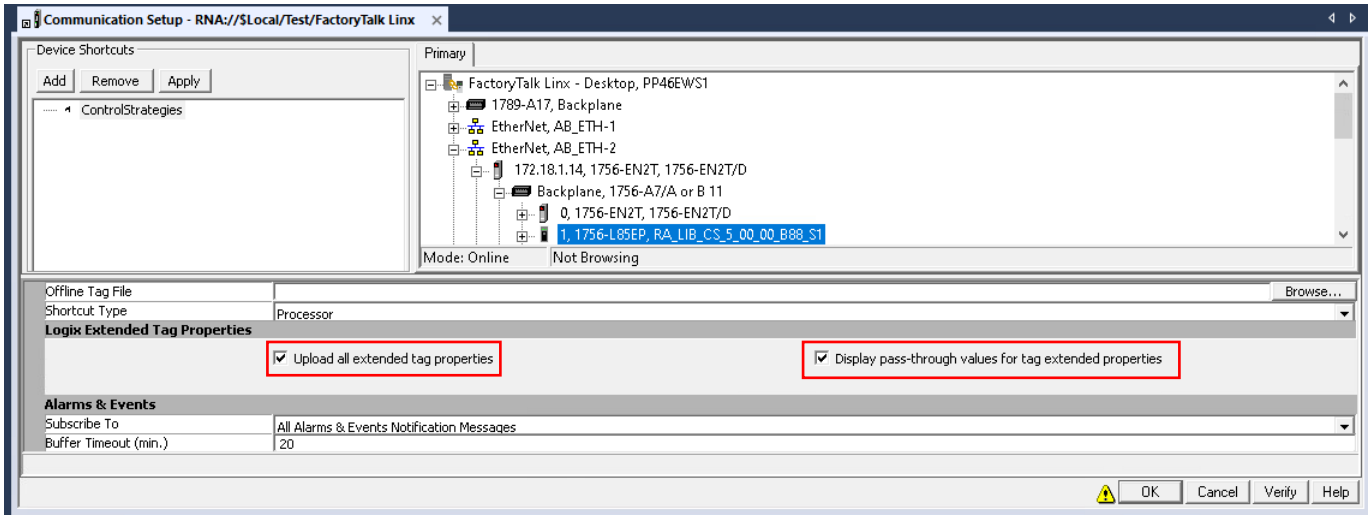


FactoryTalk Linx Device Shortcuts Configuration

The Library of Process Objects 5.0 utilizes the feature Extended Tag Properties inside Studio 5000 Logix Designer. When configuring FactoryTalk® Linx communication setup for device shortcuts, the following boxes must be checked:

- Upload all extended tag properties
- Display pass-through values for tag extended properties

  Note: Configuring the shortcuts incorrectly results in errors if extended tag properties are left blank.

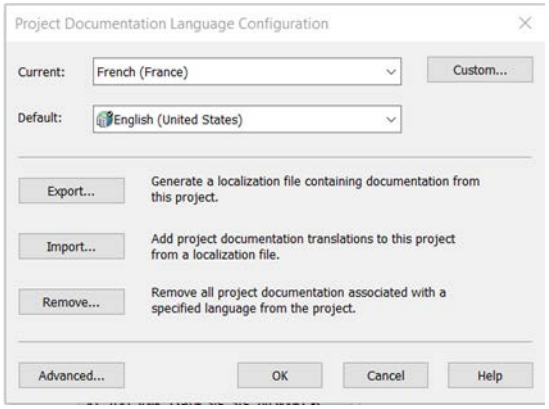


Language Switching

Language Switching in a Controller Project

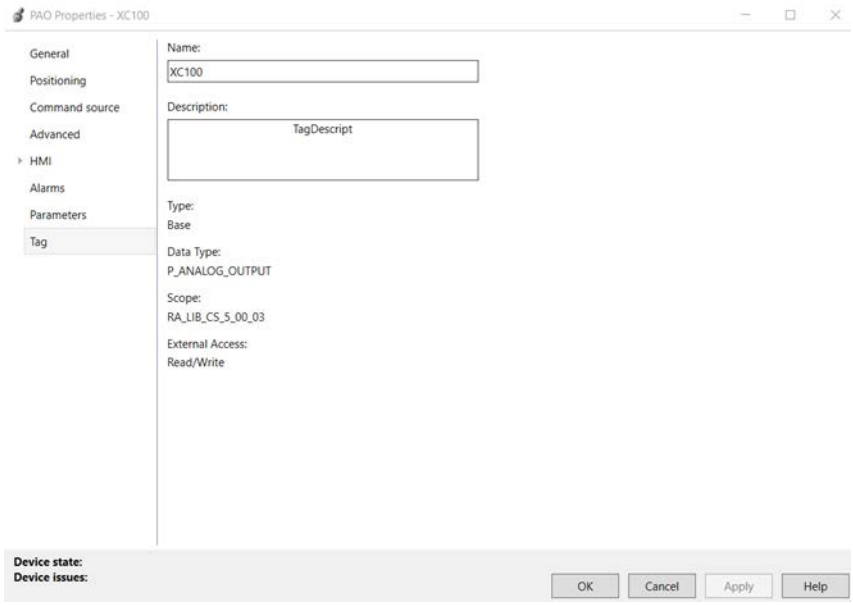
In the Logix Designer application, to display project content in languages other than English, do the following:

1. Go to Tools > Documentation Languages and select a language in the Current dropdown menu.

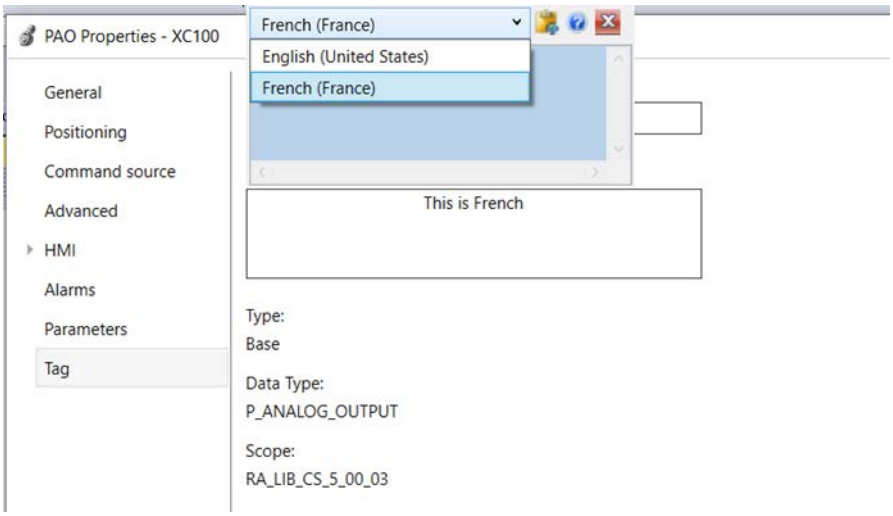


2. Click OK
3. Open a Control Strategy and open the parameters for the main object.

This example shows the Tag properties for a PAO Control Strategy.



4. In the Description box, enter the description text.
5. With your cursor in the Description box, you can switch between French and English so that you can enter the text for each language.

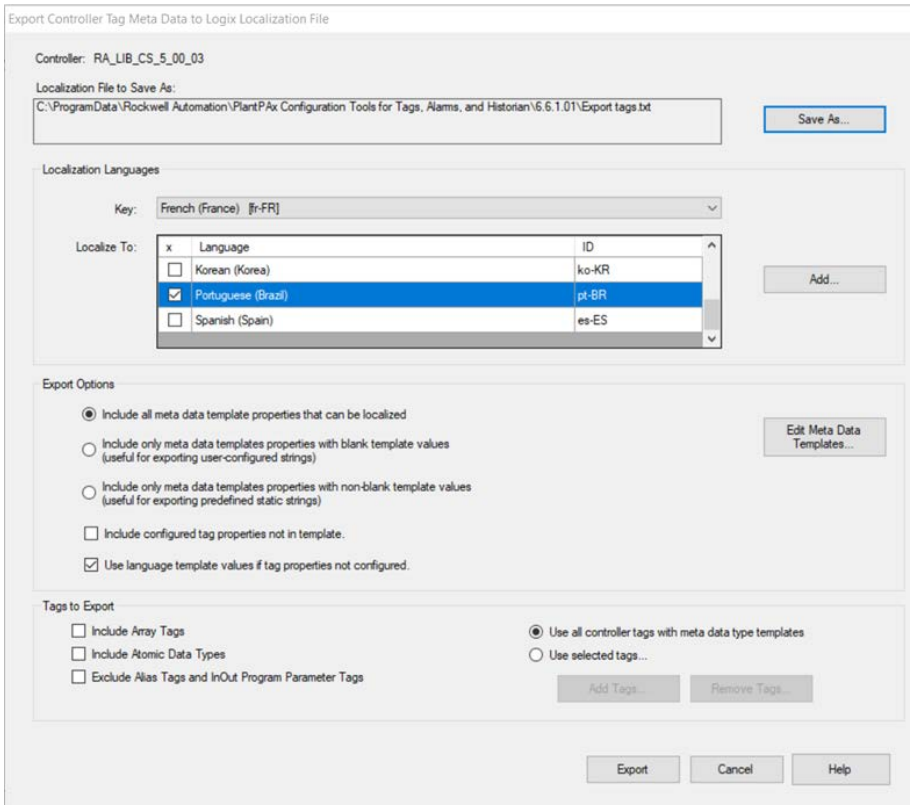


Bulk Edit Translated Content

To edit multiple text strings (rather than opening the properties for each object in your program), use the PlantPAx Configuration Tool.

1. Open the PlantPAx Configuration Tool, and if necessary, add the controller for your project.
2. Select the controller and select Export Tag Meta Data to Logix Localization File.
3. Name the export file and select the files that you want to export.
4. Select Use language template values if tag properties not configured.

There are default values in the tag properties for tags in the Control Strategies. These are typical text strings to translate.



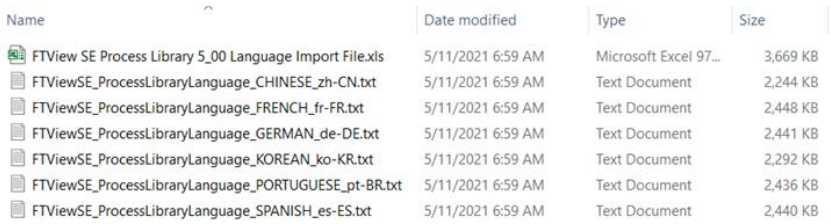
5. Select Export

The exported data opens in an Excel® spreadsheet. Adjust the columns widths to see the data. There is one column for each language that you selected. Enter the translated content for each tag property and language.

To import the translated content, open the controller project in the Logix Designer application and select Tools > Documentation Languages > Import and select the spreadsheet file with the translated strings.

Language Switching in a FactoryTalk View SE Project

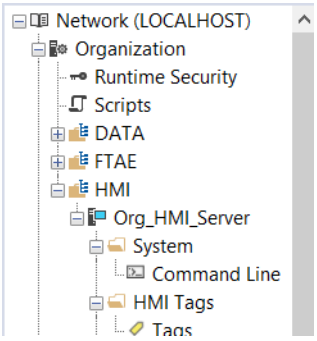
In the library Tools & Utilities > Language Translations > HMI folder, there are already translated strings for the faceplates in the .txt files. The Excel file contains all languages.



Import All Languages

If you want to import all of the languages, you can use the Excel spreadsheet.

Open the spreadsheet and adjust the columns as needed. For example, with this project:



Replace (do a Find and Replace All):

/Server:Server with / Organization/HMI:Org_HMI_Server

Save your changes.

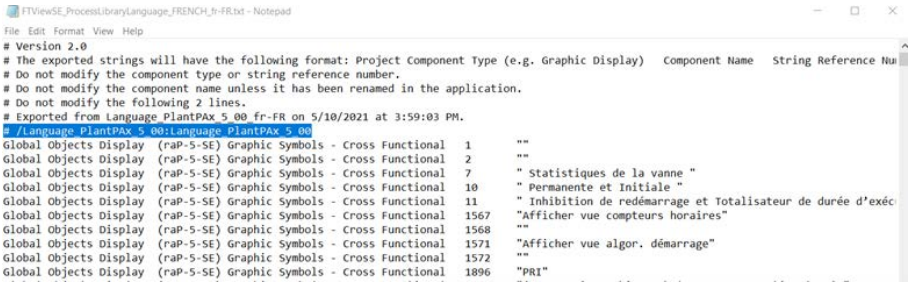
In FactoryTalk® View Studio, to import the updated translation, go to Tools > Languages.

1. Click Add to add languages.
2. Select each language and click Apply.
3. Click Import.
   - Select Import strings from an Excel spreadsheet into all application languages defined in the spreadsheet.
   - Click Next to select the updated file from the library.

It takes a few minutes to import the translated text.

Import a Single Language

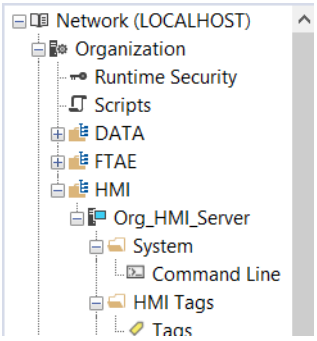
Before you can import these files into an HMI project, you must edit the path to the project. To work with a single language, open the file in a text editor.



Edit the highlighted line to the correct path:

/HMI project name/HMI folder:server name

For example, with this project:



Enter:

/Organization/HMI:Org_HMI_Server

Save your changes.

In FactoryTalk View Studio, to import the updated translation, go to Tools > Languages.

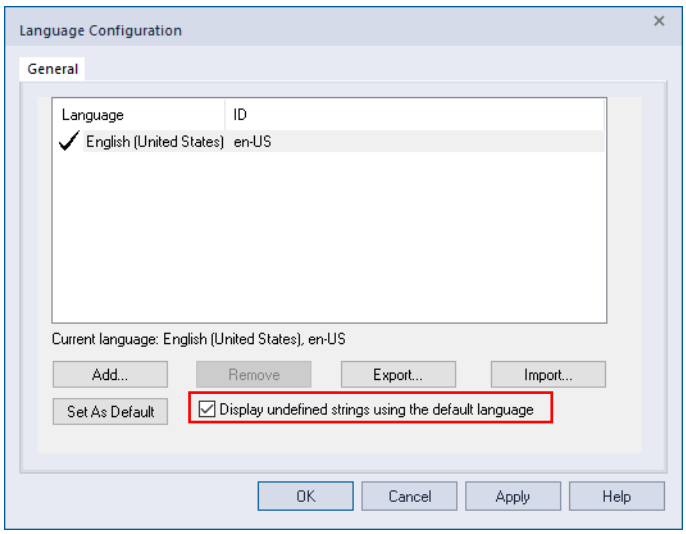
1. Click Add to add a language.
2. Select the language and click Apply
3. Click Import
   - Keep the default selection of Import strings from Unicode text files into en-US.
   - Click Next to select the updated file from the library.

It takes a few minutes to import the translated text.

FactoryTalk View SE Language Configuration

The Library of Process Objects 5.0 utilizes the feature Extended Tag Properties inside Studio 5000 Logix Designer. This allows localization of strings in the controller in the HMI Faceplates provided. When configuring languages (FactoryTalk View Studio - View Site Edition > Tools > Language Configuration) confirm the following checkbox is selected:

- Display undefined strings using the default language.

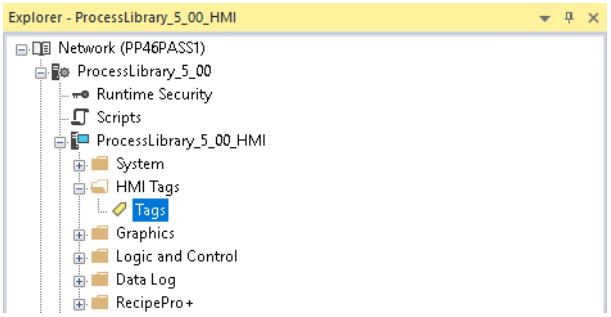


Help Files

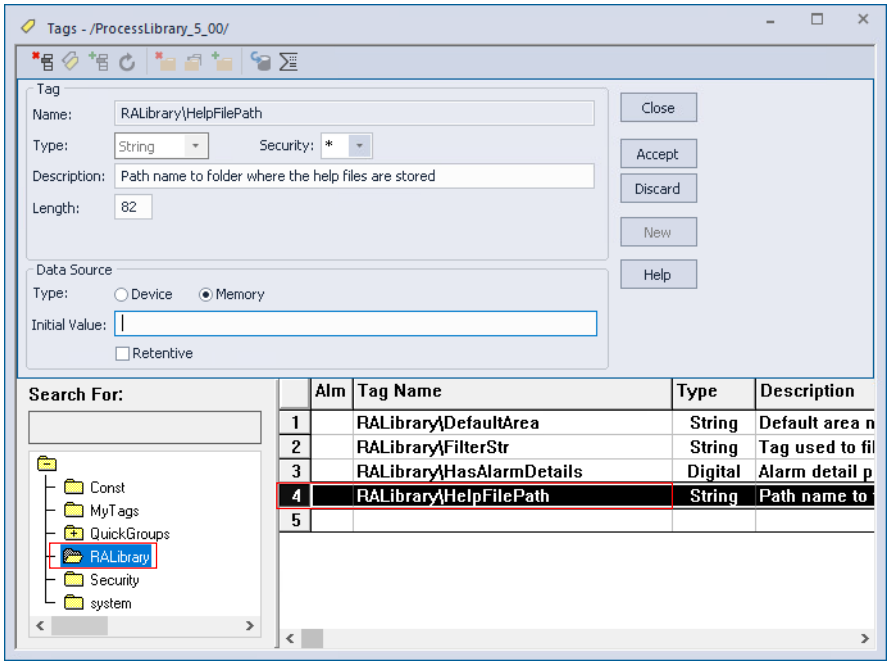
The help displays for the Library of Process Objects have been converted to PDF documents. The PDF documents can be displayed from the FactoryTalk View displays by clicking the Help

button. The help files are downloaded as part of the Library of Process Objects and are contained in the Documents folder.

1. Copy the Help files to a folder accessible by the FactoryTalk View clients.
   In this example we have copied the files to C:\Users\Rockwell\Desktop\HMI Help Files.
2. Open your project in FactoryTalk View Studio.
3. Open the Tags setting in the Folder Tree.

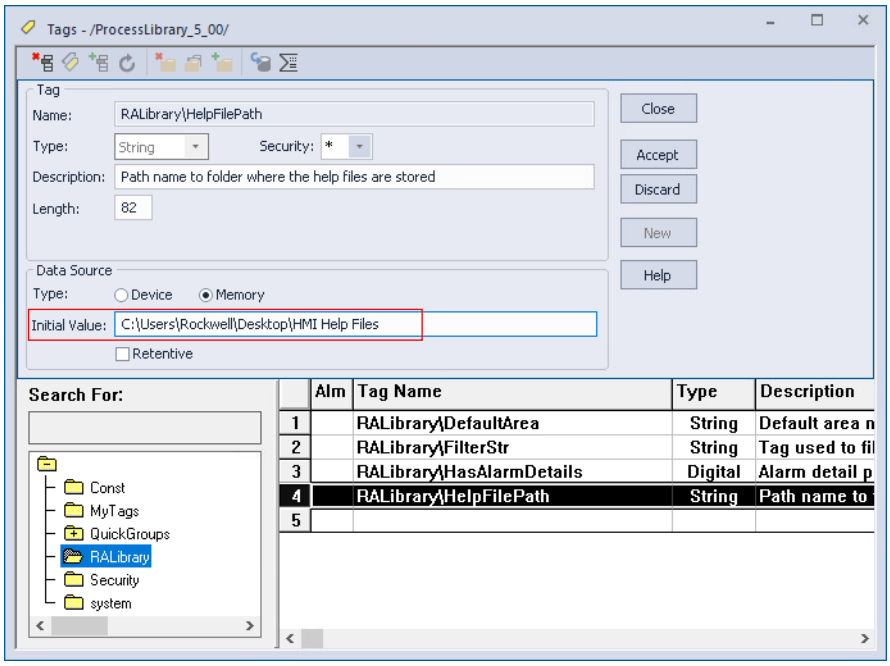


4. Select the RALibrary Folder and then Select RALibrary\HelpFilePath to access the settings for the Help Files.

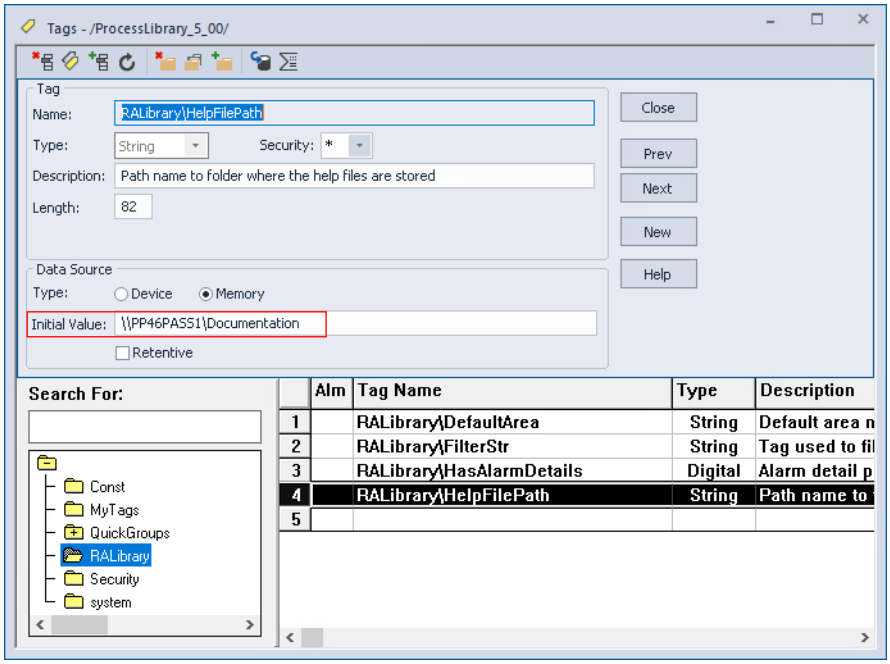


5. Enter the path to the Help Files into the Initial Data Source Field and Select Accept.

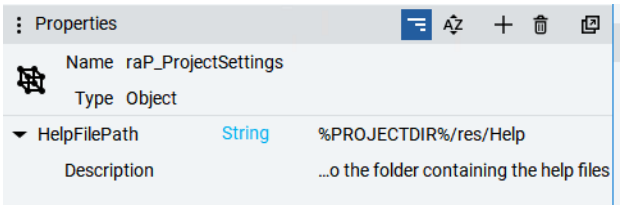
   Local Station:



   Distributed System Server:

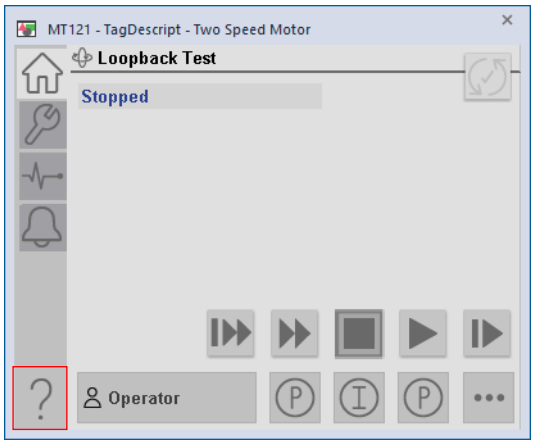


For FactoryTalk Optix, the help files are stored in the folder ProjectFiles/res/Help.



6. Close the settings display.
7. Restart FactoryTalk View Studio for the settings to take effect.

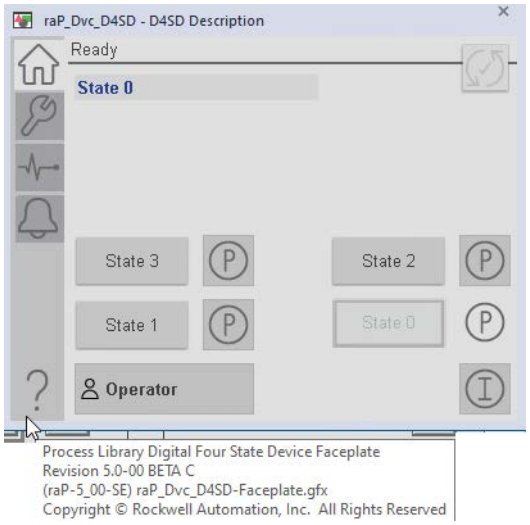
8. The Help Files can now be accessed using the Help button on the HMI Display.



Library Versions

Each library object has a revision x.yy.zz where: x is the Major Revision number, yy is the Minor Revision number, and zz is the Maintenance Release. Each release of the Process Library comes with release notes that describe the changes that were made since the last release.

Component	Example
The Add-On Instruction in Logix Designer application has revision information visible when the instruction is selected in the Controller Organizer.	Assets Add-On Instructions raP_Dvc_LgxCPU_5x80 Description: Processor Utilization (5380/5580, V33 and la... Revision: v5.0 .00 Release Revision Note: See Instruction Help for a summary of cha... Vendor: Rockwell Automation Data Type Size: 108 bytes Created: 10/20/2008 12:28:26 PM Created By: Not Available Edited: 9/17/2020 10:53:43 AM Edited By: PLANTPAXEWS\Rockwell Signature ID: <none>

Component	Example
The faceplate in FactoryTalk View software has revision information visible when the pointer is paused just inside the lower left corner of the faceplate.	
The faceplate for FactoryTalk Optix software has revision information visible at the bottom of the Help window.	

PlantPAx Process Library Migration Tool

This tool is used to migrate from previous Process Library versions to version 5.30. The PlantPAx Process Library Migration Tool provides the following:

- Updates Logix controller ACD files containing Rockwell Automation Process Library Add-On Instruction tags to corresponding Process Controller predefined process instruction tags and version 5.30 Add-On Instruction tags.
- Converts FactoryTalk View SE process graphics XML files containing global object references from previous Process Library versions to version 5.30 Process Library global objects.
- Migration of Process Library HMI libraries.
- Migration of GEMS Version 4.4 Add-On Instruction to corresponding Process Controller predefined process instruction tags and version 5.30 Add-On Instruction tags.

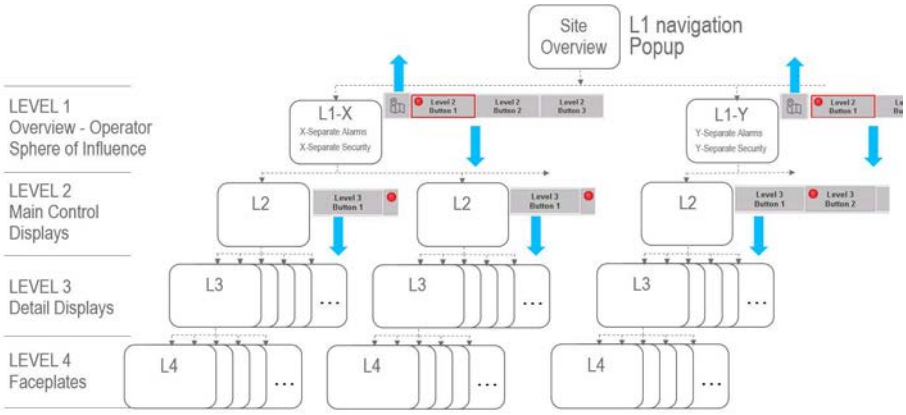
The tool reduces engineering time and migration errors. Use the tool to keep up with the latest Rockwell Automation software features and increase the lifecycle of the PlantPAx DCS.

Notes:

Graphic Framework Overview

It is important to organize an HMI application in a hierarchical way, to provide the operator and/or End User with a logical progression of complexity from main area overview down to detailed device information. ANSI/ISA-101.01-2015 outlines basic HMI design guidelines and recommends a progressive disclosure methodology with up to four levels of displays. The PlantPAx® Graphic Framework was created to assist the End User by providing a basic structure that can be used to follow the ANSI/ISA-101.01-2015 recommendations.

For more information on HMI philosophy, style guide contents, and the various display types/levels, see Rockwell Automation Process HMI Style Guide, PROCES-WP023-EN-P.



The Graphic Framework is composed of four main components, Header, Process Control Displays, Navigation, and Alarm Indication.

IMPORTANT The Graphic Framework was developed at the specific resolution of 1920x1080. The display files are a specific size and defined to open at a specific location. This should not be changed and could result in the Graphic Framework not functioning properly.

Header Display

The Header is a perpetual graphic display that is positioned at the top of each HMI client monitor to provide major navigation, annunciation, and status information for the process and the control system.

The Header is composed of several modular objects that can be selectively used to meet the needs of the End User. The following list indicates the available components in the Graphic Framework that can be used to create the Header display:

- Logo Object
- L1 Navigation Object
- Diagnostics Object
- Home Navigation Object
- Close Client Object
- Login / Logout Object
- Alarm Banner (Default sized Alarm Banner Object – 3 lines)

- Alarm Summary Navigation with Visual Alarm Indication Object
- Alarm Silence Object
- Date / Time Object
- Windows Navigation Objects
- Help Object
- Language Switching Objects
- Report Navigation Object
- Trend Navigation Object
- Documentations Navigation Object
- L2 Navigation Bar (required if not using the built-in navigation menu)

A separate header must be used for each L1 area, reflecting information within a specific operator's sphere of influence. The header will typically have a similar look and feel for each L1 area with different configurations to provide information only relevant to the operator of that L1 Area.

Process Control Displays

Process control displays are the main displays in the system that the operator interacts with. The Graphic Framework provides template displays, or default displays, that can be used to build the main graphics. These template displays can be duplicated for customization in each application. All default displays are sized the same and include different navigation and indication to allow operations to quickly assess the process status and take required actions.

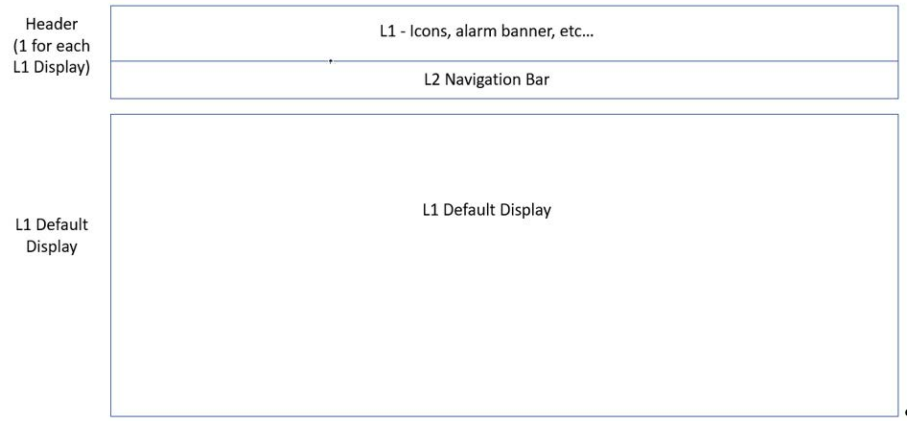
There are three process controls displays available as templates:

Display	Description
L1 Default Display Template	• This is an overview of a specific operator's sphere of influence (Overview Display) • Full graphic displays with L2 Navigation Bar visible • The first display that is populated when the operator refreshes the FactoryTalk® View SE client • Intended to be a high-level process area display typically consisting of key performance indications using trends and display objects (not just lists of numerical data)
L2 Default Display Template	• An operator's main control display designed to support typical operation modes often arranged like a process flow diagram (PFD). • Control for main operation variables and annunciation to prompt operator to access associated L3 display when necessary • Full graphic display with L2 and L3 Navigation Bar present • Typically, there are multiple L2 displays required to cover a specific operator's sphere of influence, which is represented by the L1 display.
L3 Default Display Template	• A more detailed display that is designed for troubleshooting abnormal scenarios. The L3 display design presents data that best matches to current task at hand. • Full graphic display with L2 and L3 Navigation Bar present. Simple L2 areas may not require an L3 display and therefore L3 Navigation Bar may not be required.
Display Template for use with Navigation Menu	• Full graphic display used to build L1, L2, and L3 level displays when used with the built-in FactoryTalk® View SE navigation menu. • Must configure the navigation menu in FactoryTalk View SE to use this display.

L4 displays provide finer detail and are opened as Faceplate or pop-up display from L2 and L3 displays. These would include PlantPAx standard faceplates or custom pop-up displays.

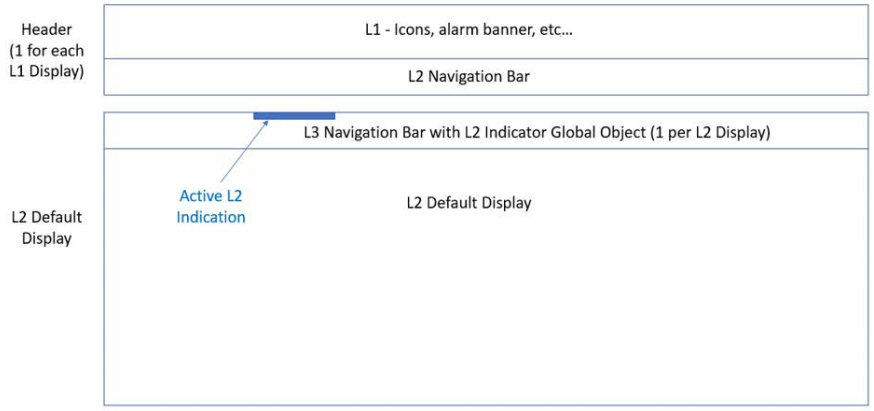
L1 Display

L1 Process Control Display is used as an overview for a single operator's sphere of influence. This is the first screen that the operator sees when the HMI client starts up and contains a high-level overview of the operator's sphere of influence as well as KPIs and indications. There will typically be one L1 Process Control Display for each L1 Area in the project. The display is typically designed to represent the various process units with key indications, trends, and rolled-up alarm status to help drive the operator to the appropriate L2 displays to address abnormal conditions.



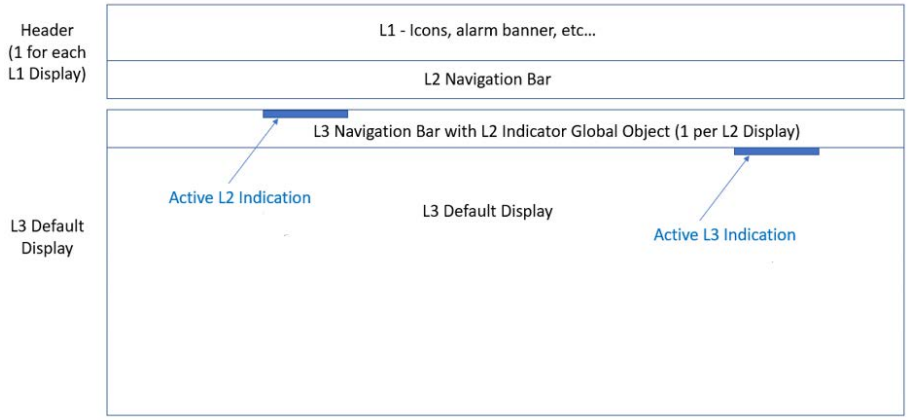
L2 Display

L2 Process Control Displays are used as the operators' main control screens. These displays provide access to the main operating parameters while concurrently providing annunciation when abnormal conditions exist. If necessary, the operator can access the associated and more detailed L3 displays to address the situation. The L2 display includes the L3 Navigation bar at the top with an indicator of the selected L2 display.



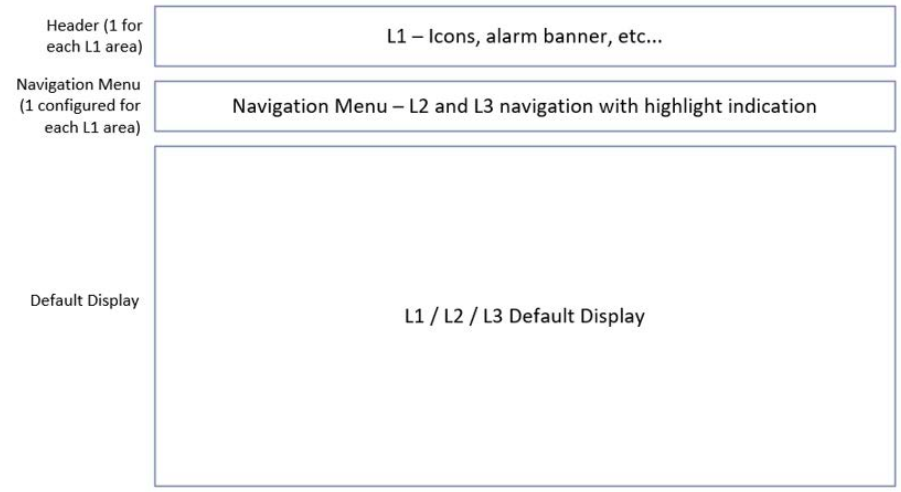
L3 Display

L3 Process Control Displays are used to access in-depth equipment details and diagnostics that may not be needed while the process is running normally. These displays are often similar to the traditional P&ID style of displays allowing the operator access to all control and monitoring information for that specific area of the plant. The L3 display includes the L3 Navigation bar at the top with an indicator of the selected display and an indicator of its associated L2 display.



Display with Navigation Menu

FactoryTalk View SE has a built-in navigation menu that can be configured for navigation to L2 and L3. When using the navigation menu, the functionality is the same, but the layout of navigation is slightly different. The default display can be used for L1, L2, or L3 detailed display.



Navigation

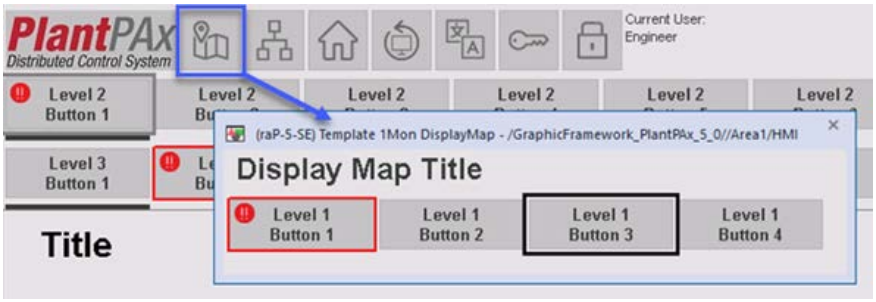
Starting with Graphic Framework version 1.00, which is independent from the PlantPAx Process Library, the graphic framework allows for selection of either the relatively new navigation menu system provided by FactoryTalk View or the legacy PlantPAx navigation system. There is no need to migrate from legacy to the navigation menus. For new projects either option can be considered. The navigation menus are not well suited for systems with multiple monitors or contextual navigation buttons. See Versioning and Design Considerations on page 58 for more information.

The Graphic Framework provides an intuitive and 'easy to configure' navigation strategy. Navigation among displays as part of the Graphic Framework can be configured and accessed from:

- L1 Navigation
- L2 Navigation
- L3 Navigation
- Navigation Menu
- Alarm Navigation
- Diagnostic Navigation
- Graphic Off-Screen Connectors

L1 Navigation

L1 Navigation allows operators to navigate to other areas of the facility. This moves the operator to another sphere of influence. The Display Map Button is used to open a pop-up display - the Display Map. This is the L1 Navigation display. This display can be expanded to include as many L1 areas as necessary for an application. Four buttons are provided by default.



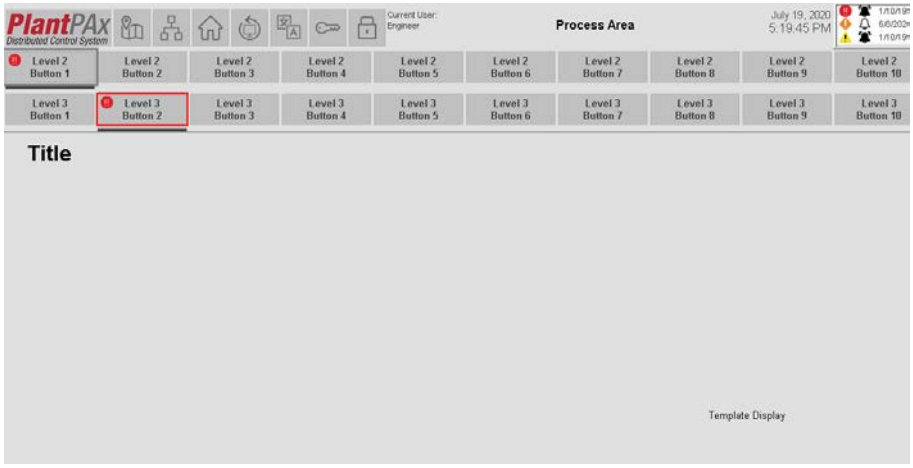
L2 Navigation

L2 Navigation is the first level of display access within a given L1 area. There is just one L2 Navigation bar used for each L1 area. The L2 Navigation bar resides within the header display and is always visible. The L2 Navigation Bar is composed of 16 buttons and can navigate to up to 16 different displays.

When the operator clicks the desired L2 button, that L2 display opens. On that L2 display, the associated L3 navigation bar opens. Each L2 button has alarm indications and these are rolled up from the L3 alarms. The L2 Navigation button text can be modified for each specific application.

L3 Navigation

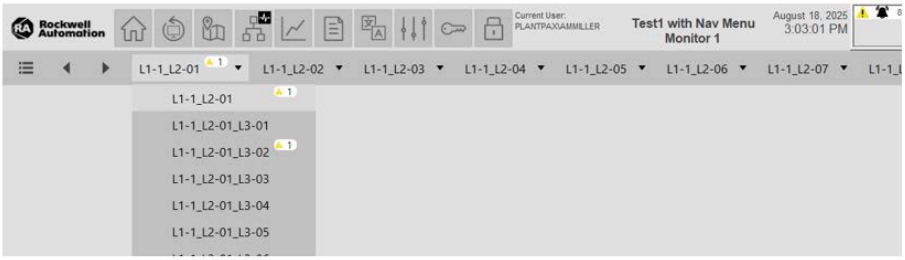
L3 Navigation is the second level of display access within a given L1 area. There are multiple L3 Navigation bars - one for each L2 button used. The L3 Navigation bar resides within the L2 and L3 Displays. Each L3 Navigation Bar is composed of 16 buttons and can navigate to up to 16 different displays. Included in the L3 Navigation bar is an indicator that shows which L2 and L3 area the operator is viewing.



When the operator clicks a desired L3 button, that L3 display opens. Each L3 button has alarm indications and these are rolled up into the associated L2 alarms. The L3 Navigation button text can be modified for each specific application.

Navigation Menu

The FactoryTalk View SE built-in Navigation Menu can be used instead of L2 and L3 Navigation bars. One navigation menu must be configured for each L1 area. As many buttons as needed can be configured (over 16), making the navigation menu a flexible option. The navigation menu is highlighted to indicate which display is currently active.



The navigation menu can be configured to include alarm indication and roll up to the associated L2 display. All text and appearance is modifiable from the navigation menu configuration.

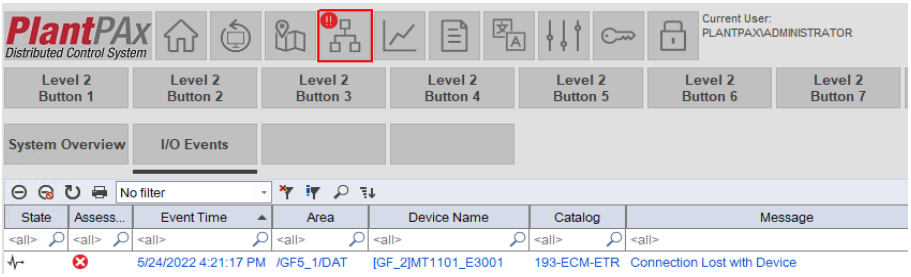
Alarm Navigation

Alarm information is accessed by pressing the Alarm Button on the header. This opens the Alarm Summary display. From the Alarm Summary display, other alarm information displays can be accessed, including the Alarm History, Alarm Shelved, and Alarm Explorer (with proper runtime security). There is a display that is associated with each of the four alarm buttons - see Global Objects for more information on Alarm Global Objects. See Displays for more information on template Alarm displays.

The Alarm Navigation has an indication below each button to show which alarm display the operator is viewing.

Diagnostic Navigation

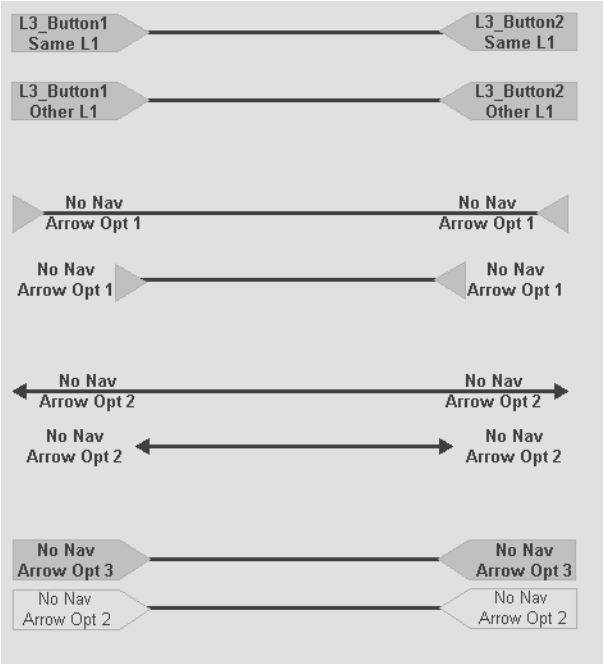
Diagnostic information is accessed by pressing one of the Diagnostic related buttons on the header (for example the System Status button). This opens the related diagnostics display with associated diagnostic navigation bar. Other diagnostic information displays can be accessed, including the System Status and Automatic Diagnostic Event Viewer. There is a display that is associated with each diagnostic type and two buttons available to be customized for additional diagnostics. See Global Objects and Displays for more information on Diagnostic Objects. See Displays for more information on template Diagnostic displays.



The Diagnostic Navigation display has an indication below each button to show which diagnostic display the operator is viewing.

Off-Screen Navigation

Graphic Off-Screen Connectors are used to supplement navigation for Operators to follow the process progression (to the left or to the right of the current display) on P&ID style screens. Various styles of off-screen navigation can be found in a Toolbox graphic.



There are three different off-screen navigation functionalities available.

Functionality	Description
Navigation to Same L1 area	This is used if the off-screen navigation is within the same L1 area. The button simply opens a new L2/L3 display within that L1 area.
Navigation to Other L1 area	This is used if the off-screen navigation is outside the current L1 area. The button needs to execute several commands to open the destination L1 area header and the desired L2/L3 display.
No Navigation (static)	The static off-screen connector does not navigate to any display. It is used as an indicator of a process inflow or outflow with no accompanying graphic - just a static indication. Various styles are offered in the toolbox.

Multi-Monitor Support

The Graphic Framework provides the structure and configuration that is needed to use multi-monitor applications during runtime. Applications utilizing the framework can be run on single, dual, or quad monitor hardware. Each monitor is formatted as described in the earlier section of this chapter - there is one header and a process display. The user can configure the header of each monitor as described in the earlier Header Display section, with one Header Display dedicated to each monitor for each L1 area. Alternatively, the user can configure all headers to be the same. See Multi-Monitor for more information on configuring an application for multi-monitor.

IMPORTANT	Multi-monitor functionality is not currently supported in the Graphic Framework when using the FactoryTalk View SE built-in Navigation Menu.

Alarm Indication

Alarm indication is embedded throughout the Graphic Framework. As mentioned in the L1/L2/L3 Navigation section, each navigation button has alarm indication available, with alarms rolled-up from L3 to L1. The Alarm button on the Header display also indicates to the operator if alarms are active in that L1 area. The following displays give the operator information on specific alarms and alarm configuration.

Display	Description
Alarm Banner	There is one Alarm Banner for each L1 area. The Alarm Banner, which resides on the Header display, will show up to three alarms. 
Alarm Summary	Each L1 area has a corresponding alarm summary. The purpose of the alarm summary is to indicate alarms within the L1 area (by severity and time) and provide the ability for the operator to interact with these alarms. Navigation to the alarm summary is accomplished by clicking the Alarm Summary Navigation button from the associated Header. The alarm summary must be configured to subscribe to alarms specific to the L1 area. Filters can be configured for each L2 alarm group for additional alarm functionality.
Alarm History	Alarm History display contains a configured Alarm and Event Log Viewer object that accesses the alarm and events historical data. Note: The alarm and event server must be configured to log the alarm data for this display to work properly. This display filters based on predefined filters.
Alarm Shelved	Alarm Shelved display contains an Alarm and Event Status Explorer object that is preconfigured to access alarm and event databases within the application with the status of "Shelved". This display can be further filtered based on alarm names. The shelved alarm display displays the alarm grouping tree to allow easy access to each alarm group.
Alarm Explorer	Alarm Explorer display contains an Alarm and Event Status Explorer object preconfigured to access A&E databases within the HMI application. This display can be further filtered based on alarm names. The alarm explorer displays the alarm grouping tree to allow easy access to each alarm group. The button to access this display has security that is built in. Only users with ability to enable/disable alarms can access this display.

Alarm Grouping and Supporting Logic

To create alarm groupings that align with the Navigation bars, additional upfront effort must be made in each controller to support this function. This effort requires using the Logical organizer in the controller files to align to the same hierarchy as in the graphical hierarchy.

Figure 1 - Pre-defined Graphical Layout

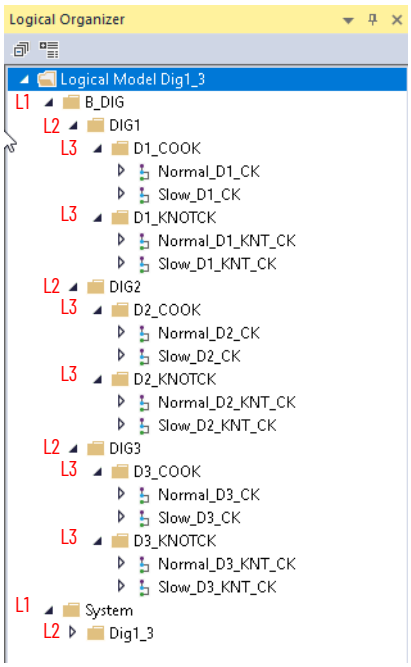
Displays		
L1	L2	L3
B_DIG	DIG1	D1_COOK
		D1_KNTCK
	DIG2	D2_COOK
		D2_KNTCK
	DIG3	D3_COOK
		D3_KNTCK
	DIG4	D4_COOK
		D4_KNTCK
	DIG5	D5_COOK
		D5_KNTCK
	DIG6	D6_COOK
		D6_KNTCK
	Charge	Chips
		Liquor
	RECOV	Condensate
		Turp
		Cooling
	ADV	Scheduler
		Kappa
		Steam
System	Dig1_3	n/a
	Dig4_6	n/a
	DigCom	n/a

The Logical Organizer folder structure must align with the predefined Graphical Hierarchy. That is, a folder in the Logical Organizer must be created for each process display that is used in the HMI. If multiple controllers are used within a single operator's sphere of influence, the same L1 - L2 - L3 architecture must be represented within each controller.

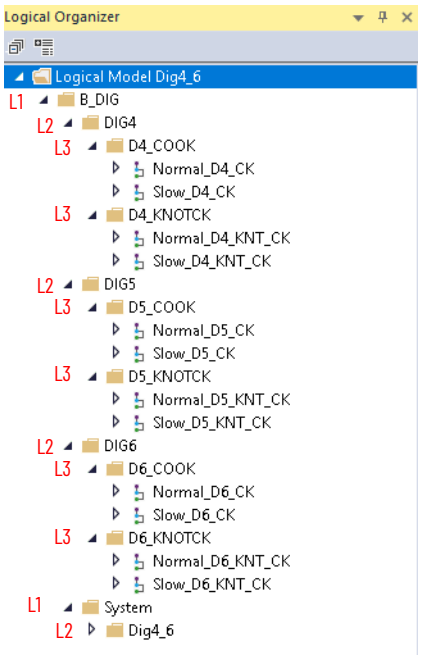
IMPORTANT If alarm grouping contains numbering, it is recommended to add a padded zero or else an unexpected alarm indication occurs. For example, if you have alarm groups D1_1, D1_10, and D1_11, the first alarm group should be modified to D1_01.

Figure 2 - Logical Organizer Aligned with Graphical Layout

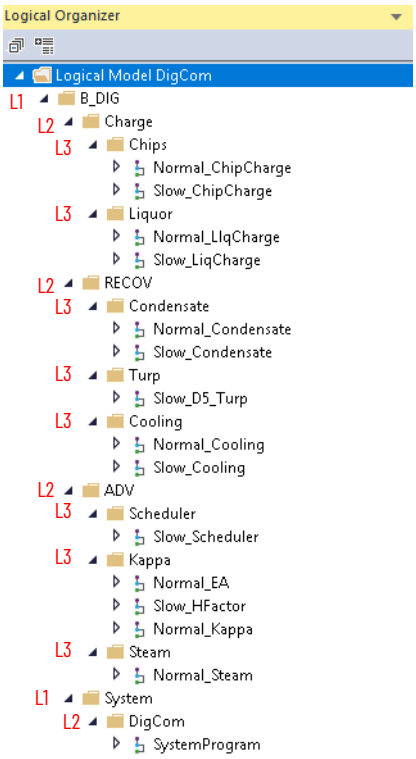
Controller: DH_1_3



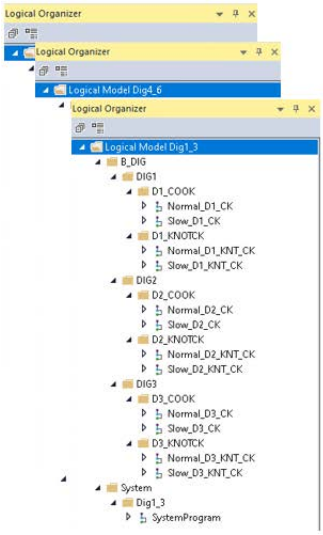
Controller: DH_4_6



Controller: DH_COM

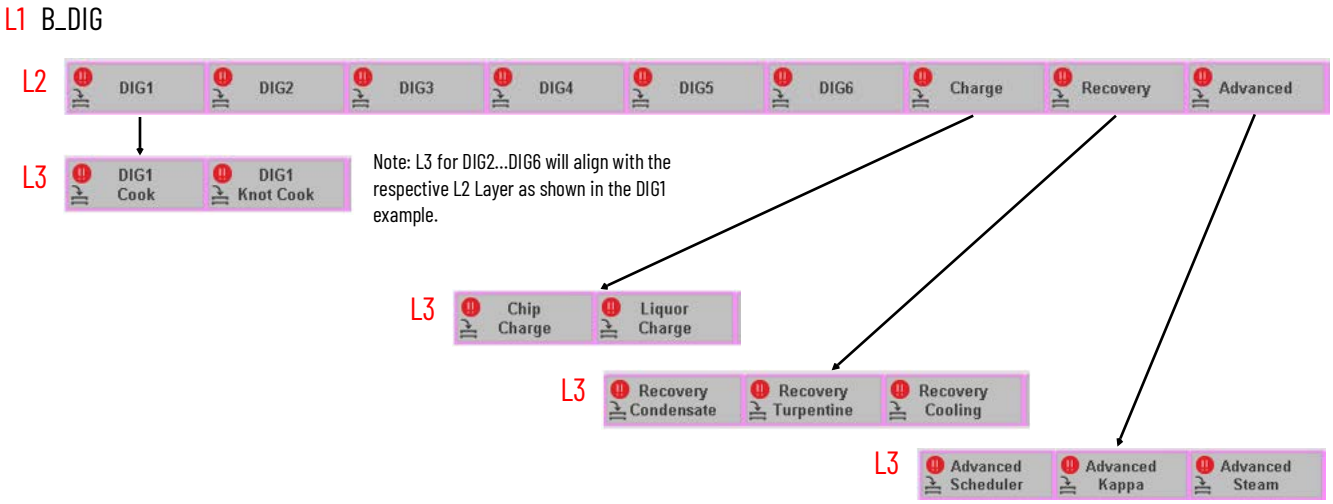


Once the folders are created, the PlantPAx Configuration Tool can be used to merge the alarm groups appropriately so that the process alarm indication displays are control equipment agnostic. In addition, if a single controller contains logic that is used by multiple operators, the folder structures of each area must be created in the Logical organizer to represent the multiple L1 hierarchies.



Alarm Groupings			Controller Source
L1	L2	L3	
B_DIG	DIG1	D1_COOK	Dig1_3
		D1_KNTCK	Dig1_3
	DIG2	D2_COOK	Dig1_3
		D2_KNTCK	Dig1_3
	DIG3	D3_COOK	Dig1_3
		D3_KNTCK	Dig1_3
	DIG4	D4_COOK	Dig4_6
		D4_KNTCK	Dig4_6
	DIG5	D5_COOK	Dig4_6
		D5_KNTCK	Dig4_6
	DIG6	D6_COOK	Dig4_6
		D6_KNTCK	Dig4_6
	Charge	Chips	DigCom
		Liquor	DigCom
	RECOV	Condensate	DigCom
		Turp	DigCom
		Cooling	DigCom
	ADV	Scheduler	DigCom
		Kappa	DigCom
		Steam	DigCom
System	Dig1_3	n/a	Dig1_3
	Dig4_6	n/a	Dig4_6
	DigCom	n/a	DigCom

The following navigation bars must be configured to align with the information in Figure 1. Alarm groupings enable the appropriate alarm roll-ups to the navigation buttons.



The alarm grouping configuration in the Logical Organizer should then be reflected on the L2 / L3 navigation for button naming and alarm breadcrumb (alarm groups).

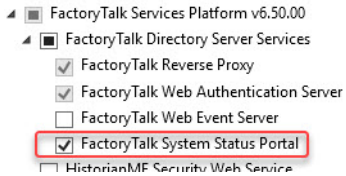
Server Status Monitoring

Server status monitoring is an important diagnostic tool that is helpful for maintenance and troubleshooting any system. There are two main options available with FactoryTalk View SE: System Status Portal and FactoryTalk Resource and Status Server.

IMPORTANT	Both server monitoring options are compatible with the Graphic Framework [v1.00]. Only the System Status Portal is available to Graphic Framework [Legacy].

System Status Portal

The system status portal is installed by default when you install any product that has FactoryTalk Services Platform v6.30 and later. Make sure that the box is checked to include the system status portal with installation.



The system status portal can be accessed from any workstation in the system joined to the FactoryTalk Directory. The portal is viewable in a web browser or web browser object in FactoryTalk View runtime, by typing https://IPAddress/FTSystemStatus, where IPAddress is the IP address of the FactoryTalk Directory server computer. See FactoryTalk Services Platform Help for more information.

The system status portal is limited to displaying the status of HMI servers, data servers (including FactoryTalk Linx, OPC UA, and OPC-DA servers), and Factory Alarm and Events servers.

FactoryTalk Resource and Status Server

FactoryTalk Resource and Status Server can be used on any system with FactoryTalk Services Platform v6.50 and later. Included in the Graphic Framework [version 1.00] are graphic symbols and faceplates for common devices and infrastructure to monitor in a system.

See FactoryTalk Resource & Status Server Configuration on page 113 for more information on configuration recommendations.

With the FactoryTalk Resource and Status server, each resource that is monitored becomes programmatically accessible in FactoryTalk View, where additional functions can be added on such as conditional alarming or tag historization. Additionally, the server can monitor network connection to devices that are not part of the FactoryTalk Directory (i.e vCenter, ESXi, iDRAC, switches, etc) which makes it a powerful tool for gathering diagnostics of an entire system.

Network Status	Workstation Status	Processes Status
Monitor status from workstation in the FactoryTalk Directory with FTRSS installed to any device on the network.	Monitor CPU, memory, storage space, NIC status, on any workstation on the FactoryTalk Directory with FTRSS installed.	Monitor any process / task available in Task Manager on any workstation on the FactoryTalk Directory with FTRSS installed

Versioning and Design Considerations

The Graphic Framework has evolved with iterations of the Process Library and PlantPAx System releases. The reason for this change is that it modularizes the key components of the system, reducing the time for updates being accessible to users and making it easier to adopt new product capabilities (FactoryTalk View and FactoryTalk® Optix™)

The following is a brief explanation of the Graphic Framework release history and versioning:

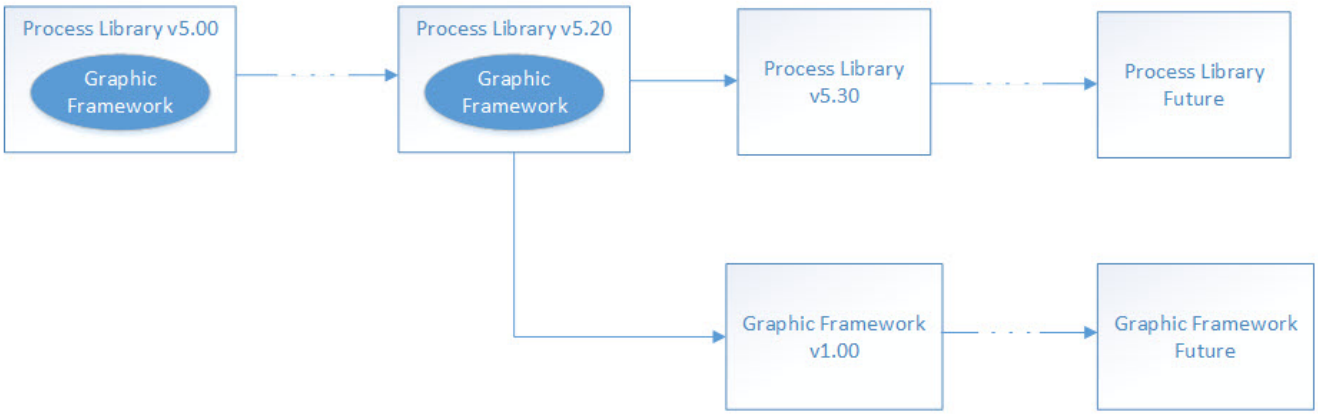
PlantPAx v5.30 release and earlier (Process Library v5.30 and earlier).

- The Graphic Framework files were included in the Process Library download from PCDC under "Process Library".
- These releases are referred to as the "Graphic Framework [Legacy]"

PlantPAx v5.40 release and later.

- The Graphic Framework files are now included as a standalone download from PCDC under "Graphic Framework".
- These releases are referred to as the "Graphic Framework [v1.00]"

IMPORTANT New major and minor releases of the Process Library (for example, Process Library 5.30 and later), will no longer include the Graphic Framework [Legacy] content.



Graphic Framework [Legacy]

Pros	Cons
• Can be used with single, dual, or quad monitor clients. • Bulk Configuration with the Graphic Framework tool. • Alarm indication and rollup. • Button indication of which display is selected.	• No tag search or built-in forward/backward/historical navigation • Limited to 16 buttons per L2 and 16 buttons per each L3. • Configuration is time-consuming and an opportunity for mistakes if created without the use of the Graphic Framework Tool. • It can be cumbersome if trying to add a button into the middle of an L2/L3 nav bar. • Lots of clicks to set up each button command on the L2/L3/L1 buttons.

Graphic Framework [v1.00]

Added support of FactoryTalk View SE v15 Navigation Menu.

You have a choice to use the legacy Graphic Framework templates or use the built-in Navigation Menu of FactoryTalk View SE.

The following table is for the Navigation Menu of FactoryTalk View SE.

Pros	Cons
• Uses the Built-in Navigation Menu of FactoryTalk View SE. • Configurable all in one place. • Ability for built-in tag search. • Ability for built-in forward, backward, and historical navigation. • Unlimited number of displays per L2 and L3 area. • Simple workflow to move, add, and reconfigure L2 and L3 areas. • Easily configure display commands directly in the navigation menu configuration. • Built in alarm source and automatic diagnostic indication and rollup. • Ability to use icons instead of text on dropdown and display navigation.	• Currently only supported for single monitor with the Graphic Framework. • Limited Bulk Configuration – Graphic Framework Tool can generate the displays, but the Navigation Menu configuration is separate. • In FactoryTalk View SE v15, if more L2 folder areas are configured than fit on a single monitor, the alarm indication is not visible for those L2 areas (The issue is fixed in v16).

Notes:

Configure the Graphic Framework

The Graphic Framework is a flexible tool that is highly configurable to each specific user. This chapter outlines the options for how to get started configuring the Graphic Framework and how to configure a project. As noted in the previous chapter, there are two varieties of the Graphic Framework that are currently supported: Graphic Framework [Legacy] and Graphic Framework [version 1.00]. See Versioning and Design Considerations for more detail.

The following table shows what topics apply to which design path.

Graphic Framework [Legacy]	Graphic Framework [version 1.00] with Legacy navigation	Graphic Framework [version 1.00] with Navigation Menu
Graphic Framework [Legacy] Configuration	Graphic Framework [v1.00] Configuration	
PlantPAx Process Library Dependencies Build Your PlantPAx HMI Application Recommended Application Naming Structure Global Objects: APP - Administrative Objects Global Objects: APP - Alarm Objects Global Objects: APP - Diagnostic Objects Global Objects: APP - Header Objects		
–	Global Objects: APP - Resource and Status Objects (raC-1_00-SE)	
Global Objects: Template Custom Objects Global Objects: TTemplate L1 Navigation		
Global Objects: Template L2 L3 Navigation		Global Objects: Template L2 L3 Navigation (Only Diagnostic and Alarm navigation needed)
Displays		
Multi-Monitor		Multi-Monitor: Create HMI Tags for Multi-Monitor and Repaint
–		Navigation Menu
–	FactoryTalk Resource & Status Server Configuration	
Macros Client File Setup (.CLI)		

The following table defines common terms that are used in the configuration.

Term	Description
Template	The term "Template" within a filename indicates that the file should be duplicated when used in the application. The duplicated file should be renamed to a title that is meaningful for the specific area or sub-subarea of the facility of your application. The original template file is to be used as a starting point for multiple files in the application and should not be modified.
App	When a file in the application contains the term "APP", the objects in these files can be used directly out of this file - the file name does not need to be duplicated or renamed.

Graphic Framework Builder Tool

The Graphic Framework Builder Tool can be used to quickly develop FactoryTalk® View SE files to import into a base application. This tool can be used to automate most of the steps explained in this chapter. For more details on the Graphic Framework Tool and how to use it, See the Product Compatibility and Download Center and search "PlantPAx Tools".

Graphic Framework [Legacy] Configuration

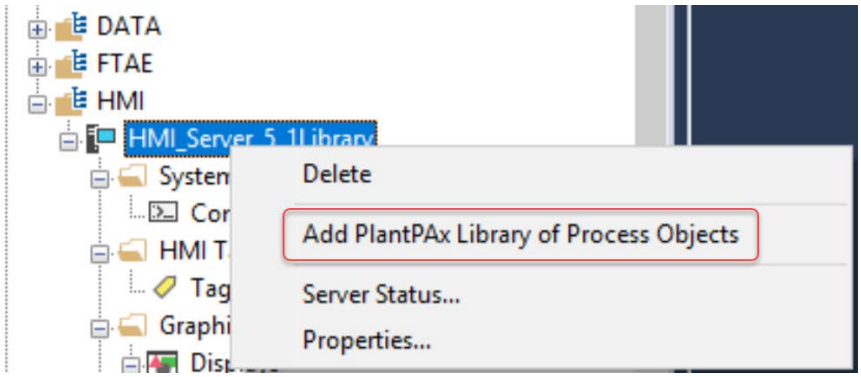
For PlantPAx® Process Library versions 5.20 and earlier, the Process Library download provides the following files to use as a starting point for the Graphic Framework. Templates are provided both with and without the PlantPAx Process Object library faceplates included.

- FTVSE_{version}_Template_{version}.APB (for example, FTVSE_13_0_Template_5_10_00.APB)
- FTVSE_{version}_TemplateWLibrary_{version}.APB (for example, FTVSE_12_0_TemplateWLibrary_5_00_00.APB)
- FTVSE_{version}_Template_{version}.zip (for example, FTVSE_12_0_Template_5_00_00.zip)
- FTVSE_{version}_TemplateWLibrary_{version}.zip (for example, FTVSE_13_0_TemplateWLibrary_5_10_00.zip)

The Graphic Framework [Legacy] can be used in one of two ways from the template files in the Process Library download:

- Restore the provided Local Station project templates (.APB) using the FactoryTalk View SE Application Manager.
- Create your own project as a Distributed or Network Station application and import the HMI server or individual files as needed.

If using FactoryTalk View SE version 13 and later, there is a third option to use the Graphic Framework [Legacy]. After creating an HMI server, right-click on the server and select "Add PlantPAx Library of Process Objects". For this option, no download from PCDC is necessary as one specific version of the Process Library is built into FactoryTalk View.

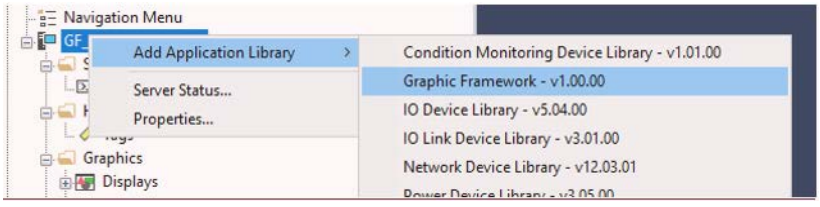


Graphic Framework [v1.00] Configuration

The Graphic Framework [v1.00] is compatible with FactoryTalk View SE v15 and later. The download provides all required content and a Windows® command script file to move the content to the FactoryTalk View SE Application Library folder.

The Graphic Framework [v1.00] can be used in one of two ways to create a starting point for a new application:

- Use the Graphic Framework builder tool to create your application.
- Run the provided CMD file to move the content to the Application Library folder in FactoryTalk View SE. Then create a project, right-click the HMI server and "Add .Application Library".



PlantPAx Process Library Dependencies

The Graphic Framework [Legacy] is dependent on the following files from the Library of Process Objects:

- Display Files (.gfx)
  - (raP-5_30-SE) raP_Opr_OrgView-TreeView
  - (raP-5_30-SE) raP_Opr_OrgView-Select
  - (raP-5_30-SE) raP_Opr_OrgView-Config
  - (raP-5_30-SE) raP_UDT_Opr_Bus-Advanced
  - (raP-5_30-SE) raP_UDT_Opr_Bus-Faceplate
- Global Object Files (.ggfx)
  - (raP-5_30-SE) Toolbox - Common Adv Objects
  - (raP-5_30-SE) Toolbox - Organization Objects
  - (raSDK-1-SE) Toolbox - Common Objects
- Macros (.mcr)
  - DefineShowHWTreeCmd
  - DefineShowTreeCmd
  - ShowTreeForObject
  - ToggleWithRemark
  - NavToFaceplate
  - NavToDisplay
- All Images files
- HMI Tag import

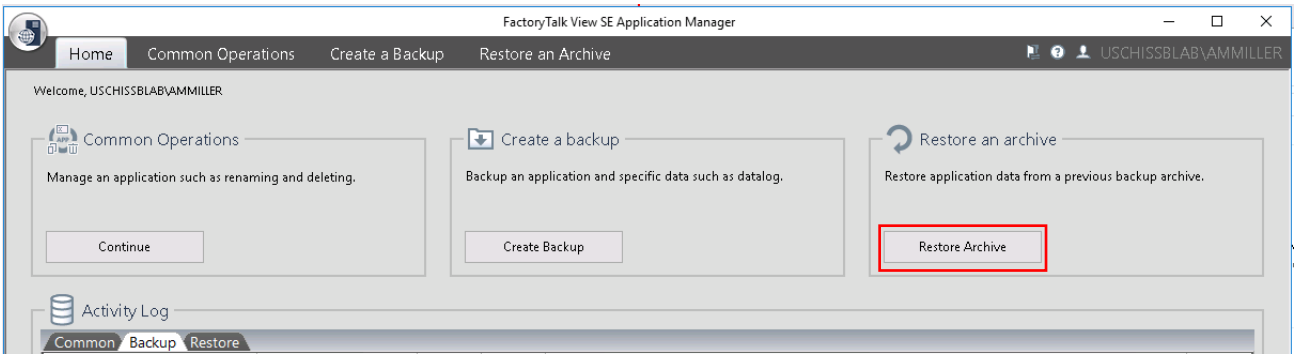
The Graphic Framework [v1.00] and later is dependent on the following files from the Library of Process Objects:

- Global Object Files (.ggfx)
  - (raSDK-1-SE) Toolbox - Common Objects
- All Images files
- HMI Tag import

Build Your PlantPAx HMI Application

Local Station Applications

1. Go to FactoryTalk® View SE Application Manager > Local Station and select Restore Archive.



2. Browse to the APB file.
3. Name the new application and select Restore.

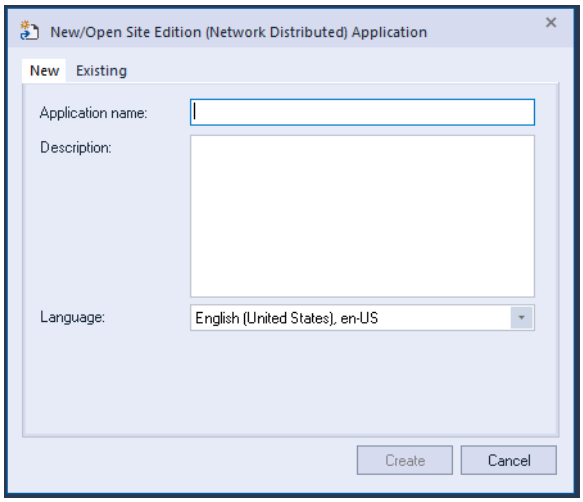
You can now open FactoryTalk® View SE Local Station and build out the application using the PlantPAx Graphic Framework.

Distributed or Network Station Applications

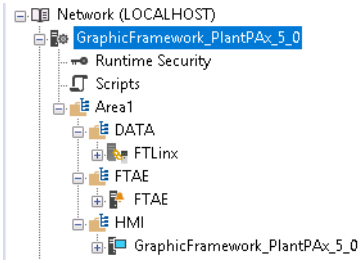
The HMI Server backup can be used for Distributed or Network Station applications. The following assumes that the server system is configured correctly and to PlantPAx recommendations. The following also assumes that the FactoryTalk® Directory is configured and all applicable servers are joined to the directory.

IMPORTANT This should be used as a rough guide only. See PlantPAx Distributed Control System Configuration and Implementation User Manual, publication PROCES-UM100 and FactoryTalk View documentation for best practice and proper system configuration.

1. Go to FactoryTalk View Studio and select either Distributed or Network Station.
2. Create an application.

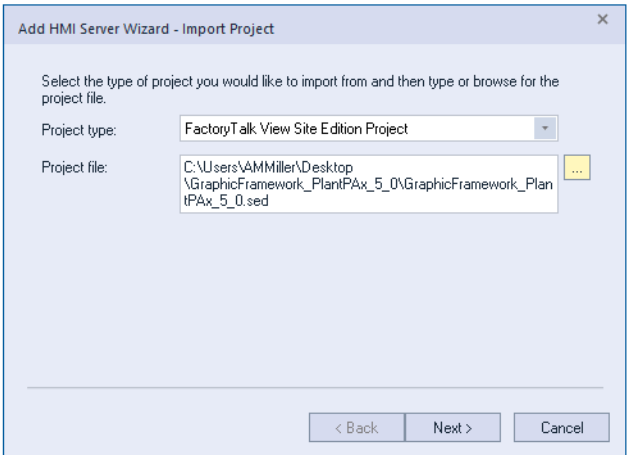
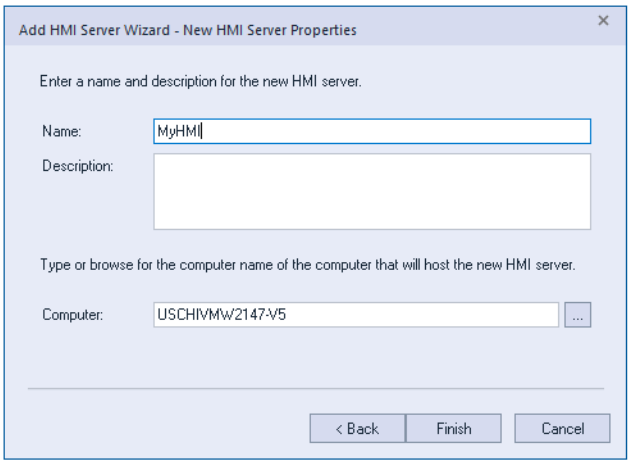


3. Build out the Area folder structure. Place only one server in each area folder



4. Extract the graphic framework - either with or without the Process Library. (Found in the library download at \Process Library\Templates\FactoryTalk View SE).

5. Right-click the HMI area folder. Select Add New Server > HMI Server. The Add HMI Server Wizard opens. Select Import a project and click Next.

On this Page	Action
Warning pop-up	Select OK
Import Project	• Select FactoryTalk View Site Edition Project. • Navigate to the HMI server backup that was extracted in step 4. 
New HMI Server Properties	• Name the HMI server • Select the computer that hosts the new HMI server 

The HMI server takes a few minutes to import. Once the import is complete, the application is ready to build out with the Graphic Framework.

Recommended Application Naming Structure

The following is a table of recommended naming structures for files that are provided in the Graphic Framework. The files with a suggested naming structure needs to be duplicated from the original file and renamed with the structure specific for your project.

Template Display Name	Suggested Name Structure	Example:
Template Display Map	[App_Name]_DisplayMap	ABC-Chem_DisplayMap
Template Diagnostic-IOEvents	[App_Name]_Diagnostic-IOEvents	ABC-Chem_Diagnostic-IOEvents
Template Diagnostic-Summary	[App_Name]_Diagnostic-Summary	ABC-Chem_Diagnostic-Summary
Template Diagnostic-SysSts	[App_Name]_Diagnostic-SysSts	ABC-Chem_Diagnostic-SysSts
Template Language-Select	[App_Name]_Language-Select	ABC-Chem_Language-Select
(raP-5_30-SE) Common-Redirect-to-4_10(1)	N/A	Use file as is
(raP-5_30-SE) Common-Redirect-to-5_00(1)	N/A	Use file as is
Template Reports	[App_Name]_Reports	ABC-Chem_Reports
Template Trend_Full	[App_Name]_Trend_Full	ABC-Chem_Trend_Full
Template Trend_Popup	[App_Name]_Trend_Popup	ABC-Chem_Trend_Popup
Template Admin-SysSecurity	[App_Name]_Admin-SysSecurity	ABC-Chem_Admin-SysSecurity
Template Header Mon1	[L1_Name]Header_Mon1	Mixing_Header_Mon1
Template Header Mon2	[L1_Name]_Header_Mon2	Mixing_Header_Mon2
Template Header Mon3	[L1_Name]_Header_Mon3	Mixing_Header_Mon3
Template Header Mon4	[L1_Name]_Header_Mon4	Mixing_Header_Mon4
(raC-1_00-SE) Template Header Nav Menu(2)	[L1_Name]_Header_Mon1	Mixing_Header_Mon1
Template Display L1	[L1_Name]	Mixing
Template Display L2	[L1_Name]_[L2_Name]	Mixing_IngredAdd
Template Display L2 no L3	[L1_Name]_[L2_Name]	Mixing_Agitate
Template Display L3	[L1_Name]_[L2_Name]_[L3_Name]	Mixing_IngredAdd_Weigh
(raC-1_00-SE) Template Display Nav Menu(2)	[L1_Name]_[L2_Name]_[L3_Name]	Mixing_IngredAdd_Weigh
Template Alarm-Explorer	[L1_Name]_Alarm-Explorer	Mixing_Alarm-Explorer
Template Alarm-History	[L1_Name]_Alarm-History	Mixing_Alarm-History
Template Alarm-Shelved	[L1_Name]_Alarm-Shelved	Mixing_Alarm-Shelved
Template Alarm-Summary	[L1_Name]_Alarm-Summary	Mixing_Alarm-Summary
(raC-1_00-SE) RSSNetworkDevice-Faceplate(2)	N/A	Use file as is
(raC-1_00-SE) RSSProcessMonitor-Faceplate(2)	N/A	Use file as is
(raC-1_00-SE) RSSWorkstation-Faceplate(2)	N/A	Use file as is

(1) Only available in Graphic Framework [Legacy].

(2) Only available in Graphic Framework [v1.00] and later.

Global Object Files	Suggested Name Structure	Example:
APP - Administrative Objects	N/A	Use file as is
APP - Alarm Objects	N/A	Use file as is
APP - Diagnostic Objects	N/A	Use file as is
APP - Header Objects	N/A	Use file as is
(raC-1_00-SE) APP – Resource and Status Objects(1)	N/A	Use file as is
Template Custom Objects	[App_Name]_CustomObjects	ABC-Chem_CustomObjects
Template L1 Navigation	[App_Name]_L1Navigation	ABC-Chem_L1Navigation
Template L2 L3 Navigation	[L1_Name]_L2L3Navigation	Mixing_L2L3Navigation

(1) Only available in Graphic Framework [v1.00] and later.

Macro File	Suggested Name Structure	Example:
Template_ClientStartup_SingleMon	[L1_Name]_ClientStartup_SingleMon	Mixing_ClientStartup_SingleMon
Template_ClientStartup_DualMon	[L1_Name]_ClientStartup_DualMon	Mixing_ClientStartup_DualMon
Template_ClientStartup_QuadMon	[L1_Name]_ClientStartup_QuadMon	Mixing_ClientStartup_QuadMon
Template_NavMenu_ClientStartup_SingleMon(1)	[L1_Name]_ClientStartup_SingleMon	Mixing_ClientStartup_SingleMon
Template_Repaint_SingleMon	[L1_Name]_Repaint_SingleMon	Mixing_Repaint_SingleMon
Template_Repaint_DualMon	[L1_Name]_Repaint_DualMon	Mixing_Repaint_DualMon

Macro File	Suggested Name Structure	Example:
Template_Repaint_QuadMon	[L1_Name]_Repaint_QuadMon	Mixing_Repaint_QuadMon
Template_NavMenu_Repaint_SingleMon(1)	[L1_Name]_Repaint_SingleMon	Mixing_Repaint_SingleMon
SetRepaint	N/A	Use file as is
NavToDisplay with mixed library	Optional macros - use only for applications with both Process Library 4.10 and Process Library 5.00 or later. See Macros section for detail.	
NavToFaceplate with mixed library		

(1) Only available in Graphic Framework [v1.00] and later.

Global Objects

The following section outlines each of the global object files available in the Graphic Framework and how each object should be used and configured. The purpose of this section is to provide the application developer with details on each global object and how to configure them. Not all global objects are required to be used in the application.

APP - Administrative Objects

The following objects are used for administrative control.

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Close Client		The purpose of the Close Client object is to shut down the client.	No configuration required. This object can be placed on the Header or on a separate administrator display.		

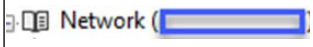
APP - Alarm Objects

The following objects are used for alarm navigation and annunciation.

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Alarm Summary Navigation		The purpose of the Alarm Summary Navigation object is to visually alert operators of current active alarms in their L1 process area and to provide navigation to the Alarm Summary. This object navigates to another Alarm Summary screen in each L1 area. This button should already be populated on the template Header display. The global object parameter values must be updated on the Header display.	101	Alarm Summary Display Name	Enter full Alarm Summary display name
			102	Alarm Group Name (Level 1)	Enter the L1 area alarm group name (for indication)
Alarm Silence Button		The purpose of the Alarm Silence Button object is to silence any active audible alarms that are assigned to that L1 area for that specific client. This button should already be populated on the Header display. The global object parameter values must be updated on the Header display.	101	Alarm and Event Banner Display Name	Enter the associated L1 Header display name that contains the alarm banner. "Invoke #101.FactoryTalkAlarmandEventBanner.SilenceAll"
Alarm Group Annunciation		The alarm annunciation objects are available for L1, L2, or L3 alarm groups. These annunciation objects are built into template objects for L1, L2, and L3 navigation objects, but can be added to additional buttons if desired. There are also larger objects available for an L1 Overview display.	101	Alarm Group Name (Level 1)	L1 Group name in FTAE and/or FTLinx (required for L1, L2, and L3 annunciation objects)
			102	Alarm SubGroup Name (Level 2)	L2 Group name in FTAE and/or FTLinx (required for L2 and L3 annunciation objects)
			103	Alarm SubSubGroup Name (Level 3)	L3 Group name in FTAE and/or FTLinx (required for L3 annunciation objects)

APP - Diagnostic Objects

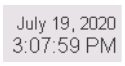
The following objects are used for hardware and software diagnostics as well as L4 trend display access.

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Software Tree View Navigation[1]		The purpose of the Software Organization Tree View button is to view the entire organization tree view for each controller in one central location. This button is used with organizational bus instructions raP_Opr_OrgView and raP_Opr_OrgView. See [appropriate section name] for more information on the configuration of the organizational bus.	101	SW Tree Identification	Enter a text string, not a tag. This displays on the button.
			102	Processor Shortcut Name	Enter the shortcut name where tree view is located. Include the area name in the parameter entry (for example, '[MyCLX]' or '/Area1/SubArea1::[MyCLX]')
Hardware Tree View Navigation[1]		The purpose of the Hardware Organization Tree View button is to view the hardware organization tree view for each controller. This button is used with organizational bus instructions raP_Opr_OrgView and raP_Opr_OrgView as well as raP_Dvc_LgxModuleSts. See [appropriate section name] for more information on configuration of the organizational bus.	101	HW Tree Identification	Enter a text string, not a tag. This displays on the button.
			102	Processor Shortcut Name	Enter the shortcut name where tree view is located. Include the area name in the parameter entry (for example, '[MyCLX]' or '/Area1/SubArea1::[MyCLX]')
Pop-up Display Trend Navigation		The trend pop-up button is intended to be placed throughout L1, L2, or L3 process displays to display TrendPro templates that are specific to the user's process.	101	Trend pop-up Display Name	Full name of the pop-up trend name. This should be (raP-5_30-SE) Template Trend_Popup or a display that is created from duplicating this display.
			102	Trend Template (Optional)	Name of the TrendPro template that should be invoked when the pop-up display opens. This can be left blank if no TrendPro templates are created yet.
			103	HMI Server Name	The exact name of the HMI server. The name next to this icon in your application: 
			104	PASS Server Name	The exact name of the server that is hosting your HMI Server (usually the PASS). You can find this name by examining the top of the application tree: 
Tree View Alarm Annunciation		The alarm annunciation object for the tree view is the same as for an L2 alarm annunciation object (see previous section). It is recommended to use with the Hardware Tree View button to annunciate any hardware-related alarms.	101	Alarm Group Name (Level 1) (for example, 'System')	L1 Group name in FTAE and/or FTLinx
			102	Alarm SubGroup Name (Level 2) (for example, 'CLX1')	L2 Group name in FTAE and/or FTLinx

(1) Only available in Graphic Framework [Legacy].

APP - Header Objects

The following objects are recommended to be placed on the Header display and provide information and specific navigation.

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
PlantPAx Logo Rockwell Automation logo		The logo object is pre-built using the PlantPAx logo or the Rockwell Automation logo. The logo object is populated in the Header display by default but can be removed to free up space on the Header. If users prefer to add their own logo, See Template Custom Objects for more information.	No configuration required.		
Time Date		The Time-Date object indicates the current time and date. This object is populated in the Header display by default.	No configuration required.		
System Status		The System Status object is used for navigation to the control system status display. The system status screen is a custom display that is developed to show diagnostics and hardware information. See Displays for more information on the template display. Use (raP-5_30-SE) Template Diagnostic-Summary as a starting point. There is an optional System Status breadcrumb that can be added to the System Status button (see Displays for Template Toolbox display). There is also an L1 alarm breadcrumb that could be used for this button (see APP - Alarm Objects for Alarm Group annunciation). Alarm bread crumb objects - L1 for the System Status header button and L2 for each individual Hardware Tree button - are provided in the alarm global object file that fits on top of the buttons for System Status. See APP - Alarm Objects for alarm group annunciation details. It is recommended that an L1 alarm group be for overall system diagnostics and that L2 subgroups be created for each controller and hardware under that controller.	101	System Status Display Name	Enter the whole display name into the parameter. Display "#101"
			The system status breadcrumb is configured as follows: Replace "[MyCLX]" with the shortcut defined for your application. If you have multiple processors, duplicate the function for each processor and use a logical "or" to combine the expressions. The system status breadcrumb should be placed on top of the System Status Button in the Header bar.		
Repaint Screen		The Repaint Screen object is used to refresh the display client. The button uses defined symbol "Repaint" to build the proper repaint macro command for that L1 area and client.	No configuration required. See Macros and Multi-Monitor to verify that the "Client Startup" macro is configured properly and repaint macros are created as required. "Repaint #2"		
Home Navigation		The purpose of the Home Navigation object is to provide a link allowing an operator to go to their "Home" area or sphere of influence. Navigates to Client Home displays (not the current L1 home displays).	No configuration required. See Macros to verify that the "Client Startup" macro is configured properly. "GoHome"		
L1 Navigation		The purpose of the L1 Navigation object is to link to a pop-up display that provides access to other L1 Process Areas within the facility. This object configured the same for all L1 Headers (it will always call up the same pop-up, regardless which L1 area is being displayed).	101	Display Map Display Name	Enter the whole display name for the Display Map display pop-up. Display "#101" /cc See Displays for more details on configuring the display map pop-up.

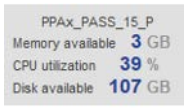
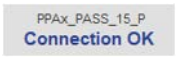
Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Administrator		The Administrator Button can be used to navigate to a custom administrator display or the provided Administrator System Security pop-up template display.	101	Administrator Display Name	Enter the whole display name for the Administrator display Display "#101"
Generic Trend Navigation		The purpose of the Generic Trend Navigation Button object is to navigate to a display prepopulated with navigation buttons to various prebuilt trends or generic trend display to allow building of ad-hoc trend displays.	101	Trends Display Name	Enter the whole display name for the Trends display Display "#101"
Full Display Trend Navigation		The purpose of the Full Display Trend Navigation button object is to navigate to the display created from the template display (raP-5_30-SE) Template Trend_Full. This display can use TrendPro templates and trend security. The user will typically use this style of display for system trend of that L1 area or for key performance indicators.	101	Trends Full Screen Display Name	Full name of the full screen trend name. This should be (raP-5_30-SE) Template Trend_Full or a display that is created from duplicating this display.
			102	Trend Template (Optional)	Name of the TrendPro template that should be invoked when the display opens. This can be left blank if no TrendPro templates are created yet.
			103	HMI Server name	The exact name of the HMI server. The name next to this icon in your application: 
			104	PASS Server name	The exact name of the server that is hosting your HMI Server (usually the PASS). You can find this name by examining the top of the application tree: 
			Display "#101" /T #102, #103, #104, $Security\ConfigTrend$		
Diagnostic Events Summary		The Diagnostics Events Summary object is used as a navigation button to access the Automatic Diagnostic Event Summary object It is recommended to use this button to navigate the display created from "(raP-5_30-SE) Template Diagnostic-IOEvents".	101	Diagnostic Display Name	Enter the whole display name for the Diagnostic display Display "#101"
Language Switching		The purpose of the Language Switching Button object is to provide ability for the user to change the HMI text to use their preferred (previously configured) language. The selection is client based and each client can choose another language if the data sources are configured with the selected language. The dynamic text is provided by the controller and the static text is provided by the HMI Server (both sources can provide information in multiple languages concurrently).	101	Language Select Display Name	Enter the whole display name for the Language Selection display Display "#101" /RP
Reports Navigation		The purpose of the Reports Navigation button object is to access web-based SQL Server Reporting Services (SSRS) reports. Once these reports are configured, you can access alarm and events reports and diagnostic reports.	101	Computer Name for the report	Enter the name of the server hosting the SSRS reports, for example "PPLib-ASIS".
			102	Port Number	Enter the port number that is used for accessing reports. Typically, by default this is 80 for HTTP.
			103	Reports Display Name	Enter the whole display name used for reports. Reports display.
			Display "#103" /T"http://#101:#102/Reports"		

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Help Button		The purpose of the Help Button object is to provide access to a User-defined Help display or PDF file. There are two separate buttons available depending on if you want to use a Help display (FTView-based) or PDF. These buttons can be added to the Header display or any other display.	Display: 101	Help Display Name	Enter the whole display name for the help display. Display #101
			PDF: 101	Help File Full file path For example: C:\Users\public\documents\help.pdf or \\PRC-PASS\Shared\help.pdf	Enter the file path to the Help PDF. The file can reside on the OWS that the client is run on or on a shared file directory. AppStart #101
Windows Navigation Button		The purpose of the Windows Navigation Button objects is to provide Windows like navigation capability within the HMI. Note: For multi-monitor applications, the display history is shared with all configured monitors. Therefore, this navigation should be concerned as a common group monitor history.	None. Buttons are ready to use and must only be added to the Header display.		
Login / Logout		The Login / Logout object is used to allow logging in and out of various users and includes an indication of the current user. Logging out will log in as a View Only User.	This object is already populated on the default Header display. Note: For log out to the view only user to work correctly, the view only user must be configured in security and added to the view only user group. The logout button is configured for user "default" password "default".		

APP - Resource and Status Objects (raC-1_00-SE)

IMPORTANT This is only available with Graphic Framework v1.00 and later.

The following objects are to be used with FactoryTalk Resource and Status Server. More information on recommended configuration of the FactoryTalk Resource and Status Server can be found here: FactoryTalk Resource & Status Server Configuration on page 113.

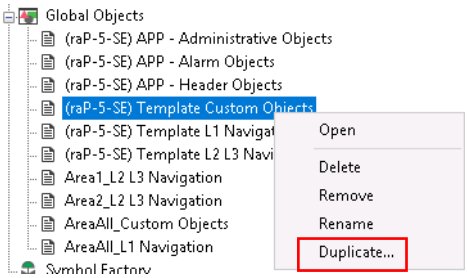
Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Workstation Monitoring		Use this object to navigate to "RSSWorkstation-Faceplate". The object provides basic information on a system workstation or virtual machine when used with FactoryTalk Resource and Status Server (FTRS server), including memory, CPU, and storage space. The object gives a visual indication when connection is lost to the workstation.	101	Workstation path	Provide the file path to the FTRS server of the workstation being monitored. For example: • "/AreaFolder/AreaSubfolder::[Workstation] • "/SysSts/PASS_P::[Workstation
			102	Network connection verification path	• "Provide the file path to the FTRS server that is monitoring the network connection to this workstation (Note: this should not be the same path as parameter 101). For example: • /AreaFolder/AreaSubfolder::[NetworkDevices]\DeviceIPAddress • /SysSts/FTD::[NetworkDevices]\172_18_1_75
			103	Number of CPU	Provide the number of CPU configured for this workstation (Max of 8 - Note: This must be a tag). For example: Const\Num4
			104	Number of NIC	Provide the number of network interface cards configured for this workstation (Max of 2 - Note: This must be a tag). For example: Const\Num1
			105	Letter of additional disk drive	Provide the letter of a second storage drive if used (Optional - Leave blank of only C drive). For example: E or F
			120	Additional display parameters	Optional parameter for opening display at a specific location. For example: /RP, /CC
			121	Additional display parameters	Optional parameter for opening display at a specific location. For example: /RP, /CC
Network Connection Monitoring		Use this object to navigate to "RSSNetworkDevices-Faceplate". The object provides basic information on network connection of a system workstation, virtual machine, switch, or other device when used with FactoryTalk Resource and Status Server (FTRS server). The object will give a visual indication when connection is lost to the workstation.	102	Network connection verification path	Provide the file path to the FTRS server that is monitoring the network connection to this workstation. For example: • "/AreaFolder/AreaSubfolder::[NetworkDevices]\DeviceIPAddress • "/SysSts/FTD::[NetworkDevices]\172_18_1_75
			120	Additional display parameters	Optional parameter for opening display at a specific location. For example: /RP, /CC
			121	Additional display parameters	Optional parameter for opening display at a specific location. For example: /RP, /CC
Processes Monitoring		Use this object to navigate to "RSSProcessMonitor-Faceplate". The object provides basic information on configured processes running on a workstation or virtual machine when used with FactoryTalk Resource and Status Server (FTRS server). The object gives a visual indication when a process is not running. **Note**: See FactoryTalk Resource & Status Server Configuration on page 113 for information on recommended processes to monitor on typical system servers.	101	Workstation path	Provide the file path to the FTRS server of the workstation being monitored. For example: • "/AreaFolder/AreaSubfolder::[Workstation] • "/SysSts/PASS_P::[Workstation
			102	Number of Tasks	Provide the number of processes on the workstation that will be monitored (Max of 10 - Note: This must be a tag). For example: Const\Num2
			1010	Workstation Name for Display	Provide text that will be displayed on the global object at runtime (Entered as text). For example: OWS FTDirectory
			103 ... 112	Task Name	Enter the name of the system process being monitored. For example: RnaDirServer, RNADirMultiplexor, and so on.
			120	Additional display parameters	Optional parameter for opening the display at a specific location. For example: /RP, /CC
			121	Additional display parameters	Optional parameter for opening display at a specific location. For example: /RP, /CC

Object	Graphic	Description	Configuration		
			Parameter Number	Description	Explanation
Server Redundancy and Switchover		Use this object to provide redundancy status on HMI, data, and alarm servers in the FactoryTalk View system. Objects can initiate server switchover if user has proper security rights.	101	HMI Server Path	Provide the path to the HMI server including any area folders between the application and the server. For example: /PlantA/HMIServerName
			102	Data Server Path	Provide the path to the Data server including any area folders between the application and the server. For example: /PlantA/DataArea/FTLinx
			103	Alarm Server Path	Provide the path to the Alarm server including any area folders between the application and the server. For example: /PlantA/FTAEserver

Template Custom Objects

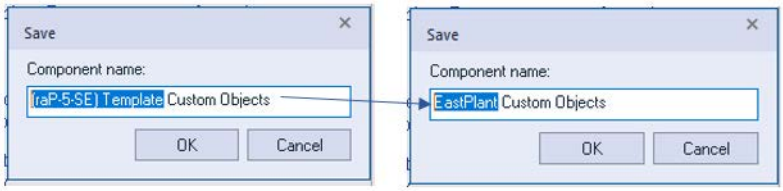
The following objects are customizable to customer's specific needs. Before customizing, duplicate and rename the file to preserve the original template file. The following steps are not required if you are not using any of the custom global objects within the file.

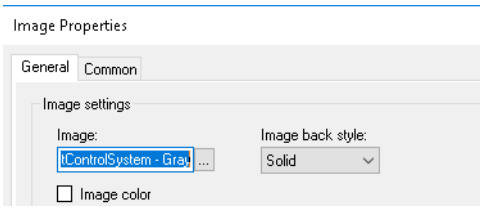
1. Go to file > Duplicate.



2. Name the new global object file.

   Use a filename that represents the application/facility. Replace only the '(raP-5_30-SE) Template' or '(raC-1_00-SE) Template' portion of the filename. This creates a file for your specific application and preserves the original template file.



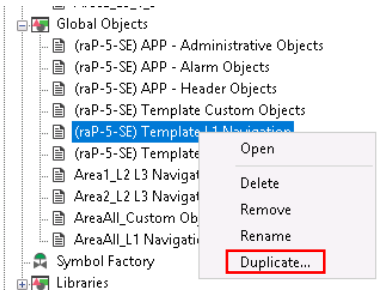
Object	Graphic	Description	Configuration
Custom Company Logo		The logo object is in the Custom Objects files can be replaced with the customer's logo. The customer logo must first be imported into the application Images folder. Once the image is imported, open the object in Custom Objects file and replace with customer logo. 	Once the image file is correct, copy the updated global object and paste it onto the header after deleting the default PlantPAx logo.
Custom Report Navigation		The purpose of the Custom Report Navigation Button object is to navigate to a display with pre-populated navigation buttons to access various prebuilt reports.	Copy and paste the button on the Header display (or any other display) in the desired location. Update the navigation as necessary.
URL Reports Navigation		The purpose of the URL Reports Navigation Button object is to pop open a web browser over the client to access the specified URL. This allows the user access to the default web browser.	Copy and paste the button on the Header display in the desired location. Update the hyperlink, as necessary.

Template L1 Navigation

This global object file is a template. The template file for L1 Navigation will only need to be utilized once for each application. This file defines the navigation to each L1 Area - one button per each L1 area. The following steps are required for all applications using the graphic framework, creating one new file for each application.

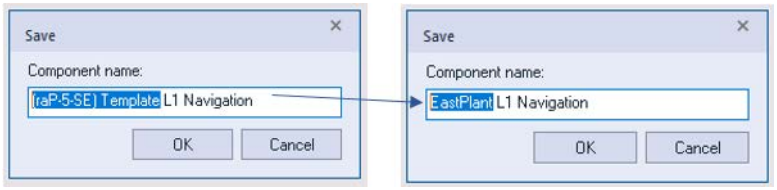
To utilize this file, use the following steps:

1. Go to file > Duplicate.



2. Name the new global object file.

   Use a filename that represents the application/facility. Replace only the '(raP-5_30-SE) Template' or '(raC-1_00-SE) Template' portion of the filename. This creates a file for your specific application and preserves the original template file.

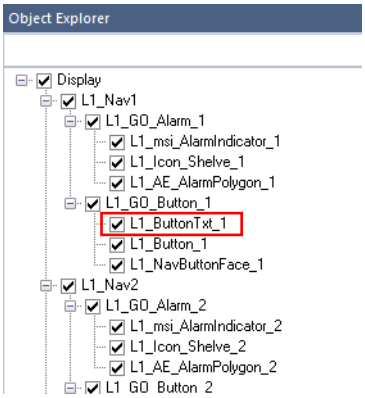


3. Duplicate the buttons as required (one for each L1 area).

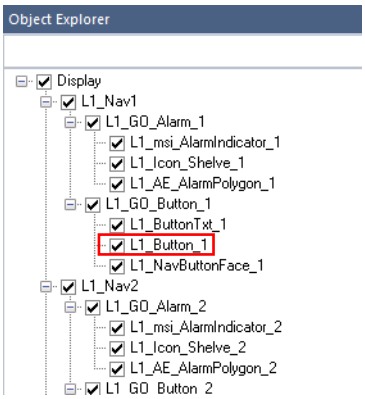
Four buttons are provided by default - not all buttons need to be used.

4. To update the text on the button, go to Object Explorer and select the L1_ButtonTxt_# object and modify as required.

   Repeat for each button.

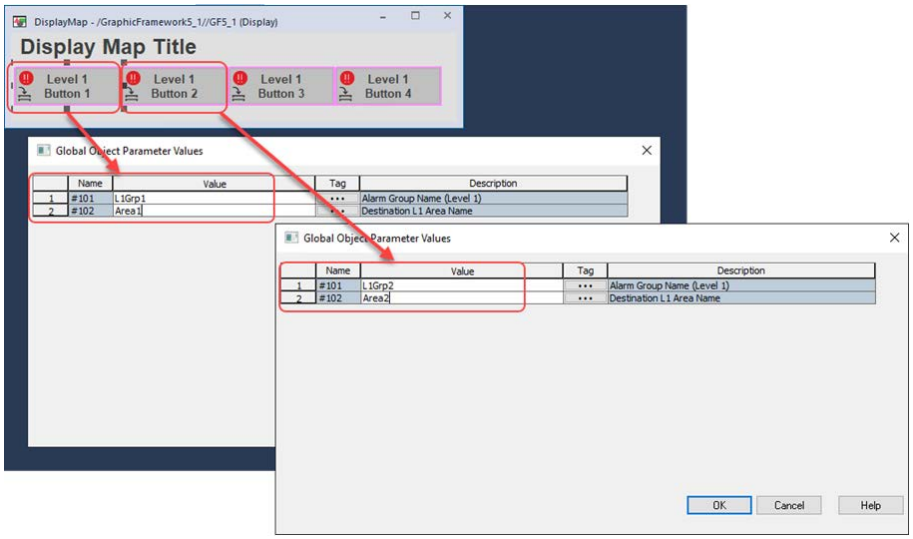


5. The navigation for each button should be left as is. The symbol "Repaint" is used with parameter #2 to call "SetRepaint" macro to the build the command for the proper macro to repaint all screens. See Macros for more information on configuration. See Multi-Monitor for more information on navigating between L1 areas with multi-monitor client workstations.



6. After you finish updating the button text, select the updated buttons in the L1 Navigation global object file and copy them to the application-specific display that is developed from the template file (raP-5_30-SE) Template Display Map or (raC-1_00-SE) Template Display Map. Delete any existing buttons and paste the new buttons.
7. For all buttons, enter the L1 alarm group parameters in the global object parameters. Also enter the name of the destination L1 area. This name should match the area name that is used for the repaint macros that are used in that L1 area. See Alarm Grouping

and Supporting Logic for more information on alarm grouping. See Macros for more information on configuration.

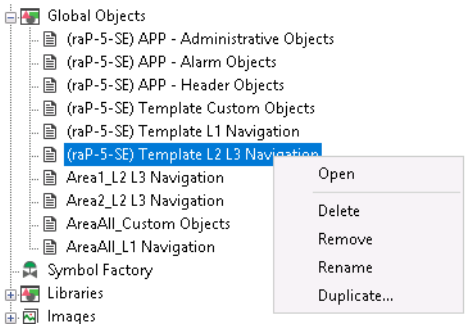


Template L2 L3 Navigation

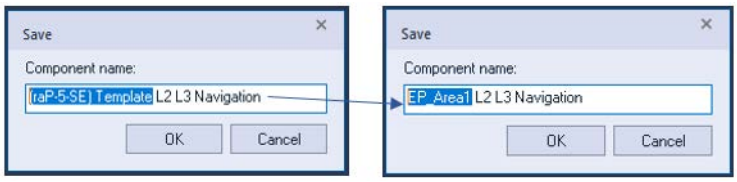
This global object file is a template. Utilize the template file for L2 / L3 Navigation once for every L1 area. This file defines the navigation to each L2 and L3 display within a given L1 Area. The following steps are required for all applications using the graphic framework, create a new file for each L1 Area.

For each L1 Area, perform the following steps:

1. Go to file > Duplicate.



2. Name the new global object file. Use a filename that represents the specific L1 area. Replace only the '(raP-5_30-SE) Template' or '(raC-1_00-SE) Template' portion of the filename. This creates a file for your specific L1 area and preserves the original template file.



There are four sets of buttons to update in the newly created global object file for each L1 Area:

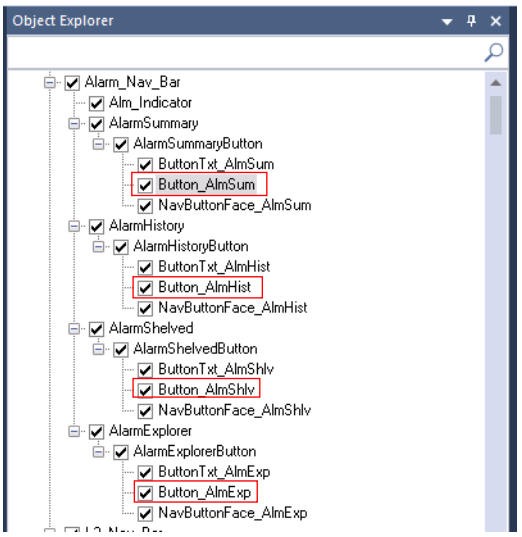
- Alarm Navigation Bar
- Diagnostic Navigation Bar
- L2 Navigation Bar

- L3 Navigation Bars

Alarm Navigation Bar

Only one Alarm Navigation bar is needed for each L1 area. For the Alarm Navigation, the button text does not need to be updated. Only the navigation must be updated.

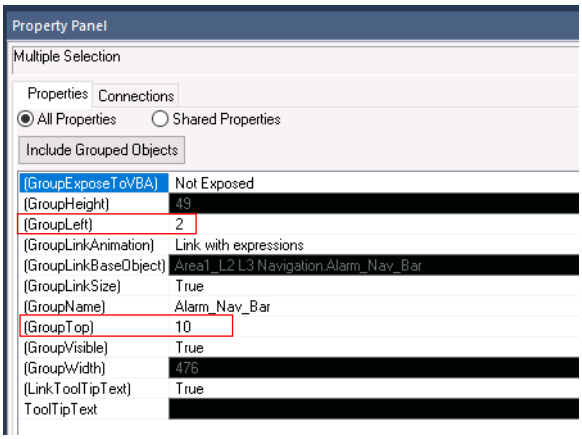
1. To update the navigation for each button, go to the alarm button > Action tab and update the display names for each of the alarm screens.



   This should match the Alarm Displays created for this L1 area (see Displays for more information on the alarm template displays).

   If the alarm display names match the recommended naming convention, you can do a "Tag Substitution" and simply replace "(raP-5_30-SE) Template" or '(raC-1_00-SE) Template' on the whole Alarm Navigation bar instead of updating each button individually.

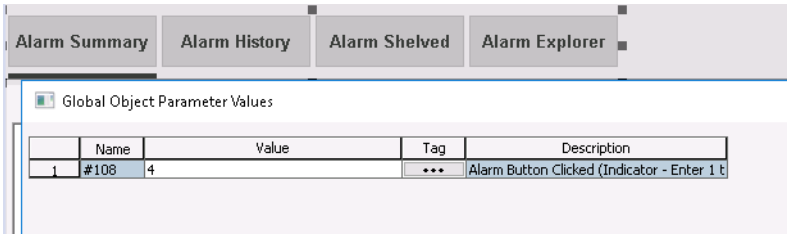
2. Copy the button bar and paste the bar in each of the four alarm displays:
   - [L1Area] Alarm-Summary
   - [L1Area] Alarm-History
   - [L1Area] Alarm-Shelved
   - [L1Area] Alarm-Explorer
3. To update the location for the alarm navigation bar in the alarm displays, go to the Alarm Navigation bar and place the Alarm Navigation bars in this location on each of the four alarm displays:
   Left - 2, Top - 10.



4. Update the global object parameter for the Alarm Button indication.

This shows the operator what alarm display is being viewed. Update the global object parameter for each alarm display:

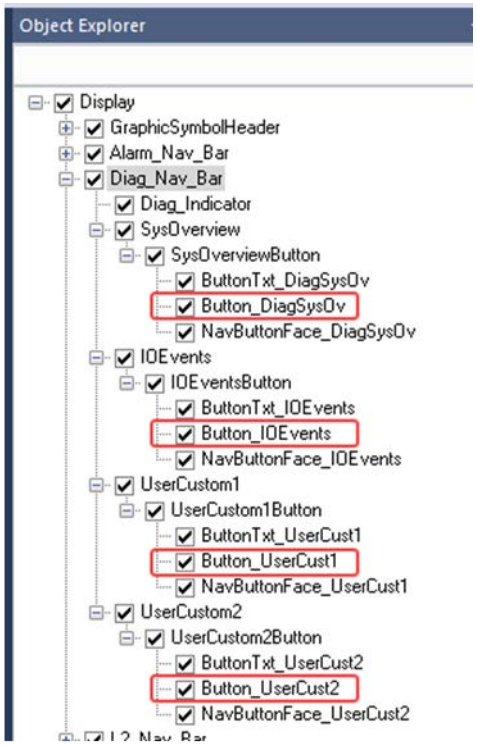
- Alarm Summary = 1
- Alarm History = 2
- Alarm Shelved = 3
- Alarm Explorer =4



Diagnostic Navigation Bar

Only one Diagnostic Navigation bar is needed for each L1 area. For the Diagnostic Navigation, the button text does not need to be updated. Only the navigation must be updated.

1. To update the navigation for each button, go to the diagnostic button > Action tab and update the display names for each of the diagnostic screens. Note: There is one user-customizable button available on this navigation bar, to be used for additional diagnostic displays as needed.



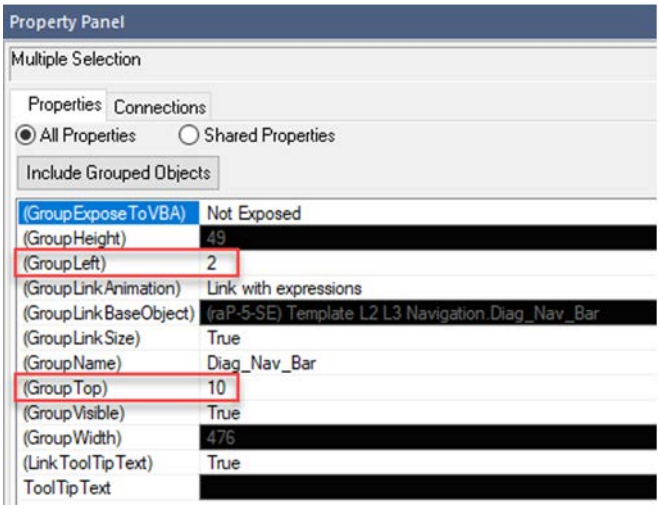
This should match the Diagnostic Displays created for this L1 area (see Displays for more information on the diagnostic template displays).

If the diagnostic display names match the recommended naming convention, you can do a "Tag Substitution" and simply replace "(raP-5_30-SE) Template" or '(raC-1_00-SE) Template' on the whole Diagnostic Navigation bar instead of updating each button individually.

2. Copy the button bar and paste the bar in each of the two diagnostic displays, as well as any user custom diagnostic displays that have been created:
   - [L1Area] Diagnostic-Summary

- [L1Area] Diagnostic-IOEvents

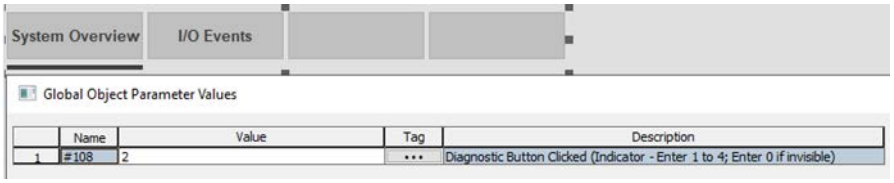
3. To update the location for the diagnostic navigation bar in the diagnostic displays, go to the Diagnostic Navigation bar and place all Diagnostic Navigation bars in this location on each of the diagnostic displays: Left - 2, Top - 10.



4. Update the global object parameter for the Diagnostic Button indication.

   This shows the operator what diagnostic display is being viewed. Update the global object parameter for each diagnostic display:

   - Diagnostic Summary = 1
   - IO Event Viewer = 2
   - User custom = 3
   - User custom =4



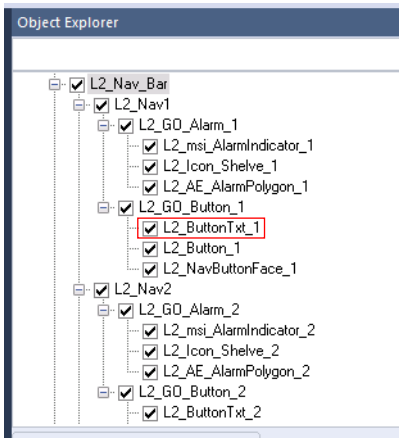
L2 Navigation Bar

Only one L2 Navigation bar is needed for each L1 area. Update the text on the buttons that are being used.

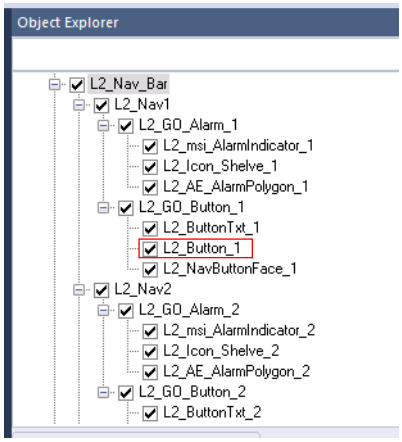
IMPORTANT	If using Graphic Framework v1.00 with the built-in navigation menu, the L2 Navigation Bar is not needed.

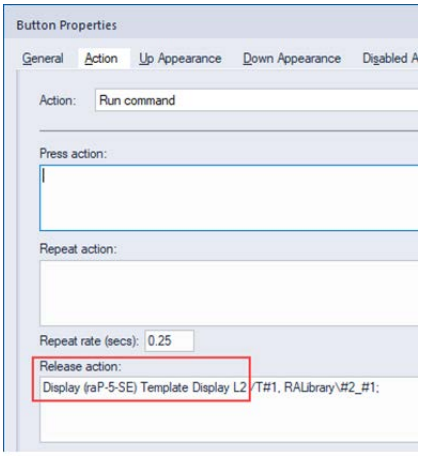
1. To update the text on the button, got to Object Explorer and select the L2_ButtonTxt_# object.

Repeat for each button.



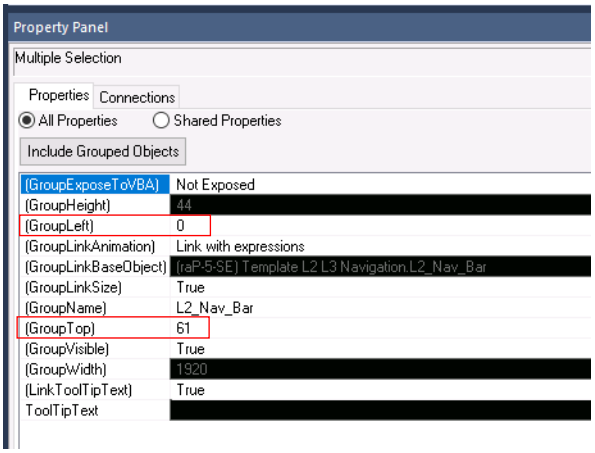
2. To update the navigation for each button, go to the L2_Button_# object > Action tab and replace the Release Action to point to the correct L2 display. Repeat for each button used.





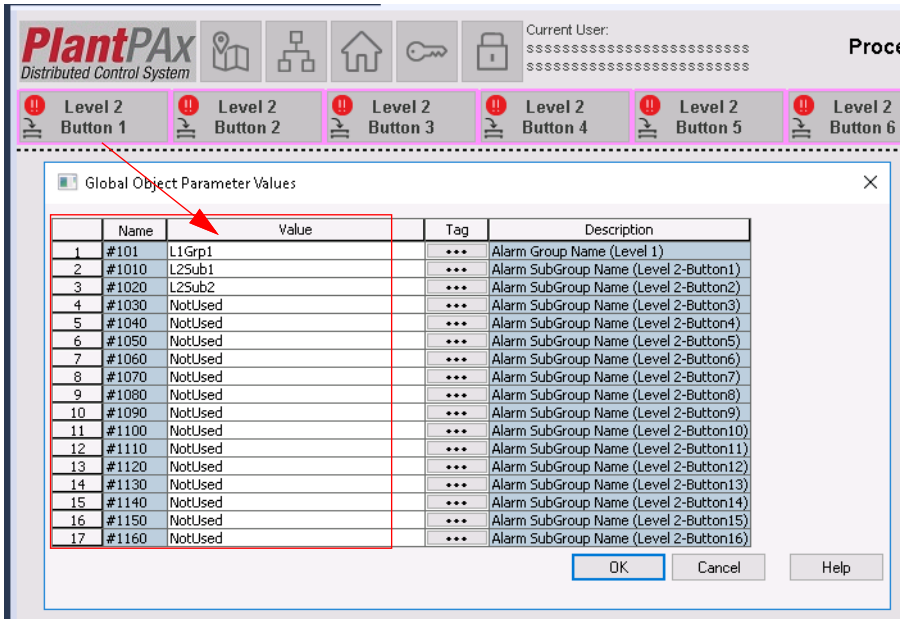
3. After you finish updating the button text and actions, select the updated button bar and copy.
4. Go to the application-specific display developed from the template files (raP-5_30-SE) Template Mon# Header or (raC-1_00-SE) Template Header Mon#, delete the existing L2 Navigation bar, and paste the new L2 Navigation bar.

5. Go to the L2 Navigation bar and place the bar in this location on the L1 Header display: Left - 0, Top - 61.



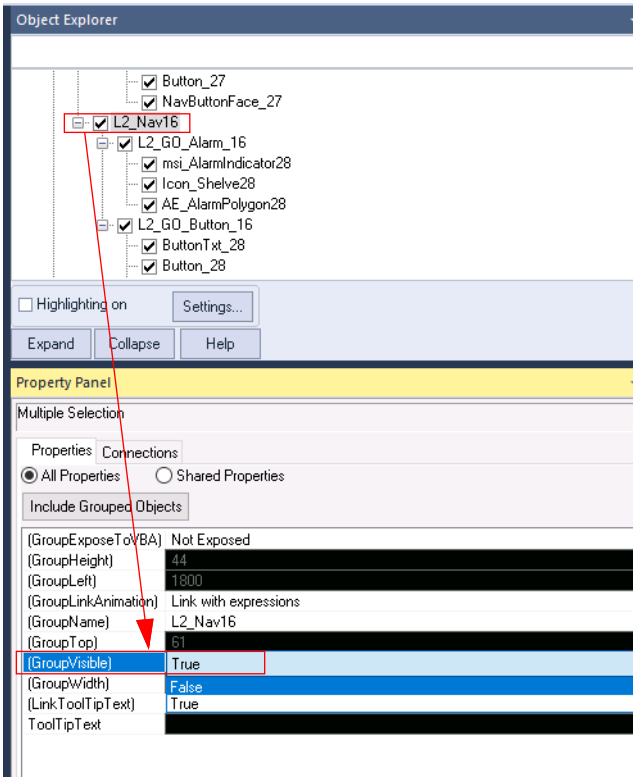
6. For all buttons (used or not used), Enter the L1 and L2 alarm group parameter in the global object parameters. See Alarm Grouping and Supporting Logic for more information.

   These fields MUST be entered with text or errors populate in FactoryTalk® Diagnostics. Enter the appropriate alarm group name for the buttons used. If button is not used, simply enter "NotUsed" as shown below. This acts as a dummy alarm group.



7. To make a button that is not used invisible, go to the Header graphic Object Explorer, select the button, and modify the Group Visible parameter.

Repeat for each button that should be invisible.



L3 Navigation Bar

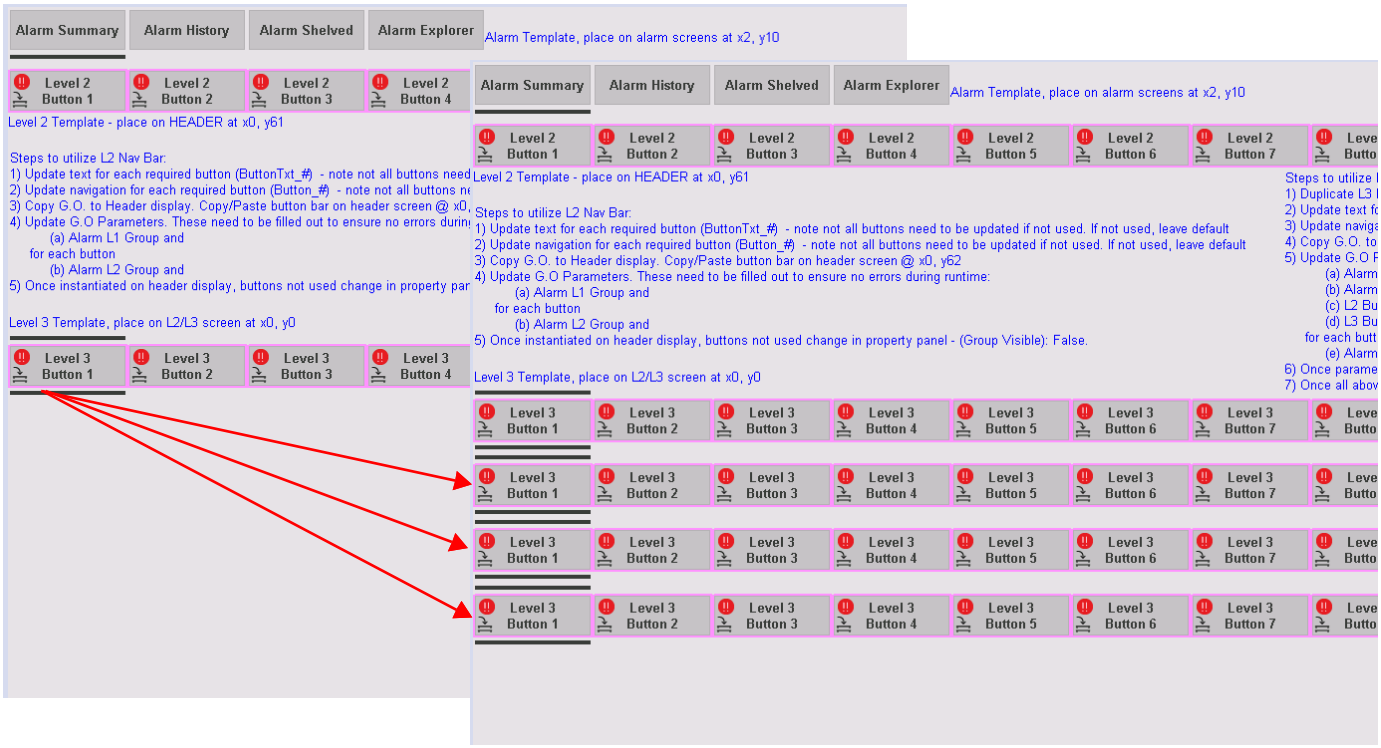
One L3 Navigation bar is needed for each L2 Navigation button that is used (or up to 16 L3 Navigation bars per L2 Navigation bar). Update and configure each of the L3 Navigation bars.

IMPORTANT If using Graphic Framework v1.00 with the built-in navigation menu, the L3 Navigation Bar is not needed.

1. Go to the L3 Navigation bar in the global object file and copy and paste as many L3 Navigation bars as needed.

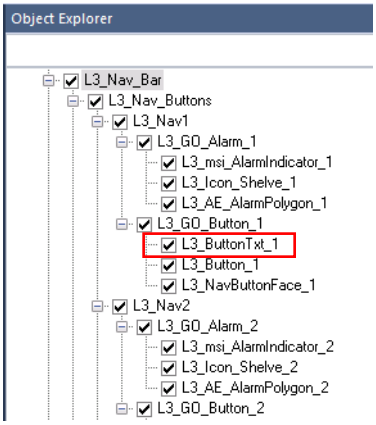
   The first L3 Navigation bar correlates to the first L2 Navigation button; the copied L3 navigation bar correlates to the second L2 Navigation button, and so on, for additional copies.

There can be as many as 16 L3 Navigation bars in the global object file for the L1. This example shows four L3 Navigation bars (only four L2 buttons are used in this example).



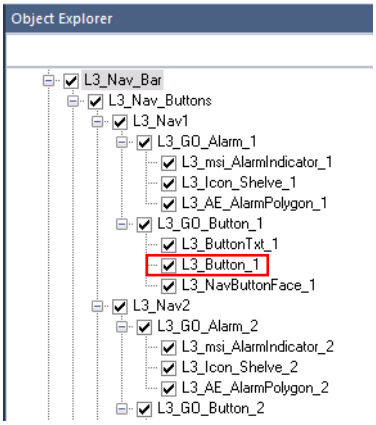
2. To update the text on the button, got to Object Explorer and select the L3_ButtonTxt_# object.

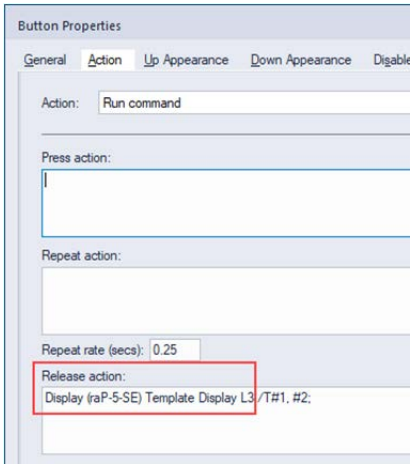
   Repeat for each button.



3. To update the navigation for each button, go to the L3_Button_# object > Action tab and replace the Release Action to point to the correct L2 display.

Repeat for each button used.



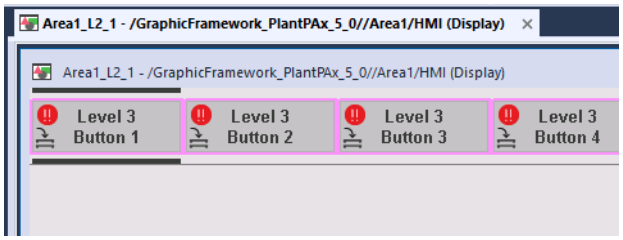


The object names in the L3 Navigation bars that are copied from the first L3 Navigation Bar do not populate new button numbers in order. Take care when configuring buttons that the correct one is selected.

4. Select the updated button bar and copy.

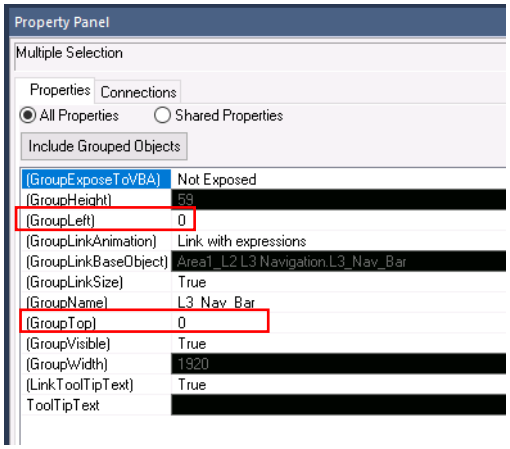
   The object names in the L3 Navigation bars that are copied from the first L3 Navigation Bar do not populate new button numbers in order. Take care when configuring buttons that you select the correct bar.

5. Go to the application-specific display developed from the template file (raP-5_30-SE) Template Display L2 or (raC-1_00-SE) Template Display L2 for this L2 area in this L1 area, delete the existing L3 navigation bar, and paste the new L3 navigation bar.



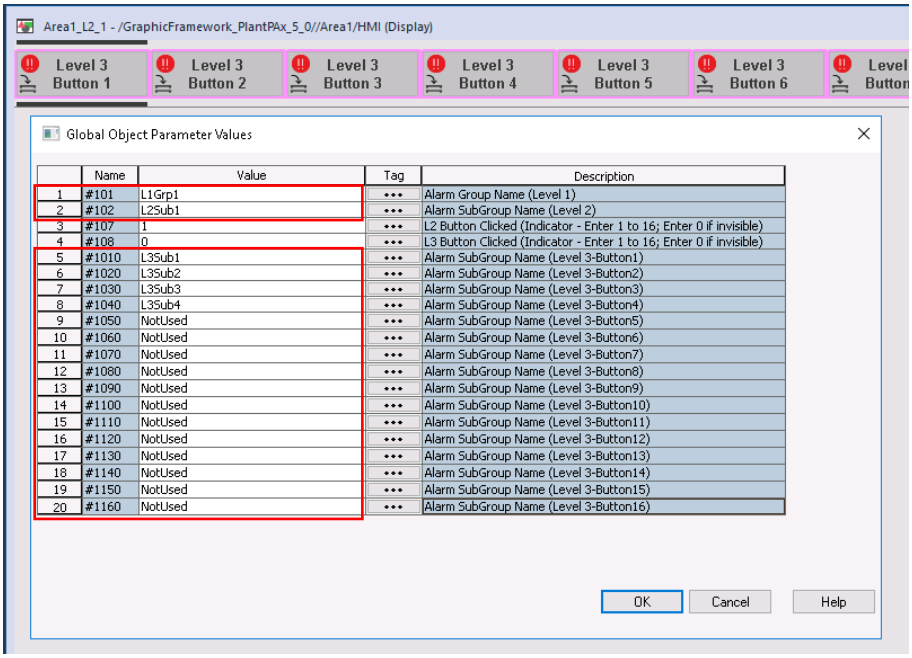
6. Place the button bar in this location on the L2 and L3 displays: Left-0, Top-0.

You can update the location on the property panel for the L3 Navigation bar while in the Header display.



7. For all buttons (used or not used), the L1, L2, and L3 alarm group parameter must be entered in the global object parameters. See Alarm Grouping and Supporting Logic for more information.

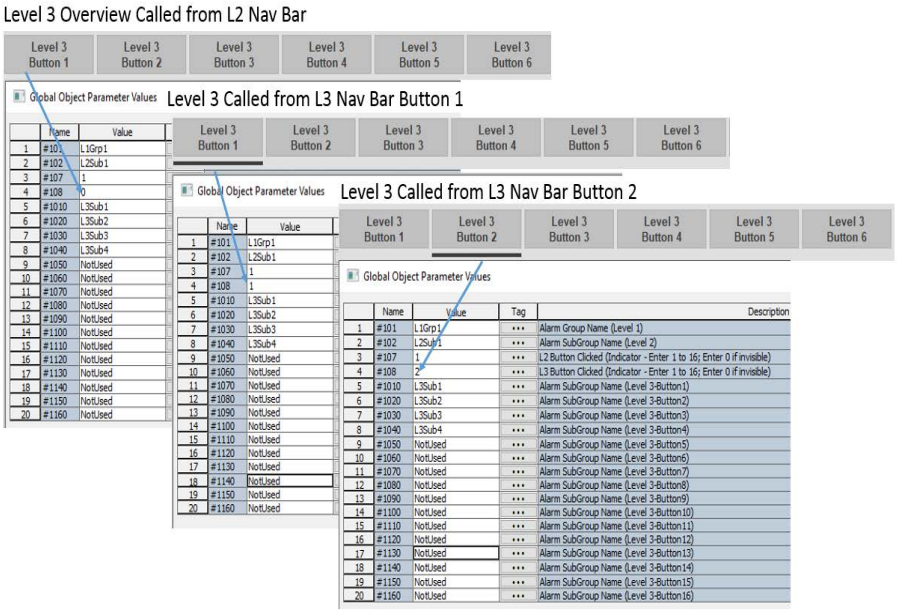
   These fields MUST be completed with text or errors populate in FactoryTalk Diagnostics. Enter the appropriate alarm group name for the buttons used. If a button is not used, simply enter "NotUsed" as shown in the following display. This acts as a dummy alarm group.



8. Edit parameters 107 and 108. Parameters #107 and #108 are used for active display indication. The indicators are horizontal dark gray lines that appear beneath the L2 and L3 navigation bars to indicate the active display.

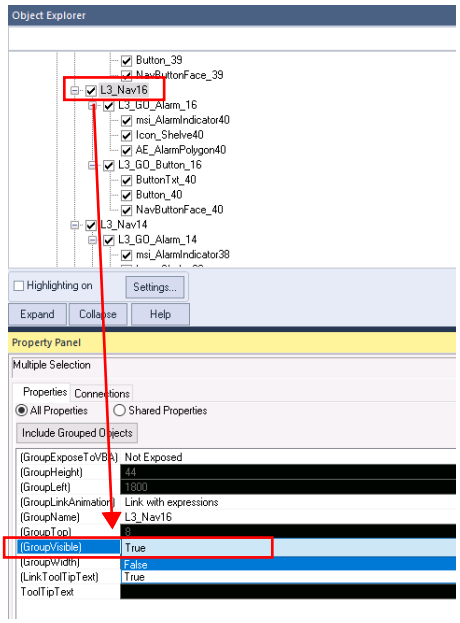
Parameter	Description
107	Parameter #107 is a component of the L2/L3 display and is used to position the indicator for the active L2 display (It appears below the L2 Navigation bar). Valid values for #107 range from 0 to 16: **L2 or L3 Display Active** #107 = 0: no indication #107 = 1...16: locates the indicator in position 1 to 16 (left to right) to indicate the active L2 selection or the L2 associated with the active L3 display.
108	Parameter #108 is a component of the L2/L3 display and is used to position the indicator for the active L3 display (it appears under the L3 Navigation bar). Valid values for #108 range from 0 to 16: **L2 Display Active** #108 = 0: no indicator appears as no L3 display is yet selected **L3 Display Active** #108 = 1...16: locates the indicator in position 1 to 16 (left to right) to indicate the active L3 selection.

The indicator uses horizontal animation with the parameter to indicate the button selected.



9. If a button on the L3 Navigation bar is not used, the button can be made invisible. While in the L2 graphic, select the populated L3 Navigation bar. In the Object Explorer, select the button to be made invisible. In the Property Panel, modify the "Group Visible" parameter from True to False.

Repeat for each button that should be invisible.



10. While in the L2 graphic, select the L3 Navigation bar and copy.
    a. In the L2 area, open all L3 graphics that are associated with this L2 area, delete the L3 Navigation bar in each of the L3 graphics and paste the updated L3 Navigation bar.
    b. Update the global object parameter #108 for each L3 graphic.

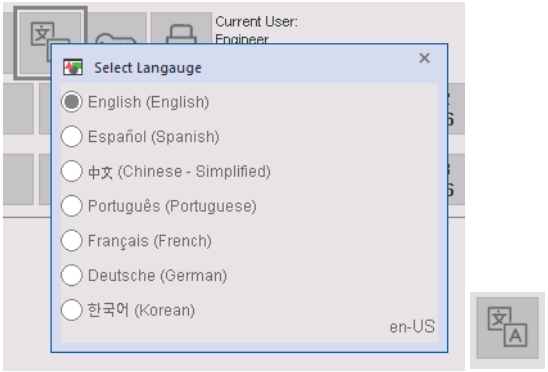
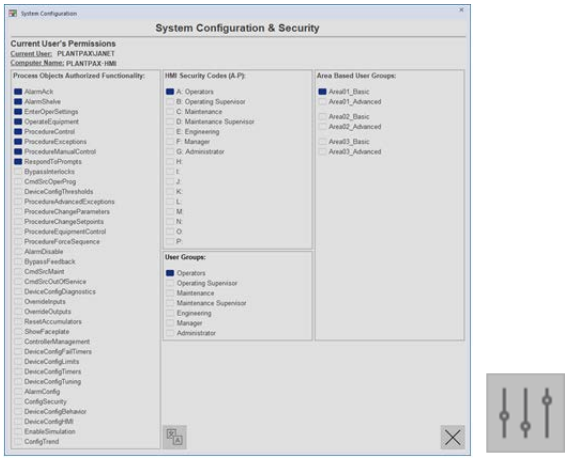
       Repeat this section for each L3 Navigation bar in the global object file.

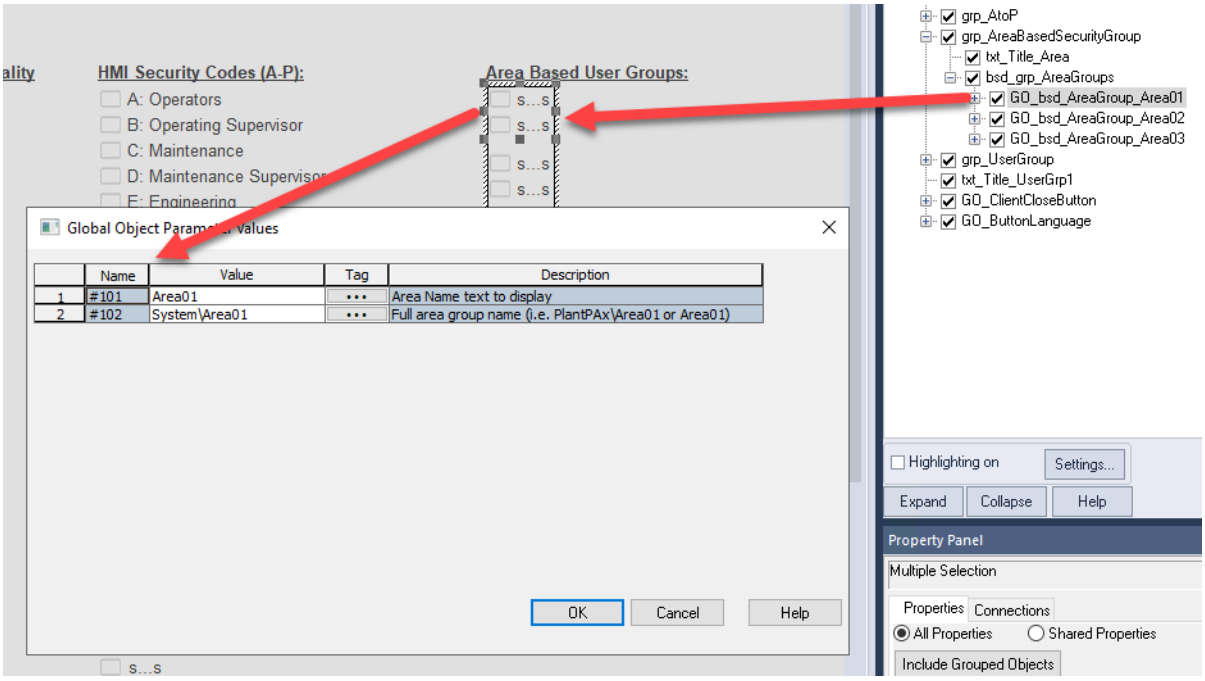
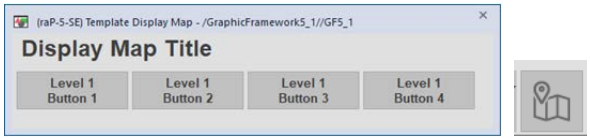
L2 Indication Only

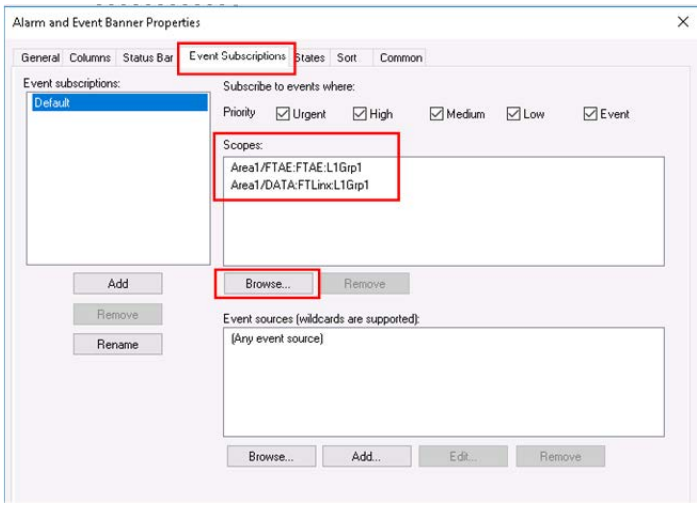
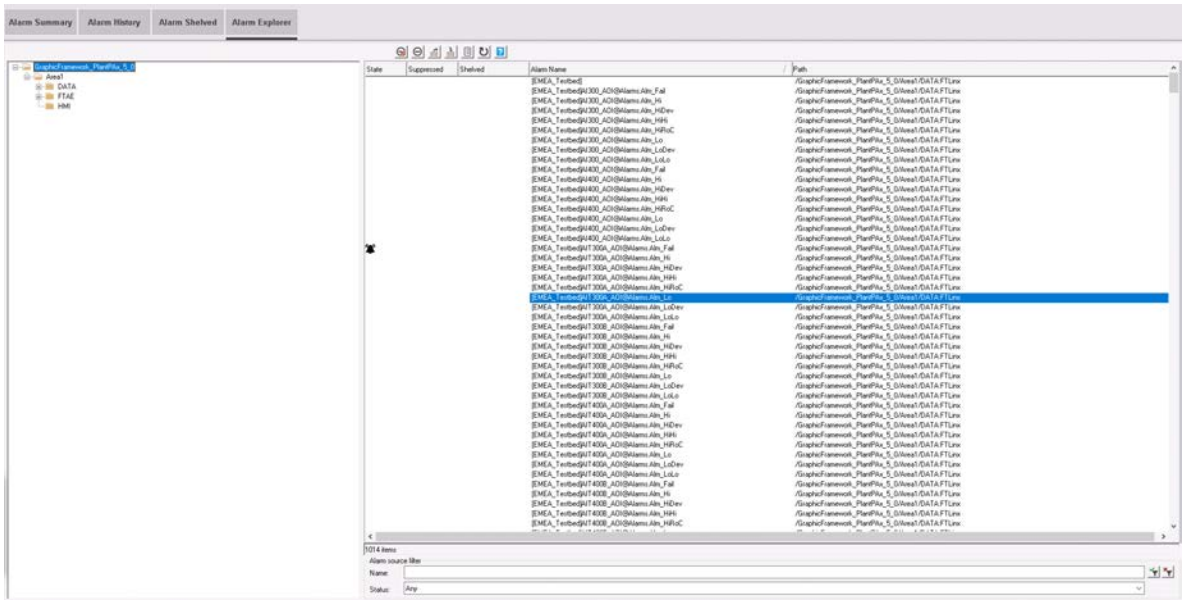
On displays where the L3 navigation bar is not utilized, a single indicator for the selected L2 screen will be used. This is placed by default on the display "(raP-5_30-SE) Template Display L2 No L3" or "(raC-1_00-SE) Template Display L2 No L3". No configuration is required in the global object file. See Displays for configuration on the default display

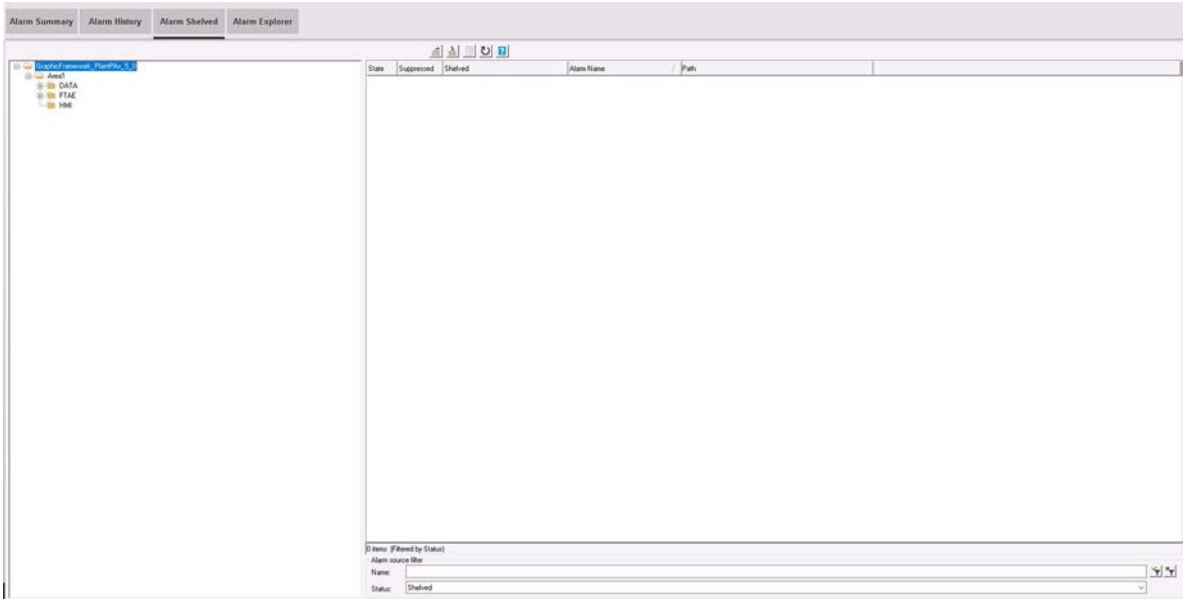
Displays

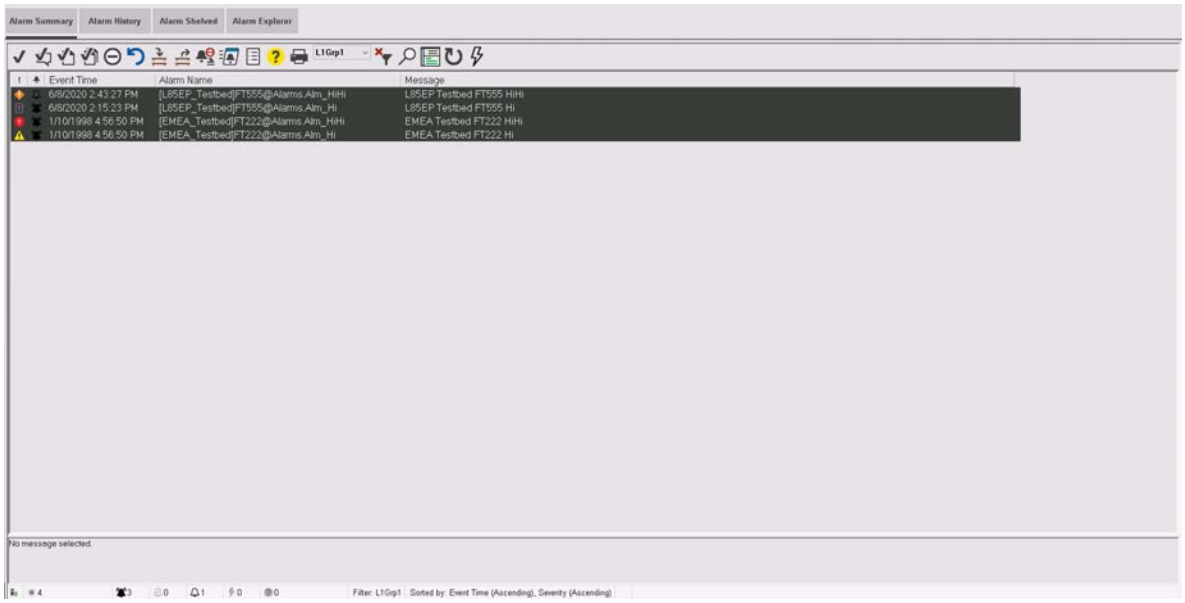
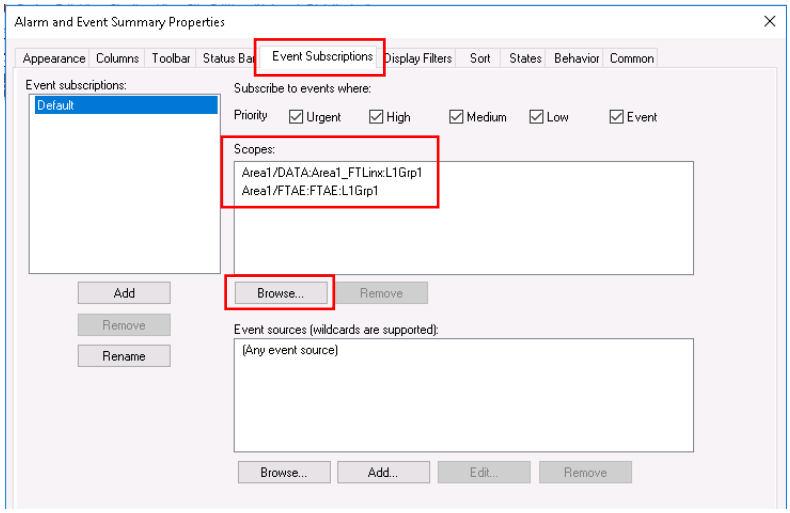
Each display is a template except for those appended with “Faceplate”. The template display should be duplicated and the prefix “(raP-5_30-SE) Template” or "(raC-1_00-SE) Template" replaced with meaningful name for each L1 area in the application. This preserves the original template to use as a starting point on additional screens. See Build Your PlantPAx HMI Application for more information on naming structure.

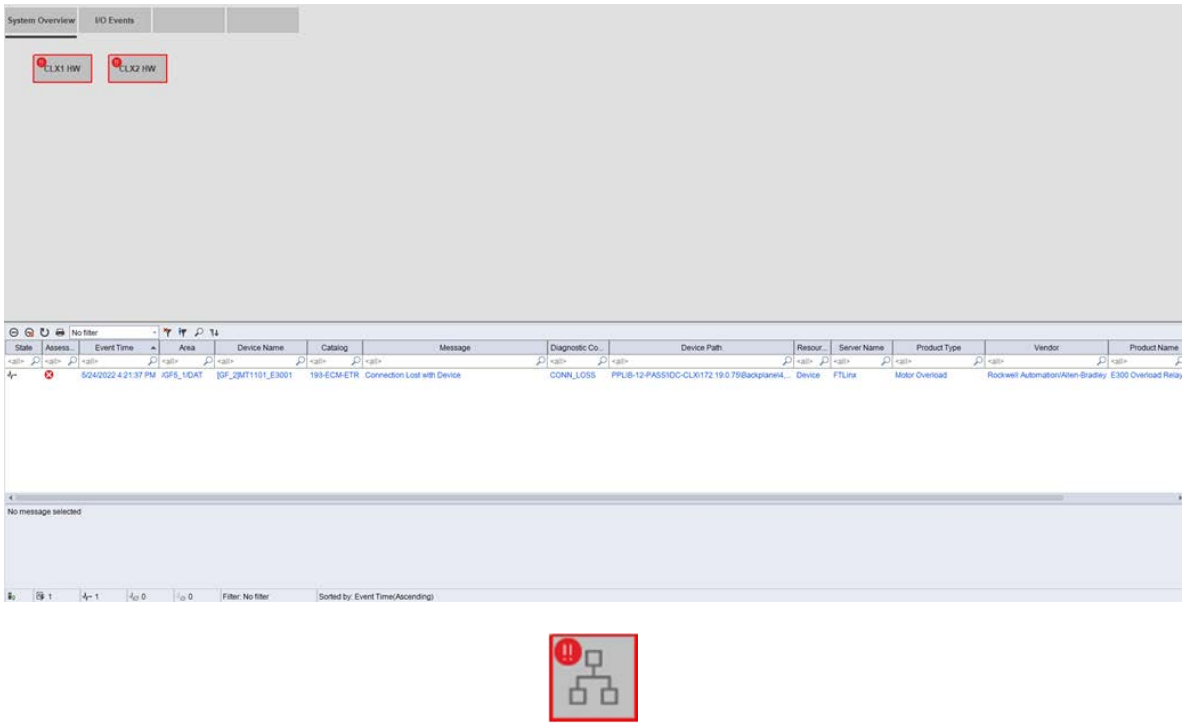
Display	Graphic	Description
Template Language-Select		This template can be used if language switching is used in the application. Only one per application is required. This should be used with the Header button for Language Switching. The display is pre-populated with typical languages used but can be modified for application-specific needs. Link this display to the Language Switching Header button. Languages that are not used for this application can be removed if desired.
Template Admin-SysSecurity		This template is used as a pop-up display for a summary of the current user's security access for A-P security, area security, and basic information such as user group and computer name. This should be used with the Header button for Administrator. Link this display to the Administrator Header button. The sections on the display (Process Objects Authorized Functionality, HMI Security Codes (A-P), and User Groups) are configured with recommended PlantPAx configuration and should not need to be modified. Update the section for Area-Based User Groups to reflect the areas used in the application. There is space at the bottom right of the pop-up display for users to add additional administrator-level content. See (Template Admin-SysSecurity (Continued)

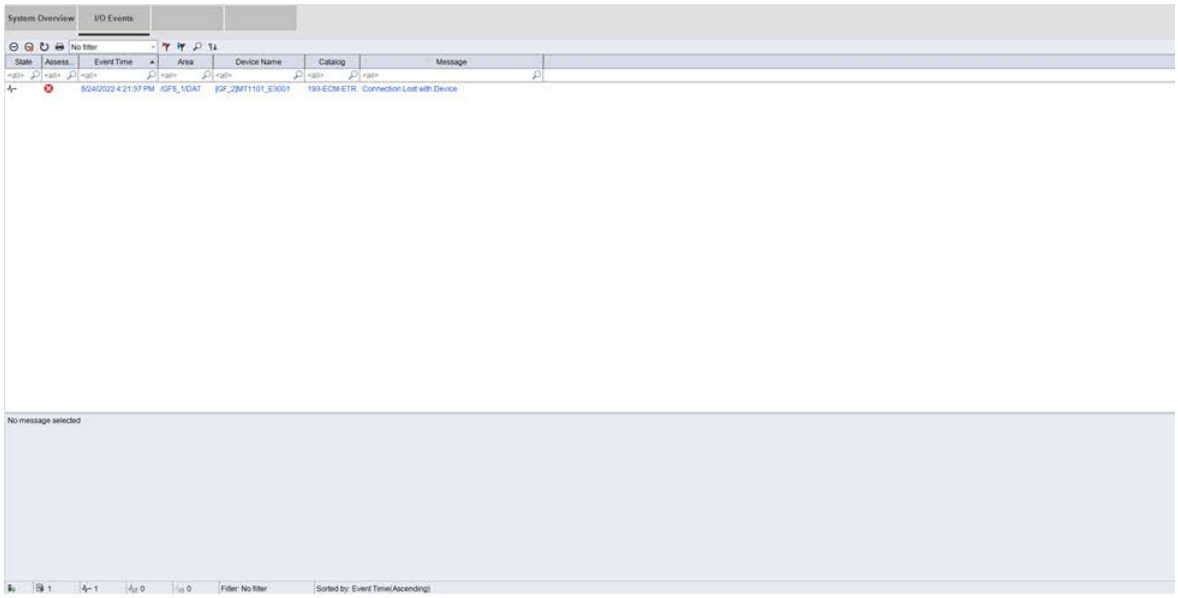
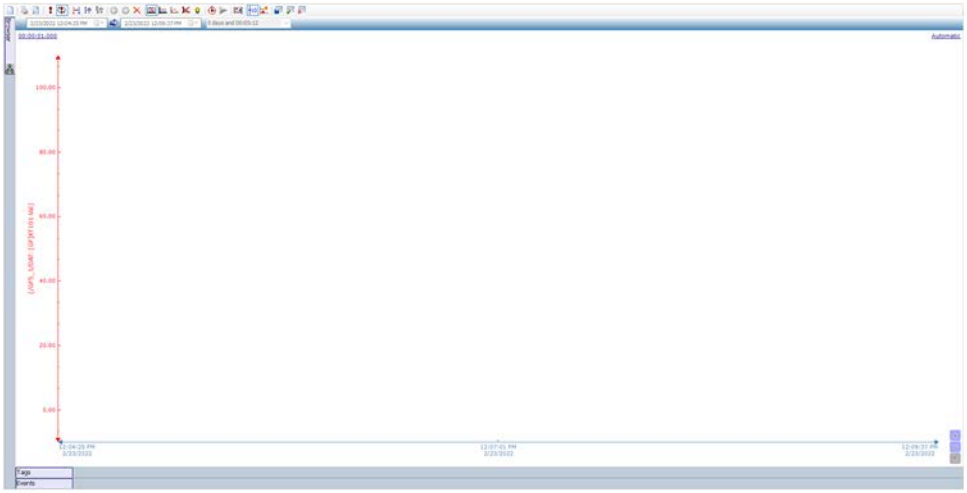
Display	Graphic	Description
(Template Admin-SysSecurity (Continued)		
		
Template Display Map		This template is used for navigation between different L1 areas. Only one per application is required. This should be used with the Header button for L1 Navigation. Link this display to L1 Navigation Header button. Update the Display title for the specific application. See Global Objects for more information on configuring the buttons on this display.

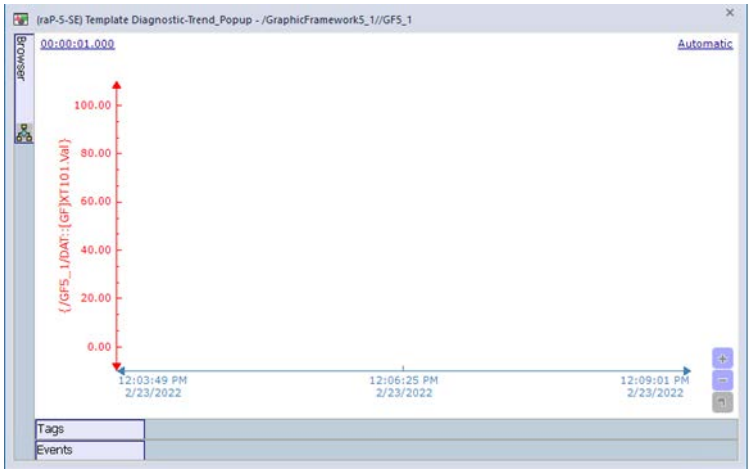
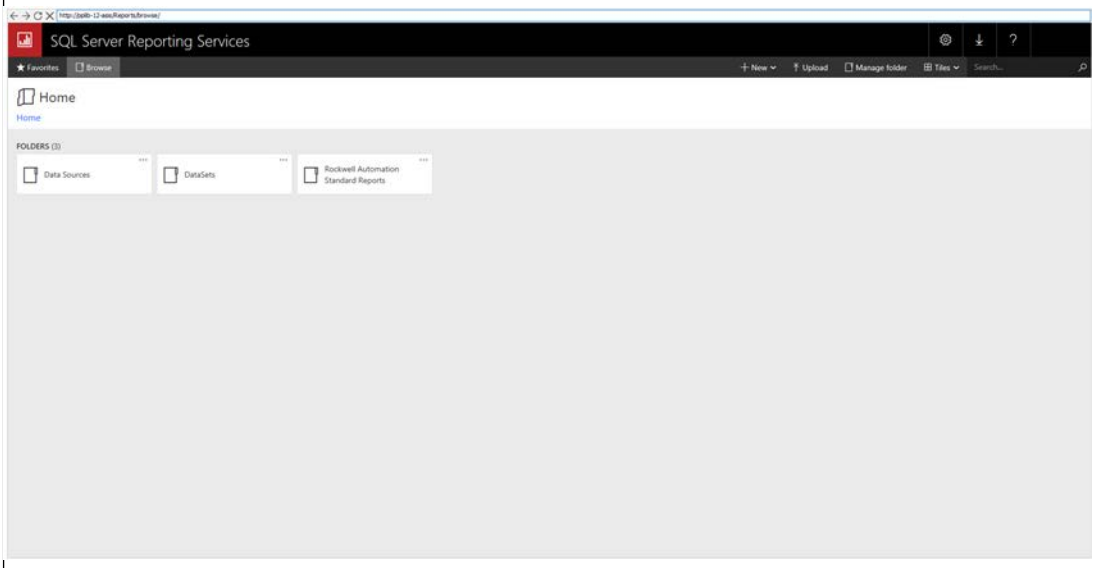
Display	Graphic	Description
Template Header Mon1 Template Header Mon2 Template Header Mon3 Template Header Mon4 Template Header Nav Menu		These templates are used for Headers for each L1 area. Depending on the monitor configuration for the operator for the L1 area, one or more Header displays can be used for each L1 area. It is up to the user whether all Headers in a multi-monitor configuration use the same Header display or use a different header for each monitor. See Multi-Monitor for more information on configuring a multi-monitor system. The buttons on the header can be modified using objects that are provided in the global object files. The L2 Navigation bar resides on this screen and is always visible. The alarm banner object must be configured for alarms in that L1 area. Open the Alarm and Event Banner Properties and select the Event Subscriptions tab. Then select the "Browse" button under "Scopes" box. Select the L1 area groups that correlate with that Header. Note: If there are alarms that are both controller based and server based, both subscriptions must be added. Every Alarm or Data server that has alarms for this L1 area must be added to the scope of the alarm banner.  The L2 Navigation bar must be configured properly - See Global Objects to configure the L2 Navigation.
Template Alarm-Explorer		This template is used for Alarm Explorer for each L1 area. This template should be used once for each L1 area. The Alarm Navigation bar must be configured - see Global Objects for more information.

Display	Graphic	Description
Template Alarm-History		This template is used for Alarm History for each L1 area. This template should be used once for each L1 area. The Alarm Navigation bar must be configured - see Global Objects for more information. Filters can be added to the Alarm History object if desired from the Alarm History properties.
Template Alarm-Shelved		This template is used for Shelved Alarms for each L1 area. This template is be used once for each L1 area. The Alarm Navigation bar must be configured - see Global Objects for more information.

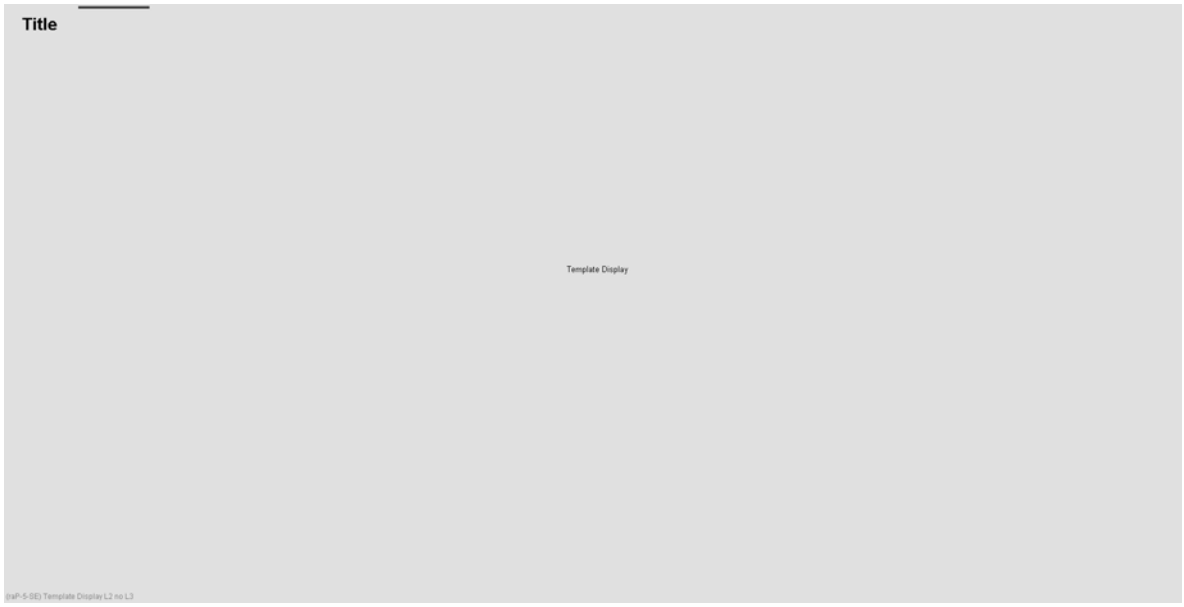
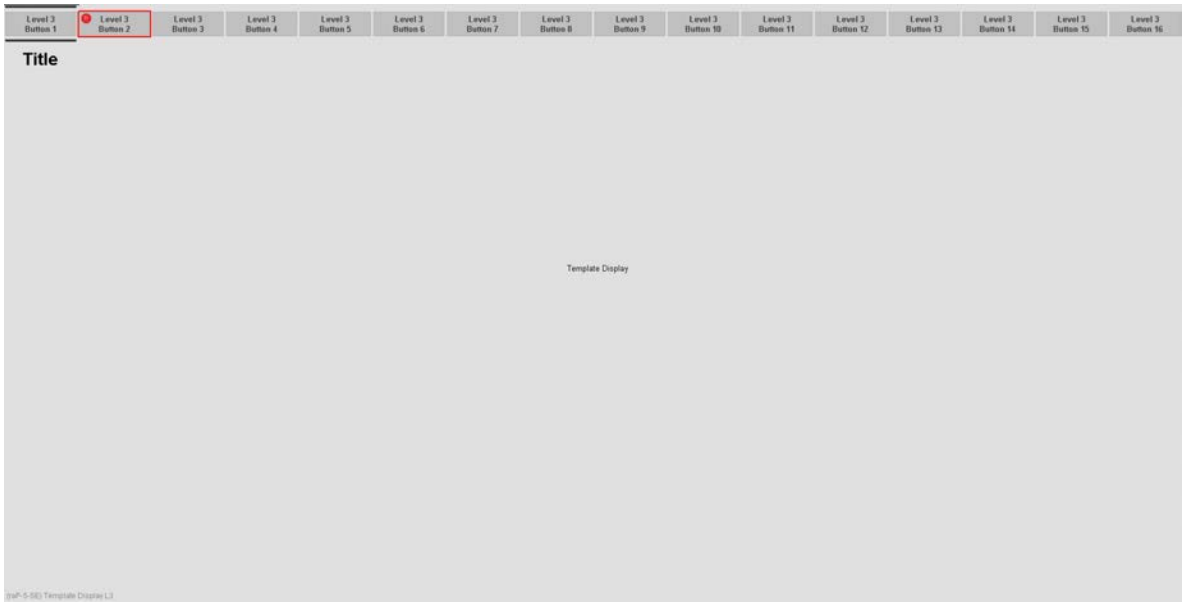
Display	Graphic	Description
Template Alarm-Summary	 	This template is used for Alarm Summary for each L1 area. This template is used once for each L1 area. Link this display to the Alarm Header button. The Alarm Navigation bar must be configured - see Global Objects for more information. Display filters can be added if desired in the Alarm and Event Summary properties. The alarm summary object must be configured for alarms in that L1 area. Open the Alarm and Event Summary Properties and select the Event Subscriptions tab. Then select the "Browse" button under "Scopes" box. Select the L1 area groups that correlate with that Header. Note: If there are alarms that are both controller based and server based, both subscriptions must be added. 

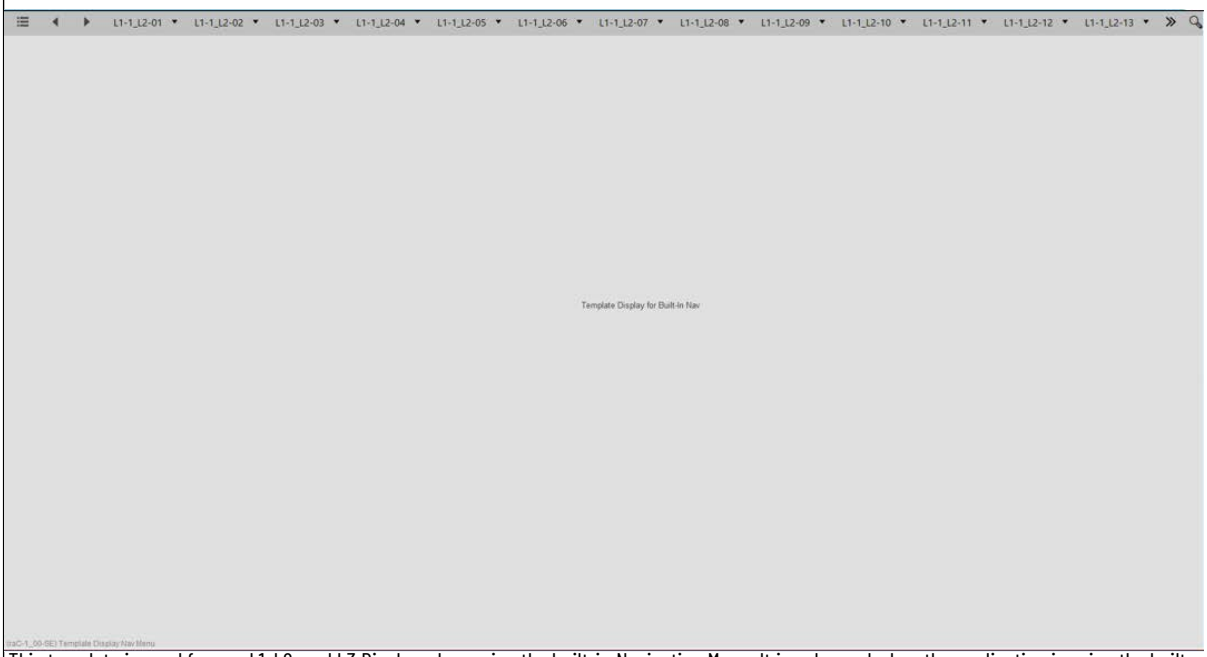
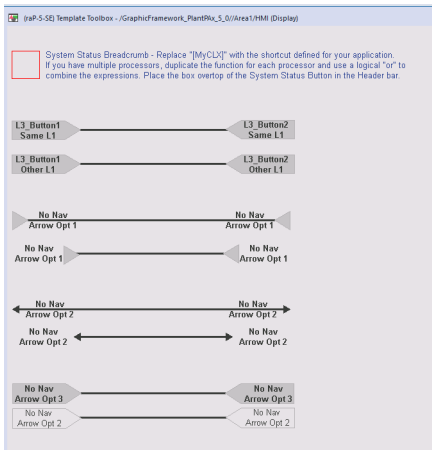
Display	Graphic	Description
Template Diagnostic-Summary		This template is used for access to hardware organizational tree view and examples of FactoryTalk Resource and Status server objects as well as additional custom diagnostic objects that a user may want to add. There is also an Automatic Diagnostic Event Summary object at the bottom of the display. This template can be used either one for the whole facility or one for each L1 area. If a display is created for each L1 area, then the event subscription scope must be adjusted for each L1 display. Otherwise, no configuration is required. See Organization, Ownership, Arbitration, and Propagation (OOAP) for configuration of the software and hardware organizational tree view objects. Link this display to the Diagnostic Events Summary Button or access via the Diagnostic Navigation bar.

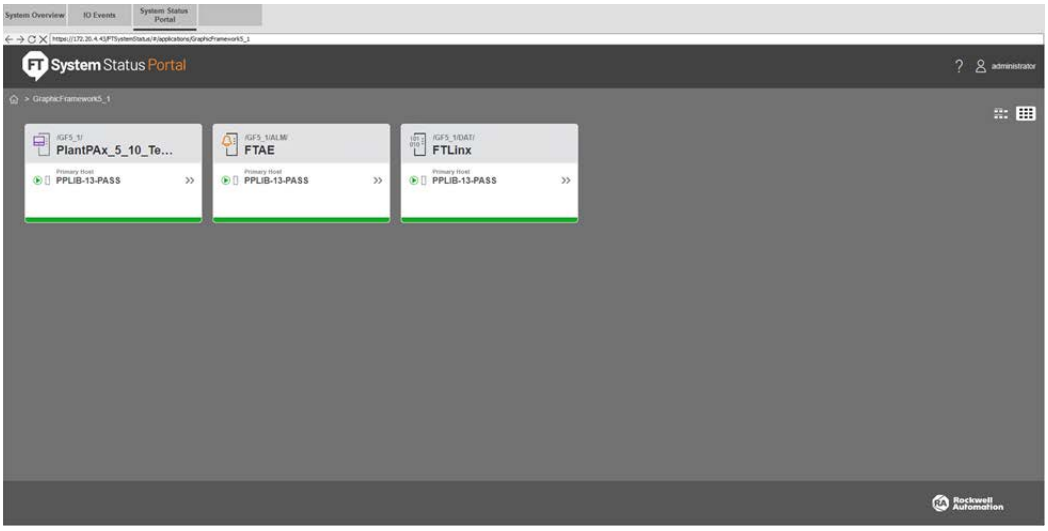
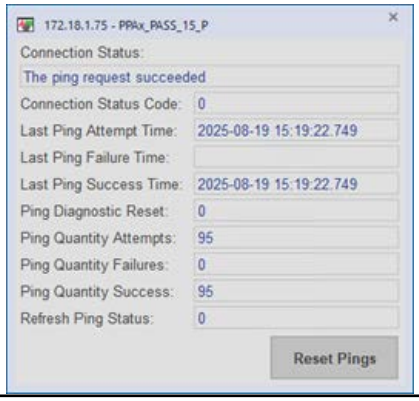
Display	Graphic	Description
Template Diagnostic-IOEvents	 	This template is used for full-page Automatic Diagnostic Event Summary. This template can be used either one for the whole facility or one for each L1 area. If a display is created for each L1 area, then the event subscription scope must be adjusted for each L1 display. Otherwise, no configuration is required. Link this display to the Diagnostic Events Summary Button or access via the Diagnostic Navigation bar.
Template Trend_Full	 	This template is used for full screen display of the TrendPro object. It is recommended to use one per application and use various TrendPro templates to view different trend configurations. The desired initial TrendPro template view can be entered as a parameter into the Trend navigation button, but different TrendPro templates can be applied from the TrendPro toolbar. No direct configuration is required for this display. Link this display to the Trend header button - see Global Objects for details on navigation configuration.

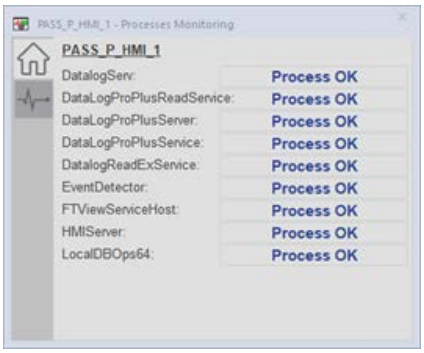
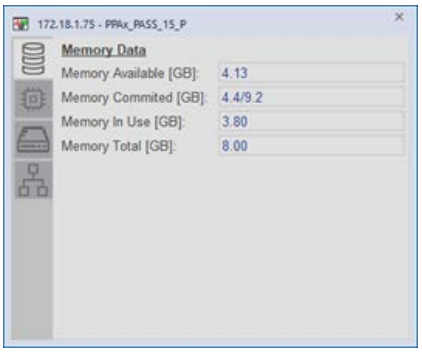
Display	Graphic	Description
Template Trend_Popup		This template is used for pop-up display of the TrendPro object. It is recommended to use one per application and use various TrendPro templates to view different trend configurations. The desired TrendPro template view can be entered as a parameter into the Trend pop-up navigation button. No direct configuration is required for this display. Link this display to the Trend pop-up button. The Trend pop-up button can be placed across L1, L2, or L3 process displays - see Global Objects for details on navigation configuration.
Template Reports		This template is used to access SSRS reports via the SE Web Browser object. The template is used once for each system. The SSRS reports must be installed and configured before using this display. Link this display to the Reports Header Button. See Global Objects for more information on configuring the buttons on this display.

Display	Graphic	Description
Template Display L1		This template is used for each L1 Display. There will be one L1 display for every L1 area. Typically, this display has an overview of that L1 area and is the first display that the operator will see when the client starts up. This display is flexible - alarm indicators can be added if desired. Link this display to the appropriate macros for client startup and screen repaint - see Macros for information.
Template Display L2		This template is used for L2 Displays that have L3 displays associated. There will be one L2 display for every L2 Navigation button that is utilized. The template will automatically display the GFX file name in the lower left corner. Typically, this graphic contains the necessary controls and indication for the operator to run the facility. Link each L2 Display to the appropriate L2 Navigation Button - see Global Objects for details on navigation configuration.

Display	Graphic	Description
Template Display L2 No L3		This template is used for simple L2 Displays that do not have L3 Displays associated (no L3 navigation bar). There will be one L2 display for every L2 Navigation button that is utilized. The template will automatically display the GFX file name in the lower left corner. Typically, this graphic contains the necessary controls and indication for the operator to run the facility. Link each L2 Display to the appropriate L2 Navigation Button - see Global Objects for details on navigation configuration. The parameter for the global object for L2 button indication must be configured. There is one parameter (#107) and is configured the same as with the L3 navigation bar. See Global Objects for details on this parameter.
Template Display L3		This template is used for each L3 Display. There is one L3 display for every L3 Navigation button that is utilized. The template will automatically display the GFX file name in the lower left corner. Typically, this graphic contains more detailed information on devices that are on associated L2 display. Link each L3 Display to the appropriate L3 Navigation Button - see Global Objects for details on navigation configuration.

Display	Graphic	Description
Template Display Nav Menu		This template is used for any L1, L2, and L3 Display when using the built-in Navigation Menu. It is only used when the application is using the built-in Navigation Menu. The template will automatically display the GFX file name in the lower left corner. See Navigation Menu on page 111 for recommended configuration of the built-in Navigation Menu with the Graphic Framework v1.00.I
Template Toolbox		This display is used as a toolbox of objects that can be copied and placed on other displays. This screen is not used on any active clients. If the off-screen navigation objects are used, then the button action and text must be updated. • Same L1 1. Copy the object onto the desired screen. 2. Update the text. 3. Update the button action – navigate directly to display. • Other L1 1. Copy the objected onto the desired screen. 2. Update the text object. 3. Update the button action - use several commands to close current L1 view and opened desired L1 header and process display. These commands are developed to work with multi-monitor. See Multi-Monitor for more information. If the system status breadcrumb is used, the animation must be updated. Replace "/Area1/DATA::[MyCLX]" with the shortcut defined for your application. If multiple processors are used, duplicated the function for each processor and use a logical OR statement to combine the expression.

Display	Graphic	Description
Template Diagnostic-SysSts Only available in FactoryTalk View v13 and later templates.	 This template is used to access the FactoryTalk System Status Portal via the SE Web Browser object. The template should be used once for each system. The FactoryTalk System Status Portal utility must be installed on system servers when installing FactoryTalk Services Platform. This display should be accessed via the Diagnostic Navigation bar. The parameter path must be updated to the IP address of the FactoryTalk Directory server.	
Common-Redirect-to-4_10 Common-Redirect-to-5_00	These two displays are optional and only needed in applications that are using both the Process Library 4.10 and Process Library 5.00 or later. The redirect displays are used with modified navigation macros - See Macros for more information. No configuration, modification, or renaming is needed to use these displays. They only need to be added into the application if the application is using both the Process Library 4.10 and Process Library 5.00 or later **Note**: This is not available in the Graphic Framework v1.00 release	
(raC-1_00-SE) RSSNetworkDevice-Faceplate		This faceplate can be used with FactoryTalk Resource and Status server (FTRS server) to view the connection status of a configured device. The faceplate shows connection information and includes a button to reset the ping count if the user has the correct security.

Display	Graphic	Description
(raC-1_00-SE) RSSProcessMonitor-Faceplate		This faceplate can be used with FactoryTalk Resource and Status server (FTRS server) to monitor the status of up to 10 windows processes running on a configured workstation or virtual machine. The faceplate shows a general status of each process that is configured and more detailed information on each process on the diagnostic tab.
(raC-1_00-SE) RSSWorkstation-Faceplate		This faceplate can be used with FactoryTalk Resource and Status server (FTRS server) to monitor basic information of a workstation or virtual machine including: • Memory available, committed, in use, and total • CPU processes • Individual speed and utilization of up to 8 CPUs • Total disk space available and used • Disk space available and used on the C drive • Disk space is available and used on an additional configurable drive (for example, E or F drive, etc) • Detailed information on up to 2 network interface cards (NICs) including adapter name, IP address, send and receive speeds.

Multi-Monitor

The Graphic Framework provides template options for single, dual, or quad-monitor client workstations. There are several adjustments to button commands, startup macros, and displays as well as additional configuration that allows the multi-monitor functionality to operate smoothly.

IMPORTANT	Multimonitor is not supported with use of the built-in Navigation Menu.

HMI Tags, Headers, and Macros

There are HMI tags that must be created for multi-monitor to work correctly. Each tag is a string that stores the file name of the header displays used in each L1 area. All four tags should be created for each L1 area, regardless of the number of monitors used by workstations in that area. For example, if there is an Area 1 with one quad-monitor workstation, Area 2 with one dual-monitor workstation, and an Area 3 with one single monitor workstation, the following tags, macros, and displays should be created.

To begin, determine the area in the system that is using the maximum number of monitors. In our example, Area 1 is a quad monitor, so the maximum number of monitors would be four. Four header displays must be created for each area to ensure that the workstation in Area 1 operates correctly. Header displays can be identical or modified to show different information on each monitor. For each of the HMI tags, the value is each Header display file name.

L1 Area	HMI Tag Name	HMI Tag Value	Macros Needed	Displays Needed
Area 1	RALibrary\Area1_M1	Area1_Mon1_Header	• Area1_ClientStatup • Area1_Repaint_QuadMon • Area2_Repaint_QuadMon • Area3_Repaint_QuadMon	• Area1_Mon1_Header • Area1_Mon2_Header • Area1_Mon3_Header • Area1_Mon4_Header • Any required process, diagnostic, or alarm display
	RALibrary\Area1_M2	Area1_Mon2_Header		
	RALibrary\Area1_M3	Area1_Mon3_Header		
	RALibrary\Area1_M4	Area1_Mon4_Header		

L1 Area	HMI Tag Name	HMI Tag Value	Macros Needed	Displays Needed
Area 2	RALibrary\Area2_M1	Area2_Mon1_Header	• Area2_ClientStatup • Area2_Repaint_DualMon • Area1_Repaint_DualMon • Area3_Repaint_DualMon	• Area2_Mon1_Header • Area2_Mon2_Header • Area2_Mon3_Header • Area2_Mon4_Header • Any required process, diagnostic, or alarm display
	RALibrary\Area2_M2	Area2_Mon2_Header		
	RALibrary\Area2_M3	Area2_Mon3_Header		
	RALibrary\Area2_M4	Area2_Mon4_Header		
Area 3	RALibrary\Area3_M1	Area3_Mon1_Header	• Area3_ClientStatup • Area3_Repaint_SingleMon • Area1_Repaint_SingleMon • Area2_Repaint_SingleMon	• Area3_Mon1_Header • Area3_Mon2_Header • Area3_Mon3_Header • Area3_Mon4_Header • Any required process, diagnostic, or alarm display
	RALibrary\Area3_M2	Area3_Mon2_Header		
	RALibrary\Area3_M3	Area3_Mon3_Header		
	RALibrary\Area3_M4	Area3_Mon4_Header		

Area 1 Macro Files

The following tables show the macros that are used for each Area. Screen displays are shown for Area 1 to provide clarity. The displays for Area 2 and Area 3 will follow the same format.

Table 1 - Area_1ClientStartup

Command	Description
Display Area1_Mon1_Header /TM1,Area1 /M1 Display Area1_L1 /TM1, RALibrary\Area1_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area1_Mon2_Header /TM2,Area1 /M2 Display Area1_L1 /TM2,RALibrary\Area1_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.
Display Area1_Mon3_Header /TM3,Area1 /M3 Display Area1_L1 /TM3,RALibrary\Area1_M3 /M3	Display commands for monitor 3 (header and process display) with required tag parameters.
Display Area1_Mon4_Header /TM4,Area1 /M4 Display Area1_L1 /TM4,RALibrary\Area1_M4 /M4	Display commands for monitor 4 (header and process display) with required tag parameters.
Define GoHome Area1_Repaint_QuadMon	Define action of "GoHome" symbol used on Home Navigation Button
Define Repaint SetRepaint QuadMon	Define action of "Repaint" symbol that is used on Repaint Button and L1 Navigation

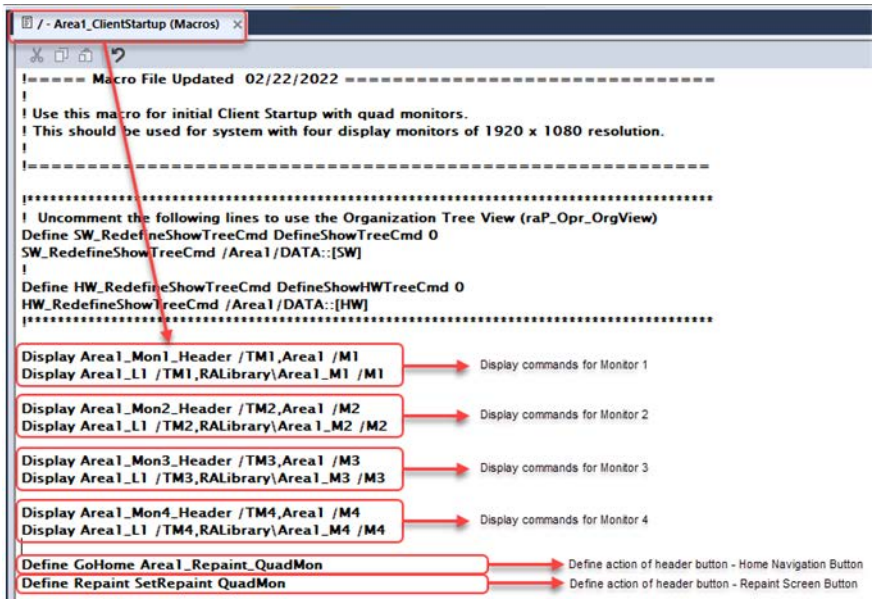


Table 2 - Area1_Repaint_QuadMon

Command	Description
Abort * /D	Abort all current displays running on the Quad monitor client
Display Area1_Mon1_Header /TM1,Area1 /M1 Display Area1_L1 /TM1, RALibrary\Area1_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area1_Mon2_Header /TM2,Area1 /M2 Display Area1_L1 /TM2,RALibrary\Area1_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.
Display Area1_Mon3_Header /TM3,Area1 /M3 Display Area1_L1 /TM3,RALibrary\Area1_M3 /M3	Display commands for monitor 3 (header and process display) with required tag parameters.
Display Area1_Mon4_Header /TM4,Area1 /M4 Display Area1_L1 /TM4,RALibrary\Area1_M4 /M4	Display commands for monitor 4 (header and process display) with required tag parameters.

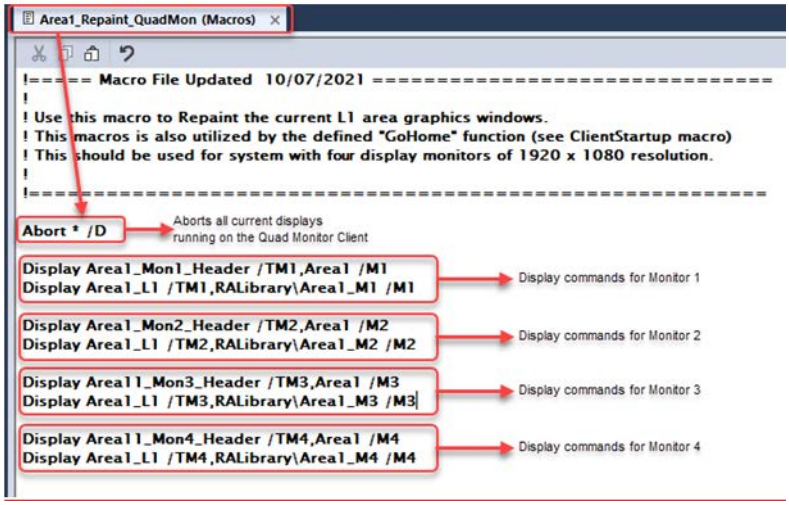


Table 3 - Area1_Repaint_DualMon

Command	Description
Abort * /D	Abort all current displays running on the dual monitor client
Display Area1_Mon1_Header /TM1,Area1 /M1 Display Area1_L1 /TM1, RALibrary\Area1_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area1_Mon2_Header /TM2,Area1 /M2 Display Area1_L1 /TM2,RALibrary\Area1_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.

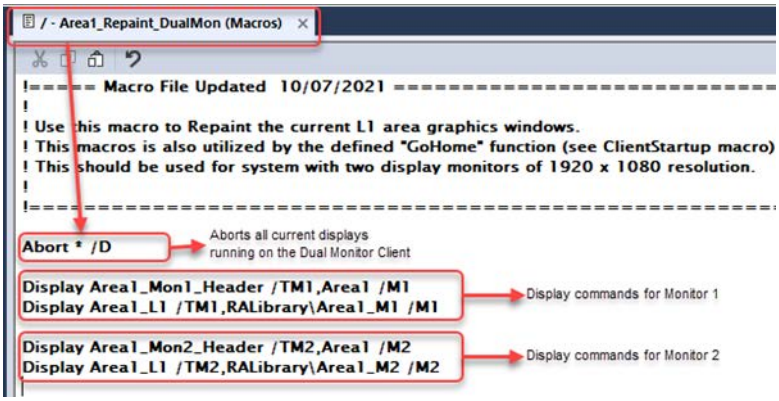
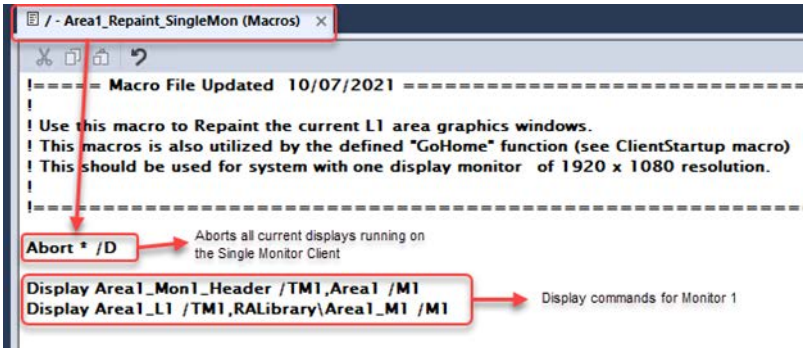


Table 4 - Area1_Repaint_SingleMon

Command	Description
Abort * /D	Abort all current displays running on the single monitor client
Display Area1_Mon1_Header /TM1,Area1 /M1 Display Area1_L1 /TM1, RALibrary\Area1_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.



Area 2 Macro Files

Table 5 - Area2_ClientStartup

Command	Description
Display Area2_Mon1_Header /TM1,Area2 /M1 Display Area2_L1 /TM1,RALibrary\Area2_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area22_Mon2_Header /TM2,Area2 /M2 Display Area2_L1 /TM2,RALibrary\Area2_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.
Define GoHome Area2_Repaint_DualMon	Define action of "GoHome" symbol used on Home Navigation Button
Define Repaint SetRepaint DualMon	Define action of "Repaint" symbol that is used on Repaint Button and L1 Navigation

Table 6 - Area2_Repaint_QuadMon

Command	Description
Abort * /D	Abort all current displays running on the quad monitor client
Display Area2_Mon1_Header /TM1,Area2 /M1 Display Area2_L1 /TM1,RALibrary\Area2_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area2_Mon2_Header /TM2,Area2 /M2 Display Area2_L1 /TM2,RALibrary\Area2_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.
Display Area2_Mon2_Header /TM3,Area2 /M3 Display Area2_L1 /TM3,RALibrary\Area2_M3 /M3	Display commands for monitor 3 (header and process display) with required tag parameters.
Display Area2_Mon2_Header /TM4,Area2 /M4 Display Area2_L1 /TM4,RALibrary\Area2_M4 /M4	Display commands for monitor 4 (header and process display) with required tag parameters.

Table 7 - Area2_Repaint_DualMon

Command	Description
Abort * /D	Abort all current displays running on the dual monitor client
Display Area2_Mon1_Header /TM1,Area2 /M1 Display Area2_L1 /TM1,RALibrary\Area2_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area2_Mon2_Header /TM2,Area2 /M2 Display Area2_L1 /TM2,RALibrary\Area2_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.

Table 8 - Area2_Repaint_SingleMon

Command	Description
Abort * /D	Abort all current displays running on the single monitor client
Display Area2_Mon1_Header /TM1,Area2 /M1 Display Area2_L1 /TM1,RALibrary\Area2_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.

Area 3 Macro Files

Table 9 - Area3_ClientStartup

Command	Description
Display Area3_Mon1_Header /TM1,Area3 /M1 Display Area3_L1 /TM1,RALibrary\Area3_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Define GoHome Area3_Repaint_SingleMon	Define action of "GoHome" symbol used on Home Navigation Button
Define Repaint SetRepaint SingleMon	Define action of "Repaint" symbol that is used on Repaint Button and L1 Navigation

Table 10 - Area3_Repaint_QuadMon

Command	Description
Abort * /D	Abort all current displays running on the quad monitor client
Display Area3_Mon1_Header /TM1,Area3 /M1 Display Area3_L1 /TM1,RALibrary\Area3_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area3_Mon2_Header /TM2,Area3 /M2 Display Area3_L1 /TM2,RALibrary\Area3_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.
Display Area3_Mon3_Header /TM3,Area3 /M3 Display Area3_L1 /TM3,RALibrary\Area3_M3 /M3	Display commands for monitor 3 (header and process display) with required tag parameters.
Display Area3_Mon4_Header /TM4,Area3 /M4 Display Area3_L1 /TM4,RALibrary\Area3_M4 /M4	Display commands for monitor 4 (header and process display) with required tag parameters.

Table 11 - Area3_Repaint_DualMon

Command	Description
Abort * /D	Abort all current displays running on the dual monitor client
Display Area3_Mon1_Header /TM1,Area3 /M1 Display Area3_L1 /TM1,RALibrary\Area3_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.
Display Area3_Mon2_Header /TM2,Area3 /M2 Display Area3_L1 /TM2,RALibrary\Area3_M2 /M2	Display commands for monitor 2 (header and process display) with required tag parameters.

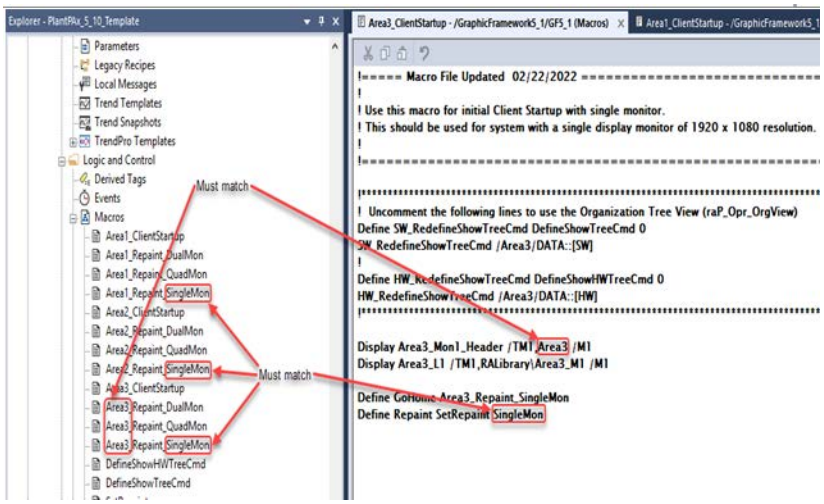
Table 12 - Area3_Repaint_SingleMon

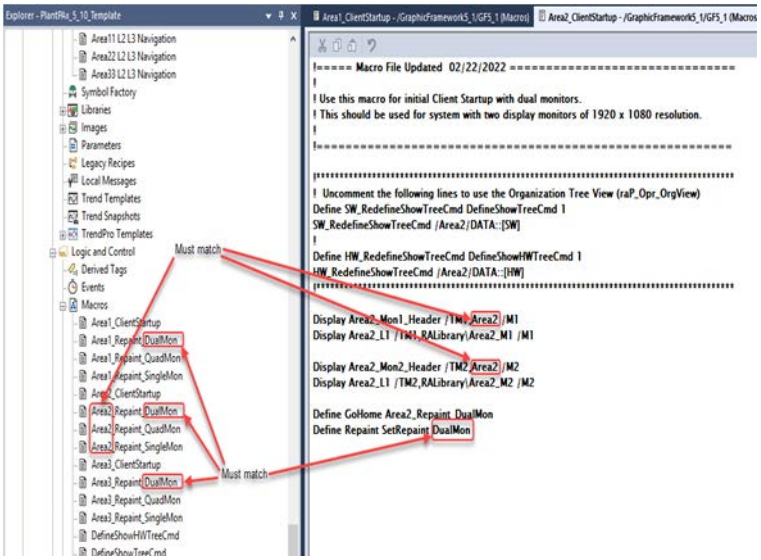
Command	Description
Abort * /D	Abort all current displays running on the dual monitor client
Display Area3_Mon1_Header /TM1,Area3 /M1 Display Area3_L1 /TM1,RALibrary\Area3_M1 /M1	Display commands for monitor 1 (header and process display) with required tag parameters.

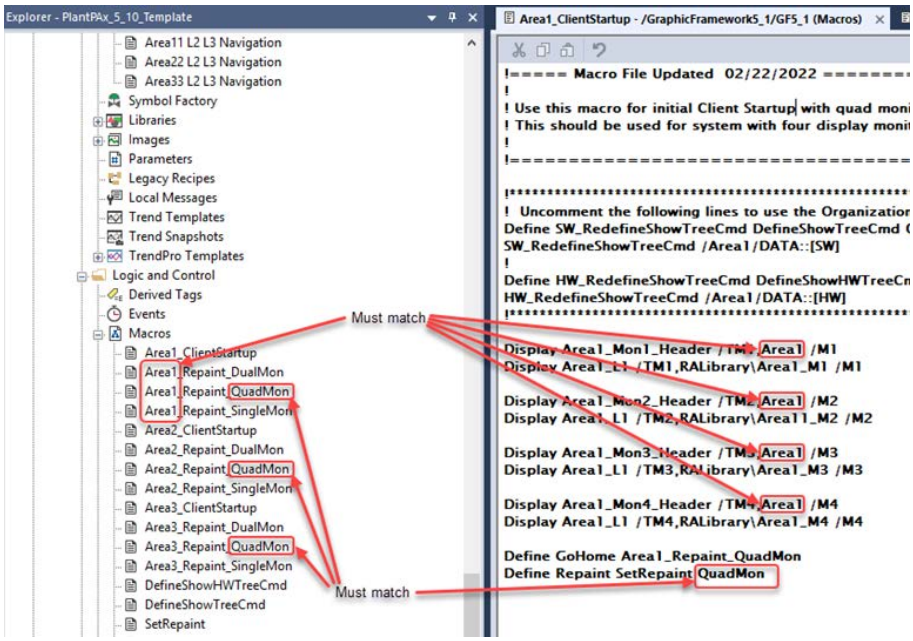
There is a naming convention that must be followed when creating each repaint macro. The file name must be in the following format, as shown in the macro file SetRepaint:

```
!===== SetRepaint created  05/20/2022 ================================
! Builds the Repaint Macro to be used by specifc client based on the current
!    L1 area and monitor quantity (defined from startup client macro)
!=====================================================================

! Parameters
!     %1 - Monitor Quantity (i.e. "QuadMon" or "4Mon, "DualMon" or "2Mon, "SingleMon" or "1Mon")
!     %2 - Area Name

%2_Repaint_%1
```

The %1 and %2 are defined in each client startup macro, as shown in the following examples.

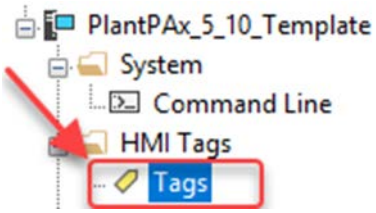




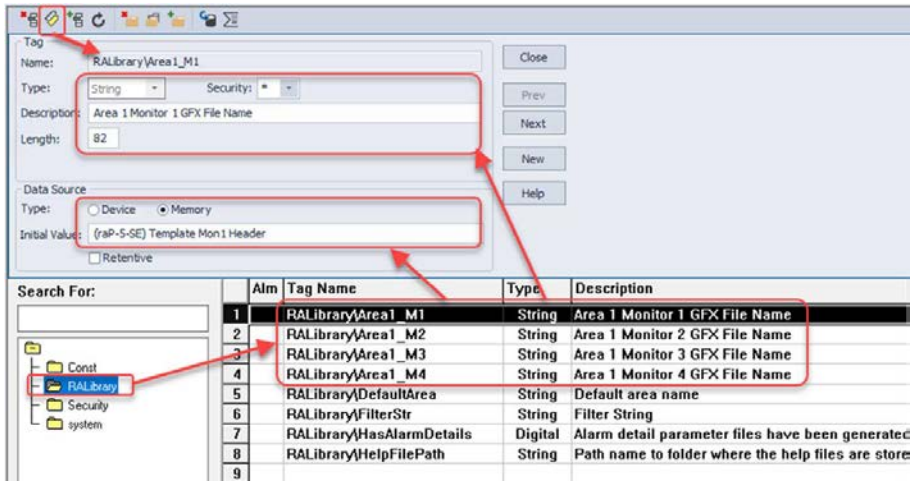


Create HMI Tags for Multi-Monitor and Repaint

1. Open Tags under HMI Tags in the application in FactoryTalk View Studio.



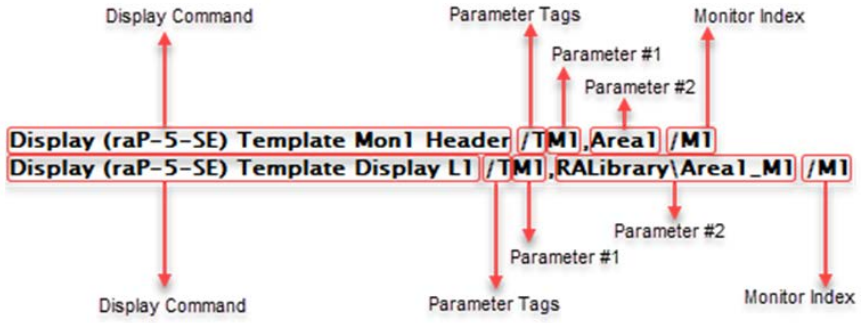
2. Open the folder "RALibrary". There are four sample tags that are created in this folder. Duplicate each tag – groupings of four tags for each L1 area, one for each possible monitor. "Area1" should be replaced with the name of your L1 area. Update the description field and enter the header file display name (*.gfx) in the initial value field for each header in the appropriate tag. Repeat for each L1 area.



Parameter Explanation

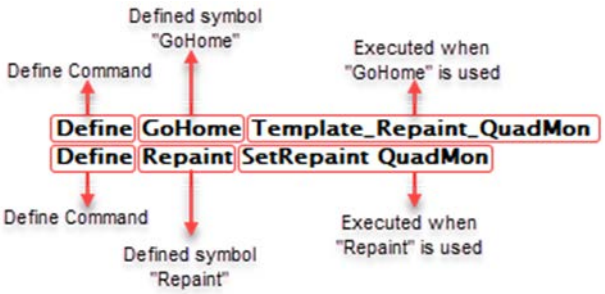
Client Startup Macro

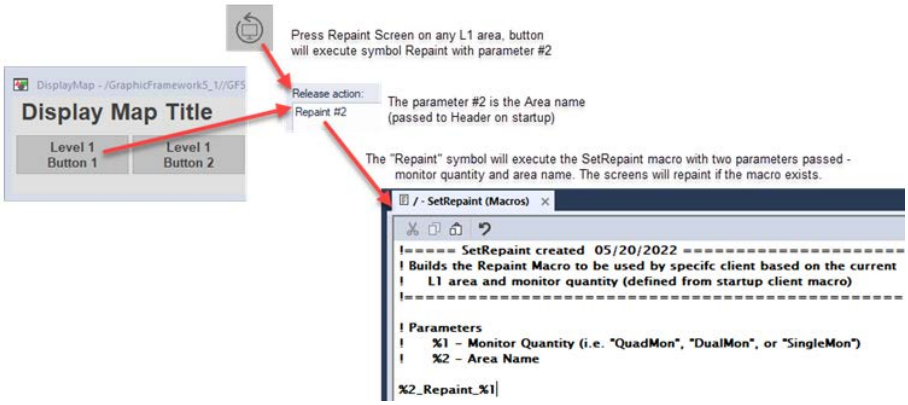
Each monitor has two button command lines that are associated with the first time that the displays are opened on the given monitor of the client. It is important that these parameters are configured correctly in the client startup macro and when switching between L1 areas. The following is an explanation of what the commands and parameters mean for each monitor, in this example for monitor 1. This needs to be duplicated for each monitor used in the client.



Display Command	Item	Description and Configuration
Display (Header Display)	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	M1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the Header Display. The parameter must be updated to M1, M2, M3, or M4 depending on which monitor the display opens on.
	Area1	Parameter #2 - Parameter is used to pass the current L1 area name to commands executed on the Header Display. The parameter must correspond to the area name for that L1 and match the area name that is used in the HMI Tags for multimonitor.
	/M1	Monitor Index - A built-in parameter in FactoryTalk View to command the display to open on a particular monitor. The parameter must be updated to /M1, /M2, /M3, or /M4 depending on which monitor it opens on.
Display (Process Display)	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	M1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the Process Display. The parameter must be updated to M1, M2, M3, or M4 depending on which monitor the display will open on.
	RALibrary\Area1_M1	Parameter #2 - Parameter is used to pass the HMI tag name for the current L1 area and current monitor. The value of the tag will be used to identify what the Header display file name is for that monitor. The parameter must be updated to the specific L1 area and monitor for this client. See HMI Tags, Headers, and Macrosfor more details on how to configure the HMI tags for multi-monitor.
	/M1	Monitor Index - A built-in parameter in FactoryTalk View to command the display to open on a particular monitor. The parameter must be updated to /M1, /M2, /M3, or /M4 depending on which monitor it opens on.

There are two defined symbols in the startup macro that are crucial for the client to operate as expected. As noted in FactoryTalk View Studio help documentation, the system command Define creates a symbol at run time on the FactoryTalk View SE Client. This command is only run on the FactoryTalk View SE Client. Symbol definitions are valid only during the current client session; they must be redefined each time that the client is restarted. Symbols are typically defined in a startup or login macro.

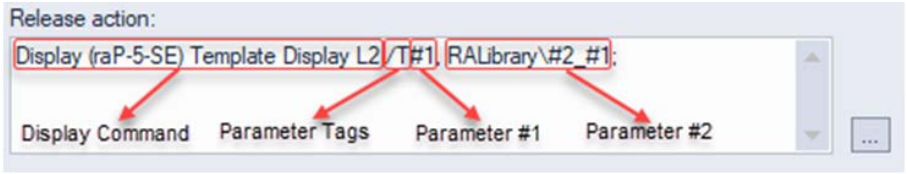




Define Command	Description and Configuration
Define GoHome	The "GoHome" symbol is used exclusively for the Home Button that is used on the header display. The "GoHome" symbol is defined when the client starts up and the definition is specific to that client. Regardless of which L1 area an operator might end up navigating to, when the Home Button is pressed, it executes the defined macro. The macro that is used for this symbol should be configured to open on the correct number of monitors for that client. The "GoHome" symbol should be used on every startup macro. Replace "Template_Repaint_QuadMon", "Template_Repaint_DualMon", or "Template_Repaint_SingleMon" with the Repaint macro to be used for this specific client.
Define Repaint	The "Repaint" symbol is used for both the Repaint Screen Button header display and L1 Navigation from the display map pop-up. The "Repaint" symbol is defined when the client starts up and the definition is specific to that client. When navigating to another L1 area or repainting the screen on any area, the symbol executes the pre-defined macro "SetRepaint", with two parameters passed to the macro. %1 Parameter - This is configured in the definition of in the startup client and should be either "QuadMon", "DualMon", or "SingleMon" depending on how many monitors that client has. User must update this in the startup client file. %2 Parameter - This is passed automatically when the button is pressed. No configuration is required as long as the display commands are configured as required in the startup macro.

L2 Navigation

Each button on the L2 Navigation bar needs the following button command constructed to pass the proper Parameter Tags to the process display.



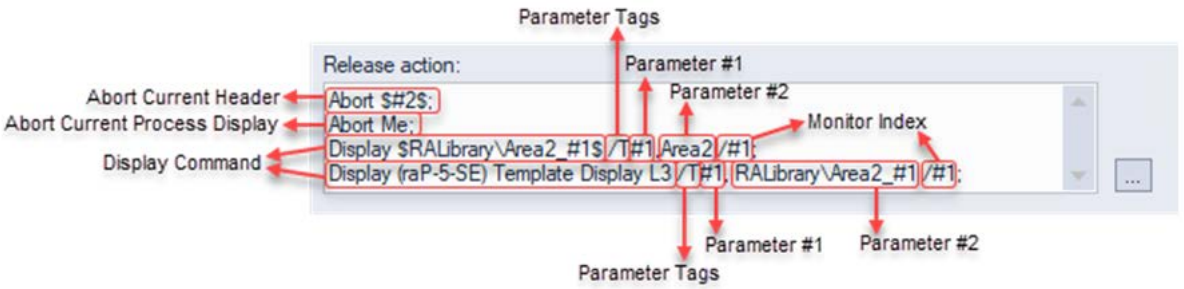
Command	Item	Description and Configuration
Display [L2 Process Display]	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	#1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the L2 process display. The "#1" will use the Parameter #1 pushed into the Header Display when it was initially opened. No configuration required; leave #1 as is.
	RALibrary\#2_#1	Parameter #2 - Parameter is used to pass the HMI tag name for the current L1 area and current monitor. The value of the tag will be used to identify what the Header display file name is for that monitor. The "#1" and "#2" will use the Parameter #1 and Parameter #2 pushed into the Header Display when it was initially opened. No configuration required; leave #1 and #2 as is. See HMI Tag, Headers, and Macros for more details on how to configure the HMI tags for multi-monitor.

L3 Navigation

Each button on the L3 Navigation bar needs the following button command constructed to pass the proper Parameter Tags to the process display. This will also be used for any Off-Screen navigation buttons within the same L1 area.

Command	Item	Description and Configuration
Display [L3 Process Display]	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	#1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the L3 process display. The "#1 uses the Parameter #1 pushed into the Process Display when it was initially opened. No configuration required; leave #1 as is.
	#2	Parameter #2 - Parameter is used to pass the HMI tag name for the current L1 area and current monitor. The value of the tag is used to identify what the Header display file name is for the monitor active for this display. The "#2 uses the Parameter #2 pushed into the Process Display when it was initially opened. No configuration required; leave #2 as is. See HMI Tags, Headers, and Macros for more details on how to configure the HMI tags for multi-monitor.

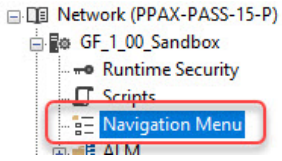
Off-Screen Navigation



Command	Item	Description and Configuration
Abort [Header Display]	Abort $#2$	This command aborts the header that is open above the process display. The "$#2$" inserts the value of the parameter "#2" that was passed into the display that the button is executing from. The value of "#2" is the HMI tag for the header display file name.
Abort [Process Display]	Abort Me	This command aborts the process display that the button is executing from.
Display [Header Display]	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	#1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the Header Display. In this instance, we take the parameter that is passed from when the display was originally opened. No configuration required.
	Area2	Parameter #2 - Parameter is used to pass the destination L1 area name to commands executed on the destination Header Display of the other L1 area. The parameter must correspond to the area name for the destination L1 area and match the area name that is used in the HMI Tags for multimonitor.
	/#1	Monitor Index - A built-in parameter in FactoryTalk View to command the display to open on a particular monitor. In this instance, it takes the parameter that is passed from when the display was originally opened. No configuration required.
Display [Process Display]	/T	Parameter Tags - A built-in parameter that indicates the next items that are listed in the command will be Parameter Tags. No configuration required.
	#1	Parameter #1 - Parameter is used to pass the monitor number to commands executed on the Header Display. In this instance, we take the parameter that is passed from when the display was originally opened. No configuration required.
	RALibrary\Area2_#1	Parameter #2 - Parameter is used to pass the HMI tag name for the destination L1 area and current monitor. The value of the tag is used to identify what the Header display file name is for that monitor in the other L1 area. The parameter must be updated to the specific L1 area, but the "#1" should be left as is. See HMI Tags, Headers, and Macros for more details on how to configure the HMI tags for multi-monitor.
	/#1	Monitor Index - A built-in parameter in FactoryTalk View to command the display to open on a particular monitor. In this instance, it takes the parameter that is passed from when the display was originally opened. No configuration required.

Navigation Menu

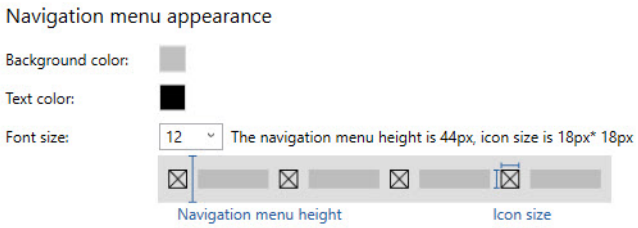
FactoryTalk View SE has a built-in navigation menu that can be used with the Graphic Framework v1.00 in place of the L2 and L3 navigation, with one navigation menu configured for each L1 area. The navigation menu is configured directly in FactoryTalk View SE.



The following are recommendations for how to configure the navigation menu when using it with the Graphic Framework [v1.00]. These directions assume that all L1, L2, L3, and header displays have been populated for the user's application. When using the navigation menu, the headers should be developed using the template file "(raC-1_00-SE) Template Header Nav Menu" and L1, L2, and L3 displays should be developed using the template file "(raC-1_00-SE) Template Display Nav Menu".

1. Open the Navigation Menu by double-clicking it in the application explorer.
2. Select the "Add navigation menu" button  . Create a navigation menu for each L1 area.
3. Double-click one of the navigation menus created in the previous step. The following settings are recommended for each navigation menu.

Property	Recommended setting
Background color	RGB {192,192,192}
Text color	RGB {0,0,0}
Font size	12
Enable Backward, Forward, and History buttons	Feature is optional
Enable search	Feature is optional

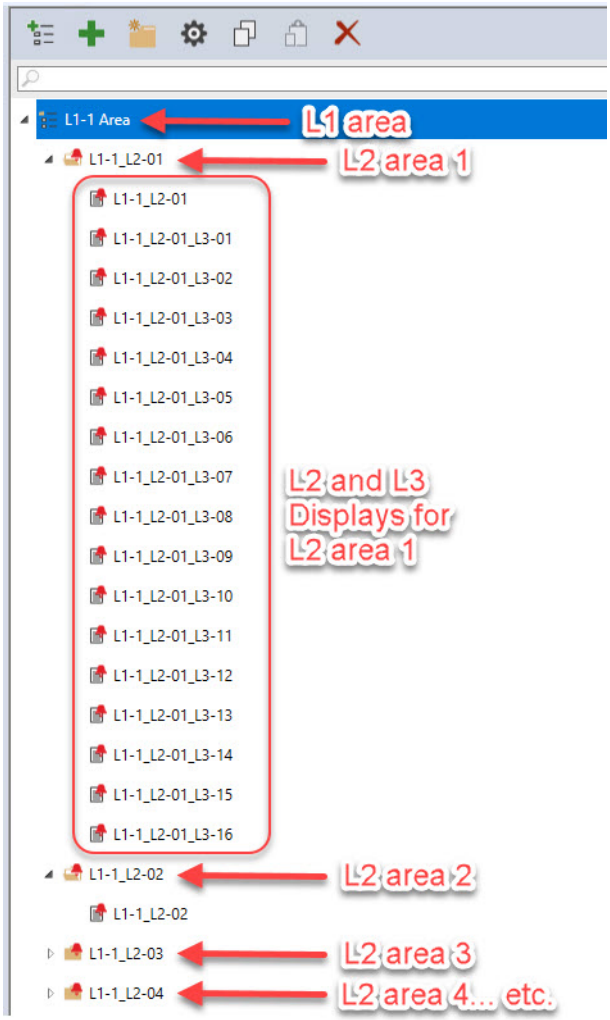


4. Click the OK button to save the settings and return to the main configuration menu.
5. Click the "Add menu items" button  . Add all L2 and L3 displays that are available in this L1 area.

Note: It is OK if the displays are added in any order. They can be reorganized after being added.

You can save time by pre-organizing your L2 and L3 into L2 folders in application explorer before adding the display to the navigation menu and the tool pre-populates the navigation folders in the same way.

6. Organize your navigation menu into folders if they are not already organized. Each L2 area should have a folder as shown.



7. Click the first L2 folder and then select "Settings". Add the following alarm source to create alarm indication roll-up on that folder. Replace "L1_Name1" with the alarm group name for this L1 area and replace "L2_Name01" with the alarm group name for this L2 area.

   L1_Name1.L2_Name01::*

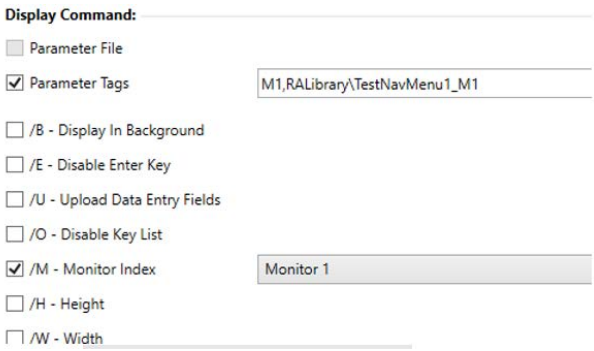
IMPORTANT	Include the "*" as shown. This is a wildcard that allows any alarm in these groups to indicate on this folder.

8. You can optionally add a device path to create automatic diagnostic indication on the folder. Click OK when done configuring the L2 folder settings to return to the main navigation menu configuration window.
9. Repeat the previous two steps for each L2 folder.
10. Click the first L3 display in the first L2 folder and then select "Settings". Click "Notification" tab at the top. Add the following alarm source to create alarm indication roll-up on that display. Replace "L1_Name1" with the alarm group name for this L1 area, replace "L2_Name01" with the alarm group name for this L2 area, and replace "L3_Name01" with the alarm group name for this L3 area.

    L1_Name1.L2_Name01.L3_Name01::*

IMPORTANT	Include the "*" as shown. This is a wildcard that allows any alarm in these groups to indicate on this folder.

11. You can optionally add a device path to create automatic diagnostic indication on the display. Click the "Command" tab to return back to the main L3 display settings.
12. Verify that the "Name" and "Linked Display" are correct.
13. Add the following "Parameter Tags" and "Monitor Index". Click OK to close the display settings once the display commands are configured.



Note: See section Create HMI Tags for Multi-Monitor and Repaint on page 106 for details on creating HMI tags to allow navigation to function properly between L1 areas.

Note: Only single monitor is supported with navigation menu and Graphic Framework v1.00. The monitor index selection is recommended to be configured for future multimonitor functionality.

14. Repeat steps 10...13 for each L2 display call under each L2 folder. The alarm source entry should match step 7. The display commands should match step 13.
15. Exit the Navigation Menu configuration when all folders and displays are configured.

    Make sure that the following macro files are used to create the application macros when using the Navigation Menu:

- Template_NavMenu_ClientStartup_SingleMon
- Template_NavMenu_Repaint_SingleMon

These macros contain the correct call for header, navigation menu, and main display.

The following is a summary of pros and cons to using the Navigation Menu compared to legacy button navigation with L2 and L3 button bars:

Pros	Cons
• Built into FactoryTalk View SE • Configurable in one place • Built-in tag search • Built-in forward, backward, and historical navigation • Unlimited number of displays per L2 and L3 area • Simple to move, add, and reconfigure L2 and L3 areas • Easily configure display commands directly in the navigation menu configuration • Built in alarm source and automatic diagnostic indication and rollup • Option to use icons instead of text on dropdown and display navigation	• Currently only supported for single monitor with the Graphic Framework • No tool to bulk configure the navigation - must manually be configured directly in FactoryTalk View SE • If more L2 folder areas are configured than fit on a single monitor, the alarm indication is not visible for those L2 areas.

FactoryTalk Resource & Status Server Configuration

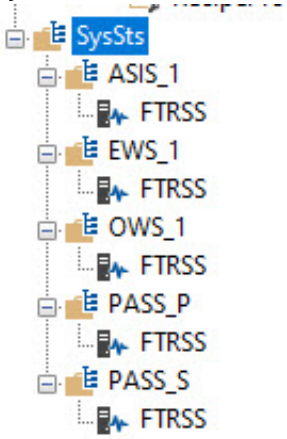
The FactoryTalk Resource and Status server (FTRS server) is a new feature available starting in FactoryTalk View SE v15 with FactoryTalk Services Platform v6.50 and later. Faceplates and global object for use with FactoryTalk Resource and Status server is only available with the Graphic Framework v1.00. The following section outlines recommendations for setup and configuration of the Resource and Status server on your system.

Setup

The FactoryTalk Resource and Status server can monitor three main items:

- Performance of the workstation it is installed on, including CPU usage, storage space, and memory capacity.
- Status of specific system processes of the workstation if configured to do so.
- Connection of the workstation to other devices in the system including other workstations, network switches, and infrastructure components.

The Resource and Status server can be installed on any workstation or virtual machine that is joined to the FactoryTalk Directory. It is recommended to place each instance of the Resource and Status server in a separate area - this will optimize data reference lookup. Here is an example of a recommended configuration:



Device Status Configuration

On the "Device Status" tab of the FTRS server configuration, network connection to different devices can be set up. Here are some recommendations for the device status configuration:

- It is best practice to keep the update rates slow (large intervals between pings and large intervals of each cycle) to reduce the loading on the system network.
- It is recommended to configure a device once per system. For instance, pick the workstation that hosts the FactoryTalk Directory and check the connection from that server to all other devices in the system from there. This reduces duplicate network connection checking to the same device.
- Fill out the Alias Name so that it is populated on the global object.

Process Configuration

On the "Process" tab of the FTRS server configuration, specific tasks or processes running on a system can be set up to be monitored. Here is a list of recommended tasks to monitor based on typical servers and workstations in a system:

System Device	Process Name	Purpose
FactoryTalk Directory server	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	

System Device	Process Name	Purpose
Process Automation System Server (PASS)	DatalogServ.exe	HMI Failure
	DataLogProPlusReadService.exe	
	DataLogProPlusServer.exe	
	DataLogProPlusService.exe	
	DatalogReadExService.exe	
	EventDetector.exe	
	FTViewServiceHost.exe	
	HMIServer.exe	
	LocalDBOps64.exe	
	RuntimeSearch.Service.Host.exe	
	RuntimeSearchListenService.exe	
	SAUserServ.exe	
	ServerFramework.exe	
	ServerScriptService.exe	
	TagSrv.exe	
	ViewSharedService.exe	
	w3wp.exe	
	RSLinxNG.exe	FactoryTalk Linx Failure
	RnaAeServer.exe	FTAE Failure
	RnaAlarmDetector.exe	
	FTAE_HistServ.exe	
	DataAccessServiceHost.exe	
	FTAEArchiver.exe	
	FTAE.Web.DataProvider.Service.exe	
	FTAE.Web.DataProvider.RunTime.exe	
	FTAE.Web.DataProvider.Historian.exe	
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
Thin Manager server	ThinServer.exe	TM Failure
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
Engineering Workstation	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
Operator Workstation	DisplayClient.exe	Client Failure
	SAUserServ.exe	
	SEGfxVBACli.exe	
	CommandCliSrv.exe	
	RPMClientSideService.exe	
	RSAOAServer.exe	Activation Failure
	RnaAlarmMux.exe	FTAE Failure
	FTAEExpressionServer.exe	
	FTAECommandServer.exe	
	RnaAlarmMailRuleConfig.exe	
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
Information Management Server (SQL)	sqlservr.exe	SQL Failure
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	

System Device	Process Name	Purpose
Information Management Server (FactoryTalk Historian)	pibasess.exe	FactoryTalk Historian Failure
	pilogsrv.exe	
	pinetmgr.exe	
	pisnapss.exe	
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
FactoryTalk Asset Centre Server	RA.FTAC.Server.exe	FTAC Server Failure
	Rsvchost.exe	FTAC Diagnostic Failure
	w3wp.exe	FTAC Web Service Failure
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	
FactoryTalk Asset Centre Agent	RA.FTAC.AgentControllerService.exe	FTAC Agent Controller Failure
	RA.FTAC.InventCrawler.Server.exe	FTAC Asset Inventory Disaster Recovery Failure
	RA.FTAC.MonitorRuntime.exe	FTAC Change Detect Service Failure
	Rsvchost.exe	FTAC Diagnostic Failure
	VerificationAgent.exe	FTAC Logix Disaster Recovery Failure
	RnaDirServer.exe	FTD Failure
	RNADirMultiplexor.exe	

Macros

Macros are an important component in the graphic framework. There are several macros that are provided as a template.

- Template_ClientStartup_SingleMon
- Template_ClientStartup_DualMon
- Template_ClientStartup_QuadMon
- Template_NavMenu_ClientStartup_SingleMon
- Template_Repaint_SingleMon
- Template_Repaint_DualMon
- Template_Repaint_QuadMon
- Template_NavMenu_Repaint_SingleMon
- SetRepaint

The following are optional macros that are used for applications using both Process Library 4.10 and Process Library 5.00 or later.

- NavToFaceplate with mixed library
- NavToDisplay with mixed library

Template_ClientStartup

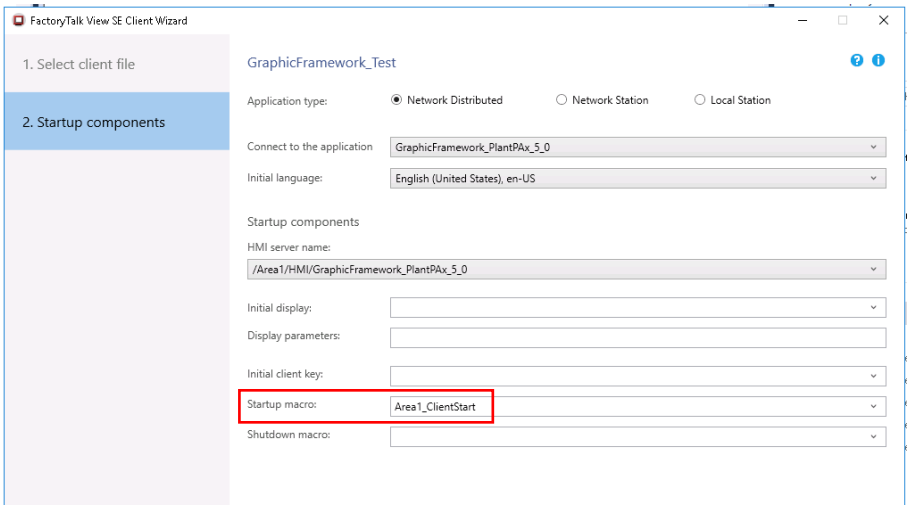
Number of Monitors	Commands
1	!===== Macro File Updated 02/22/2022 ========== ! ! Use this macro for initial Client Startup with single monitor. ! This should be used for system with a single display monitor of 1920 x 1080 resolution. ! !========== !********** ! Uncomment the following lines to use the Organization Tree View (raP_Opr_OrgView) ! Define SW_RedefineShowTreeCmd DefineShowTreeCmd 0 ! SW_RedefineShowTreeCmd /Area1/DATA::[Hardware] ! ! Define HW_RedefineShowTreeCmd DefineShowHWTreeCmd 0 ! HW_RedefineShowTreeCmd /Area1/DATA::[Hardware] !********** Display (raP-5-SE) Template Mon1 Header /TM1,Area1 /M1 Display (raP-5-SE) Template Display L1 /TM1,RALibrary\Area1_M1 /M1 Define GoHome Template_Repaint_SingleMon Define Repaint SetRepaint SingleMon
1 - With Navigation Menu	! — Template_NavMenu_ClientStartup_SingleMon Macro — Revision 1.00-00 ———— ! ! Use this macro for initial Client Startup with single monitor. ! This should be used for system with a single display monitor of 1920 x 1080 resolution. ! !========== Display (raC-1_00-SE) Template Header Nav Menu /TM1,Template_NavMenu /M1 /DT DisplayNavigationMenu "L1-1 Area" /DT Display (raC-1_00-SE) Template Display Nav Menu /TM1,RALibrary\Template_NavMenu_M1 /M1 Define GoHome Template_NavMenu_Repaint_SingleMon Define Repaint SetRepaint SingleMon
2	!===== Macro File Updated 02/22/2022 ========== ! ! Use this macro for initial Client Startup with dual monitors. ! This should be used for system with two display monitors of 1920 x 1080 resolution. ! !========== !********** ! Uncomment the following lines to use the Organization Tree View (raP_Opr_OrgView) ! Define SW_RedefineShowTreeCmd DefineShowTreeCmd 0 ! SW_RedefineShowTreeCmd /Area1/DATA::[Hardware] ! ! Define HW_RedefineShowTreeCmd DefineShowHWTreeCmd 0 ! HW_RedefineShowTreeCmd /Area1/DATA::[Hardware] !********** Display (raP-5-SE) Template Mon1 Header /TM1,Area1 /M1 Display (raP-5-SE) Template Display L1 /TM1,RALibrary\Area1_M1 /M1 Display (raP-5-SE) Template Mon2 Header /TM2,Area1 /M2 Display (raP-5-SE) Template Display L1 /TM2,RALibrary\Area1_M2 /M2 Define GoHome Template_Repaint_DualMon Define Repaint SetRepaint DualMon

Number of Monitors	Commands
4	!===== Macro File Updated 02/22/2022 ========== ! ! Use this macro for initial Client Startup with quad monitors. ! This should be used for system with four display monitors of 1920 x 1080 resolution. ! !========== !********** ! Uncomment the following lines to use the Organization Tree View (raP_Opr_OrgView) ! Define SW_RedefineShowTreeCmd DefineShowTreeCmd 0 ! SW_RedefineShowTreeCmd /Area1/DATA::[Hardware] ! ! Define HW_RedefineShowTreeCmd DefineShowHWTreeCmd 0 ! HW_RedefineShowTreeCmd /Area1/DATA::[Hardware] !********** Display (raP-5-SE) Template Mon1 Header /TM1,Area1 /M1 Display (raP-5-SE) Template Display L1 /TM1,RALibrary\Area1_M1 /M1 Display (raP-5-SE) Template Mon2 Header /TM2,Area1 /M2 Display (raP-5-SE) Template Display L1 /TM2,RALibrary\Area1_M2 /M2 Display (raP-5-SE) Template Mon3 Header /TM3,Area1 /M3 Display (raP-5-SE) Template Display L1 /TM3,RALibrary\Area1_M3 /M3 Display (raP-5-SE) Template Mon4 Header /TM4,Area1 /M4 Display (raP-5-SE) Template Display L1 /TM4,RALibrary\Area1_M4 /M4 Define GoHome Template_Repaint_QuadMon Define Repaint SetRepaint QuadMon

The Client Startup macro should be linked to the Startup Macro selected in the client file configuration. There should be at least one Client Startup macro for every L1 area.

```
Define SW_RedefineShowTreeCmd DefineShowTreeCmd 0
SW_RedefineShowTreeCmd /Area1/DATA::[Hardware]

Define HW_RedefineShowTreeCmd DefineShowHWTreeCmd 0
HW_RedefineShowTreeCmd /Area1/DATA::[Hardware]
```

The two "Define" functions that are shown in the preceding screenshot are used to configure the Client Startup Macro for use with the Hardware and Software Tree Views. For each client used, the number at the end of these "Define" calls should increment by one (for example, if you have five clients in a system, each client would be assigned a different number: 0, 1, 2, 3, 4, etc). The shortcut that is defined for each in the second line should be a valid shortcut that is used for to initialize on. The shortcut should include the full area and short name.



The main purpose of this macro is to open the header and the L1 overview display for each monitor. The specific displays must be updated for each macro that is created to point to the Header and screen for that L1 area.

The macro is also used to define the GoHome and Repaint symbol commands. GoHome is used for Home button on the Header. The definition of the GoHome has to be updated to point to the specific L1 area and client repaint macro. The Repaint symbol is used for Repaint Screens button as well as the L1 Navigation. The number of monitors that are used by the client should be updated here (QuadMon, DualMon, or SingleMon)

Template_Repaint

The repaint macro is identical to the client startup macro, except the symbol definitions are not executed. At least one repaint macro should be created for every L1 area. If there multiple clients per L1 area with differing number of monitors, one repaint macro per each client, with specified monitor quantity, should be created for each L1 area. The repaint macro is used by the Repaint symbol, which executes the macro SetRepaint. The repaint macro should be created regardless of if the Repaint button is used, because it will be used by the Home button and for navigation between L1 areas.

The following example shows a system with four display monitors.

```
!===== Macro File Updated  10/07/2021 ================================
!
! Use this macro to Repaint the current L1 area graphics windows.
! This macros is also utilized by the defined "GoHome" function (see ClientStartup macro)
! This should be used for system with four display monitors of 1920 x 1080 resolution.
!
!=======================================================

Abort * /D

Display (raP-5-SE) Template Mon1 Header /TM1,Area1 /M1
Display (raP-5-SE) Template Display L1 /TM1,RALibrary\Area1_M1 /M1

Display (raP-5-SE) Template Mon2 Header /TM2,Area1 /M2
Display (raP-5-SE) Template Display L1 /TM2,RALibrary\Area1_M2 /M2

Display (raP-5-SE) Template Mon3 Header /TM3,Area1 /M3
Display (raP-5-SE) Template Display L1 /TM3,RALibrary\Area1_M3 /M3

Display (raP-5-SE) Template Mon4 Header /TM4,Area1 /M4
Display (raP-5-SE) Template Display L1 /TM3,RALibrary\Area1_M4 /M4
```

SetRepaint

The SetRepaint macro is used to build the correct repaint macro to use based on the area parameter and quantity of monitors that are configured in the startup client macro. No configuration of this macro is required. It must exist in the macro list for navigation to work properly.

```
!===== SetRepaint created  05/20/2022 ======================
! Builds the Repaint Macro to be used by specifc client based on the current
!    L1 area and monitor quantity (defined from startup client macro)
!============================================================

! Parameters
!     %1 - Monitor Quantity (i.e. "QuadMon", "DualMon", or "SingleMon")
!     %2 - Area Name

%2_Repaint_%1
```

NavToDisplay with Mixed Library / NavToFaceplate with Mixed Library

The two macros "NavToDisplay with mixed library" and "NavToFaceplate with mixed library" are only necessary for applications that are using both the Process Library 4.10 and Process Library 5.00 or later. There is added logic in these two macros to ensure that navigation is possible between 4.10 object faceplates and 5.00 object faceplates and vice versa. The macros work with two redirect displays - "(raP-5_30-SE) Common-Redirect-to-4_10" and "(raP-5_30-SE) Common-Redirect-to-5_00".

The macros are used in place of the NavToDisplay and NavToFaceplate. To use:

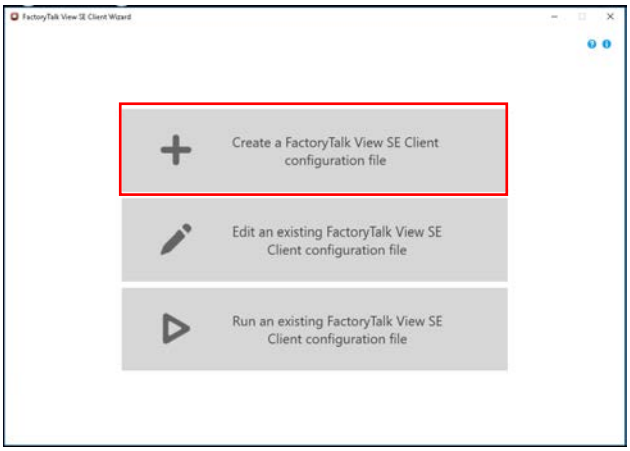
1. Rename the existing NavToDisplay and NavToFaceplate to a temporary name such as NavToDisplay_Original and NavToFaceplate_Original. This is in case the macros are needed for review in the future, they will already be available.
2. Rename "NavToDisplay with mixed library" and "NavToFaceplate with mixed library" to NavToDisplay and NavToFaceplate respectively. The macros are ready use.

```
!   Macro "NavToDisplay with mixed library"
!   version 5.10-00 Release
!   Rockwell Automation Library of Process Objects
!
!  ***  Alternate version for compatability with 4.10 Library objects  ***
!  ***  Rename this macro to "NavToDisplay" when you have 4.10 and 5.00 (or newer)  objects in the same application  ***
!
!  This macro navigates to the faceplate for the object specified by the by the given Path and Tag names
!  The parameters are separated by spaces.  Parameters are as follows:
!
!   %1 - Object Tag Name
!   %2 - Pass thru information (or "{x}" if not used)
!   %3 - Display Type
!   %4 - Display Parameter
!   %5 - Display Parameter

!   An example:
!   NavToDisplay [MyPath]MyObject {x} "Faceplate" /X100 /Y200
!
! Copyright © Rockwell Automation, Inc.  All Rights Reserved

! If the @Library Extended Tag Property is not found, then assume this is a 4.x object
If Comm_Err( {%1.@Library} ) Then
  Display (raC-5-SE) Common-Redirect-to-4_10 /T%1 /RP
Else
  Display ($%1.@Library$-SE) $%1.@Instruction$-%3 /T{Const\Num2},%1,%2,%4,%5,{x} %4 %5
Endif;


!   Macro "NavToFaceplate with mixed library"
!   version 5.10-00 Release
!   Rockwell Automation Library of Process Objects
!
!  ***  Alternate version for compatability with 5.00 or newer Library objects  ***
!  ***  Rename this macro to "NavToFaceplate" when you have 4.10 and 5.00 (or newer) objects in the same application  ***
!
!  This macro navigates to the faceplate for the object specified by the by the given Path and Tag names
!  The parameters are separated by spaces.  Parameters are as follows:
!
!   %1 - Object Tag Name
!   %2 - Pass thru information (Usually Object Path Name not including tag)
!   %3 - Display Parameter
!   %4 - Display Parameter

!   An example:
!   NavToFaceplate [MyPath]MyObject [MyPath] /X100 /Y200
!
! Copyright © Rockwell Automation, Inc.  All Rights Reserved

! If the tag HMI_Lib is not found, then assume this is a 5.x object
If Comm_Err( {%1.HMI_Lib} ) Then
  Display (raC-5-SE) Common-Redirect-to-5_00 /T%1 /RP
Else
  Display ($%1.HMI_Lib$) $%1.HMI_Type$-Faceplate /T{Const\Num2},%1,%2,%3,%4,{x} %3 %4
Endif;
```

Client File Setup (.CLI)

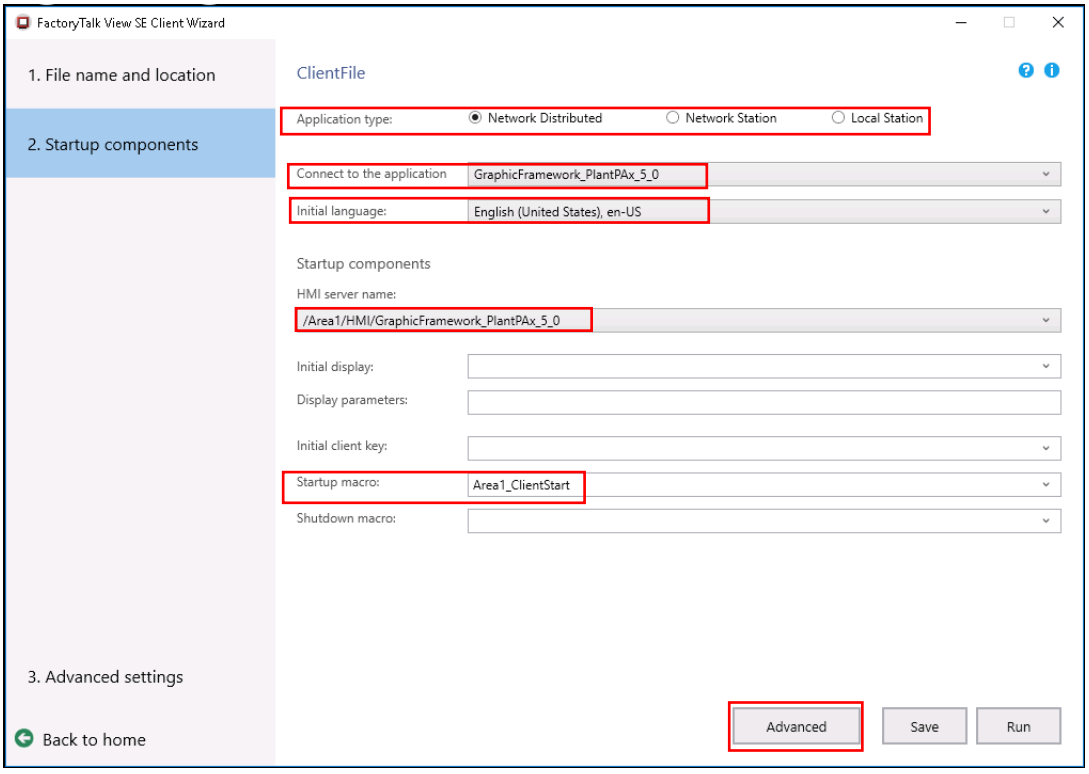
Configure a basic client file to use with the Graphic Framework.

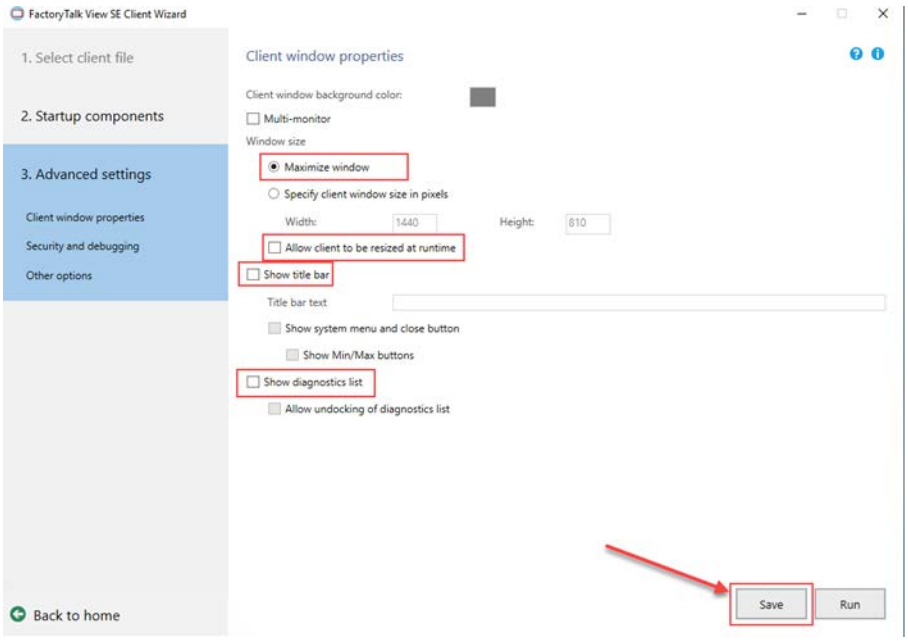
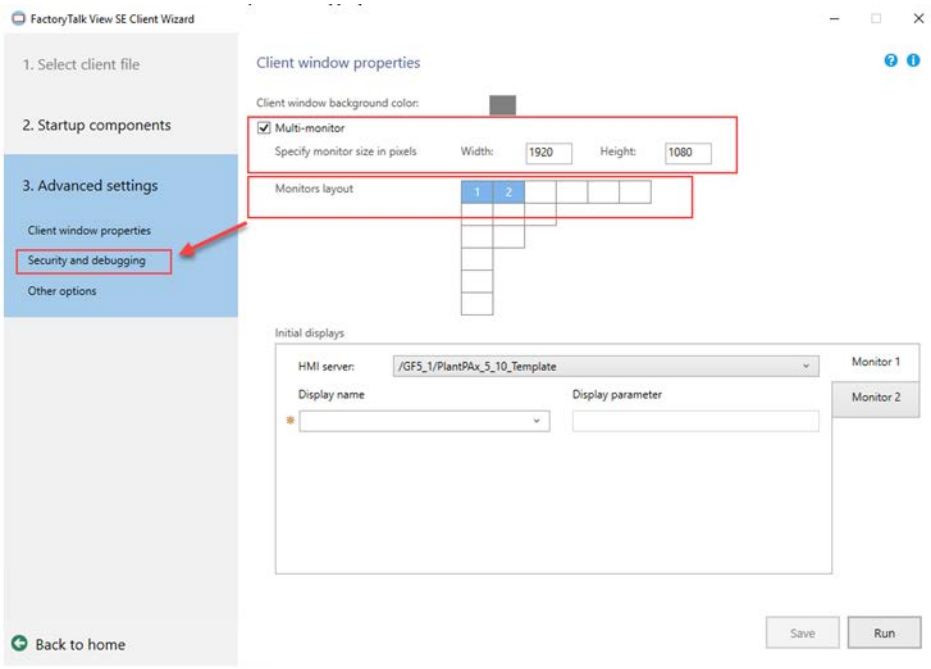
1. Go to FactoryTalk View SE Client Wizard and select Create a FactoryTalk View SE Client configuration file.

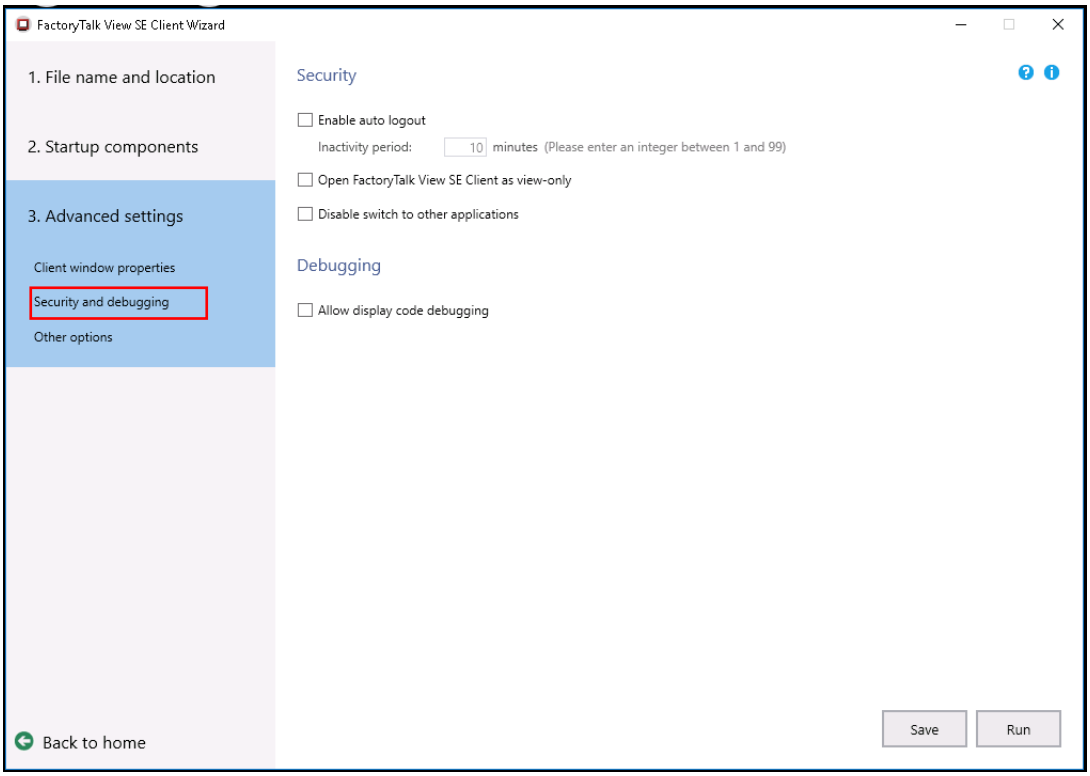
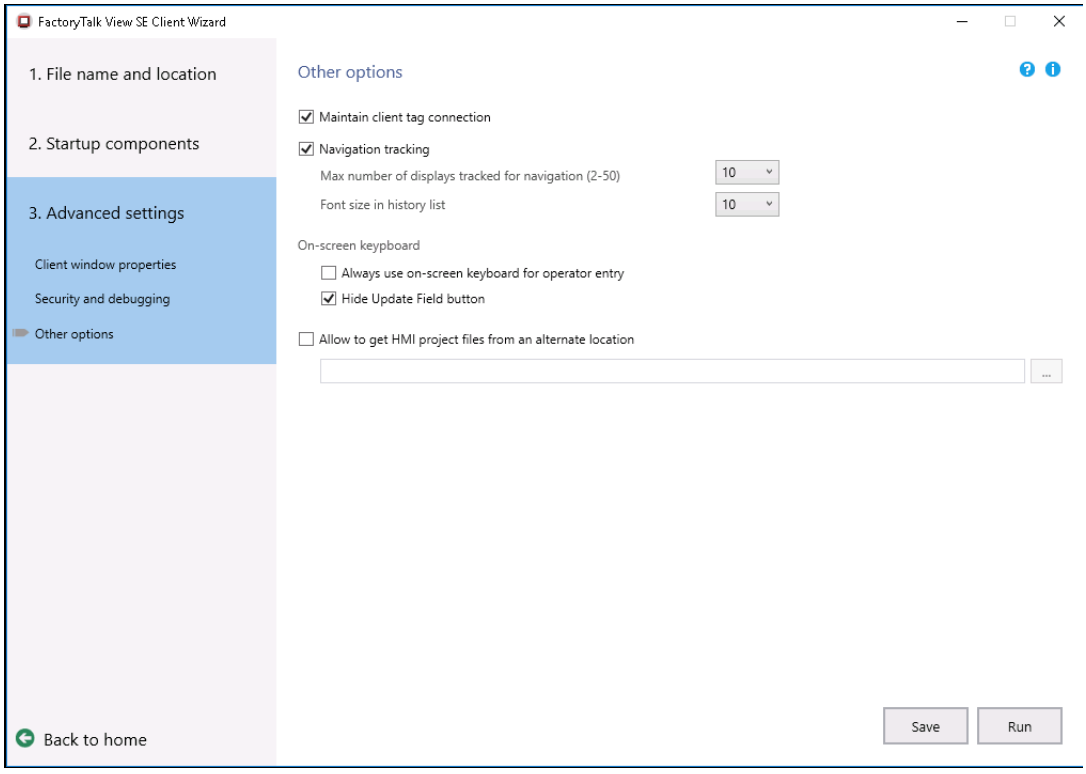


2. In the wizard, set the following:

On this Page	Action
File Name and Location	Name the client file and select the store location. In most cases, the store location should be the OWS desktop.
Startup Components	Select the appropriate application type. Connect to the correct application and select the initial language. Select the HMI server name within your application. Select the Startup Macro created in Macros. Select Advanced.



On this Page	Action
Advanced Settings	Select "Maximize Window" - Note: It is assumed that all monitors in the system have a resolution of 1920x1080. The PlantPAx Graphic Framework is designed to work with this resolution. Unselect “Allow Client to be resized at runtime”. Unselect “Show title bar” and unselect “Show diagnostic list”. Save the configuration. 
Advanced Settings - Multi-Monitor	If multiple monitors are used for the station that will run this client, check the box for "Multi-monitor". Additional options appear. Specify the monitor size as 1920x1080 and pick the desired monitor layout. In the following example, dual monitors side by side are being used. No additional configuration is required for each monitor - the startup macro that is selected on the Startup Components tab configures the screens when the client starts. Save and select the "Security and debugging". 

On this Page	Action
Security and Debugging	Depending on application requirements, select or unselect the “Enable auto logout”, “Open FactoryTalk View SE Client as view-only”, or “Disable switch to other applications”. The Debugging feature is only used for troubleshooting. Select “Other Options” tab. 
Other Options	Review the options and modify if necessary. Leave at default if there are not application-specific requirements. Save and close or Select Run to run the Client file. 

Notes:

Organization, Ownership, Arbitration, and Propagation (OOAP)

Overview

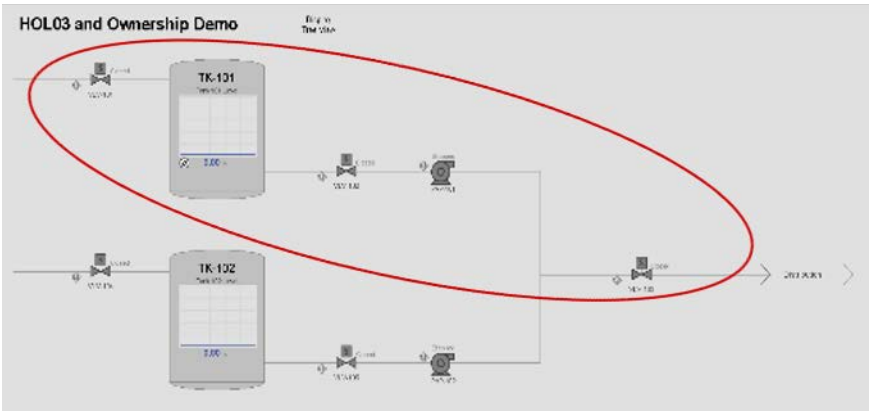
When should you use the OOAP functionality?

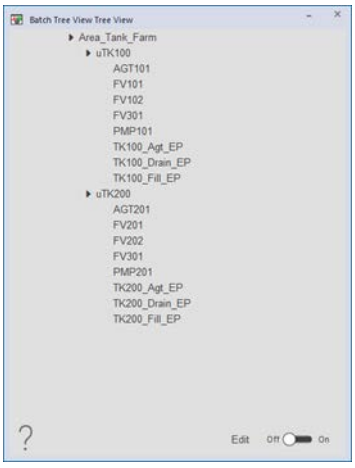
- A hierarchy exists among the equipment in the system
- Shared devices among equipment groups
- Design includes the use of Add-On Instruction process objects Area, Unit, Equipment Phase, Equipment Module, or Sequencer. These objects provide additional functionality when configured as parents in the organizational trees.
  - Show tree view faceplate navigation function from object faceplates provides a quick visualization of the state of an equipment group for troubleshooting.
  - PCmd_OwnerAcq and PCmd_OwnerRel interface from these Add-On Instructions provide a standard approach to organizing the children of the equipment group for control.
- Requirements exist for monitoring the aggregated statuses of a specific equipment group for maintenance or reporting purposes
- A Secondary Hardware Organization provides consolidated I/O Module status. Control Strategies can be informed of faults along the connection path it is in.

When should you NOT use the OOAP functionality?

- All equipment operates independently

Organization is a method by which parent / child relationships can be created and modified among PlantPAx® Instructions. Organization provides a method to propagate a selected subset of commands (related to command source, alarms, and so forth) from the parent down to its children or propagate the aggregate of a selected subset of status (related to command source, alarms, and so forth) from the children up to one or more parents.





Organizational views can be many nodes deep and wide, and numerous organizational views can reference the same devices to suit the needs of the user. The structure and view of these organizational trees can be modified online from the HMI by adding child nodes underneath existing nodes. The same bus object can be referenced on multiple nodes in the organization tree. When a bus object is referenced in multiple nodes in the organization, the parent node must be unique.

Organizing equipment provides the ability to coordinate commands of related equipment and view their related status (equipment modules or phase modules), or alternatively to monitor specific equipment or equipment types as a maintenance function.

A special use case of the organization function is the hardware bus that uses the raP_Opr_LogixModuleSts instruction to monitor the connection status of supported hardware modules. This is configured independently to provide an organizational view of a controller I/O tree configuration. By using the organizational tree for hardware, the alarms can be gated for child modules allowing root cause analysis by only alarming the highest module, which caused the alarm in the tree.

There are three basic capabilities that can be added incrementally:

Function	Description
Organization	Create parent/child relationships among objects to provide a method to propagate commands from the parent to its children and propagate status from the children to the parent.
Ownership	Uses the Organization backbone to allow a parent to take ownership of its children by placing them into Program state and accept the Owner's ID
Arbitration	Provides ability to manage and prioritize ownership of shared equipment

The implementation of these three capabilities can provide the following benefits:

- Provides the ability to command devices as an Equipment Group
- No additional Logix-based code is required to group equipment
- Consolidates information (alarms, device modes)
- Queues ownership requests and lets users set arbitration rules
- Applying standard solutions not only for direct device control, but also for device management, which enables a common look and feel throughout the complete application

The two major data structures that are the backbone of OOAP are the Bus and Node arrays. The bus is an interface to PlantPAx objects. It is used to propagate a specific set of statuses and commands through any user-defined organizational structure. A single element in the node array maintains the location of a bus object in the larger organizational tree. A single bus object can exist in multiple locations on the node array as a child. For example, a PVLV object could be a child to multiple equipment modules in the organization. The configuration for which commands and statuses are propagated between parents and children is maintained on the node array.

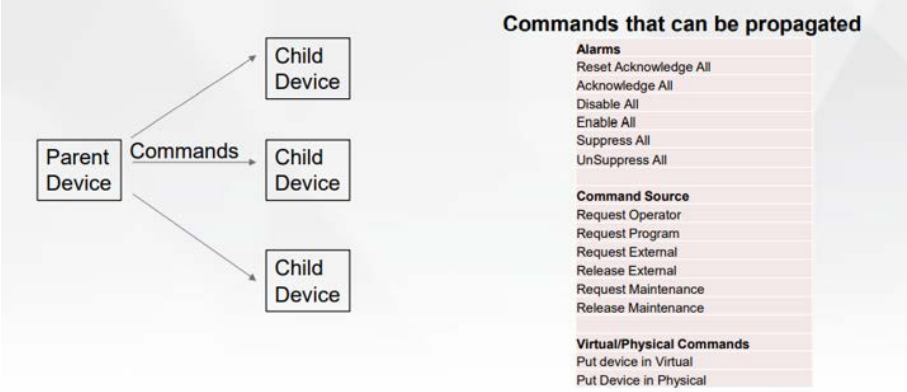
The individual elements in the bus array can be either real-time or non-real-time components.

- Real-Time: A bus object assigned to a controller executed PlantPAx Process Object (PVLV, PMTR, EMGen, and so on). These objects are time critical.
- Non-Real-Time: Elements that are purely organizational. Either for access or the aggregation of statuses. These objects are not time critical.

Propagation

The organizational trees that users create by defining parent/child relationships among equipment will immediately establish command and status propagation on the bus. Users can define which commands and statuses get propagated at each node's configuration display. Commands can be issued to children and child statuses can be viewed from the bus faceplate display. Designers can reference the bus command and status bits defined below in logic as needed. For example, Bus[Parent_Index].Obj.Inp_Cmd.14 will place all child devices in the external mode with a single command. The Inp_CmdAll.14 bit places the parent AND its children into external. It is recommended to latch and unlatch bus commands when implementing any user-defined logic with the following commands.

Propagated Commands

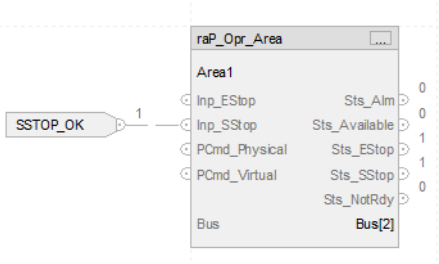


Commands Issued from the Parent Bus Object and Propagated to the Children in the Organization		
Bus[x].Obj.Inp_Cmd Bit	**Command**	**Description**
1	Reset/Ack All	Issue a Reset/Ack All to all objects
2	Reset	Issue a Reset to all objects
3	Disable alarms	Disable all alarms
4	Enable alarms	Enable all alarms
5	Suppress alarms	Suppress all alarms
6	Unsuppress alarms	Unsuppress all suppressed alarms
7	Unshelve alarms	Unshelve all shelved alarms
10	Request Virtual	Request all to be in Virtual
11	Request Physical	Request all to be in Physical
12	Request Oper	Request all to be in Operator
13	Request Prog	Request all to be in Program
14	Request Ext	Request all to be in External
15	Release Ext	Release all from External
16	Request Maint	Request all to be in Maintenance
17	Release Maint	Release all from Maintenance
28	Unit State Request 1	Unit User-Defined State Request 1
29	Unit State Request 2	Unit User-Defined State Request 2
30	Unit State Request 3	Unit User-Defined State Request 3
31	Unit State Request 4	Unit User-Defined State Request 4

Commands Issued from the Parent Bus Object that are Issued to the Parent and its Children in the Organization		
Bus[x].Inp_CmdAll Bit	**Command**	**Description**
1	Reset/Ack All	Issue a Reset/Ack All to all objects
2	Reset	Issue a Reset to all objects
3	Disable alarms	Disable all alarms
4	Enable alarms	Enable all alarms
5(1)	Suppress alarms	Suppress all alarms
6(1)	Unsuppress alarms	Unsuppress all suppressed alarms
7	Unshelve alarms	Unshelve all shelved alarms
10	Request Virtual	Request all to be in Virtual
11	Request Physical	Request all to be in Physical
12	Request Oper	Request all to be in Operator
13	Request Prog	Request all to be in Program
14	Request Ext	Request all to be in External
15	Release Ext	Release all from External
16	Request Maint	Request all to be in Maintenance
17	Release Maint	Release all from Maintenance

(1) The Area, Unit, EMGen, and EPGen Add-On Instructions can issue a PCmd_SuppressAllAlms and PCmd_UnSuppressAllAlms command using the bus. These commands use the CmdAll bus interface bits 5 and 6 to issue the suppress or unsuppress command to the parent AND its children.

The Line Level High commands defined in the previous table can be utilized to stop and start equipment as a group. For example, a software stop OK input signal can be mapped to the Area.Inp_Sstop input. Any Unit object that is defined as a child underneath that Area object in the organizational tree that is also configured to accept software stop commands through the bus goes to a not ready state when the software stop input = 0.



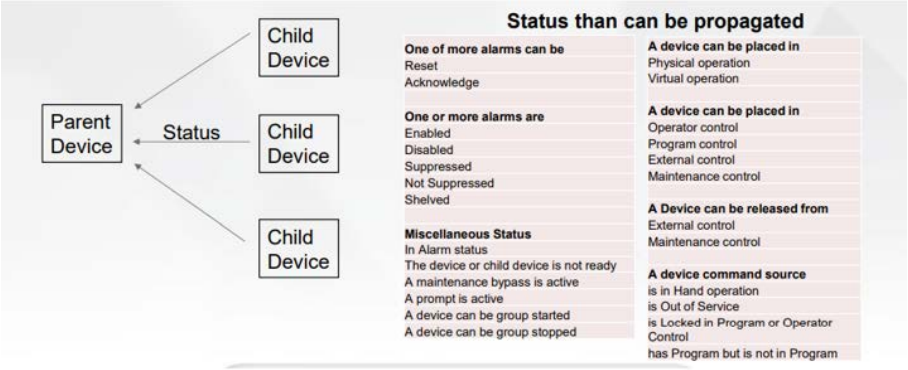
Likewise, an Interlock not OK status from a Unit object can propagate down the organizational tree to multiple Equipment Phase objects to set a not ready status and prevent operation.

Line Level High Commands Issued and Propagated to both Parent and Child		
Bus[x].Out_CmdLLH Bit	**Command**	**Description**
0	Emergency Stop(1)	1 = Emergency Stop OK (Inp_Estop)
1	Software Stop	1 = Software Stop OK (Inp_SStop)
2	Permissive OK(2)	1 = Permissive OK
3	Interlock OK(2)	1 = Interlocks OK

(1) Exists in the Area and Unit Add-On Instructions only.

(2) Exists in the Unit, EMGen, and EPGen Add-On Instructions only.

Propagated Status



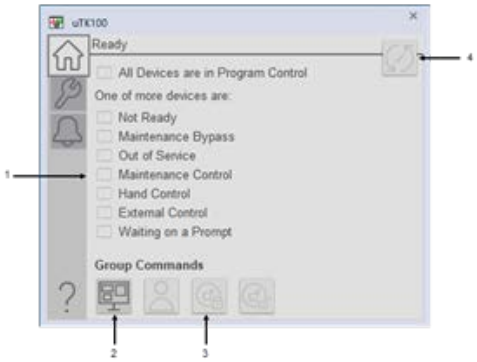
Bus[x].Obj.Out_Sts Bit	Status Produced and Propagated from Child to Parent	Description
0	Alarms Active	At least one alarm is active for this object or its children
1	Alarms/Object to be Reset	At least one alarm is ready for reset for this object or its children
2	Ready for Reset	At least one object or child is ready for reset
3	Alarms Enabled	At least one alarm is enabled for this object or its children
4	Alarms Disabled	At least one alarm is disabled for this object or its children
5	Alarms Unsuppressed	At least one alarm is not suppressed for this object or its children
6	Alarms Suppressed	At least one alarm is suppressed for this object or its children
7	Alarms Shelved	At least one alarm is shelved for this object or its children
8	Object Not Ready	At least one object or child is not ready
9	Maint Bypass Active	At least one object or child has a Maint Bypass active
10	Objects in Physical	At least one object or child is in Physical
11	Objects in Virtual	At least one object or child is in Virtual
12	Ready for Oper Request	At least one object or child is ready for an Oper request
13	Ready for Prog Request	At least one object or child is ready for a Prog request
14	Ready for Ext Request	At least one object or child is ready for an Ext request
15	Ready for Ext Release	At least one object or child is ready for an Ext release request
16	Ready for Maint Request	At least one object or child is ready for a Maint request
17	Ready for Maint Release	At least one object or child is ready for a Maint release request
18	Objects in Hand	At least one object or child is in hand
19	Objects Out of Service (OoS)	At least one object or child is Out of Service
20	Objects in Oper or Prog Locked	At least one object or child is Oper or Prog Locked
21	Objects has Prog, is not in Prog	At least one object or child has Prog but is not in Prog
22	Prompt is active	At least one object or child has an active prompt
28	Unit user-defined state 1	At least one child of the Unit is not in state 1
29	Unit user-defined state 2	At least one child of the Unit is not in state 2
30	Unit user-defined state 3	At least one child of the Unit is not in state 3
31	Unit user-defined state 4	At least one child of the Unit is not in state 4

In versions 5.20 and earlier, the Bus[X].Obj.CMDLLHMask tag is used to isolate children Out_Sts aggregation from the parent status. This allows a single bit to be used to indicate that at least one child is in alarm or not ready for an interlock or permissive on a parent object. These would correlate to Bus[x].obj.CMDLLHMask.0 and Bus[x].obj.CMDLLHMask.8 respectively. CMDLLHMask is a temporary placeholder for children only status. The Sts_NrdyChildAlm output can be used with the Area, Unit, Equipment Module, and Equipment phase objects for this purpose.

FactoryTalk View SE Bus Faceplate

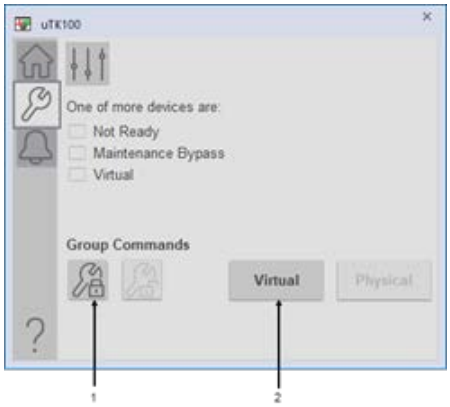
Many of the statuses and commands that are defined above can be found on the Bus Faceplate display. You can access the bus faceplate from the TreeView display or from object faceplates with the bus faceplate navigation button.

Operator Tab



Item	Description
1	Child Status indication
2	Request Children Program or Operator Command Source
3	Acquire/Release Children External Command Source
4	Reset/Acknowledge All Child Alarms

Maintenance Tab



Item	Description
1	Acquire/Release Children Maintenance Command Source
2	Request Children Virtual or Physical Mode

Alarms Tab

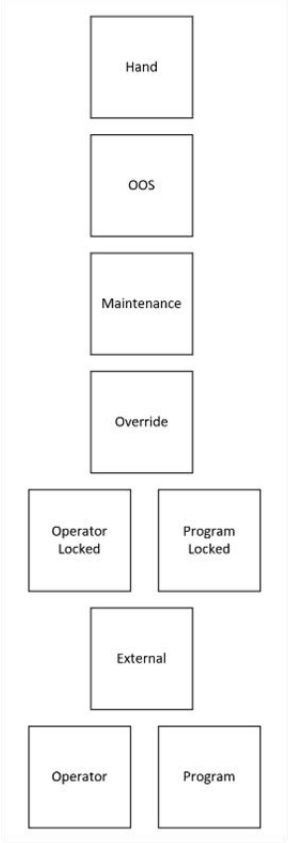
Item	Description
1	Disable all Child Alarms
2	Enable all Child Alarms
3	Un-shelve All Child Alarms
4	Reset/Acknowledge All Child Alarms

Bus Ownership

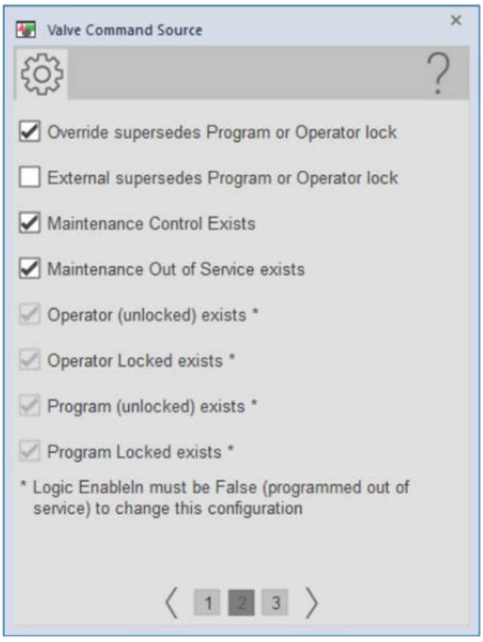
Ownership is built into the organizational bus backbone. It provides the ability for a parent in the tree to know that it has the child entity of interest assigned to it for control and that the child is placed in the program command source state for control by the parent. While the raP_Opr_Owner ownership function is available for use as a stand-alone component, the optional add-on bus functionality within the process objects utilizes ownership based on the parent-child relationships that the user defines within an organizational tree. This modular ownership approach allows functional groups to be created by the control designer while the ownership rules are maintained independently in each process object command source configuration. The ownership and organization functions are asynchronous, which allows the designer to create an equipment module in code independent of the devices and equipment, which it uses and takes ownership of.

Ownership is an independent function from the command and status propagation that OOAP provides. While users can utilize the bus faceplate or bus input commands at a parent object to command a set of children to program mode, that method does not prevent one of those child objects from being taken out of program mode nor does it provide a means to check that a child is no longer controllable by program logic. By requesting ownership via the bus, the user can ensure that a set of children is going to be controlled by only that parent object's program logic. When a child object accepts a parent object ownership request, it is using that parent objects bus ID as the ownership ID. The parent object will then continuously reassert ownership of the program command source to that object, which locks the child into program mode. The raP_Opr_OrgScan Add-On Instruction continuously checks that all children of that parent are in the appropriate state for control (Organized) and that the child owner IDs match the bus index of that parent (Owned). This eliminates the need for large amounts of controller code that was previously required to manage and control groups of equipment.

When a child is owned by a parent via the bus, the program mode is the only command source class that is owned. The External, Override, Maintenance, OOS, and Hand command source states still maintain priority over the core program and operator command source classes regardless of ownership status. All underlying command source class prioritization rules remain the same regardless of if the object is owned or not. A child that is owned by a parent and is placed into the Program Locked state can still be placed into Maintenance mode by a user that has sufficient security privileges. While the child is not in the program state, the parent of that child has a false children organized qualifier. When maintenance mode is released, the child returns to program locked and the parent regains the children organized status.



Bus ownership is dependent upon the underlying command source configuration of each child. For instance, if a child device program and program (locked) command source states are not configured to exist then bus ownership will not work. A child that is configured to have only the operator and program (locked) command source states to exist would return to operator mode each time ownership of that child is released or unbound.



The ability for a parent owner to operate the owned child control modules consists of two main qualifiers:

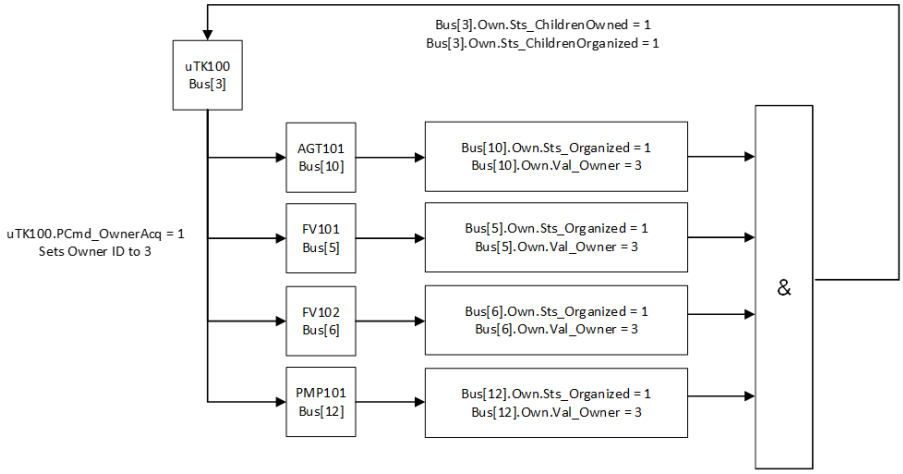
- Owned - The entity of interest has an owner ID of the interested owner assigned to it.
- Organized - The entity of interest is in the appropriate command source mode to be controlled.

The ownership function performs no action on the final control device, but instead provides a method to identify and prioritize owners so that other logic can use the 'Owned & Organized' status as permission for an Equipment Module or Equipment Phase to execute (Go / No Go Flag).

The parent bus element reports the status of its children with the aggregation of these two qualifiers. The parent process object Add-On Instructions (EMGen, EPGen, Sequencer, Area, and Unit) will automatically report these statuses. If the parent entity is not one of these objects, then the owned and organized statuses can be referenced at Bus[Parent_Index].Own.Sts_ChildrenOwned and Bus[Parent_Index].Own.Sts_ChildrenOrganized.

- Sts_ChildrenOwned – The aggregate of all children owned statuses. All children Val_Owner = Parent Bus Index.
- Sts_ChildrenOrganized – The aggregate of all children internal organized statuses. All children are in the program command source state.
- Sts_ChildrenGood – All children are owned and organized.

The following diagram illustrates the unit object uTK100 requesting ownership of four children using the uTK100.PCmd_OwnerAcq parameter. The raP_Opr_OrgScan Add-On Instruction monitors the owner ID and organized status of the children objects and defines a combined owned and organized status of the children for the unit object. The unit object then outputs uTK100.Sts_ChildrenOwned and uTK100.Sts_ChildrenOrganized statuses to be referenced in logic as needed.



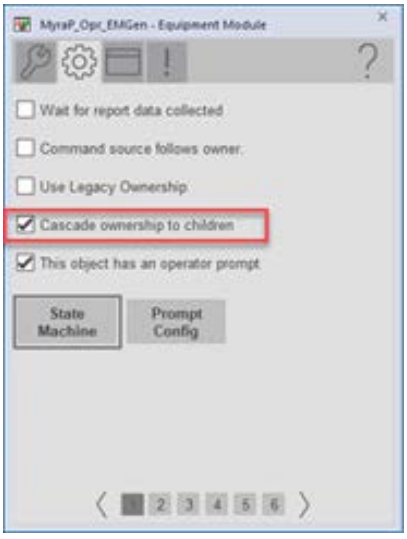
If a selected child object should not be owned when its parent requests ownership of its children, then the raP_Opr_OrgDeviceCtrl instruction can be used to issue a suppress ownership command at the child node. If ownership was suppressed at FV101 in the example above, then the Bus[5].Own.Val_Owner value would remain 0 while the Bus[3].Sts_ChildrenOwned and Bus[3].Sts_ChildrenOrganized values at the parent would remain equal to 1. The suppress ownership command leaves the FV101 object available to be owned by another parent in the organization while also allowing the parent object to proceed with its operation. For more information regarding the raP_Opr_OrgDeviceCtrl instruction, refer to Chapter 9. This instruction is available in version 5.30.00 of the Process Library and later

Ownership Configuration

The Sequencer, Area, Unit, Equipment Module, and Equipment Phase objects are designed to act as an owner (parent) in the organization. The implementation of Area, Unit, Equipment Module, Equipment Phase, and Sequencer is specific to each application. The tank farm example shows an instance of each type excluding the sequencer. These objects provide ownership configuration parameters that define how ownership can be requested from the object and how the resulting ownership statuses are managed.

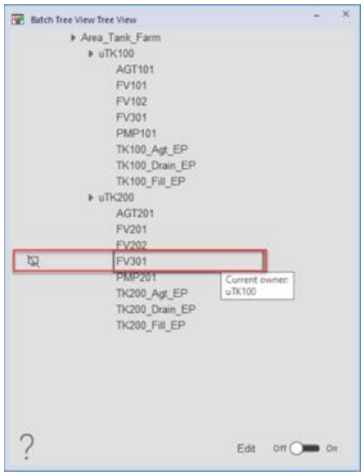
Parent	Guidelines
Area	An Area is the top-level ownership object. An Area groups Units together so that it can manage: • Command Source for a group of equipment • Alarms for a group of equipment If you must sequence equipment (rather than group equipment), use either a Sequencer, Equipment Module, or Equipment Phase
Unit	A Unit groups equipment. Units operate relatively independent of one another. A Unit manages: • Command Source for a group of equipment • Alarms for a group of equipment If you must sequence equipment (rather than group equipment), use either a Sequencer, Equipment Module, or Equipment Phase
Equipment Module	An Equipment Module groups and sequences equipment. Use an Equipment Module when you want to apply a custom state model to the equipment. For more information see EM/EP User Manual, publication PROCES-UM110.
Equipment Phase	An Equipment Phase groups and sequences equipment. Use an Equipment Phase when you apply the ISA 88.01 state model using PhaseManager™. For more information see PhaseManager Software User Manual, publication LOGIX-UM001. For more information see EM/EP User Manual, publication PROCES-UM110.
Sequencer	A sequencer groups and sequences equipment. Use a sequencer when you want to implement a procedure to operate equipment in a prescribed order. For more information see the Sequencer Object User Manual, publication PROCES-RM202.

There can be many layers of parent/child relationships in an organizational tree. By default, the parent objects will automatically request ownership of its children when it becomes owned by its parent above it. The Sequencer, EMGen, EPGen, Area, and Unit objects can disable this functionality by setting the Cfg_CascadeOwn = 0.

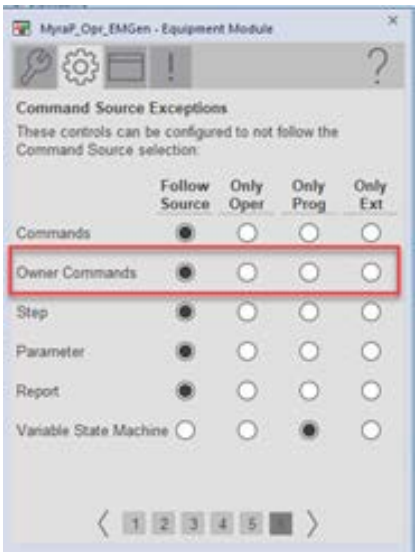


The parental objects provide a Sts_NrdyChildNotUsable indication. This status indicates that the object requested ownership of its children and has not yet received an Owned and Organized status from at least one child. The annunciation of this status can be delayed with the Cfg_Child2BGoodTmr setting shown in the following display.

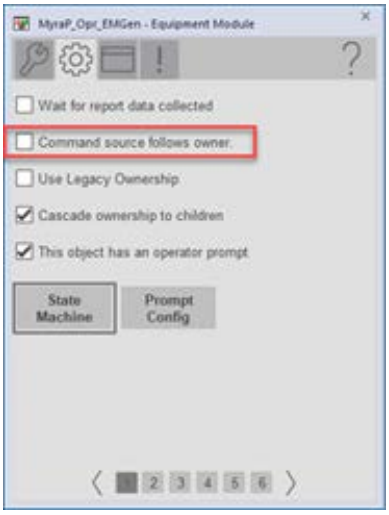
In the following tree dislay, FV301 is a shared device of both uTK100 and uTK200. Because FV301 is owned by uTK100 the FV301 node underneath uTK200 indicates a child not usable indication.



You can define which command source is able to request ownership of children. This is available in the EMGen, EPGen, Area, and Unit objects. This configuration can be used to prevent operators from requesting child ownership from the object faceplates. When Owner Commands is set to "Only Prog" or "Only Ext" then the faceplate buttons to Acquire and Release ownership will be disabled.



The "command source follows owner" configuration in the EMGen, EPGen, Area, and Unit objects can be used to automatically take ownership of children when the object is placed into the program command source state. Using this function disables operator ownership request functionality. If this configuration is used and the owner commands are set to only program in the command source exceptions faceplate then the children of the object will always be owned. This is not recommended if there are shared devices among parents in the system.



Although many objects (including PAI and PDI) can be included in the organizational tree, only objects with a Command Source can be owned. Objects are in Program command source state when owned by a parent Bus element.

Request Ownership

The parent control point can issue ownership requests from the Operator, Program, External, or Maintenance command source states. All ownership requests and statuses propagate through the bus. Once owned, the owner function issues a request to place its children into the program state.

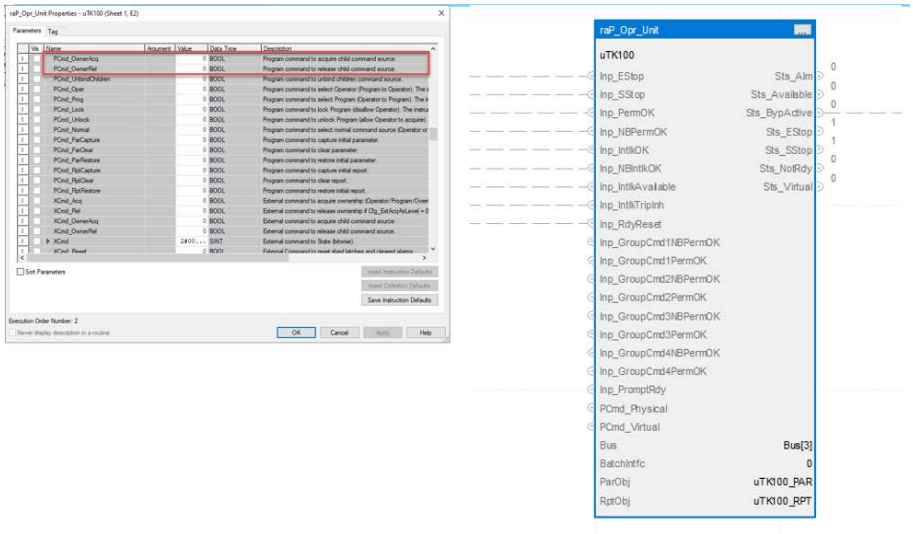
Ownership is established when a parent requests ownership, all its children grant this request, and all its children are placed into the program command source state (Children that belong to this parent are ready to accept parental requests). When a child grants a parent ownership request then the Owner ID of that child will be equal to the parent bus index.

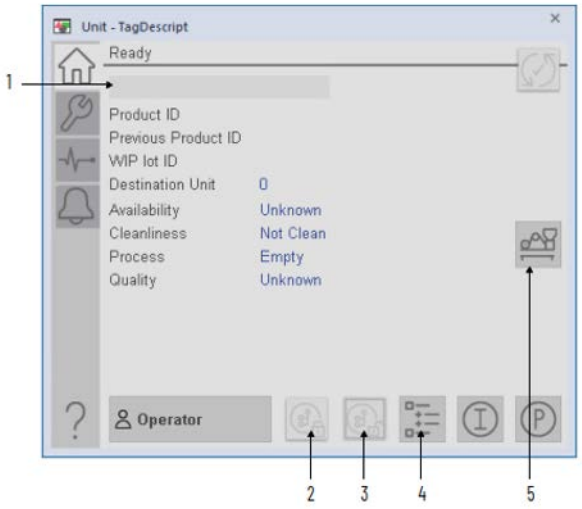
The parent can request ownership programmatically or via the faceplate for the associated parent Add-On Instruction.

Parent Object	Associated Add-On Instruction
Area	raP_Opr_Area
Unit	raP_Opr_Unit
Equipment Module	raP_Opr_EMGen
Equipment Phase	raP_Opr_EPGen
Sequencer	raP_Opr_Seq

For example, a Unit object that is a parent can request ownership programmatically via the PCmd parameters. To request ownership, you must set PCmd_OwnerAcq = 1. The PCmd_OwnerRel parameter releases ownership. The command source exceptions for owner commands must be set to "Only Prog" or the unit must be in the program command source state to accept the PCmd_OwnerAcq and PCmd_OwnerRel when the command source exception is set to "Follow Source".

A Unit parent can request ownership via the Operator tab on the faceplate:





Item	Description
1	Displays the current state of the object
2	Acquire child command source
3	Release child command source
4	Navigates to the tree view for this object
5	Navigates to the Bus faceplate

For object and visualization parameters, see Object and Visualization Parameters, publication PROCES-RD201.

Additional commands exist on the bus data structure that pertain to ownership. The commands in the table below are used within parental object Add-On Instructions Sequencer, raP_Opr_EMGen, raP_Opr_EPGen, raP_Opr_Seq, raP_Opr_Unit, and raP_Opr_Area.

Parent bus ownership Commands	Bus Ownership Command Description
Cfg_CascadeOwn	Suppress automatic child ownership when a parent is owned by another parent. 1 = When this bus element is owned, do not automatically own its children.
PCmd_OwnerAcq/Pcmd_OwnerRel	Parent request to own its children. 1 = Own children even if this element is not owned.
PCmd_UnbindChildren	Parent request to unbind all children. 1 = Allow ownership but do not enforce command source state. You are able to change the command source of the child while it is owned.

Child-level commands are available on the bus. These can be used in specific use cases where all children should not be affected by ownership the same way.

Child Bus Ownership Commands	Bus Ownership Command Description
Bus[x].Own.Inp_InitializeReq	Command to reinitialize the ownership of object x.
Bus[x].Inp_ProgDoNotInclude(1)	Input to unbind a bus object and not to include the object in the aggregation of Sts_ChildrenOrganized.

(1) Available in versions 5.10.01 and later.

Unbinding Ownership

When a parent acquires ownership of its children the parent will continuously reassert program class ownership over those children. By doing this, the owning parent is forcing the child command source state to Program. If the child object is configured to have the Program Lock and Program command source states, then it is forced to Program Lock. A Pcmd_Unlock command would only work for one scan of the controller before the parent reasserts ownership again and places the object back into Program Lock. Similar behavior would be exhibited if that object was configured to have only the program command source state. A user could attempt to place the object into the operator state from the object command source faceplate, but the parent would once again reassert ownership and place the device back into program on the next scan of the raP_Opr_OrgScan Add-On Instruction.

When there is a requirement to allow certain devices to change to operator mode while owned, the developer can utilize the unbind commands that are available on the bus or from the parental objects. An unbound child object will not continuously be forced to the program state by its parent. This allows a user to change the command source state of a child object to operator while still owned. A child object that is configured to have the Program Locked command source state transitions from Program Locked to Program mode when owned by a parent and an unbind command is present. Parental Add-On Instruction objects provide a PCmd_UnbindChildren command to unbind all children of that parent while owned. If this command is used at a parent and one of the children are placed into operator mode, then the parent Add-On Instruction loses the children organized status and then loses the children good status. Unbind commands have no effect on the ownership of a child object. Parent Add-On Instruction objects output a Sts_ChildrenUnbound when the PCmd_UnbindChildren command is used

The Bus[index].Own.Inp_UnbindAll input can be used to selectively unbind all four ownership command sources simultaneously at any single process object that is connected to the bus. This may be desirable in cases where all children of an object may not need to be unbound. When using the Inp_UnbindAll input, the parent recognizes if it has lost organization when that child is placed outside of program mode.

Exclusion

Excluding an object from organized status accumulation at a parent unbinds that object and allows the operator to take it out of program mode without the parent object recognizing that the child is unorganized. Users can use the Bus[index].Inp_ProgDoNotInclude command to unbind and exclude a given bus object everywhere it is defined in the organizational tree. If an object has multiple parents in the tree and exclusion is only desired under a single parent, then the raP_Opr_OrgDeviceCtrl instruction can be used to unbind and exclude an object at a single node in the tree. For more information regarding the raP_Opr_OrgDeviceCtrl instruction, see to Chapter 9. This instruction is available in version 5.30.00 of the Process Library and later.

The Bus[index].Inp_ProgDoNotInclude input can be utilized to selectively unbind individual process objects that are connected to the bus and exclude that object from the aggregation of the children organized status of its parent when that parent has requested ownership. This allows you to place that excluded object into operator mode while owned and still maintain the children organized and children good statuses at the parent object. This may be necessary in cases where the parent object is configured to stop or hold when its children are no longer organized. While an object is excluded from organized status, the object is still be owned by the owning entity. The Bus[index].Inp_ProgDoNotInclude command should be set each time ownership is requested and reset at the time ownership is released. If unbind commands are maintained after ownership is released, then the process object will remain in its current command source state the next time ownership is requested. An example use case for the Inp_ProgDoNotInclude input would be for a PVSD or PPID child object that an operator may need to keep running while adjusting setpoint parameters manually. This input would allow the operator to place the object into operator mode to adjust the process without stopping or holding the parent object due to organization being lost.

Workflow

Before you begin, confirm that you have the following:

- Controller project and HMI project communicating to controller
- System hierarchy for equipment grouping
- Graphic framework installed
- Configured HMI shortcuts
- Displays created

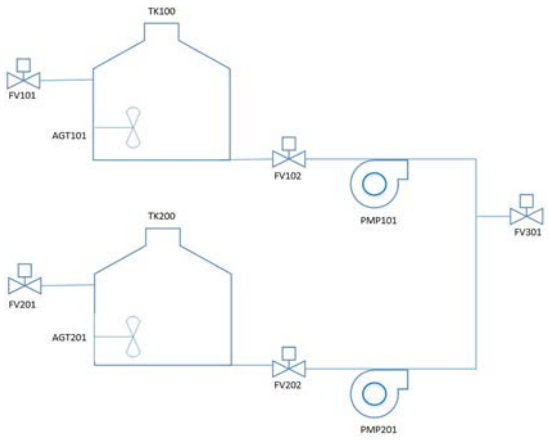
IMPORTANT Application Code Manager and the PlantPAx Configuration Tool provide simplified workflows that are more efficient to initially configure organization. However, the manual process that is detailed in this document can be useful for minor edits.

The recommended workflow for large implementations of the bus is as follows:

1. Build the base project structure and device logic using Application Code Manager. The Process Library download provides ACM libraries that can define the controller code for the OrgScan and OrgView implementations. ACM provides interfaces to assign devices to bus elements in bulk. Object parameters can be set to automatically create mappings to the Bus[x].Own.Out_OwnerCmd and Bus[x].Own.Inp_OwnerSts as well as create an Arbitration() object where needed.
2. Generate the .ACD file in Application Code Manager.
3. Import the .ACD file into the PlantPAx Configuration Tool. Access the Organization Bus Configuration tool by right-clicking the imported controller and selecting "Configured Organization Bus."
4. Configure bus element names and navigation from the organization bus configuration tool. All bus elements names can be set to the associated Logix tag names with a single click.
5. Using the organization bus tool, drag-and-drop controller tags from the logical organizer window to the BusNode Tree window to easily define the parent and child relationships in the organization.
6. Configure the bus and node element command and status propagation masking.
7. Configure the FactoryTalk® View SE startup macro definitions and graphic displays.

Manually Configure Organizations

1. In a process controller project, create the organization logic (create Bus and Node arrays)
2. Define the Bus elements
3. Implement the OrgView elements that store data for each HMI client
4. Create the Organizational Tree (define nodes and configure propagation and displays)



The Tank Farm Area has:

- Unit TK100 has children FV101, FV102, FV301, PMP101, and AGT101
- Unit TK200 has children FV201, FV202, FV301, PMP201, and AGT201

Create the Organization Logic

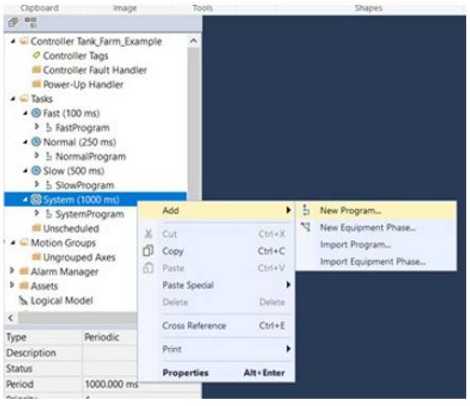
Before you begin, import the process library Add-On Instructions into your controller project. All logic must be in one controller. The following Add-On Instructions are required:

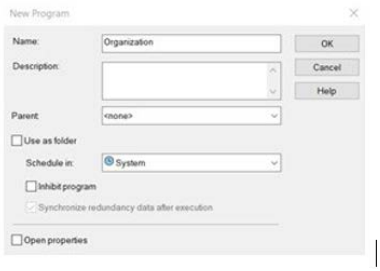
- raP_Opr_OrgScan Organizational Scan
- raP_Opr_OrgView Organizational View

Create the Organization Program

In a process controller project, create the organization program.

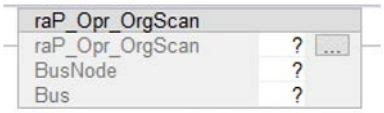
1. Add a new program in the appropriate Task and name it Organization. The default task is the System task.





2. Add a new routine to the Organization program and name it MainRoutine.

3. Open the MainRoutine in the Ladder Diagram editor and add an raP_Opr_OrgScan Add-On Instruction to manage the propagation of commands/status and Ownership within the Organization Tree.



Create Array Tags

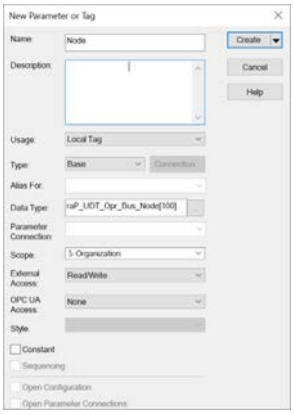
Create two array tags for these parameters that are used in this Add-On Instruction:

- Bus array with an element for each device in your system hierarchy. Make the array larger than the number of devices so you have room for future additions.
- BusNode array with an element for each device and its relationship to other devices. Make the node array twice as large as the Bus array so that you can account for devices that have relationships with multiple other devices, and you have room for future additions

1. Create controller-scoped tag OrgScan of data type raP_Opr_OrgScan.

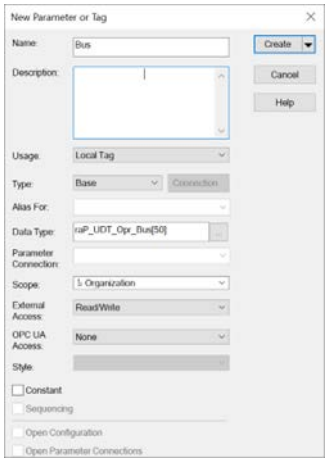
2. Create a controller-scoped array tag Node of type raP_UDT_Opr_Bus_Node.

   The array tag must be named Node. For this example, edit the array to have 100 elements. You can have more nodes than bus elements to define multiple parent/child relationships.



3. Create a controller-scoped array tag Bus of type raP_UDT_Opr_Bus.

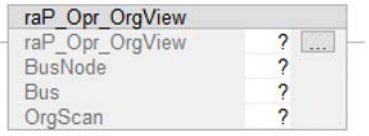
   The array tag must be named Bus. Edit the array elements to have 50. Create more elements than your original list of bus elements to leave space to add future elements.



IMPORTANT You can have as many as 500 Bus elements per controller (1500 for 5.00.04 or later)

4. Add a rung with an instance of the raP_Opr_OrgView Add-On Instruction for each HMI client that will be using the organizational tree. This example assumes five HMI clients, so there are five individual rungs.

   Right-click the raP_Opr_OrgView ? and create a tag OrgView of data type raP_Opr_OrgView.



5. Edit the Data Type and enter the array dimensions so that you have an element for each HMI client. In this example there are five HMI clients so edit the array to have five elements (one element for each of the five HMI clients in this example).

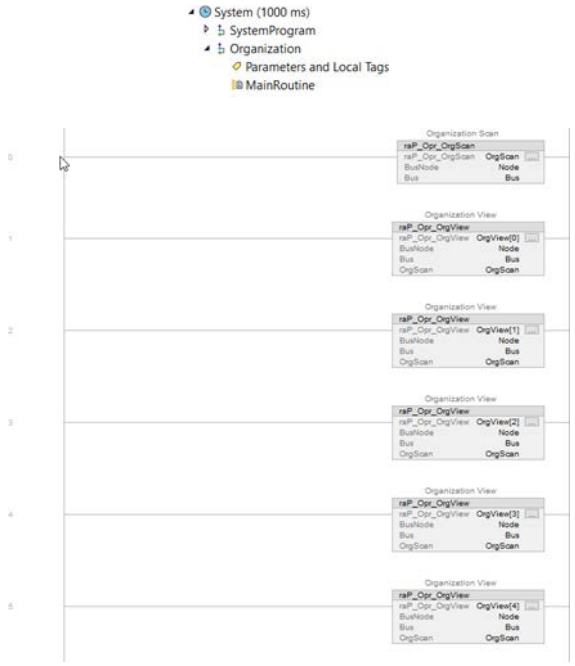
IMPORTANT	Each HMI Client startup macro must specify which of the five OrgView elements it is assigned to. Correlating an OrgView element to each unique client in the system allows for users to view and navigate the organizational tree view independently from each other. Users can also configure the Organization view to behave differently from client to client. Editing of the tree can be disabled for a particular client from the OrgView Config faceplate. Each OrgView element can maintain a different starting node for each client. When a start node is defined for a client, that starting node will be the uppermost node in the tree view in place of the <root> node.

6. Enter the BusNode, Bus, and Orgscan tags you created for the OrgScan instruction.

   Select one OrgView[x] element for each rung/instance to correspond with each HMI client.

Example Logic

The final logic for this example looks like the following:

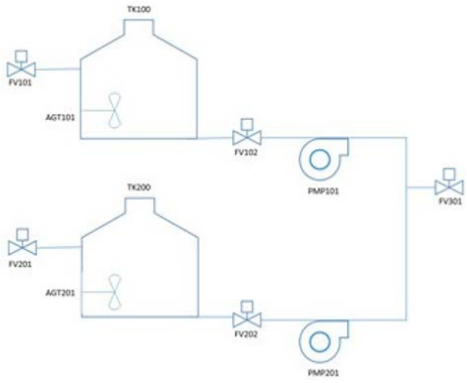


Define the Bus Elements

Organization requires a mechanism to collect entities into a structure. Each entity that is grouped

into an organization must be assigned a unique identifier. The Bus array provides this mechanism, in addition to providing an interface to exchange commands and status. Each PlantPAx bus-capable control element must be assigned a bus element. Generic bus elements can be added as logical containers as necessary to group control elements and other generic bus elements.

Example Bus Elements



You need a bus element for every possible owner (parent) and child. For this example, create these elements. The elements can be in any order, and you can use any names that are appropriate for your application.

Bus Element	Name	Description
Bus[0]		Reserved
Bus[1]		
Bus[2]	Area_Tank_Farm	Process area
Bus[3]	uTK100	Unit TK100
Bus[4]	uTK200	Unit TK200
Bus[5]	FV101	Valve FV101 owned by TK100
Bus[6]	FV102	Valve FV102 owned by TK100
Bus[7]	FV201	Valve FV201 owned by TK200
Bus[8]	FV202	Valve FV202 owned by TK200
Bus[9]	FV301	Valve FV301 shared by TK100 and TK200
Bus[10]	AGT101	Agitator AGT101
Bus[11]	AGT201	Agitator AGT201
Bus[12]	PMP101	Pump PMP101
Bus[13]	PMP201	Pump PMP201
Bus[14]	TK100_Fill_EP	Fill Equipment Phase for TK100
Bus[15]	TK100_Drain_EP	Drain Equipment Phase for TK100
Bus[16]	TK100_Agt_EP	Agitate Equipment Phase for TK100
Bus[17]	TK200_Fill_EP	Fill Equipment Phase for TK200
Bus[18]	TK200_Drain_EP	Drain Equipment Phase for TK200
Bus[19]	TK200_Agt_EP	Agitate Equipment Phase for TK200
Bus[20]...Bus[50]	Empty; for future additions	

IMPORTANT Release 5.00.00 of the PlantPAx library, supports as many as 500 bus elements. Release 5.00.04 and later supports as many as 1500 bus elements.

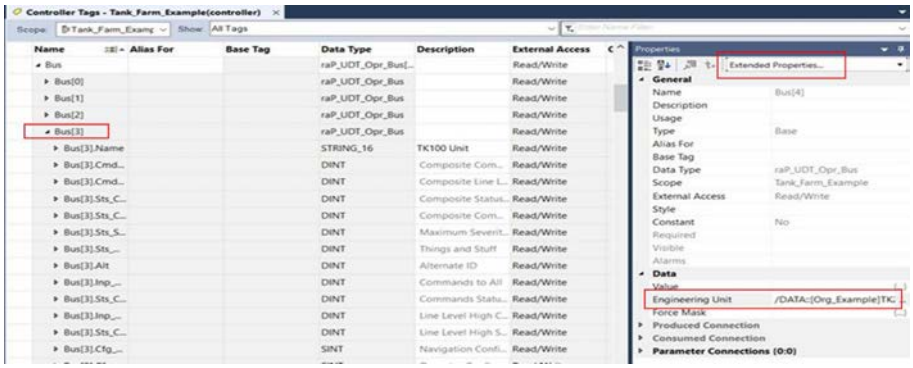
Each bus element has multiple members. To see the members, on the OrgScan or OrgView instruction, right-click Bus and select Monitor "Bus".

Expand each element to configure the device.

1. Enter the name of each element in the Value Field for each Bus[x].Name. The organizational tree uses this value for the device name.
2. Add Extended Properties > Engineering Unit to the element and specify the shortcut for the Data value. The shortcut ties the element to the display faceplate. The shortcut must be defined in the HMI project.

   In this example:

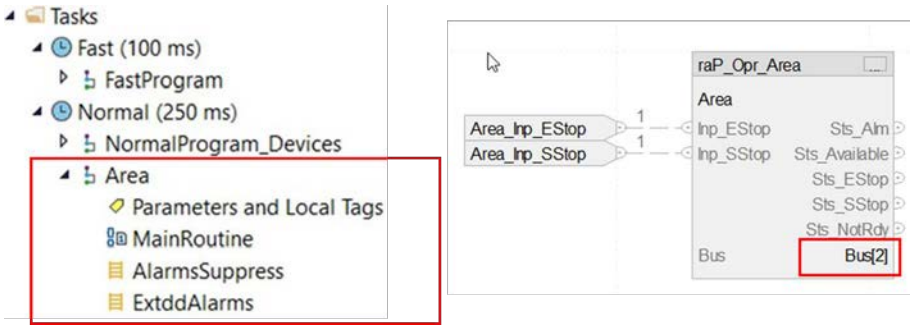
- DATA is the name of the data server
- Org_Example is the name of the shortcut in the HMI project
- uTK100 is the name of the element



Configure the Area Instance

Import the PlantPAx area control strategy: CS_raP_Opr_Area. This defines the Area. In this example, the main routine identifies Bus[2] element for the area.

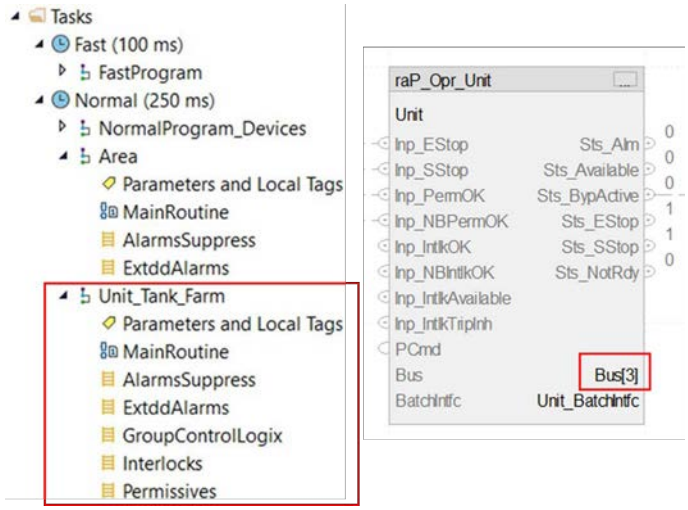
On the Import Configuration dialog, find and replace all instances of 'raP_Opr_Area' in Tags and Descriptions with 'Area.'



Configure the Unit Instances

Import the PlantPAx unit control strategy: CS_raP_Opr_Unit. This defines a Unit. In this example, the main routine defines Bus[3] (Unit_TK100) and Bus[4] (Unit_TK200) elements for a Unit.

On the Import Configuration dialog, find and replace all instances of 'raP_Opr_Unit' in Tags and Descriptions with 'Unit_TK100' for the first instance and 'Unit_TK200' for the second instance.



Create instances for the TK100 and TK200 Units.

Configure the Equipment Phase or Equipment Module Instances

Import the appropriate PlantPAx control strategy:

- CS_raP_Opr_EM_Gen (Equipment Module)
- CS_raP_Opr_EP_Gen (Equipment Phase)

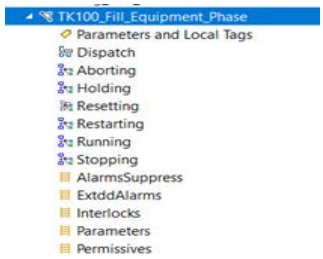
This example uses Equipment Phases. Edit the Bus element appropriately:

- Bus[14] for Unit TK100_Fill_EP
- Bus[15] for Unit TK100_Drain_EP
- Bus[16] for Unit TK100_AGT_EP
- Bus[17] for Unit TK200_Fill_EP
- Bus[18] for Unit TK200_Drain_EP
- Bus[19] for Unit TK200_AGT_EP

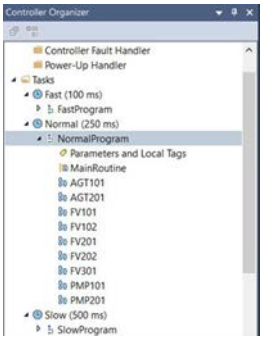
On the Import Configuration dialog, find and replace all instances of 'raP_Opr_EP_Gen' in Tags and Descriptions with the appropriate phase name.

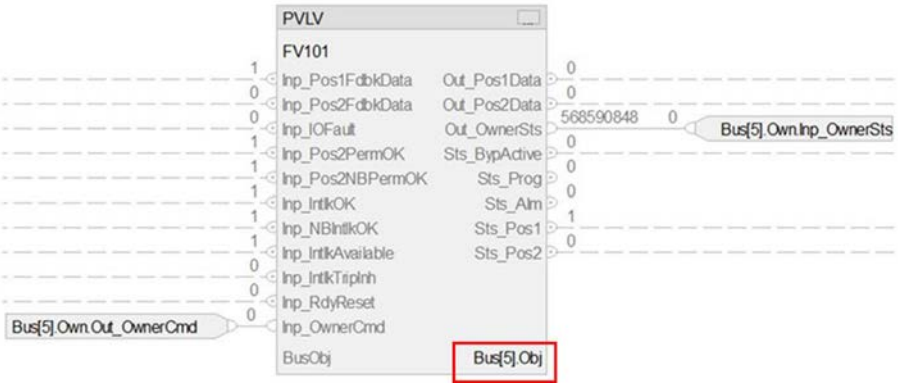
Configure the Device Instances to Use the Bus Elements

Use the PlantPAx control strategies to create the logic for your application. Import the appropriate control strategy for each device. Assign the bus element of each device to the process instruction.

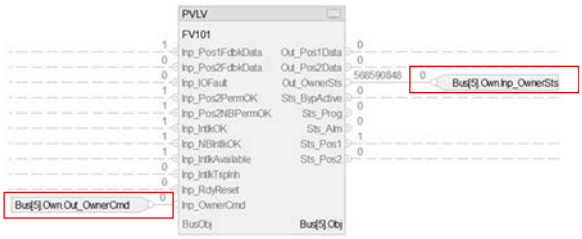


The Bus Object must match the device element in the Bus array.





Each device that you want to add to the node tree must have an associated process instruction.

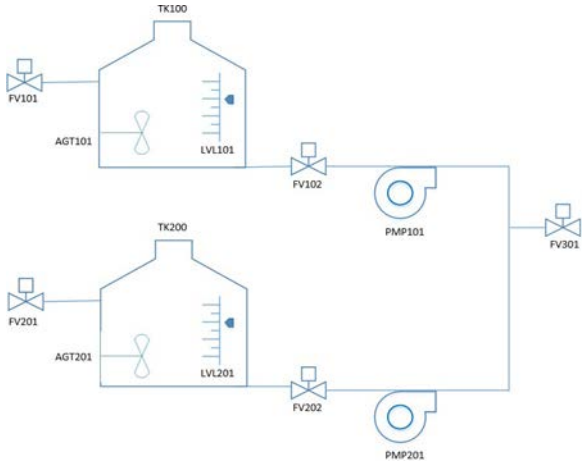


Each process object that requires a BusObj connection also requires the mapping of the Bus[x].Own.Out_OwnerCmd to the Object.Inp_OwnerCmd and the Bus[x].Own.Inp_OwnerSts to the Object.Out_OwnerSts. This mapping allows for bus ownership to place objects in the requested command source state and then report that status back to the parent object. The Bus[x].Own index references should match the Bus[x].Obj index reference. Bus capable Add-On Instructions do not require this mapping because the bus connection is for the entire Bus UDT structure. The Bus.Own mapping exists internal to the instruction in this case.

Add Devices

Bus elements can be in any order, and you can use any names that are appropriate for your application. To add additional devices to the Bus, reference and use a Bus array element that is not being used by another object, excluding 0.

In this example, add level indicators to both tanks.



Bus Element	Name	Description
Bus[20]	LVL101	Level for TK100
Bus[21]	LVL201	Level for TK200
Bus[22]...Bus[50]	Empty; for future additions	Bus[22]...Bus[50]

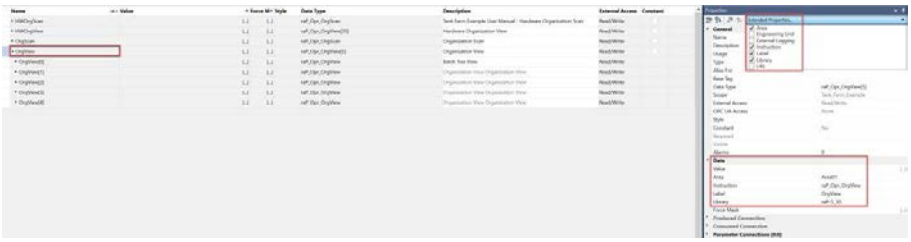
Define the OrgView Elements

Each control entity and container has been assigned a unique identifier (Bus element), and each tree and its entities can be updated dynamically using the OrgScan instruction (Node element).

To facilitate the creation of the HMI client access, you must also define an OrgView array, which will store the data for each client. This is an array of type raP_UDT_Opr_OrgView. There should be one OrgView instruction that is configured for each HMI Client. OrgView provides a window to allow online manipulation of each organizational tree. The Organization View object allows online changes to the Organization Tree (Nodes) while providing a view to the various organizational trees. Only one client can edit the organization at a time. You must request edit control before making updates. A configurable timeout is used to release edit control at expiration.

Example OrgView Elements

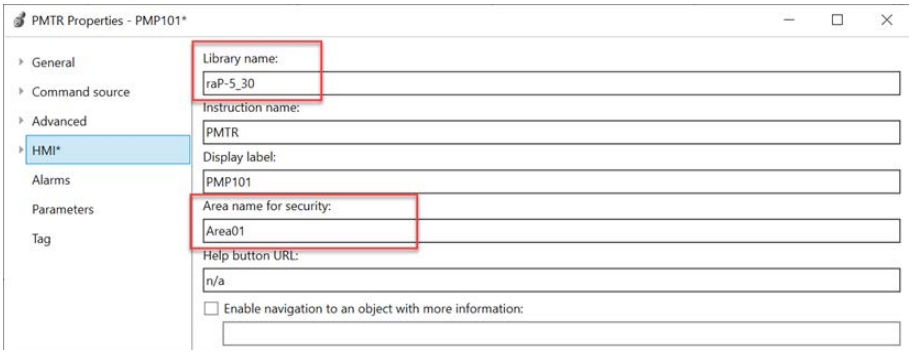
For each OrgView element, enable the Area, Instruction, and Library Extended Properties. These properties tie the display back to the organization structure.



Then in the Data, enter this information:

- Area = Area01
- Instruction = raP_Opr_OrgView
- Library = raP-5_30
- URL = n/a

IMPORTANT	The Area and Library names must match the settings in the HMI page of the device properties. The library parameter must match the latest version of that library at the time of implementation. See the following example:



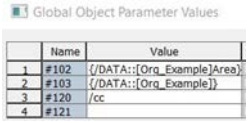
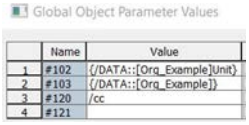
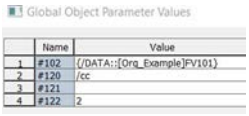
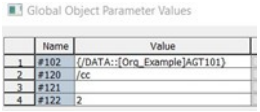
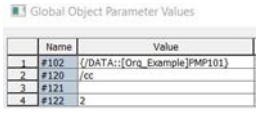
These settings are the default names when you import a process control strategy.

Create the Organizational Tree in the HMI Client

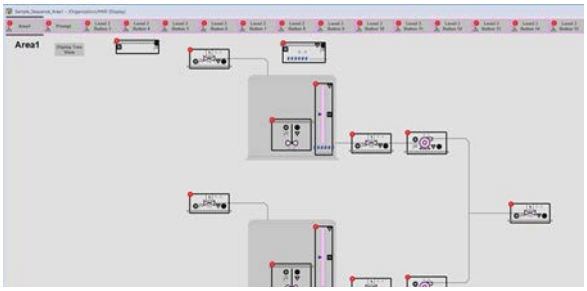
Make sure that you have the application configured with the appropriate user groups. This example uses the default Area01_Basic and Area01_Advanced user groups with a user Engineer_A.

Configure the Client Display

1. Build the client display with the elements you defined in the Bus. This example uses these graphic symbols in the display:

Graphic Symbol	Description
Area (You must have one Area symbol)	(raP-5_30-SE) Graphic Symbols - raP_Opr_Area.gffx 
Unit	(raP-5_30-SE) Graphic Symbols - raP_Opr_Unit.gffx 
Valve (one solenoid symbol for each valve)	(raP-5_30-SE) Graphic Symbols - PVLV.ggfx 
Agitator (one agitator symbol for each agitator)	(raP-5_30-SE) Graphic Symbols - PMTR.ggfx 
Pump (one blower symbol for each pump)	(raP-5_30-SE) Graphic Symbols - PMTR.ggfx 
Display Tree View button	(raP-5_30-SE) Graphic Symbols - Cross Functional.gffx

2. Select and copy the 'Display Tree View' button from the '(raP-5_30-SE) Graphic Symbols - Cross Functional.ggfx' global object file. Paste this button to the desired display.



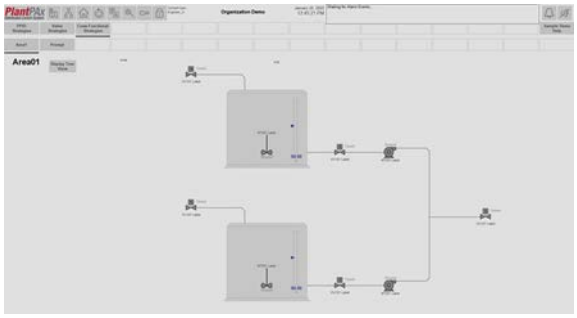
3. Add the following lines to the client startup macro. The Template_ClientStartup macro file provides an example that can be uncommented and adjusted as needed. The template example uses the OrgView element 0 and /Area1/DATA::[Hardware] for the controller topic. Another startup macro must be created for each HMI client in the system so that each client can be assigned another OrgView element. The macros DefineShowTreeCmd and DefineShowHWTreeCmd are referenced in the startup macros and must exist in the project.

Line	Description
Define SW_RedefineShowTreeCmd DefineShowTreeCmd X	X = Client number (OrgView element) 0 in this example for client 0
SW_RedefineShowTreeCmd /Area/DATASERVER::[TOPIC]	Area = Area for data server DATASERVER = data server name TOPIC = shortcut name (path to controller) DATA::[Org_Example] in this example

See Client Startup Macro on page 107 to complete this configuration.

IMPORTANT The PlantPAx Graphic Framework Tool provides an interface for defining hardware and software bus commands for the client startup macros in the L1 configuration.

4. Generate the client display.

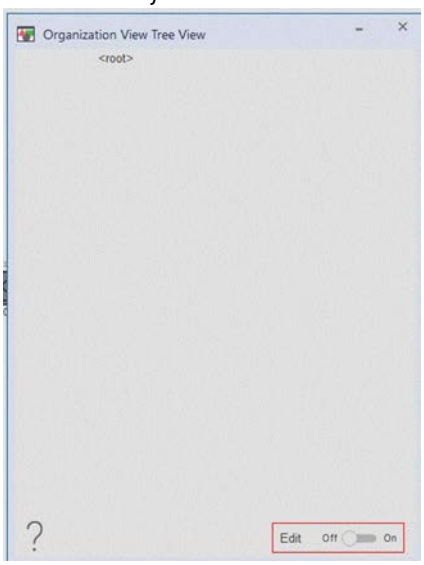


Build the Node Tree

The node tree is where you define system hierarchy and relationships among objects. Build the complete node tree once. You can specify a start node within the node tree or each client. These examples show how to manually perform the processes, you can also use the PlantPAx Configuration Tool.

1. Sign in to the client display and click the Display Tree View button.

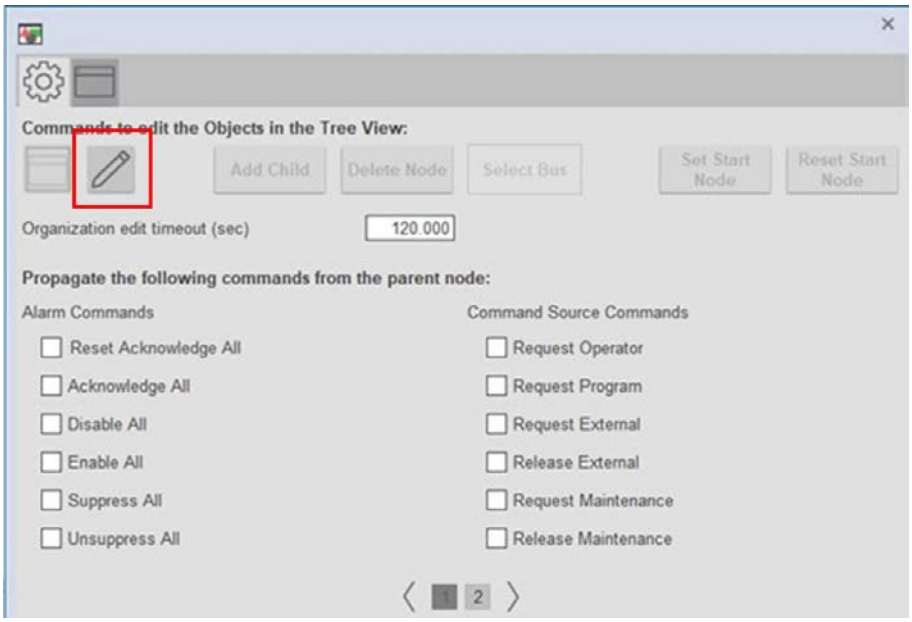
The initial Tree View shows only the <root>.



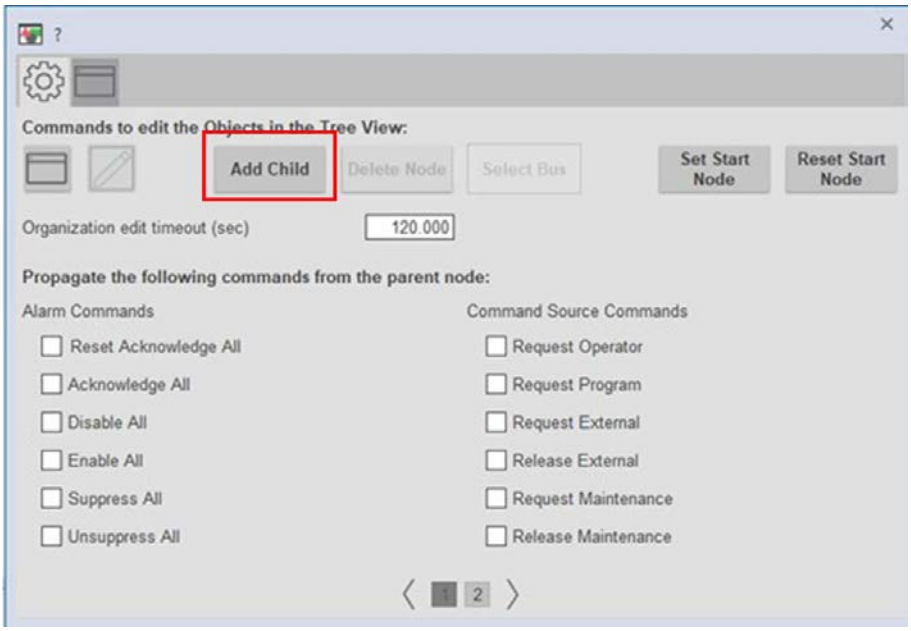
2. Enable Edit mode

   In Edit Mode, you can define the Bus that is associated with each Node, define parents or children, and define propagation configuration.

3. Select <root> to display the edit options.
4. Select the edit button.

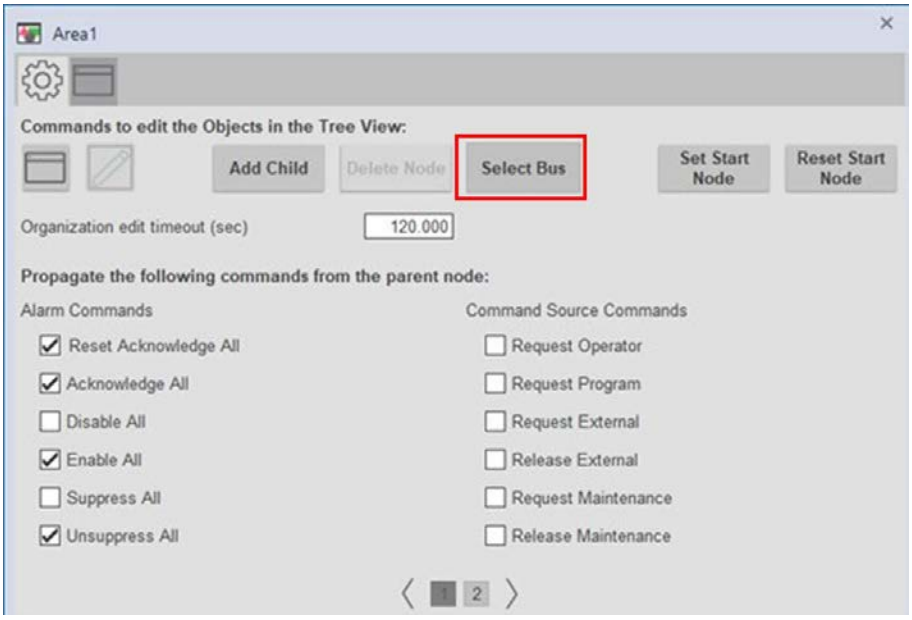


5. Select the Add Child button to add a node.

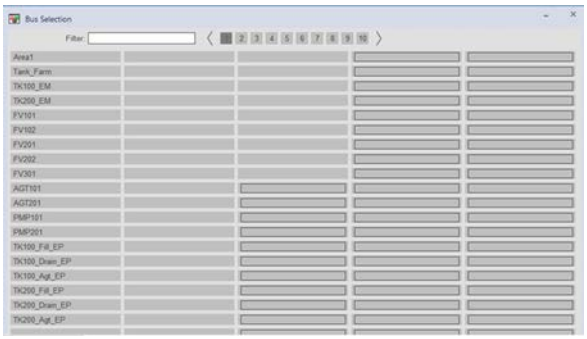


This adds a ? to the display tree.

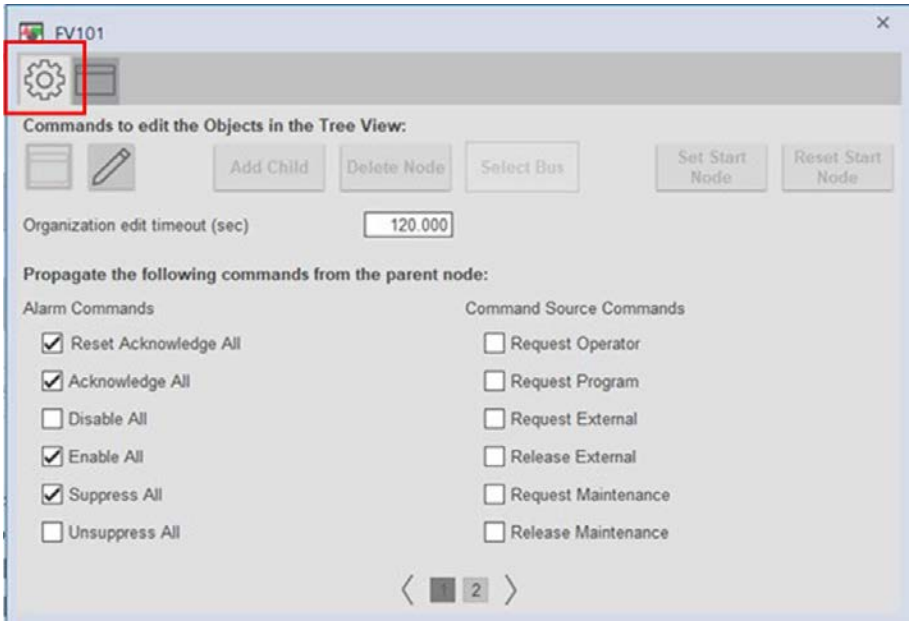
6. Select the ? to edit the node.
7. Select the edit button and click the Select Bus button to assign a bus element to the new node element.



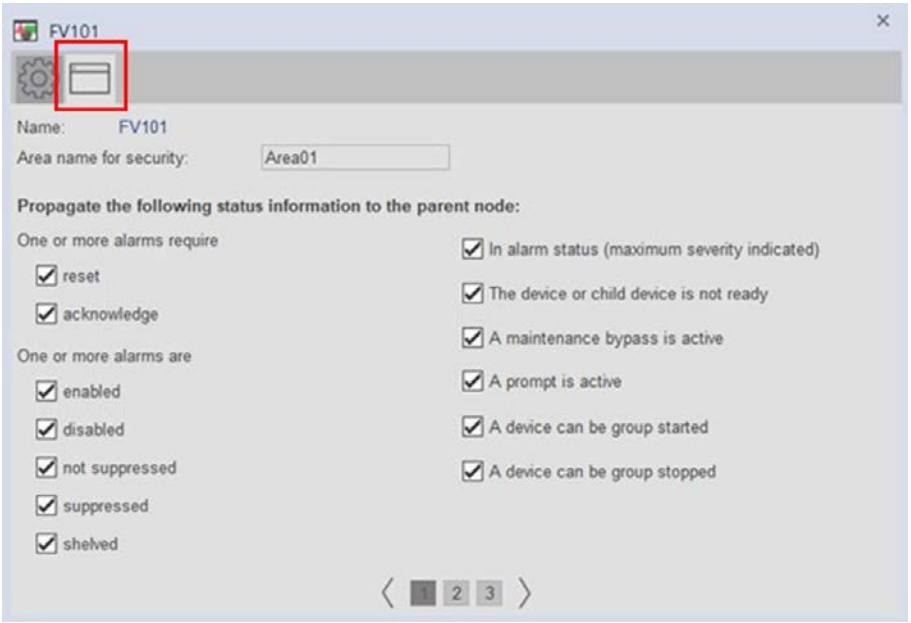
8. Select the bus element.



9. Define how to propagate commands (Engineering Tab).



10. Define how to propagate status (HMI Tab).



11. Define navigation (HMI Tab, page 3).

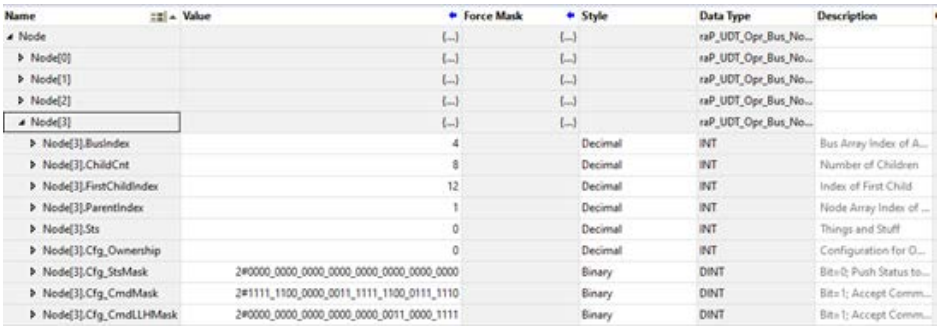
IMPORTANT Navigation selections above that are marked with an * use the Extended Tag Property @EngineeringUnit for the Bus Array instance (for example, Bus[37].@EngineeringUnit) because the @Navigation Extended Tag Property is not available in the user-defined data type (UDT) bus array.

Option	Description	Engineering Unit Extended Tag Property
nowhere - node not navigable	There is no configured navigation for the node	not used
Bus Faceplate	For use when the node does not have an associated display or faceplate. The generic faceplate can be configured to show any of the standard bus commands or status information.	not used
Device Faceplate	Show a standard library faceplate (or other faceplate that uses the same navigation method)	Device Tag. For example, "/DATA::[Tank_Farm]AGT101"
Device Faceplate with Shortcut	For use with faceplates that require the path as the third display parameter (These include Sequencer, Area, Unit, EM, and EP)	Device Tag. For example, "/DATA::[Tank_Farm]uTK100"
Specific Display	A custom display. Display parameters are: #1 - read/write tag (constant 2) #2 - Bus instance tag from the bus array #3 - Area security tag from OrgView	FactoryTalk View display name

Complete Organization Tree for this Example

The organization tree updates the Node array in the controller.

Node Element	Organization Tree Element
Node[0]	<root>
Node[1]	Area_Tank_Farm
Node[2]	uTK100
Node[3]	uTK200



Set Start Node

You build the complete node tree once. Once you have a complete node tree, you can select which node to start for each client. For example, you can have one client view only the uTK100 portion and another client view only the uTK200 portion.

1. Select the start node.
2. Click the pencil button and click the Set Start Node button to define the start node.

   For example, this view shows unit uTK100 as the start node. The other nodes are not viewable by this client.

Node Array Guidelines

Node array elements maintain the location of a single bus object in the organizational tree. Each entry that you make in the Node Tree by adding a child underneath a parent that child becomes an element in the Node array. Bus objects can exist as parents or children in multiple locations in the Node array.

- The maximum size of the node array is 3000 elements.
- Size the node array double the size of the bus array. This provides flexibility for expansion as well as assigning bus objects to more than a single node tree location.
- Do not change the node array manually within the Logix Designer application.
- Do not directly reference node array elements in logic. The node array location of a bus object can change. As children are added to the node tree, the nodes below those children will be shifted downward. As children are deleted, the nodes below will be shifted upward. The raP_Opr_OrgDeviceCtrl object is provided in versions 5.30.00 of the library and later to allow a designer to manipulate nodes in the tree if necessary.

If you create your organization via the PlantPAx Configuration Tool, the Node array is correctly configured.

Node Tooltip Information

Tooltip information is provided when you hover over a node. When a bus object is owned, the tooltip will provide the name of the parent bus object that currently has ownership. When the Tree View edit toggle is "on" the tooltip text provides Node index, Bus Index, Current Owner Name, and Owner Bus Index information. When the Tree View edit toggle is "off" the tooltip text provides the Current Owner Name and the Bound/Unbound status information.

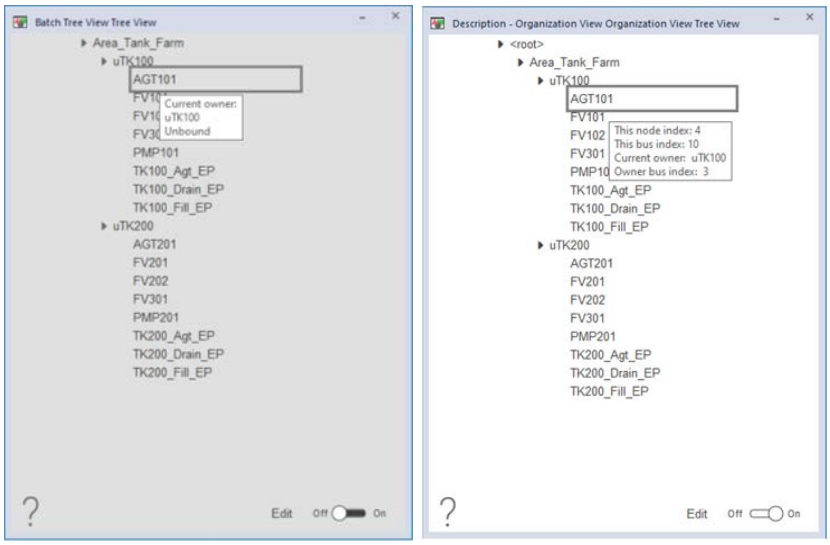
Additional context is provided in the following table:

Current Owner Tooltip Text	Description
None	The bus object is program ownable. It is not currently owned by a parent.
n/a	The bus object either does not have an associated command source or it has a command source with a program class that does not exist.
n/i	Ownership is suppressed at this node

A node status text is provided below the current owner to indicate when a node has been manipulated using an unbind or node command from the raP_Opr_OrgDeviceCtrl object.

Node Status Tooltip Text	Description
Unbound(1)	The bus object program ownership is unbound. This allows a user to place the object command source into operator mode as needed while still maintains ownership status.
DoNotOwn	Ownership at this node is suppressed.
Excluded	The object is unbound and excluded from organization accumulation at the parent.
DoNotPrp	Propagation of status and commands are suppressed at this node.

(1) Version 5.20.00 of the library provides tooltip indication for Bound/Unbound Status only. Releases before 5.20.00 do not provide node status tooltip text.



Status Indicators

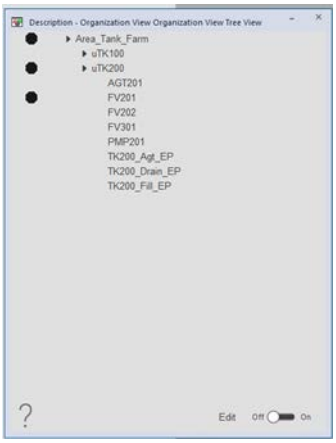
When configured to be available, specific status' can be viewed from any node in an organizational tree via the Bus faceplate. Select status' are represented by breadcrumbs, which appear in the organizational tree next to the nodes that are affected. Status' related to alarms, command source, and virtualization are supported. The status' available on the faceplate are determined by the configuration previously entered.

When a condition occurs which produces a status point, those statuses may be relative to this object or any of its children. The active maximum priority alarm status symbol is displayed.

Status Symbol	Description
	This object or one of its children is not ready.
	Object 'not usable' by parent. Either the object is owned by another parent, or the object's command source helps prevent program control.
	This object or one of its children is alerting the operator (Attention.)
	This object or one of its children is in Virtual.
	This object or one of its children has an active Maintenance Bypass.
	This object or one of its children has an alarm inhibited. Only displayed if an active alarm is not present and no acknowledgment is required.
	This object or one of its children requires an alarm acknowledgment. Only displayed if an active alarm is not present.
	This object or one of its children has an active alarm with urgent priority.

Status Symbol	Description
	This object or one of its children has an active alarm with high priority.
	This object or one of its children has an active alarm with medium priority.
	This object or one of its children has an active alarm with low priority.

Example of the Not Ready symbol propagating up the tree from FV201 to the Area_Tank_Farm node:



Arbitration

The optional Arbitration function manages optional queues for shared devices when using the Owner function. It takes an ownership acquisition request, which is pending , and places it on the appropriate class queue. Any pending ownership request is presented to the ownership function as a request in the order in which it resides on the queue. Once the current Owner releases its request, the next in the queue can take ownership.

The Arbitration queue:

- Enables user-defined arbitration rules to be applied to shared resources within a class.
- Handles and maintains multiple simultaneous requests for ownership.
- Manages a FIFO of the IDs of Ownership requests within a single command source class. (Operator, Program, External, Maintenance)

In the following example:

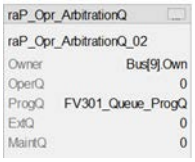
- TK100 and TK200 both have FV301 in their organization tree.

- Program an object for FV301 (Bus[9])

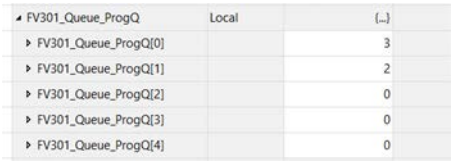
IMPORTANT For ownership commands and statuses to propagate, you must map the parameter pins as indicated in the example.

The parameter pins that are connected must align with the bus object index.

- There is an optional queue for each class request (Operator/Program/External/Maintenance). Currently, only Program is used, the rest are reserved for future functionality.



You can program logic to manipulate or reorder the entries on the individual queues. This queue shows ownership requests from Bus[3] (TK200) and Bus[2] (TK100).



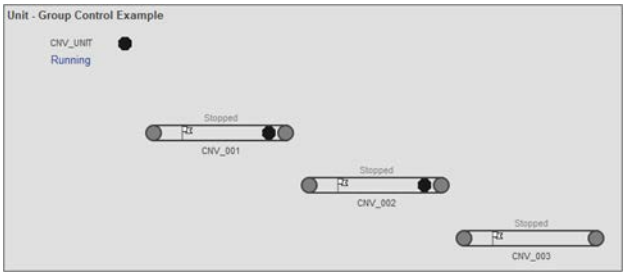
IMPORTANT The user should not directly add or delete items on the individual queues.
These functions are handled by the ownership functionality.

For object and visualization parameters, see Object and Visualization Parameters, PROCES-RD201.

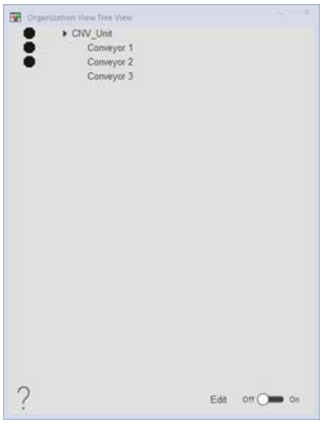
Unit Group Control

Use of the organization bus is a pre-requisite to using the Unit object. This is because the Unit object is designed to group children together and provide a propagation mechanism for aggregating status from child objects and issuing commands to child objects. Units can represent any group of equipment. This can be a tank within the S88 model, or it can represent a group of conveyors for a material handling application. The Unit provides four user-defined state commands that allow the designer to start and stop equipment as a group. The LLH commands for Emergency Stop, Software Stop, Permissive OK, and Interlock OK can be mapped to the children of the unit to prevent the grouped equipment from starting when the Unit object is not ready.

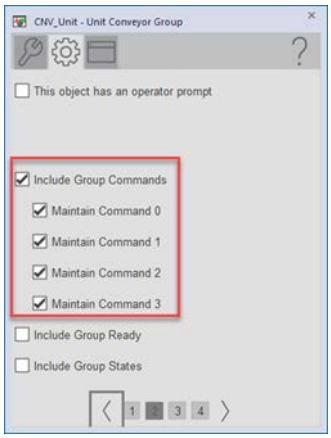
The following example demonstrates how a group of three conveyors can be controlled from the Unit faceplate. Depending upon the designer's preference, the equipment in the group can be started in a cascaded fashion or simultaneously. The unit in this example provides a conveyor start group command to start CNV_003 first and cascade back to CNV_001 as the downstream conveyors receive running feedback. A conveyor stop group command is provided to stop the conveyors in a similar fashion and a conveyor immediate stop group command is provided to stop all conveyors simultaneously.



The organizational structure is provided below. The conveyor equipment must be the immediate children of the Unit object. The node configuration for each object must be set to allow the propagation of the Line Level High Commands and the Group Commands for this functionality to work.



To enable the group command buttons on the Unit faceplate the unit must be configured to include group commands from the engineering faceplate as well as have defined a description for each group command bit. The maintain command configurations will latch each group command request until the corresponding state status bit = 0. The include group ready and include group states configurations determine if the Unit objects will check the Inp_Sts or Inp_RdyOk inputs. When these settings are off, the Unit uses the bus inputs.



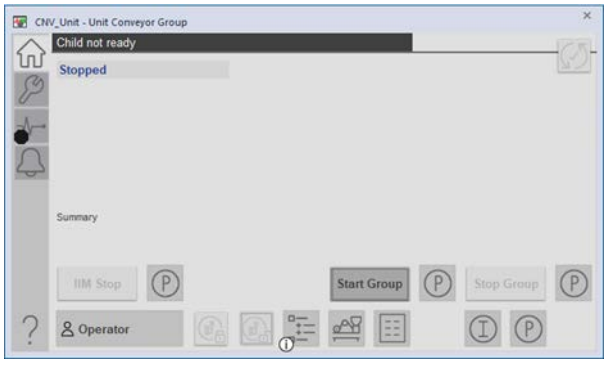
The configuration results in the following unit faceplate.

Unit Command 1 (Bus cmd.28) – Stop Group

Unit Command 2 (Bus cmd.29) – Start Group

Unit Command 3 (Bus cmd.30) - Unused

Unit Command 4 (Bus cmd.31) - Immediate Stop Group

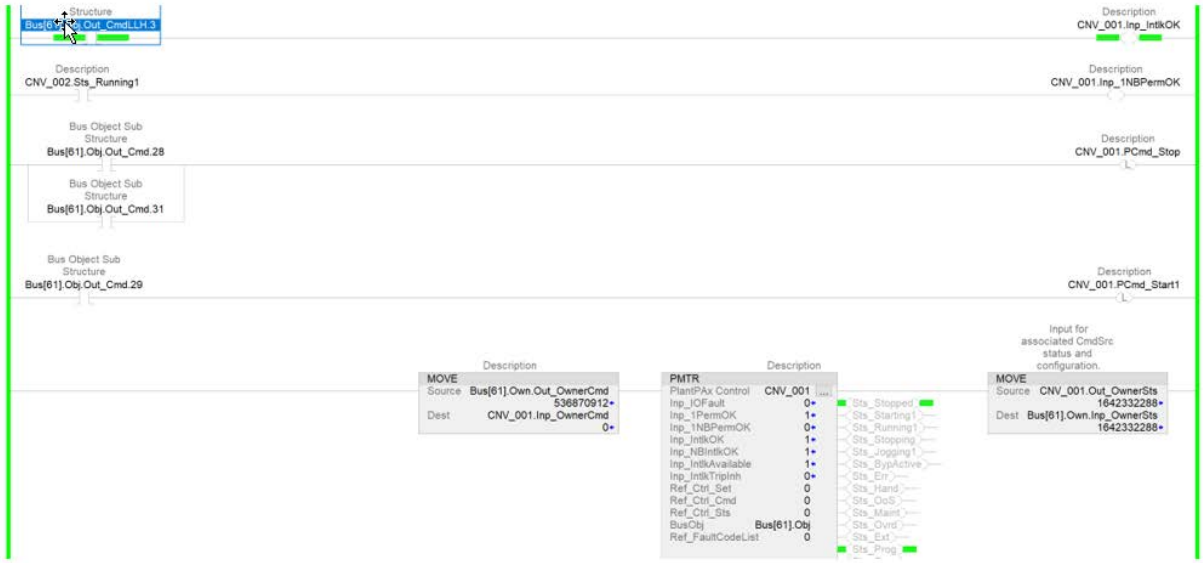


Unit Group Object Logic Example

The Unit conveyor group example uses the bus object assignments in the table below. The conveyor bus object elements must be referenced directly in the example logic due to the process object blocks not mapping the Unit line level high and state request bus commands and statuses internally. The group control bus commands and statuses are intended to provide the designer flexibility to define each state according to a wide variety of application requirements.

Bus Element	Name	Description
Bus[50]	CNV_Unit	Conveyor Group
Bus[61]	CNV_001	Conveyor 1
Bus[62]	CNV_002	Conveyor 2
Bus[63]	CNV_003	Conveyor 3

The following logic defines the motor input group logic for CNV_001. The Conveyor 2 Running status is mapped to the permissive OK input of conveyor 1 to ensure that the downstream conveyor starts first. Conveyor 3 is the most downstream conveyor and instead maps the Unit permissive OK status Bus[x].Obj.Out_CmdLLH.2.



Similarly, the CNV_003 object checks for the CNV_001 and CNV_002 Sts_Stopped signal when a group stop is requested to allow for the most upstream conveyors to stop first. An immediate stop request would bypass this check.

Bus Output	Motor Input	Description
Bus[x].Obj.Out_CmdLLH.3	MTR.Inp_IntlkOK	Group Interlock OK
Bus[x].Obj.Out_Cmd.28	MTR.PCmd_Stop	Group Stop Command
Bus[x].Obj.Out_Cmd.31	MTR.PCmd_Stop	Group Immediate Stop Command
Bus[x].Obj.Out_Cmd.29	MTR.PCmd_Start1	Group Start Command
Bus[x].Obj.Out_CmdLLH.2	MTR.Inp_1NBPermOK	Group Permissive OK

The following logic defines the motor output group logic for CNV_001. It is important to note that the status mapping for each of these states is the inverse of the target state for the corresponding unit group command. The aggregated status at the Unit object is defined as "At least one child is not Started" so the conveyor stopped status is mapped to the bus input status. This is what controls the availability of the group commands from the Unit faceplate.

Motor Output	Bus Input	Description
MTR.Sts_Running1	Bus[x].Obj.Inp_Sts.28	Equipment not in State 1
MTR.Sts_Stopped	Bus[x].Obj.Inp_Sts.29	Equipment not in State 2
NOT MTR.Sts_Stopped	Bus[x].Obj.Inp_Sts.31	Equipment not in State 4

Hardware Organization Bus

The logic that is demonstrated in the previous organization example can be duplicated for the purpose of created an organizational tree for the project hardware. The hardware organization bus functionality utilizes a set of hardware-centric Add-On Instructions that are bus capable. These Add-On Instructions are listed in the table below.

Hardware Bus Add-On Instruction	Description
raP_Dvc_LgxCPU_5x80(1)	Logix CPU Utilization
raP_Dvc_Lgx_ChangeDet	Logix Change Detection
raP_Dvc_LgxRedun	Logix Redundant Controller Monitor
raP_Dvc_LgxTaskMon	Logix Task Monitor
raP_Dvc_LgxModuleSts(2)	Logix I/O Module Connection Status

(1) raP_Dvc_LgxCPU_5x80 will not accept any bus commands from a parent or provide bus statuses as a child. It is intended to be used as the root object in the hardware organization tree. You can issue the commands that are listed below from the Bus Faceplate for the Logix CPU bus object to command all children accordingly.

(2) raP_Dvc_LgxModuleSts is the only hardware bus Add-On Instruction that uses the Not Ready (bit 8), Physical/Virtual (bits 10 and 11), and Bypassed/Check Connection (bits 26 and 27) status/commands described in the tables below. When used with the hardware bus organization, the Sts_IOfault output of this Add-On Instruction includes child module IO fault status.

The Hardware Bus provides:

- One-click alarm management commands for all I/O Modules connected to a single controller.
- One-click virtual mode or bypass mode commands for all I/O Modules that are connected to a single controller.
- Nearest point of failure error detection for connection paths.

The following statuses are propagated up the tree from the children to parent. These statuses are visible from the Hardware Tree View or the Bus Faceplate. Statuses 8, 9, 10, 11, 26, and 27 are specific to the raP_Dvc_LgxModuleSts.

HWBus[x].Out_Sts Bit	Status Produced and Propagated	Description
0	Alarms Active	At least one alarm is active for this object or its children
1	Alarms/Object to be Reset	At least one alarm is ready for reset for this object or its children
2	Ready for Reset	At least one object or child is ready for reset
3	Alarms Enabled	At least one alarm is enabled for this object or its children
4	Alarms Disabled	At least one alarm is disabled for this object or its children
5	Alarms Unsuppressed	At least one alarm is not suppressed for this object or its children
6	Alarms Suppressed	At least one alarm is suppressed for this object or its children
7	Alarms Shelved	At least one alarm is shelved for this object or its children

HWBus[x].Out_Sts Bit	Status Produced and Propagated	Description
8	Not Ready	I/O Module Connection Status is not "running" (Connected) and not bypassed
9	Bypass	Maintenance Bypass device status
10	Physical	At least one object or child is in Physical
11	Virtual	At least one object or child is in Virtual
26	Check Connection	This object or its children has a connection that is not bypassed
27	Bypassed	This object or its children has a connection that is bypassed

The IO Fault and Alarm Gate bus inputs are used within the raP_Dvc_LgxModuleSts Add-On Instruction. These inputs get propagated up the tree and allow the Add-On Instruction to set the Sts_IOFault when a child module has an IO Fault. The alarm gate is used to notify child modules that a parent node has already raised an alarm and it does not need to.

HWBus[x].Out_CmdLLH Bit	Status Issued and Propagated	Description
8	IO Fault	IO fault active
9	Alarm Gate	Gate child alarm when parent alarm is already active

The commands that are listed below can be issued from parent to child using the Generic Bus Faceplate. Commands 10,11,26, and 27 are specific to the raP_Dvc_LgxModuleSts.

HWBus[x].Inp_Cmd Bit	Commands Issued and Propagated	Description
3	Disable alarms	Disable all alarms
4	Enable alarms	Enable all alarms
5	Suppress alarms	Suppress all alarms
6	Unsuppress alarms	Unsuppress all suppressed alarms
7	Unshelve alarms	Unshelve all shelved alarms
10	Request Virtual	Request all to be in Virtual
11	Request Physical	Request all to be in Physical
26	IO Connection Bypass	Do not check IO Connection Status
27	IO Connection Check	Check IO Connection Status

Create the Hardware Organization Logic

Before you begin, import the process library Add-On Instructions into your controller project. All logic must be in one controller. The following Add-On Instructions are required:

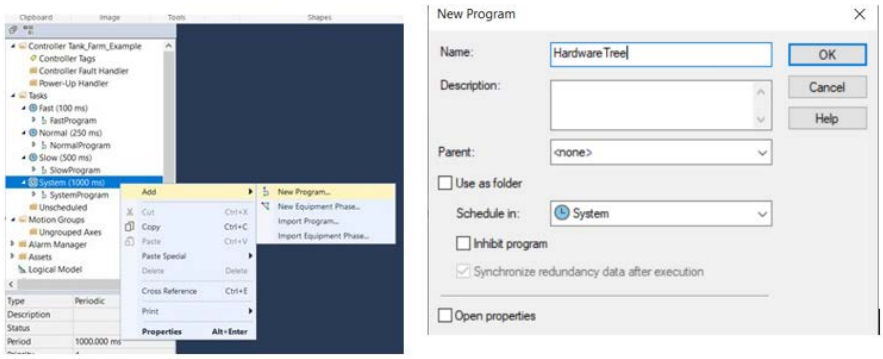
- raP_Opr_OrgScan Organizational Scan
- raP_Opr_OrgView Organizational View

IMPORTANT Application Code Manager and the PlantPAx Configuration Tool provide simplified workflows that are more efficient to initially configure the hardware organization.

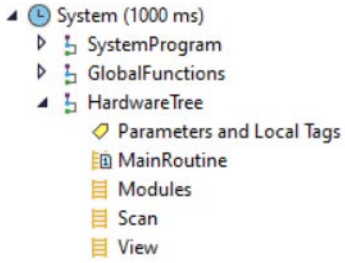
Create the HardwareTree Program

In a process controller project, create the HardwareTree program

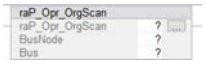
1. Add a new program in the appropriate Task and name it HardwareTree. The default task is the System task.



2. Add a new routine to the HardwareTree program and name it MainRoutine. Repeat the process for a modules, scan, and view routine. Add JSR instructions to the MainRoutine for the modules, scan, and view routines.



3. Open the Scan routine and in the Ladder Diagram editor, add an raP_Opr_OrgScan Add-On Instruction to manage the propagation of commands/status and Ownership within the Hardware Organization Tree.



Create Array Tags

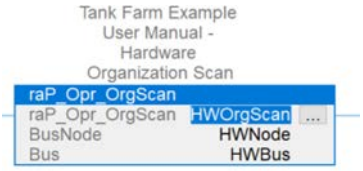
In the next steps, you create two array tags for these parameters that are used in this Add-On Instruction:

Bus array with an element for each piece of hardware in your I/O configuration. Make the array larger than the number of devices so you have room for future additions.

BusNode array with an element for each piece of hardware in the I/O configuration and its relationship to other hardware. Make the node array twice as large as the Bus array so that you have room for future additions

1. Right-click the raP_Opr_OrgScan ? and create a new, controller-scoped tag HWOrgScan of data type raP_Opr_OrgScan.

2. Right-click the BusNode ? and create a new, controller-scoped array tag HWNode of type raP_UDT_Opr_Bus_Node. The array tag must be named HWNode. Edit the array elements to have 100.
3. Right-click the Bus ? and create a new, controller-scoped array tag HWBus of type raP_UDT_Opr_Bus. The array tag must be named HWBus. Edit the array elements to have 50. Create more elements than your original list of bus elements to leave space to add future elements.



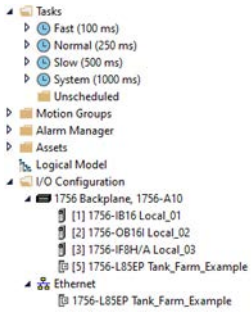
4. In the View routine and in the Ladder Diagram editor add an raP_Opr_OrgView Add-On Instruction for each HMI client that will be using the hardware organizational tree. This Example assumes five HMI clients so there are five individual rungs.
   - Right-Client the raP_OprView ? and create a new tag HWOrgView of datatype raP_Opr_Orgview. Edit the Data Type and enter the array dimensions so that you have an element for each HMI Client.
   - Enter the BusNode, Bus, and Orgscan tags you created for the HWOrgScan instruction. Select one HWOrgView[x] element for each rung/instance to correspond with each HMI Client.

Define the HWBus Elements

Similar to the software bus example outlined n this document, each entity that is grouped

into a hardware organization must be assigned a unique identifier. The HWBus array provides this mechanism, in addition to providing an interface to exchange commands and status.

Example I/O Configuration and Task Model

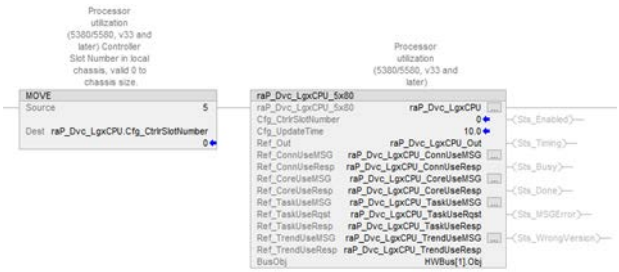


The elements can be in any order, and you can use any names that are appropriate for your application. The Change Detection and Redundancy Monitor Add-On Instructions were not used in this example.

Bus Element	Name	Description
Bus[0]		Reserved
Bus[1]	Tank_Farm_CLX	Controller
Bus[2]	Local_01	Local Controller Rack - Slot 1
Bus[3]	Local_02	Local Controller Rack - Slot 2
Bus[4]	Local_03	Local Controller Rack - Slot 3
Bus[5]	Task_Monitor_100ms	Fast Task Monitor
Bus[6]	Task_Monitor_250ms	Normal Task Monitor
Bus[7]	Task_Monitor_500ms	Slow Task Monitor
Bus[8]	Task_Monitor_1000ms	System Task Monitor

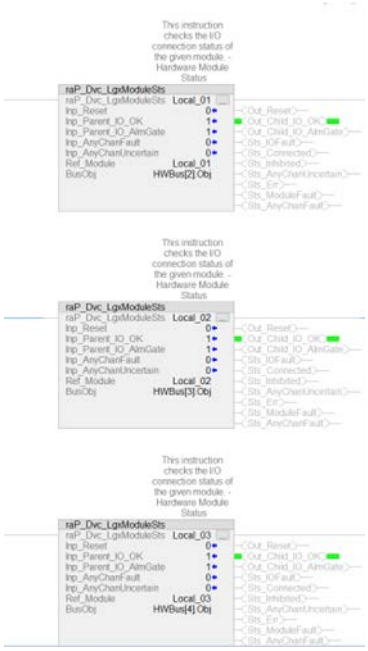
Configure the LCPU Instance

Import the PlantPAx raP_Dvc_LgxCPU_5x80_5_30_00_Rung.L5X. This defines the Logix CPU Monitor. Assign HWBus[1].Obj to the BusObj InOut Parameter.



Configure the Module Status Instances

Import the PlantPAx raP_Dvc_LgxModuleSts_5_30_00_AOI.L5X. This creates the Logix module status Add-On Instruction. Create an instance of this Add-On Instruction for each IO Module in the I/O Configuration. Assign HWBus[2].Obj, HWBus[3].Obj, HWBus[4].Obj to the BusObj InOut Parameter for the Local_01, Local_02, and Local_03 instances respectively.

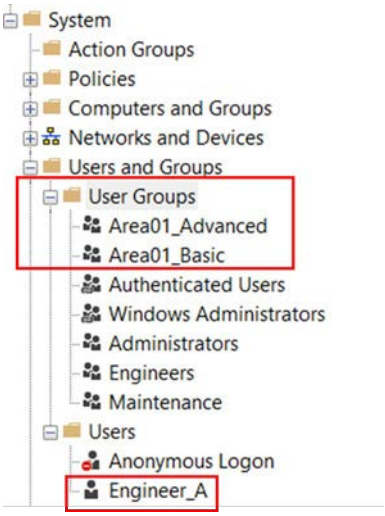


Configure the Task Monitor Instances

Import the PlantPAx raP_Dvc_LgxTaskMon_5_30_00_AOI.L5X. This creates the Logix task monitor Add-On Instruction. Create an instance of this Add-On Instruction for each task in the project. Size the raP_Dvc_LgxTaskMon tag accordingly to the number of tasks in the project. Each task has an instance with raP_Dvc_LgxTaskMon[x]. Assign HWBus[5].Obj, HWBus[6].Obj, HWBus[7].Obj, HWBus[8].Obj to the BusObj InOut Parameter for the Fast, Normal, Slow, and System task instances respectively.

Create the Organizational Tree in the HMI Client

Make sure that you have the application configured with the appropriate user groups. This example uses the default Area01_Basic and Area01_Advanced user groups with a user Engineer_A.



Configure the Client Display

1. Select and copy the 'Hardware Tree' button from the '(raP-5_30-SE) Graphic Symbols - Cross Functional.ggfx' global object file. Paste this button to the desired display.
2. Add the following lines to the client startup macro. The Template_ClientStartup macro file provides an example that can be uncommented and adjusted as needed. The template example uses the HWOrgView element 0 and /Area1/DATA::[Hardware] for the controller topic. Another startup macro must be created for each HMI client in the system so that each client can be assigned another HWOrgView element. The macros DefineShowTreeCmd and DefineShowHWTreeCmd are referenced in the startup macros and must exist in the project.

Line	Description
Define HW_RedefineShowTreeCmd DefineShowHWTreeCmd X	X = Client number (HWOrgView element) 0 in this example for client 0
HW_RedefineShowTreeCmd /Area/DATASERVER::[TOPIC]	Area = Area for data server DATASERVER = data server name TOPIC = shortcut name (path to controller) DATA:[Org_Example] in this example

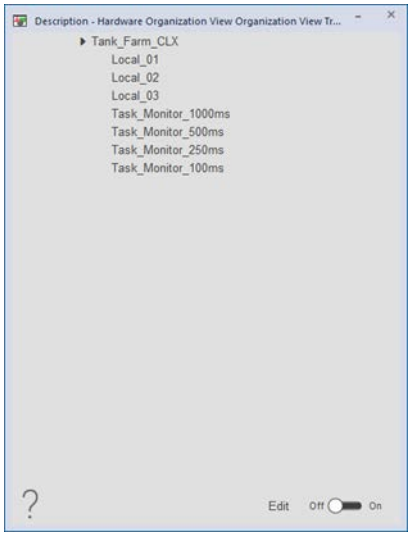
See Client Startup Macro on page 107to complete this configuration.

IMPORTANT The PlantPAx Graphic Framework Tool provides an interface for defining the client startup macros in the L1 configuration.

3. Generate the Client Display

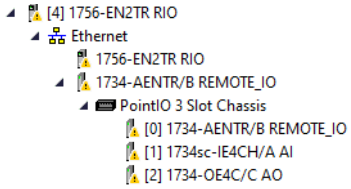
4. Follow the steps detailed in the software organization example Build the Node Tree on page 154 to add the HWBus objects to the hardware tree view. The Tank_Farm_CLX LCPU instruction is used as the parent object with the Logix module status and task monitor bus objects as children.

   The hardware tree view organization appears similar to the following structure:

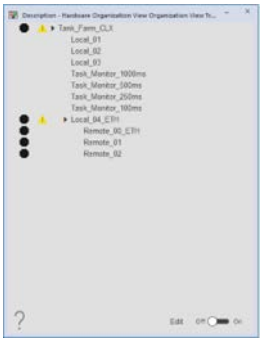


Hardware Tree Alarm Gating

Status and commands can be propagated through the tree for alarming and indications of individual module issues. Optional gating allows alarms from modules higher in the tree to gate alarms of modules lower in the tree to avoid 'nuisance' alarms. By propagating fault status downward though a tree hierarchy, a single fault status can be used to indicate any fault in the chain. This is useful for consolidating fault status when a module supplying input is located in a remote location. Consider the I/O configuration below. An Ethernet card is added to the I/O configuration from the previous hardware tree example for a remote POINT I/O™ rack.



The parent module in this case recognizes that the child modules are also in alarm and get those child alarm statuses in the hardware tree. The Local_04_ETH module status instance shows an in-alarm status for connection status and gate that status at the Point I/O modules that are its children in the hardware tree.



Notes:

Ownership (raP_Opr_Owner)

The raP_Opr_Owner (Ownership) Add-On Instruction extends the functionality of the PCMDSRC (Command Source) instruction to allow for ownership requests and owner ID book-keeping functionality.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

Use this instruction when it is desirable to maintain ownership IDs and manage ownership arbitration between the ownership classes (Opr, Prog, Ext, and Maint).

The raP_Opr_Owner functionality is included in the Bus Organizational UDT (raP_UDT_Opr_Bus). It is not necessary to create a separate raP_Opr_Owner instance to obtain ownership functionality between parent child relationships that are configured in organizational trees that are processed by a raP_Opr_OrgScan instruction.

Functional Description

The raP_Opr_Owner Add-On Instruction is used to accept and process ownership requests by ID utilizing a PCMDSRC (Command Source) instruction for class arbitration rules. The basic class arbitration rules are implemented by the PCMDSRC instruction, which ownership requests are allowed, which ownership requests 'win' when multiple ownership requests are made by different classes of owners, and so on.

The raP_Opr_Owner instruction uses positive value DINTs as ownership IDs.

This instruction yields status as to the current owner IDs maintained if any. The ultimate 'winning' owner class and ID are also produced as status.

The state of 'Organization' is also indicated through status. This status indicates if the device/ object is in the correct PCMDSRC state for its ultimate owner and the status of any children if present and aggregated (that is, through the BUS organization). In this way you can determine if this device/ object is in the proper condition for operation.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline

implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_Owner_5.30.00_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

The raP_Opr_Owner Instruction uses no visualization files or components.

Operations

Command Sources

The raP_Opr_Owner instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Opr_Owner Instruction uses no alarms.

Virtualization

The raP_Opr_Owner Instruction has no Virtualization capability.

Execution

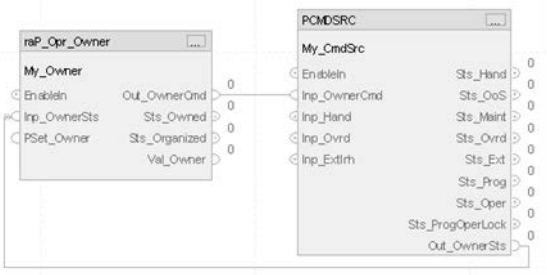
The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	The raP_Opr_Owner instruction clears all owner status and ID fields, and releases any ownership that is currently applied when scanned false or with the EnableIn=0.
Powerup (prescan, first scan)	The raP_Opr_Owner instruction clears all owner status and ID fields on PreScan/first scan.
Postscan	No SFC Postscan logic is provided.

For more information, see the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010.

Programming Examples

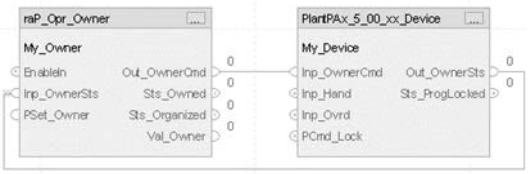
The raP_ Opr_Owner instruction must be coupled with a PCMDSRC instruction or a device/object that contains a PCMDSRC instruction. There are input and output parameters to accomplish the interface:



The 'Owner.Out_OwnerCmd' output parameter sends any pending ownership requests to the PCMDSRC owner interface parameter of the PCMDSRC (Inp_OwnerCmd).

The 'Owner.Inp_OwnerSts' input parameter receives existence, configuration, and current state information from the associated PCMDSRC (Out_OwnerSts).

When using a Process Object 5.00.xx and later device/object these parameters are supplied on the object to interface the ownership instruction with the object's internal PCMDSRC:



When using a Process Object 5.00.xx and later device/object as a participant on the Bus, the Bus referenced ownership interface parameters are used as input and output to the device/object:

Graphic Symbols

There are no graphic symbols or HMI graphic support for the raP_Opr_Owner instruction.

Faceplates

There is no faceplate for the raP_Opr_Owner instruction.

Notes:

Arbitration (raP_Opr_ArbitrationQ)

The raP_Opr_ArbitrationQ (Arbitration) Add-On Instruction extends the functionality of the raP_Opr_Owner (Ownership) instruction to allow for the queuing of ownership requests within an ownership class.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

Use this instruction if you want to extend the functionality of the raP_Opr_Owner to include multiple ownership requests within the same ownership class. One raP_Opr_ArbitrationQ instruction can be associated to a single raP_Opr_Owner to perform optional queuing of any of the four ownership classes (Oper, Prog, Ext, Maint).

Functional Description

The raP_Opr_ArbitrationQ Add-On Instruction is used to manage arrays of owner IDs for each class of ownership. Ownership requests made of the associated raP_Opr_Owner are intercepted by the raP_Opr_ArbitrationQ instruction and placed into a queue (DINT array) in the order in which they are received. By default, the earliest entry is used by the raP_Opr_Owner for ownership evaluation. As ownership requests and releases are made of the raP_Opr_Owner, the raP_Opr_ArbitrationQ instruction manages the addition and deletion of these requests and releases in the respective queues.

Use of the raP_Opr_ArbitrationQ instruction is optional. It extends the functionality of the raP_Opr_Owner instruction. Use the raP_Opr_ArbitrationQ instruction when there are multiple entities that could simultaneously request owner ship of this entity AND you wish to maintain their order of request or manipulate the requests for prioritization.

Items in the queues can be reordered by user programming to accommodate prioritization schemes.

IMPORTANT	You should never add or delete IDs on the queue by user programming. Addition and deletion of IDs is done by the instruction itself based on the ownership requests made by the associated raP_Opr_Owner instruction.

The following image shows how a raP_Opr_ArbitrationQ instruction configured with the raP_Opr_Owner instruction 'My_Owner' as its associated owner instruction. Further, it is configured to have queues for Oper, Prog, and Maint owner classes. It does not use a queue for the External owner class:

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline

implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_ArbitrationQ_5.30.00_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

The raP_Opr_ArbitrationQ Instruction uses no visualization files or components.

Operations

Command Sources

The raP_Opr_ArbitrationQ instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Opr_ArbitrationQ Instruction uses no alarms.

Virtualization

The raP_Opr_Owner Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

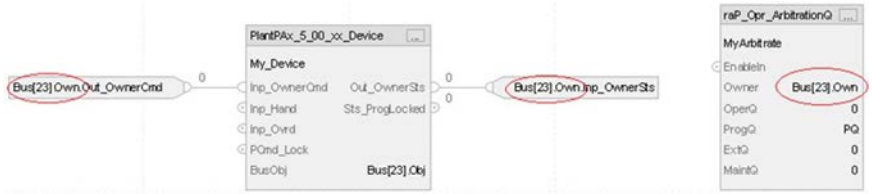
Condition	Description
EnableIn False (false rung)	The raP_Opr_Arbitration instruction clears all queues and counters when scanned false or with the EnableIn=0.
Powerup (prescan, first scan)	The raP_Opr_Arbitration instruction clears all queues and counters on PreScan/first scan.
Postscan	No SFC Postscan logic is provided.

For more information, see the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010.

Programming Examples

The example in the Function Description section shows the basic use of the raP_Opr_ArbitrationQ Add-On Instruction for extending an ownership instruction. Typically, the raP_Opr_ArbitrationQ instruction is used in association with a Bus-resident entity. The following shows an arbitration instruction that is associated with a Bus referenced entity. The owner field is the '.Own' sub-element of the Bus structure:

A Bus enabled PlantPAx® device/object has its own Bus element. When extending the ownership functionality with the arbitration instruction, use the same Bus element reference that is used for that device/object:



Graphic Symbols

There are no graphic symbols or HMI graphic support for the raP_Opr_ArbitrationQ instruction.

Faceplates

There is no faceplate for the raP_Opr_ArbitrationQ instruction.

Notes:

Organizational Scan (raP_Opr_OrgScan)

The raP_Opr_OrgScan (Organizational Scan) Add-On Instruction processes user-defined organizational trees to propagate status information from child nodes to parent nodes, and to propagate commands from parent nodes to child nodes. Further ownership requests and status can be propagated between parent and child nodes. The functionality to edit any organizational trees is built into this Add-On Instruction and edit requests are executed synchronously with the organizational scan.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

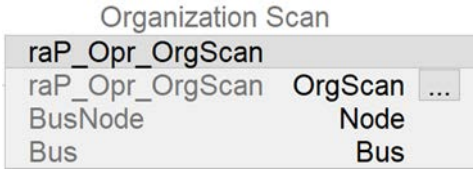
Guidelines

Use this instruction if you want to build parent child relationships between controller-resident entities and propagate status, command, and ownership functionality between them.

Functional Description

The raP_Opr_OrgScan Add-On Instruction is used to propagate the information between elements of organizational trees and allow ownership relationships between those elements. It also maintains the organizational tree editing functions and the edit token ownership.

A single raP_Opr_OrgScan instruction is to be used to scan all organizational trees in a controller. As such, a single instance of the Add-On Instruction is to be scanned unconditionally in a slow, low- priority task.



'Node' is an array that is composed of elements of type 'raP_UDT_Opr_Bus_Node'. This array must be of sufficient length to accommodate the maximum possible number of organizational tree nodes. Typical systems can have 100...1000 nodes depending upon the complexity of the organizational trees. Significant scans can occur when Node arrays with greater than 500 elements are used.

'Bus' is an array that is composed of elements of type 'raP_UDT_Opr_Bus'. This array must be of sufficient length to accommodate the maximum possible number of devices/objects that you wish to place on the Bus.

IMPORTANT	The name of the Node array must be 'Node' and the name of the Bus array must be 'Bus' for raP_Opr_OrgView operation.

Edit functionality and Edit Token management is also maintained in the raP_Opr_OrgScan instruction. All editing of the nodal organizational trees occurs through this instruction instance.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_OrgScan_5.30.00_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

The raP_Opr_OrgScan Instruction uses no visualization files or components.

Operations

Command Sources

The raP_Opr_OrgScan instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source. A raP_Opr_Owner and PCMDSRC instances are present in the Add-On Instruction to facilitate edit token ownership by an HMI client through a raP_Opr_OrgView instance.

Alarms

The raP_Opr_OrgScan Instruction uses no alarms.

Virtualization

The raP_Opr_OrgScan Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Opr_OrgScan instruction must always be scanned true. In relay ladder logic, the raP_Opr_OrgScan instruction must be by itself on an unconditional rung.
Powerup (prescan, first scan)	All status and internal limits are cleared on prescan/first scan and array bounds and existing node configuration are checked.
Postscan	No SFC Postscan logic is provided.

For more information, see the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010.

Programming Examples

The example in the Function Description section shows the basic use of the raP_Opr_OrgScan Add-On Instruction. The raP_Opr_OrgScan can be executed from any controller language. But must be executed unconditionally. The scan update of all nodes in a system may require long scan times, therefore it is recommended to be executed from a slow, low-priority task. Further, the associated timeouts (Program, and so on) should be lengthened accordingly.

Graphic Symbols

There are no graphic symbols or HMI graphic support for the raP_Opr_OrgScan instruction.

Faceplates

There is no faceplate for the raP_Opr_OrgScan instruction.

Organizational View (raP_Opr_OrgView)

The raP_Opr_OrgView (Organizational View) Add-On Instruction continuously scans the organizational trees and queues the information into a standard hierarchical tree view for presentation on a single HMI (FactoryTalk® View SE) client.

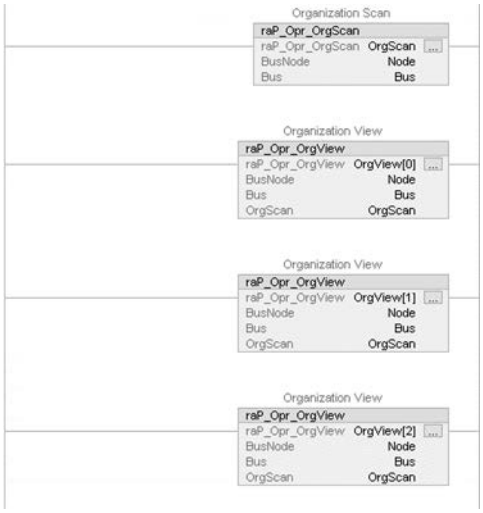
For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

Use this instruction if you want to display organizational tree information on an HMI client. Another instance of the raP_Opr_OrgView instruction must be instantiated and scanned within the user project for each intended HMI client.

Functional Description

The raP_Opr_OrgView Add-On Instruction is used to read and format organizational tree information into a standard tree view on the HMI. The Add-On Instruction automatically adjusts the tree view based on user edits of the organizational tree. It is intended to have one raP_Opr_OrgView instruction instance for each HMI client viewing the organizational trees in a controller. This is so each HMI client can have a tree view that is unaffected by the actions of another client (expand, collapse, edit, and so on).



Here there are three raP_Opr_OrgView instances that are associated with the primary raP_Opr_OrgScan instance to update three individual HMI clients.

IMPORTANT	An array of raP_Opr_OrgView backing tags must be created at the controller scope of name 'OrgView.' Each element of the array is used as the backing tag for each instance servicing a single HMI client. See Organizational Scan (raP_Opr_OrgScan) on page 143 for other naming requirements.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_View_5.30.00_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

Command Sources

The raP_Opr_OrgView instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Opr_OrgView Instruction uses no alarms.

Virtualization

The raP_Opr_OrgView Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Opr_OrgView instruction must always be scanned true. In relay ladder logic, the raP_Opr_OrgView instruction must be by itself on an unconditional rung.
Powerup (prescan, first scan)	All status, internal HMI buffers, and internal limits are cleared on prescan/first scan and array bounds and existing node configuration are checked.
Postscan	No SFC Postscan logic is provided.

For more information, see the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010.

Organizational Node Interface (raP_Opr_OrgDeviceCtrl)

The raP_Opr_OrgDeviceCtrl Add-On Instruction extends the functionality of the raP_Opr_OrgScan (Organizational Scan) and raP_Opr_Owner (Ownership) instructions to provide program inputs to suppress ownership, suppress propagation, or exclude a child from organized status accumulation at the child level.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

Use this instruction to optionally extend OOAP functionality by manipulating the ownership or propagation behavior at a single child in the organizational tree.

This instruction utilizes the parent bus index and child bus index inputs to find the relational node in the tree where that parent-child relationship exists. When edits to the tree occur which result in the shifting of the node array data, this instruction will detect those edits and update the relational node accordingly. Any active suppression or exclusion being asserted on the relational node will be maintained during tree edits and will shift as needed.

An instance of the raP_Opr_OrgDeviceCtrl instruction can be used for each parent/child relationship defined in the organizational tree to manage a given node and conditionally perform any combination of the following functions on the child:

- Do not own this child when ownership is requested by the parent.
- Suppress all command and status propagation to and from a child.
- Unbind ownership at the child and exclude it from children organized status aggregation at its parent.

The ownership and propagation functions can be useful in cases where a device or equipment group is shared amongst multiple parents but does not need to be owned in all situations. Suppressing ownership while still propagating status from the child allows that parent to monitor that child while still letting other parents claim ownership and use it as needed. If a child's node should be ignored completely, ownership and propagation can be suppressed at the same time. With these functions, if the child object goes into an alarm or not ready state, the parent is not affected. This can often be the case with certain parent objects that can be configured to stop or hold on to a child in alarm or not useable. When a child's ownership is suppressed by a parent it is no longer included in the accumulation of the Sts_ChildrenOwned and Sts_ChildrenOrganized outputs at the parent.

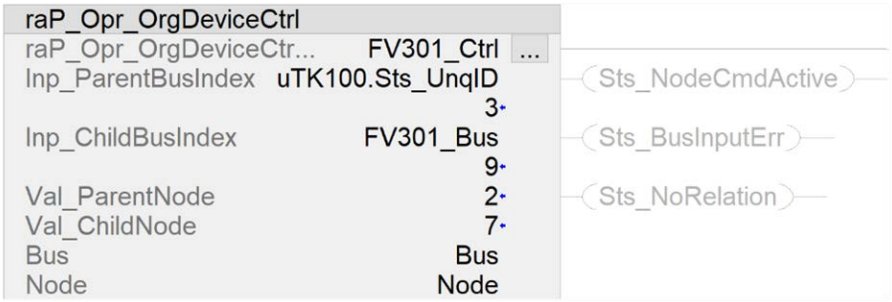
The exclusion function is valuable in scenarios where an operator may be required to put a child device into operator mode to adjust setpoints (PVSD, PPID, etc.). While excluded, the child will not be included in the aggregation of the parent's children organized status allowing the operator to place the object into operator mode and adjust setpoints. Excluding a child from organization at the node with the raP_Opr_OrgDeviceCtrl AOI will only exclude that child from the organized status of the parent to that node. A parent still retains ownership of the child while the child is excluded from organization.

If the Bus[ChildObjectIdx].Inp_ProgDoNotInclude is used to exclude the child from organization then that child will be excluded from organization accumulation in every location it exists in the tree (all nodes). Refer to the unbinding ownership section of chapter 4 in this document for more detail.

Functional Description

Use of the raP_Opr_OrgDeviceCtrl instruction is optional. It extends the OOAP functionality defined in chapter 4. This Add-On Instruction is used to alter the ownership and propagation behavior of a single node in the organizational tree. Multiple instances of the raP_Opr_OrgDeviceCtrl AOI can be defined to manage the behavior of multiple nodes in the organization as needed.

The following image shows how a raP_Opr_OrgDeviceCtrl instruction is configured with the parent and child bus index references. When the parent or child object is an raP_Opr_Area, raP_Opr_Unit, raP_Opr_EMGen, or raP_Opr_EPGen object the Sts_UnqID output parameter can be used as the bus index input. The designer can explicitly use the bus index number of the desired parent or child instead of a tag as well.



The following program command inputs are available for the controls designer to set in logic as needed to control the behavior of the selected node. All nodes will propagate commands and status, accept ownership, and are included in organization by default.

Node Commands[1]	Description
PCmd_SuppressOwn	1 = Suppress Ownership at Child
PCmd_UnsuppressOwn	1 = Unsuppress Ownership at Child. Node will be included in ownership by default unless a Pcmd_SuppressOwn command has been given
PCmd_Exclude	1 = Unbind and Exclude Child from Organization Status of Parent
PCmd_Include	1 = Bind and Include Node into Organization Accumulation. Node will be included by default unless a Pcmd_Exclude command has been given
Pcmd_DoNotPropagate	1 = Suppress propagation of Status and Commands From/To Parent
PCmd_Propagate	1 = Unsuppress Propagation of Status and Command form/to the child. Node will propagate by default unless a Pcmd_DoNotPropagate command has been given

(1) For more information on unbinding ownership and excluding from organization refer the Unbinding Ownership and Exclusion sections in Chapter 4.

The following status outputs are provided:

Status	Description
Sts_NodeCmdActive	1 = Node Ownership Suppression, Organization Exclusion, or Propagation Suppression Active
Sts_SuppressOwn	1 = Ownership at child node is being suppressed for the parent node
Sts_Exclude	1 = All inclusion of Ownership and Status For Parent/Child Bus Pair Suppressed at the Child Node
Sts_DoNotPropagate	1 = Propagation of Alarms and Status For Parent/Child Bus Pair Suppressed at the Child Node
Sts_BusInputErr	1 = Parent or Child Bus Index OOR
Sts_NoRelation	1 = No Relationship Found for Parent/Child Bus Pair in the Organization Tree

IMPORTANT The raP_Opr_OrgDeviceCtrl AOI supports only a single instance of a parent/child relationship in the node tree. Adding a parent with all its children in the organizational tree more than once will create multiple nodes that maintain the same parent child relationship. Doing so will result in unpredictable functionality of this Add-On Instruction. The raP_Opr_OrgDeviceCtrl AOI is capable of only finding a single node.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline

implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_OrgDeviceCtrl_5.30.00_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

The raP_Opr_OrgDeviceCtrl Instruction uses no visualization files or components.

Operations

Command Sources

The raP_Opr_OrgDeviceCtrl instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Opr_OrgDeviceCtrl Instruction uses no alarms.

Virtualization

The raP_Opr_OrgDeviceCtrl Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

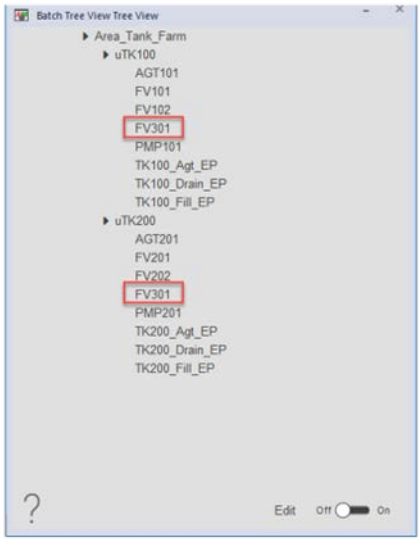
Condition	Description
EnableIn False (false rung)	The raP_Opr_OrgDeviceCtrl instruction will continue to monitor the node array for changes and shift relational node to be suppressed as needed.
Powerup (prescan, first scan)	No Powerup prescan logic is provided.
Postscan	No SFC Postscan logic is provided.

For more information, see the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010.

Programming Examples

Shared Child Device Example

Consider the tree structure example from Chapter 4. The objects uTK100 (Bus 3) and uTK200 (Bus 4) share the object FV301 (Bus 9) as a child. Neither uTK100 nor uTK200 in this case require FV301 during the Agitate or Fill phases. The raP_Opr_Arbitration() instruction can be implemented to allow the opposite tank to wait in a FIFO queue to gain ownership of FV301 when the other tank has completed. If this is not desirable, the control designer can selectively own the children under each tank based on the action required instead.



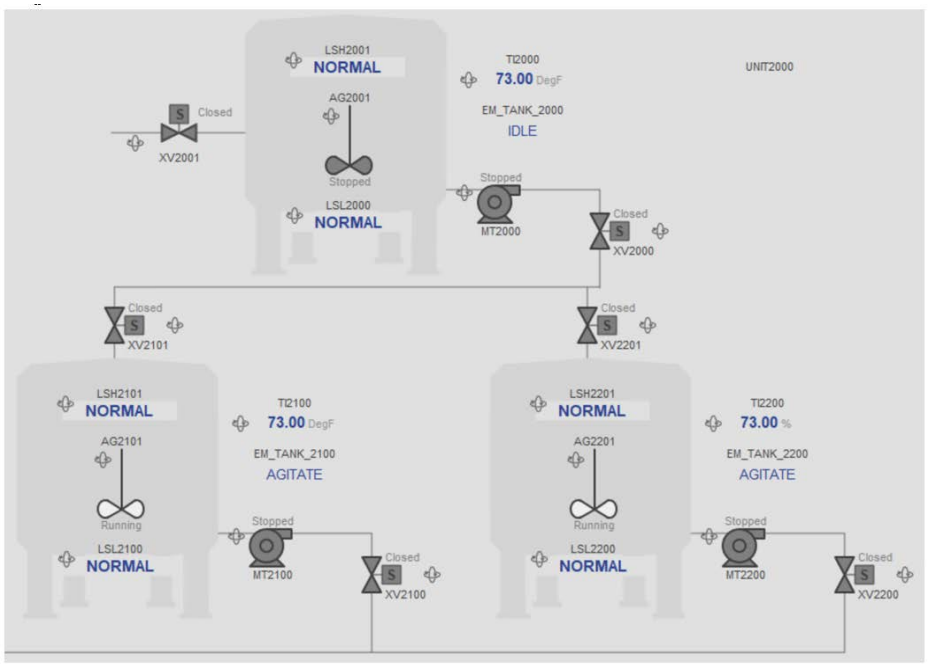
The FV301_Ctrl instance of the raP_Opr_OrgDeviceCtrl AOI below identifies node 7 that maintains the parent-child relationship between uTK100 and FV301. Selectively owning FV301 during the fill and agitate states allows TK200 to go into the Drain state without having to wait for TK100 to complete.

An identical program scoped instance of FV301_Ctrl can also be defined in the logic for uTK200 to be used to selectively own FV301 based on its requested state.

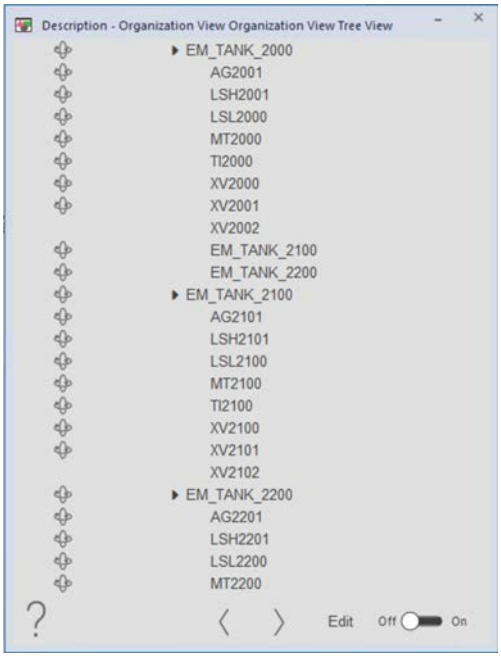
If an engineer were to edit the tree in a running system and add a new node for a Tank 300 unit under the Area_Tank_Farm parent, then all the child nodes underneath Tank 100 and Tank 200 will shift downward in the array to accommodate that addition. The raP_Opr_OrgDeviceCtrl AOI will detect edits such as this and update the FV301_Ctrl.Val_ChildNode accordingly. Any active suppression will be updated and shifted as well when the FV301_Ctrl.Val_ChildNode is updated.

Selectively Owning a Parent

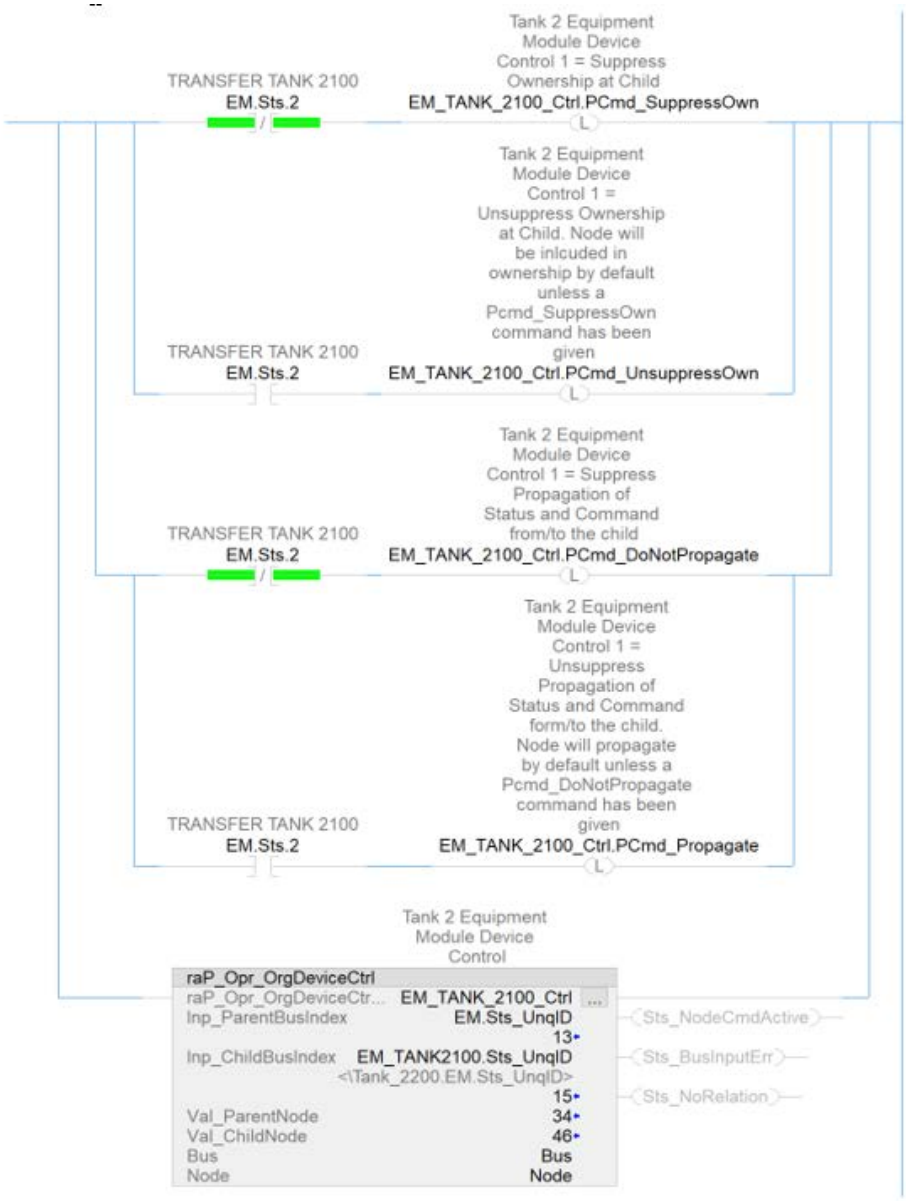
The second example below illustrates how the ownership suppression function can be used on another parent in the tree. Tank 2000 below (Pictured at the top) can transfer to either Tank 2100 or Tank 2200. If Tank 2000 is actively transferring to Tank 2100 then Tank 2200 must be left available to transfer out to the next area (Tank 3000).



By defining the tree hierarchy as shown below, the designer can give Tank 2000 the option to only own EM_TANK_2100 or EM_TANK_2200 depending on which transfer state it is in. Leaving EM_TANK_2200 unowned allows it to simultaneously transfer out to another tank.
EM_TANK_2100 can be placed as a child underneath another parent without its children being redefined in the tree again underneath it.

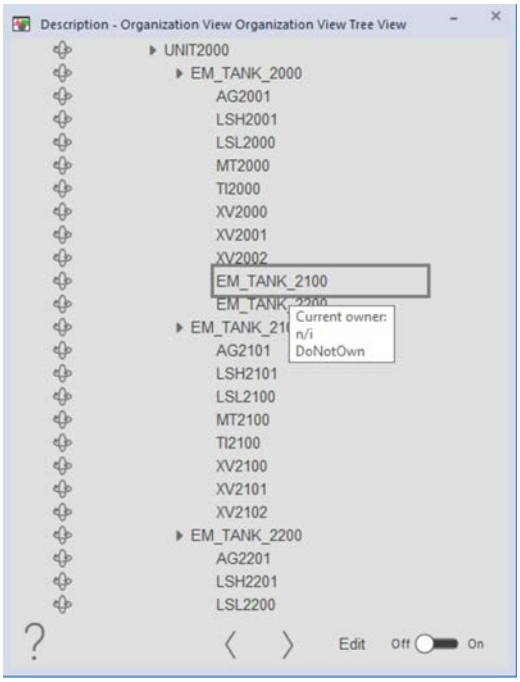


The logic below is defined in the EM_TANK_2000 program to suppress ownership and propagation of the Tank 2100 equipment module when the Tank 2000 equipment module is not in the transfer to Tank 2100 state.



When the EM_TANK_2100 child has ownership suppressed it is indicated with "DoNotOwn" on the Tree View tooltip display when hovering over that node. Similarly, when a node is excluded, the tooltip will indicate "Excluded" and if status and command propagation is suppressed the

tooltip will indicate “DoNotPrp”. The ownership suppressed tooltip indication will take precedence over all other node status texts.



Limitations

The use of the raP_Opr_OrgDeviceCtrl is dependent on the design of the organizational tree. In both examples above, the Add-On Instruction is used to find a single node in the tree that can then be manipulated as needed. If that parent-child relationship is added to another location in the tree in either case, then there would be 2 nodes in the tree maintaining the same relationship. The raP_Opr_OrgDeviceCtrl would not be capable of suppressing that second node resulting in undesirable functionality. Consideration should be made during the design of the tree to prevent such cases. In the second example, the EM_TANK_2100 and EM_TANK_2200 bus objects are added as children underneath EM_TANK_2000 to more easily manage tank transfers. These bus objects were added as nodes without their children in this extra location due to this limitation. Not re-adding each child of EM_TANK_2100 again underneath the EM_TANK_2000 parent allows the designer the capability to still use the raP_Opr_OrgDeviceCtrl AOI with EM_TANK_2100 devices (AG2101, XV2100, etc.). This approach also has the advantage of reducing the number of nodes used in the system.

Graphic Symbols

There are no graphic symbols or HMI graphic support for the raP_Opr_OrgDeviceCtrl instruction.

Faceplates

There is no faceplate for the raP_Opr_OrgDeviceCtrl instruction.

Process Area Module (raP_Opr_Area)

The raP_Opr_Area object groups Units together, and provides a propagation mechanism for aggregating status from Unit objects, and broadcasting commands to Unit objects.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

The Area group is based in a controller.

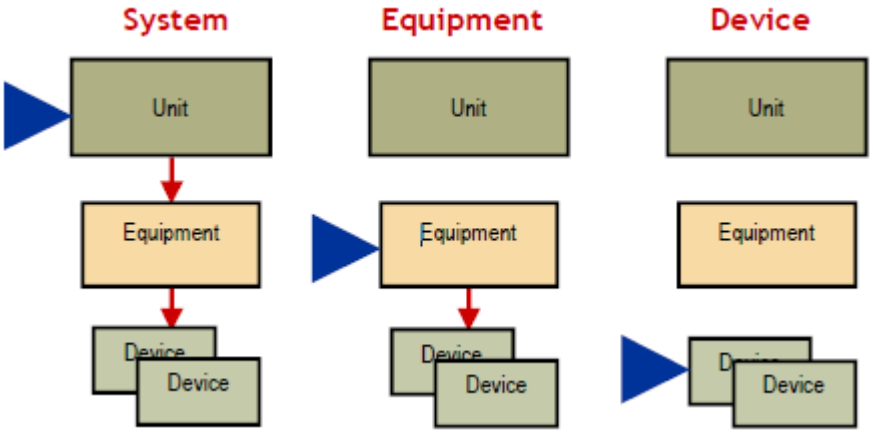
The Area is responsible for managing the equipment that is associated to that Area. These responsibilities include, but are not limited to, the following:

- Command Source management for a group of equipment.
- Alarm management for a group of equipment.
- Aggregate (propagation) status (for items such as: command source, alarm, configuration errors, and so forth) and provided “bread crumbs” for navigation.
- Provide broadcast (propagation) command mechanism.
- Detects failure conditions, such as Emergency Stop and Software Stop.
- Provides mechanism for extended Alarms.
- Monitor various Area failure conditions, and produce alarms.
- Provide a propagation mechanism to allow the Area to receive status from and send commands to a group of equipment.
- Provides the ability to produce a Software Stop condition based on any of the following:
  - Alarm from any lower-level object
  - Software Stop input
  - Area Alarm

Functional Description

Command Source Management

Allows you to interact with the system at various levels.



Use the PCMDSRC PlantPAx® instruction to manage the command source (owner) of an instruction or control strategy. For more information, see PlantPAx Process Control Instructions,
publication PROCES-RM215.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_Area_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

Command Sources

The raP_Opr_Area instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Opr_Area Instruction uses the following alarms, which are implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
E-Stop trip	Alm_EStopTrip	Raised when an emergency stop condition triggers a change in state of the Area.
S-Stop trip	Alm_SStopTrip	Raised when a software stop condition triggers a change in state of the Area.

Virtualization

The raP_Opr_Area Instruction has no Virtualization capability.

Execution

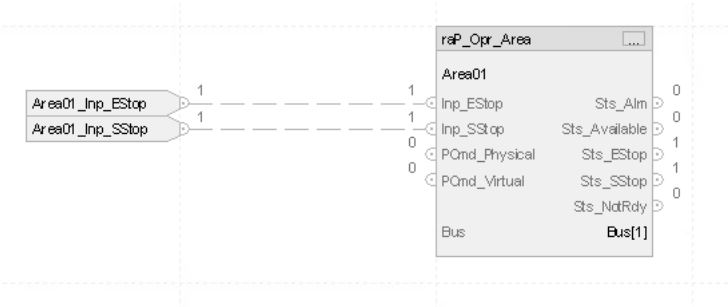
The handling of instruction execution conditions.

Condition	Description
EnableIn False (False Rung)	Handle processing for EnableIn False (False Rung) the same as if the Area were Disabled by Command. The Area outputs are de-energized and the Area is shown as Disabled on the HMI.
Powerup (Pre-scan, First Scan)	Handles processing of command source and alarms on Pre-scan and Powerup. On Powerup, the Area is treated as if it were Commanded to reset all program and operator commands
Postscan (SFC Transition)	No SFC Postscan logic is provided.

See Logix 5000 Controllers Add-On Instructions: Programming Manual, 1756-PM010 for more information.

Programming Example

The example in the Functional Description section shows the basic use of the raP_Opr_Area Add-On Instruction. Typically, the raP_Opr_Area instruction is used in association with a Bus-resident entity. The following example shows an area instruction that is associated with a Bus referenced entity. The area is also connected to 2 inputs, emergency stop, and software stop that can trigger two alarms. The area object typically is used in an S88 application but can be applied to suit numerous hierarchy layouts.



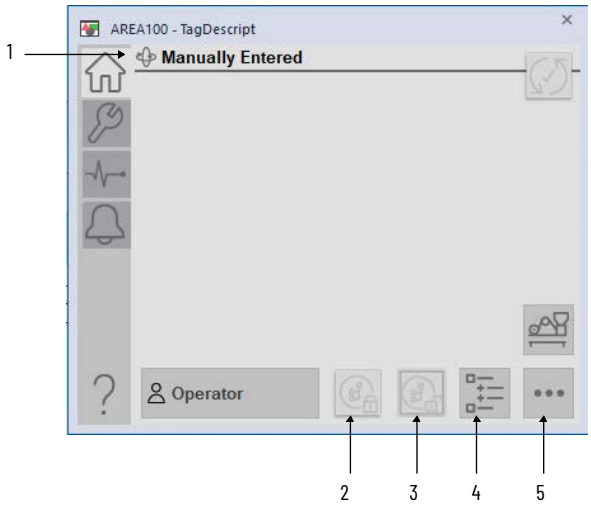
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactortyTalk Optix Graphic Symbol	Description
GO_PAREA 	raP_5_30_GS_raP_Opr_Area 	The raP_Opr_Area object groups Units together, and provides a propagation mechanism for aggregating status from Unit objects, and broadcasting commands to Unit Objects.

FactoryTalk View SE Faceplates

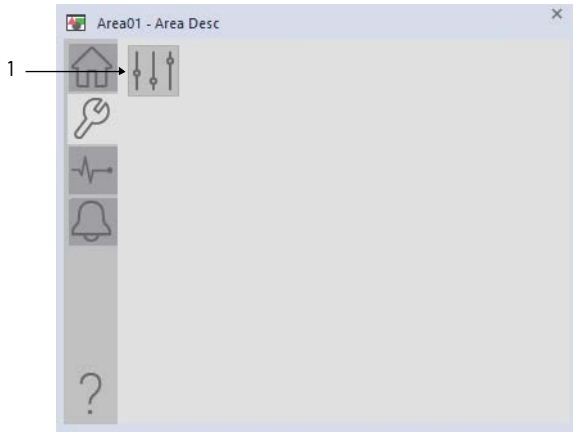
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab



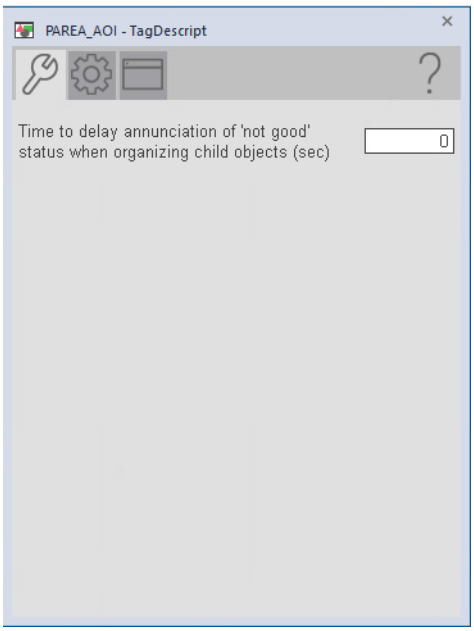
Item	Description
1	Displays the current state of the object
2	Acquire child command source
3	Release child command source
4	Display organizational tree view for this object
5	Display more information

Maintenance Tab



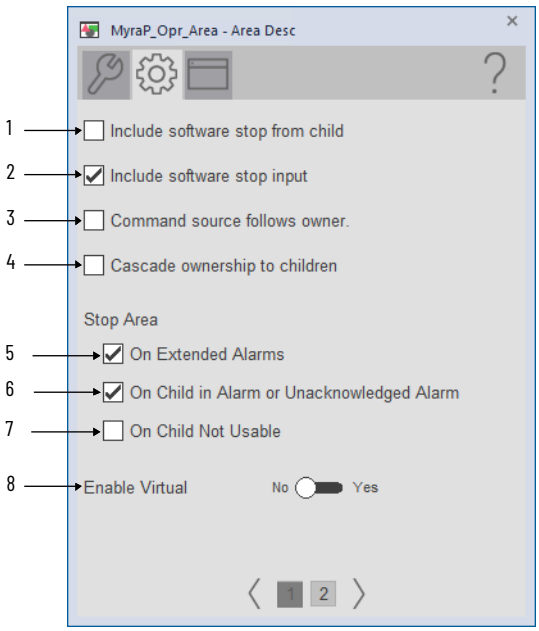
Item	Description
1	Display Advanced Properties

Advanced Maintenance Tab

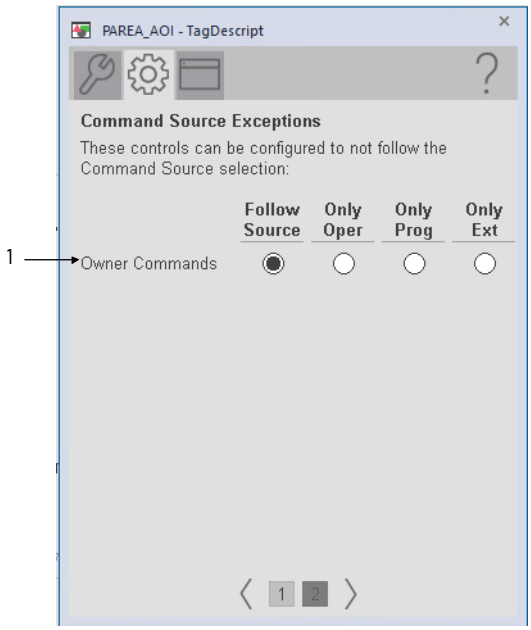


The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

Engineering Tab

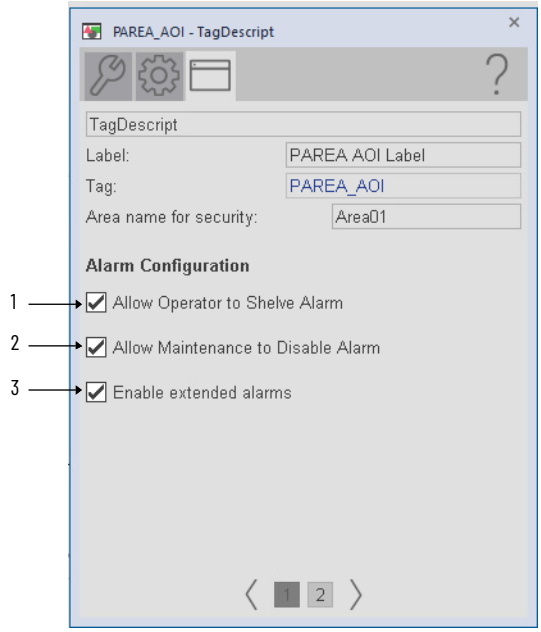


Item	Description
1	Select to include software stop from child
2	Select to include software stop input
3	Select to have the Command source follow the owner
4	Select to cascade ownership to children (children will be owned when this object is owned)
5	Select to stop unit on extended alarms
6	Select to stop unit when Child is in Alarm or Unacknowledged Alarm
7	Select to stop unit when Child cannot be put into Program or is owned by another owner.
8	Select yes to enable virtual mode

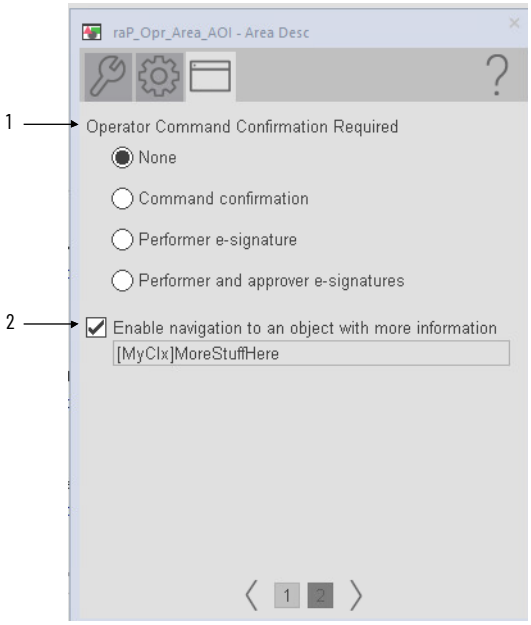


Item	Description
1	Use the radio buttons for the area owner commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).

HMI Configuration Tab



Item	Description
1	Select to allow Operator to shelve alarm
2	Select to allow Maintenance to disable alarm
3	Select to enable extended alarms



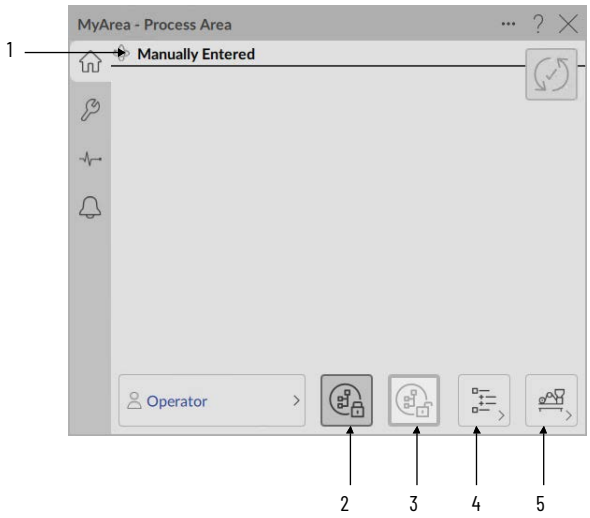
Item	Description
1	Select an option for Operator Command Confirmation Requirements
2	Select to allow navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the objects faceplate.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

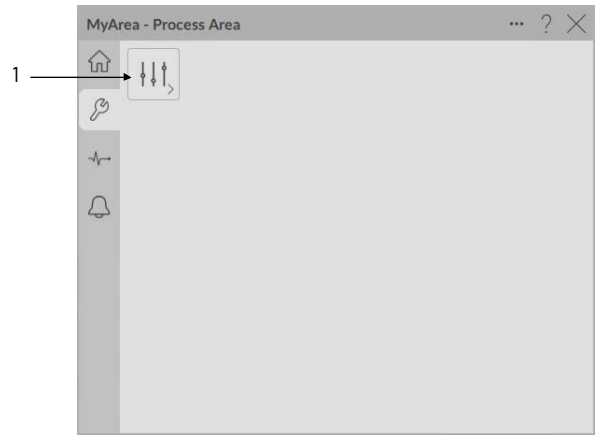
Any feature that is contained in the FactoryTalk® Optix™ faceplates has the same functionality as used in the FactoryTalk® View SE faceplates. See FactoryTalk View SE Faceplates on page 200

Operator Tab



Item	Description
1	Displays the current state of the object
2	Acquire child command source
3	Release child command source
4	Display organizational tree view for this object
5	Display more information

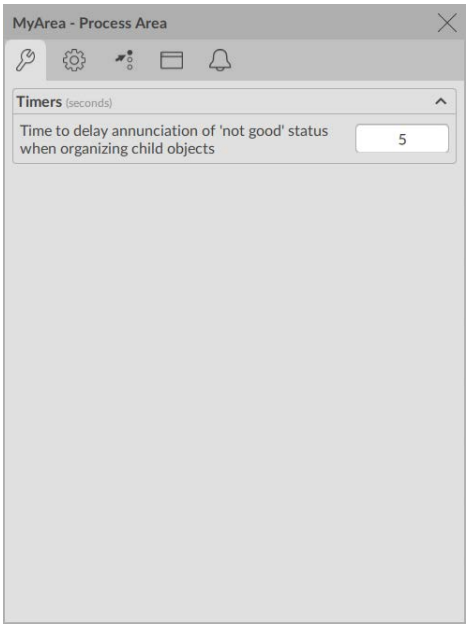
Maintenance Tab



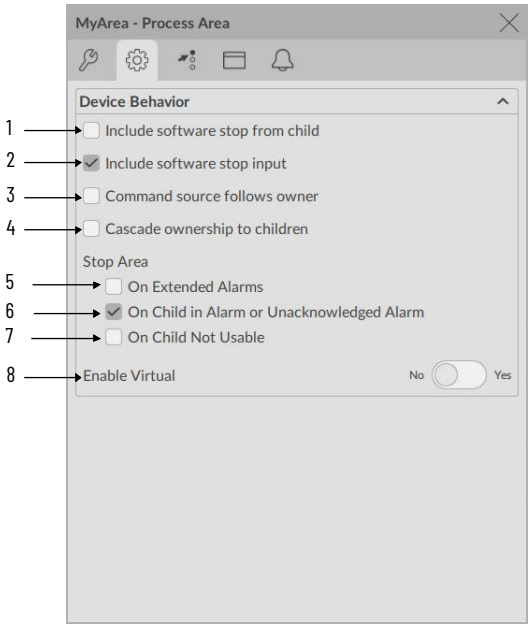
Item	Description
1	Display Advanced Properties

Advanced Maintenance Tab

The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

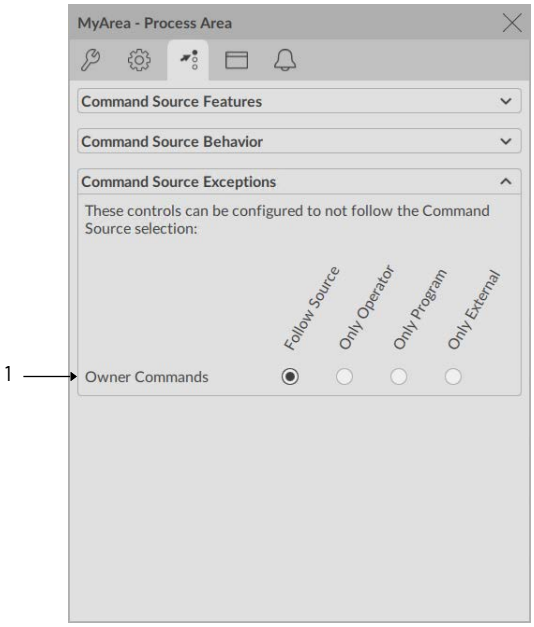


Advanced Engineering Tab



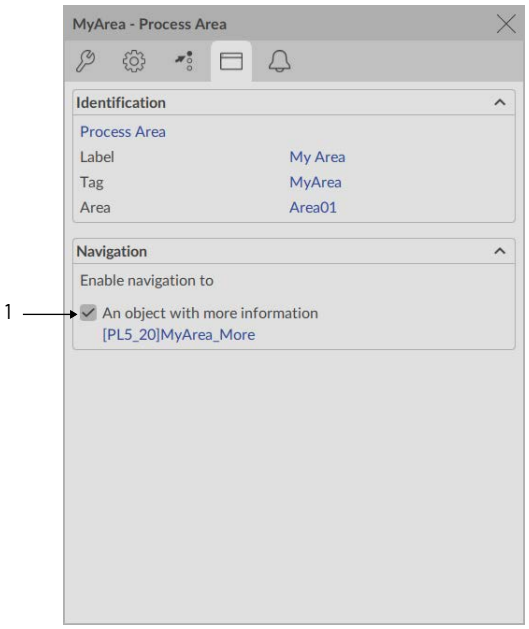
Item	Description
1	Select to include software stop from child
2	Select to include software stop input
3	Select to have the Command source follow the owner
4	Select to cascade ownership to children (children will be owned when this object is owned)
5	Select to stop unit on extended alarms
6	Select to stop unit when Child is in Alarm or Unacknowledged Alarm
7	Select to stop unit when Child cannot be put into Program or is owned by another owner.
8	Select yes to enable virtual mode

Advanced CmdSrc Tab - Command Source Exceptions



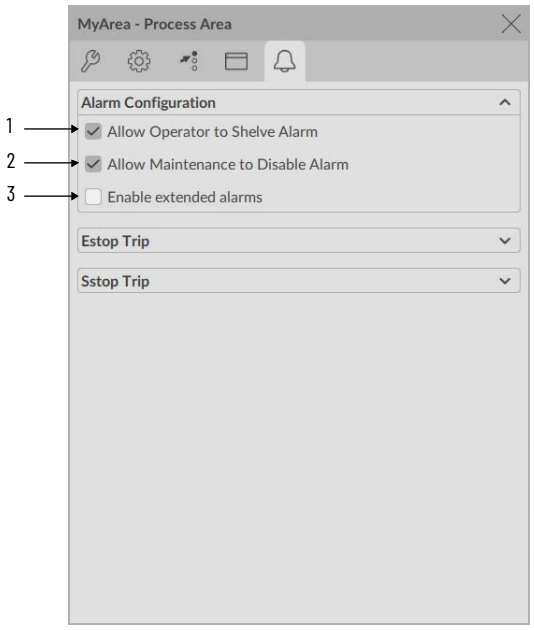
Item	Description
1	Use the radio buttons for the area owner commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).

Advanced HMI Configuration Tab



Item	Description
2	Select to allow navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the .@Library and .@Instruction extended tag properties to display the objects faceplate.

Advanced Alarm Configuration



Item	Description
1	Select to allow Operator to shelve alarm
2	Select to allow Maintenance to disable alarm
3	Select to enable extended alarms

Notes:

Process Unit (raP_Opr_Unit)

The raP_Opr_Unit object groups Equipment together, and provides a propagation mechanism for aggregating status from Equipment, and broadcasting commands to Equipment. As an example each vessel, tank, mixer, machine, etc... within the control system would be considered a Unit.

- Units are presumed to operate on only one batch at a time.
- Units operate relatively independently of one another.
- This term applies to both the physical equipment and the equipment entity.
- Examples of major processing activities are; react, crystallize, and make a solution.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

The raP_Opr_Unit (Process Unit) object controls a Unit in various command sources and monitors for fault conditions.

Use when:

- You want to consolidate status from groups of equipment. These statuses include:
  - Alarm Status
  - Alarm Priority
  - Command Source
  - Configuration Errors
- You want to manage and to following functions for a group of equipment, with a "global" set of commands:
  - Command Source
  - Alarm Acknowledge
  - Alarm Reset
- You want to apply permissive conditions to a group of equipment.
- You want to shut down groups of equipment based on a single alarm that occurs in any related equipment.
- You want to issue user-defined commands to equipment.

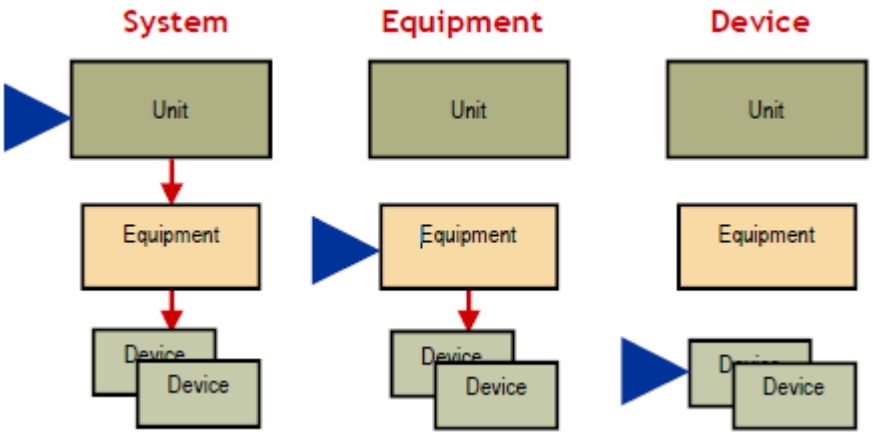
The raP_Opr_Unit object also:

- Provides an interface to parameter display, data entry and configuration.
- Provides an interface to resultant (report) data display and configuration.
- Provides interface to Prompt Response and configuration

Functional Description

Command Source Management

Allows the user to interact with the system at various levels.



Use the PCMDSRC PlantPAx® instruction to manage the command source (owner) of an instruction or control strategy. For more information, see PlantPAx Process Control Instructions,
publication PROCES-RM215.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

Controller File The raP_Opr_Unit_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

Command Sources

The raP_Opr_Area instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Program Structure

The raP_Opr_Unit Instruction may be implemented using a program as a container (recommended). The following table outlines suggested program structure and routine naming:

Routine	Description
Object Name	Contains raP_Opr_Unit instance, external function instances (Interlock, Permissive, Extended Alarms), and routine calls.
AlarmsSuppress	Contains raP_Opr_Unit alarm suppression logic.
Interlocks	Contains raP_Opr_Unit interlock mapping from interlock conditions to _Intlk block.
Parameters	Contains raP_Opr_Unit parameter mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to raP_Opr_Unit instance.
Permissives	Contains raP_Opr_Unit permissive mapping from permissive conditions to _Perm block.
Group Command Permissives	Contains raP_Opr_Unit group commands (1-4) permissive mapping from permissive conditions to _Perm block.
Reports	Contains raP_Opr_Unit report mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to raP_Opr_Unit instance.
ExtddAlarms	Contains raP_Opr_Unit instances of external alarm instances and trigger logic.

IMPORTANT The raP_Opr_Unit Instruction may be implemented without the program structure that is defined in the previous table; this is provided as an example.

Alarms

The raP_Opr_Unit Instruction uses the following alarms, which are implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
E-Stop trip	Alm_EStopTrip	Raised when an emergency stop condition triggers a change in state of the Unit.
Group Command 1 Fail	Alm_GroupCmd1Fail	Raised when the defined Group Command 1 fails to execute on the Unit.
Group Command 2 Fail	Alm_GroupCmd2Fail	Raised when the defined Group Command 2 fails to execute on the Unit.
Group Command 3 Fail	Alm_GroupCmd3Fail	Raised when the defined Group Command 3 fails to execute on the Unit.
Group Command 4 Fail	Alm_GroupCmd4Fail	Raised when the defined Group Command 3 fails to execute on the Unit.
Interlock trip	Alm_IntlkTrip	Raised when an interlock condition triggers a change in state of the Unit.
S-Stop trip	Alm_SStopTrip	Raised when a software stop condition triggers a change in state of the Unit.

Virtualization

The raP_Opr_Unit Instruction has no Virtualization capability.

Execution

The handling of instruction execution conditions.

Condition	Description
EnableIn False (False Rung)	Handle processing for EnableIn False (False Rung) the same as if the Area were Disabled by Command. The Area outputs are de-energized and the Area is shown as Disabled on the HMI.
Powerup (Pre-scan, First Scan)	Handles processing of modes and alarms on Pre-scan and Powerup. On Powerup, the Area is treated as if it were Commanded to reset all program and operator commands
Postscan (SFC Transition)	No SFC Postscan logic is provided.

See Logix 5000 Controllers Add-On Instructions: Programming Manual, 1756-PM010, for more information.

Local Message

The object raP_Opr_Unit utilizes local message display elements to display Step Names, Material Names, and Summary information. A default local message file is provided for each information type. This default local message file populates the local message display elements from tags in the controller. For Step Names and Material names, these are the same controller tags used in previous versions of the library. The difference is that 512 messages are available, rather than the 99 messages in the previous version. To upgrade from previous versions, developers need to add the local message file to the project and set the @Navigation property of the specified tag to the Local Message file name (see the following table).

Information	Default Local Message File	File Name Reference	Default Controller Data
Material Name	SystemMaterialNames	Sts_eMtrl.@Navigation	System.Enum.Materials[x].@Description
Step Description	SystemStepDescriptions	Sts_eStep.@Navigation	System.Enum.Step_Desc[x].@Description
Summary Information	SystemSummary	Sts_eSummary.@Navigation	System.Enum.Summary_Desc[1].@Description

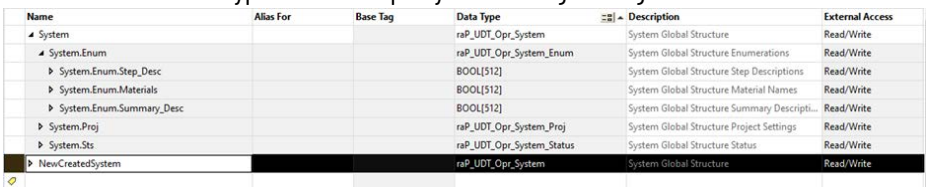
Users may add customized local messages for individual objects by creating a new local message file and populating the file with the customized strings or tag references. Then set the @Navigation property of the specified tag to the name of the new custom file.

FactoryTalk Optix Local Message

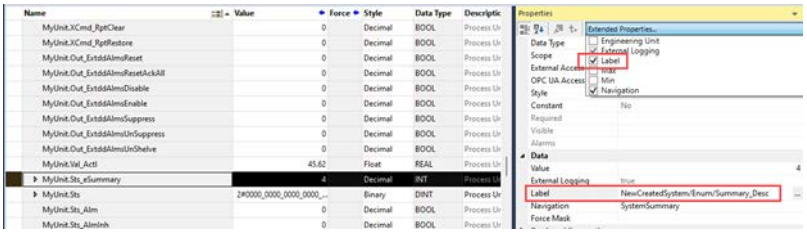
The object raP_Opr_Unit uses the @Label property of the specified tag to input the path for displaying Step Names, Material Names, and Summary information. FactoryTalk® Optix™ does not require Local Message files. For Step Names, Material Names, and Summary, these are the same controller tags used with FactoryTalk ViewSE. FactoryTalk Optix directly retrieves Controller Data using the path specified in the @Label property of Logix Designer. Users must set the path information into @Label property of the specified tag to a Controller Data Path (see the following table).

Information	Reference	Customized Controller Data Path	Default Controller Data
Material Name	Sts_eMtrl@Label	System/Enum/Materials	System.Enum.Materials[x].@Description
Step Description	Sts_eStep@Label	System/Enum/Step_Desc	System.Enum.Step_Desc[x].@Description
Summary Information	Sts_eSummary@Label	System/Enum/Summary_Desc	System.Enum.Summary_Desc[1].@Description

Users may add Customized Controller Data for individual objects by creating a new tag member with the Data Type "raP_UDT_Opr_System" in Logix Designer.

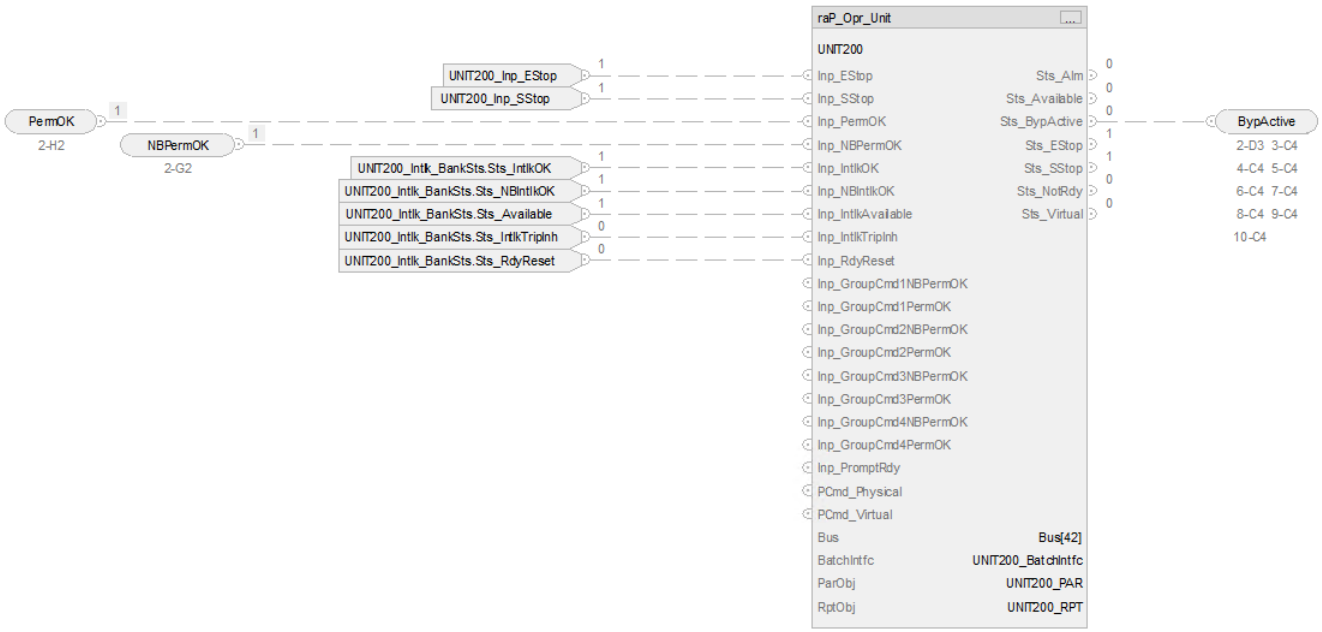


Then set the @Label property of the specified tag to the Customized Controller Data Path and import the tag with the customized extended properties into FactoryTalk Optix.

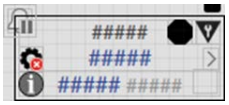


Programming Example

The example in the Functional Description section shows the basic use of the raP_Opr_Unit Add-On Instruction. Typically, the raP_Opr_Unit instruction is used in association with a Bus-resident entity. The following example shows a unit instruction that is associated with a Bus referenced entity. The unit is also connected to 2 inputs, Emergency stop, and software stop that can trigger 2 alarms. Units also allow for connections to permissives and interlocks. The unit object typically is used in an S88 application but can be applied to suit numerous hierarchy layouts. The unit allows for optional connections to parameters, reports, and FTBatch interface objects.



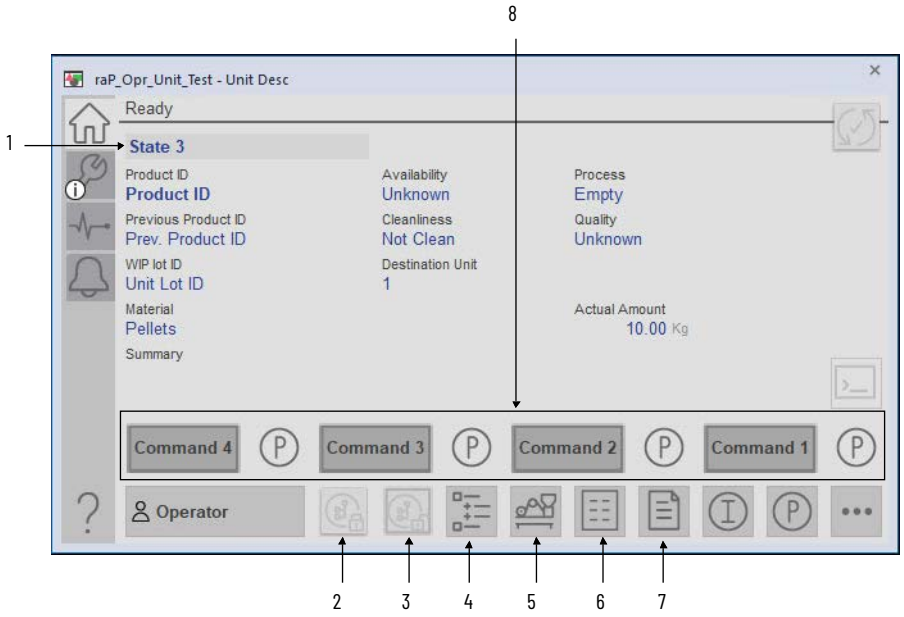
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_PUNIT 	raP_5_30_GS_raP_Opr_Unit 	The raP_Opr_Unit object groups Equipment together, and provides a propagation mechanism for aggregating status from Equipment, and broadcasting commands to Equipment.

FactoryTalk View SE Faceplates

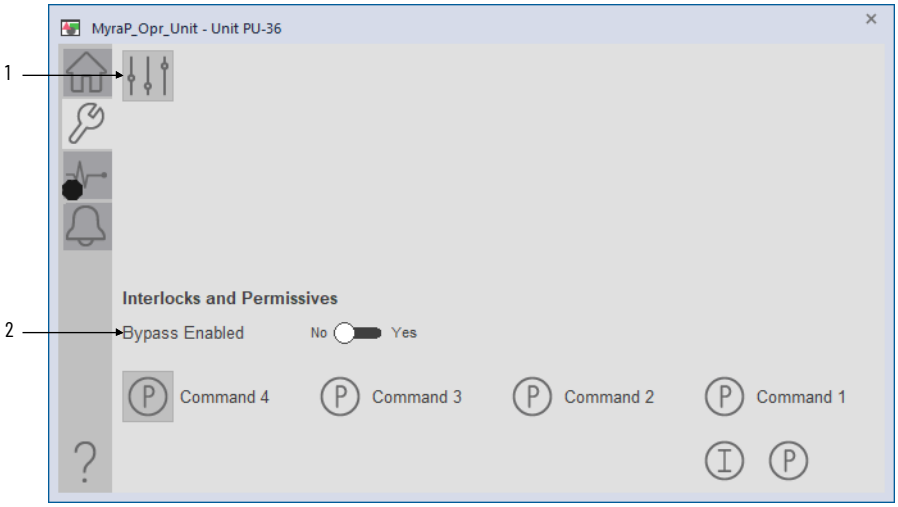
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab



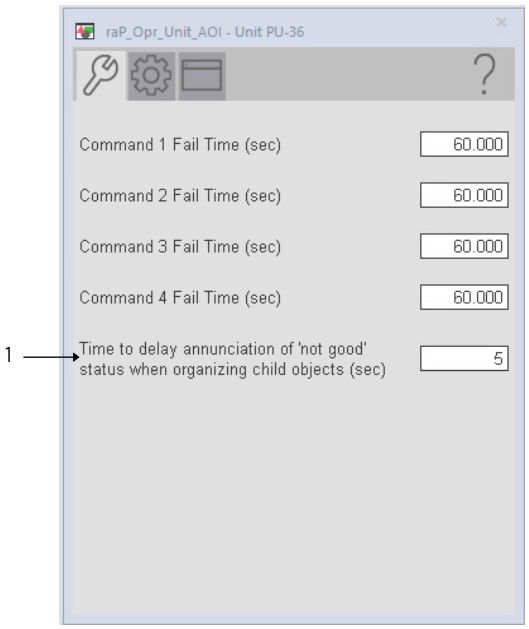
Item	Description
1	Displays the current state of the object
2	Acquire child command source
3	Release child command source
4	Display tree view for this object
5	Display the Bus faceplate for this object
6	Show parameter display
7	Show report display
8	Command user-defined function (1, 2, 3, or 4)

Maintenance Tab



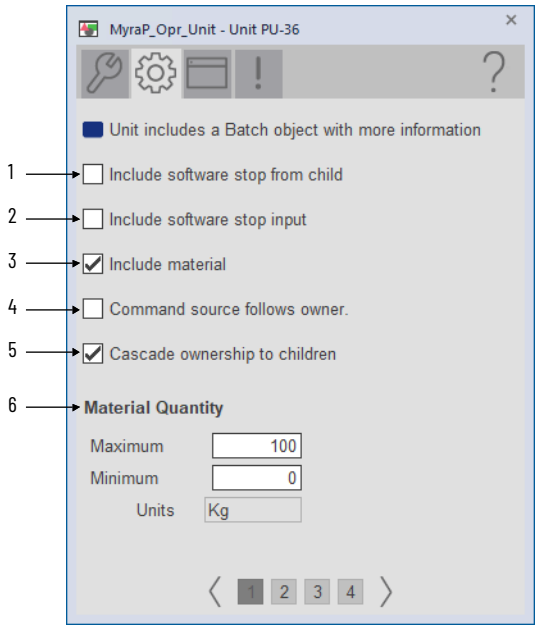
Item	Description
1	Display Advanced Properties
2	Select yes to enable bypass

Advanced Maintenance Tab

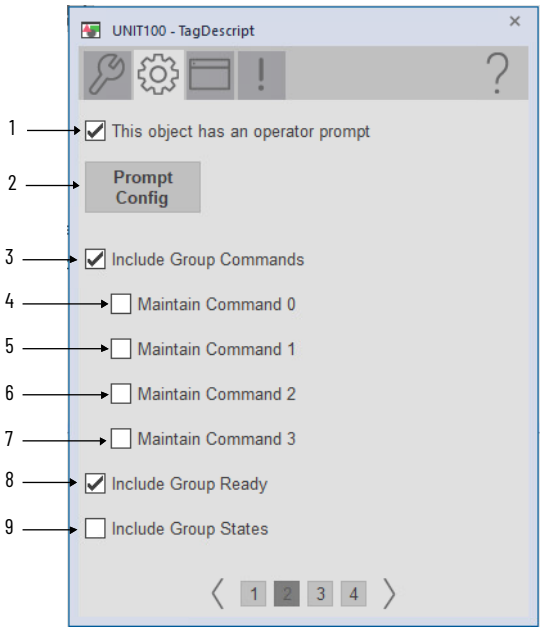


Item	Description
1	The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

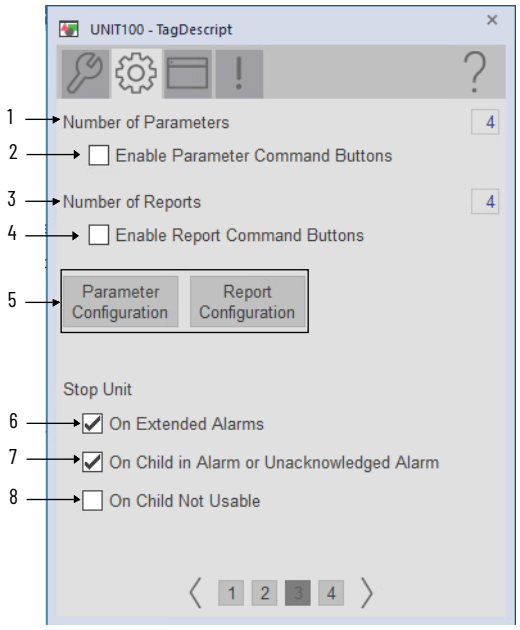
Engineering Tab



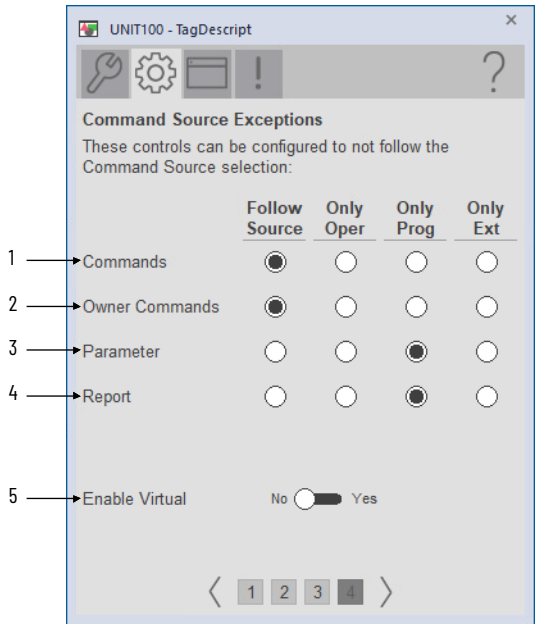
Item	Description
1	Select to include a software stop from child object
2	Select to include software stop input
3	Select to include material
4	Select to have the command source follow the owner.
5	Select to cascade ownership to children (children will be owned when this object is owned)
6	Enter the material maximum and minimum quantities as well as the units.



Item	Description
1	Select to enable an operator prompt
2	Select to open the Prompt configuration
3	Enable User-Defined Group Commands
4	Enable level command for Command 0
5	Enable level command for Command 1
6	Enable level command for Command 2
7	Enable level command for Command 3
8	Enable external ready mapping to group commands
9	Enable User-Defined Group States

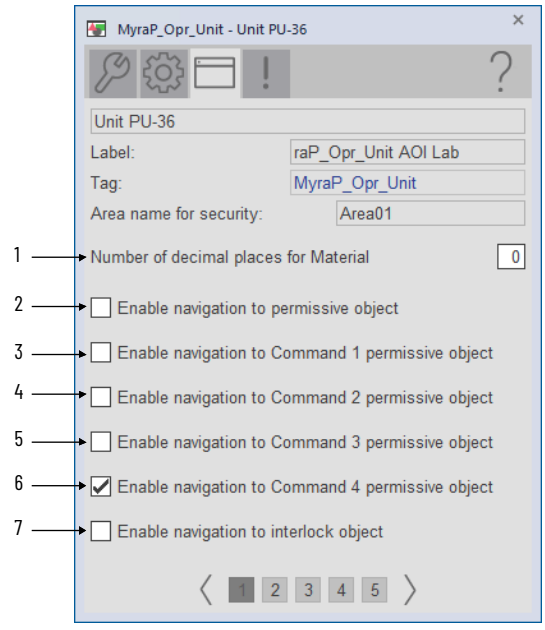


Item	Description
1	Number of Parameters configured
2	Select to enable parameter command buttons
3	Number of Reports configured
4	Select to enable report command buttons
5	Select to show parameter configuration display (left) or report configuration display (right)
6	Select to stop unit on extended alarms
7	Shed Unit actions on active child alarm, or unacknowledged child alarm
8	Shed Unit actions on child not usable, cannot be owned or in a state that makes it unusable.

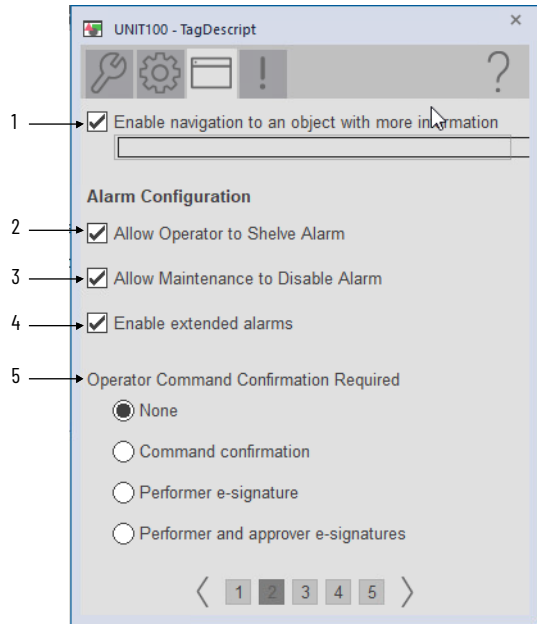


Item	Description
1	Use the radio buttons for the unit commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
2	Use the radio buttons for the unit owner commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
3	Use the radio buttons for the unit parameter commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
4	Use the radio buttons for the unit report commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
5	Select to enable virtual mode

HMI Configuration Tab



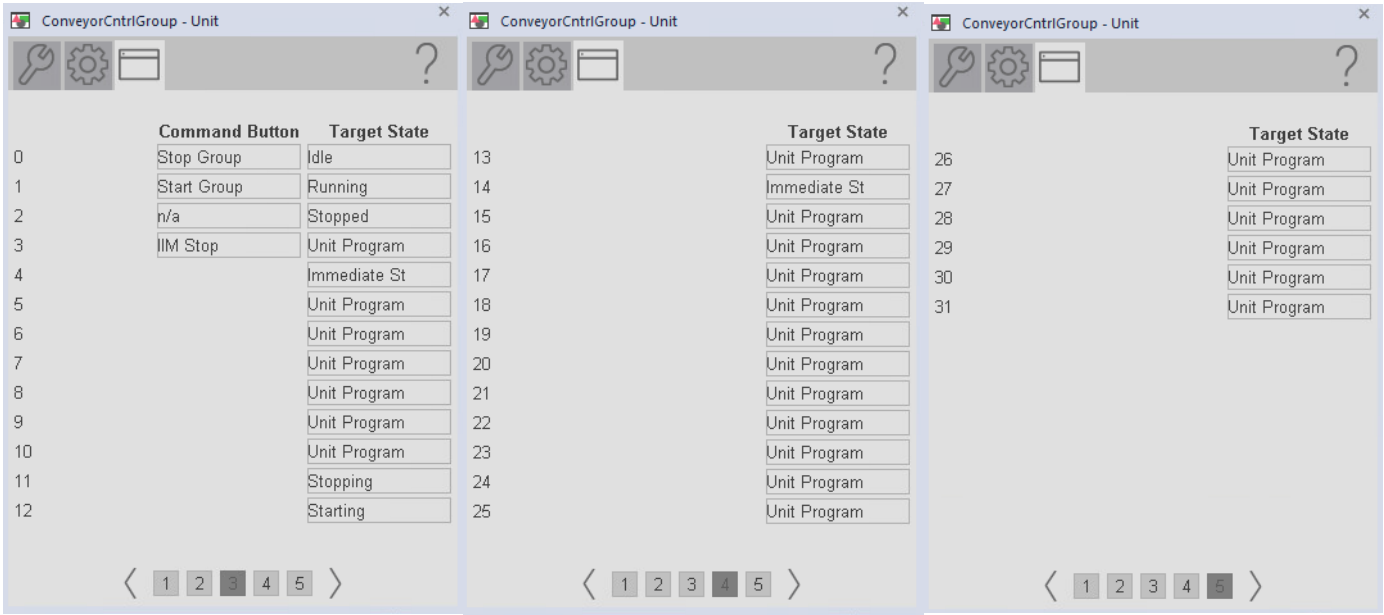
Item	Description
1	Enter the number of decimal places for the material
2	Select to enable navigation to the permissive object
3	Select to enable navigation to the Command 1 permissive object
4	Select to enable navigation to the Command 2 permissive object
5	Select to enable navigation to the Command 3 permissive object
6	Select to enable navigation to the Command 4 permissive object
7	Select to enable navigation to the interlock object



Item	Description
1	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the objects faceplate.
2	Select to allow Operator to shelve alarm
3	Select to allow Maintenance to disable the alarm
4	Select to enable extended alarms
5	Select an option for Operator Command Confirmation Requirements

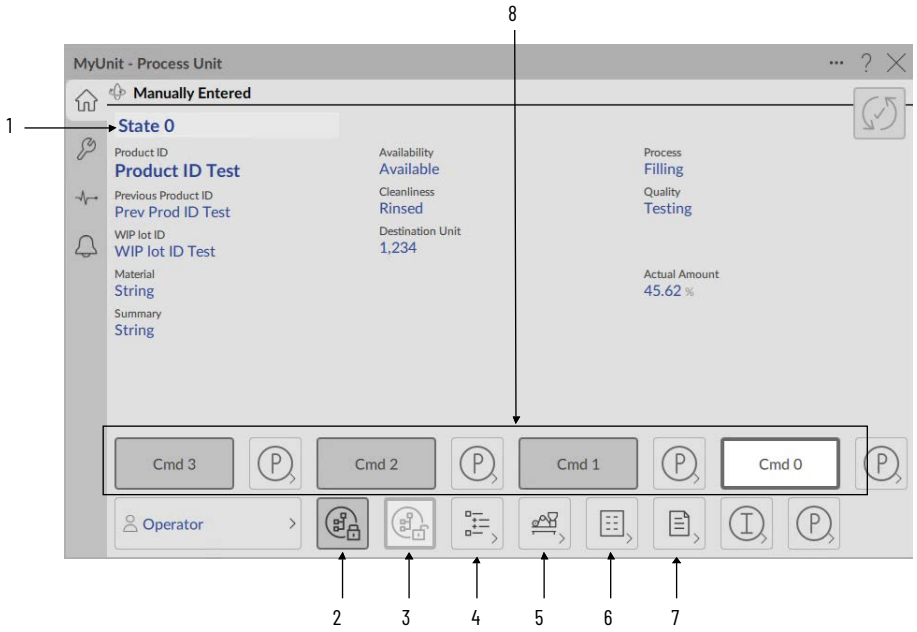
The Configuration – HMI Tab has the following purpose:

- Displays configuration of Command Buttons and Target State text (displayed on Operator Tab) for the Equipment Object.



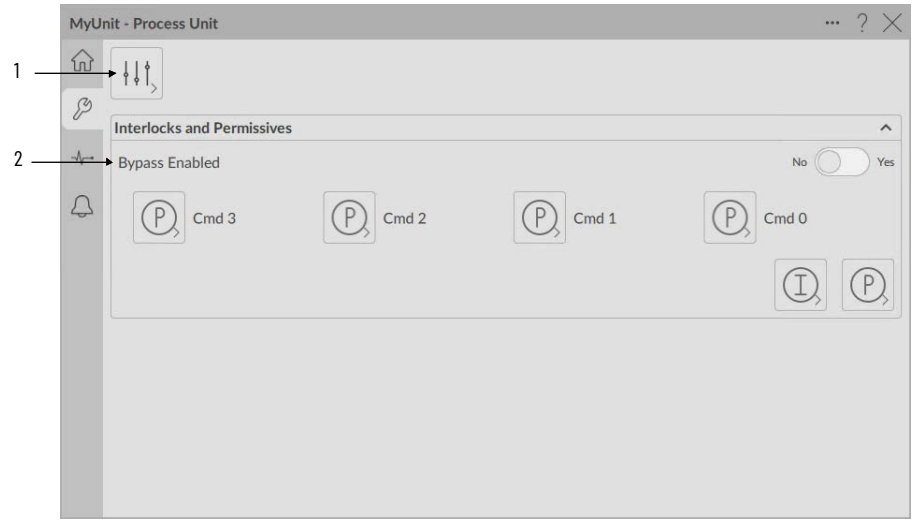
FactoryTalk Optix Faceplates

Operator Tab



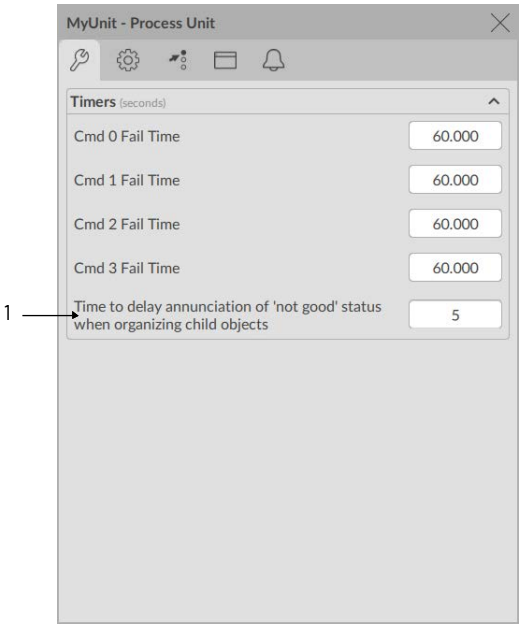
Item	Description
1	Displays the current state of the object
2	Acquire child command source
3	Release child command source
4	Display tree view for this object
5	Display the Bus faceplate for this object
6	Show parameter display
7	Show report display
8	Command user-defined function (0, 1, 2, or 3)

Maintenance Tab



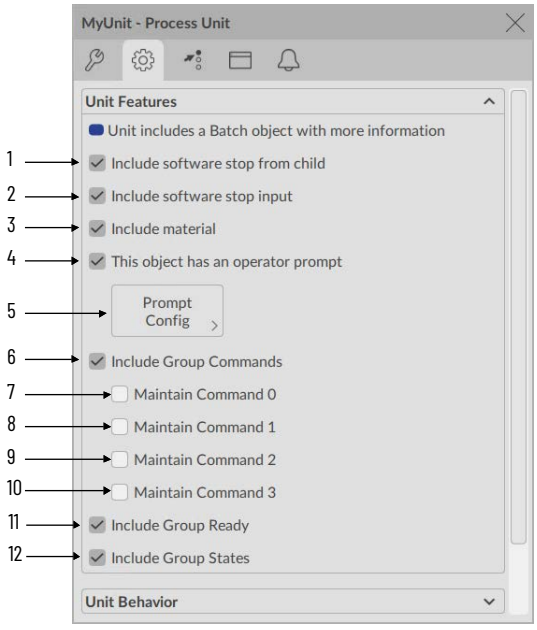
Item	Description
1	Display Advanced Properties
2	Select yes to enable bypass

Advanced Maintenance Tab



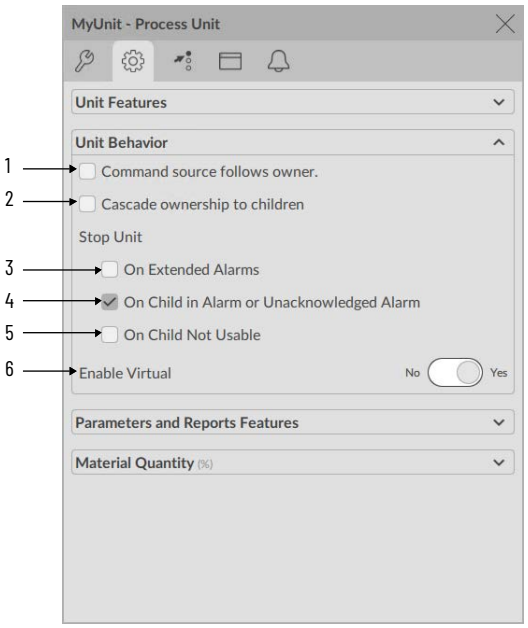
Item	Description
1	The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

Advanced Engineering Tab - Unit Features



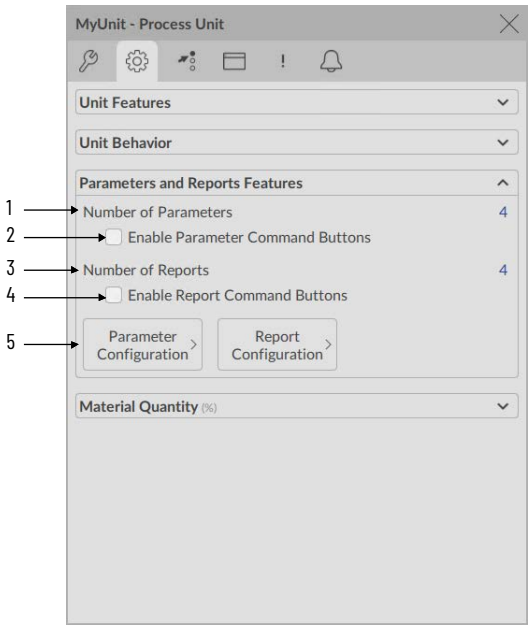
Item	Description
1	Select to include a software stop from child object
2	Select to include software stop input
3	Select to include material
4	Select to enable an operator prompt
5	Select to open the Prompt configuration
6	Enable User-Defined Group Commands
7	Enable level command for Command 0
8	Enable level command for Command 1
9	Enable level command for Command 2
10	Enable level command for Command 3
11	Enable external ready mapping to group commands
12	Enable User-Defined Group States

Advanced Engineering Tab - Unit Behavior



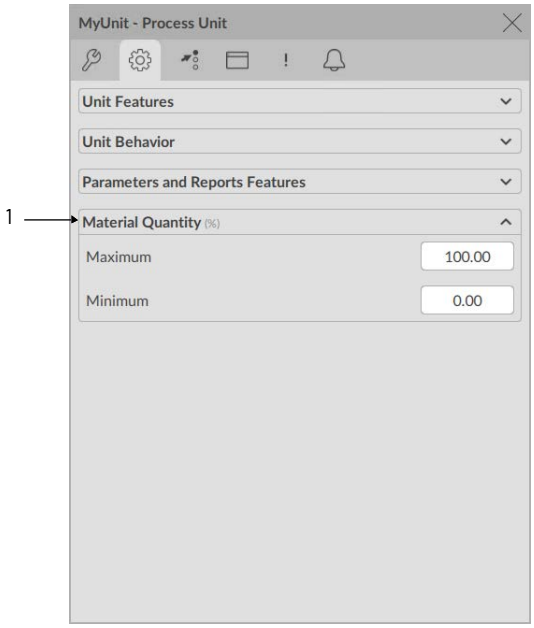
Item	Description
1	Select to have the command source follow the owner.
2	Select to cascade ownership to children (children will be owned when this object is owned)
3	Select to stop unit on extended alarms
4	Select to stop unit on active child alarm, or unacknowledged child alarm
5	Select to stop unit on child not usable, cannot be owned or in a state that makes it unusable.
6	Select to enable virtual mode

Advanced Engineering Tab - Parameters and Reports Features



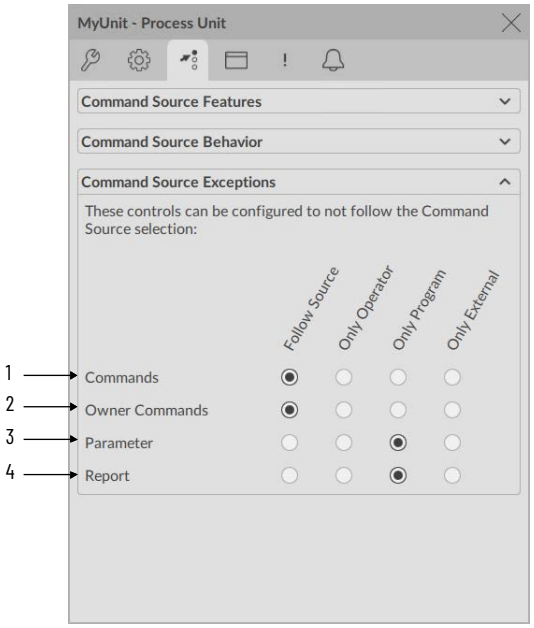
Item	Description
1	Number of Parameters configured.
2	Select to enable parameter command buttons
3	Number of Reports configured.
4	Select to enable report command buttons
5	Select to show parameter configuration display (left) or report configuration display (right)

Advanced Engineering Tab - Material Quantity



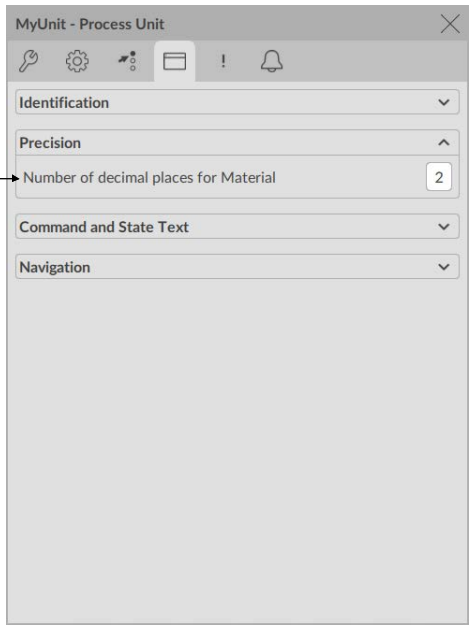
Item	Description
1	Enter the material maximum and minimum quantities.

Advanced CmdSrc Tab - Command Source Exceptions



Item	Description
1	Use the radio buttons for the unit commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
2	Use the radio buttons for the unit owner commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
3	Use the radio buttons for the unit parameter commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).
4	Use the radio buttons for the unit report commands to follow the overall command source of the instruction, or to "keep" particular source (operator, program, or external).

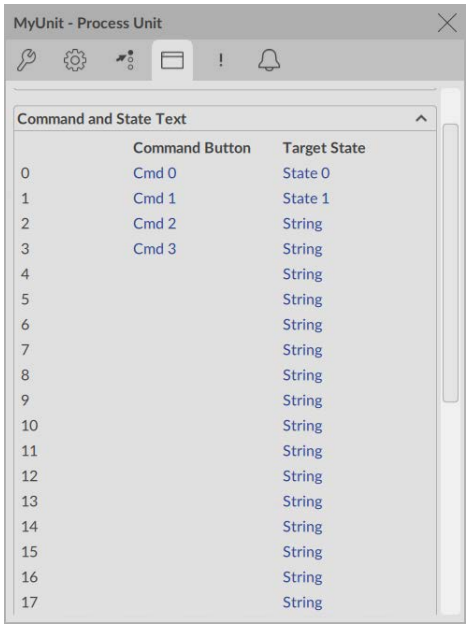
Advanced HMI Configuration Tab - Precision



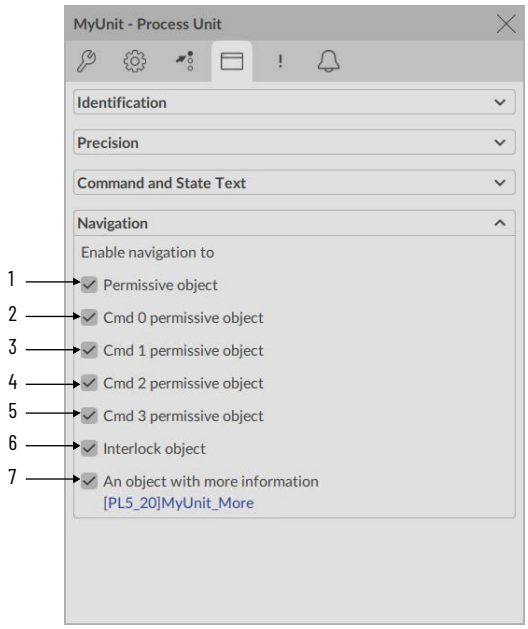
Item	Description
1	Enter the number of decimal places for the material

Advanced HMI Configuration Tab - Command and State Text

This faceplate displays configuration of Command Buttons and Target State text (displayed on Operator Tab) for the Equipment Object.



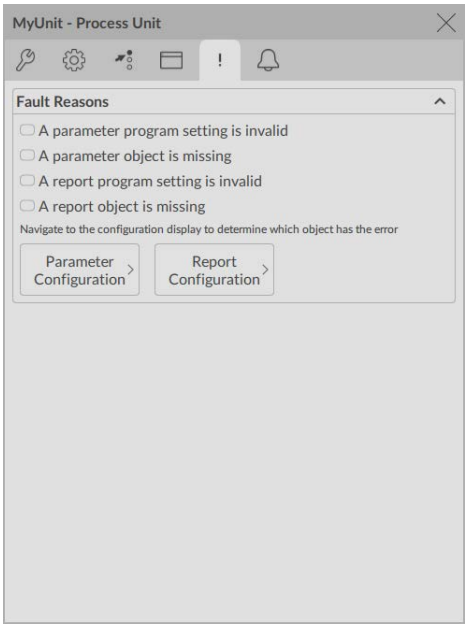
Advanced HMI Configuration Tab - Navigation



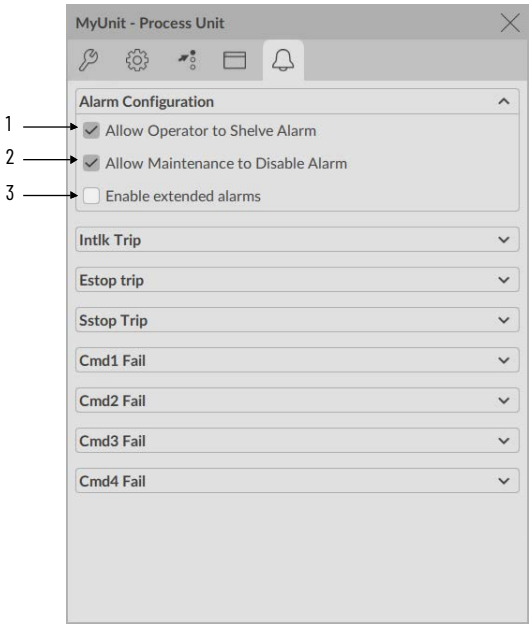
Item	Description
1	Select to enable navigation to the permissive object
2	Select to enable navigation to the Command 0 permissive object
3	Select to enable navigation to the Command 1 permissive object
4	Select to enable navigation to the Command 2 permissive object
5	Select to enable navigation to the Command 3 permissive object
6	Select to enable navigation to the interlock object
7	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the .@Library and .@Instruction extended tag properties to display the objects faceplate.

Advanced Faults Tab

The Faults tab shows information on the status of the objects. Select the Parameter and Report configuration buttons to determine which object has the fault.



Advanced Alarm Configuration



Item	Description
1	Select to allow Operator to shelve alarm
2	Select to allow Maintenance to disable alarm
3	Select to enable extended alarms

Notes:

Generic Equipment Module (raP_Opr_EMGen)

An equipment module is a functional group of equipment that can conduct a finite number of specific minor processing activities. An equipment module is typically centered around a piece of process equipment (a weigh tank, a process heater, a scrubber, and so forth). This term applies to both the physical equipment and the equipment entity.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

The raP_Opr_EMGen (Generic Equipment Module) object controls an Equipment Module in various command sources and monitors for fault conditions.

Use when:

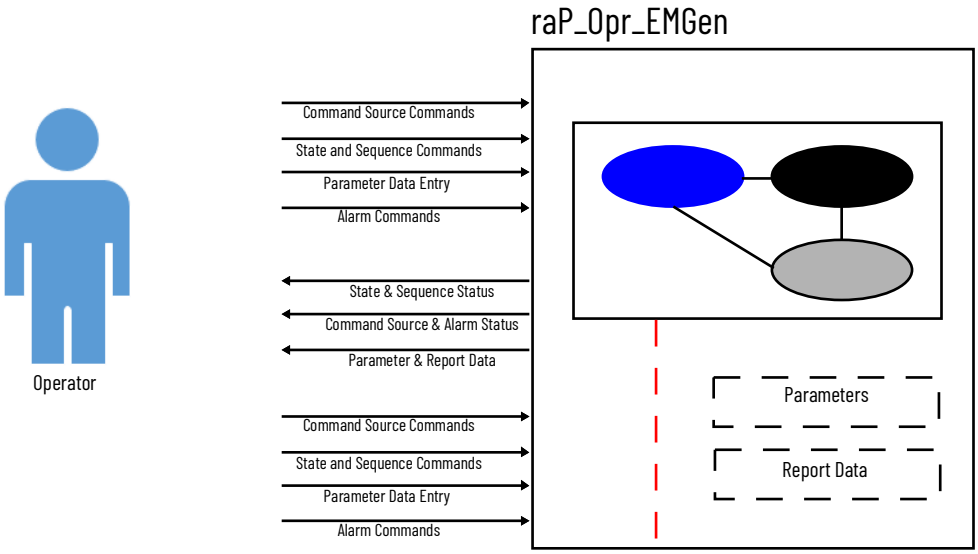
- You want to group equipment, and you want to apply a custom state model
- You want to provide the following for a group of equipment
  - Apply a mode model to the equipment group
  - Definable Commands and states
  - Apply interlocks and/or permissives to the group of equipment
  - Parameter that define the behavior of the group of equipment
  - Report / Resultant data from the group of equipment
  - A faceplate that allows monitoring / control of the equipment grouping
  - Alarm if any device fails
  - Monitor step (description), and allow forcing of steps in maintenance command source
  - Allow configurable alarms for certain process / equipment failure conditions

Functional Description

Program

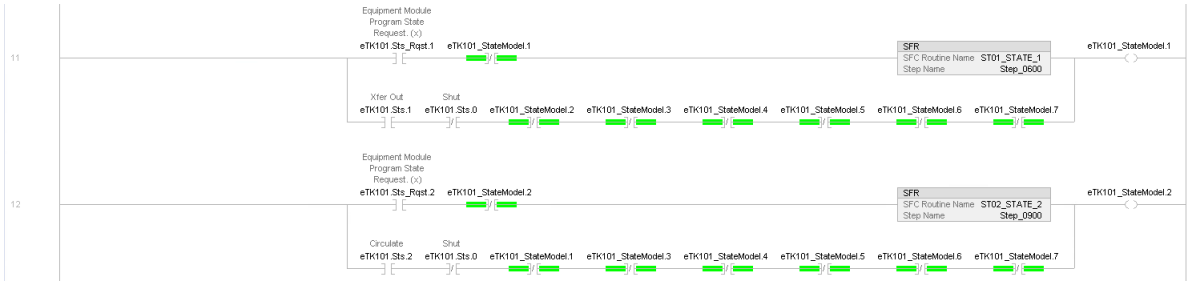
Dispatch

Contains raP_Opr_EMGen instruction and any external instructions required.



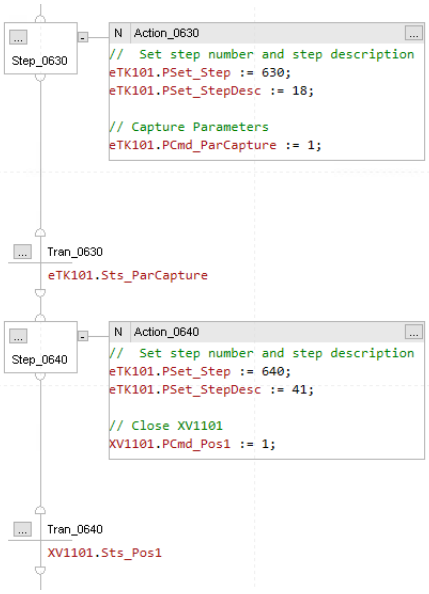
StateModel

Contains your state model (if state model is implemented external to raP_Opr_EMGen



STxx_<State> Routines

Contains your logic that sequences and coordinates devices (implement states as required)



Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller File

The raP_Opr_EMGen_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

The primary operations of raP_Opr_EMGen (Generic Equipment Module) are to:

- Provides user-defined states, and commands
- Allow monitoring of sequence Step, and display sequence Status.
- Monitor permissive conditions to help prevent Equipment Module operation.
- Monitor interlock conditions to help prevent Equipment Module operation or create failure condition.
- Provide the ability to force steps (maintenance)
- Monitor various Equipment Module failure conditions, and produce alarms.
- Operate in maintenance, program, and operator command source.
- Provide an "available" status for use by automation logic, to indicate that the Equipment Module is available for operation.
- Provide a propagation mechanism to allow the Equipment Module to publish status to and receive status from a group of equipment.
- Provides an interface to parameter display, data entry, and configuration.
- Provides an interface to resultant (report) data display and configuration.
- Allows configurable state effect of Alarm and Permissive
- Provides interface to Prompt Response and configuration

Command Sources

The raP_Opr_EMGen (Generic Equipment Module) uses the standard command source operations that are implemented using an embedded PCMDSRC instruction. See PlantPAx Process Control Instructions, publication PROCES-RM215 for more information.

State Model

The raP_Opr_EMGen (Generic Equipment Module) Add-On Instruction allows the creation of a customized state model, for a particular instance.

Depending on your requirements, you may choose to write your own State module logic and make the appropriate connections to the raP_Opr_EMGen, or you may choose to use one of the provided raP_Opr_VSM (Variable State Module) configurations (S88, NUMUR, PackML, Equipment, and Generic), or create your own using this provided Add-On Instruction. You can then make the appropriate connections to the raP_Opr_EMGen. Each instance of the raP_Opr_EMGen needs an instance of the raP_Opr_VSM Instruction.

The raP_Opr_EMGen (Generic Equipment Module) provides up to 32 state commands (PCmd) and 32 state status's (Sts), which may be used when creating a custom state model.

Program Structure

The raP_Opr_EMGen (Generic Equipment Module) may be implemented using a program as a container (recommended). The following table outlines suggested program structure and routine naming:

Routine	Description
Dispatch	Contains raP_Opr_EMGen instance, external function instances (Interlock, Permissive, Associated Device), and routine calls.
AlarmsSuppress	Contains raP_Opr_EMGen alarm suppression logic.
Interlocks	Contains raP_Opr_EMGen interlock mapping from interlock conditions to _Intlk block.
Parameters	Contains raP_Opr_EMGen parameter mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to raP_Opr_EMGen instance.
Permissives	Contains raP_Opr_EMGen permissive mapping from permissive conditions to _Perm block.
Reports	Contains raP_Opr_EMGen report mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to raP_Opr_EMGen instance.
_StateModel	Contains raP_Opr_EMGen state module program logic.
ExtddAlarms	Contains raP_Opr_EMGen instances of external alarm instances and trigger logic.
St<xx>_<StateDesc>	Contains raP_Opr_EMGen state logic.

IMPORTANT The raP_Opr_EMGen (Generic Equipment Module) may be implemented without the program structure that is defined in the preceding table; this is provided as an example.

Alarms

The raP_Opr_EMGen Instruction uses the following alarms, which are implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
Device alarms	Alm_DvcAlms	Raised when a device within the Equipment Module has an alarm.
Interlock trip	Alm_IntlkTrip	Raised when an interlock condition triggers a change in state of the Equipment Module.
Report data	Alm_RptData	Raised when new report data are available.

Virtualization

The raP_Opr_EMGen Instruction has no Virtualization capability.

Execution

Condition	Description
EnableIn False (False Rung)	Handle processing for EnableIn False (False Rung) the same as if the Equipment Module were Disabled by Command. The Equipment Module outputs are de-energized and the Equipment Module is shown as Disabled on the HMI.
Powerup (Pre-scan, First Scan)	Handles processing of command sources and alarms on Pre-scan and Powerup. On Powerup, the Equipment Module is treated as if it were Commanded to Reset all Program and Operator commands.
Postscan (SFC Transition)	No SFC Postscan logic is provided.

See Logix 5000 Controllers Add-On Instructions: Programming Manual, 1756-PM010 for more information.

ATTENTION: Disabling the raP_Opr_EMGen Add-On Instruction causes Equipment Module outputs to become de-energized.

Local Message

The object raP_Opr_EMGen utilizes local message display elements to display Step Names, Material Names, and Summary information. A default local message file is provided for each information type. This default local message file populates the local message display elements from tags in the controller. For Step Names and Material names, these are the same controller tags that are used in previous versions of the library. The difference is that 512 messages are available, rather than the 99 messages in the previous version. To upgrade from previous versions, developers must add the local message file to the project and set the @Navigation property of the specified tag to the Local Message file name (see the following table).

Information	Default Local Message File	File Name Reference	Default Controller Data
Material Name	SystemMaterialNames	Sts_eMtrl.@Navigation	System.Enum.Materials[x].@Description
Step Description	SystemStepDescriptions	Sts_eStep.@Navigation	System.Enum.Step_Desc[x].@Description
Summary Information	SystemSummary	Sts_eSummary.@Navigation	System.Enum.Summary_Desc[1].@Description

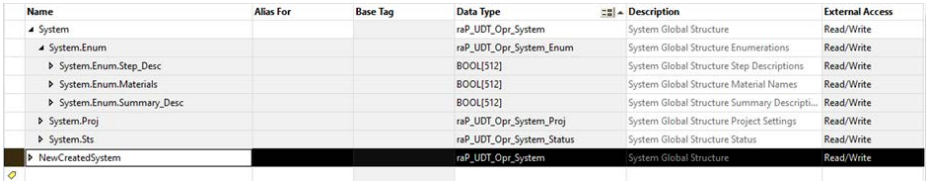
Users may add customized local messages for individual objects by creating a new local message file and populating the file with the customized strings or tag references. Then set the @Navigation property of the specified tag to the name of the new custom file.

FactoryTalk Optix Local Message

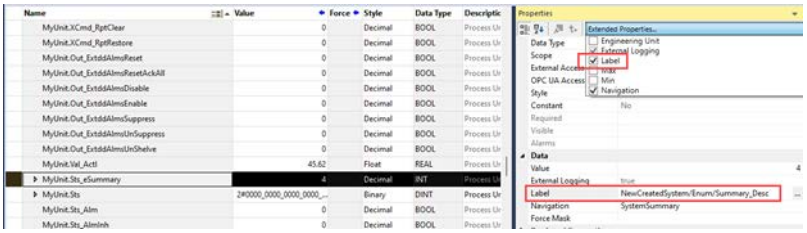
The object raP_Opr_Unit uses the @Label property of the specified tag to input the path for displaying Step Names, Material Names, and Summary information. FactoryTalk Optix does not require Local Message files. For Step Names, Material Names, and Summary, these are the same controller tags used with FactoryTalk ViewSE. FactoryTalk Optix directly retrieves Controller Data using the path specified in the @Label property of Logix Designer. Users must set the path information into @Label property of the specified tag to a Controller Data Path (see the following table).

Information	Reference	Customized Controller Data Path	Default Controller Data
Material Name	Sts_eMtrl@Label	System/Enum/Materials	System.Enum.Materials[x].@Description
Step Description	Sts_eStep@Label	System/Enum/Step_Desc	System.Enum.Step_Desc[x].@Description
Summary Information	Sts_eSummary@Label	System/Enum/Summary_Desc	System.Enum.Summary_Desc[1].@Description

Users may add Customized Controller Data for individual objects by creating a new tag member with the Data Type "raP_UDT_Opr_System" in Logix Designer.

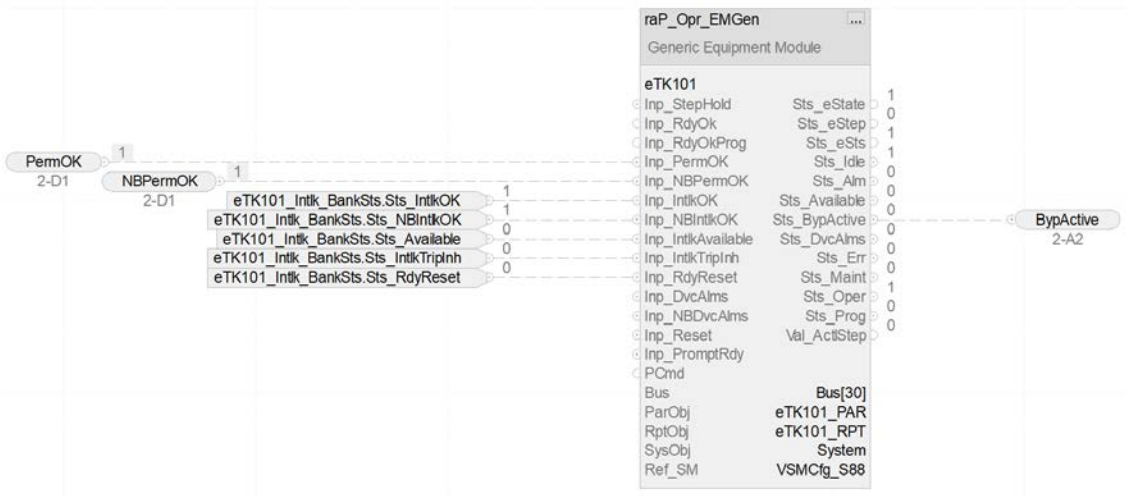


Then set the @Label property of the specified tag to the Customized Controller Data Path and import the tag with the customized extended properties into FactoryTalk Optix.

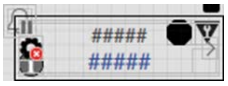


Programming Example

The example in the Functional Description section shows the basic use of the raP_Opr_EMGen Add-On Instruction. Typically, the raP_Opr_EMGen instruction is used in association with a Bus-resident entity. The following shows a generic equipment module instruction that is associated with a Bus referenced entity. The generic equipment module also allows for connections to permissives, interlocks, and a System tag. The generic equipment module typically is used in an S88 application but can be applied to suit numerous hierarchy layouts. The generic equipment module allows for optional connections to parameter and report interface objects.



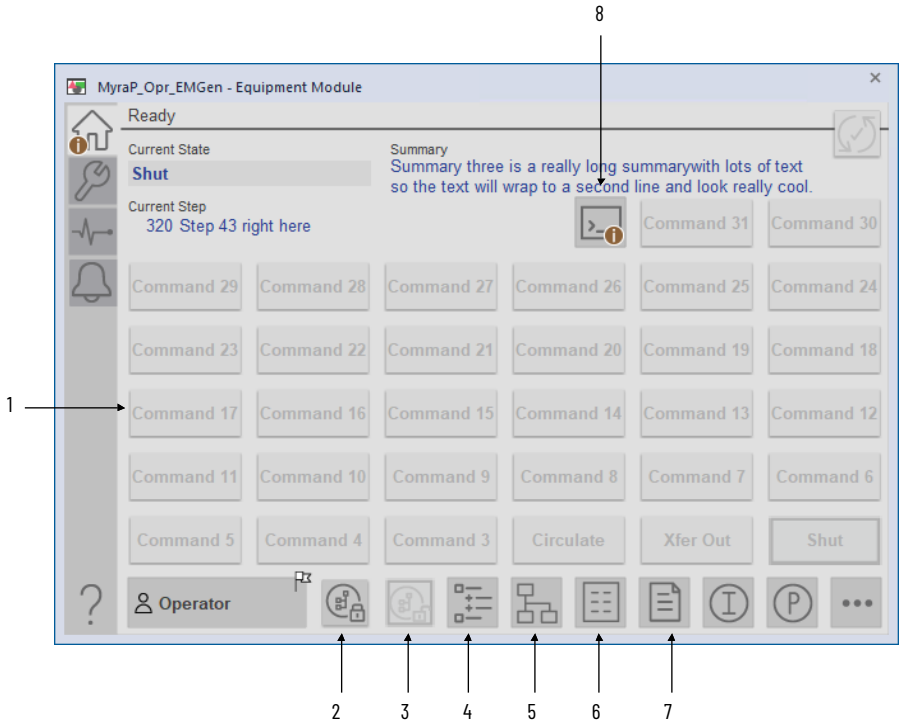
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_PEMGEN 	raP_5_30_GS_raP_Opr_EMGen 	The raP_Opr_EMGen (Generic Equipment Module) object controls an Equipment Module in a variety of command sources and monitors for fault conditions.

FactoryTalk View SE Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab



Item	Description
1	Command buttons with command text
2	Acquire child command source
3	Release child command source
4	Display tree view for this object
5	Show State Detail display. **Note**: This button is only visible if the EM has been configured with a variable state machine. The display to be launched will vary based on the extended tag property that is entered for the EM.Sts_HasVSM.@Navigation tag. The @Navigation text will be appended to the detail display name. For example: EM.Sts_HasVSM.@Navigation = S88 would launch (raP-5_30-SE) raP_Opr_EMGen-Detail_S88. EM.Sts_HasVSM.@Navigation = NAMUR would launch (raP-5_30-SE) raP_Opr_EMGen-Detail_NAMUR.
6	Show paameter display
7	Show report display
8	Show prompt response display

Pre-defined State Detail Displays

State detail displays are available to be used with pre-defined control strategies for Equip, NAMUR, PackML, and S88 state machines. There is also a generic template display that can be customized as needed.

These displays are launched from the state detail navigation button on the home tab of the raP_Opr_EMGen faceplate. The display to launch will be determined based on the extended tag property entered for the EM.Sts_HasVSM.@Navigation tag. The @Navigation text will be appended to the detail display name.

For example,

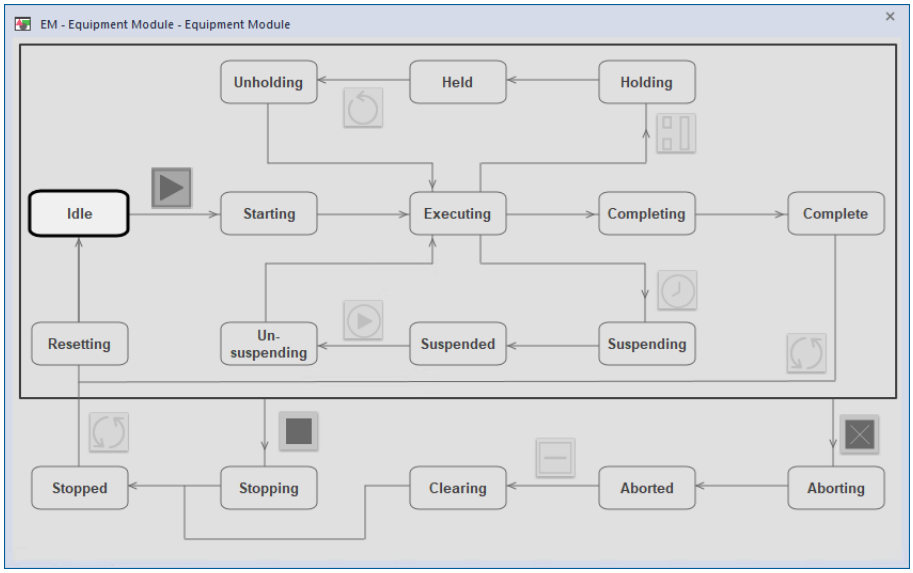
EM.Sts_HasVSM.@Navigation = S88 would launch:

(raP-5_30-SE) raP_Opr_EMGen-Detail_S88.

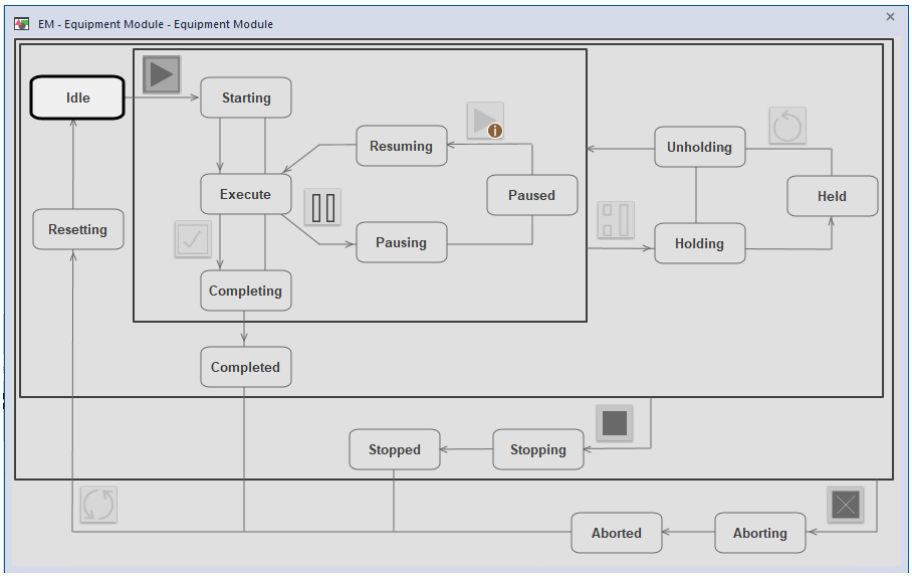
EM.Sts_HasVSM.@Navigation = NAMUR would launch:

(raP-5_30-SE) raP_Opr_EMGen-Detail_NAMUR.

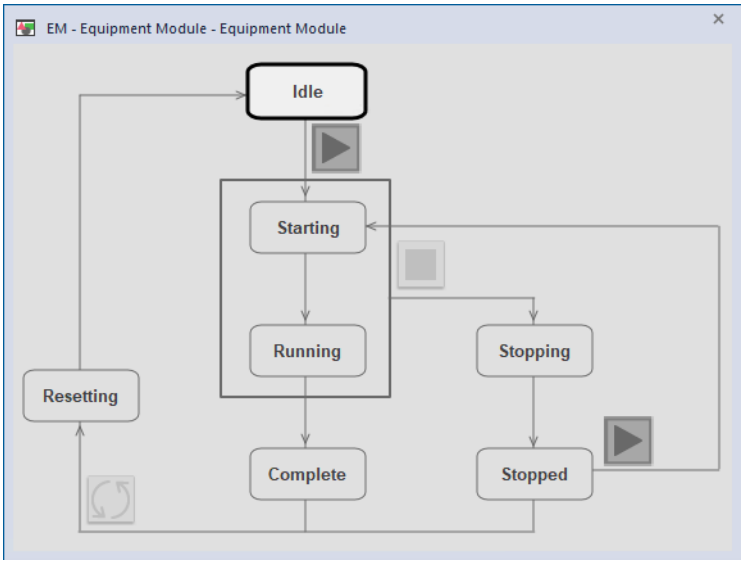
raP_Opr_EMGen-Detail_PackML



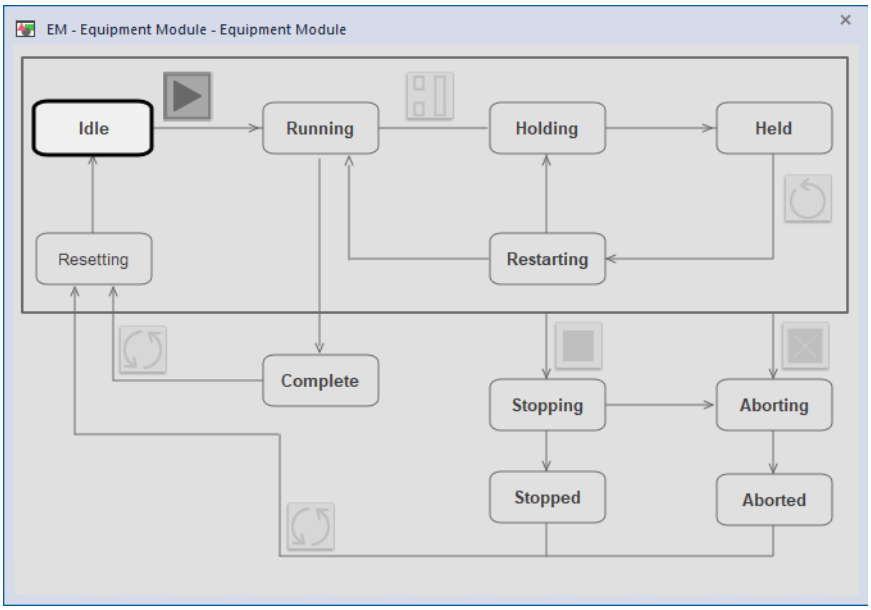
raP_Opr_EMGen-Detail_NAMUR



raP_Opr_EMGen-Detail_Equip

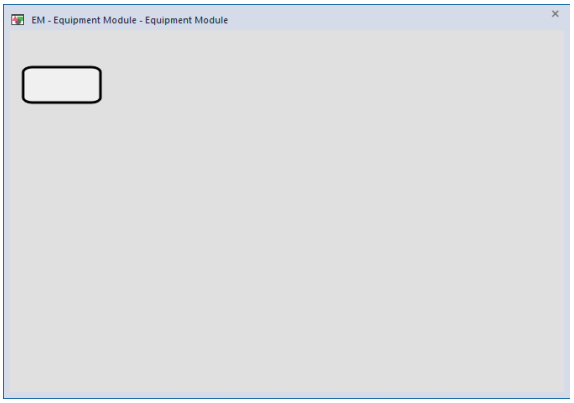


raP_Opr_EMGen-Detail_S88

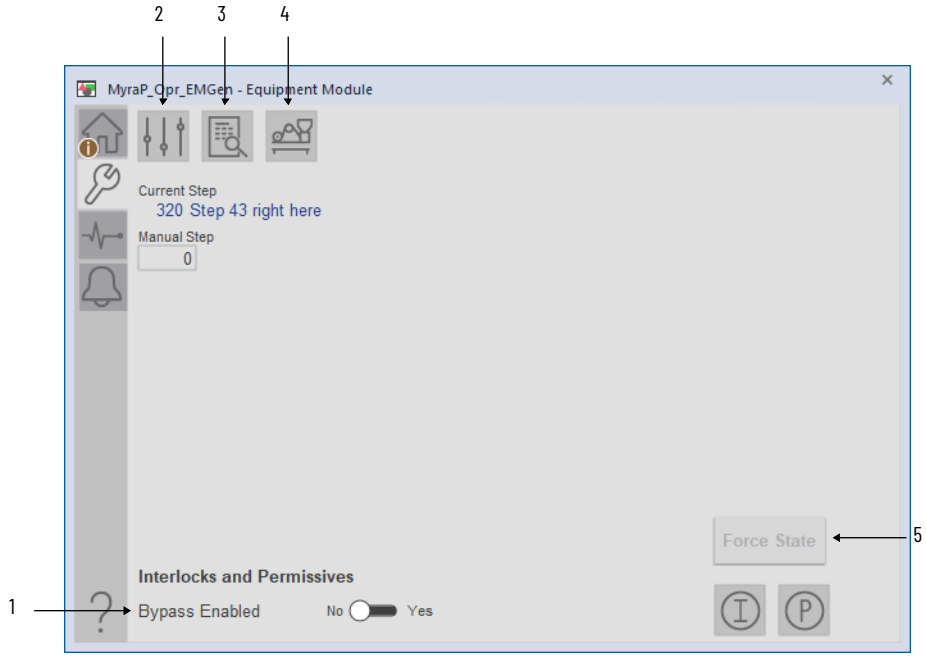


raP_Opr_EMGen-Detail_Template

This display can be customized as needed.



Maintenance Tab

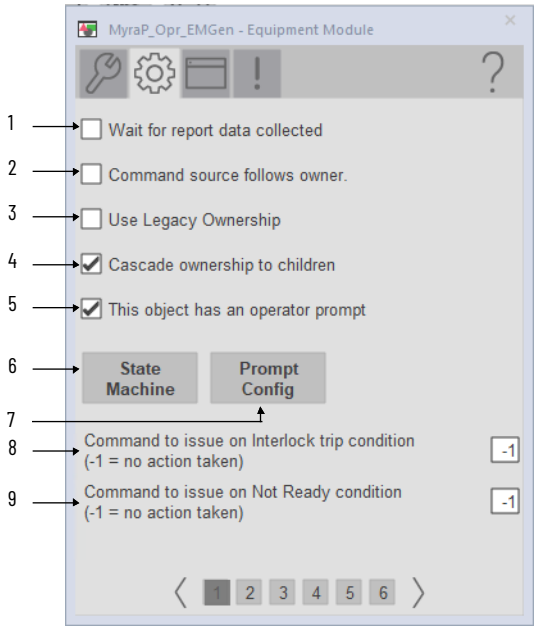


Item	Description
1	Select yes to enable bypass
2	Display advanced properties
3	Navigation to detail display
4	Display the Bus faceplate for this object.
5	The state force button is a Maintenance source command that is used to set the Sts_StateCompleteRqst bit. For an EM instance with the VSM enabled, this will set the Inp_StateComplete parameter and transition the current state to complete. For an instance without the VSM the user will need to connect the Sts_StateCompleteRqst to the required location in their SM logic.

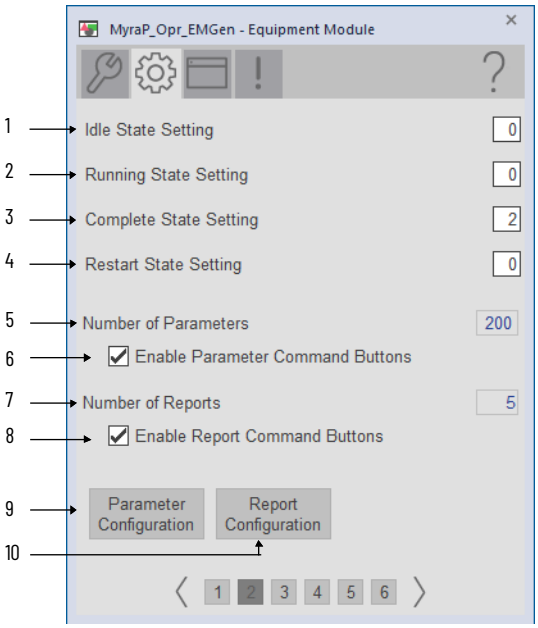
Advanced Maintenance Tab

The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

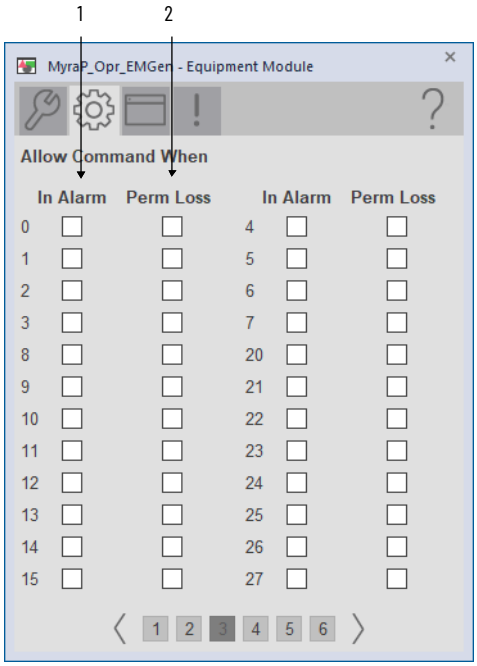
Engineering Tab



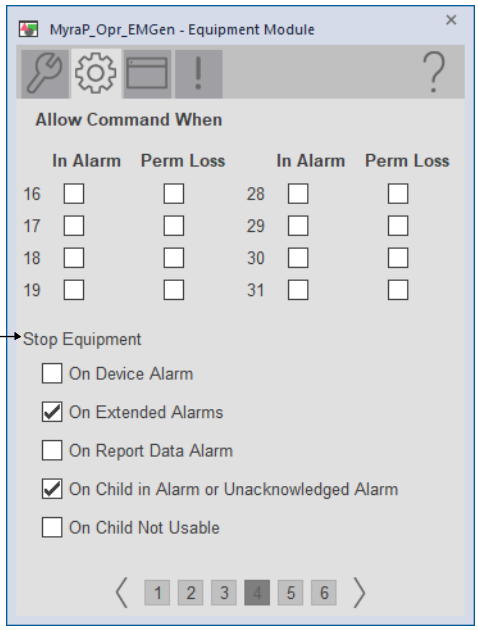
Item	Description
1	Wait for report data to be collected.
2	Command source follows parent object.
3	Enable legacy ownership, use PCmd_Owner.
4	Select to cascade ownership to children (children will be owned when this object is owned)
5	Select to enable the Operator prompt.
6	Select to navigate to the Variable State Machine (VSM) faceplate to configure the object's state machine controls.
7	Select to navigate to the Prompt faceplate to configure this object's prompts.
8	Enter the command number to issue on an interlock trip condition. If no action should be taken, enter -1.
9	Enter the command number to issue on a not ready condition. If no action should be taken, enter -1.



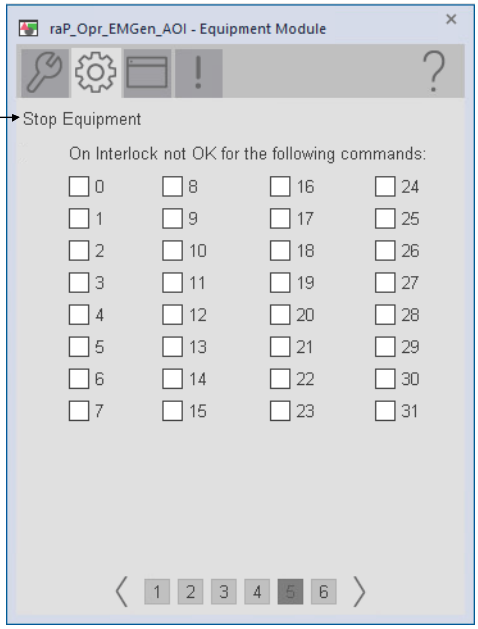
Item	Description
1	Define the Idle State for Status indication.
2	Define the Running State for Status indication.
3	Define the Complete State for Status indication.
4	Define the Restart State for Status indication.
5	Define the number of Parameters.
6	Select to enable parameter command buttons
7	Define the number of Reports.
8	Select to enable report command buttons
9	Show parameter configuration display
10	Show report configuration display



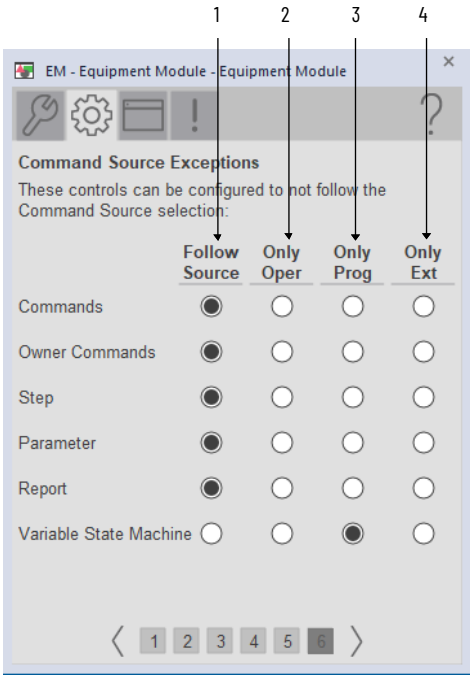
Item	Description
1	Select to allow Operator command execution with active alarm condition
2	Select to allow Operator command execution with loss of permissive



Item	Description
1	Select conditions to stop equipment

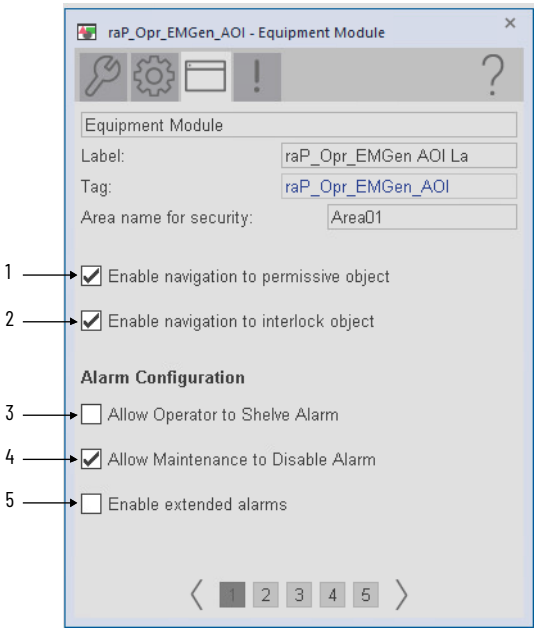


Item	Description
1	Stop equipment module on interlock trip. Bit based condition applies to only its state, Bit 0 will only affect operation of state 0, bit 31 effects state 31.

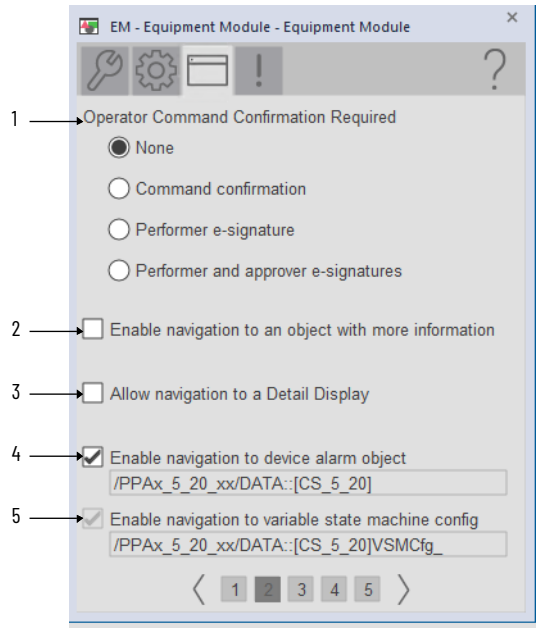


Item	Description
1	Control of this feature is determined by the current command source
2	This feature will always be commanded by the Operator
3	This feature will always be commanded by the Program Logic
4	This feature will always be commanded by the External Source

HMI Configuration Tab

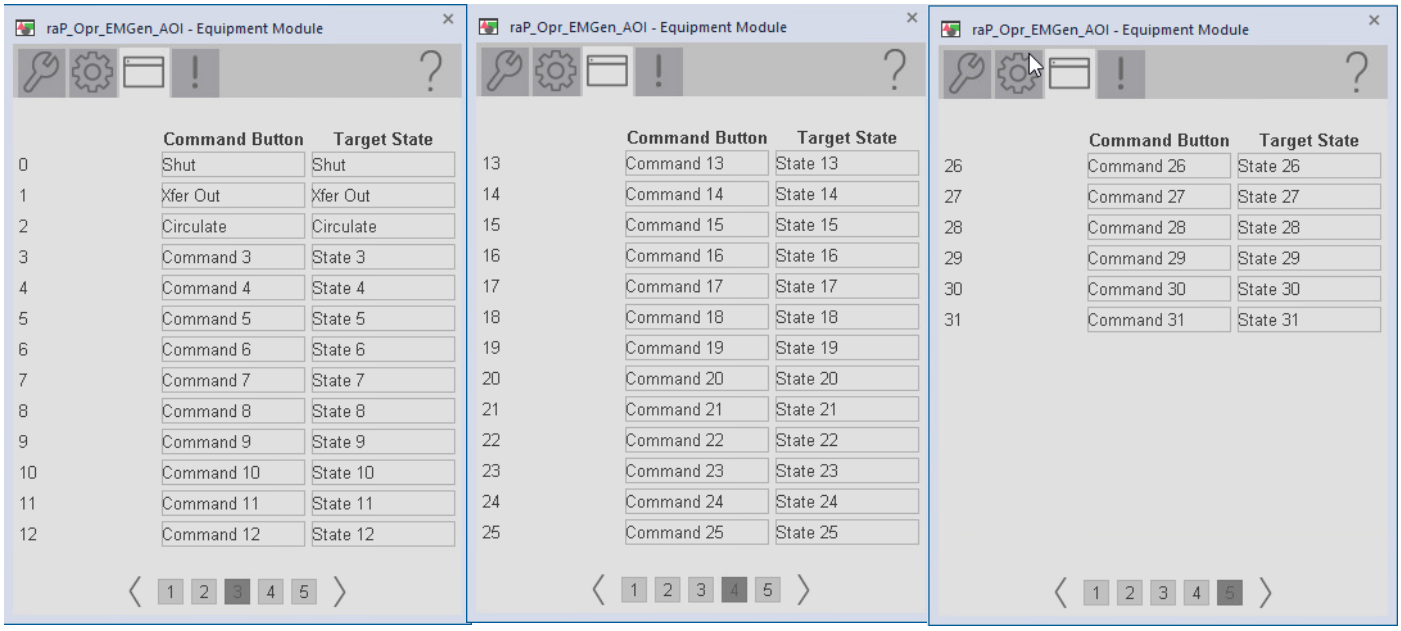


Item	Description
1	Select to enable navigation to permissive object
2	Select to enable navigation to interlock object
3	Select to allow Operator to shelve alarm
4	Select to allow Maintenance to disable alarm
5	Select to enable extended alarms



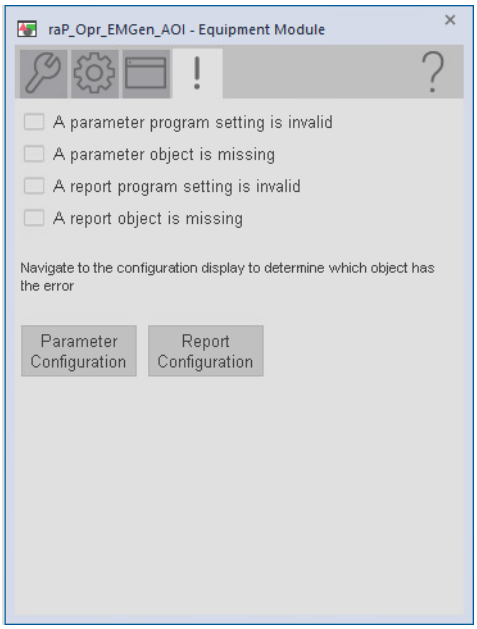
Item	Description
1	Select an option for Operator Command Confirmation Requirements
2	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the objects faceplate.
3	Select to allow navigation to detail display
4	Select to allow navigation to a device alarm object.
5	Select to allow navigation to the variable state machine configuration.

Define the Command Button and Target Stages on pages three, four, and five..



Faults Tab

The Faults tab shows information on the status of the objects. You the Parameter and Report configuration buttons to determine which object has the fault.

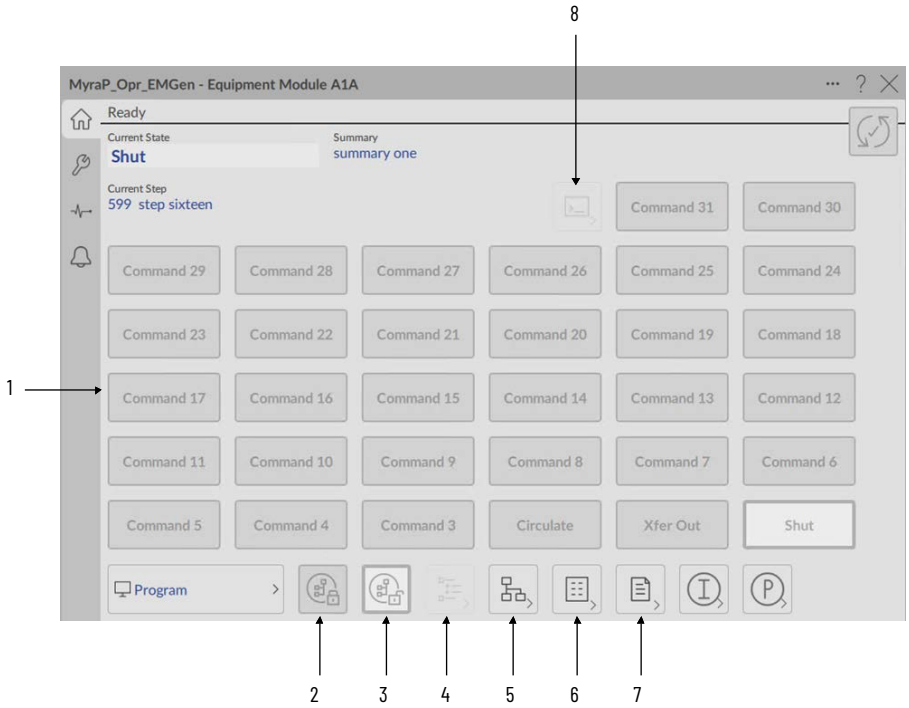


FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Any feature that is contained in the FactoryTalk Optix faceplates has the same functionality as used in the FactoryTalk View SE faceplates. See FactoryTalk View SE Faceplates on page 234.

Operator Tab



Item	Description
1	Command buttons with command text
2	Acquire child command source
3	Release child command source
4	Display tree view for this object
5	Show State Detail display. **Note**: This button is only visible if the EM has been configured with a variable state machine. The display to be launched will vary based on the extended tag property that is entered for the EM.Sts_HasVSM.@Navigation tag. The @Navigation text will be appended to the detail display name. For example: EM.Sts_HasVSM.@Navigation = S88 would launch (raP-5_30-SE) raP_Opr_EMGen-Detail_S88. EM.Sts_HasVSM.@Navigation = NAMUR would launch (raP-5_30-SE) raP_Opr_EMGen-Detail_NAMUR.
6	Show parameter display
7	Show report display
8	Show prompt response display

Pre-defined State Detail Displays

State detail displays are available to be used with pre-defined control strategies for Equip, NAMUR, PackML, and S88 state machines. There is also a generic template display that can be customized as needed.

These displays are launched from the state detail navigation button on the home tab of the raP_Opr_EMGen faceplate. The display to launch will be determined based on the extended tag property entered for the EM.Sts_HasVSM.@Navigation tag. The @Navigation text will be appended to the detail display name.

For example,

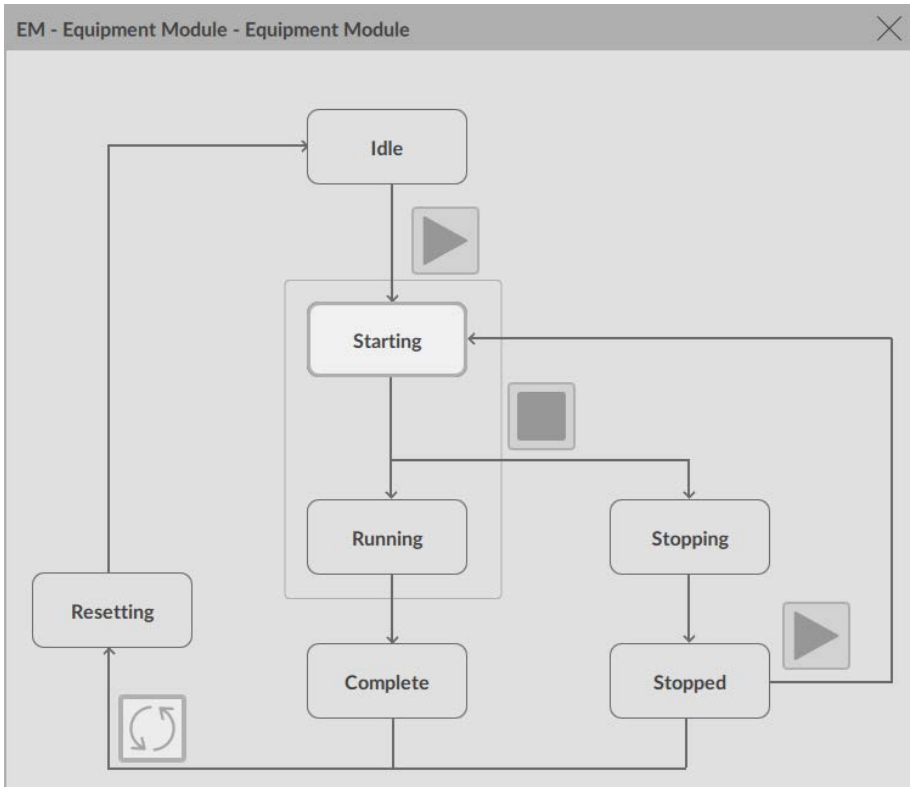
EM.Sts_HasVSM.@Navigation = S88 would launch:

(raP-5_30-SE) raP_Opr_EMGen-Detail_S88.

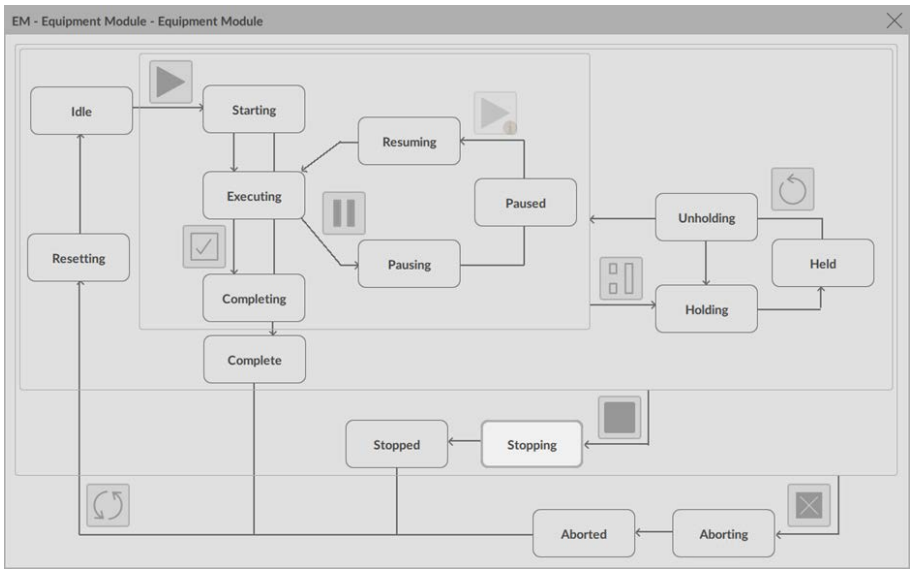
EM.Sts_HasVSM.@Navigation = NAMUR would launch:

(raP-5_30-SE) raP_Opr_EMGen-Detail_NAMUR.

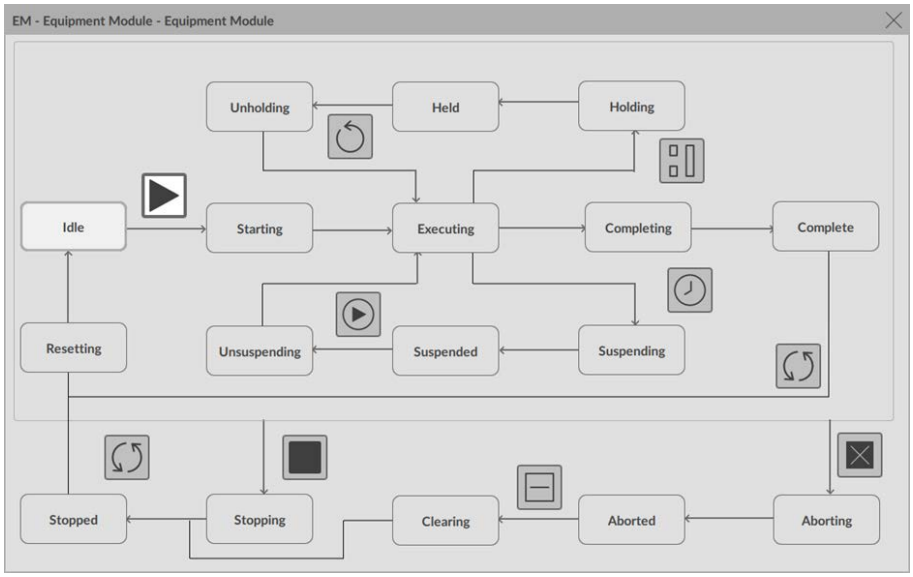
raP_Opr_EMGen-Detail_Equip



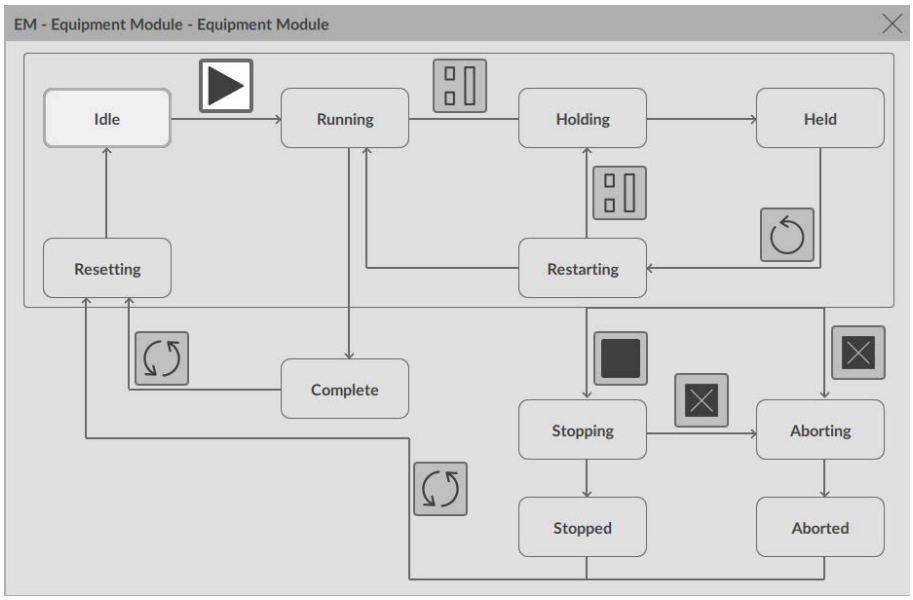
raP_Opr_EMGen-Detail_NAMUR



raP_Opr_EMGen-Detail_PackML

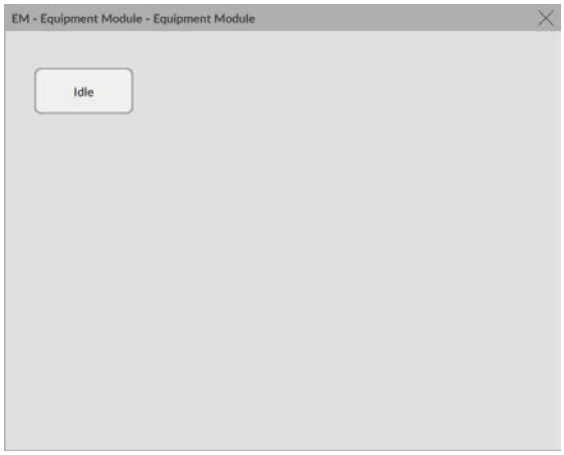


raP_Opr_EMGen-Detail_S88

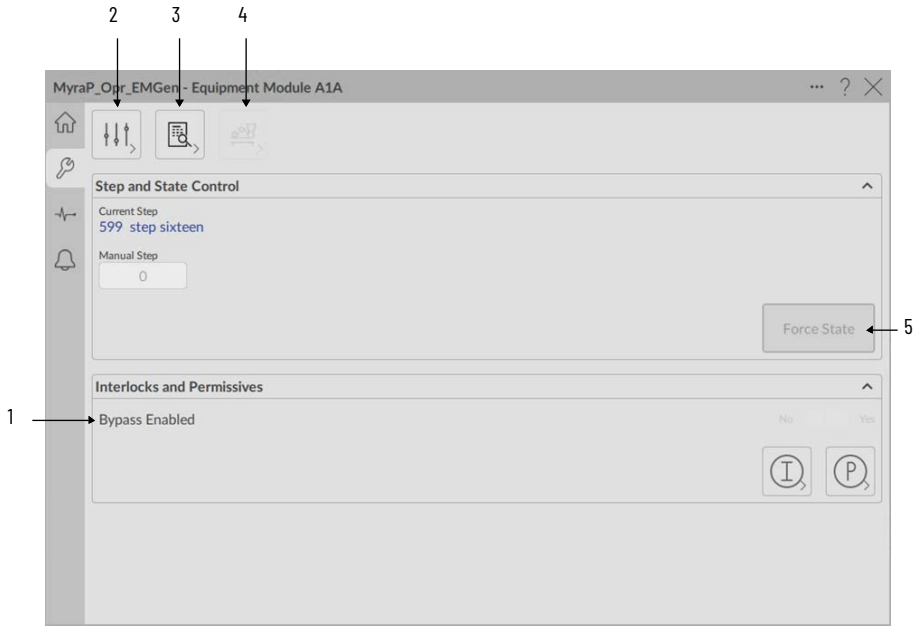


raP_Opr_EMGen-Detail_Template

This display can be customized as needed.



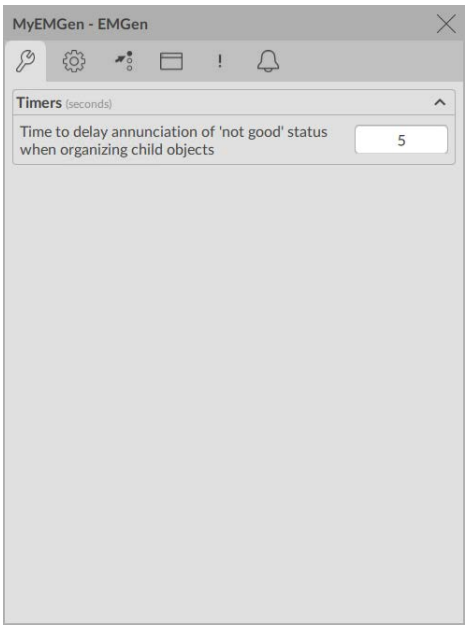
Maintenance Tab



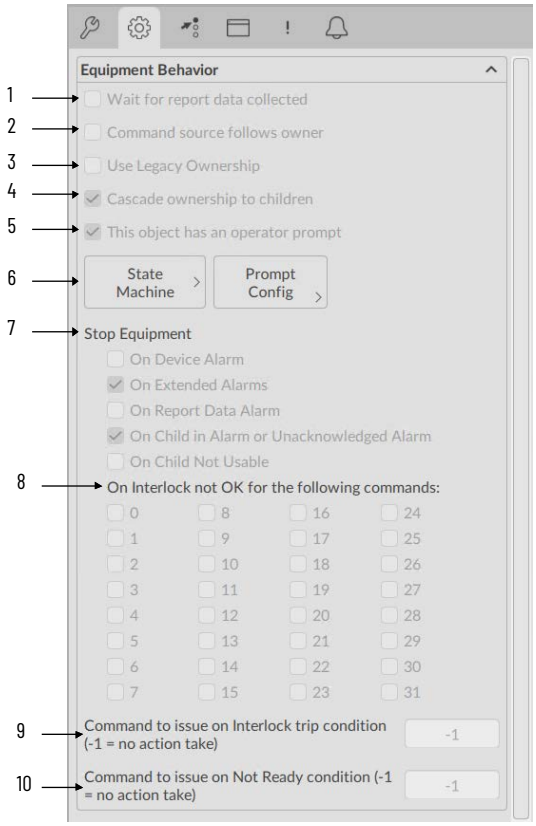
Item	Description
1	Select yes to enable bypass
2	Display advanced properties
3	Navigation to detail display
4	Display the Bus faceplate for this object (future)
5	The state force button is a Maintenance source command that is used to set the Sts_StateCompleteRqst bit. For an EM instance with the VSM enabled, this will set the Inp_StateComplete parameter and transition the current state to complete. For an instance without the VSM the user will need to connect the Sts_StateCompleteRqst to the required location in their SM logic.

Advanced Maintenance Tab

The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

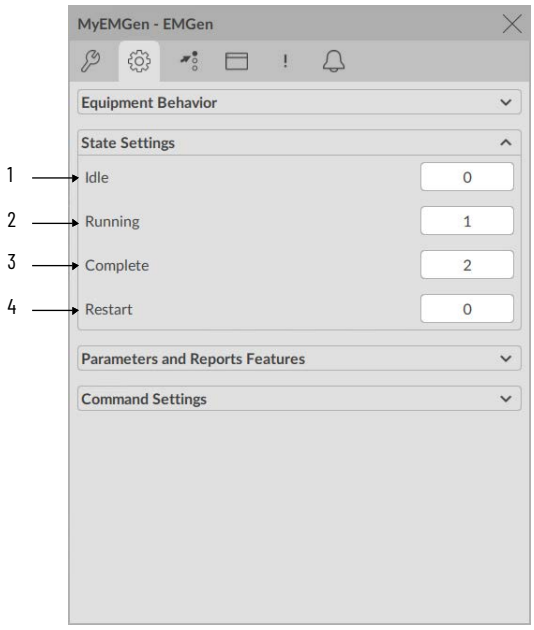


Advanced Engineering Tab - Equipment Behavior



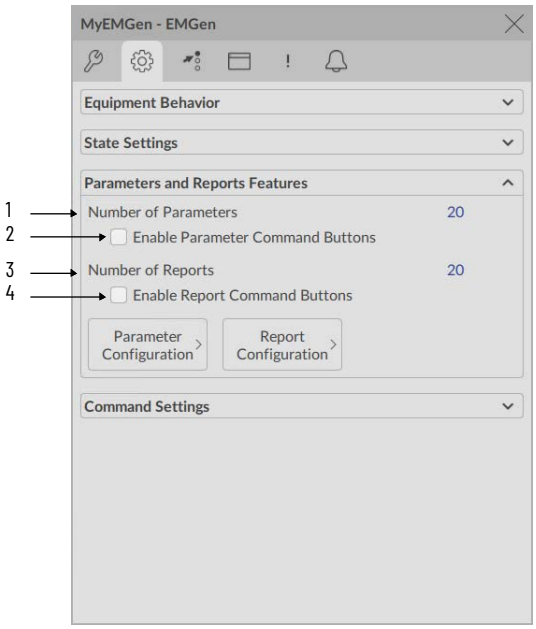
Item	Description
1	Wait for report data to be collected.
2	Command source follows parent object.
3	Enable legacy ownership, use PCmd_Owner.
4	Select to cascade ownership to children (children will be owned when this object is owned)
5	Select to enable the Operator prompt.
6	Select to navigate to the Prompt faceplate.
7	Select conditions to stop equipment
8	Stop equipment module on interlock trip. Bit based condition applies to only its state, Bit 0 will only affect operation of state 0, bit 31 effects state 31.
9	Enter the command number to issue on an interlock trip condition. If no action should be taken, enter -1.
10	Enter the command number to issue on a not ready condition. If no action should be taken, enter -1.

Advanced Engineering Tab - State Settings



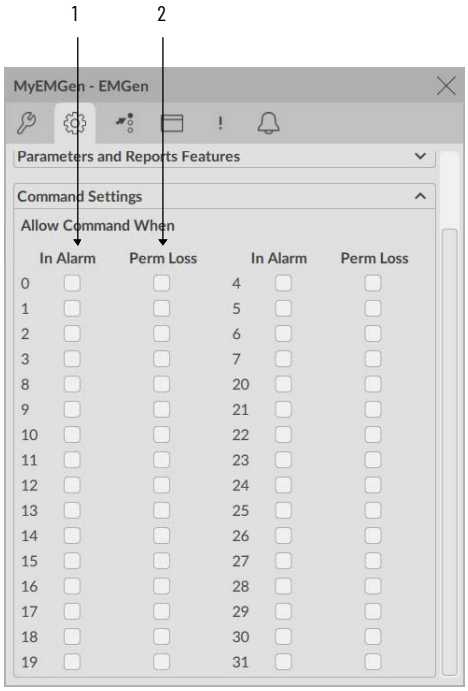
Item	Description
1	Define the Idle State for Status indication.
2	Define the Running State for Status indication
3	Define the Complete State for Status indication.
4	Define the Restart State for Status indication.

Advanced Engineering Tab - Parameters and Reports



Item	Description
1	Number of Parameters configured.
2	Select to enable parameter command buttons
3	Number of Reports configured.
4	Select to enable report command buttons

Advanced Engineering Tab - Command Settings



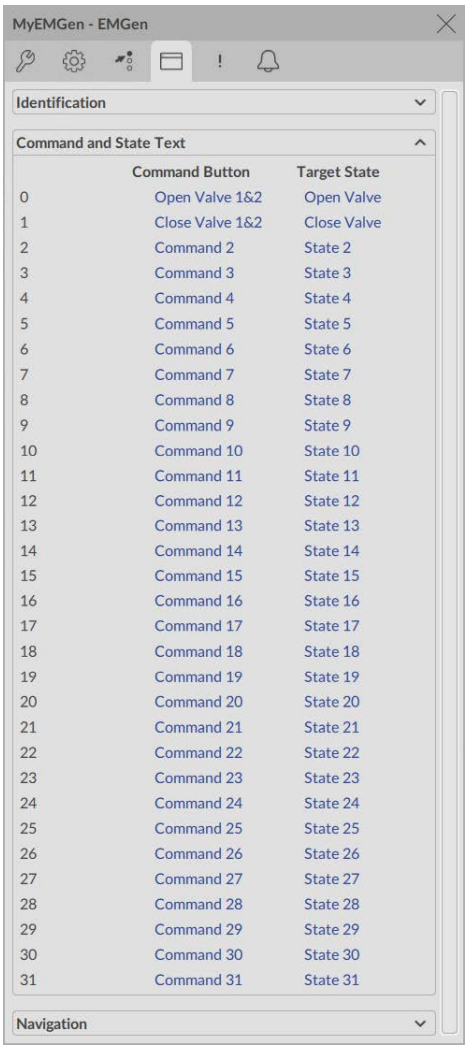
Item	Description
1	Select to allow Operator command execution with active alarm condition
2	Select to allow Operator command execution with loss of permissive

Advanced Command Source Tab - Command Source Exceptions

Item	Description
1	Control of this feature is determined by the current command source
2	This feature will always be commanded by the Operator
3	This feature will always be commanded by the Program Logic
4	This feature will always be commanded by the External Source

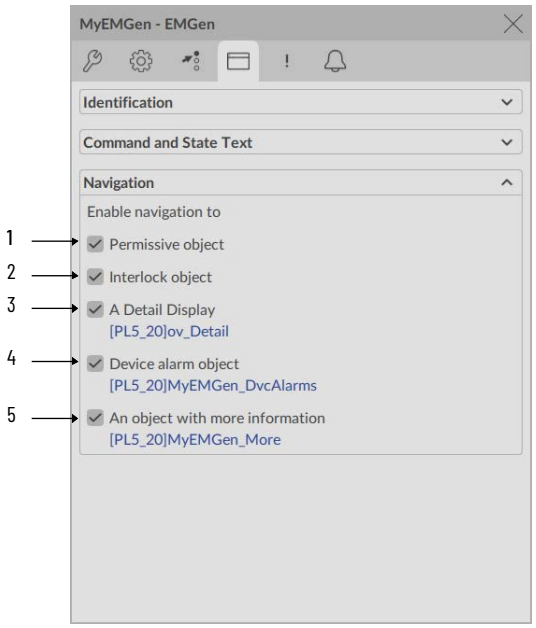
Advanced HMI Configuration Tab - Command and State Text

Display the Command Button and Target Stages.



	Command Button	Target State
0	Open Valve 1&2	Open Valve
1	Close Valve 1&2	Close Valve
2	Command 2	State 2
3	Command 3	State 3
4	Command 4	State 4
5	Command 5	State 5
6	Command 6	State 6
7	Command 7	State 7
8	Command 8	State 8
9	Command 9	State 9
10	Command 10	State 10
11	Command 11	State 11
12	Command 12	State 12
13	Command 13	State 13
14	Command 14	State 14
15	Command 15	State 15
16	Command 16	State 16
17	Command 17	State 17
18	Command 18	State 18
19	Command 19	State 19
20	Command 20	State 20
21	Command 21	State 21
22	Command 22	State 22
23	Command 23	State 23
24	Command 24	State 24
25	Command 25	State 25
26	Command 26	State 26
27	Command 27	State 27
28	Command 28	State 28
29	Command 29	State 29
30	Command 30	State 30
31	Command 31	State 31

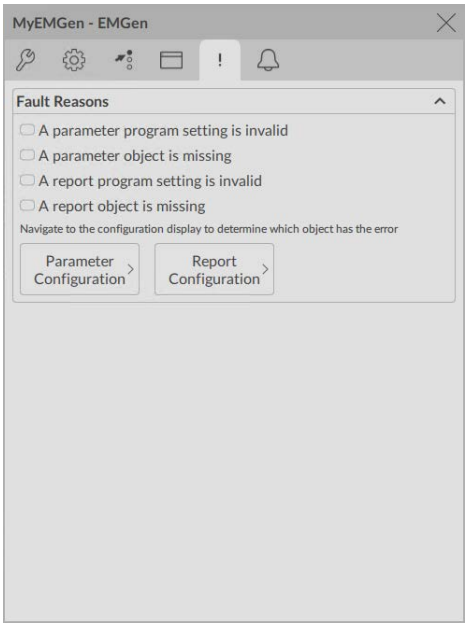
Advanced HMI Configuration Tab - Navigation



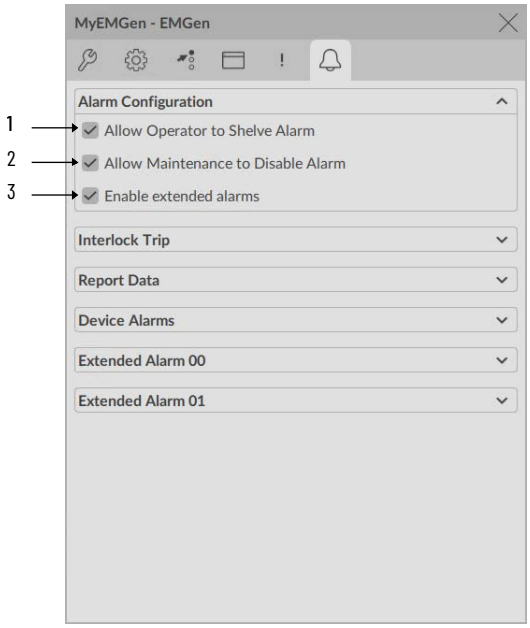
Item	Description
1	Select to enable navigation to permissive object
2	Select to enable navigation to interlock object
3	Select to allow navigation to detail display
4	Select to allow navigation to a device alarm object.
5	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the .@Library and .@Instruction extended tag properties to display the objects faceplate.

Advanced Faults Tab

The Faults tab shows information on the status of the objects. Select the Parameter and Report configuration buttons to determine which object has the fault.



Advanced Alarm Configuration



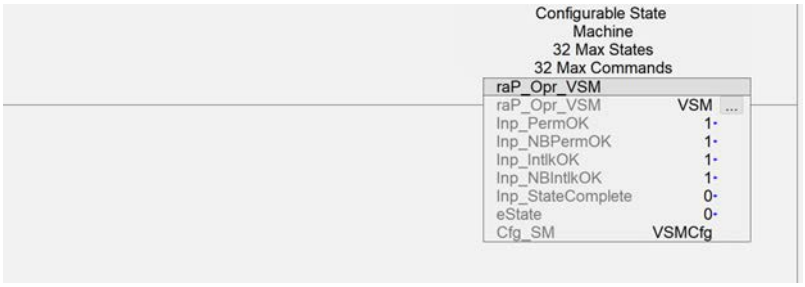
Item	Description
1	Select to allow Operator to shelve alarm
2	Select to allow Maintenance to disable alarm
3	Select to enable extended alarms

Variable State Machine (raP_Opr_VSM)

The Variable State Machine (raP_Opr_VSM) allows you to create your own state machine with up to 32 states and 32 commands.

The state machine configuration is contained in a separate data structure instance (raP_UDT_Opr_VSMCfg). This is so that multiple instances of the state machine can use the same state machine configuration. This facilitates uniformity and ease of editing and troubleshooting.

Functional Description



Permissive

The permissive OK inputs determine the ability to be commanded from the configured idle state. These inputs do not influence state done (Inp_StateDN) behavior. If the permissive is not OK, then the state machine cannot be commanded to leave the configured idle state. All valid commands are inhibited for that state and the VSM Sts_NrdyPerm=1. The Sts_CmdIPInh shows the commands that the configuration would normally allow but are currently inhibited.

Name	Type	Default	Description
Inp_PermOK	BOOL	Default=0	Input for bypassable permissive.
Inp_NBPermOK	BOOL	Default=0	Input for non-bypassable permissives.
Sts_NrdyPerm	BOOL		Status to indicate that the permissive is not OK when the state machine is in idle.
Sts_CmdIPInh	DINT		Bit mapped status that indicates which valid commands are being inhibited by either an Interlock or a permissive.

Interlock

Optionally, you can issue a command to the state machine when an interlock is not OK. When this configuration causes a command to be issued, then the Sts_IntlkTrip=1 for one scan when that command is issued. The state machine will not trip when an interlock is not OK while in the configured idle state.

Optionally, you can block commands from being issued to the state machine when an interlock is not OK. When this configuration is causing commands to be inhibited, then the Sts_NrdyIntlk=1. The Sts_CmdIPInh shows the commands that the configuration would normally allow but are currently inhibited.

Name	type	Default	Description
Inp_IntlkOK	BOOL	Default=0	Input for bypassable interlocks.
Inp_NBIntlkOK	BOOL	Default=0	Input for non-bypassable interlocks.
Cfg_IntlkAsPerm	BOOL	Default=0	Configuration to use the interlock status as a permissive and an interlock.
Cfg_CmdIntlkAsEdge	BOOL	Default=0	Configuration to treat the interlock as a leading edge triggered signal.
Set_IntlkCmd(1)	SINT	Default= -1 (no command)	Command number to issue to the state machine if the interlock is not OK.
Sts_IntlkTrip	BOOL		Status asserted if the interlock caused a valid command to be issued to the state machine as set in Set_IntlkCmd. When asserted this status is high (1) for a single scan.
Sts_NrdyIntlk	BOOL		Status to indicate that the interlock is not OK and is inhibiting valid commands.
Sts_CmdIPInh	DINT		Bit mapped status that indicates which valid commands are being inhibited by either an Interlock or a permissive.
Cfg.Cfg_InhCmdOnIntlk	DINT	Default=0	Bit mapped configuration to inhibit individual commands when an interlock is not OK.

(1) When the VSM is used within the raP_Opr_EMGen object, the Set_IntlkCmd is set by the EM.Cfg_IntlkTripState value, which is defined from the equipment module advanced faceplate.

Command Source

The raP_Opr_VSM has no internal CmdSrc instruction. An external CmdSrc can be used to communicate the CmdSrc state to the raP_Opr_VSM via the Inp_eSrc input. The raP_Opr_VSM will then update its command enables and options that are associated with the different CmdSrc states.

raP_Opr_VSM.Inp_eSrc := CmdSrc.Sts_eSrc

On startup, reset or initialization if the raP_Opr_VSM detects the absence of a CmdSrc via this input then it enables Operator and Program states internally. This enables functionality for both Operator and Program inputs as well as the associated status.

Command Source	Description
Operator/Maintenance	OCMD – DINT. One Shot Latch the bit corresponding to the command. Ordy_CmdEnable – DINT. Command is enabled when corresponding bit is on (1).
Program	PCMD – DINT. One Shot Latch the bit corresponding to the command. PSet_StateNum – SINT. Set the state number of the desired state.
External	XCMD - DINT. One Shot Latch the bit corresponding to the command. XRdy_CmdEnable - DINT. Command is enabled when corresponding bit is on (1).
Override	Inp_OvrdCmd - DINT. One Shot Latch the bit corresponding to the command. Inp_OvrdStateNum - SINT. Set the state number of the desired state.
OoS	When Out of Service (or the instruction is scanned false) the raP_Opr_VSM will continuously initialize. Not allowing any commands or state changes. The state machine is held in state zero (0).
Hand	Inp_HandStateNum - SINT. Set the state number of the desired state.

State Complete

Name	Description
Inp_StateComplete	Input for external logic to tell the state machine that the current state is complete.
Cfg_StateCompleteAsEdge	Configuration to require the Inp_StateComplete to go low before it is again recognized.

Idle State

Name	type	Default	Description
Cfg_IdleStateNum	SINT	0	Configured Idle State Number (0...31). VSM Initializes to the configured idle state. Permissive are checked while in the idle state and the interlocks will not trip while in the idle state. The return address will always be updated while in idle.

IMPORTANT When the VSM is used within the raP_Opr_EMGen object, the Cfg_IdleStateNum is assigned the same value as the EM.Cfg_IdleState configuration automatically.

Auxiliary Commands

Name	Description
Inp_CmdAux1:	1 = Assert command number assigned to Set_CmdAux1
Inp_CmdAux2	1 = Assert command number assigned to Set_CmdAux2
Cfg_CmdAux1AsEdge	1 = Inp_CmdAux1 as Edge
Cfg_CmdAux2AsEdge	1 = Inp_CmdAux2 As Edge
Set_CmdAux1(1)	Command Number to issue when Inp_AuxCmd1 = 1
Set_CmdAux2	Command Number to issue when Inp_AuxCmd2 = 1.

(1) When the VSM is used within the raP_Opr_EMGen object, the Set_CmdAux1 is set by the EM.Cfg_NrdyState value, which is defined from the equipment module advanced faceplate.

State Machine Configuration (raP_UDT_Opr_VSMCfg)

State Configuration	Type	Description
Cfg_Cmd4All	DINT	Cfg_Cmd4All.[Command Number] = 1. Commands available in all states.
Cfg_DontUpdRetSt	DINT	Cfg_DontUpdRetSt.[State Number] = 1. Don't update the return address in this state.
Cfg_InhCmdOnIntlk	DINT	Cfg_InhCmdOnIntlk.[Command Number] = 1. Don't allow this command if the interlock is not OK.
Cfg_CmdMask	DINT[32]	Cfg_CmdMask[State Number] = 1. Allows that command in that state.
Cfg_CmdTarget	SINT[32]	Cfg_CmdTarget[Command Number] = Target State Number. Defines the destination state of a command.
Cfg_SCTarget	SINT[32]	Cfg_SCTarget[State Number] = Target State when Inp_StateComplete = 1.

Cfg_Cmd4All

Each bit of this DINT corresponds to a command in the state machine. When the corresponding bit is set, this command is available to all CmdSrc owners simultaneously when the command is enabled. This can be useful if a 'Stop' command should always be available to any CmdSrc owner when enabled.

Example:

Cfg_Cmd4All = 2# 0000 0000 0000 0000 0000 1010 0000 0010

Commands 1, 9, 11 are available to be asserted by Oper, Prog, External, and Maintenance command inputs when those commands are enabled.

Cfg_DontUpdRetSt

Each bit of this DINT corresponds to a state in the state machine. Internally to the state machine there is a register that maintains the number of the state as it proceeds through the states. An option for command targets is to use the state number stored in this register. When the corresponding bit is set this state number is not updated with the corresponding state when it is current. This can be useful if multiple states can receive a 'Hold' command. But upon 'Restart' it is desirable to return to the state from which it came instead of the last state encountered or a single specified state.

Example:

Cfg_DontUpdRetSt = 2# 0000 0000 0000 0000 0000 0000 1101 0000

States 4, 6, 7 will not update this register as it passes through them. If in one of these states and a command to transition to the state held in the register, then the state machine transitions to the last recorded state (not one of these). If a state machine has 'Starting' and 'Running' states but both can be interrupted by a 'Hold' command then it may be desirable to return to whichever state was interrupted. In this case, the corresponding bits for the 'Held' And 'Restarting' states would be set. Leaving the state number for either 'Starting' or 'Holding' in the register. Then the complete target for the 'Restarting' state would be the return state in the register.

Cfg_InhCmdOnIntlk

Each bit of this DINT corresponds to a command in the state machine. When the corresponding bit is set, this command is inhibited when normally it would be enabled by the configuration but the interlock is not OK. This is useful to inhibit 'Start' or 'Restart' commands if the interlock is not OK.

Example:

Cfg_InhCmdOnIntlk = 2# 0000 0000 0000 0000 0100 0001 0000 0001

Commands 0, 8, 14 are inhibited if the interlock is not OK.

Cfg_CmdMask

Each DINT element corresponds to a state in the state machine. Each bit of that element corresponds to a command in the state machine.

Example:

Cfg_CmdMask[3] = 2# 0000 0000 0000 0000 0000 0010 0011 1000

In state 3 commands 3, 4, 5, 9 are enabled.

Cfg_CmdTarget

Each SINT element corresponds to a command in the state machine. The numerical value of that element is the number of the state that is the target when this command is received.

Example:

Cfg_CmdTarget[2] = 5

When command 2 is received the target state is state number 5.

Cfg_SCTarget

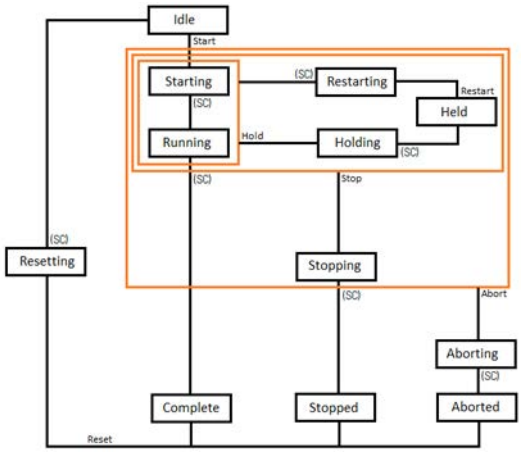
Each SINT element corresponds to a state in the state machine. The numerical value of that element is the number of the state that is the target when this state is complete.

Example:

Cfg_SCTarget[7] = 9

When state 7 is declared complete (by Inp_StateComplete=1) then the target state is state number 9.

Configuration Example



States	State Complete Target	Start(1)	Reset(1)	Restart(1)	Hold(1)	Stop(1)	Abort(1)
0 - Idle	0 Idle	X					
1 - Starting	2 Running				X	X	X
2 - Running	3 Complete				X	X	X
3 - Complete	3 Complete		X				
4 - Holding	5 Held					X	X
5 - Held	5 Held			X		X	X
6 - Restarting	-1 (last state)				X	X	X
7 - Stopping	8 Stopped						X
8 - Stopped	8 Stopped		X				
9 - Aborting	10 Aborted						
10 - Aborted	10 Aborted		X				
11 - Resetting	0 Idle						

(1) Command accepted in state.

IMPORTANT Commands of higher precedence are assigned a higher bit number than commands of lower precedence. When multiple simultaneous commands are received the command with the highest precedence Is the command that is executed.

Allowable Command Mask	State: Commands
Cfg_CmdMask[0] = 2# 0000 0000 0000 0000 0000 0000 0000 0001	Idle: Start
Cfg_CmdMask[1] = 2# 0000 0000 0000 0000 0000 0000 0011 1000	Starting: Hold, Stop, Abort
Cfg_CmdMask[2] = 2# 0000 0000 0000 0000 0000 0000 0011 1000	Running: Hold, Stop, Abort
Cfg_CmdMask[3] = 2# 0000 0000 0000 0000 0000 0000 0000 0010	Complete: Reset
Cfg_CmdMask[4] = 2# 0000 0000 0000 0000 0000 0000 0011 0000	Holding: Stop, Abort
Cfg_CmdMask[5] = 2# 0000 0000 0000 0000 0000 0000 0011 0100	Held: Restart, Stop, Abort
Cfg_CmdMask[6] = 2# 0000 0000 0000 0000 0000 0000 0011 1000	Restarting: Hold, Stop, Abort
Cfg_CmdMask[7] = 2# 0000 0000 0000 0000 0000 0000 0010 0000	Stopping: Abort
Cfg_CmdMask[8] = 2# 0000 0000 0000 0000 0000 0000 0000 0010	Stopped: Reset
Cfg_CmdMask[9] = 2# 0000 0000 0000 0000 0000 0000 0000 0000	Aborting: *None
Cfg_CmdMask[10] = 2# 0000 0000 0000 0000 0000 0000 0000 0010	Aborted: Reset
Cfg_CmdMask[11] = 2# 0000 0000 0000 0000 0000 0000 0000 0000	Resetting: *None

State Target of Command	Command: State
Cfg_CmdTarget[0] = 1	Start: Starting
Cfg_CmdTarget[1] = 11	Reset: Resetting
Cfg_CmdTarget[2] = 6	Restart: Restarting

State Target of Command	Command: State
Cfg_CmdTarget[3] = 4	Hold: Holding
Cfg_CmdTarget[4] = 7	Stop: Stopping
Cfg_CmdTarget[5] = 9	Abort: Aborting

State Target of This State Complete	State: State Complete Target
Cfg_SCTarget[0] = 0	Idle: Idle
Cfg_SCTarget[1] = 2	Starting: Running
Cfg_SCTarget[2] = 3	Running: Complete
Cfg_SCTarget[3] = 3	Complete: Complete
Cfg_SCTarget[4] = 5	Holding: Held
Cfg_SCTarget[5] = 5	Held: Held
Cfg_SCTarget[6] = -1	Restarting: *Return State (-1)
Cfg_SCTarget[7] = 8	Stopping: Stopped
Cfg_SCTarget[8] = 8	Stopped: Stopped
Cfg_SCTarget[9] = 10	Aborting: Aborted
Cfg_SCTarget[10] = 10	Aborted: Aborted
Cfg_SCTarget[11] = 0	Resetting: Idle

Cfg_Cmd4All = 2# 0000 0000 0000 0000 0000 0000 0010 0000

This makes the 'Abort' command (5) available to all CmdSrc owners regardless of the current CmdSrc state when enabled.

Cfg_DontUpdRetSt = 2# 0000 0000 0000 0000 0000 0000 0111 0000

Don't record the state numbers for the 'Holding' (4), 'Held' (5) and 'Restarting' (6) states into the return state register. This works with Cfg_SCTarget[6]= -1. When in 'Starting' or 'Running' the return state register is updated to note where it was. When a 'Hold' command is issued the state machine transitions through the holding chain of states but the return state register still has the 'Starting' or 'Running' state number in it. When 'Restarting' is done, it transitions to the state on the return state register.

Cfg_InhCmdOnIntlk = 2# 0000 0000 0000 0000 0000 0000 0000 0101

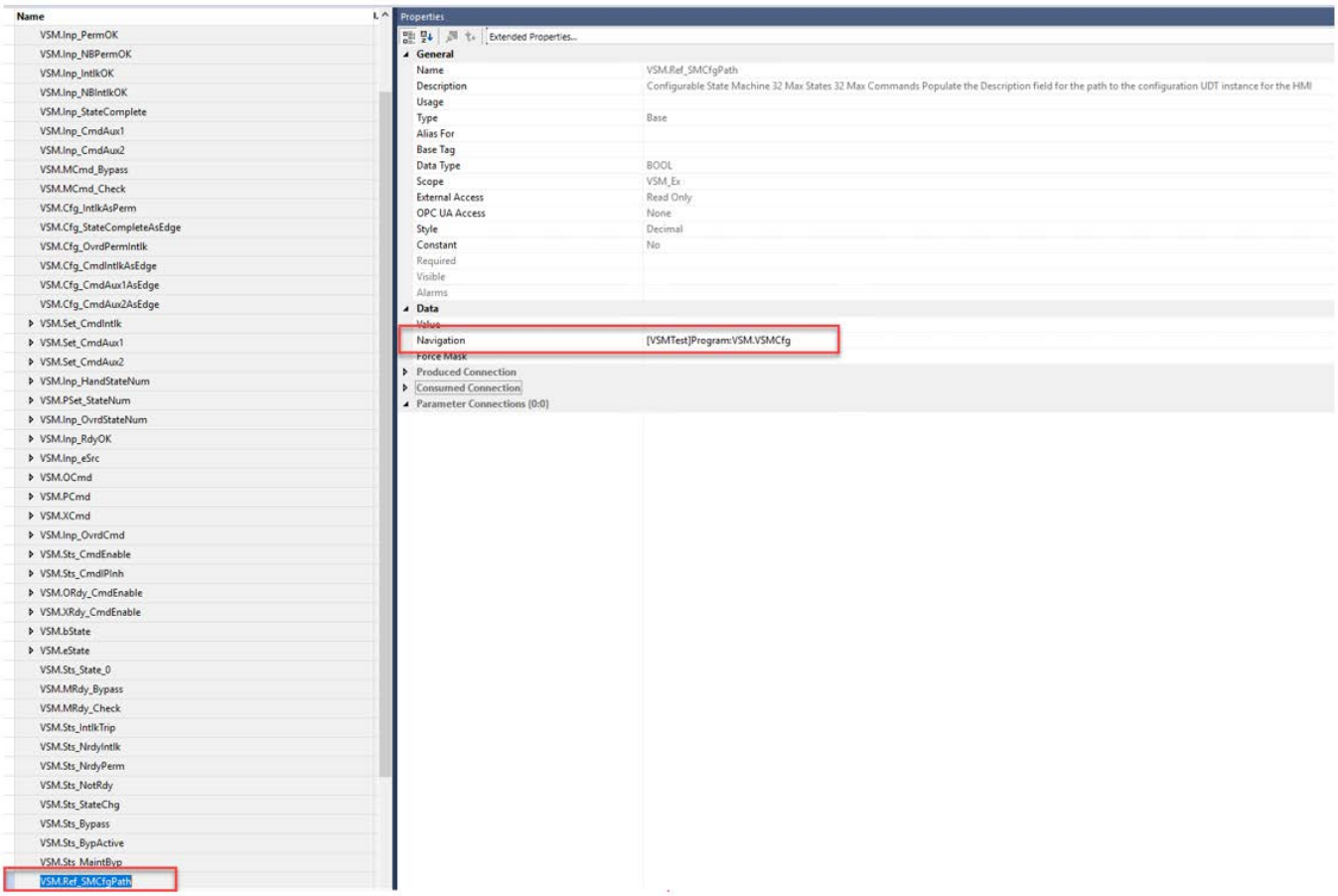
If the interlock is not OK, then inhibit the 'Start' (0) and 'Restart' (2) commands.

Graphic Symbols

A Graphic Symbol (global object) is created once and can be referenced multiple times on multiple displays in an application. When changes are made to the original (base) object, the instantiated copies (reference objects) are automatically updated. Use of graphic symbols, with tag structures in the ControlLogix® system, aid consistency and save engineering time. The configuration tag for the state machine backing tag is stored in the Parent Add-On Instruction tag Ref_SMCfgPath.@Navigation. The macro NavToVSM.mcr is used to process the indirect reference.

Graphic Symbol Name	Graphic Symbol	Description
GO_nav_VSM	s...s	Show the VSM Config display
GO_nav_VSM_Indirect	State Machine	Display the VSM Config from a faceplate display

The graphic faceplates require the tag name for the Cfg_VSM tag. You must set the @Navigation extended tag property on the VSM.Ref_SMCfgPath tag.



FactoryTalk View SE Faceplates

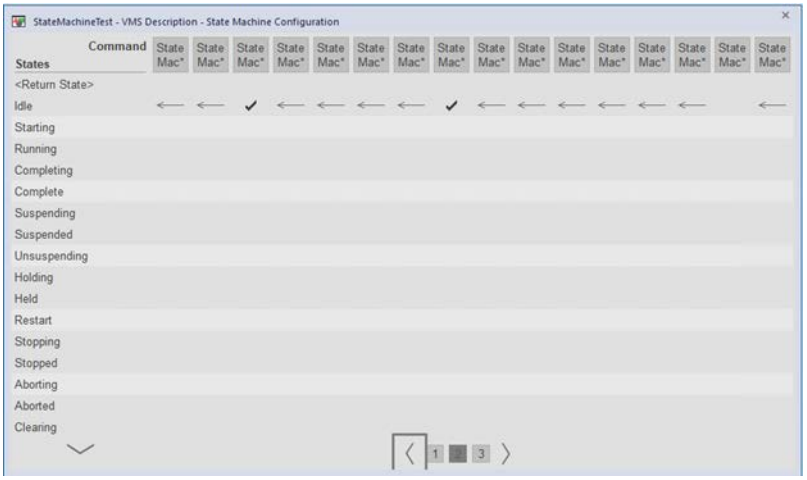
Configuration Page 1

Page 1 displays the command configurations for commands 0-15.

Item	Description
1	Check Box = The command is allowed in this state
2	Arrow = Target State when the command is executed
3	Shift the list downward to allow the user to view the remaining of the 32 states.
4	Each Command button navigates the user to a Command Configuration faceplate. From here the user can configure the states in which that command is available and which state is the target state for that command.

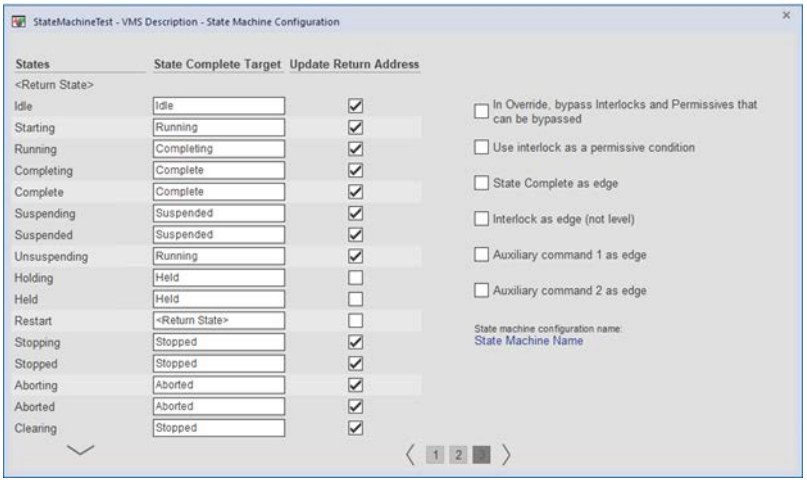
Configuration Page 2

Page two displays the command configurations for commands 16...31.

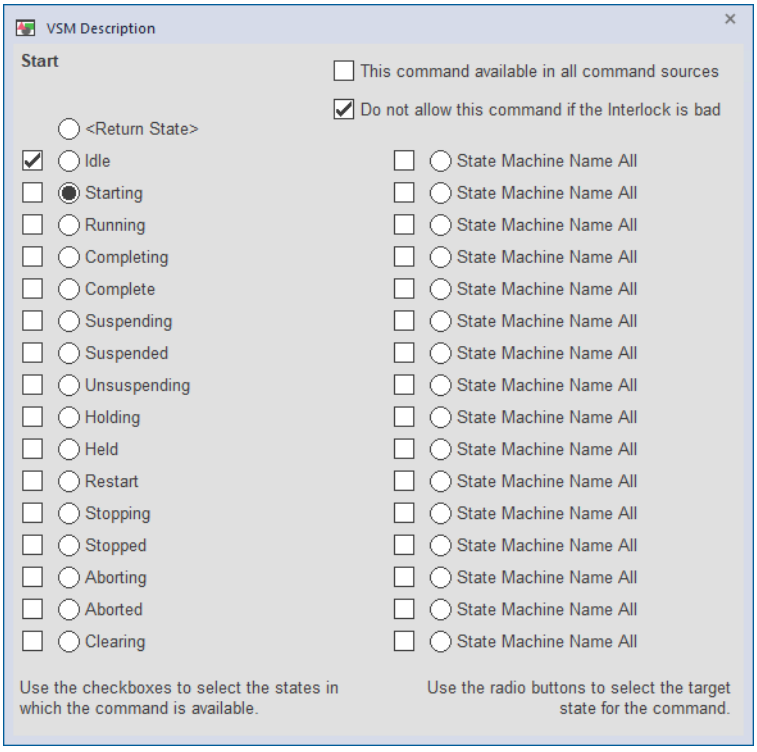


Configuration Page 3

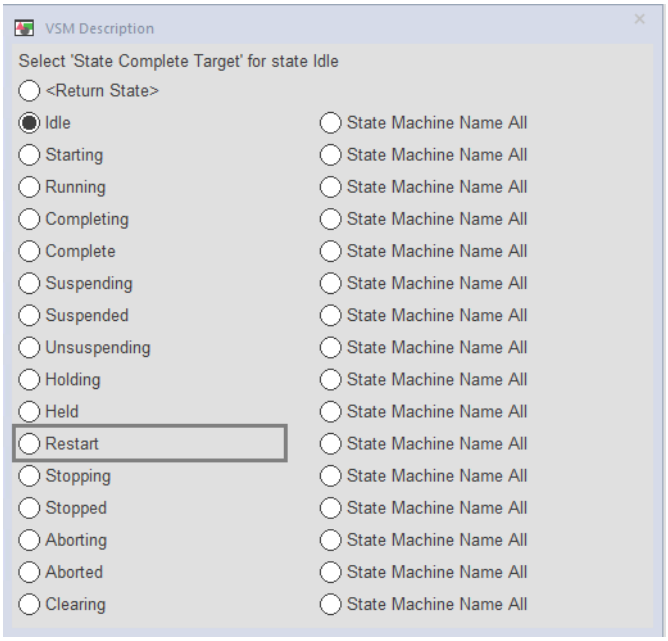
Page 3 provides a view of the state configuration. By clicking any of the state complete targets, you can navigate to the Select-State faceplate display and change the state complete target.



Command Configuration



Select State



Generic Equipment Phase (raP_Opr_EPGen)

An equipment phase is a functional group of equipment that can conduct a finite number of specific minor processing activities when directed by a (recipe) phase.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

The raP_Opr_EPGen (Generic Equipment Phase Module) object controls an Equipment Phase in various command sources and monitors for fault conditions.

Use when:

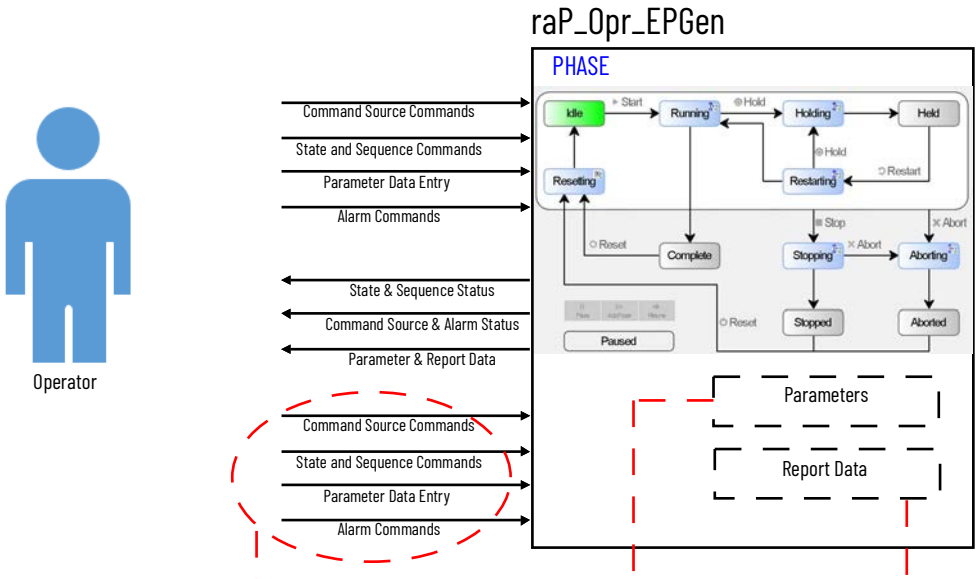
- You want to group equipment, and you want to apply the ISA 88.01 state model using PhaseManager™
- You want to provide the following for a group of equipment
  - Apply a mode model to the equipment group
  - Apply interlocks and/or permissives to the group of equipment
  - Parameters that define the behavior of the group of equipment
  - Report / Resultant data from the group of equipment
  - A faceplate that allows monitoring / control of the equipment grouping
  - Monitor step (description), and allow forcing of steps in maintenance command source
  - Allow alarms to be defined for certain process / equipment failure conditions
  - Alarming function, including alarms based on device failure.

Functional Description

Phase Manager Program

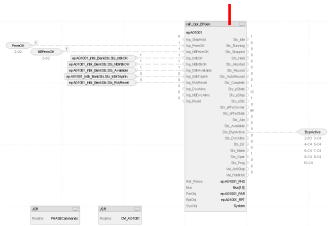
Dispatch

Contains raP_Opr_EPGen instruction and any external instructions required.



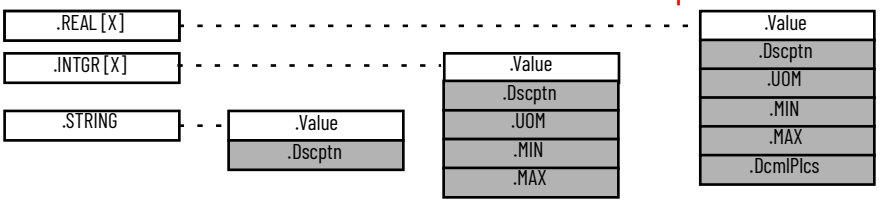
PHASECommands

Contains commands from your logic to raP_Opr_EPGen (as required)
Note: FactoryTalk® Batch issues commands directly to raP_Opr_EPGen via Phase Manager - no logic is required.



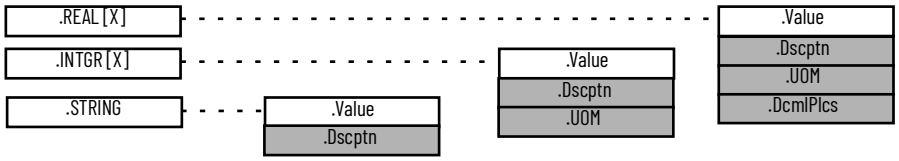
Parameters

Contains logic that maps parameters to raP_Opr_EPGen to Phase Manager tags (Input)



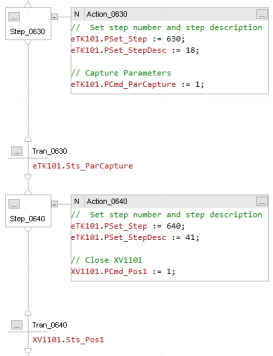
Reports

Contains logic that maps report data from raP_Opr_EPGen to Phase Manager tags (Output)



Phase State Routines

Contains your logic that sequences and coordinates devices (implement states as required)



Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller File

The raP_Opr_EPGen_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

The primary operations of the raP_Opr_EPGen (Generic Equipment Phase Module) are to:

- Provides ISA 88 states, and associated commands
- Provides program structure as a container for coordination and sequencing logic.
- Provides Parameter display, and data entry (operator command source).
- Provides resultant (report) data display.
- Allow monitoring of process Step, and display Equipment Phase status.
- Monitor permissives, helping prevent Equipment Phase operation.
- Monitor interlock conditions to help prevent Equipment Phase operation or create failure condition.
- Provide a stepping mechanism, including the ability to force steps (maintenance)
- Monitor various Equipment Phase failure conditions, and produce alarms.
- Operate in maintenance, program, and operator command sources.
- Provide an "available" status for use by automation logic, to indicate that the Equipment Phase is available for operation.
- Provide a propagation mechanism to allow the Equipment Phase to publish status to and receive status from a group of equipment.
- Provides an interface to parameter display, data entry, and configuration.
- Provides an interface to resultant (report) data display and configuration.
- Provides interface to Prompt Response and configuration

Command Sources

The raP_Opr_EPGen (Generic Equipment Phase Module) uses the standard command sources that are implemented using an embedded PCMDSRC instruction. See PlantPAx Process Control Instructions, publication PROCES-RM215 for more information.

Phase Manager

The raP_Opr_EPGen (Generic Equipment Phase Module) is designed to operate with PhaseManager.

PhaseManager provides the following:

- ISA 88 state model, which can be executed by FactoryTalk® Batch, Studio 5000 Logix Designer®, or program logic.

- Program Structure, which contains phase state routines
- Program scoped tags, which allow individual tags to be configured as Input (parameters from FTBatch) or Output (resultant data, or report data, to FTBatch) for a particular PhaseManager phase.
- Phase data structure, which allows interface to the PhaseManager phase
- An instruction set for issuing commands, and controlling the execution of the PhaseManager phase.

The raP_Opr_EPGen (Generic Equipment Phase Module) provides an embedded reference to an associated PhaseManager Phase. The raP_Opr_EPGen (Generic Equipment Phase Module):

- Provides a human machine interface (faceplate), which allows control and monitoring of the PhaseManager phase.
- Provides a predefined human machine interface (faceplate), which allows input and monitoring of parameters (linked to program tags) and monitoring of resultant/report data (linked to program tags)
- Provides a normalized logic command interface, which utilizes the PhaseManager instruction set.

Program Structure

The raP_Opr_EPGen (Generic Equipment Phase Module) utilizes the PhaseManager program structure, as follows:

Routine	Description
Dispatch	Contains raP_Opr_EPGen instance, external function instances (Interlock, Permissive, Associated Device), and Phase State routinecalls. Phase state routines are a component of a PhaseManager Phase. • One or all available Phase state routines may be implemented. • The logic within a particular Phase state routine is executed when the Phase is in the corresponding state. • Any implemented Phase state routine requires a Phase State Complete instruction (which the state engine uses to determine the state is complete).
Aborting	Used for shutting down equipment in an emergency situation. If you have implemented Stopping, you would at a minimum duplicate the stopping logic within Aborting. In some cases, the sequence in an emergency situation (Aborting) differs from the orderly shutdown of equipment (Stopping).
Holding	Used if equipment or a subset of equipment must be shut down when the phase enters the hold state. It may also be advantageous to release owned equipment if maintaining ownership while held constrains production by maintaining ownership of shared equipment.
Resetting	Used to perform "clean-up" activities such as release owned equipment, etc.
Restarting	Implemented if Holding is implemented. Used to bring equipment from the state that it is in at the end of the Holding state back to the state that it was in prior Holding. Used in conjunction with PSet_HoldIndx and Val_LastRunningStep to return execution to the proper sequence step.
Running	Use to start up equipment, and acquire ownership of equipment (if necessary).
Stopping	Use if equipment must be shut down in a given sequence.
AlarmsSuppress	Contains EP_GEN alarm suppression logic.
Interlocks	Contains EP_GEN interlock mapping from interlock conditions to <EP_GEN>_Intlk block.
Parameter	Contains EP_GEN parameter mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to EP_GEN instance.
Permissives	Contains EP_GEN permissive mapping from permissive conditions to <EP_GEN>_Perm block.
<EP_GEN>_PhaseCommands	Maps commands from EP_GEN to PhaseManager commands
<EP_GEN>_PXRQ	PXRQ Routine container. Use the PRXQ instruction to initiate communication with FTBatch software.
Reports	Contains EP_GEN report mapping to and from Parameter blocks (_ParRpt (Enum, Integer, Real, String)) to EP_GEN instance.
ExtddAlarms	Contains EP_GEN instances of external alarm instances and trigger logic.

IMPORTANT Routines listed in the table above, are located within the PhaseManager program. This represents an example for implementing PhaseManager with the raP_Opr_EPGen (Generic Equipment Phase Module).

PhaseManager may be implemented without the raP_Opr_EPGen (Generic Equipment Phase Module), in which case the PhaseManager program may be structured as desired. See the PhaseManager User Manual, Publication LOGIX-UM001.

Alarms

The raP_Opr_EPGen Instruction uses the following alarms, which are implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
Device alarms	Alm_DvcAlms	Raised when a device within the Equipment Phase has an alarm.
Interlock trip	Alm_IntlkTrip	Raised when an interlock condition triggers a change in state of the Equipment Phase.
Report data	Alm_RptData	Raised when new report data are available.

Virtualization

The raP_Opr_EPGen Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (False Rung)	Handle processing for EnableIn False (False Rung) the same as if the Equipment Module were Disabled by Command. The Equipment Module outputs are de-energized and the Equipment Module is shown as Disabled on the HMI.
Powerup (Pre-scan, First Scan)	Handles processing of command sources and alarms on Pre-scan and Powerup. On Powerup, the Equipment Module is treated as if it were Commanded to Reset all Program and Operator command.
Postscan (SFC Transition)	No SFC Postscan logic is provided.

See Logix 5000 Controllers Add-On Instructions: Programming Manual, 1756-PM010 for more information.

ATTENTION: Disabling the raP_Opr_EPGen Add-On Instruction causes Equipment Phase outputs to become de-energized.

Local Message

The object raP_Opr_EPGen utilizes local message display elements to display Step Names, Material Names, and Summary information. A default local message file is provided for each information type. This default local message file populates the local message display elements from tags in the controller. For Step Names and Material names, these are the same controller tags that are used in previous versions of the library. The difference is that 512 messages are available, rather than the 99 messages in the previous version. To upgrade from previous versions, developers must add the local message file to the project and set the @Navigation property of the specified tag to the Local Message file name (see table below).

Information	Default Local Message File	File Name Reference	Default Controller Data
Material Name	SystemMaterialNames	Sts_eMtrl.@Navigation	System.Enum.Materials[x].@Description
Step Description	SystemStepDescriptions	Sts_eStep.@Navigation	System.Enum.Step_Desc[x].@Description
Summary Information	SystemSummary	Sts_eSummary.@Navigation	System.Enum.Summary_Desc[1].@Description

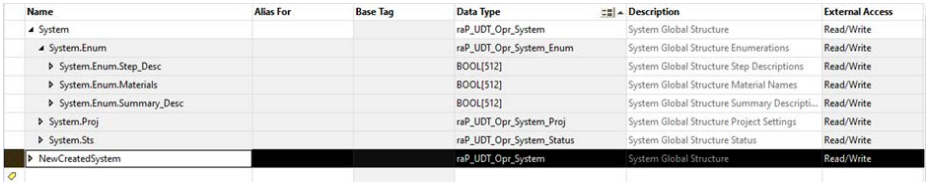
Users may add customized local messages for individual objects by creating a new local message file and populating the file with the customized strings or tag references. Then set the @Navigation property of the specified tag to the name of the new custom file.

FactoryTalk Optix Local Message

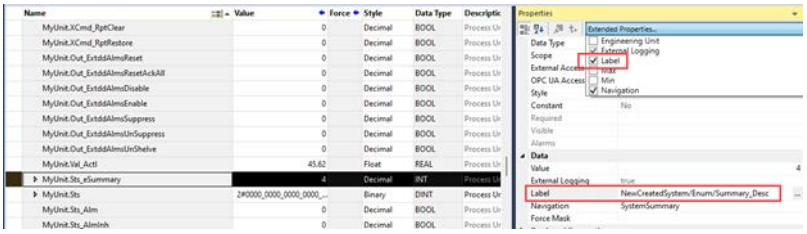
The object raP_Opr_Unit uses the @Label property of the specified tag to input the path for displaying Step Names, Material Names, and Summary information. FactoryTalk® Optix™ does not require Local Message files. For Step Names, Material Names, and Summary, these are the same controller tags used with FactoryTalk ViewSE. FactoryTalk Optix directly retrieves Controller Data using the path specified in the @Label property of Logix Designer. Users must set the path information into @Label property of the specified tag to a Controller Data Path (see the following table).

Information	Reference	Customized Controller Data Path	Default Controller Data
Material Name	Sts_eMtrl@Label	System/Enum/Materials	System.Enum.Materials[x].@Description
Step Description	Sts_eStep@Label	System/Enum/Step_Desc	System.Enum.Step_Desc[x].@Description
Summary Information	Sts_eSummary@Label	System/Enum/Summary_Desc	System.Enum.Summary_Desc[1].@Description

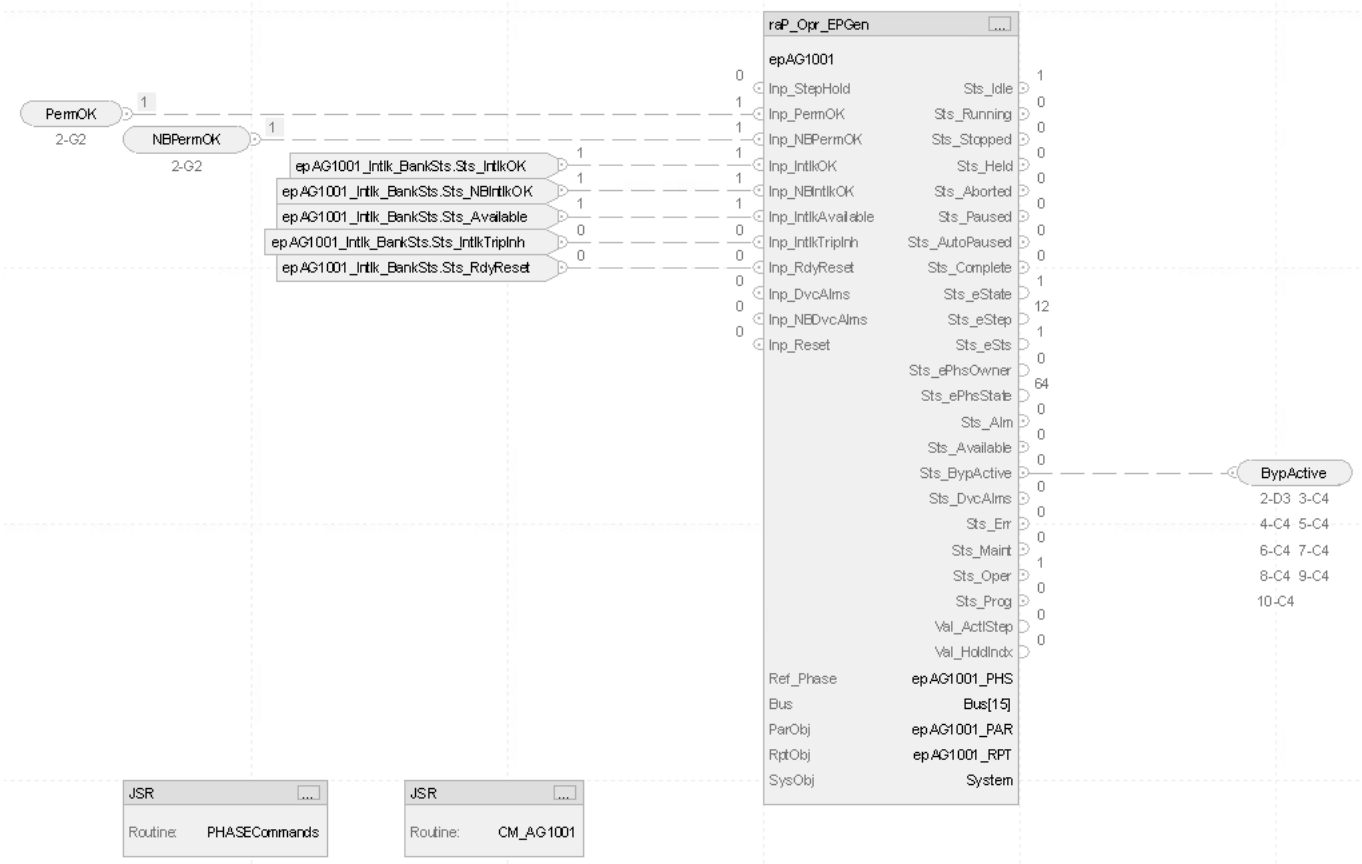
Users may add Customized Controller Data for individual objects by creating a new tag member with the Data Type "raP_UDT_Opr_System" in Logix Designer.



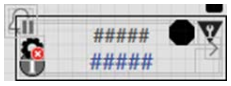
Then set the @Label property of the specified tag to the Customized Controller Data Path and import the tag with the customized extended properties into FactoryTalk Optix.



Programming Example



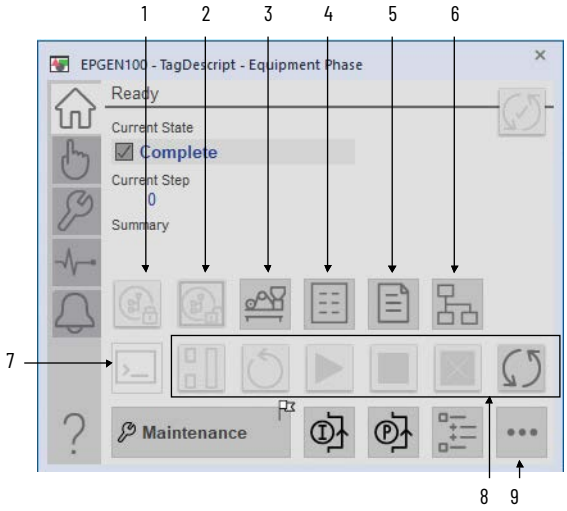
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_PEPGEN 	raP_5_30_GS_raP_Opr_EPGen 	The raP_Opr_EMGen (Generic Equipment Module) object controls an Equipment Module in various command sources and monitors for fault conditions.

FactoryTalk View SE Faceplates

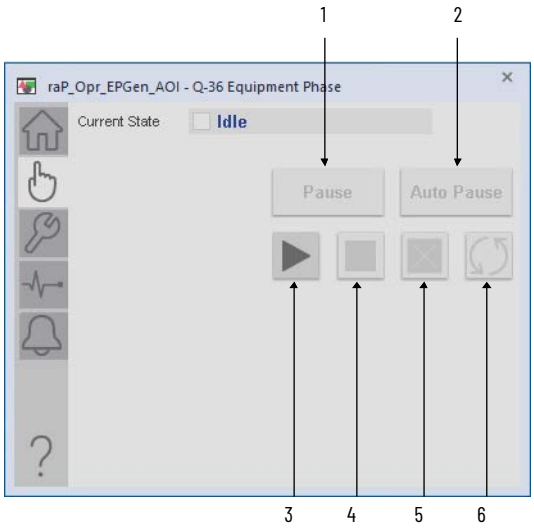
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28

Operator Tab



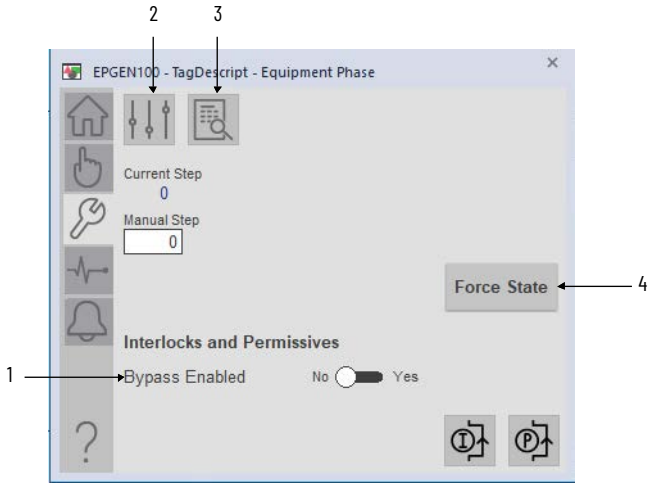
Item	Description
1	Acquire child command source
2	Release child command source
3	Display Bus faceplate for this object
4	Show parameter display
5	Show report display
6	Show State Detail display
7	Respond to Prompt request
8	Phase Commands (from left to right): Hold phase, Restart phase, Start phase, Stop phase, Abort phase, Reset phase
9	Display more information

Manual Control



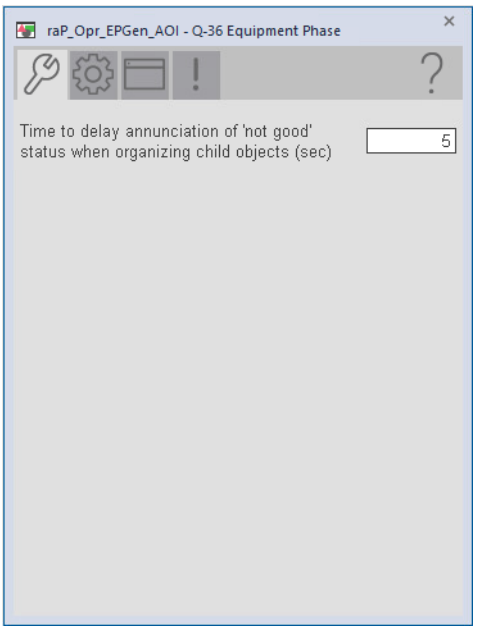
Item	Description
1	Pause phase
2	Auto pause the phase
3	Start phase
4	Stop phase
5	Abort phase
6	Reset phase

Maintenance Tab



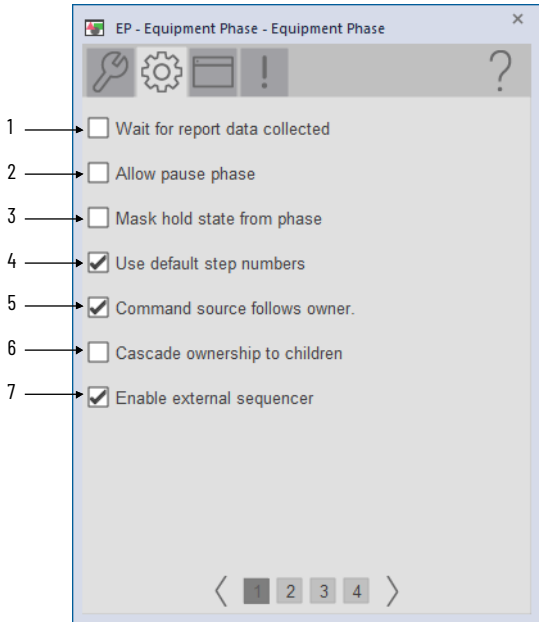
Item	Description
1	Select Yes to enable bypass
2	Display Advanced Properties
3	Navigate to detail display
4	The state force button is a Maintenance source command that is used to set the Sts_StateCompleteRqst bit. For an instance without the VSM the user will need to connect the Sts_StateCompleteRqst to the required location in their SM logic.

Advanced Maintenance

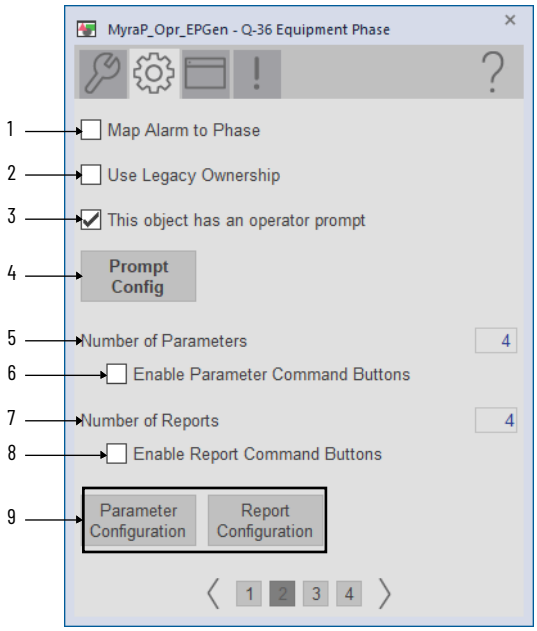


The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

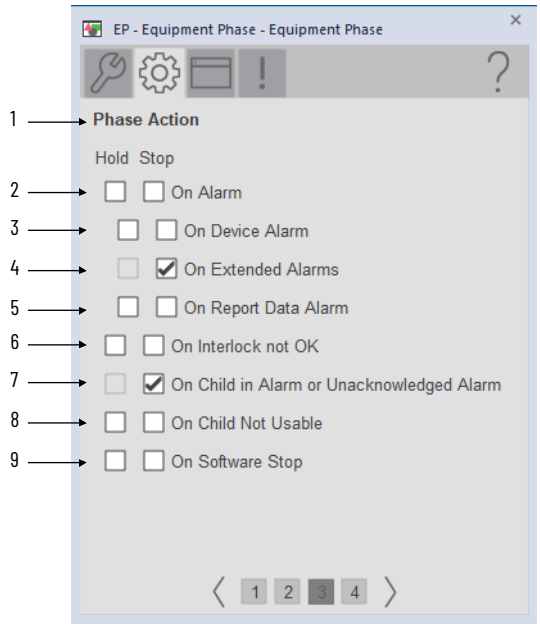
Engineering Tab



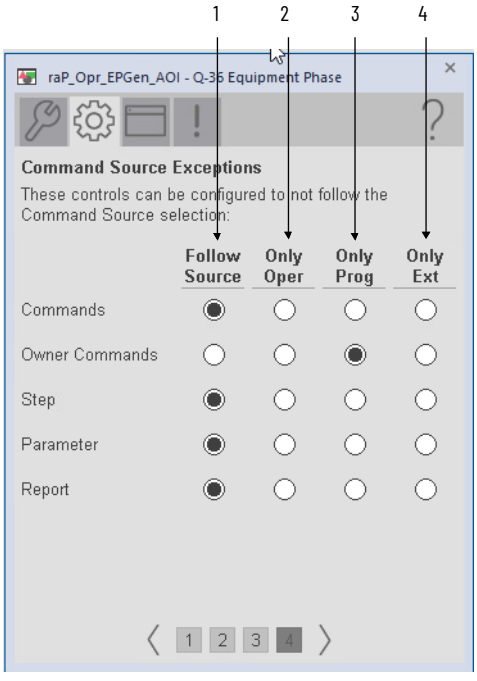
Item	Description
1	Select to wait for report data that is collected before alarming.
2	Select to allow a pause phase
3	Select to mask hold state from phase
4	Select to use default step numbers
5	Select to have the command source follow the owner
6	Select to cascade ownership to children (children will be owned when this object is owned)
7	This phase has an external sequence that is associated to it. (FTBatch)



Item	Description
1	Map alarm code form the equipment phase to the phase with the PFL instruction.
2	Use legacy object ownership. Use PCmd_Owner to set Val_Owner.
3	Select to enable an operator prompt
4	Select to open the Prompt configuration
5	Define the number of Parameters.
6	Select to enable parameter command buttons
7	Define the number of Reports.
8	Select to enable report command buttons
9	Select to show parameter configuration display (left) or report configuration display (right)

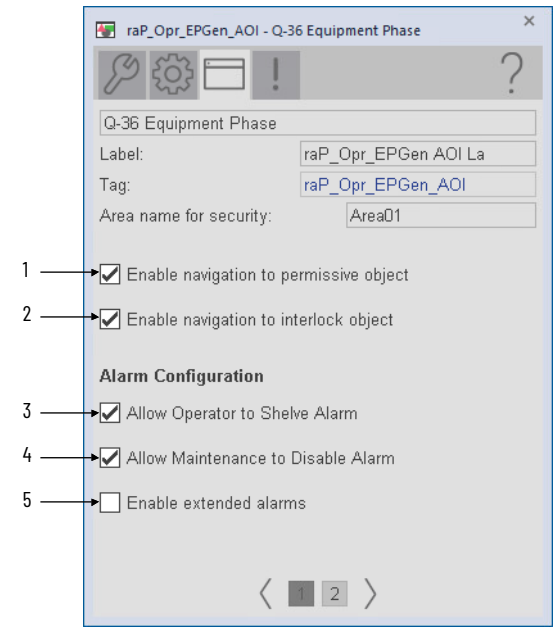


Item	Description
1	Select conditions to Hold or Stop phase
2	On alarm active
3	Device alarms condition, external devices connected via Inp_DvcAlms
4	Extended alarms active, connected via Inp_ExtddAlmsAlm
5	Report data alarm, report data not received from external system in set time.
6	Interlock not ok
7	Child alarm active or UnAckd via OOAP
8	Child cannot be owned or is in a state that makes it unusable via OOAP
9	Software stop active

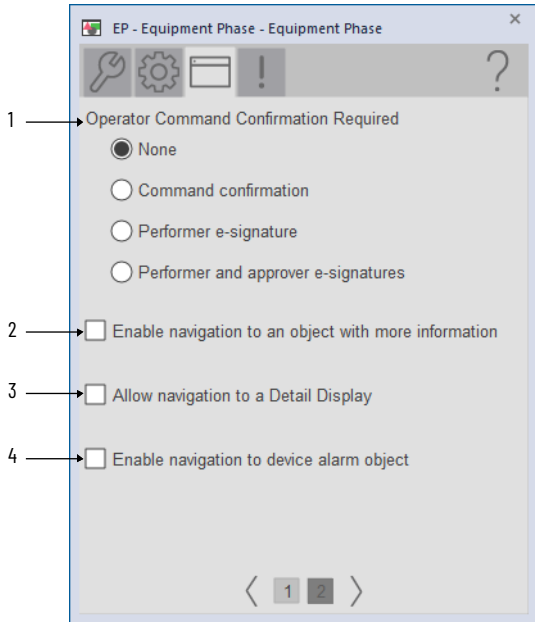


Item	Description
1	Control of this feature is determined by the current command source
2	This feature will always be commanded by the Operator
3	This feature will always be commanded by the Program Logic
4	This feature will always be commanded by the External Source

HMI Configuration Tab



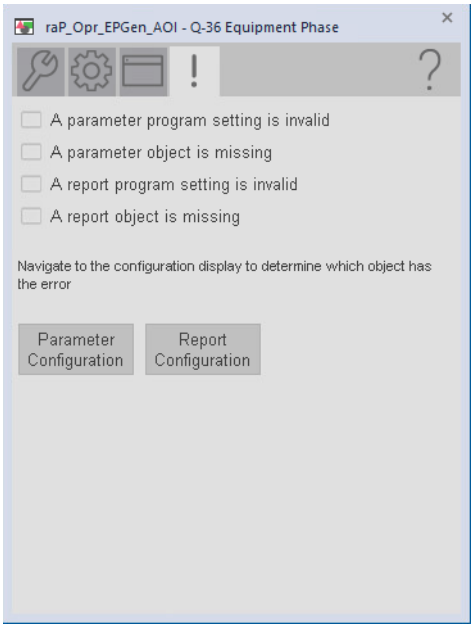
Item	Description
1	Select to enable navigation to permissive object
2	Select to enable navigation to interlock object
3	Select to allow Operator to shelve alarm
4	Select to allow Maintenance to disable alarm
5	Select to enable extended alarms



Item	Description
1	Select an option for Operator Command Confirmation Requirements.
2	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the objects faceplate.
3	Select to allow navigation to detail display.
4	Select to allow navigation to a device alarm object.

Faults Tab

The Faults tab contains specific reasons that the device is not ready.

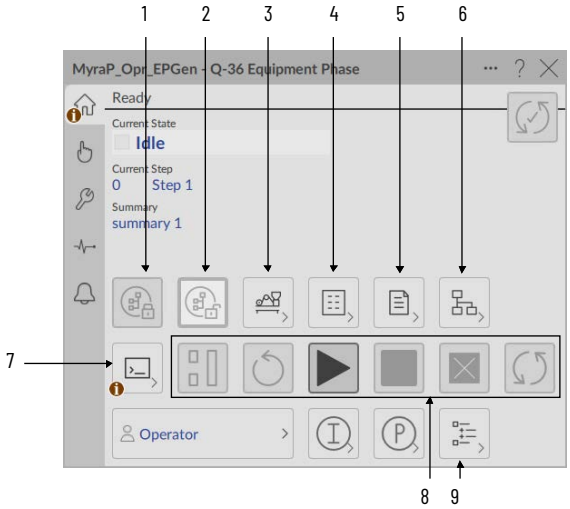


FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

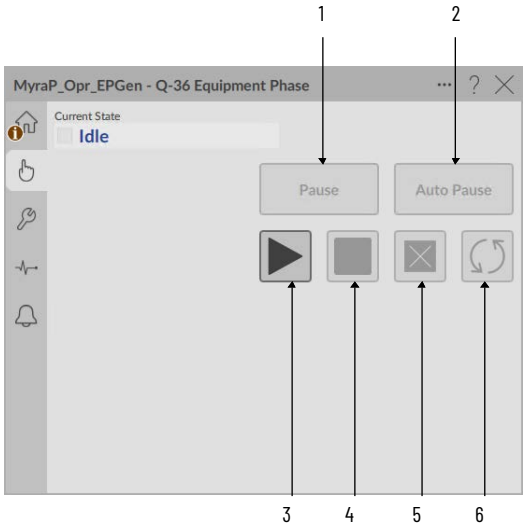
Any feature that is contained in the FactoryTalk Optix faceplates has the same functionality as used in the FactoryTalk View SE faceplates. See FactoryTalk View SE Faceplates on page 274.

Operator Tab



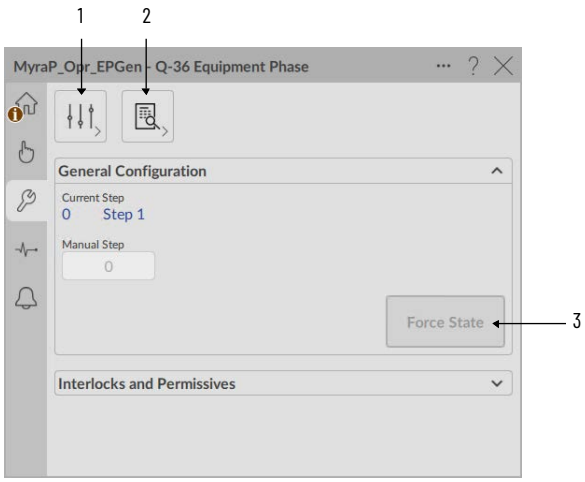
Item	Description
1	Acquire child command source
2	Release child command source
3	Display Bus faceplate for this object
4	Show parameter display
5	Show report display
6	Show State Detail display
7	Respond to Prompt request
8	Phase Commands (from left to right): Hold phase, Restart phase, Start phase, Stop phase, Abort phase, Reset phase
9	Display more information

Manual Control



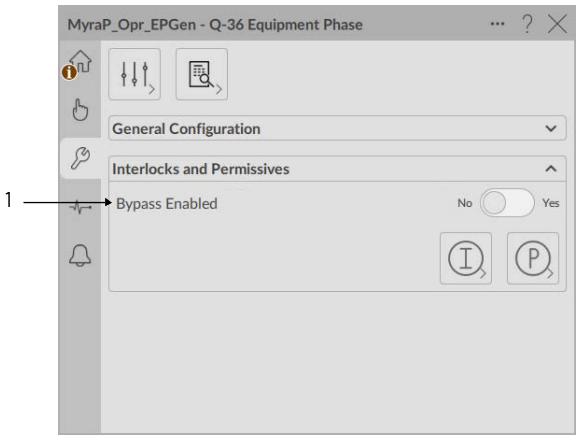
Item	Description
1	Pause phase
2	Auto pause the phase
3	Start phase
4	Stop phase
5	Abort phase
6	Reset phase

Maintenance Tab



Item	Description
1	Display Advanced Properties
2	Navigate to detail display
3	The state force button is a Maintenance source command that is used to set the Sts_StateCompleteRqst bit. For an instance without the VSM the user will need to connect the Sts_StateCompleteRqst to the required location in their SM logic.

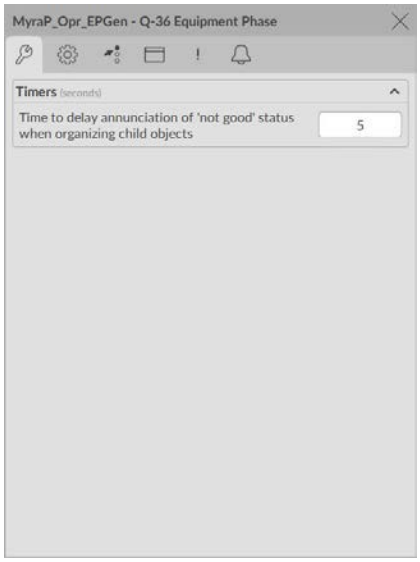
Maintenance Tab - Interlocks and Permissives



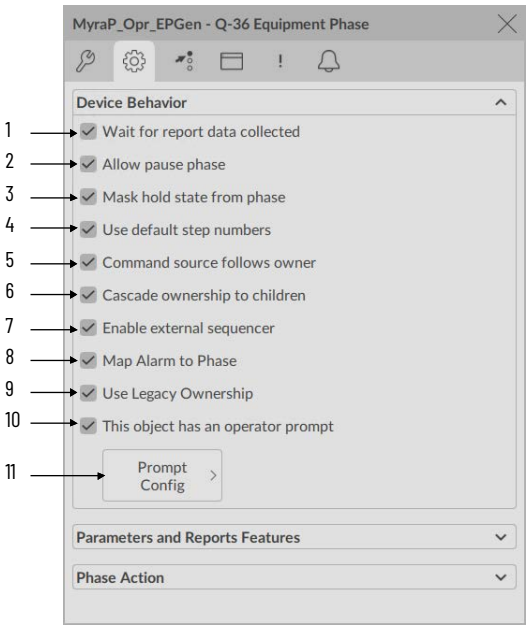
Item	Description
1	Select Yes to enable bypass

Advanced Maintenance Tab

The timer creates a delay for the Tree View to indicate that Children are 'not good' upon ownership acquisition. This is done to avoid nuisance indications on the Tree View while waiting for children to be acquired. The default of five seconds is sufficient delay for most applications. You may wish to raise that value if child acquisition takes longer than this. This can occur if the organization has many nested organizational levels or nested elements have relatively long scan intervals. This value is limited less than 3600 seconds.

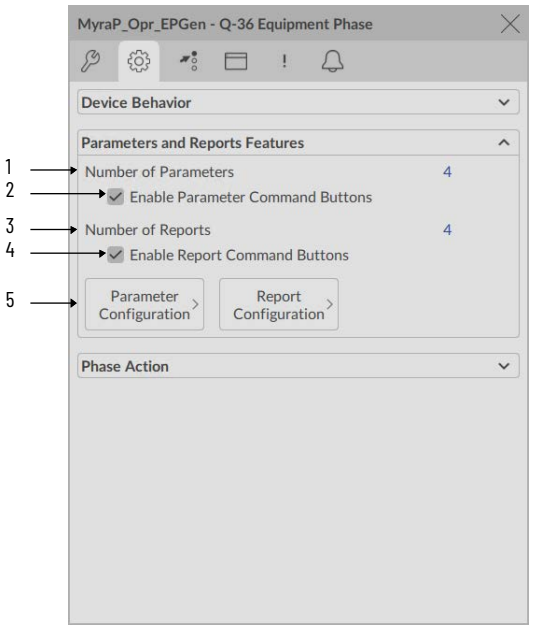


Advanced Engineering Tab - Device Behavior



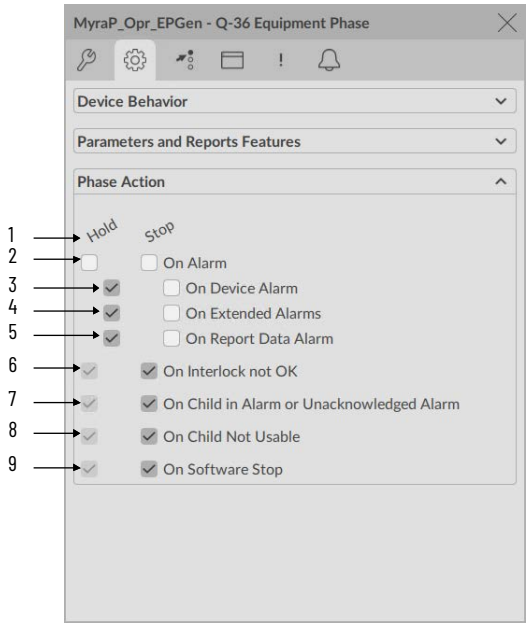
Item	Description
1	Select to wait for report data that is collected before alarming.
2	Select to allow a pause phase.
3	Select to mask hold state from phase.
4	Select to use default step numbers.
5	Select to have the command source follow the owner.
6	Select to cascade ownership to children (children will be owned when this object is owned)
7	This phase has an external sequence that is ass.ociated to it. (FTBatch).
8	Map alarm code form the equipment phase to the phase with the PFL instruction.
9	Use legacy object ownership. Use PCmd_Owner to set Val_Owner.
10	Select to enable an operator prompt.
11	Select to navigate to the Prompt faceplate.

Advanced Engineering Tab - Parameters and Reports Features



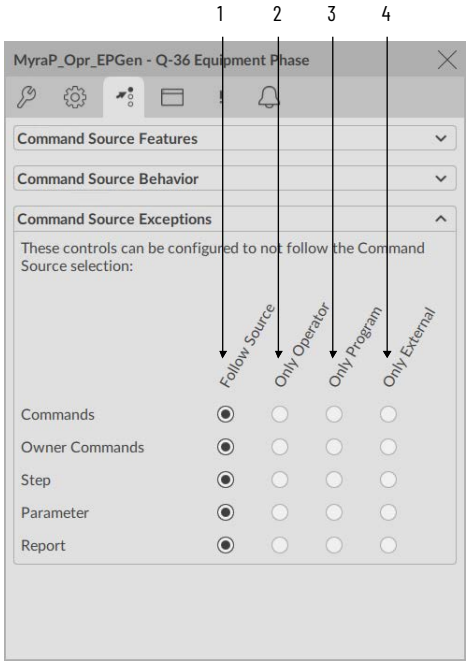
Item	Description
1	Number of Parameters configured.
2	Select to enable parameter command buttons.
3	Number of Reports configured.
4	Select to enable report command buttons.
5	Select to show parameter configuration display (left) or report configuration display (right).

Advanced Engineering Tab - Phase Action



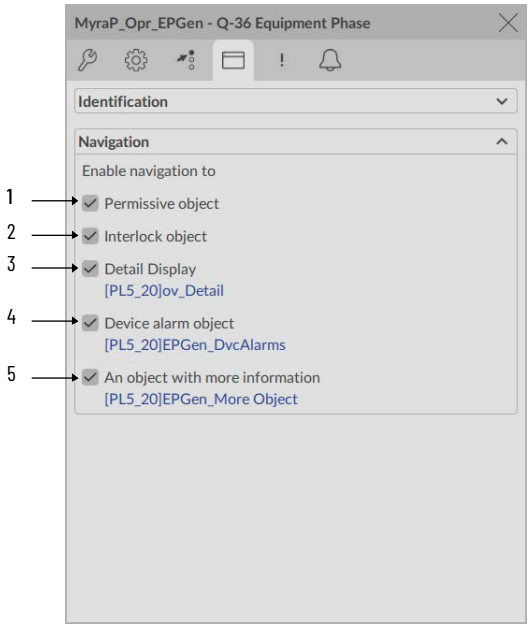
Item	Description
1	Select conditions to Hold or Stop phase
2	On alarm active
3	Device alarms condition, external devices connected via Inp_DvcAlms
4	Extended alarms active, connected via Inp_ExtddAlmsAlm
5	Report data alarm, report data not received from external system in set time.
6	Interlock not ok
7	Child alarm active or UnAckd via OOAP
8	Child cannot be owned or is in a state that makes it unusable via OOAP
9	Software stop active

Advanced Command Source Tab - Command Source Exceptions



Item	Description
1	Control of this feature is determined by the current command source
2	This feature will always be commanded by the Operator
3	This feature will always be commanded by the Program Logic
4	This feature will always be commanded by the External Source

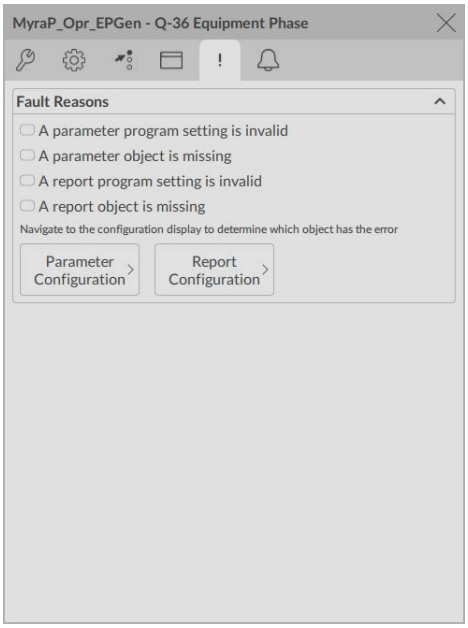
Advanced HMI Configuration Tab - Navigation



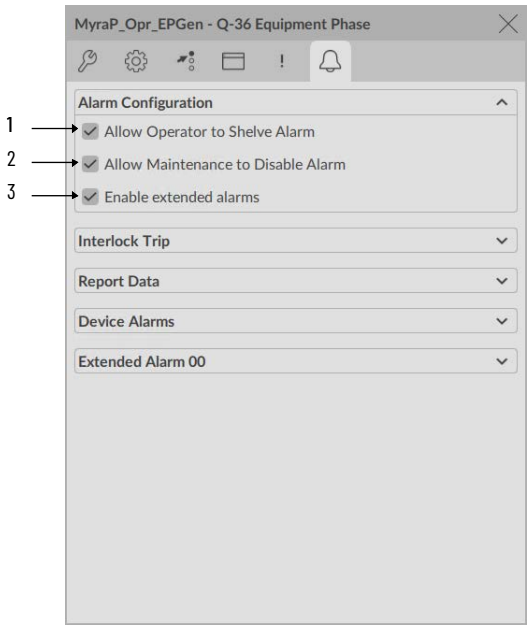
Item	Description
1	Select to enable navigation to permissive object
2	Select to enable navigation to interlock object
3	Select to allow navigation to detail display
4	Select to allow navigation to a device alarm object.
5	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the .@Library and .@Instruction extended tag properties to display the objects faceplate.

Advanced Faults Tab

The Faults tab shows information on the status of the objects. Select the Parameter and Report configuration buttons to determine which object has the fault.



Advanced Alarm Configuration



Item	Description
1	Select to allow Operator to shelve alarm
2	Select to allow Maintenance to disable alarm
3	Select to enable extended alarms

Notes:

Parameter and Reports (raP_Tec_ParRpt)

The raP_Tec_ParRpt Add-On Instruction is used to implement parameter data items. The raP_Tec_ParRpt instruction may be used as follows:

- For a read-only parameter
- For a read/write parameter
- For a parameter of type Integer, Real, String, or Enumeration
- Equipment Module (raP_Opr_EMGen) and Equipment Phase (raP_Opr_EPGen) are designed to work with the raP_Tec_ParRpt instruction

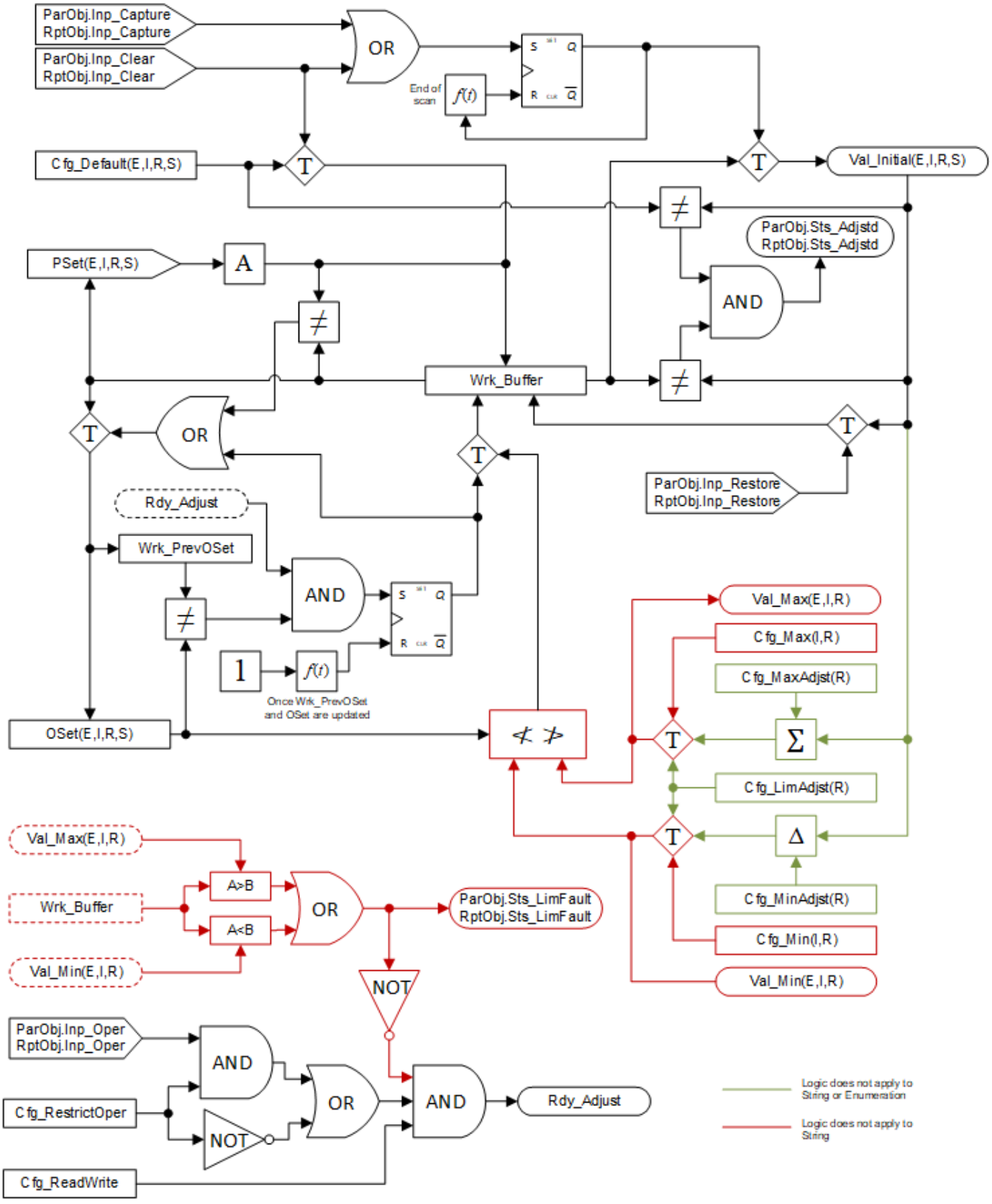
For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Guidelines

Use when:

- You need the ability to view or modify a parameter from either the HMI or from logic
- You must arbitrate parameter input based on mode
- You need the ability to limit the value of a parameter, from either the HMI or logic
- You need the ability to capture an initial parameter value (based on a trigger), and provide an indication if the parameter was adjusted from the initial value
- You must limit the adjustment of a parameter within a deadband relative to an initial value
- You must apply command confirmation (that is, Electronic Signature) to parameter entry from the HMI.
- Your parameter is read-only or read/write
- Your need a Parameter (recipe) or Report (resultant) parameter
- Your parameter is of data type: Integer, Real, String, or is an Enumeration

Functional Description



Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller File

The raP_Tec_ParRpt_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

The primary operations of the raP_Tec_ParRpt (Parameter Instruction) are:

- Captures the initial value of the parameter (snap shot) when the Trigger goes TRUE. Maintains the initial value until the Clear input goes TRUE.
- Permits or denies Operator adjustment of the parameter value. When permitted, allows the adjustment of the parameter value within a deadband of the initial parameter value based on configured limits.
- Compares the initial parameter value to the present parameter value and produces an "Adjusted" status.
- Allows initial parameter values to be restored, when the Reset to Initial input goes TRUE.
- Limits the value of the parameter based on configured Minimum and Maximum limits, and produces a status when the parameter value is beyond limits.
- Allows parameter to be configured as Read, or Read/Write
- Allows a default parameter value to be configured, restores defaults when Clear input goes TRUE.
- Allows the configuration of a text description, and units of measure (engineering units) for the parameter.
- When configured to allow Operator entry and read/write, and "Operator" mode input is true; produces "Ready to Adjust" status, and allows the parameter value to be entered from the HMI.
- Allows command confirmation to be applied to parameter entry from the HMI: No Signature, Performer Signature only, or Performer and Approvers Signatures.

Command Sources

The raP_Tec_ParRpt instruction does not have command sources. However, the raP_Tec_ParRpt provides an input to monitor for Operator mode, and uses this to arbitrate request to modify the parameter value.

Alarms

The raP_Tec_ParRpt instruction does not generate any alarms.

Virtualization

The raP_Tec_ParRpt Instruction has no Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (False Rung)	Handle processing for EnableIn False (False Rung) the same as if the Equipment Module were Disabled by Command. The Equipment Module outputs are de-energized and the Equipment Module is shown as Disabled on the HMI.
Powerup (Pre-scan, First Scan)	Handles processing of modes and alarms on Pre-scan and Powerup. On Powerup, the Equipment Module is treated as if it were Commanded to Reset all Program and Operator command.
Postscan (SFC Transition)	No SFC Postscan logic is provided.

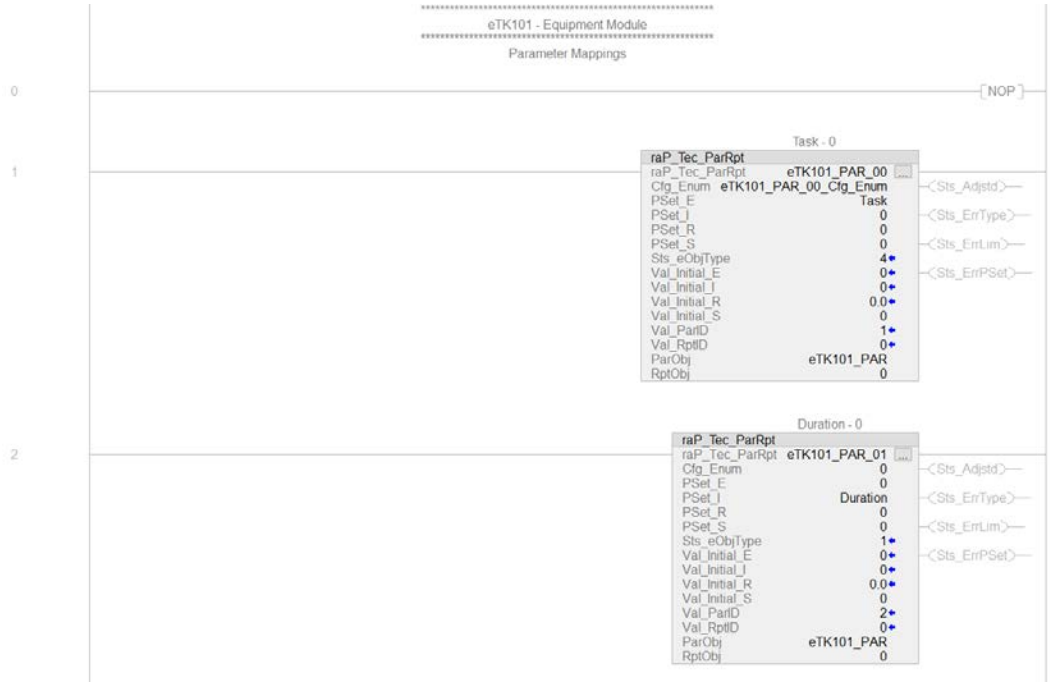
See Logix 5000 Controllers Add-On Instructions: Programming Manual, 1756-PM010 for more information.

ATTENTION: Disabling the raP_Tec_ParRpt Add-On Instruction causes Equipment Phase outputs to become de-energized.

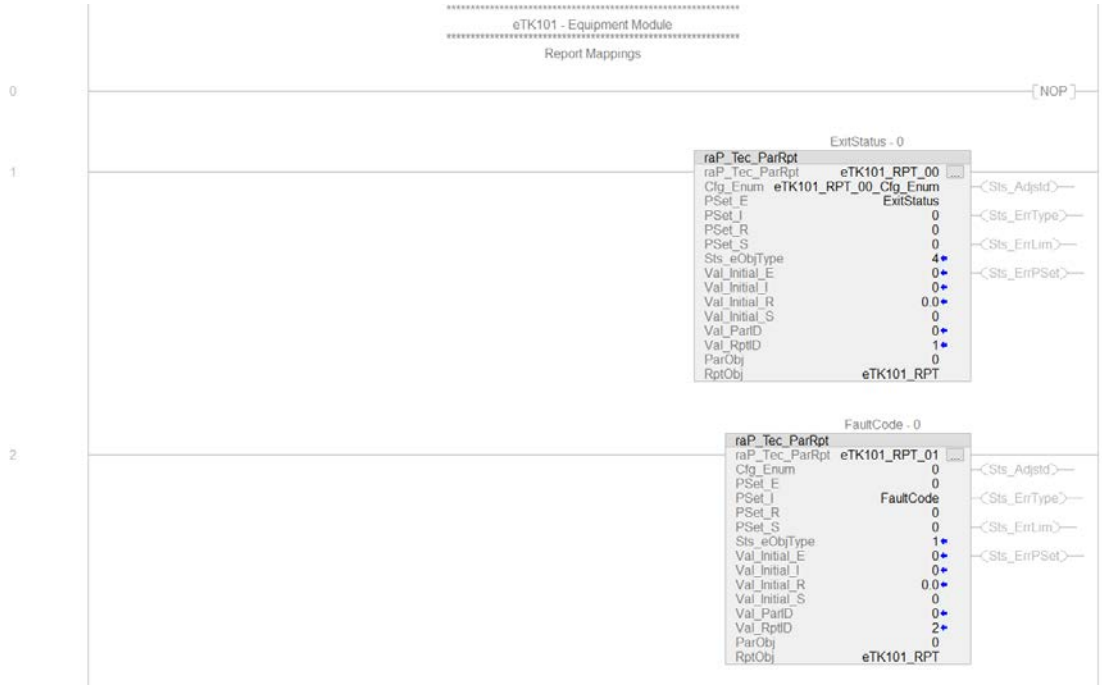
Programming Example

The example in the Function Description section shows the basic use of the raP_Opr_ParRpt Add-On Instruction. Typically, the raP_Opr_ParRpt instruction is not used on its own. The instruction is used with the Unit, EMGen, or the EPGen instructions, the instruction is used for both the Parameters and Reports. Multiples of each can be created as long as it follows the naming convention and has a unique number that is associated with it. The raP_Opr_ParRpt instruction allows for four different options, Enumeration, String, Integer, or Real, only one of these types can be used per instance. When used with the EPGen, the PSet parameter tag is associated to the FTBatch parameter or report.

Parameter Program Example

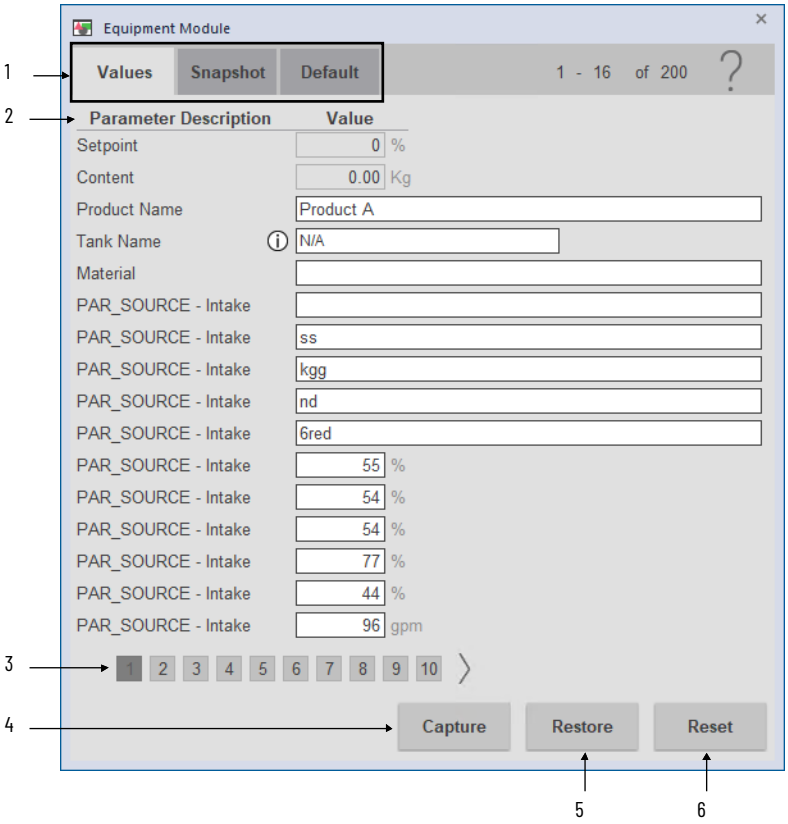


Reports Program Example



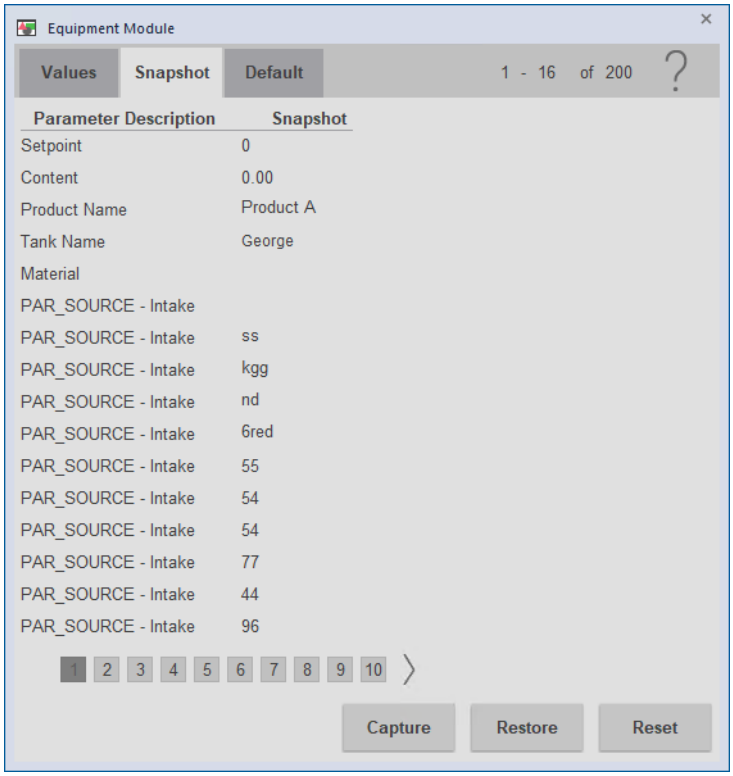
FactoryTalk View SE Faceplates

Parameter Display

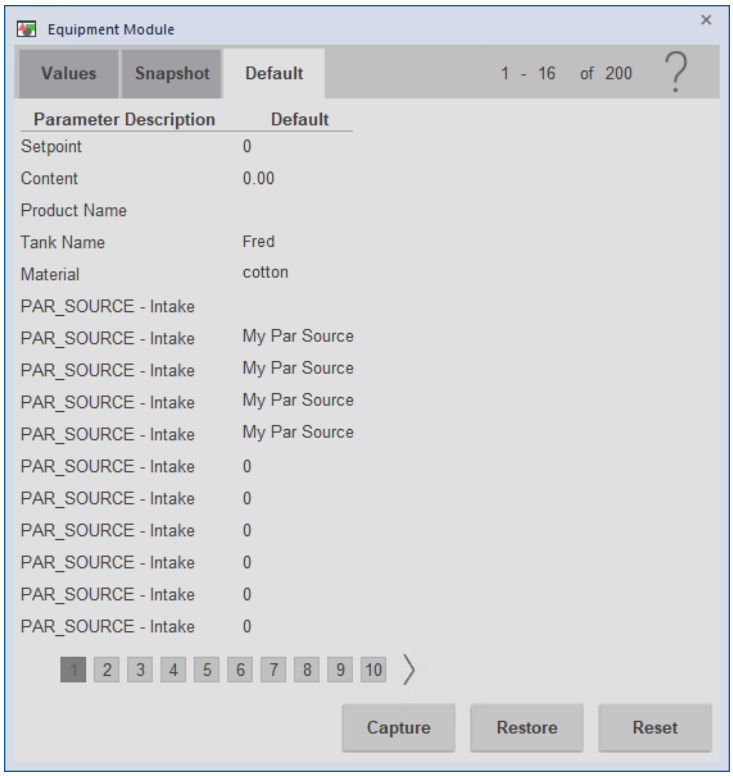


Item	Description
1	Select to navigate between current values, snapshot values, and default values.
2	List of each parameter value and description.
3	Navigation between up to 480 parameters (only visible if more than 16 parameter values used).
4	Capture Snapshot. Press to capture current values into snapshot values.
5	Restore Snapshot. Press to restore snapshot values into current values.
6	Reset to defaults. Press to reset current values to default values.

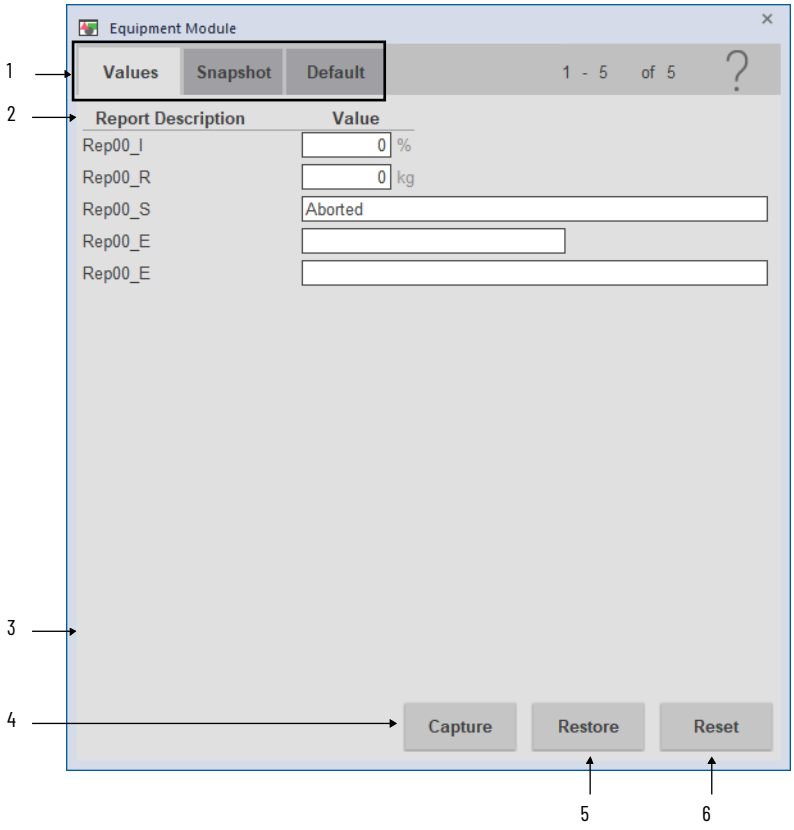
The following page shows the information captured by the snapshot.



The following page shows the default information.

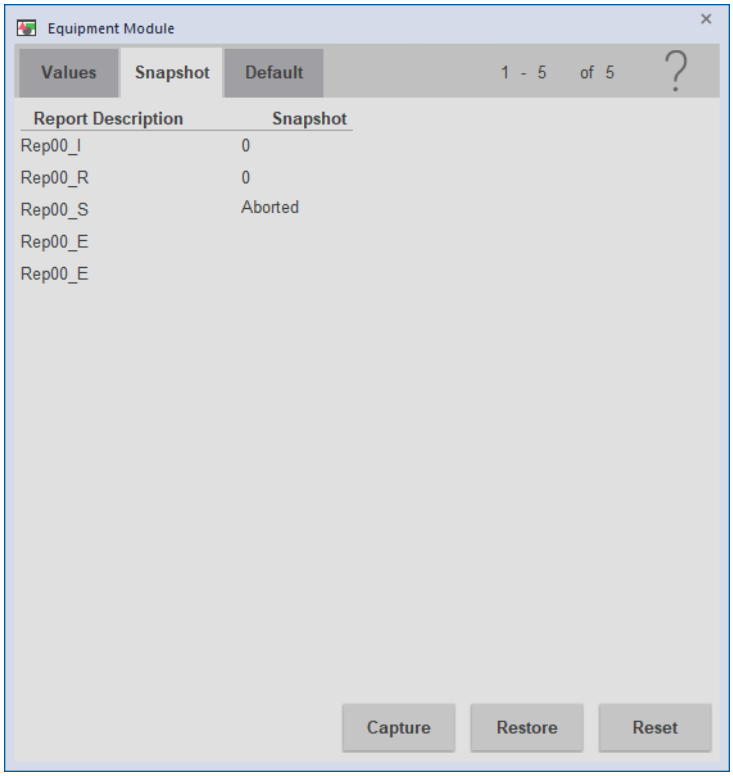


Report Display

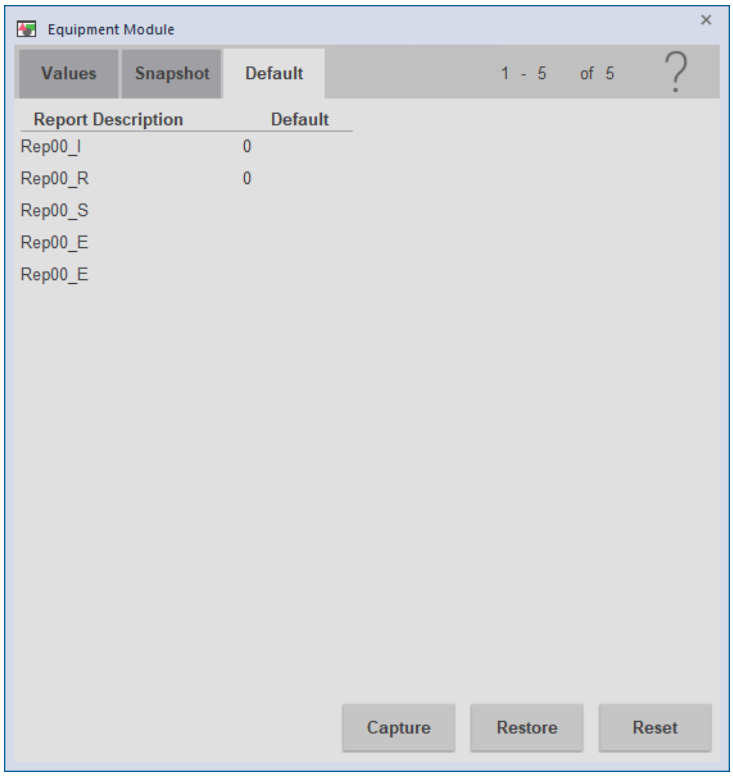


Item	Description
1	Select to navigate between current values, snapshot values, and default values.
2	List of each report value and description.
3	Navigation between up to 480 reports (only visible if more than 16 report values used). See Parameter Display on page 292.
4	Capture Snapshot. Press to capture current values into snapshot values
5	Restore Snapshot. Press to restore snapshot values into current values
6	Reset to defaults. Press to reset current values to default values

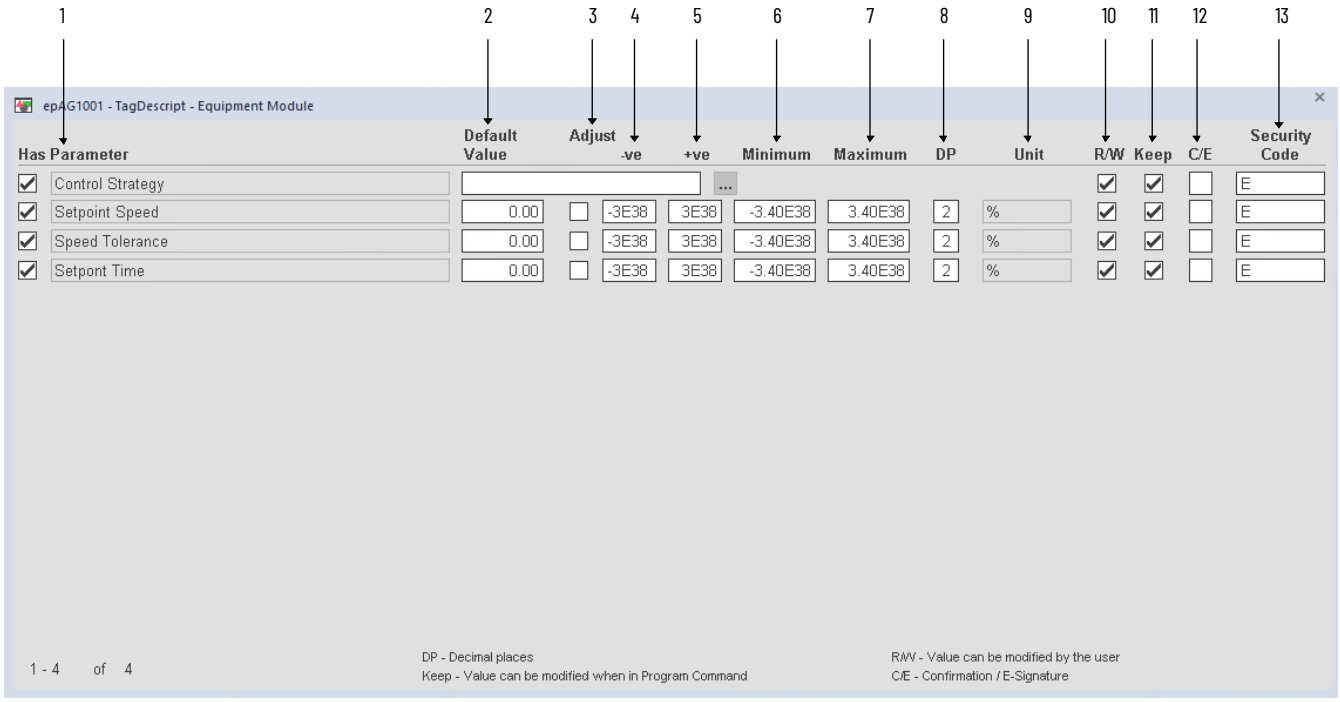
The following page shows the information captured by the snapshot.



The following page shows the default information.

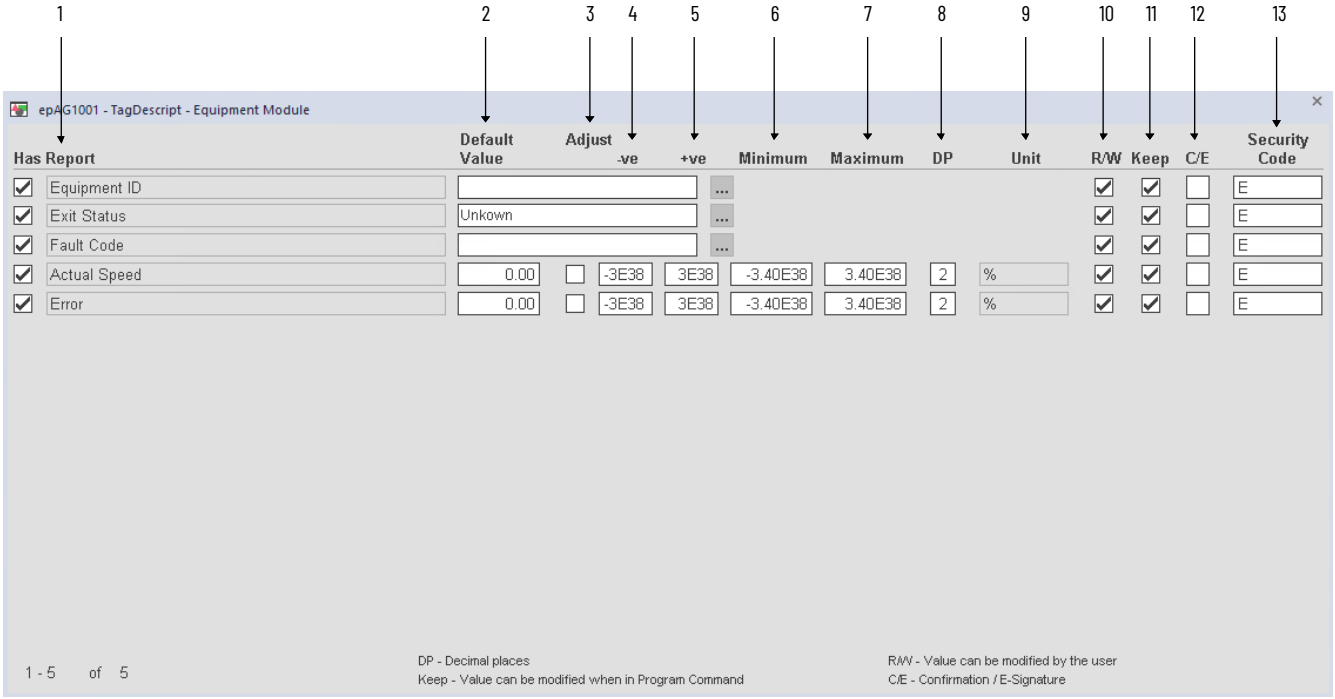


Parameter Configuration



Item	Description
1	Parameter Description
2	Default value of Parameter
3	Allow limit adjust
4	Integer minimum adjust value of parameter.
5	Integer maximum adjust value of parameter.
6	Minimum value of Parameter
7	Maximum value of Parameter
8	Enter the decimal places to display.
9	Engineering unit of Parameter.
10	Parameter value can be modified by the operator when enabled.
11	Parameter value can be modified when in Program Command.
12	Enable Confirmation / E-Signature of Parameter.
13	Assign User Roles Security Level of Parameter.

Report Configuration



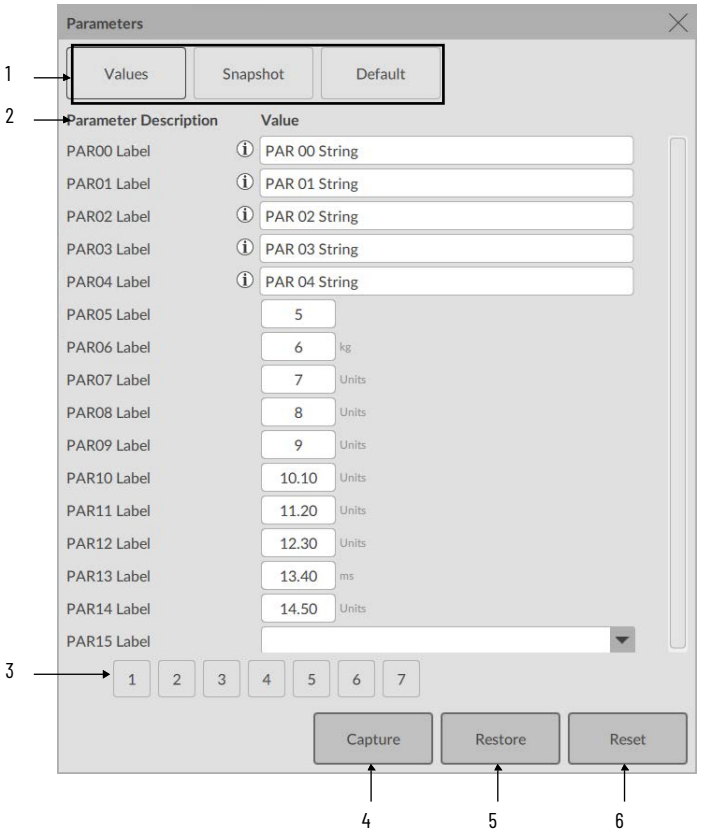
Item	Description
1	Report Description
2	Default value of Report
3	Allow limit adjust
4	Integer minimum adjust value of parameter.
5	Integer maximum adjust value of parameter.
6	Minimum value of Report
7	Maximum value of Report
8	Enter the decimal places to display.
9	Engineering unit of Report.
10	Report value can be modified by the operator when enabled.
11	Report value can be modified when in Program Command.
12	Enable Confirmation / E-Signature of Report.
13	Assign User Roles Security Level of Report.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

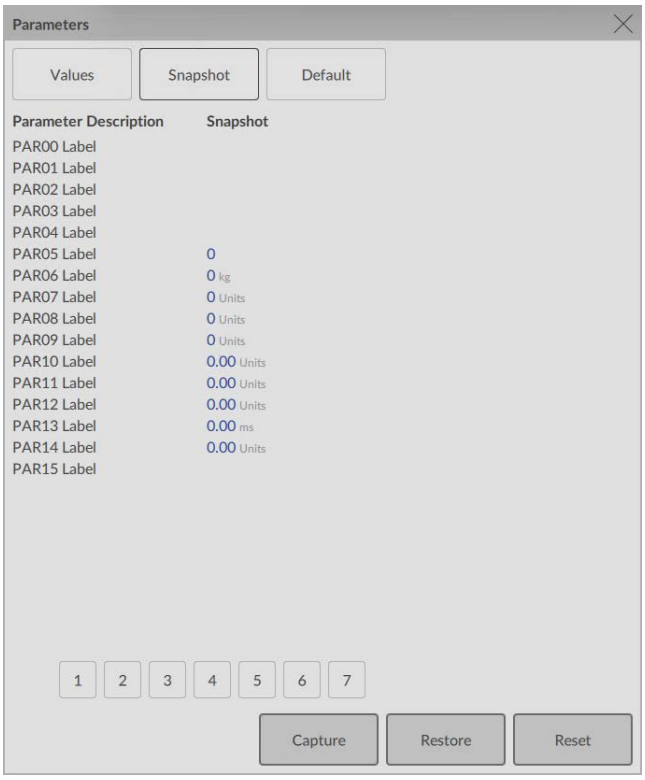
Any feature that is contained in the FactoryTalk® Optix™ faceplates has the same functionality as used in the FactoryTalk® View SE faceplates. See FactoryTalk View SE Faceplates on page 292.

Parameter Display

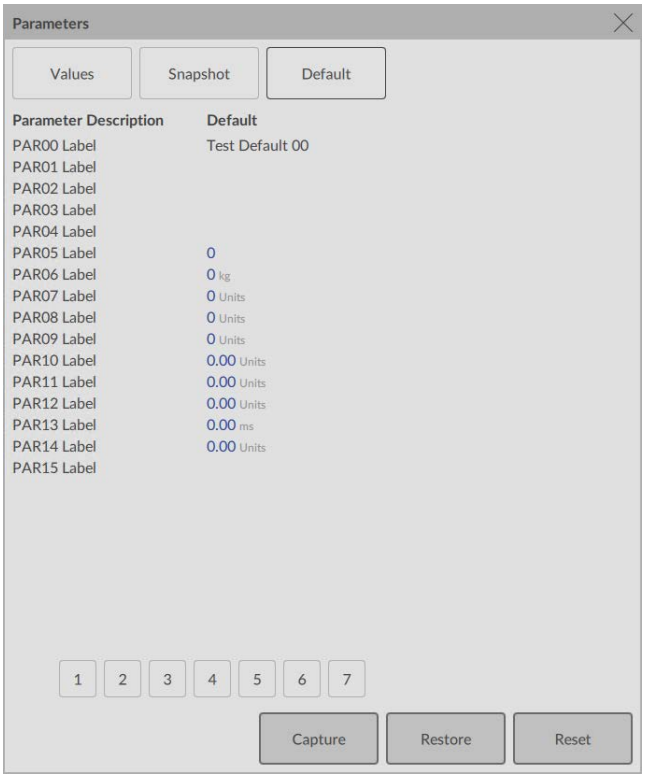


Item	Description
1	Select to navigate between current values, snapshot values, and default values.
2	List of each parameter value and description.
3	Navigation to more parameters (only visible if more than 16 parameter values used).
4	Capture Snapshot. Press to capture current values into snapshot values.
5	Restore Snapshot. Press to restore snapshot values into current values.
6	Reset to defaults. Press to reset current values to default values.

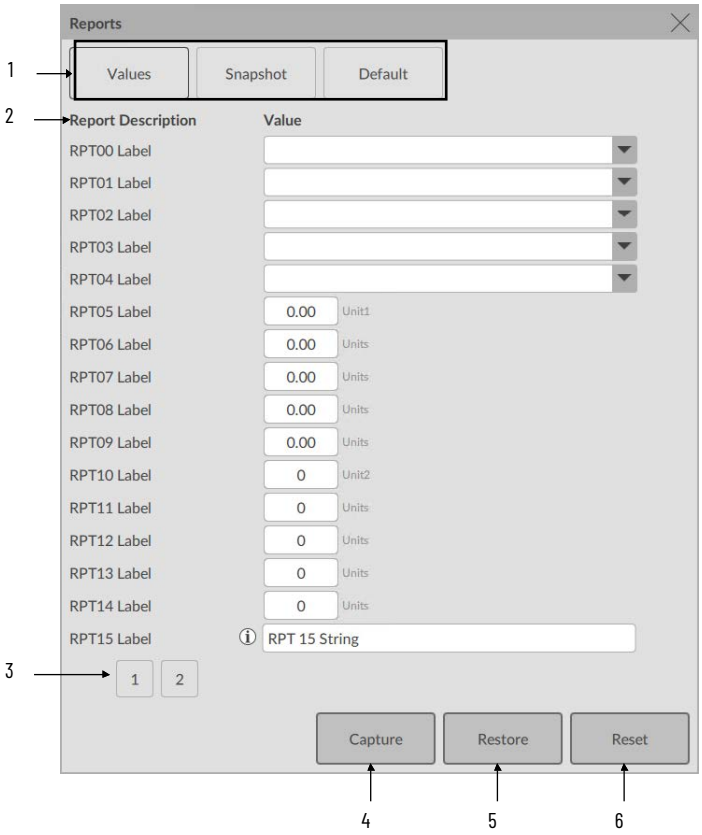
The following display shows the information captured by the snapshot.



The following display shows the default information.

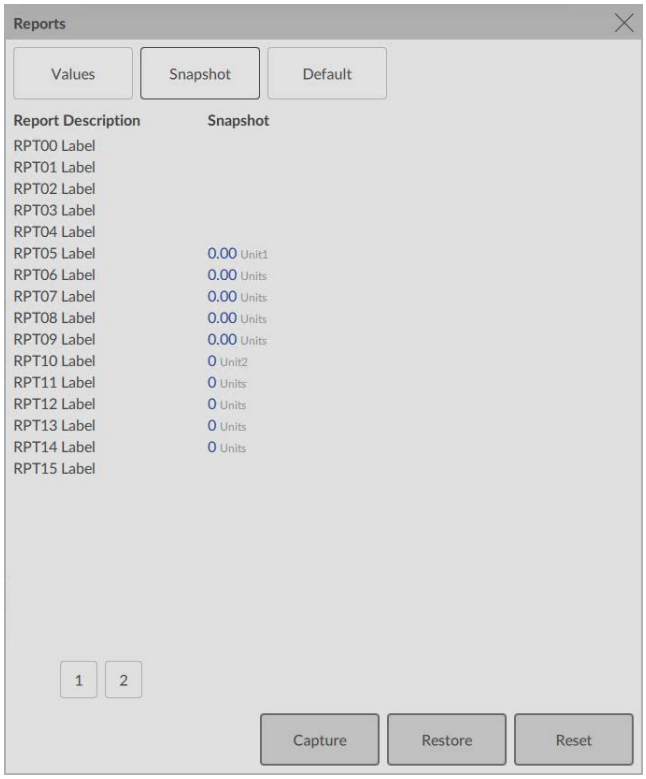


Report Display

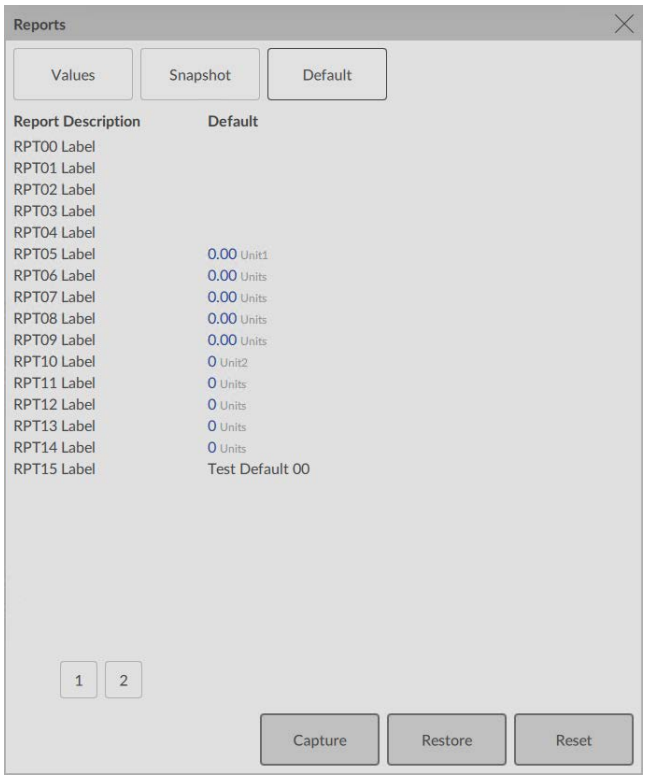


Item	Description
1	Select to navigate between current values, snapshot values, and default values.
2	List of each report value and description.
3	Navigation to more parameters (only visible if more than 16 report values used). See Parameter Display on page 298
4	Capture Snapshot. Press to capture current values into snapshot values.
5	Restore Snapshot. Press to restore snapshot values into current values.
6	Reset to defaults. Press to reset current values to default values.

The following display shows the information captured by the snapshot.



The following display shows the default information.



Notes:

Operator Prompt (raP_Opr_Prompt)

The raP_Opr_Prompt (Operator Prompt) Add-On Instruction is a universal mechanism for operator interaction that can be used within a control scheme. The instruction presents an operator with configurable message or data fields and accepts operator response data and confirmation.

The Sequencer Add-on Instruction instruction also uses the prompt instruction. For more information on the Sequencer instruction, see Rockwell Automation Sequencer Object, Publication PROCES-RM202.

Guidelines

Use a prompt to request input from an operator. The input can be any of the following:

- Acknowledging the prompt
- Viewing and confirming data
- Making a selection
- Entering numeric data
- Entering text data

Do **not** use a prompt in place of an alarm or an alert:

- An alarm, per ANSI/ISA-18.2-2016, is used to notify an operator of an abnormal situation that requires a response
- An alert is used to notify an operator of an abnormal situation that does not require a timely response

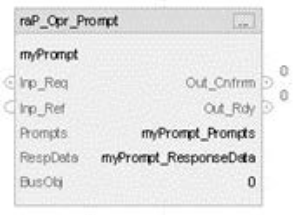
A prompt requires a response, but does not advise of an abnormal situation.

	Normal Operation	Abnormal Situation
Operator Response Not Required	Normal values and status	Alert
Operator Response Required	Prompt (raP_Opr_Prompt)	Alarm

Functional Description

The RespData tag at the bottom of the raP_Opr_Prompt function block lets you define where to store operator responses. This tag stores any operator response as a string in the application. This tag needs to be unique to every instance of the raP_Opr_Prompt instruction.

The optional BusObj tag allows the prompt to participate on the organizational bus. The entry should be a unique bus element in the bus array. With this field populated, an 'operator attention' indicator is propagated up through any organizational tree in which this bus element is assigned.



Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Opr_Prompt_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

Command Sources

The raP_Opr_Prompt Add-On Instruction does not use command sources.

Alarms

The raP_Opr_Prompt Instruction uses the following alarm, which is implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
Alert timeout	Alm_AlertTimeOut	Raised when no response to a posted prompt has been entered within the configured time.

Virtualization

The raP_Opr_Prompt Instruction has no Virtualization capability.

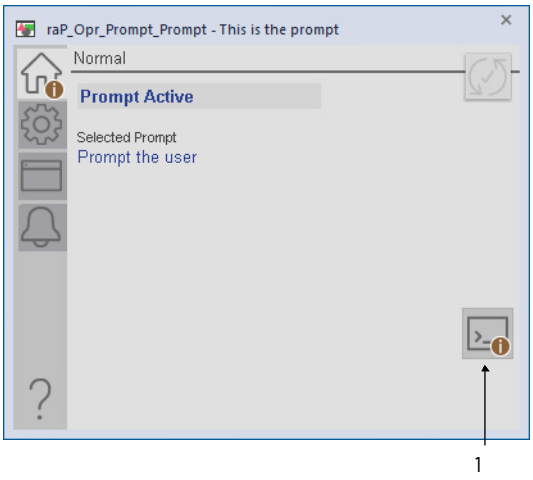
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_Prompt 	raP_5_30_raP_Opr_Prompt_GS 	Standard Prompt graphic symbol

FactoryTalk View SE Faceplates

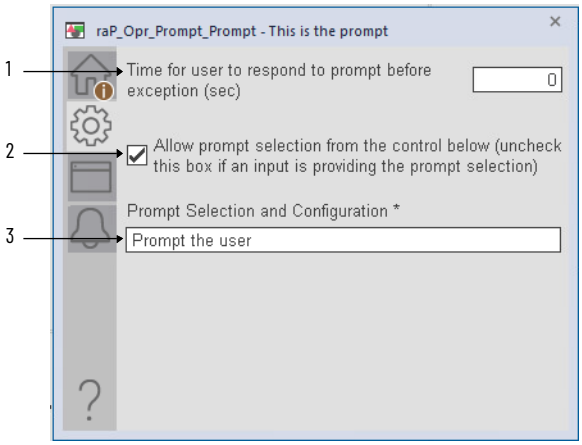
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab



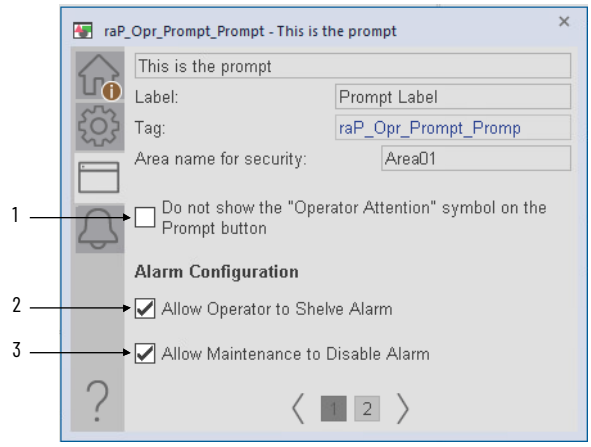
Item	Description
1	Navigate to the Prompt Response display

Engineering Tab

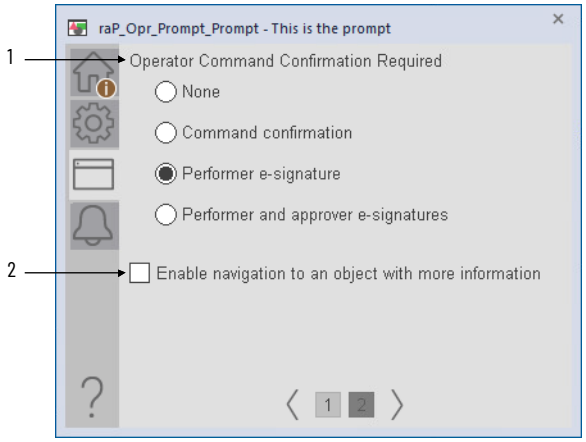


Item	Description
1	Configure the maximum time without a user response before an exception occurs
2	Allow prompt selection and configuration from this display
3	Navigates to the prompt selection and configuration display

HMI Tab



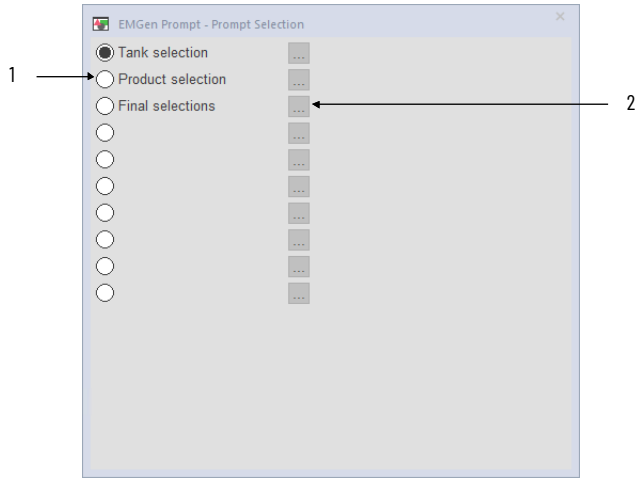
Item	Description
1	Hides the operator attention symbol on the Prompt Response button when there is an active prompt
2	Select to allow Operator to shelve alarm
3	Select to allow Maintenance to disable alarm



Item	Description
1	Select an option for Operator Command Confirmation Requirements
2	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the <backing tag>.@Library and <backing tag>.@Instruction extended tag properties to display the objects faceplate.

Selection

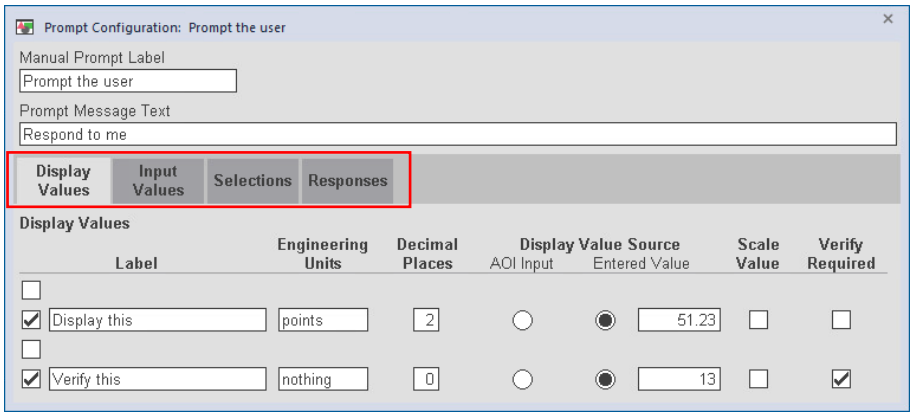
The Prompt Selection display provides access to the configuration dialog box for a given prompt configuration in the prompts array by clicking the corresponding browse button.



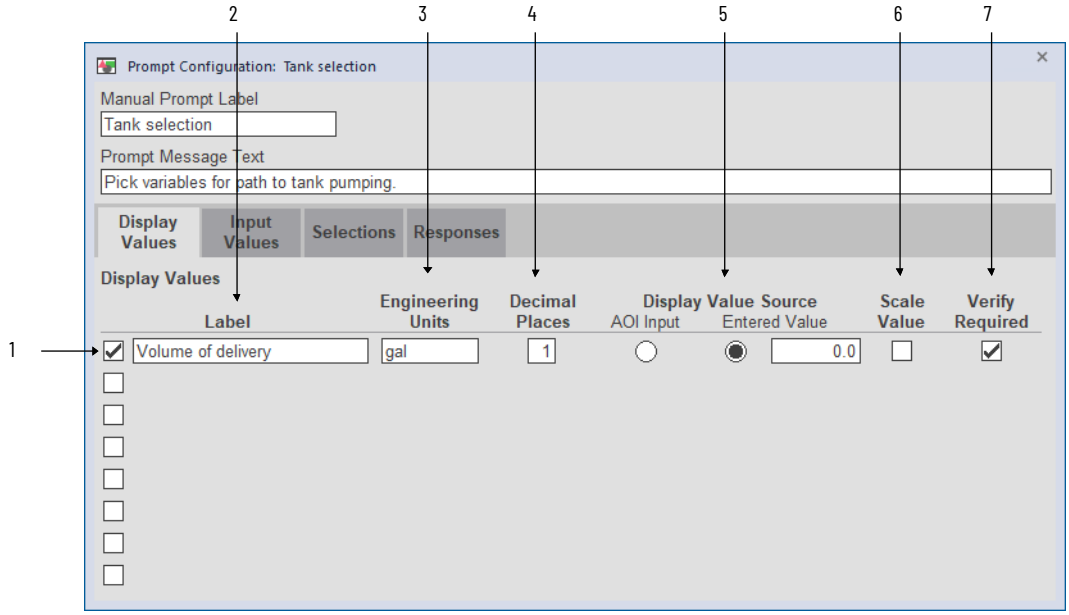
Item	Description
1	Select the radio button to select a prompt.
2	Select to open the configuration faceplate.

Configuration

The Prompt Configuration dialog box has four sections to configure a prompt. The sections are Display Values, Input Values, Selection Options, and Response Prompts. Each of the four sections has the ability to add up to eight individually labeled items.

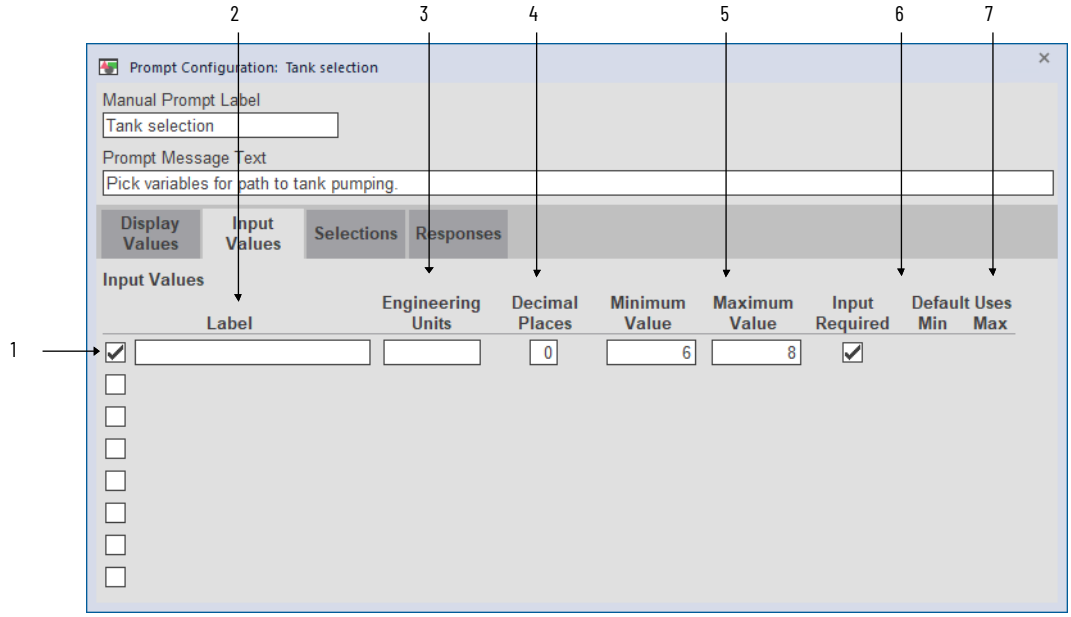


Display Values



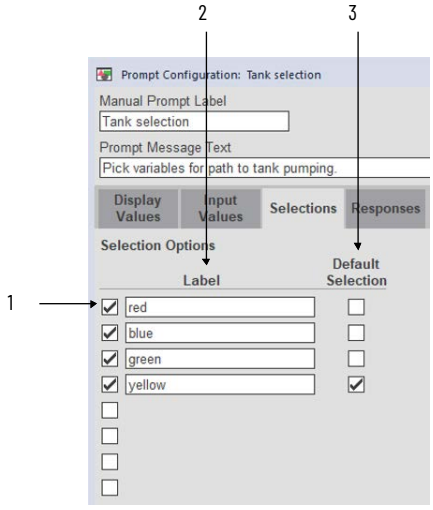
Item	Description
1	Select to enable a numeric display field.
2	Enter a label.
3	Enter an engineering unit.
4	Enter the decimal places to display.
5	Select to either display a value from the prompt 'AOI Input' or the value that you enter in the box that appears.
6	Select to scale the value by the entered value and the Inp_ScalePct.
7	Select to require the operator to verify the displayed value.

Input Values



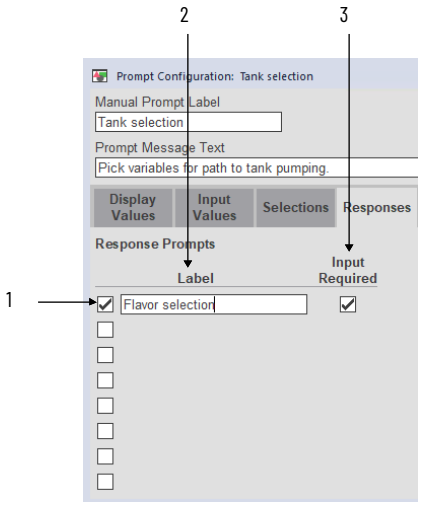
Item	Description
1	Select to enable a numeric input
2	Enter a label for the input value.
3	Enter an engineering unit.
4	Enter the decimal places to display.
5	Enter a minimum value for the entry
6	Enter a maximum value for the entry.
7	Select to require an operator to enter a value.
8	If an input is not required, click Minimum or Maximum to be used for the entry.

Selection Options



Item	Description
1	Select to enable a label text box.
2	Enter a label for the selection option.
3	Select to designate a selection as the default.

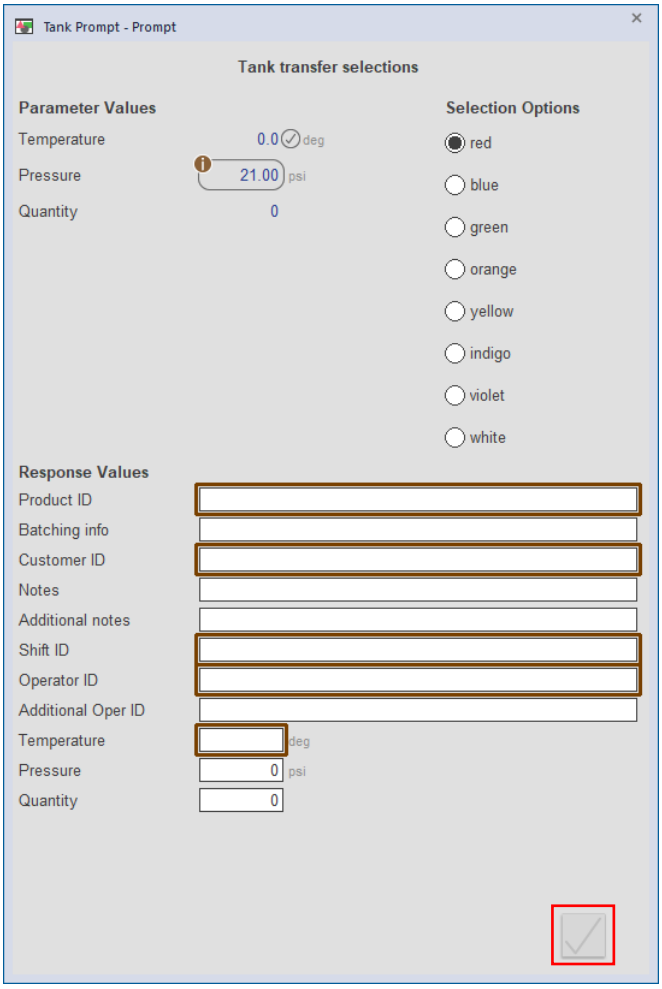
Response Prompts

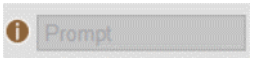


Item	Description
1	Select to enable a response prompt.
2	Enter a label for the response prompt.
3	Select to require an input.

Response

This faceplate lets the operator review and record data based on the prompt. All values are configured on the prompt configuration faceplate. The operator selects the checkbox to continue.



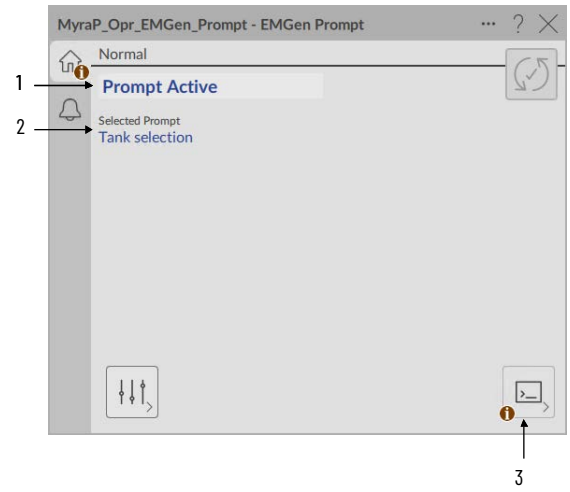
Graphic Symbol Name	FactoryTalk View SE Graphic Symbol	Description
GO_nav_PromptResponse		Standalone prompt button that assumes the raP_Opr_Prompt instruction is present, and the button is always visible. The Prompt instruction controls the enabled state and alert indicator visibility of the button.
GO_nav_PromptResponseTxt		Prompt display indicator for use on faceplates and displays for objects that possibly do not have a prompt instruction.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

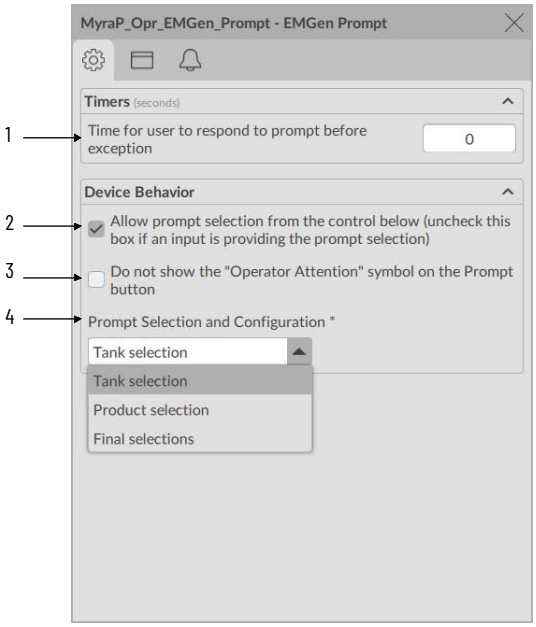
Any feature that is contained in the FactoryTalk® Optix™ faceplates has the same functionality as used in the FactoryTalk® View SE faceplates. See FactoryTalk View SE Faceplates on page 305.

Operator Tab



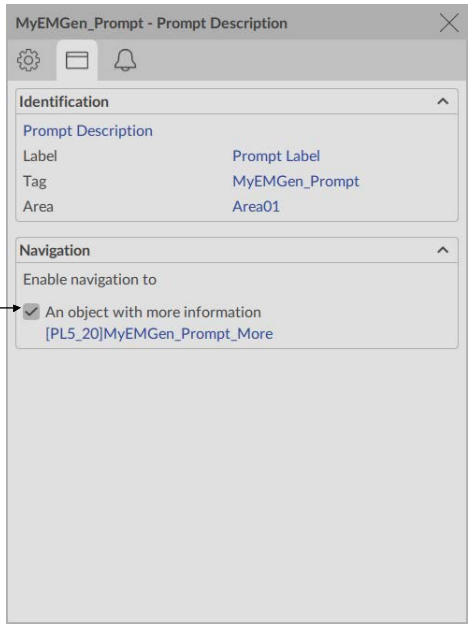
Item	Description
1	Displays the Prompt mode status.
2	Displays the Prompt selection.
3	Navigate to the Prompt Response display.

Advanced Engineering Tab



Item	Description
1	Configure the maximum time without a user response before an exception occurs.
2	Allow prompt selection and configuration from this display.
3	Hides the operator attention symbol on the Prompt Response button when there is an active prompt.
4	Select a prompt from Prompt Selection drop down menu.

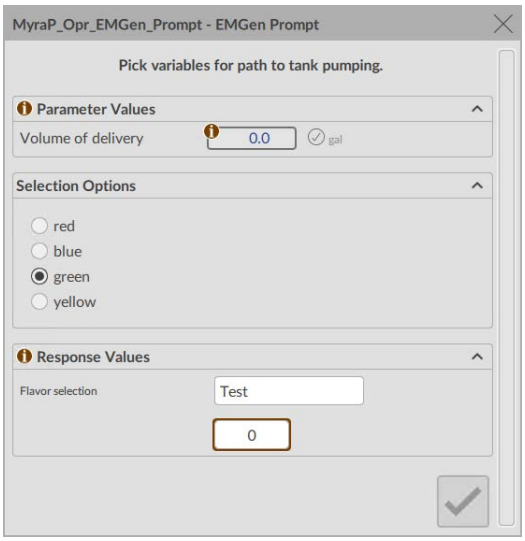
Advanced HMI Configuration Tab



Item	Description
1	Select to enable navigation to an object with more information. You configure the tag name of the object that you want to navigate to in the extended tag property "Cfg_HasMoreObj.@Navigation". It uses the .@Library and .@Instruction extended tag properties to display the objects faceplate.

Response

This faceplate lets the operator review and record data based on the prompt. All values are configured on the prompt configuration faceplate in FactoryTalk View SE. The operator selects the checkbox to continue.



Logix Diagnostic Objects

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Logix Change Detector (raP_Dvc_LgxChangeDet)

The raP_Dvc_LgxChangeDet (Logix Change Detector) Add-On Instruction monitors another Logix controller on the network and checks for changes that impact operation. Changes that can be monitored include downloads, online edits, I/O forcing, and controller mode changes.

No visualization elements are supplied with the raP_Dvc_LgxChangeDet instruction.

Guidelines

Use this instruction if you want to monitor a Logix controller for changes, to be sure that the correct application is being run for regulatory, quality, or security reasons.

Do not use this instruction in these situations:

- You have only one Logix controller. The raP_Dvc_LgxChangeDet instruction is intended to be run in a controller other than the one being monitored. Although the raP_Dvc_LgxChangeDet instruction can be configured to monitor the controller in which it is running, because it runs in controller logic, it cannot detect when the controller in which it is running is placed in Program mode.
- You have software, such as FactoryTalk® AssetCentre that monitors controllers on a secured network. This software provides much more extensive change tracking and auditing than the raP_Dvc_LgxChangeDet Add-On Instruction.

Functional Description

The raP_Dvc_LgxChangeDet instruction includes a source protected Add-On Instruction for use with Studio 5000 Logix Designer® software, version 33 or later, and Logix controllers. This instruction is intended to be used in one Logix controller to monitor another controller for changes.

Although this instruction must be executed in a Logix controller with firmware revision 33 or later, it can monitor controllers running firmware revision 12 or later.

The raP_Dvc_LgxChangeDet instruction monitors a Logix controller for the following types of changes:

- New entries being made in the change log, such as the following:
  - Modify, insert, or delete logic in Run or Program mode
  - Accept, assemble, or cancel edits
  - Enable, disable, or remove forces
  - Reconfigure a module

  - Change an output list
  - Send the 'Set Attribute' MSG or 'SSV' to a controller object class or instance
  - Send the 'Set Attribute List' MSG to a controller object class or instance
  - Send the 'Set Attribute All' MSG to a controller object class or instance
  - Apply attributes to a controller object class or instance
  - Create, delete, or reset a controller object instance
- Download of a different application
- Partially import into an application
- Download of an application without logic changes (but saved configuration data that has changed)
- Download of an application that contains offline edits
- Restore an application from an external drive source, such as a Secure Digital (SD) card

This instruction also reports the following:

- Controller/application 'check' value for change detection
- Date and time on the controller clock (YYYY-MM-DD hh:mm:ss)
- Day of the week based on the controller date
- Controller keyswitch position and mode
- Major and minor fault indications

The raP_Dvc_LgxChangeDet instruction is provided as a rung import for installation. Importing this rung into your ladder diagram routine:

- Imports the Add-On Instruction definition
- Create an instruction instance.
- Creates and completes all required tags and data structures for the instruction.

IMPORTANT	Once the rung is imported, and before downloading and running the application, set the path in each of the referenced Message structures to point to the Logix controller to be monitored.

The interval at which this instruction checks for changes and updates its status is configurable, from 1...60 seconds.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Dvc_LgxChangeDet_5.30.**00**_RUNG.L5X rung import file must be imported into the controller project for the controller that is performing the monitoring. It is not necessary to add any logic to the controller being monitored. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

There are no visualization files because the raP_Dvc_LgxChangeDet object does not use Graphic Symbols or Faceplates.

Operations

Command Sources

The raP_Dvc_LgxChangeDet instruction has no commands or outputs that are intended to control equipment and therefore does not have any command sources.

Alarms

The raP_Dvc_LgxChangeDet Instruction uses the following alarm, which is implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
Change detected	Alm_ChangeDetected	Raised when a change in controller operation such as a download or online edit has been detected.

Virtualization

The raP_Dvc_LgxChangeDet Add-On Instruction does not have a Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Dvc_LgxChangeDet instruction must always be scanned true. In relay ladder logic, the raP_Dvc_LgxChangeDet instruction must be by itself on an unconditional rung. If the Rung Import provided with the Rockwell Automation® Library is used to install this instruction, the proper rung is created for you.
Powerup (prescan, first scan)	On Prescan, any commands that are received before First Scan are discarded. The update timer and internal polling status are reset. On first scan, the Change Detected internal status latch is cleared.
Postscan (SFC transition)	No SFC Postscan logic is provided.

See to the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010, for more information.

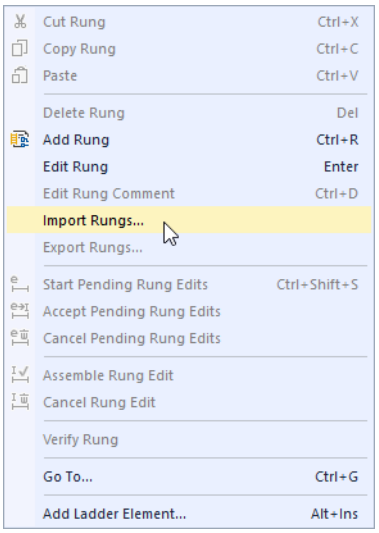
Programming Example

The raP_Dvc_LgxChangeDet instruction is provided fully configured as a rung import; so little programming is required for the instruction to be used. This programming example shows how the rung import is used to instantiate the raP_Dvc_LgxChangeDet instruction.

Since the raP_Dvc_LgxChangeDet instruction is a rung import, it must be created in a Ladder Diagram routine. By default, raP_Dvc_LgxChangeDet checks controllers for changes only every 5 seconds, so the ladder routine does not need to run in a fast periodic task.

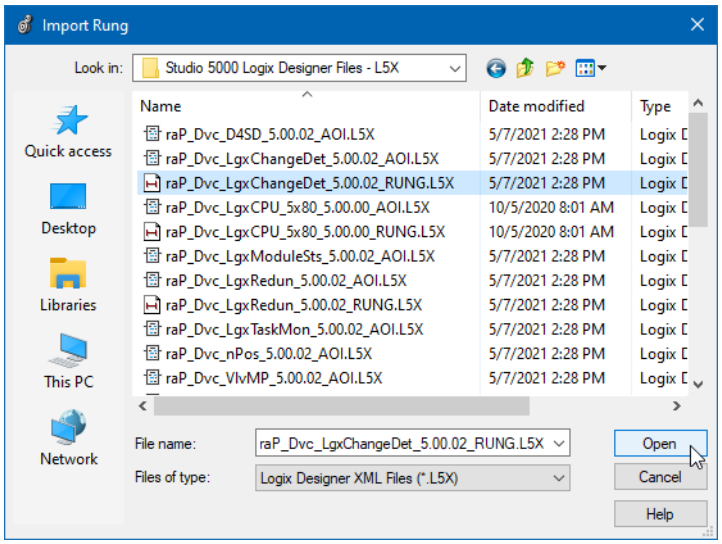
The following steps describe how you instantiate raP_Dvc_LgxChangeDet in your routine.

1. In your ladder routine, right-click where to insert the rungs and select Import Rungs.

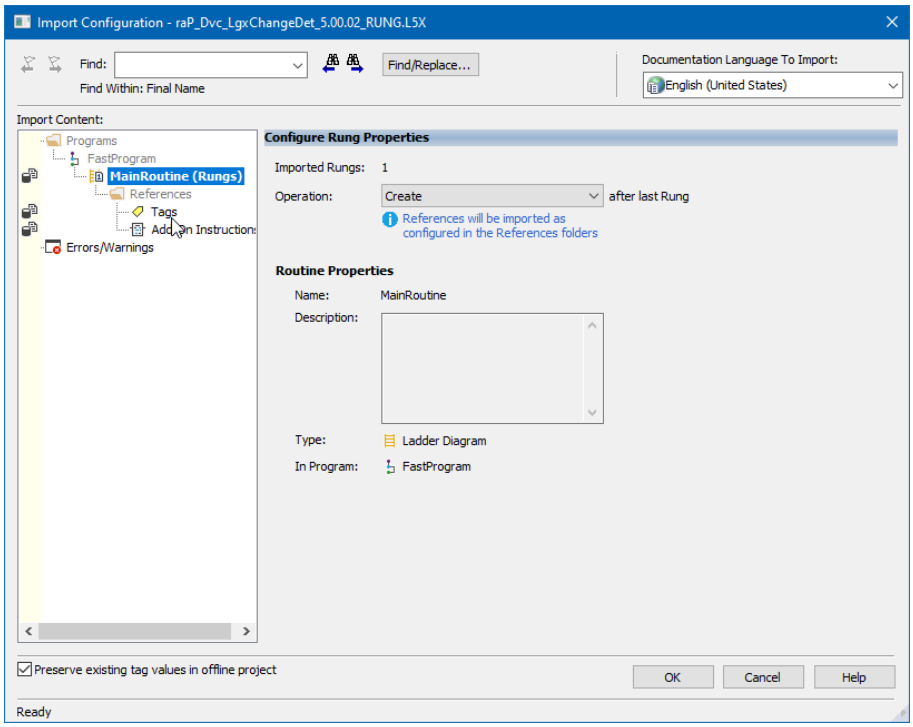


   The Import Rungs dialog box appears.

2. Select the raP_Dvc_LgxChangeDet rung import file that is named in Required Files on page 314.
3. Select Open.



The Import Configuration dialog box appears.



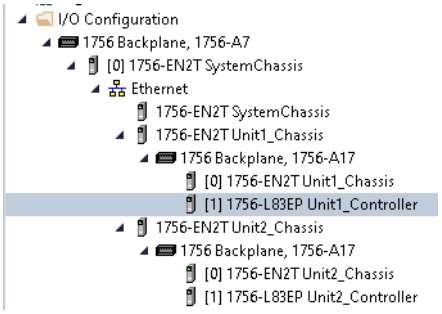
4. Rename the tags being imported to incorporate the name of the controller being monitored.

   One controller can monitor several others. Adding the controller name to the tag makes it easier to track the individual instances when monitoring multiple controllers.

5. Select OK.

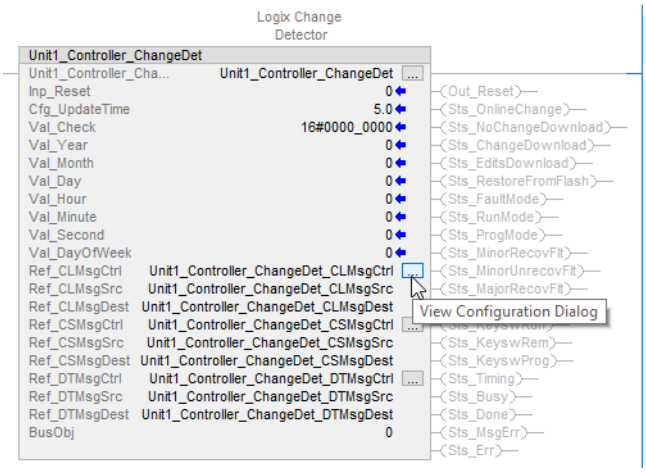
6. To point to the controller being monitored for changes, change the path in each of the MSG control tags.

   If you create a link to the controller in the I/O tree configuration, enter the name that is assigned to that controller.



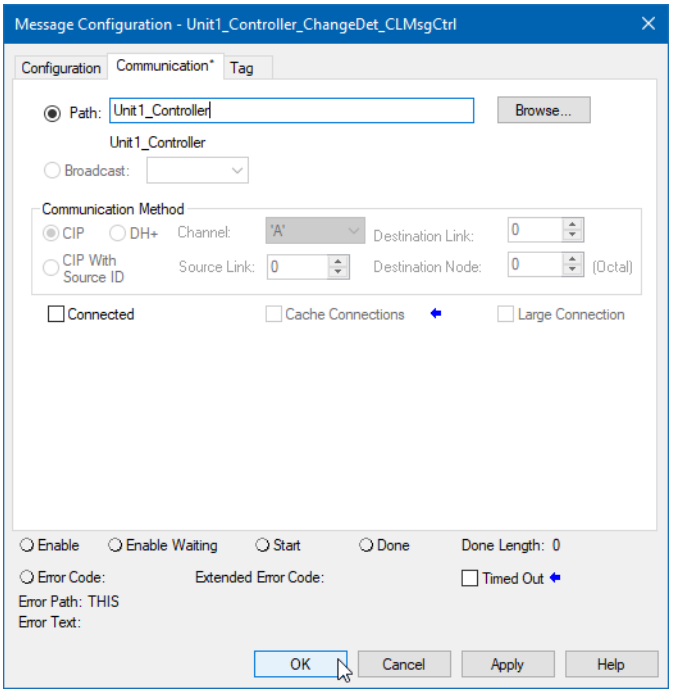
7. Complete the following steps for each of the three MSG control tags.

a. Select the ellipsis next to the MSG control tag.



The Message Configuration dialog box appears.

b. Select the Communication tab and change the path to the controller link created in the I/O tree.

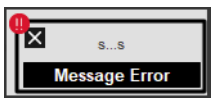
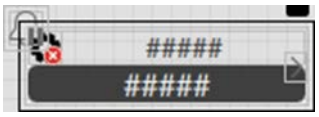


c. Select OK.

8. Place the controller in RUN mode.

Status bits on the raP_Dvc_LgxChangeDet instruction indicate changes that are made to the monitored controller. Set Cmd_AckAll to 1 to clear the latched-in detections.

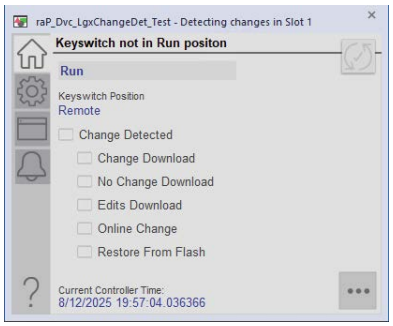
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_LgxChangeDet 	raP_5_30_raP_Dvc_LgxChangeDet_GS 	Standard raP_Dvc_LgxChangeDet graphic symbol

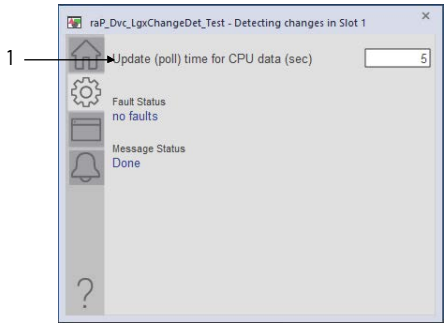
FactoryTalk View SE Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab

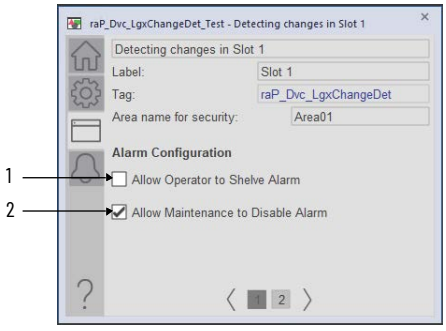


Engineering Tab



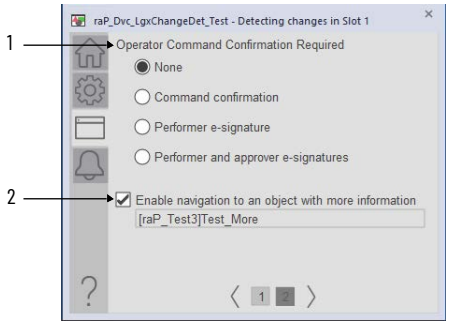
Item	Description
1	Enter the update time for CPU data.

HMI Configuration Tab 1



Item	Description
1	Select to allow Operator to shelve the alarm.
2	Select to allow Maintenance to disable the alarm.

HMI Configuration Tab 2



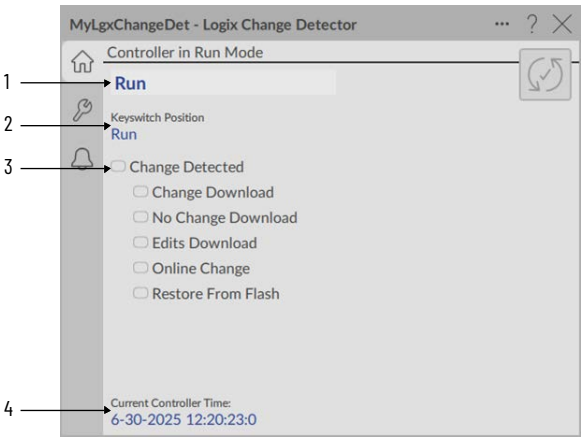
Item	Description
1	Select the type of confirmation required for Operator commands.
2	Select to enable navigation to an object with more information (Cfg_HasMoreObj is set to true.) This can be configured to navigate to an object backing tag or a UDT tag that has Instruction and Library defined.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

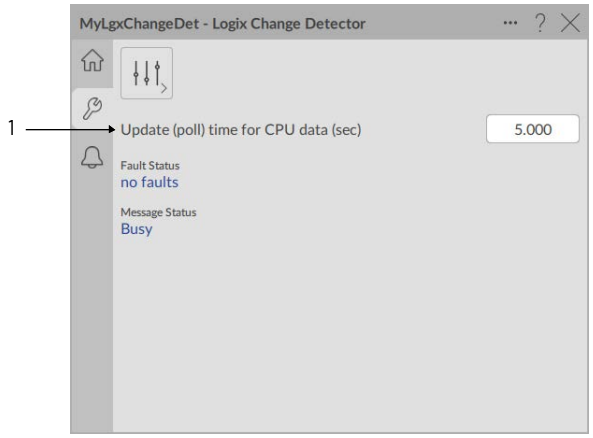
Any feature that is contained in the FactoryTalk® Optix™ faceplates has the same functionality as used in the FactoryTalk View SE faceplates. See FactoryTalk View SE Faceplates on page 319 for descriptions of the features.

Operator Tab



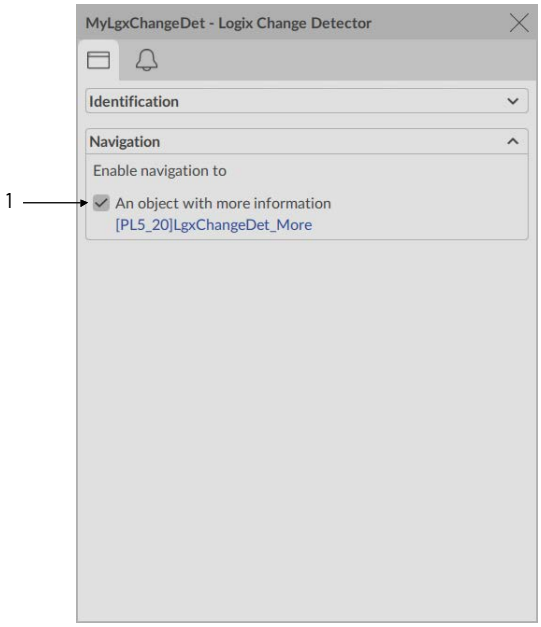
Item	Description
1	Displays the controller status mode.
2	Displays the keyswitch position status
3	Displays the change detected status.
4	Displays the current controller time.

Maintenance Tab



Item	Description
1	Enter the update time for CPU data.

Advanced HMI Configuration Tab - Navigation



Item	Description
1	Select to enable navigation to an object with more information (Cfg_HasMoreObj is set to true.) This can be configured to navigate to an object backing tag or a UDT tag that has Instruction and Library defined.

Logix Controller CPU Utilization (raP_Dvc_LgxCPU_5x80)

The raP_Dvc_LgxCPU_5x80 (Logix Controller CPU Utilization) Add-On Instruction monitors a Logix controller, and provides information on controller CPU utilization, communication usage, and other information. Data that is provided by the raP_Dvc_LgxCPU_5x80 instruction is useful to diagnose communication or control responsiveness issues and in tuning the performance of control tasks for optimum controller performance.

The raP_Dvc_LgxCPU_5x80 instruction can be loaded as part of a control application and disabled (default) until needed. The instruction can also be enabled at a slow update rate for general controller monitoring. The update rate can be increased, if necessary, as directed by a Rockwell Automation Technical Support representative to help diagnose controller performance issues.

The global object and faceplate in the following image are examples of the HMI that is provided with this library object.

Guidelines

Use this instruction in these situations:

- Monitor general controller resource utilization:
  - Processor utilization
  - Communication capacity
  - Networking performance and connection usage
- Gather data to help resolve a specific issue under the direction of a Rockwell Automation Technical Support representative
- Tune the periods or priorities of multiple tasks in a controller to optimize control and observe how changes in task configuration affect CPU and other resource usage in the controller
- For use with CompactLogix® 5380 Controllers and ControlLogix 5580 Controllers. Firmware / software must be version 33 and later.

Do not use this instruction at a high update rate on a continuing basis. The raP_Dvc_LgxCPU_5x80 instruction increases the communication load on the controller when it is polling for performance data. At high update rates, the resource load that the raP_Dvc_LgxCPU_5x80 instruction polling generates can affect control performance, especially if you already have a fully loaded controller.

Functional Description

The raP_Dvc_LgxCPU_5x80 instruction collects and summarizes various data from the Logix controller that is being monitored. This information includes the following:

- Processor Identity information:
  - Catalog number and description
  - Major and minor firmware revision numbers
- Communication Responsiveness information:
  - CPU% used for responding to communication requests
  - Optimized Packets that are used for responding to communication requests

IMPORTANT	The raP_Dvc_LgxCPU_5x80 instruction does not support SoftLogix™ 5800 or RSLogix™ Emulate 5000 controllers.

- CPU utilization (%):
  - Continuous task (or unused CPU, if no continuous task)
  - Periodic and Event tasks
  - Responding to communication requests (such as from HMI)
  - System (I/O scan, timer updates, everything else)
- Communication connection usage:
  - Total connections available
  - Connections that are used for each of several classes of communication
  - Unconnected buffers and cached messages
- I/O Forcing status
- Controller minor faults

The items that are listed previously are displayed on several faceplate tabs, with summary information on the main (home) tab.

IMPORTANT	We recommend that you access the raP_Dvc_LgxCPU_5x80 faceplate when you contact Rockwell Automation Technical Support. The information on the Operator (home) tab is often requested when you call. You also need your Studio 5000 Logix Designer software serial number or other license or support contract information. The Maintenance tab has a space for you to record this information for reference.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

For use with Logix 5x80 controllers, one instance of raP_Dvc_LgxCPU_5x80_5.30.**00**_RUNG.L5X must be imported into the controller. It is recommended to only use the rung import when configuring the AOI in a project. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Operations

Command Sources

The raP_Dvc_LgxCPU_5x80 instruction is not intended to control equipment and therefore does not have any command sources. However, there are two program commands and two maintenance commands available to enable and disable collection of data (PCmd_Enable, PCmd_Disable, MCmd_Enable, MCmd_Disable). The maintenance commands are only available via the HMI faceplate.

Alarms

The raP_Dvc_LgxCPU_5x80 Add-On Instruction does not provide any alarms. If an alarm is required, define the output status to be alarmed as a Logix Tag Based Alarm.

Virtualization

The raP_Dvc_LgxCPU_5x80 Add-On Instruction does not have a Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	The raP_Dvc_LgxCPU_5x80 instruction has no EnableInFalse logic and does nothing on a false rung. Data that are associated with the instruction are left in their last state.
Powerup (prescan, first scan)	Logic is sure that the window time is sent to the controller when it transitions to Run mode. Previously active polling (before power down or transition to Program mode) is canceled. High-water data that is stored in the instruction (not built in to the controller status registers) are cleared.
Postscan (SFC transition)	No SFC Postscan logic is provided.

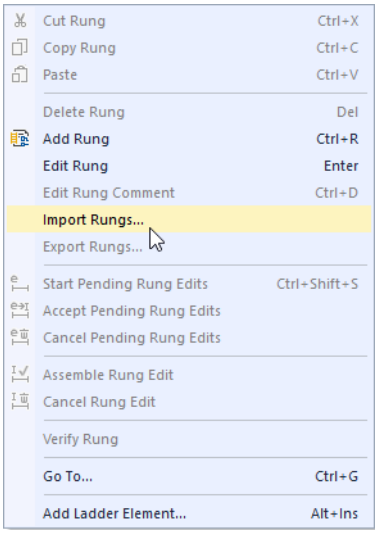
See to the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010, for more information.

Programming Example

The raP_Dvc_LgxCPU_5x80 instruction is provided fully configured as a rung import; therefore, little programming is required for the instruction to be used. This programming example shows how the rung import is used to instantiate the raP_Dvc_LgxCPU_5x80 instruction.

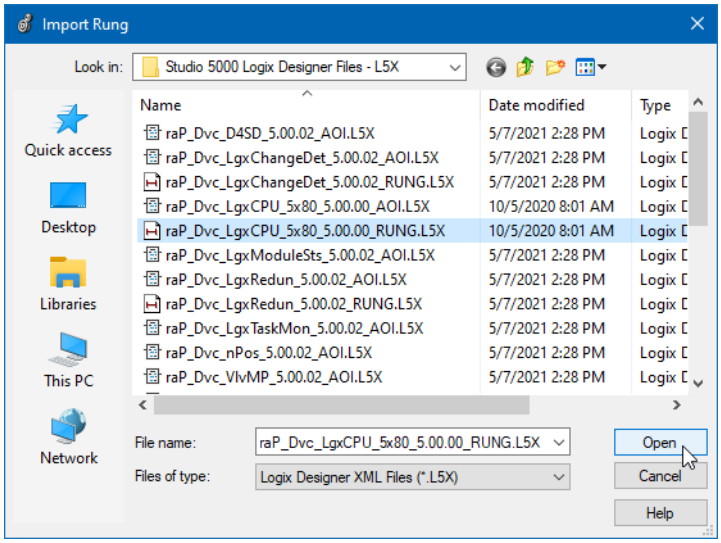
Because raP_Dvc_LgxCPU_5x80 is a rung import, it must be created in a ladder diagram routine. The following steps describe how to instantiate raP_Dvc_LgxCPU_5x80 in your routine.

1. In your ladder routine, right-click where to insert the rungs and select Import Rungs.



The Import Rungs dialog box appears.

2. Select the appropriate raP_Dvc_LgxCPU_5x80 rung import file that is named in Required Files on page 323.



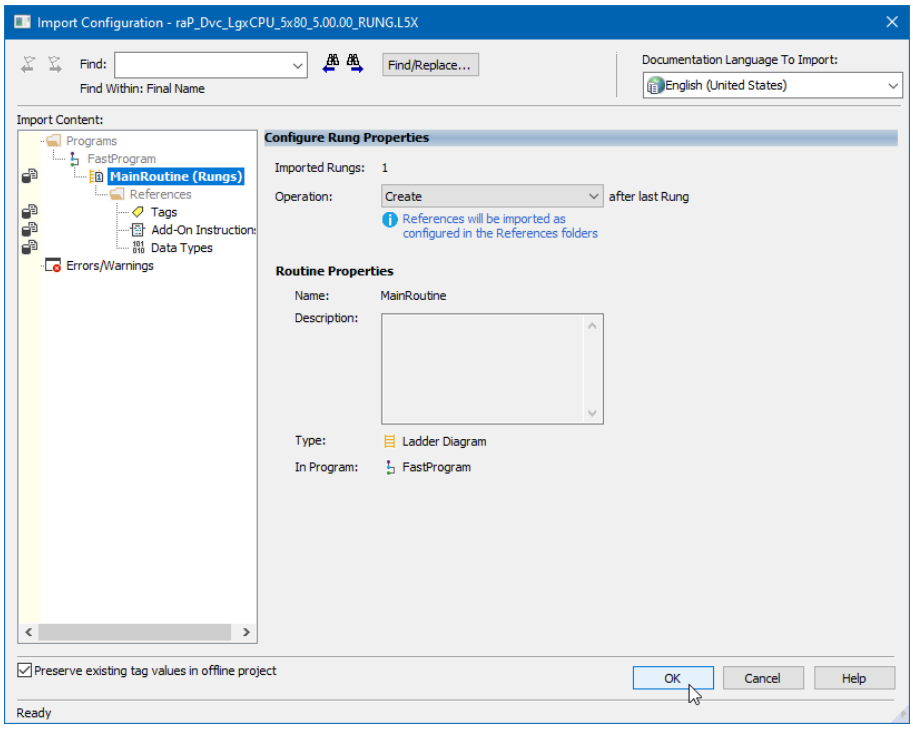
3. Select Open.

   The Import Configuration dialog box appears.

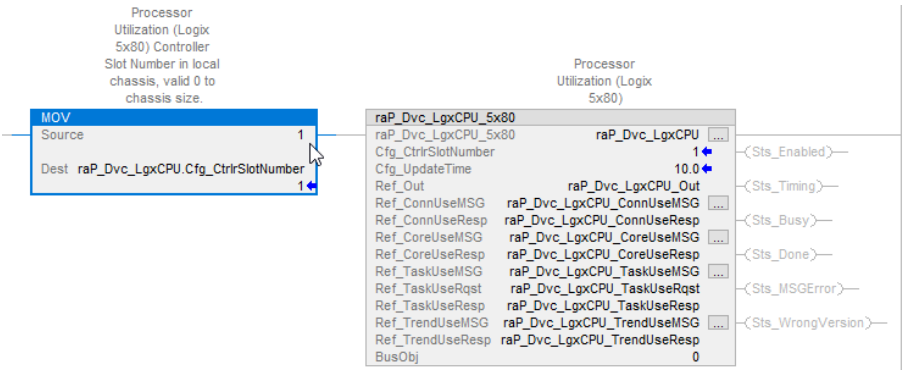
IMPORTANT Do not change tag names in the Import Configuration.

There must be one instance only of the raP_Dvc_LgxCPU_5x80 instruction in any controller project.

4. To create the instance of raP_Dvc_LgxCPU_5x80, select OK.



5. Set the controller slot number in the Source of the MOV.



   Set this value before putting the controller into Run mode. If the value is changed, it requires a transition from Program to Run on the controller for the new value to take effect.

6. Select the Finalize All Edits in Program icon.
7. To finalize all edits, Select Yes.

Graphic Symbols

A Graphic Symbol (global object) is created once and can be referenced multiple times on multiple displays in an application. When changes are made to the original (base) object, the instantiated copies (reference objects) are automatically updated. Use of graphic symbols, with tag structures in the ControlLogix® system, aid consistency and save engineering time.

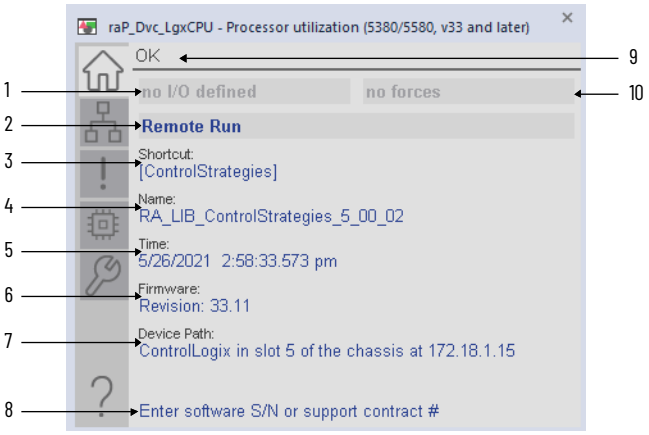
FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_LgxCPU 	raP_5_30_raP_Dvc_LgxCPU_5x80_GS 	This global object is used to view controller CPU utilization for Logix 5x80 controllers at firmware version 33 or later.

FactoryTalk View SE Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab

The Faceplate initially opens to the Operator (Home) tab. From here, an operator can monitor the device status and manually operate the device when it is in Operator command source.

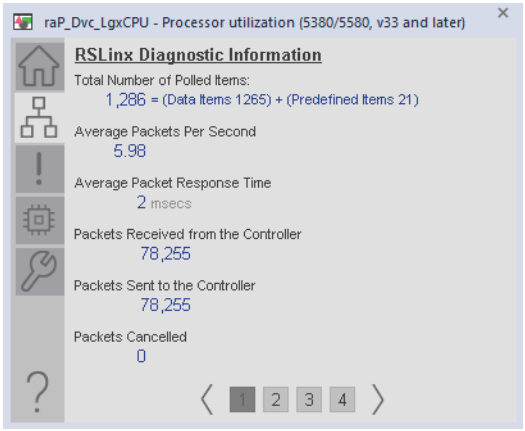


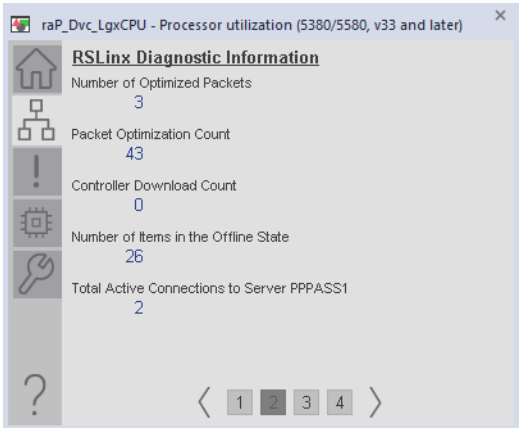
Item	Description
1	I/O communication status
2	Current controller mode
3	Device shortcut
4	Processor name defined in RSLogix 5000®
5	Current date and time
6	Current firmware revision
7	Path from the HMI server to the device
8	Serial number or support agreement. This number is used when contacting Rockwell Automation technical support.
9	Controller OK indicator
10	I/O forcing status indicator

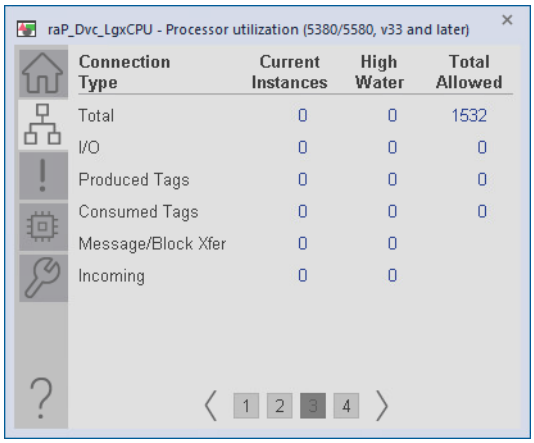
Communication Tab

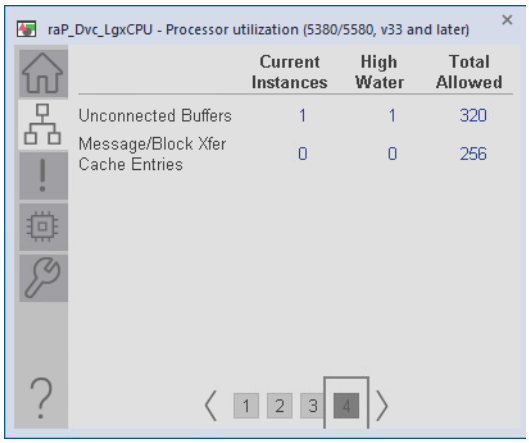
The pages in the Communication tab display the following information:

- Nested bar graph and numeric displays that show the approximate percent CPU available for responding to communication requests from the HMI (outer bar). The outer bar graph changes color from green to yellow when CPU availability for communication is low.
- The approximate percent CPU that is actually being used for responding to communication requests (inner bar). The inner bar graph changes color from blue to red when nearly all CPU availability for communication is being used.
- The count of RSLinx® optimized packets that are currently used.
- The high-water value of optimized packets that are used.
- The largest optimized packet instance number that is used in the controller.
- Diagnostic counters for the FactoryTalk® Linx software driver that is being used by the HMI to communicate with the controller.
- The number of connections that are being used, the highest number that is used, and the total available connections for several types of data transfers.
- Data also includes statistics for message instructions that are using unconnected buffers and message cache entries.



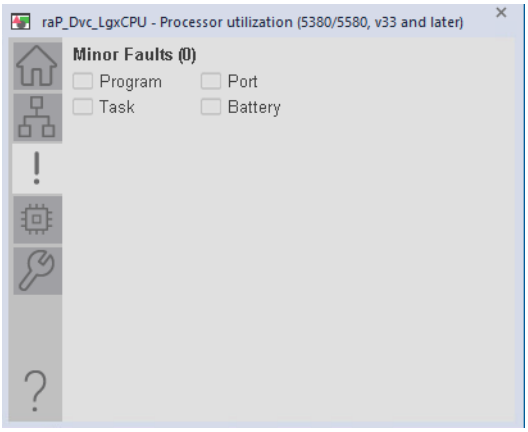






Faults Tab

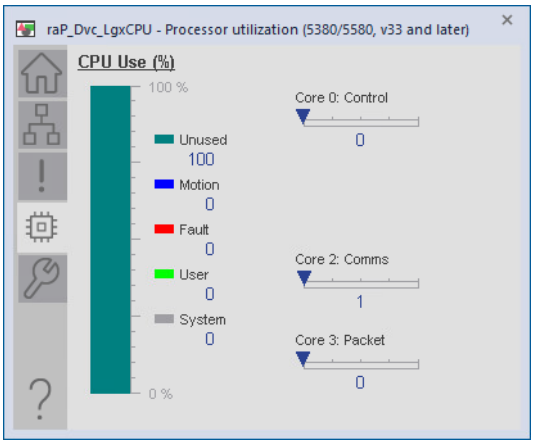
The Faults tab contains the list of minor faults and the fault count. There is an indicator to display the status of each fault. A blue indicator box shows that the fault is active.



Performance Tab

The Performance tab shows the approximate CPU percentage that is used by each of the major activities for the controller. If there is a continuous task running in the controller, the top segment of the bar graph shows the CPU used by the continuous task. If there is no continuous task, the top segment shows the percentage CPU free (unused). The CPU percentages do not necessarily add up to 100% because of the variability between execution cycles of the listed tasks and rounding errors.

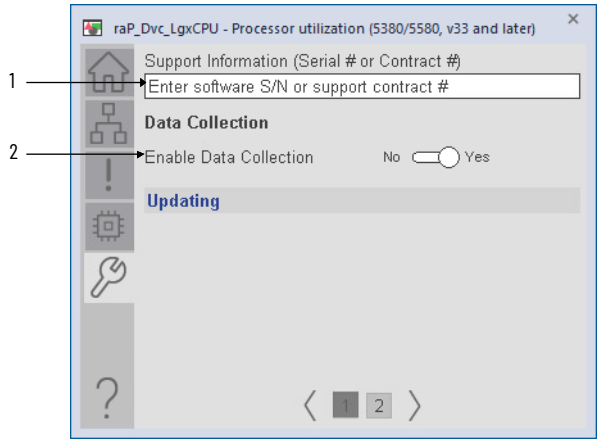
The raP_Dvc_LgxCPU_5x80 instruction is used with a multi-core controller. The bar graph on the left represents the CPU percentage that is used of the control core (Core 0).



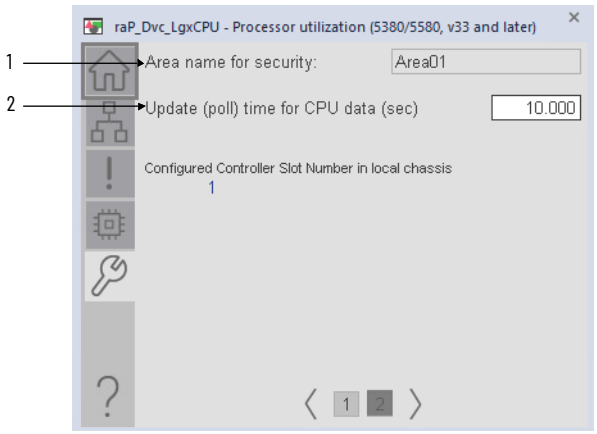
Maintenance Tab

The Maintenance tab shows the following information:

- An indicator to show whether data collection (polling) is enabled or disabled
- An indicator to show when the instruction is waiting before the next data collection (poll) and when a poll is in progress
- An indicator to show when a poll is busy or the result of the last poll
- (Data Received or Error)
- Configuration values, some of which cannot be changed from the faceplate



Item	Description
1	Enter a serial number for your Studio 5000 Logix Designer software, the contract number for your TechConnect[SM], or other technical support contract information. This information is then available for ready reference if you call Rockwell Automation Technical Support.
2	Enable / Disable Data Collection **IMPORTANT**: The raP_Dvc_LgxCPU_5x80 instruction accomplishes its data collection by using MSG instructions to the controller (MSG to self), which uses some controller communication resources. You can leave data collection disabled until it is needed. Some faceplate data is monitored without using the polling messages and is still displayed. When disabled, only data collection via MSG instructions is disabled. Other data can still be updated and displayed on the faceplate. Data not updated when collection is disabled is not displayed.

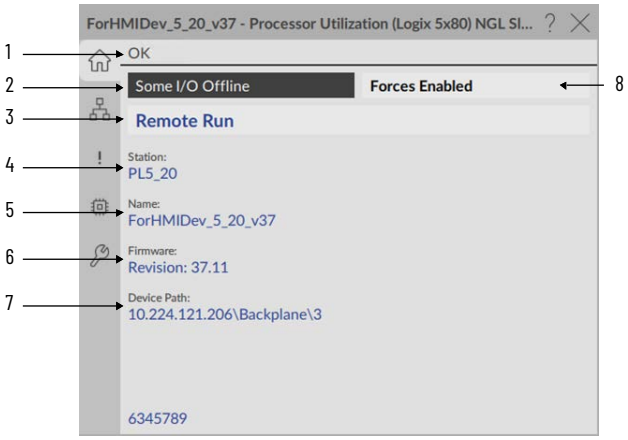


Item	Description
1	Area name for security
2	Enter the interval that is used to collect and update data that is displayed on the other faceplate tabs. **IMPORTANT**: If you set this parameter too low, it can result in a flood of messages to the controller and possibly affect control performance. Do not use a value less than 5 seconds unless instructed to do so by a Rockwell Automation Technical Support specialist.

FactoryTalk Optix Faceplates

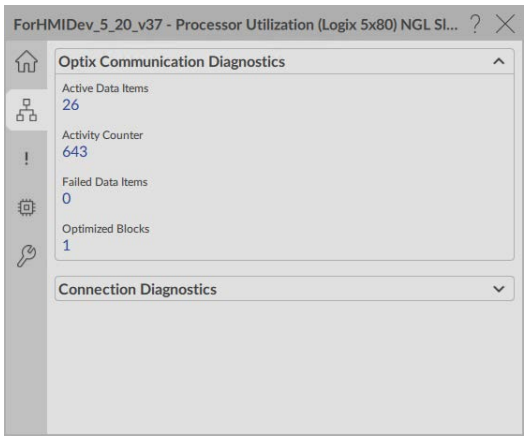
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab

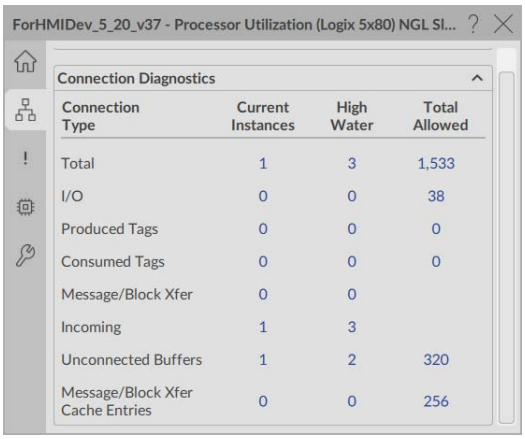


Item	Description
1	Controller OK indicator
2	I/O communication status
3	Current controller mode
4	Device shortcut
5	Processor name defined in RSLogix 5000
6	Current firmware revision
7	Path from the HMI server to the device
8	I/O forcing status indicator

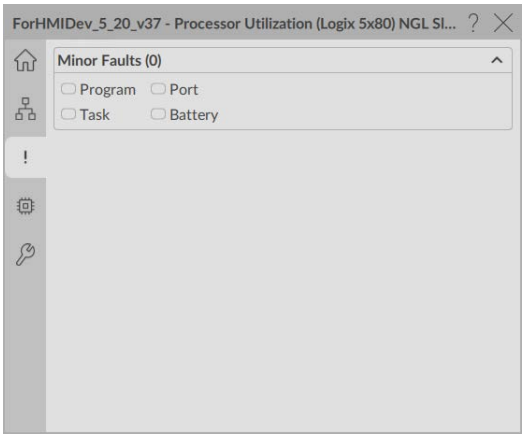
Communication Tab - Optix Communication Diagnostics



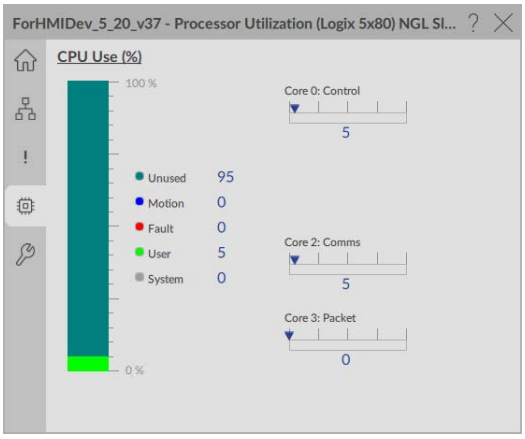
Communication Tab - Connection Diagnostics



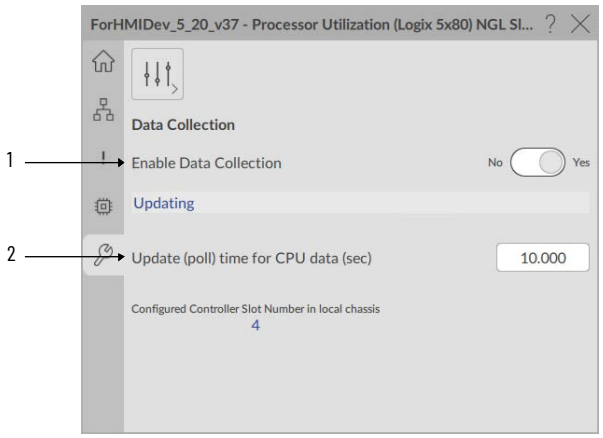
Faults Tab



Performance Tab

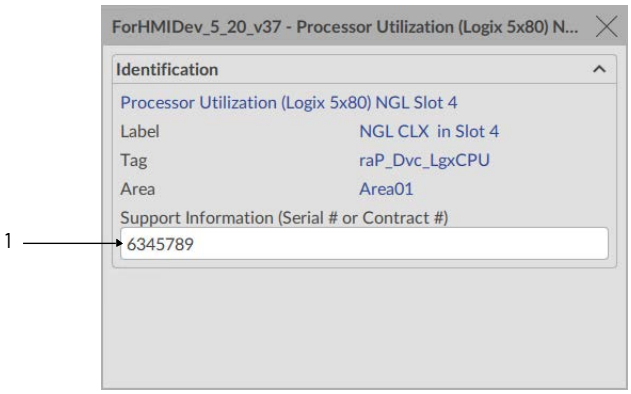


Maintenance Tab



Item	Description
1	Enable / Disable Data Collection. **IMPORTANT**: The raP_Dvc_LgxCPU_5x80 instruction accomplishes its data collection by using MSG instructions to the controller (MSG to self), which uses some controller communication resources. You can leave data collection disabled until it is needed. Some faceplate data is monitored without using the polling messages and is still displayed. When disabled, only data collection via MSG instructions is disabled. Other data can still be updated and displayed on the faceplate. Data not updated when collection is disabled is not displayed.
2	Enter the interval that is used to collect and update data that is displayed on the other faceplate tabs. **IMPORTANT**: If you set this parameter too low, it can result in a flood of messages to the controller and possibly affect control performance. Do not use a value less than 5 seconds unless instructed to do so by a Rockwell Automation Technical Support specialist.

Advanced HMI Configuration Tab



Item	Description
1	Enter a serial number for your Studio 5000 Logix Designer software, the contract number for your TechConnect^{SM}, or other technical support contract information. This information is then available for ready reference if you call Rockwell Automation Technical Support.

Logix Redundant Controller Monitor (raP_Dvc_LgxRedun)

The raP_Dvc_LgxRedun (Logix Redundant Controller Monitor) Add-On Instruction monitors one redundant pair of Logix controllers. The instruction checks primary and secondary controller status that can affect the ability of the system to switch to the back-up controller on a failure of the primary.

Guidelines

Use this instruction in these situations:

- You are using Logix controllers in a redundant configuration.
- You want to monitor the status of the redundant controller pair.
- You want to display this status to operators, maintenance personnel, or engineers.

Do not use this instruction in these situations:

- You are using single Logix controllers, not in a redundant configuration. The raP_Dvc_LgxRedun instruction is designed around the ControlLogix® Enhanced Redundancy System architecture, by using information from the 1756-RM2 Redundancy Modules. The raP_Dvc_LgxRedun Add-On Instruction does not verify in a non-redundant system because the data items it monitors do not exist in a non-redundant configuration.
- Your controllers are in an accessible location and the indicators on the controllers, network modules, and redundancy modules provide sufficient information about redundancy status.

For more information, see the ControlLogix Enhanced Redundancy System User Manual, publication 1756-UM535.

Functional Description

The raP_Dvc_LgxRedun instruction is provided as a rung import for installation. Importing this rung into your ladder diagram routine:

- imports the Add-On Instruction definition
- creates an instruction instance
- creates and completes all required tags and data structures for the instruction

Once the rung is imported, and before you download and run the application, set the path in each Message tag that references the input/output parameters of the instruction to point to slot that contains the 1756-RM2 module in the local chassis ('1, <slot>').

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Dvc_LgxRedun_5.30.**00**_RUNG.L5X rung import file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

For the object and visualization parameters, see PlantPAx Process Objects, publication PROCES-RD200, and PlantPAx Visualization Files, publication PROCES-RD201.

Operations

The raP_Dvc_LgxRedun instruction monitors a redundant pair of Logix controllers and provides the following information and capabilities:

- Determines and displays whether the current primary controller is in Chassis 'A' or Chassis 'B' (as defined by user configuration)
- Displays the Chassis A and Chassis B Redundancy Module (1756-RM2) status
- Displays the Controller A and Controller B redundancy status
- Displays the Controller A and Controller B keyswitch positions
- Displays the overall compatibility between modules in Chassis A and modules in Chassis B
- Displays the synchronization progress in percent complete
- Displays the amount of data that is transferred from the Primary redundancy module to the Secondary in the most recent transfer, and the most sent in any transfer (high-water mark)

This instruction also supports the following commands, if enabled in the configuration:

- Initiate a switchover from Primary to Secondary
- Initiate a resynchronization of the system (if it does not take place automatically)

Command Sources

The raP_Dvc_LgxRedun instruction has no commands or outputs that are intended to control equipment and so does not have any command sources.

Alarms

The raP_Dvc_LgxRedun Instruction uses the following alarm, which is implemented by using Tag Based Alarms.

Alarm	Alarm Name	Description
Secondary not ready	Alm_SecNotRdy	Raised when the secondary controller in a redundant pair is not ready to take over control on loss of the primary.

Virtualization

The raP_Dvc_LgxRedun Add-On Instruction does not have a Virtualization capability.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Dvc_LgxRedun instruction must always be scanned true. In relay ladder logic, the raP_Dvc_LgxRedun instruction must be by itself on an unconditional rung. If the Rung Import provided with the Rockwell Automation is used to install this instruction, the proper rung is created for you.
Powerup (prescan, first scan)	On Pre-scan, any commands that are received before first scan are discarded.
Postscan (SFC transition)	No SFC Postscan logic is provided.

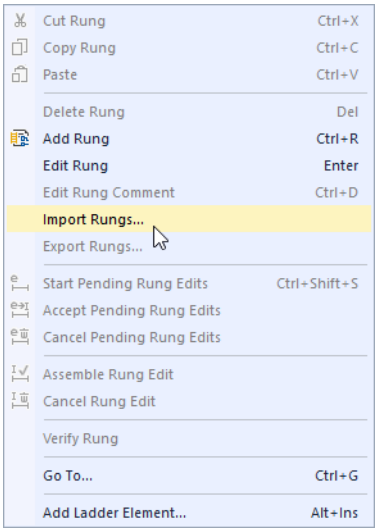
See to the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010, for more information.

Programming Example

The raP_Dvc_LgxRedun instruction is provided fully configured as a rung import, so little programming is required for the instruction to be used. This programming example shows how the rung import is used to instantiate the raP_Dvc_LgxRedun instruction.

As raP_Dvc_LgxRedun is a rung import, it must be created in a Ladder Diagram routine. The following steps describe how you instantiate raP_Dvc_LgxRedun in your routine.

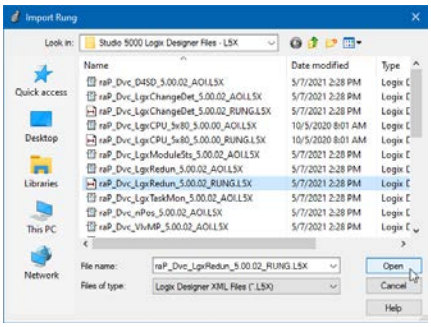
1. In your ladder routine, right-click where to insert the rungs and select Import Rungs.



   The Import Rungs dialog box appears.

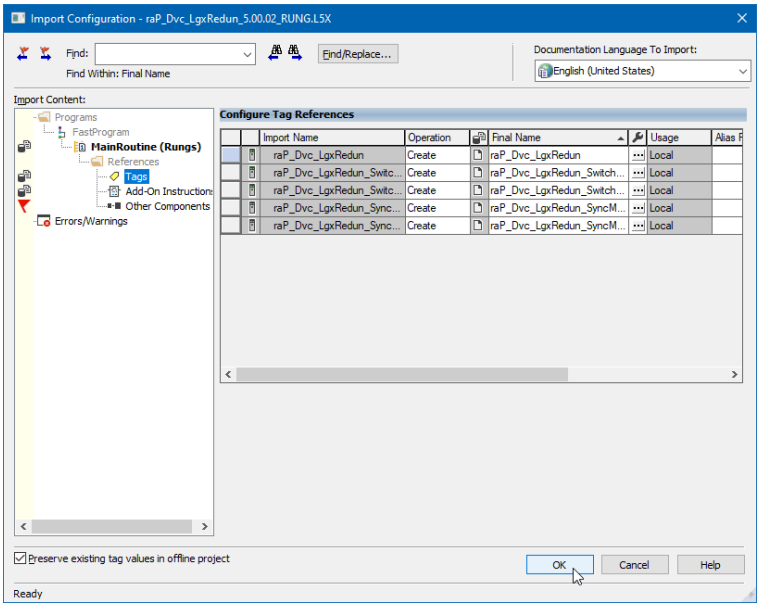
2. Select the appropriate raP_Dvc_LgxRedun rung import file that is named in Required Files on page 334.

3. Select Open.



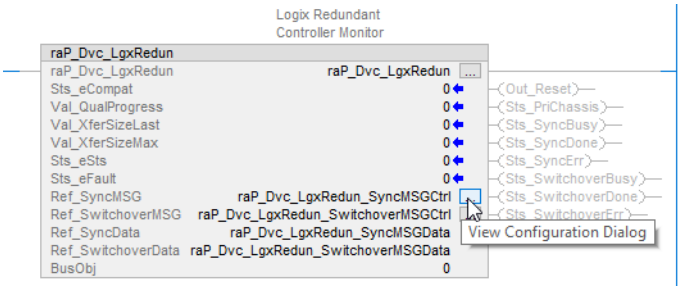
The Import Configuration dialog box appears.

4. To create the instance of raP_Dvc_LgxRedun, select OK.



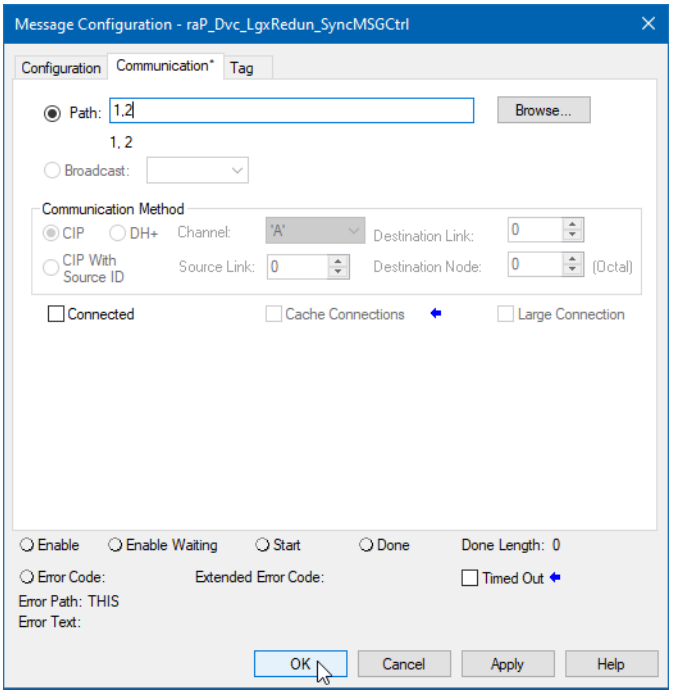
5. Complete the following steps for each of the two MSG controls to set the path to point to the 1756-RM2 module in the local chassis.

   a. Select the ellipsis next to the MSG control tag.



The Message Configuration dialog box appears.

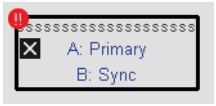
b. To set the second number in the path to the slot number of the 1756-RM2 module, Select the Communication tab.



c. Select OK.

Graphic Symbols

A Graphic Symbol (global object) is created once and can be referenced multiple times on multiple displays in an application. When changes are made to the original (base) object, the instantiated copies (reference objects) are automatically updated. Use of graphic symbols, with tag structures in the ControlLogix system, aid consistency and save engineering time.

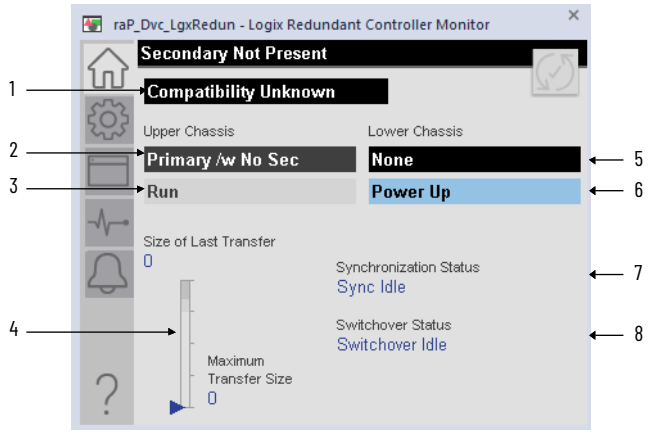
FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_LgxRedun 	raP_5_30_raP_Dvc_LgxRedun_GS 	This global object is used for redundancy modules.

FactoryTalk View SE Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab

The Operator tab provides status information on the primary and secondary controllers.

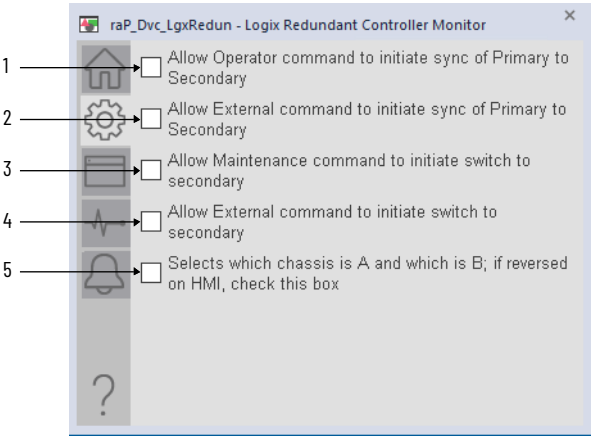


Item	Description
1	Compatibility status
2	Chassis A (upper chassis) status
3	Chassis A (upper chassis) controller mode
4	Transfer size and status
5	Chassis B (lower chassis) status
6	Chassis B (lower chassis) controller status
7	Synchronization status
8	Switchover status

Engineering Tab

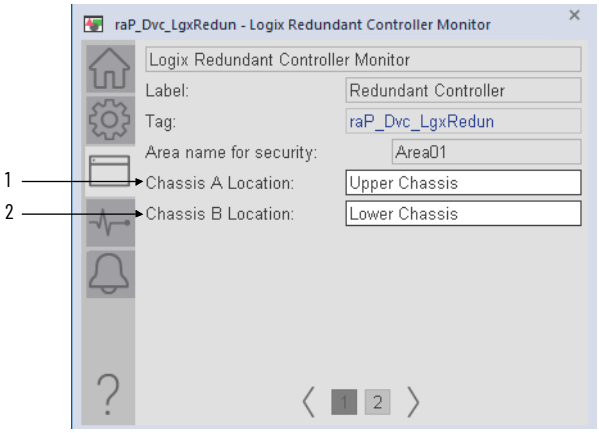
The Engineering tab provides access to device configuration parameters and ranges, options for device and I/O setup, security area, displayed text, and faceplate-to-faceplate navigation settings, for initial system commissioning or later system changes.

On the Engineering tab, you can identify and configure each chassis and configure display, switchover, and synchronization options.

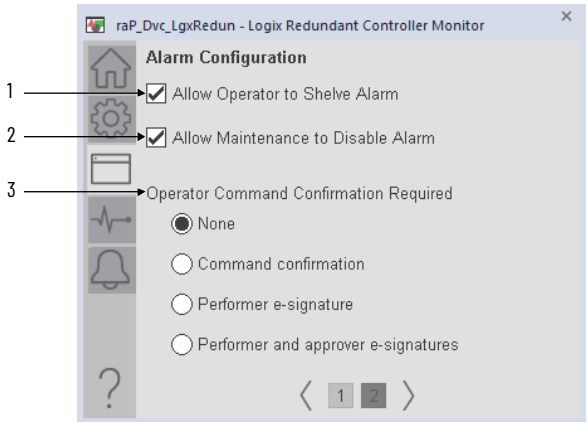


Item	Description
1	Select to enable the Operator command to initiate synchronization of the primary controller to the secondary controller.
2	Select to enable the External command to initiate synchronization of the primary controller to the secondary controller.
3	Select to enable the Maintenance command to switch to the secondary controller.
4	Select to enable the External command to switch to the secondary controller.
5	Select to designate chassis A and chassis B on the HMI.

HMI Configuration Tab



Item	Description
1	Enter a name for the location of Chassis A location.
2	Enter a name for the location of Chassis B location.



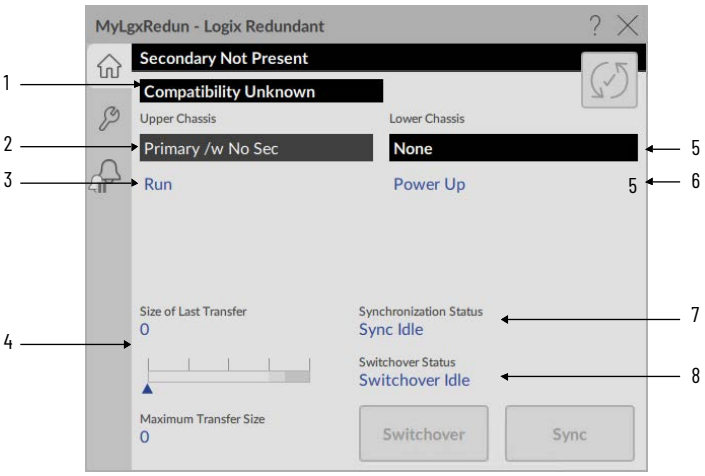
Item	Description
1	Select to allow Operator to shelve the alarm.
2	Select to allow Maintenance to disable the alarm.
3	Select the type of confirmation required for Operator commands.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

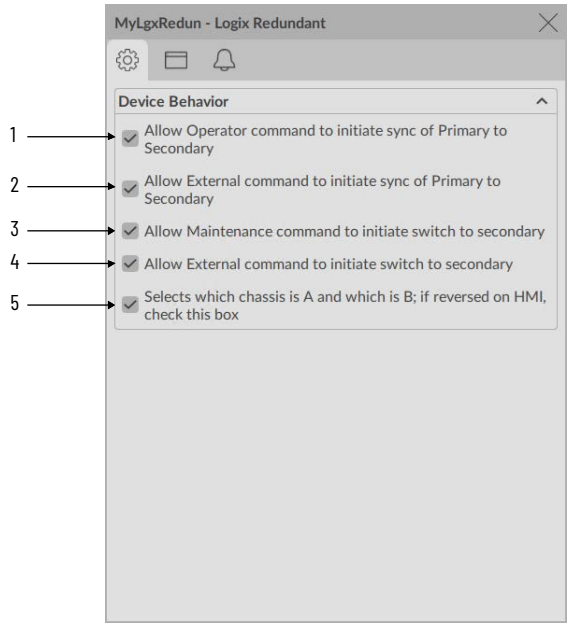
Any feature that is contained in the FactoryTalk Optix faceplates has the same functionality as used in the FactoryTalk View SE faceplates. See FactoryTalk View SE Faceplates on page 339 for descriptions of the features.

Operator Tab



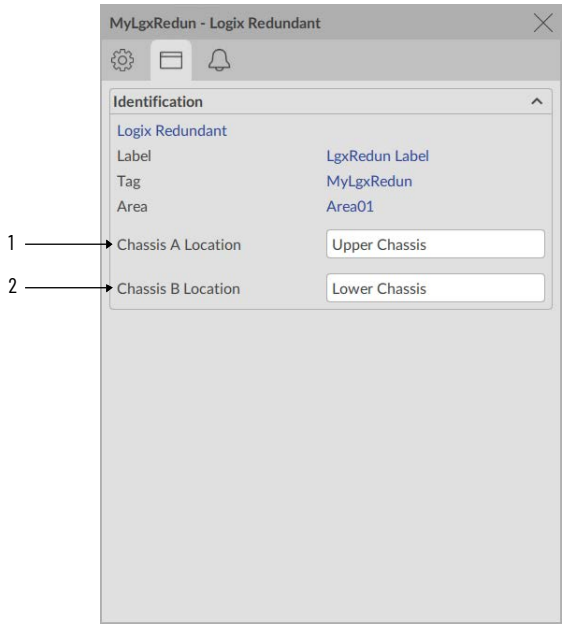
Item	Description
1	Compatibility status
2	Chassis A (upper chassis) status
3	Chassis A (upper chassis) controller mode
4	Transfer size and status
5	Chassis B (lower chassis) status
6	Chassis B (lower chassis) controller status
7	Synchronization status
8	Switchover status

Advanced Engineering Tab



Item	Description
1	Select to enable the Operator command to initiate synchronization of the primary controller to the secondary controller.
2	Select to enable the External command to initiate synchronization of the primary controller to the secondary controller.
3	Select to enable the Maintenance command to switch to the secondary controller.
4	Select to enable the External command to switch to the secondary controller.
5	Select to designate chassis A and chassis B on the HMI.

Advanced HMI Configuration Tab



Item	Description
1	Enter a name for the location of Chassis A location.
2	Enter a name for the location of Chassis B location.

Logix Module Status (raP_Dvc_LgxModuleSts)

The raP_Dvc_LgxModuleSts (Logix Module Status) Add-On Instruction monitors the connection status of one module or device in the I/O configuration tree of the Logix controller, and monitors it for any I/O channel faults on the module. The instruction provides an "I/O fault" status to dependent equipment, and provides a "Module Fault" status and alarm if the connection to the module is lost. It also provides an "Any Channel Fault" status and alarm if any I/O channel on the module reports a fault.

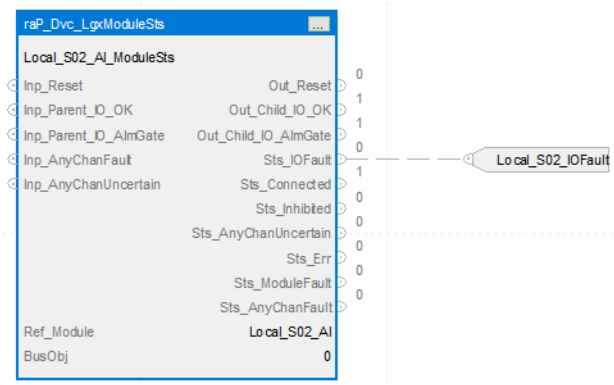
Guidelines

Use this instruction if you want to monitor the I/O connection status of a given module or device. This instruction is for use in CompactLogix 5380 and ControlLogix 5580 controllers using software / firmware version 33 or later.

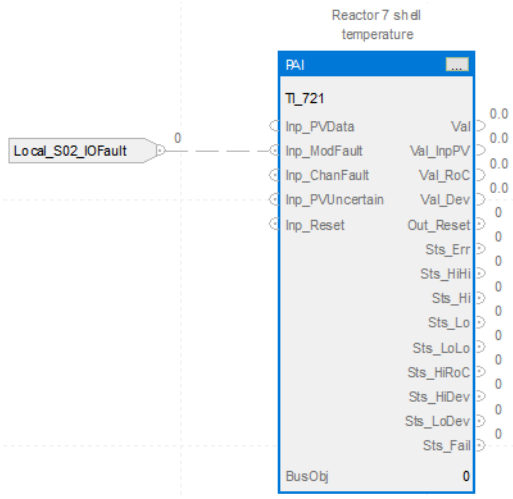
Functional Description

The raP_Dvc_LgxModuleSts Add-On Instruction is used to check the I/O connection status for the given module or device. The instruction provides an I/O Fault status output, which is 1 when the connection is lost, and 0 when the connection to the I/O module is OK and running normally. This status available for use by other Add-On Instructions that use inputs or outputs of the given I/O module or device.

The following images show how the I/O Fault status output is connected to instructions that use the module being monitored. Here is the code showing the raP_Dvc_LgxModuleSts instruction getting the connection status for the module in Local chassis, Slot 2:



That status is passed along to the Analog Input instruction, which uses an input on that module:



The raP_Dvc_LgxModuleSts instruction can be used to provide the connection status for any connected device (anything with a Requested Packet Interval) in the I/O Configuration tree in Studio 5000 Logix Designer® application. These devices include I/O modules, network communication modules, motor drives, overload relays, flowmeters, analyzers, weigh scales and other devices on EtherNet/IP™ or another I/O network, or in the chassis containing the controller.

IMPORTANT	Entry of a name for an I/O module or other device in the I/O Configuration is optional. However, in order for the raP_Dvc_LgxModuleSts instruction to refer to the module or device, you MUST give the module or device a name.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix® firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Dvc_LgxModuleSts_5.30.**00**_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

Command Sources

The raP_Dvc_LgxModuleSts instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Dvc_LgxModuleSts Instruction uses the following alarms, which are implemented using Tag Based Alarms:

Alarm	Alarm Name	Description
Module Fault	Alm_ModuleFault	Raised when the I/O connection to the module or device is not in the Running state.
Any Channel Faulted	Alm_AnyChanFault	Raised when any I/O channel on the module is reporting a fault.

Virtualization

Virtualization allows the raP_Dvc_LgxModuleSts instruction to report a virtual connection status for use in test, demonstration, or training systems. The raP_Dvc_LgxModuleSts Add-On Instruction can be selected to virtual or physical (normal) operation. When physical operation is selected, the actual module connection status is monitored, and an I/O Fault status and Module Fault alarm is reported if the connection is not running. When virtual operation is

selected, the actual module connection status is ignored; the Set_VirtualConnectedSts input parameter determines the reported connection status.

Set_VirtualConnectedSts value	Description
1	Connected, the connection status is reported as OK
0	Faulted, the connection status is reported as faulted, the Sts_IOFault status is raised for dependent devices, and the Alm_ModuleFault alarm is raised.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Dvc_LgxModuleSts instruction must always be scanned true. In relay ladder logic, the raP_Dvc_LgxModuleSts instruction must be by itself on an unconditional rung.
Powerup (prescan, first scan)	All commands, including alarm acknowledge and reset, virtual or physical selection, maintenance bypass and check, plus all latched internal fault status bits, are cleared on prescan.
Postscan (SFC transition)	No SFC Postscan logic is provided.

See to the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010, for more information.

Programming Examples

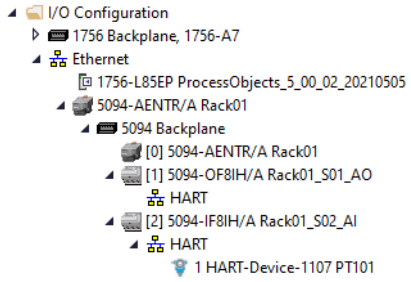
The example in the Function Description section shows the basic use of the raP_Dvc_LgxModuleSts Add-On Instruction for monitoring a module connection. The instruction can also monitor and alarm channel faults on the I/O module, using some simple external logic. For many discrete modules, the individual channel fault bits are collected into a single INT or DINT (16-bit or 32-bit integer), and if the value of this integer is not zero, there is at least one channel fault:

The "NEQ" instruction determines that there is at least one channel fault, and this status is forwarded to the raP_Dvc_LgxModuleSts instruction via the Inp_AnyChanFault input pin.

Some analog modules use a similar grouping of channel faults; others require the user to "OR" the individual channel faults in the external logic:

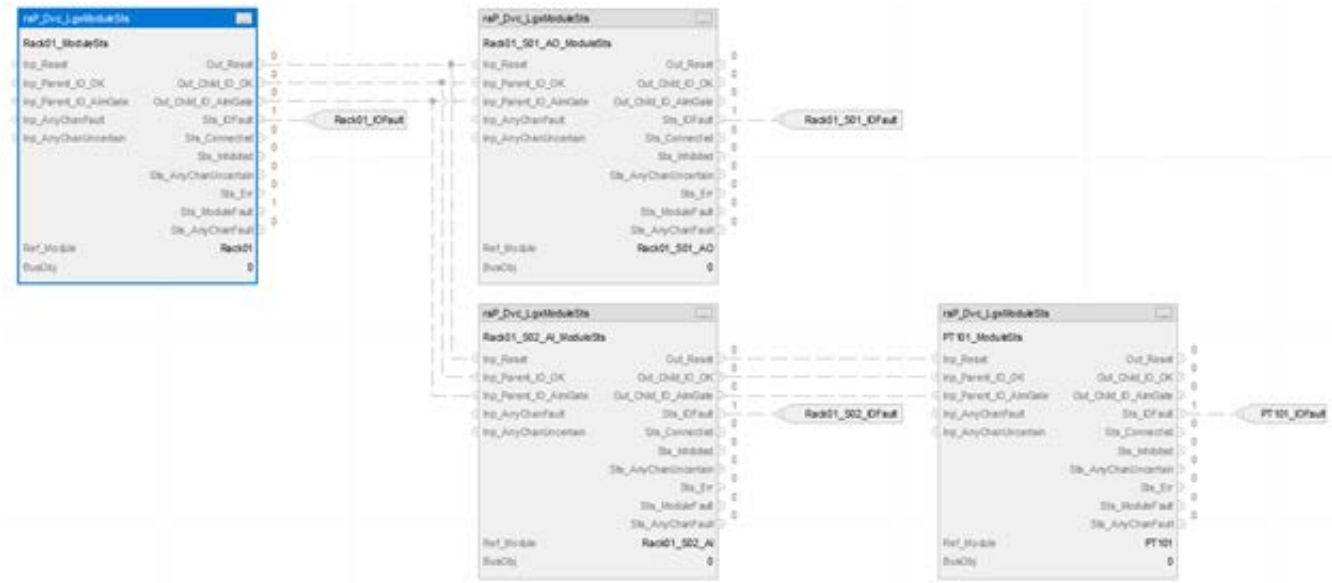
The raP_Dvc_LgxModuleSts also has the capability to be organized, via connecting pins or via the optional Bus, to match the I/O hierarchy. This organization can help prevent alarm floods by inhibiting the alarms from lower-level modules when a higher-level connection fault causes a cascading I/O failure.

Using the following I/O configuration as an example:



If the connection to the 5094-AENTR adapter fails, all devices under it will report I/O connection loss, and a flood of Module Fault alarms occur. By wiring the raP_Dvc_LgxModuleSts instructions into a hierarchy, the fault that is detected for the 5094-

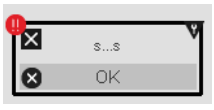
AENTR can be cascaded to the I/O Fault inputs of all dependent devices, AND can be used to inhibit the Module Fault alarms at the lower-levels, reducing the number of alarms generated.



If the I/O adapter (5094-AENTR) connection fails, the HART analog input module (5094-IF8IH) and the PT101 device logic will be informed of the I/O fault condition (via the Rack01_S02IOFault and PT101_IOFault IREFs), but the analog input module and PT101 device Module Fault alarms are inhibited (via the child I/O alarm gate connections), so that only the root cause module fault alarm will be generated.

Graphic Symbols

A Graphic Symbol (global object) is created once and can be referenced multiple times on multiple displays in an application. When changes are made to the original (base) object, the instantiated copies (reference objects) are automatically updated. Use of graphic symbols, with tag structures in the ControlLogix system, aid consistency and save engineering time.

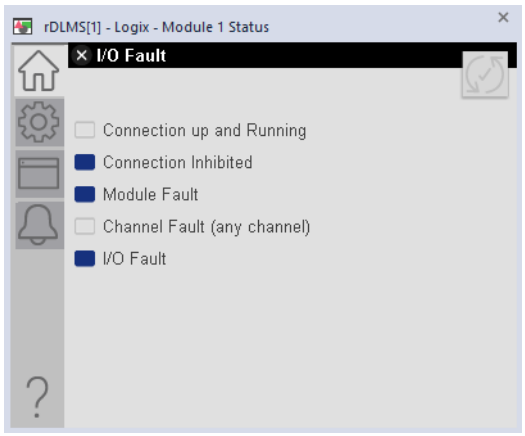
FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_LgxModuleSts 	raP_5_30_raP_Dvc_LgxModuleSts_GS 	This global object is used for module status.

FactoryTalk View SE Faceplates

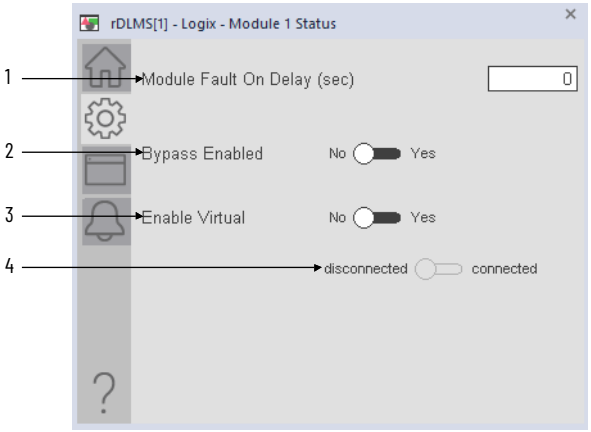
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Operator Tab

The operator tab displays the status of the module.

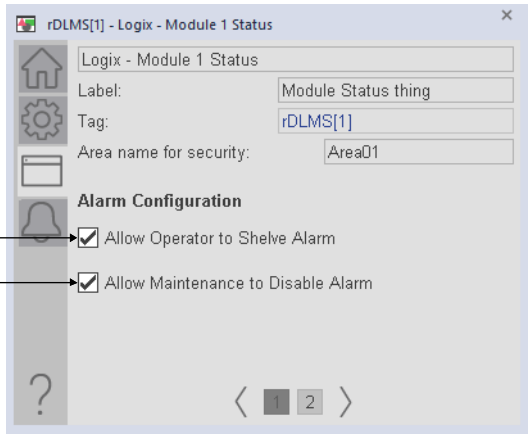


Engineering Tab

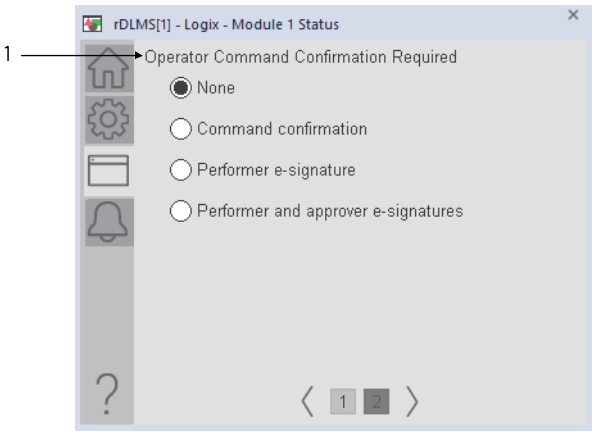


Item	Description
1	Enter the delay, in seconds, after an I/O communication fault is detected before raising the Alm_ModuleFault alarm. This delay may be needed to avoid an alarm flood when a network or I/O adapter fault cascades down to several modules. The delay allows time for the parent fault to inhibit the individual module fault alarms.
2	Select yes to bypass (block) the generation of the I/O Fault status (Sts_IOFault). Select no to enable I/O Fault status generation.
3	Select yes to enable virtual operation; the actual module connection status is ignored, and the virtual connection status setting (#4) is used instead. Select no to enable physical operation; the actual module connection is monitored.
4	When virtual operation is selected, use this selector to set the virtual connection status. When set to disconnected, an I/O Fault status is generated (if not bypassed).

HMI Configuration Tab



Item	Description
1	Select to allow Operator to shelve the alarm.
2	Select to allow Maintenance to disable the alarm.



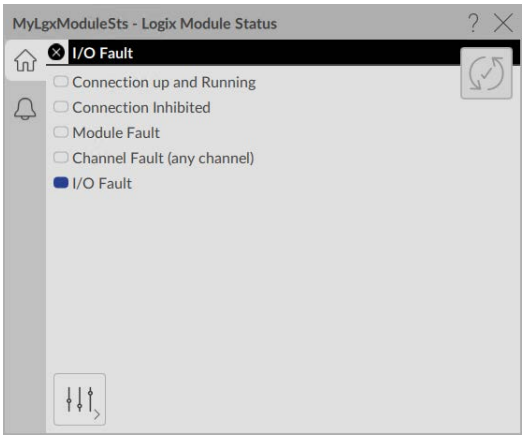
Item	Description
1	Select the type of confirmation required for Operator commands.

FactoryTalk Optix Faceplates

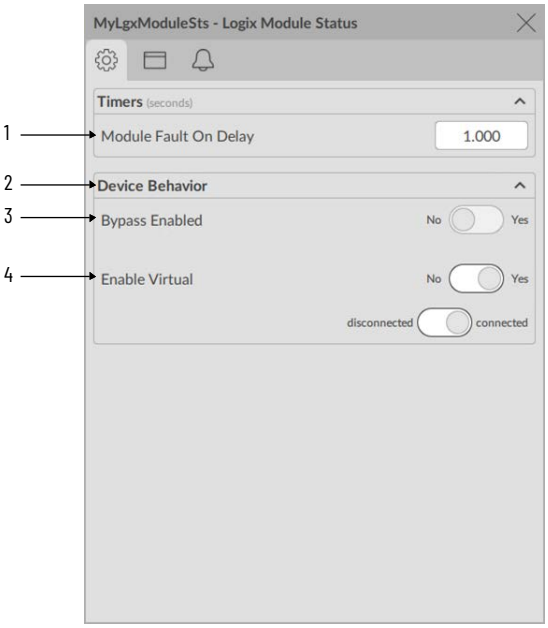
There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Any feature that is contained in the FactoryTalk Optix faceplates has the same functionality as used in the FactoryTalk® View SE faceplates. See FactoryTalk View SE Faceplates on page 348 for descriptions of the features.

Operator



Advanced Engineering



Item	Description
1	Enter the delay, in seconds, after an I/O communication fault is detected before raising the Alm_ModuleFault alarm. This delay may be needed to avoid an alarm flood when a network or I/O adapter fault cascades down to several modules. The delay allows time for the parent fault to inhibit the individual module fault alarms.
2	Select yes to bypass (block) the generation of the I/O Fault status (Sts_IOFault). Select no to enable I/O Fault status generation.
3	Select yes to enable virtual operation; the actual module connection status is ignored, and the virtual connection status setting (#4) is used instead. Select no to enable physical operation; the actual module connection is monitored.
4	When virtual operation is selected, use this selector to set the virtual connection status. When set to disconnected, an I/O Fault status is generated (if not bypassed).

Logix Task Monitor (raP_Dvc_LgxTaskMon)

The raP_Dvc_LgxTaskMon (Logix Task Monitor) Add-On Instruction monitors one task running in a Logix controller to provide task statistics, such as task scan time and overlap count.

The instruction also provides the following:

- Display Values for Task configuration: Task Name, Priority, Rate, Watchdog settings
- Display Values for Task statistics: Last and Max Execution time, Overlap Count
- Display of the Task Inhibit / Active Status
- Configuration of a "Plan" Execution Time, the time in which the Task is expected to always complete (including interrupts by higher priority Tasks)
- An optional Alarm when Last (actual) Execution Time exceeds Plan
- A Reset command, which clears and acknowledges the Over Plan Alarm.
- Maintenance Commands to clear the Max Execution Time and Overlap Count.

Guidelines

Use this instruction in these situations:

- Monitor the execution of one or more tasks in a Logix controller
- Set an alarm when task execution time exceeds a 'plan' threshold

Do **not** use this instruction if you are using suitable software or another method to monitor controller task execution.

Functional Description

The raP_Dvc_LgxTaskMon instruction includes an Add-On Instruction for use with:

- Studio 5000 Logix Designer® software, version 33 or later
- Logix controllers, firmware revision 33 or later
- Graphic symbol and faceplate display for use with FactoryTalk® View Site Edition software, version 13 or later.

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Dvc_LgxTaskMon_5.30.**00**_AOI.L5X Add-On Instruction must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

See Visualization Files on page 21 for general information on visualization files.

Operations

The raP_Dvc_LgxTaskMon Add-On Instruction does not use modes or virtualization.

Command Sources

The raP_Dvc_LgxModuleSts instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Dvc_LgxTaskMon Instruction uses the following alarms, which are implemented by using Tag Based Alarms.

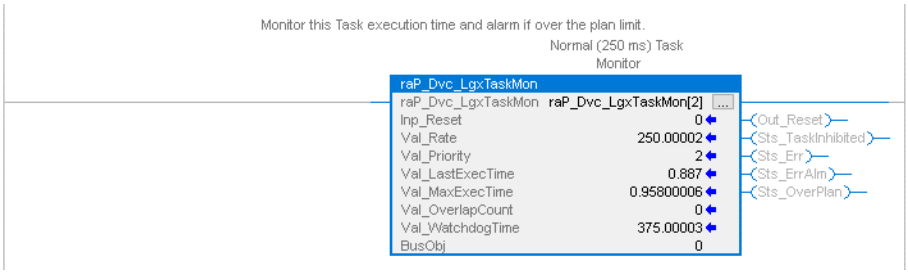
Alarm	Alarm Name	Description
Over Plan	Alm_OverPlan	Task execution time exceeds plan limit.

Virtualization

Virtualization allows the raP_Dvc_LgxModuleSts instruction to report a virtual connection status for use in test, demonstration, or training systems. The raP_Dvc_LgxModuleSts Add-On Instruction can be selected to virtual or physical (normal) operation. When physical operation is selected, the actual module connection status is monitored, and an I/O Fault status and Module Fault alarm is reported if the connection is not running. When virtual operation is selected, the actual module connection status is ignored; the Set_VirtualConnectedSts input parameter determines the reported connection status.

Set_VirtualConnectedSts value	Description
1	Connected, the connection status is reported as OK
0	Faulted, the connection status is reported as faulted, the Sts_IOFault status is raised for dependent devices, and the Alm_ModuleFault alarm is raised.

Programming Example

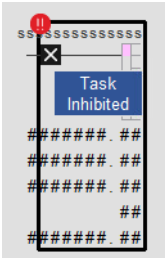
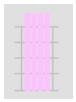


Place an instance like this in each controller task and have it always scanned true. In ladder diagram, you can use an array of backing tags, assigning a different array member for each task. For the default PlantPAx® process controller task model, the following assignments are recommended and are included in the PlantPAx template application:

raP_Dvc_LgxTaskMon[0]	System (1000 ms) task
raP_Dvc_LgxTaskMon[1]	Fast (100 ms) task
raP_Dvc_LgxTaskMon[2]	Normal (250 ms) task
raP_Dvc_LgxTaskMon[3]	Slow (500 ms) task

The task in which the instruction is instantiated will be checked each scan for its status and to determine whether its execution time exceeds the plan limit for that task.

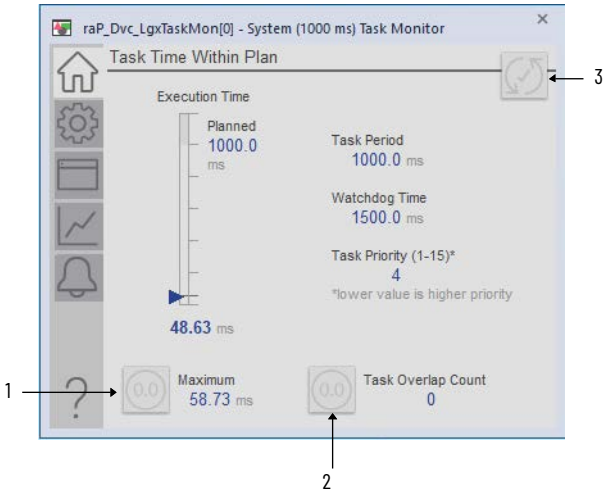
Graphic Symbols

FactoryTalk View SE Graphic Symbol	FactoryTalk Optix Graphic Symbol	Description
GO_TaskMon 	raP_5_30_raP_Dvc_LgxTaskMon_GS_TaskMon 	This global object provides task statistics for one task in a Logix controller.
GO_TaskMonSummary 	raP_5_30_raP_Dvc_LgxTaskMon_GS_TaskMonSummary 	This object provides a graphic representation of eight L_TaskMon objects in a controller. Click this object to display a summary screen of all eight L_TaskMon objects.

FactoryTalk View SE Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

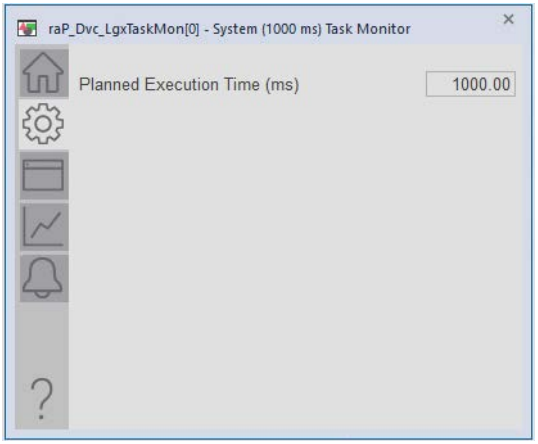
Operator Tab



Item	Description
1	Click to reset the Max Execution Time.
2	Click to reset Task Overlap Count.
3	Click to reset and acknowledge all alarms.

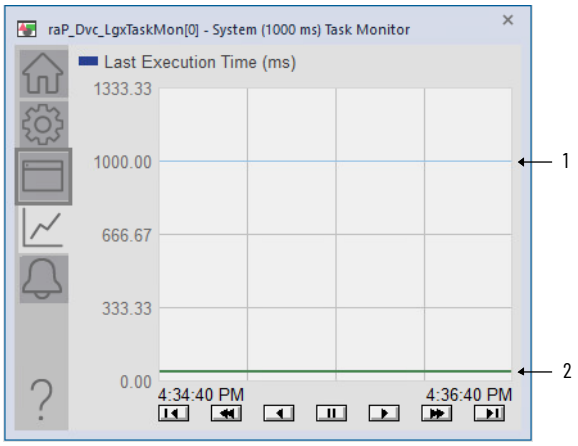
Engineering Tab

The maintenance tab provides access to the planned execution time.



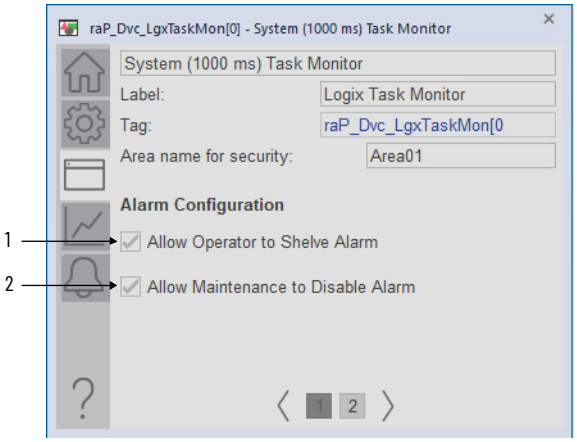
Trends Tab

The Trends tab shows trend charts of key device data over time. These faceplate trends provide a quick view of current device performance to supplement, but not replace, dedicated historical or live trend displays.

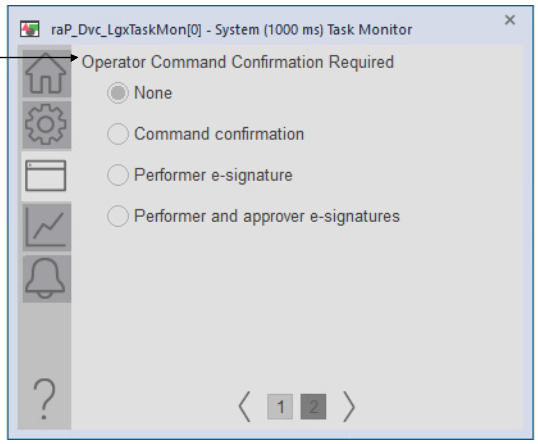


Item	Description
1	Planned execution time (blue line)
2	Last execution time (green line)

HMI Tab



Item	Description
1	Select to allow Operator to shelve alarm.
2	Select to allow Maintenance to disable alarm.



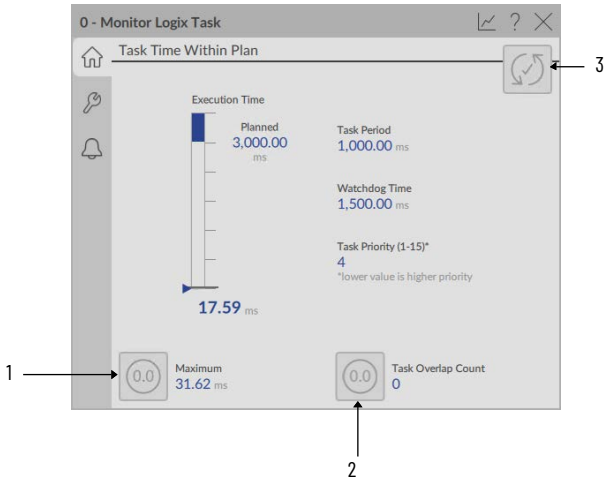
Item	Description
1	Select to configure operator command confirmation. This action would take place after any operator command.

FactoryTalk Optix Faceplates

There are basic faceplate attributes that are common across all instructions. The Trends tab, Diagnostics tab, and Alarms tab all have the same basic functionality and are not described in this section. See Basic Faceplate Attributes on page 28.

Any feature that is contained in the FactoryTalk® Optix™ faceplates has the same functionality as used in the FactoryTalk® View SE faceplates. See FactoryTalk View SE Faceplates on page 353 for descriptions of the features.

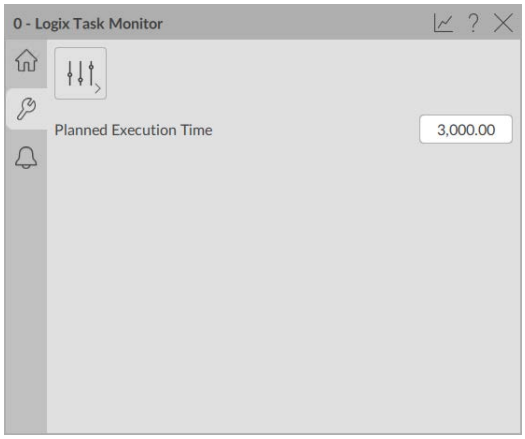
Operator Tab



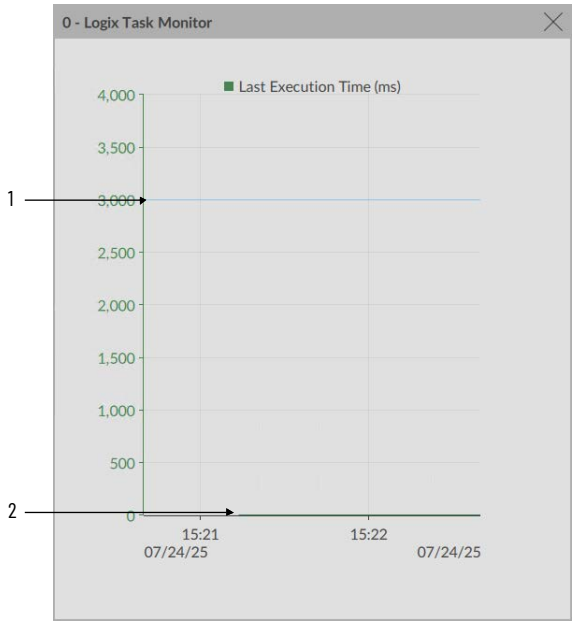
Item	Description
1	Click to reset the Max Execution Time.
2	Click to reset Task Overlap Count.
3	Click to reset and acknowledge all alarms.

Maintenance Tab

The maintenance tab provides access to the planned execution time.



Trends Tab



Item	Description
1	Planned execution time (blue line)
2	Last execution time (green line)

Logix Event (raP_Tec_LgxEvent)

The raP_Tec_LgxEvent (Logix Event) Add-On Instruction captures any of 16 event bit rising edge transitions and records the lowest-order rising edge bit as the reason of the event. The instruction provides an "I/O fault" input to monitor parent IO conditions. It also provides a Reset to clear the event reason.

Guidelines

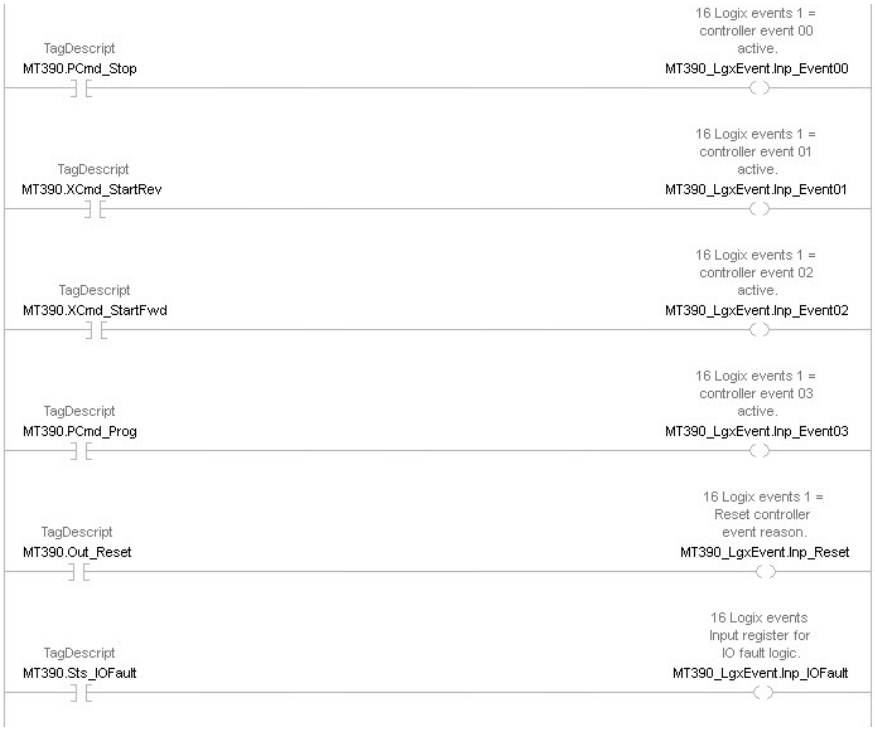
Use this instruction if you want to monitor up to 16 User-defined events, per object.

Functional Description

The raP_Tec_LgxEvent Add-On Instruction is used to capture any of 16 bit rising edge transitions and records the lowest order rising edge bit as the reason of the event. The events are published in the Sts_Reasons parameter. Reasons are only captured if the Inp_IOFault and

the Inp_Reset parameters are low. (0) Event reasons are cleared by setting the Reset input parameter. (1) Event input conditions can be connected to any user-defined logic.

The following images show how the event inputs are mapped to the instruction. Ladder logic is typically used to allow for more complex trigger conditions. Here is the code showing the mapping of four event triggers, as well as the Reset and IOFault:



Those inputs are the source of the raP_Tec_LgxEvent instruction:

The following images show how the event Reasons are assigned a User description for the Event. Each individual event is allowed a unique description to be applied. The description must be changed/updated to what the you want to be displayed in the reports.

Name	Usage	Value	Force Mask	Style	Data Type	Description
raP_Tec_LgxEvent_01.Sts_Reason		2#0000_0000_0000_00...		Binary	INT	16 Logix events Reason of individual controller event.
raP_Tec_LgxEvent_01.Sts_Reason.0		0		Decimal	BOOL	16 Logix events Reason Zero
raP_Tec_LgxEvent_01.Sts_Reason.1		0		Decimal	BOOL	16 Logix events Reason One
raP_Tec_LgxEvent_01.Sts_Reason.2		0		Decimal	BOOL	16 Logix events Reason Two
raP_Tec_LgxEvent_01.Sts_Reason.3		0		Decimal	BOOL	16 Logix events Reason Three
raP_Tec_LgxEvent_01.Sts_Reason.4		0		Decimal	BOOL	16 Logix events Reason Four
raP_Tec_LgxEvent_01.Sts_Reason.5		0		Decimal	BOOL	16 Logix events Reason Five
raP_Tec_LgxEvent_01.Sts_Reason.6		0		Decimal	BOOL	16 Logix events Reason Six
raP_Tec_LgxEvent_01.Sts_Reason.7		0		Decimal	BOOL	16 Logix events Reason Seven
raP_Tec_LgxEvent_01.Sts_Reason.8		0		Decimal	BOOL	16 Logix events Reason Eight
raP_Tec_LgxEvent_01.Sts_Reason.9		0		Decimal	BOOL	16 Logix events Reason Nine
raP_Tec_LgxEvent_01.Sts_Reason.10		0		Decimal	BOOL	16 Logix events Reason Ten
raP_Tec_LgxEvent_01.Sts_Reason.11		0		Decimal	BOOL	16 Logix events Reason Eleven
raP_Tec_LgxEvent_01.Sts_Reason.12		0		Decimal	BOOL	16 Logix events Reason Twelve
raP_Tec_LgxEvent_01.Sts_Reason.13		0		Decimal	BOOL	16 Logix events Reason Thirteen
raP_Tec_LgxEvent_01.Sts_Reason.14		0		Decimal	BOOL	16 Logix events Reason Fourteen
raP_Tec_LgxEvent_01.Sts_Reason.15		0		Decimal	BOOL	16 Logix events Reason Fifteen

Required Files

Add-On Instructions are reusable code objects that contain encapsulated logic that can streamline implementing your system. This lets you create your own instruction set for programming logic as a supplement to the instruction set provided natively in the ControlLogix firmware. An Add-On Instruction is defined once in each controller project, and can be instantiated multiple times in your application code as needed.

Controller Files

The raP_Tec_LgxEvent_5.30.**00**_AOI.L5X Add-On Instruction definition file must be imported into the controller project to be able to be used in the controller configuration. The service release number (boldfaced) can change as service revisions are created.

Visualization Files

There are no visualization files because the raP_Tec_LgxEvent object does not use Graphic Symbols or Faceplates.

Operations

Command Sources

The raP_Tec_LgxEvent instruction has no commands or outputs that are intended to control equipment and therefore does not have any selection of active command source.

Alarms

The raP_Tec_LgxEvent Instruction has no Alarms.

Virtualization

The raP_Tec_LgxEvent Instruction has no Virtualization.

Execution

The following table explains the handling of instruction execution conditions.

Condition	Description
EnableIn False (false rung)	No EnableIn False logic is provided. The raP_Tec_LgxEvent instruction must always be scanned true. In relay ladder logic, the raP_Tec_LgxEvent instruction must be by itself on an unconditional rung.
Powerup (prescan, first scan)	No SFC Prescan logic is provided.
Postscan (SFC transition)	No SFC Postscan logic is provided.

See to the Logix 5000 Controllers Add-On Instructions Programming Manual, publication 1756-PM010, for more information.

Programming Examples

The example in the Function Description section shows the basic use of the raP_Tec_LgxEvent Add-On Instruction for capturing events.

Graphic Symbols

There are no visualization files associated with this object.

Faceplates

There are no visualization files associated with this object

Notes:

FactoryTalk View Customization Tool

Overview

This customization tool lets you create a color palette to change the colors for global objects and displays.

The Color Change tool uses three types of files:

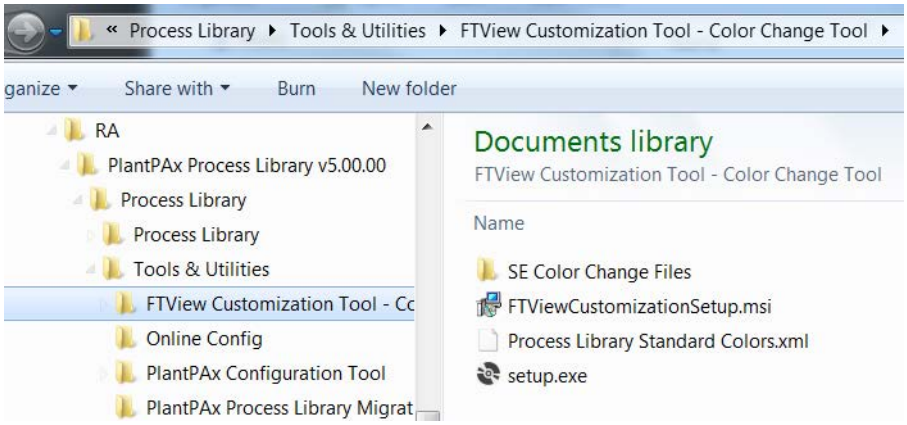
- **FactoryTalk® View Graphics .xml file:** This file is exported from the FactoryTalk View graphic (display or global object) in the View Studio software program. Once changes are made, it is imported into the View Studio software program to change the colors in the display or global object.
- **Color Association File:** This .xml file matches a color instance in the FactoryTalk View Graphics .xml file to the color palette entry. There is one Color Association File (CAXML) for each FactoryTalk View Graphics .xml file. The tool creates and maintains the CAXML file.
- **Color Palette:** This .xml file defines the colors for an application. The tool creates and maintains the .xml file. There is one color palette file for all FactoryTalk View Graphics .xml files that are being customized. If you want to change the color, it is done in the color palette.

We suggest that you make a copy of the color palette .xml file if you plan to use the color tool.

Install Tool File

Obtain the Color Change tool as part of the Library of Process Objects download from the Product Compatibility and Download Center at https://www.rockwellautomation.com/rockwellautomation/support/downloads.page.

Access the tool from the Process Library download. Choose RA>Process Library v*X.X*>Tools & Utilities>FTView Customization Tool - Color Change Tool and double-click FTViewCustomizationSetup.msi.



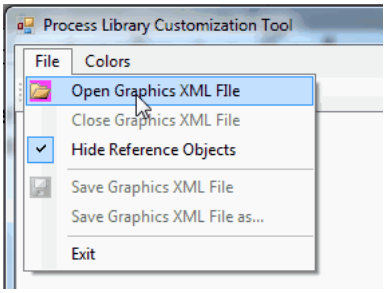
This file installs the program and adds a shortcut to the Start menu under 'PlantPAx®.'

Use the Tool with Library Objects

The download includes .xml exports for all global objects and display files in the library (for FactoryTalk View SE software). Make sure that you also download the CAXML and Process Library Standard Colors .xml files.

Follow these steps to change colors in the process library.

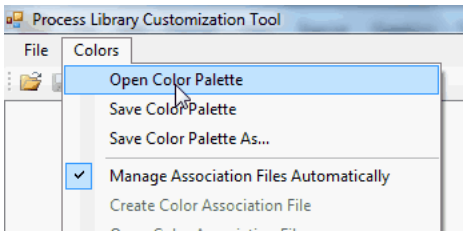
1. From the Process Library Customization Tool File menu, Select Open Graphic XML File.



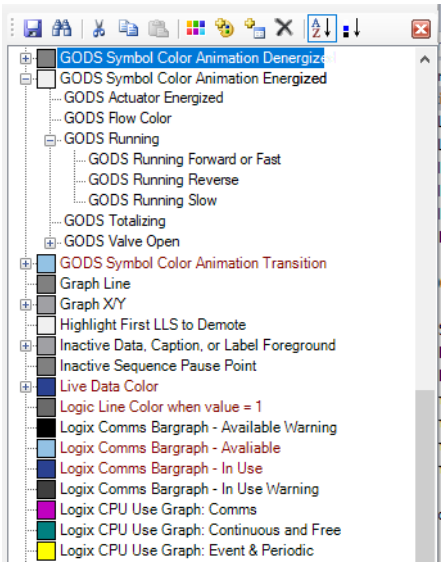
   The Open Graphics XML Files dialog box appears.

   Multiple global object and display files can be opened simultaneously from the file open dialog box.

2. Select the Colors tab and choose Open Color Palette.

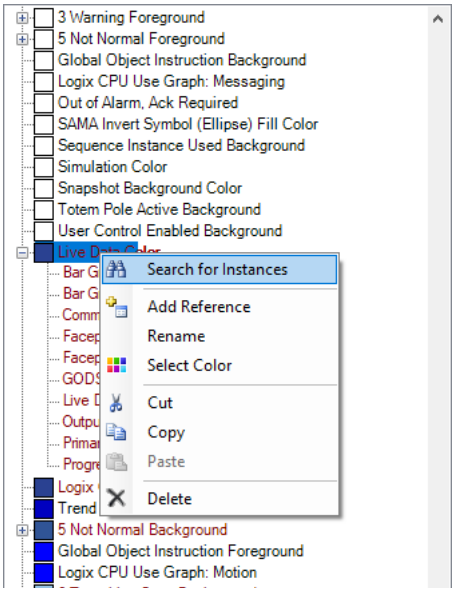


3. Select the colors that you want to change in the palette.

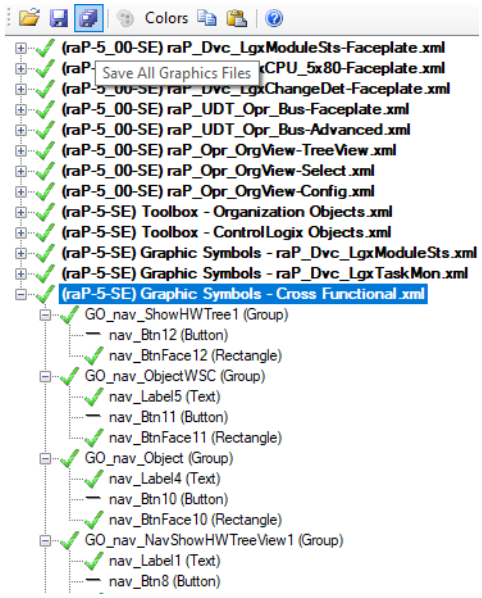


4. To select a new color, Select the Choose Color icon.
5. Repeat step 4 to change each color.

6. To see where a color is used, right-click a color and choose Search for Instances.



7. To save all graphic files (along with their association files) and the color palette file, Select Save All  .



8. Import the files into the FactoryTalk View software program.

   There are bulk import files for the displays (BatchImport_Displays_PlantPAx)Library.xml) and global objects (BatchImport_Global_PlantPAx)Library.xml).

Modifying the Color Palette

The color palette appears in a tree format that shows a parent-child relationship between colors. 'Base Colors' are shown with a color box next to them.
'Reference Colors' reference either a Base Color or another Reference Color.

By changing a Base Color, all Reference Colors under it change. For example, you can create a generic Base Color, called 'Energized', and then reference it with the Reference Color, called 'Running'.

Do not delete Color palette entries unless they are known to be unused. To see if a color palette entry is being used, right-click the color and choose 'Find Color Instances'.

Any color palette entry (Reference or Base Color) can be moved to reference another color. This action is done by simply dragging the color to be moved and dropping it on the new color to reference. When a color that has references is moved, all of its references move as well.

To make a Reference Color a Base Color, right-click the Reference Color and select 'Make Base Color' from the context menu.

Color palette entries are stored with an integer code. That integer code is used in the association file. Renaming a color palette entry does not break any existing associations. Multiple color palette entries can have the same name, but this practice is not recommended.

Follow these color palette considerations:

- Once a color palette entry is deleted and the palette is saved, the only way to restore associations is to recreate them manually.
- Object names in FactoryTalk View software usually have a number on the end. Names are considered to be similar if they are the same after the ending number is removed.

Use the Tool with Other FactoryTalk View Software Files

The color palette must be applied to FactoryTalk View software files that are not part of the Rockwell Automation® Library. Graphic elements in the file must be associated to the color palette. You must create associations and save them in a color association file. When opening an .xml graphics file, if the file already has an association file (CAXML), it is automatically opened as well. If an association file does not exist, it is created.

Follow these steps to create associations.

1. From the Process Library Customization Tool File menu, Select Open Graphic XML File.

   The Open Graphics XML Files dialog box appears.
2. Select an object from the tree on the left, and its colors appear in the center of the screen.
3. To associate a color from the palette, select the palette color and drag it to the text box next to the color display box.

   Once all colors for an object are associated with the color palette, a check appears next to the object in the tree.

   Colors that are used for the object only are displayed. For example, if an object is configured as 'Transparent', its background color does not show up in the tool. Also, instances of global objects from display files do not appear in the object tree. The tree can be configured to show instances of global objects. These objects do not have any color instances because their parent global objects control their colors.
4. Copy and paste functions have been included to allow quick creation of color associations. To use these functions, right-click the graphic object in the tree on the left and a menu appears.
   - **Copy Color Associations:** Use this function to copy the color associations for the object. If the object is a group, the color configuration for all group members is copied.

- **Paste Color Associations (this Object only):** Use this function to paste the previously copied color associations to the selected object. This option is not available if the selected object is a group that has members with color associations.
- **Paste Color Associations (to all group members):** Use this function to paste the previously copied color associations to the new object and all of its members. This option is available only if the source and destination objects are groups with members that have similar names and object types.
- **Copy and Paste Color Associations to Similar Objects with Names like 'Xxxx#'**: This option copies the selected object and searches objects with a similar name and object type. Color associations are copied to all objects with similar names and types in any of the currently open graphics files. If the objects are groups, then the group members must have similar names and object types. Be careful when you use this feature to help prevent unwanted changes.

Notes:

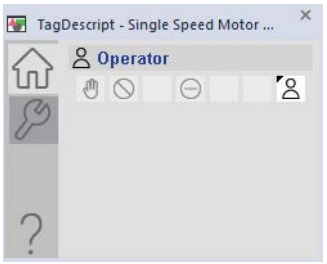
Command Sources and Device Virtualization

Command Sources

The Command Source selection determines the source of Commands and Settings for the object. For example, when the Command Source is Operator, the object processes Commands and Settings from the Operator.

Highlighted indicators on the object faceplate display show which sources have requested control. If more than one source is requesting control, multiple indicators are highlighted. The sources are shown in priority order, and the highlighted source furthest to the left has control. If that source relinquishes control, the next source in priority order assumes control of the object.

A triangle in the upper left corner (as seen in the following screenshot on the icon in the far right) indicates the "Normal" command source.



Command Source	Description
Operator	The Operator controls the object. Operator Commands, such as OCmd_Start and OCmd_Stop, and Operator Settings, such as OSet_SP and OSet_CV, from the HMI are accepted.
Program	Program logic controls the object. Program Commands, such as PCmd_Start and PCmd_Stop, and Program Settings, such as PSet_SP and PSet_CV, are accepted.
External	An external system or other external devices control the object via logic. External Commands, such as XCmd_Start and XCmd_Stop, and External Settings, such as XSet_SP, XSet_CV, from Logic are accepted. Examples of external devices and systems that may control an object include a SCADA master system or local pilot devices (push buttons, switches, pilot lights).
Override	Priority logic controls the object and supersedes Operator, Program, and External control. The Override Command Input (Inp_OvrdCmd) and other Override settings are accepted. If so configured (for example, Cfg_OvrdPermIntlk=1), bypassable interlocks and permissives are bypassed.
Maintenance	Maintenance controls the object and supersedes Operator, Program, External, and Override control. Operator Commands and Settings from the HMI are accepted. Bypassable interlocks and permissives are bypassed, and feedback timeout checks are not processed.
Out of Service	The object may be placed Out of Service by Maintenance from the HMI (Maintenance Out of Service). The object may also be placed Out of Service by scanning the instruction false (in a ladder diagram implementation) or by exposing and wiring the EnableIn input pin and setting it false (in a Function Block Diagram implementation). When the object is Out of Service, outputs are held de-energized / at zero, and alarms are inhibited.
Hand	Hardwired circuits or other logic outside the instruction controls the object, ignoring outputs of the instruction. The instruction tracks the state of the object via inputs for bumpless transfer back to another command source.

Not all Command Sources are used in every object.

PCMDSRC	Operator	Program	External	Override	Maintenance	Out of Service	Hand
raP_Opr_Area	x	x	x		x	x	
raP_Opr_Unit	x	x	x		x	x	
raP_Opr_EMGen	x	x	x		x	x	
raP_Opr_EPGen	x	x	x		x	x	

Virtualization

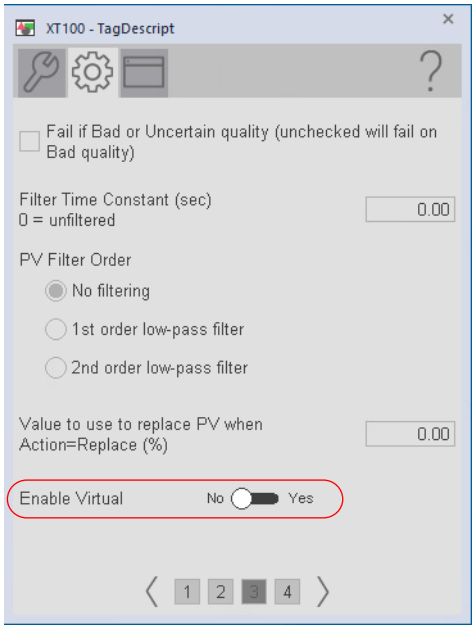
Virtualization is used with device objects to simulate operation of a device instead of controlling the actual device. Virtualization is used for such activities as system testing or operator training, where the process is shut down or not connected to the controller.

When a device is set to Physical operation, the actual field device I/O are monitored or controlled, and the field device operates normally, on-process.

When a device object is set to Virtual operation, the I/O for the field device are ignored, and the device operates in one of these manners:

- For monitored devices, such as analog and discrete inputs, a virtual process variable (PV) is provided, either by simulation logic or by entry from the HMI faceplate.
- For controlled devices, such as valves, motors and drives, the outputs are held de-energized (at zero) and the object responds in a "loopback" manner, as if an actual device were connected. So a valve object, while keeping outputs de-energized, reports valve status to the operator and to program logic as if the valve were opening and closing normally.

To select Virtual or Physical operation, go to the Advanced faceplate for the device and toggle the Virtual / Physical selector.

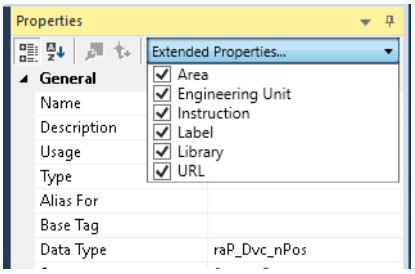


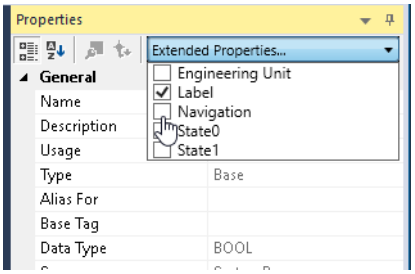
Tag Extended Properties and Default Alarm Settings

Tag extended properties must be configured to drive the text on the operations faceplate. See Logix 5000 Controllers I/O and Tag Data, publication 1756-PM004 for more information on extended tags.

Access to alarms is via <backing_tag>.@Alarms.<alarm_name>.<alarm_parameter>.

You must select the extended properties to populate for each tag and then enter the values.





raP_Dvc_LgxChangeDet

Common	
raP_Dvc_LgxChangeDet.@Description	"Logix Change Detector"
raP_Dvc_LgxChangeDet.@Area	"Area01"
raP_Dvc_LgxChangeDet.@Instruction	"raP_Dvc_LgxChangeDet"
raP_Dvc_LgxChangeDet.@Label	"Controller Name"
raP_Dvc_LgxChangeDet.@Library	"raP-5_30"
raP_Dvc_LgxChangeDet.@URL	"n/a"
raP_Dvc_LgxChangeDet.Cfg_HasMoreObj.@Navigation	" "

Alarms		Alarm Default Message	Severity
raP_Dvc_LgxChangeDet.Sts_ChangeDetected.@Label	"Logic change detected"	/*S:0 %.@Description*/: Controller logic change detected	1000

raP_Dvc_LgxCPU_5x80

Common	
raP_Dvc_LgxCPU_5x80.@Description	"Processor utilization (5380/5580, v33 and later)"
raP_Dvc_LgxCPU_5x80.@Area	"Area01"
raP_Dvc_LgxCPU_5x80.@Instruction	"raP_Dvc_LgxCPU_5x80"
raP_Dvc_LgxCPU_5x80.@Label	" "
raP_Dvc_LgxCPU_5x80.@Library	"raP-5_30"
raP_Dvc_LgxCPU_5x80.@URL	"n/a"
raP_Dvc_LgxCPU_5x80.Cfg_HasMoreObj.@Navigation	" "

raP_Dvc_LgxModuleSts

Common	
raP_Dvc_LgxModuleSts.@Description	"Logix - Module Status"
raP_Dvc_LgxModuleSts.@Area	"Area01"
raP_Dvc_LgxModuleSts.@Instruction	"raP_Dvc_LgxModuleSts"
raP_Dvc_LgxModuleSts.@Label	"Module Name"
raP_Dvc_LgxModuleSts.@Library	"raP-5_30"
raP_Dvc_LgxModuleSts.@URL	"n/a"

Alarms		Alarm Default Message	Severity
raP_Dvc_LgxModuleSts.Sts_AnyChanFault.@Label	"I/O channel fault"	/*S:0 %.@Description*/: An I/O channel is faulted	500
raP_Dvc_LgxModuleSts.Sts_ModuleFault.@Label	"I/O module fault"	/*S:0 %.@Description*/: I/O module or device connection fault	500

raP_Dvc_LgxRedun

Common	
raP_Dvc_LgxRedun.@Description	"Logix Redundant Controller Monitor"
raP_Dvc_LgxRedun.@Area	"Area01"
raP_Dvc_LgxRedun.@Instruction	"raP_Dvc_LgxRedun"
raP_Dvc_LgxRedun.@Label	"Redundant Controller"
raP_Dvc_LgxRedun.@Library	"raP-5_30"
raP_Dvc_LgxRedun.@URL	"n/a"

Alarms		Alarm Default Message	Severity
raP_Dvc_LgxRedun.Sts_SecNotRdy.@Label	"Secondary not ready"	Redundancy: secondary controller not ready to take control.	500

raP_Dvc_LgxTaskMon

Common	
raP_Dvc_LgxTaskMon.@Description	"Logix Task Monitor"
raP_Dvc_LgxTaskMon.@Area	"Area01"
raP_Dvc_LgxTaskMon.@Instruction	"raP_Dvc_LgxTaskMon"
raP_Dvc_LgxTaskMon.@Label	"Logix Task Monitor"
raP_Dvc_LgxTaskMon.@Library	"raP-5_30"
raP_Dvc_LgxTaskMon.@URL	"n/a"

Alarms		Alarm Default Message	Severity
raP_Dvc_LgxTaskMon.Sts_OverPlan.@Label	"Task scan time over plan"	Controller Task /*S:0 %Tag1*/: execution time over plan Tag1 = raP_Dvc_LgxTaskmon.Sts_sName (Task Name as configured in the controller.):STRING	500

raP_Opr_ArbitrationQ

Common	
raP_Opr_ArbitrationQ.@Description	" "
raP_Opr_ArbitrationQ.@Area	"Area01"
raP_Opr_ArbitrationQ.@Instruction	" "
raP_Opr_ArbitrationQ.@Label	" "
raP_Opr_ArbitrationQ.@Library	"raP-5_30"
raP_Opr_ArbitrationQ.@URL	"n/a"

raP_Opr_Area

Common	
raP_Opr_Area.@Description	"Area"
raP_Opr_Area.@Area	"Area01"
raP_Opr_Area.@Instruction	"raP_Opr_Area"
raP_Opr_Area.@Label	"Area"
raP_Opr_Area.@Library	"raP-5_30"
raP_Opr_Area.@URL	"n/a"
raP_Opr_Area.Cfg_HasMoreObj.@Navigation	" "

General	
raP_Opr_Area.Sts_ExtddAlms.@Label	"Extended alarm"

Alarms		Alarm Default Message	Severity
raP_Opr_Area.Sts_EStopTrip.@Label	“Emergency stop”	"/*S:0 %.@Description*/: Emergency stop	750
raP_Opr_Area.Sts_SStopTrip.@Label	“Software stop”	"/*S:0 %.@Description*/: Software stop	750

raP_Opr_EMGen

Common	
raP_Opr_EMGen.@Description	"Generic Equipment Module"
raP_Opr_EMGen.@Area	"Area01"
raP_Opr_EMGen.@Instruction	"raP_Opr_EMGen"
raP_Opr_EMGen.@Label	"Equipment Module"
raP_Opr_EMGen.@Library	"raP-5_30"
raP_Opr_EMGen.@URL	" "
raP_Opr_EMGen.Cfg_HasMoreObj.@Navigation	" "
raP_Opr_EMGen.Sts_eStep.@Navigation	“SystemStepDescriptions”
raP_Opr_EMGen.Sts_eSummary.@Navigation	"SystemSummary"
raP_Opr_EMGen.Cfg_HasDvcAlmsObjNav@Navigation	" "
raP_Opr_EMGen.Ref_SMCfgPath@Navigation	" "

General	
raP_Opr_EMGen.Cfg_HasDetailDisplay.@Navigation	" "
raP_Opr_EMGen.Sts.0.@Description	"State 1"
raP_Opr_EMGen.Sts.1.@Description	"State 2"
raP_Opr_EMGen.Sts.2.@Description	"State 3"
raP_Opr_EMGen.Sts.3.@Description	"State 4"
raP_Opr_EMGen.Sts.4.@Description	"State 5"
raP_Opr_EMGen.Sts.5.@Description	"State 6"
raP_Opr_EMGen.Sts.6.@Description	"State 7"
raP_Opr_EMGen.Sts.7.@Description	"State 8"
raP_Opr_EMGen.Sts.8.@Description	"State 9"
raP_Opr_EMGen.Sts.9.@Description	"State 10"

General	
raP_Opr_EMGen.Sts.10.@Description	"State 11"
raP_Opr_EMGen.Sts.11.@Description	"State 12"
raP_Opr_EMGen.Sts.12.@Description	"State 13"
raP_Opr_EMGen.Sts.13.@Description	"State 14"
raP_Opr_EMGen.Sts.14.@Description	"State 15"
raP_Opr_EMGen.Sts.15.@Description	"State 16"
raP_Opr_EMGen.Sts.16.@Description	"State 17"
raP_Opr_EMGen.Sts.17.@Description	"State 18"
raP_Opr_EMGen.Sts.18.@Description	"State 19"
raP_Opr_EMGen.Sts.19.@Description	"State 20"
raP_Opr_EMGen.Sts.20.@Description	"State 21"
raP_Opr_EMGen.Sts.21.@Description	"State 22"
raP_Opr_EMGen.Sts.22.@Description	"State 23"
raP_Opr_EMGen.Sts.23.@Description	"State 24"
raP_Opr_EMGen.Sts.24.@Description	"State 25"
raP_Opr_EMGen.Sts.25.@Description	"State 26"
raP_Opr_EMGen.Sts.26.@Description	"State 27"
raP_Opr_EMGen.Sts.27.@Description	"State 28"
raP_Opr_EMGen.Sts.28.@Description	"State 29"
raP_Opr_EMGen.Sts.29.@Description	"State 30"
raP_Opr_EMGen.Sts.30.@Description	"State 31"
raP_Opr_EMGen.Sts.31.@Description	"State 32"
raP_Opr_EMGen.Sts_ExtddAlms.@Label	"Extended alarm"
raP_Opr_EMGen.XCmd.0.@Description	"Command 1"
raP_Opr_EMGen.XCmd.1.@Description	"n/a"
raP_Opr_EMGen.XCmd.2.@Description	"n/a"
raP_Opr_EMGen.XCmd.3.@Description	"n/a"
raP_Opr_EMGen.XCmd.4.@Description	"n/a"
raP_Opr_EMGen.XCmd.5.@Description	"n/a"
raP_Opr_EMGen.XCmd.6.@Description	"n/a"
raP_Opr_EMGen.XCmd.7.@Description	"n/a"
raP_Opr_EMGen.XCmd.8.@Description	"n/a"
raP_Opr_EMGen.XCmd.9.@Description	"n/a"
raP_Opr_EMGen.XCmd.10.@Description	"n/a"
raP_Opr_EMGen.XCmd.11.@Description	"n/a"
raP_Opr_EMGen.XCmd.12.@Description	"n/a"
raP_Opr_EMGen.XCmd.13.@Description	"n/a"
raP_Opr_EMGen.XCmd.14.@Description	"n/a"
raP_Opr_EMGen.XCmd.15.@Description	"n/a"
raP_Opr_EMGen.XCmd.16.@Description	"n/a"
raP_Opr_EMGen.XCmd.17.@Description	"n/a"
raP_Opr_EMGen.XCmd.18.@Description	"n/a"
raP_Opr_EMGen.XCmd.19.@Description	"n/a"
raP_Opr_EMGen.XCmd.20.@Description	"n/a"
raP_Opr_EMGen.XCmd.21.@Description	"n/a"
raP_Opr_EMGen.XCmd.22.@Description	"n/a"
raP_Opr_EMGen.XCmd.23.@Description	"n/a"
raP_Opr_EMGen.XCmd.24.@Description	"n/a"
raP_Opr_EMGen.XCmd.25.@Description	"n/a"
raP_Opr_EMGen.XCmd.26.@Description	"n/a"
raP_Opr_EMGen.XCmd.27.@Description	"n/a"
raP_Opr_EMGen.XCmd.28.@Description	"n/a"

General	
raP_Opr_EMGen.XCmd.29.@Description	"n/a"
raP_Opr_EMGen.XCmd.30.@Description	"n/a"
raP_Opr_EMGen.XCmd.31.@Description	"n/a"

Alarms		Alarm Default Message	Severity
raP_Opr_EMGen.Sts_DvcAlms.@Label	"Device alarm"	"/*S:0 %.@Description*/: Device alarm	500
raP_Opr_EMGen.Sts_IntlkTrip.@Label	"Interlock trip"	"/*S:0 %.@Description*/: Interlock trip	500
raP_Opr_EMGen.Sts_RptData.@Label	"Report data not collected"	"/*S:0 %.@Description*/: Report data	500

raP_Opr_EPGen

Common	
raP_Opr_EPGen.@Description	"Generic Equipment Phase"
raP_Opr_EPGen.@Area	"Area01"
raP_Opr_EPGen.@Instruction	"raP_Opr_EPGen"
raP_Opr_EPGen.@Label	"Equipment Phase"
raP_Opr_EPGen.@Library	"raP-5_30"
raP_Opr_EPGen.@URL	"n/a"
raP_Opr_EPGen.Cfg_HasMoreObj.@Navigation	" "
raP_Opr_EPGen.Sts_eStep.@Navigation	"SystemStepDescriptions"
raP_Opr_EPGen.Sts_eSummary.@Navigation	"SystemSummary"
raP_Opr_EPGen.Cfg_HasDvcAlmsObjNav@Navigation	" "

General	
#2.Cfg_HasDetailDisplay.@Navigation	" "
#2.Sts_ExtddAlms.@Label	"Extended alarm"

Alarms		Alarm Default Message	Severity
raP_Opr_EPGen.Sts_DvcAlms.@Label	"Device alarm"	"/*S:0 %.@Description*/: Device alarm	500
raP_Opr_EPGen.Sts_IntlkTrip.@Label	"Interlock trip"	"/*S:0 %.@Description*/: Interlock trip	500
raP_Opr_EPGen.Sts_RptData.@Label	"Report data not collected"	"/*S:0 %.@Description*/: Report data	500

raP_Opr_ExtddAlm

Common	
raP_Opr_ExtddAlm.@Description	"Extended alarm"
raP_Opr_ExtddAlm.@Area	"Area01"
raP_Opr_ExtddAlm.@Instruction	"raP_Opr_ExtddAlm"
raP_Opr_ExtddAlm.@Label	"Alarm"
raP_Opr_ExtddAlm.@Library	"raP-5_30"
raP_Opr_ExtddAlm.@URL	"n/a"

raP_Opr_OrgScan

Common	
raP_Opr_OrgScan.@Description	" "
raP_Opr_OrgScan.@Area	"Area01"
raP_Opr_OrgScan.@Instruction	" "
raP_Opr_OrgScan.@Label	" "
raP_Opr_OrgScan.@Library	"raP-5_30"
raP_Opr_OrgScan.@URL	"n/a"

raP_Opr_OrgView

Common	
raP_Opr_OrgView.@Description	" "
raP_Opr_OrgView.@Area	"Area01"
raP_Opr_OrgView.@Instruction	" "
raP_Opr_OrgView.@Label	" "
raP_Opr_OrgView.@Library	"raP-5_30"
raP_Opr_OrgView.@URL	"n/a"

raP_Opr_Prompt

Common	
raP_Opr_Prompt.@Description	"Operator Prompt"
raP_Opr_Prompt.@Area	"Area01"
raP_Opr_Prompt.@Instruction	"raP_Opr_Prompt"
raP_Opr_Prompt.@Label	"Prompt"
raP_Opr_Prompt.@Library	"raP-5_30"
raP_Opr_Prompt.@URL	"n/a"
raP_Opr_Prompt.Cfg_HasMoreObj.@Navigation	" "

Alarms		Alarm Default Message	Severity
raP_Opr_Prompt.Sts_AlertTimeOut.@Label	“Prompt timed out"	/*S:0 %.@Description*/ Prompt time out alarm	500

raP_Opr_Prompt_Core

Common	
raP_Opr_Prompt.@Description	"Operator Prompt"
raP_Opr_Prompt.@Area	"Area01"
raP_Opr_Prompt.@Instruction	"raP_Opr_Prompt"
raP_Opr_Prompt.@Label	"Prompt"
raP_Opr_Prompt.@Library	"raP-5_30"
raP_Opr_Prompt.@URL	"n/a"
raP_Opr_Prompt.Cfg_HasMoreObj.@Navigation	" "
raP_Opr_Unit.Sts_eMtrl.@Navigation	"SystemMaterialNames"
raP_Opr_Unit.Sts_eSummary.@Navigation	"SystemSummary"

raP_Opr_Unit

Common	
raP_Opr_Unit.@Description	"Unit"
raP_Opr_Unit.@Area	"Area01"
raP_Opr_Unit.@Instruction	"raP_Opr_Unit"
raP_Opr_Unit.@Label	"Unit"
raP_Opr_Unit.@Library	"raP-5_30"
raP_Opr_Unit.@URL	"n/a"
raP_Opr_Unit.Cfg_HasMoreObj.@Navigation	" "

General	
raP_Opr_Unit.Sts.0.@Description	"State 0"
raP_Opr_Unit.Sts.1.@Description	"State 1"
raP_Opr_Unit.Sts.2.@Description	"State 2"
raP_Opr_Unit.Sts.3.@Description	"State 3"
raP_Opr_Unit.Sts_ExtddAlms.@Label	"Extended alarm"
raP_Opr_Unit.XCmd.0.@Description	"Group Command 0"
raP_Opr_Unit.XCmd.1.@Description	"n/a"
raP_Opr_Unit.XCmd.2.@Description	"n/a"
raP_Opr_Unit.XCmd.3.@Description	"n/a"
raP_Opr_Unit.Val_Actl.@EngineeringUnit	"Kg"

Alarms		Alarm Default Message	Severity
raP_Opr_Unit.Sts_EStopTrip.@Label	"Emergency stop"	"/*S:0 %.@Description*/: Emergency stop	750
raP_Opr_Unit.Sts_GroupCmd1Fail.@Label	"Group Command 1 Failed"	"/*S:0 %.@Description*/: Group Command 1 Failed	500
raP_Opr_Unit.Sts_GroupCmd2Fail.@Label	"Group Command 2 Failed"	"/*S:0 %.@Description*/: Group Command 2 Failed	500
raP_Opr_Unit.Sts_GroupCmd3Fail.@Label	"Group Command 3 Failed"	"/*S:0 %.@Description*/: Group Command 3 Failed	500
raP_Opr_Unit.Sts_GroupCmd4Fail.@Label	"Group Command 4 Failed"	"/*S:0 %.@Description*/: Group Command 4 Failed	500
raP_Opr_Unit.Sts_IntlkTrip.@Label	"Interlock trip"	"/*S:0 %.@Description*/: Interlock trip	500
raP_Opr_Unit.Sts_SStopTrip.@Label	"Software stop"	"/*S:0 %.@Description*/: Software stop	750

raP_Tec_ParRpt

Common	
raP_Tec_ParRpt_PAR_XX.@Description	"Parameter"
raP_Tec_ParRpt_PAR_XX.@Area	"Area01"
raP_Tec_ParRpt_PAR_XX.@Instruction	"raP_Tec_ParRpt"
raP_Tec_ParRpt_PAR_XX.@Label	"Parameter Label"
raP_Tec_ParRpt_PAR_XX.@Library	"raP-5_30"
raP_Tec_ParRpt_PAR_XX.@URL	"n/a"
raP_Tec_ParRpt_PAR_XX.@EngineeringUnit	"%"
raP_Tec_ParRpt_RPT_XX.@Description	"Report"
raP_Tec_ParRpt_RPT_XX.@Area	"Area01"
raP_Tec_ParRpt_RPT_XX.@Instruction	"raP_Tec_ParRpt"
raP_Tec_ParRpt_RPT_XX.@Label	Report Label"
raP_Tec_ParRpt_RPT_XX.@Library	"raP-5_30"
raP_Tec_ParRpt_RPT_XX.@URL	"n/a"
raP_Tec_ParRpt_RPT_XX.@EngineeringUnit	"%"

Notes:

HMI Navigation

Tag Naming Conventions

The following table describes the tag naming conventions and syntax to follow when programming to achieve navigation among HMI Faceplate objects.

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
PAH	–	–	–
PAI	Nav to HART Device	PAH : _Dvc	If PAI Tag Name = XT100 PAH tag name = XT100_Dvc
	Nav to EH Objects	raP_Dvc_EH_Flowmeter/Sensor: _Dvc	If PAI tag name = XT100 EH Object tag name = XT100_Dvc
	Nav to FF/PA Objects	raP_Dvc_AP_FFLink/PALink: _Dvc raP_Dvc_EN2FFR/EN2PAR: _Dvc	If PAI tag name = XT100 FF/PA Object tag name = XT100_Dvc
PAID	–	–	–
PAIM	–	–	–
PAO			If PAO Tag Name = XC100
	Nav to HART Device	PAH : _Dvc	PAH tag name = XC100_Dvc
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = XC100_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank0 tag name = XC100_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank0 tag name = XC100_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank0 tag name = XC100_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank0 tag name = XC100_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank0 tag name = XC100_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank0 tag name = XC100_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank0 tag name = XC100_Intlk_7
PBL	–	–	–
PDBC	–	–	–
PDI	–	–	–
PDO			If PDO Tag Name = XY100
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = XY100_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = XY100_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = XY100_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = XY100_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = XY100_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = XY100_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = XY100_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = XY100_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM tag name = XY100_Perm
PDOSE	–	–	–
PFO	–	–	–
PHLS	–	–	–

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
PLLS			If PLLS Tag Name = GRPMTR100
	PLLS Ref_Tag(InOut)	PLLS Ref_Motors (InOut) : _Motors	PLLS Ref_Motors (InOut) = GRPMTR100_Motors
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = GRPMTR100_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = GRPMTR100_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = GRPMTR100_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = GRPMTR100_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = GRPMTR100_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = GRPMTR100_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = GRPMTR100_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = GRPMTR100_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM tag name = GRPMTR100_Perm
PINTLK			If PDO Tag Name = XY100
	PINTLK (InOut)_Intlk_BankSts	PINTLK Ref_IntlkBankSts (InOut) : _Intlk_BankSts	PINTLK (InOut) - XY100_Intlk_BankSts
PMTR			If PMTR = MT321
	Device Reference Control Set	PMTR Ref_Ctrl_Set (InOut) : _CtrlSet	PMTR Ref_Ctrl_Set (InOut) = MT321_CtrlSet
	Device Reference Control Commands	PMTR Ref_Ctrl_Cmd (InOut) : _CtrlCmd	PMTR Ref_Ctrl_Cmd (InOut) = MT321_CtrlCmd
	Device Reference Control Commands Status	PMTR Ref_Ctrl_Sts (InOut) : _CtrlSts	PMTR Ref_Ctrl_Sts (InOut) = MT321_CtrlSts
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = MT321_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = MT321_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = MT321_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = MT321_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = MT321_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = MT321_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = MT321_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = MT321_Intlk_7
	Nav to Permissive 1	PPERM : _1Perm	PPERM 1 tag name = MT321_1Perm
	Nav to Permissive 2	PPERM : _2Perm	PPERM 2 tag name = MT321_2Perm
	Nav to RunTime	PRT : _RunTime	PRT tag name = MT321_RunTime
	Nav to Restart Inhibit	PRI : _ResInh	PRI tag name = MT321_ResInh
	Nav to Device Object	Device Object : _Dvc	Device Object tag name = MT321_Dvc
PPERM	—	—	—
PPID			If PPID Tag Name = XIC700:
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = XIC700_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = XIC700_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = XIC700_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = XIC700_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = XIC700_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = XIC700_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = XIC700_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = XIC700_Intlk_7
PPTC	—	—	—
PRI	—	—	—
PRT	—	—	—
PTST			If PTST Tag Name = QI102
	Calibration Table Reference	PTST Cfg_CalTbl (InOut) : _CalTable	PTST Cfg_CalTable (InOut) tag name = QI102_CalTable

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
PVLV			If PVLV : XV110
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = XV110_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = XV110_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = XV110_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = XV110_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = XV110_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = XV110_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = XV110_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = XV110_Intlk_7
	Nav to Permissive 1 (Motorized Valve)	PPERM : _Pos1Perm (Motorized Valve)	PPERM 1 tag name = XV110_Pos1Perm (Motorized Valve)
	Nav to Permissive 2 (Solenoid and Motorized Valve)	PPERM : _Pos2Perm (Solenoid and Motorized Valve)	PPERM 2 tag name = XV110_Pos2Perm (Solenoid and Motorized Valve)
	Nav to Valve Statistics	PVLVS : _ValveStats	PVLVS tag name = XV110_ValveStats
PVLVS	—	—	—
PVSD			If PVSD : MT390
	Device Reference Control Set	PVSD Ref_Ctrl_Set (InOut) : _CtrlSet	PVSD Ref_Ctrl_Set (InOut) = MT390_CtrlSet
	Device Reference Control Commands	PVSD Ref_Ctrl_Cmd (InOut) : _CtrlCmd	PVSD Ref_Ctrl_Cmd (InOut) = MT390_CtrlCmd
	Device Reference Control Commands Status	PVSD Ref_Ctrl_Sts (InOut) : _CtrlSts	PVSD Ref_Ctrl_Sts (InOut) = MT390_CtrlSts
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = MT390_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = MT390_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = MT390_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = MT390_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = MT390_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = MT390_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = MT390_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = MT390_Intlk_7
	Nav to Forward Permissive	PPERM : _FwdPerm	PPERM Forward tag name = MT390_FwdPerm
	Nav to Reverse Permissive	PPERM : _RevPerm	PPERM Reverse tag name = MT390_RevPerm
	Nav to RunTime	PRT : _RunTime	PRT tag name = MT390_RunTime
	Nav to Restart Inhibit	PRI : _ResInh	PRI tag name = MT390_ResInh
	Nav to Device Object	Device Object : _Dvc	Device Object tag name = MT390_Dvc
PNPOS			If PNPOS Tag Name = NP0100
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = NP0100_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = NP0100_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = NP0100_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = NP0100_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = NP0100_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = NP0100_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = NP0100_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = NP0100_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM tag name = NP0100_Perm

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
PVLVMP			If PVLVMP = XV120
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = XV120_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = XV120_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = XV120_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = XV120_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = XV120_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = XV120_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = XV120_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = XV120_Intlk_7
	Nav to Open Interlock Bank 0	PINTLK : _OpenIntlk_0	PINTLK Bank0 tag name = XV120_OpenIntlk_0
	Nav to Open Interlock Bank 1	: _OpenIntlk_1	PINTLK Bank1 tag name = XV120_OpenIntlk_1
	Nav to Open Interlock Bank 2	: _OpenIntlk_2	PINTLK Bank2 tag name = XV120_OpenIntlk_2
	Nav to Open Interlock Bank 3	: _OpenIntlk_3	PINTLK Bank3 tag name = XV120_OpenIntlk_3
	Nav to Open Interlock Bank 4	: _OpenIntlk_4	PINTLK Bank4 tag name = XV120_OpenIntlk_4
	Nav to Open Interlock Bank 5	: _OpenIntlk_5	PINTLK Bank5 tag name = XV120_OpenIntlk_5
	Nav to Open Interlock Bank 6	: _OpenIntlk_6	PINTLK Bank6 tag name = XV120_OpenIntlk_6
	Nav to Open Interlock Bank 7	: _OpenIntlk_7	PINTLK Bank7 tag name = XV120_OpenIntlk_7
	Nav to Upper Seat Interlock Bank 0	PINTLK : _UpperIntlk_0	PINTLK Bank0 tag name = XV120_UpperIntlk_0
	Nav to Upper Seat Interlock Bank 1	: _UpperIntlk_1	PINTLK Bank1 tag name = XV120_UpperIntlk_1
	Nav to Upper Seat Interlock Bank 2	: _UpperIntlk_2	PINTLK Bank2 tag name = XV120_UpperIntlk_2
	Nav to Upper Seat Interlock Bank 3	: _UpperIntlk_3	PINTLK Bank3 tag name = XV120_UpperIntlk_3
	Nav to Upper Seat Interlock Bank 4	: _UpperIntlk_4	PINTLK Bank4 tag name = XV120_UpperIntlk_4
	Nav to Upper Seat Interlock Bank 5	: _UpperIntlk_5	PINTLK Bank5 tag name = XV120_UpperIntlk_5
	Nav to Upper Seat Interlock Bank 6	: _UpperIntlk_6	PINTLK Bank6 tag name = XV120_UpperIntlk_6
	Nav to Upper Seat Interlock Bank 7	: _UpperIntlk_7	PINTLK Bank7 tag name = XV120_UpperIntlk_7
	Nav to Lower Seat Interlock Bank 0	PINTLK : _LowerIntlk_0	PINTLK Bank0 tag name = XV120_LowerIntlk_0
	Nav to Lower Seat Interlock Bank 1	: _LowerIntlk_1	PINTLK Bank1 tag name = XV120_LowerIntlk_1
	Nav to Lower Seat Interlock Bank 2	: _LowerIntlk_2	PINTLK Bank2 tag name = XV120_LowerIntlk_2
	Nav to Lower Seat Interlock Bank 3	: _LowerIntlk_3	PINTLK Bank3 tag name = XV120_LowerIntlk_3
	Nav to Lower Seat Interlock Bank 4	: _LowerIntlk_4	PINTLK Bank4 tag name = XV120_LowerIntlk_4
	Nav to Lower Seat Interlock Bank 5	: _LowerIntlk_5	PINTLK Bank5 tag name = XV120_LowerIntlk_5
	Nav to Lower Seat Interlock Bank 6	: _LowerIntlk_6	PINTLK Bank6 tag name = XV120_LowerIntlk_6
	Nav to Lower Seat Interlock Bank 7	: _LowerIntlk_7	PINTLK Bank7 tag name = XV120_LowerIntlk_7
	Nav to Cavity Interlock Bank 0	PINTLK : _CavityIntlk_0	PINTLK Bank0 tag name = XV120_CavityIntlk_0
	Nav to Cavity Interlock Bank 1	: _CavityIntlk_1	PINTLK Bank1 tag name = XV120_CavityIntlk_1
	Nav to Cavity Interlock Bank 2	: _CavityIntlk_2	PINTLK Bank2 tag name = XV120_CavityIntlk_2
	Nav to Cavity Interlock Bank 3	: _CavityIntlk_3	PINTLK Bank3 tag name = XV120_CavityIntlk_3
	Nav to Cavity Interlock Bank 4	: _CavityIntlk_4	PINTLK Bank4 tag name = XV120_CavityIntlk_4
	Nav to Cavity Interlock Bank 5	: _CavityIntlk_5	PINTLK Bank5 tag name = XV120_CavityIntlk_5
	Nav to Cavity Interlock Bank 6	: _CavityIntlk_6	PINTLK Bank6 tag name = XV120_CavityIntlk_6
	Nav to Cavity Interlock Bank 7	: _CavityIntlk_7	PINTLK Bank7 tag name = XV120_CavityIntlk_7
	Nav to Valve Statistics	PVLVMP : _ValveStats	PVLVS tag name = XV120_ValveStats

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
PD4SD			If PD4SD : D4SD100
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = D4SD100_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = D4SD100_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = D4SD100_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = D4SD100_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = D4SD100_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = D4SD100_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = D4SD100_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = D4SD100_Intlk_7
	Nav to Permissive 0	PPERM : _0Perm	PPERM 0 tag name = D4SD100_Perm
	Nav to Permissive 1	PPERM : _1Perm	PPERM 1 tag name = D4SD100_Perm
	Nav to Permissive 2	PPERM : _2Perm	PPERM 2 tag name = D4SD100_Perm
	Nav to Permissive 3	PPERM : _3Perm	PPERM 3 tag name = D4SD100_Perm
	Nav to Valve Statistics	PD4SD : _ValveStats	PVLVS tag name : D4SD100_ValveStats
raP_Opr_Area			If raP_Opr_Area = Area01
	Nav to Extended Alarms	raP_Opr_ExtddAlm: _ExtddAlm_00 ... _ExtddAlm_32	raP_Opr_ExtddAlm: Area01_ExtddAlm_00 ... _ExtddAlm_32
raP_Opr_EMGen			If raP_Opr_EMGen Tag Name = eTK101
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = eTK101_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = eTK101_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = eTK101_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = eTK101_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = eTK101_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = eTK101_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = eTK101_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = eTK101_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM tag name = eTK101_Perm
	Nav to Extended Alarms	raP_Opr_ExtddAlm: _ExtddAlm_00 ... _ExtddAlm_32	raP_Opr_ExtddAlm tag name = eTK101_ExtddAlm_00 ... _ExtddAlm_32
	Nav Parameters	raP_Tec_ParRpt: _PAR_00 ... _PAR_496	raP_Tec_ParRpt tag name = eTK101_PAR_00 ... _PAR_496
	Nav Reports	raP_Tec_ParRpt: _RPT_00 ... _RPT_496	raP_Tec_ParRpt tag name = eTK101_RPT_00 ... _RPT_496
raP_Opr_EPGen			If raP_Opr_EPGen Tag Name = epAG1001
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = epAG1001_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = epAG1001_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = epAG1001_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = epAG1001_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = epAG1001_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = epAG1001_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = epAG1001_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = epAG1001_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM tag name = eTK101_Perm
	Nav to Extended Alarms	raP_Opr_ExtddAlm: _ExtddAlm_00 ... _ExtddAlm_32	raP_Opr_ExtddAlm tag name = epAG1001_ExtddAlm_00 ... _ExtddAlm_32
	Nav Parameters	raP_Tec_ParRpt: _PAR_00 ... _PAR_496	raP_Tec_ParRpt tag name = epAG1001_PAR_00 ... _PAR_496
	Nav Reports	raP_Tec_ParRpt: _RPT_00 ... _RPT_496	raP_Tec_ParRpt tag name = epAG1001_RPT_00 ... _RPT_496
raP_Opr_ExtddAlm	Extended Alarms	—	—

Instruction Tag Reference/ Navigation Syntax			
Instruction	**Navigation / References**	**Navigation / Reference Tag Name Syntax**	**Navigation / Reference Tag Name Example**
raP_Opr_Unit			If raP_Opr_Unit Tag Name = GroupControl
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 tag name = GroupControl_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 tag name = GroupControl_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 tag name = GroupControl_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 tag name = GroupControl_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 tag name = GroupControl_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 tag name = GroupControl_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 tag name = GroupControl_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 tag name = GroupControl_Intlk_7
	Nav to Permissive	PPERM : _Perm PPERM : 1_Perm PPERM : 2_Perm PPERM : 3_Perm PPERM : 4_Perm	PPERM tag name = GroupControl_Perm PPERM tag name = GroupControl_1Perm PPERM tag name = GroupControl_2Perm PPERM tag name = GroupControl_3Perm PPERM tag name = GroupControl_4Perm
	Nav to Extended Alarms	raP_Opr_ExtddAlm: _ExtddAlm_00 ... _ExtddAlm_32	raP_Opr_ExtddAlm tag name = GroupControl_ExtddAlm_00 ... _ExtddAlm_32
	Nav Parameters	raP_Tec_ParRpt: _PAR_00 ... _PAR_496	raP_Tec_ParRpt tag name = epAG1001_PAR_00 ... _PAR_496
	Nav Reports	raP_Tec_ParRpt: _RPT_00 ... _RPT_496	raP_Tec_ParRpt tag name = epAG1001_RPT_00 ... _RPT_496
raP_Tec_ParRpt	Parameters and Reports	—	—
raP_Opr_Prompt			If raP_Opr_Prompt Tag Name = MyPrompt
	Prompt Instance Configuration Data	raP_Opr_Prompt Prompts (InOut) : _Prompts	raP_Opr_Prompt Prompts (InOut) Tagname = MyPrompt _Prompts
		raP_Opr_Prompt RespData (InOut) : _ResponseData	raP_Opr_Prompt RespData (InOut) Tagname = MyPrompt_ResponseData
raP_Opr_Seq			If raP_Opr_Seq : Seq_101
	Reference Sequencer Step	raP_Opr_Seq Ref_Steps (InOut) : _Steps	raP_Opr_Seq Ref_Steps (InOut) : Seq_101_Steps
	Nav to Interlock Bank 0	PINTLK : _Intlk_0	PINTLK Bank0 Tagname = Seq_101_Intlk_0
	Nav to Interlock Bank 1	: _Intlk_1	PINTLK Bank1 Tagname = Seq_101_Intlk_1
	Nav to Interlock Bank 2	: _Intlk_2	PINTLK Bank2 Tagname = Seq_101_Intlk_2
	Nav to Interlock Bank 3	: _Intlk_3	PINTLK Bank3 Tagname = Seq_101_Intlk_3
	Nav to Interlock Bank 4	: _Intlk_4	PINTLK Bank4 Tagname = Seq_101_Intlk_4
	Nav to Interlock Bank 5	: _Intlk_5	PINTLK Bank5 Tagname = Seq_101_Intlk_5
	Nav to Interlock Bank 6	: _Intlk_6	PINTLK Bank6 Tagname = Seq_101_Intlk_6
	Nav to Interlock Bank 7	: _Intlk_7	PINTLK Bank7 Tagname = Seq_101_Intlk_7
	Nav to Permissive	PPERM : _Perm	PPERM Tagname = Seq_101_Perm
	Nav to Boolean Input	raP_Opr_SeqBoolInp : _BoolInp	raP_Opr_SeqBoolInp Tagname = Seq_101_BoolInp
	Reference Boolean Input Sequencer Step	raP_Opr_SeqBoolInp Ref_Steps (InOut) : _Steps	raP_Opr_SeqBoolInp Ref_Steps (InOut) Tagname = Seq_101_Steps
	Nav to Boolean Output	raP_Opr_SeqBoolOut : _BoolOut	raP_Opr_SeqBoolOut Tagname = Seq_101_BoolOut
	Reference Boolean Output Sequencer Step	raP_Opr_SeqBoolOut Ref_Steps (InOut) : _Steps	raP_Opr_SeqBoolOut Ref_Steps (InOut) Tagname = Seq_101_Steps
	Nav to Real Output	raP_Opr_SeqRealOut : _RealOut	raP_Opr_SeqRealOut Tagname = Seq_101_RealOut
	Reference Real Output Sequencer Step	raP_Opr_SeqRealOut Ref_Steps (InOut) : _Steps	raP_Opr_SeqRealOut Ref_Steps (InOut) Tagname = Seq_101_Steps
	Nav to Prompt Core	raP_Opr_Prompt_Core : _Prompt	raP_Opr_Prompt_Core Tagname = Seq_101_Prompt
	Reference Prompt Core Prompt Instance Configuration Data	raP_Opr_Prompt_Core Prompts (InOut) : _Prompts	raP_Opr_Prompt_Core Prompts (InOut) Tagname = Seq_101_Prompts
	Reference Prompt Core Prompt Response Data	raP_Opr_Prompt_Core RespData (InOut) : _ResponseData	raP_Opr_Prompt_Core RespData (InOut) Tagname = Seq_101_ResponseData

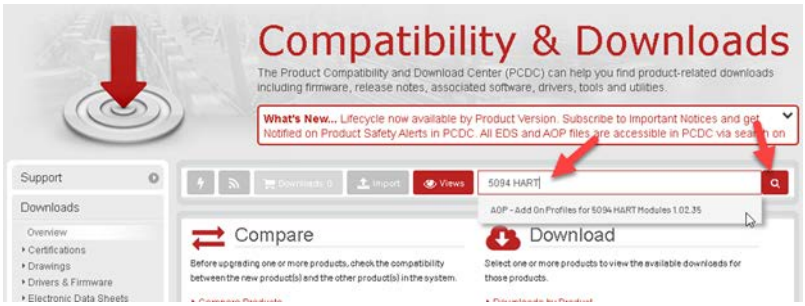
5094-IF8IH to PAH Configuration Example

This appendix describes how to configure a HART device using a newer HART I/O module, such as the 5094-IF8IH, and the PAH instruction, in a PlantPAx® 5.0 system. This example requires a system that meets PlantPAx 5.0 system requirements, including using Version 33 or later of Studio 5000 Logix Designer® software.

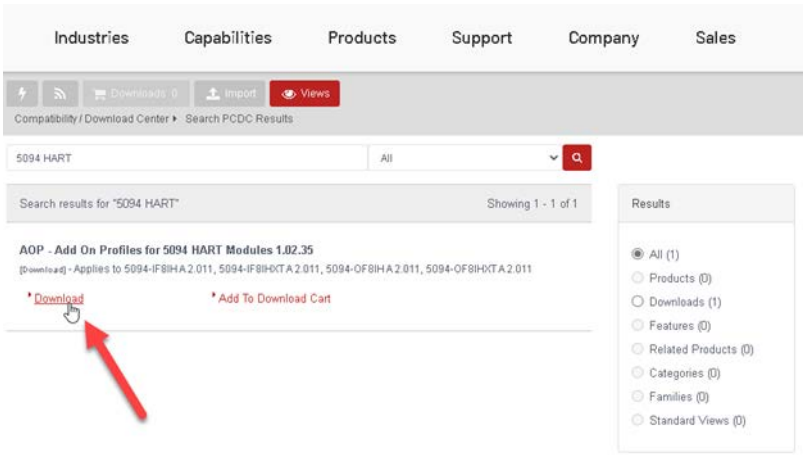
Download and install the 5094 HART Analog Add-On Profile

The Add-on Profile can be accessed from the Product Compatibility and Download Center.

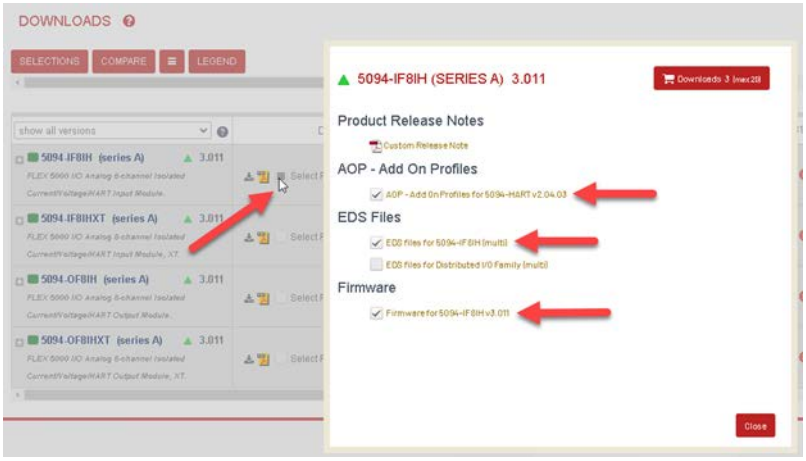
1. Search for "5094-HART".



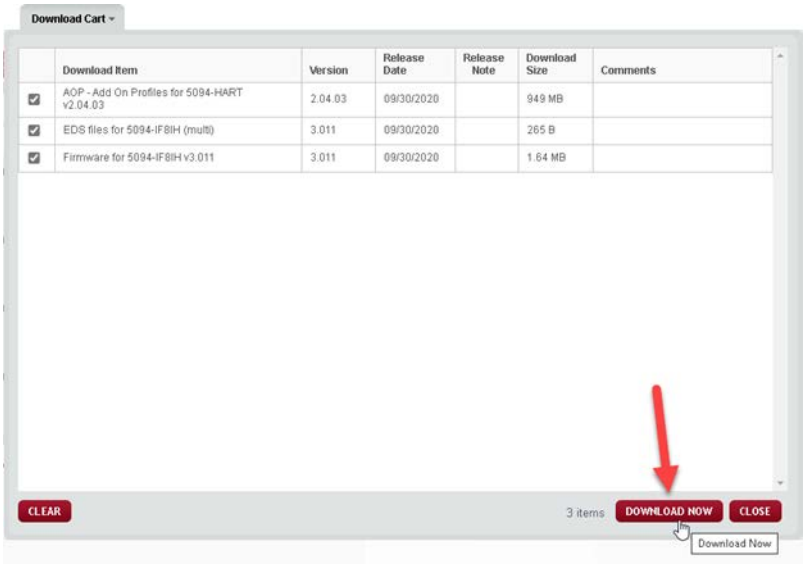
2. Select Download.



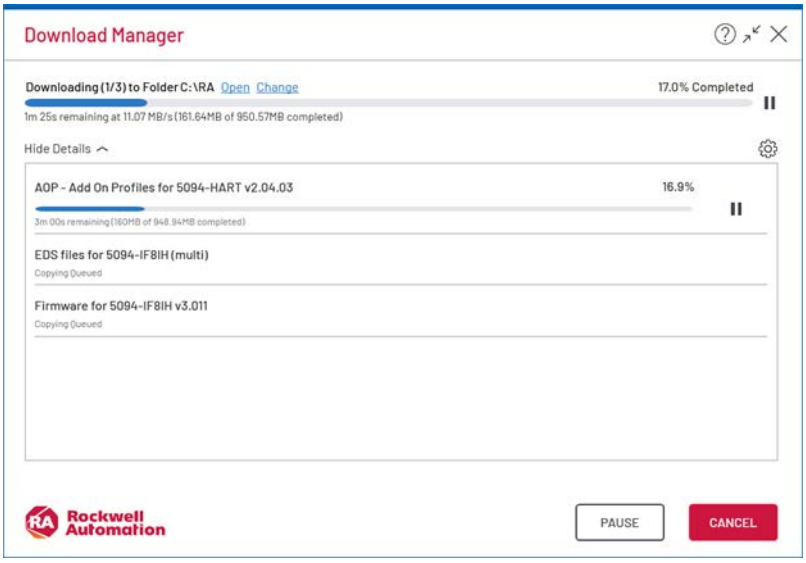
3. Select Files, then select the Add-on Profiles, EDS Files, and Firmware.



4. Select Download Now.



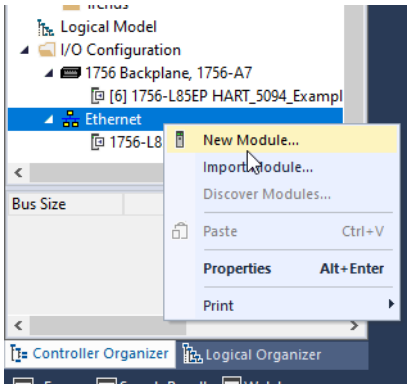
The files are downloaded into a zip file using the download manager.



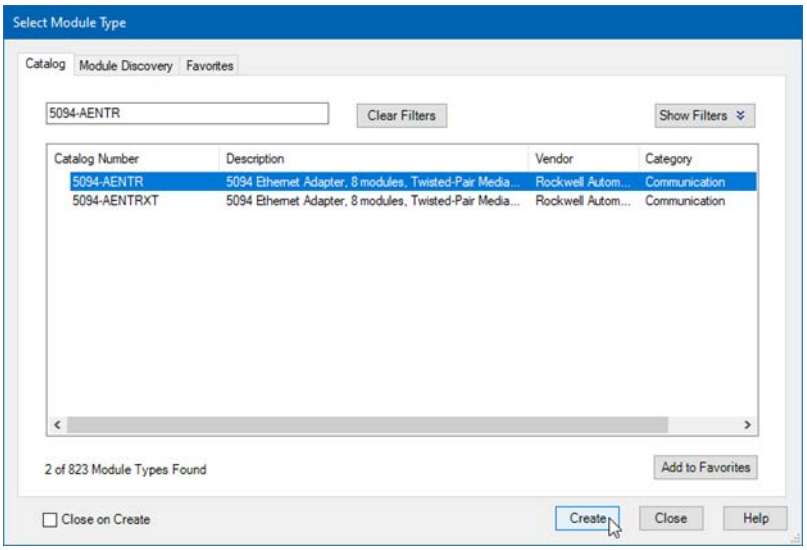
5. Extract the files from the ZIP folder.
6. Run mpsetup.exe as Administrator.

Add the 5094 Adapter Module to the Project I/O Configuration

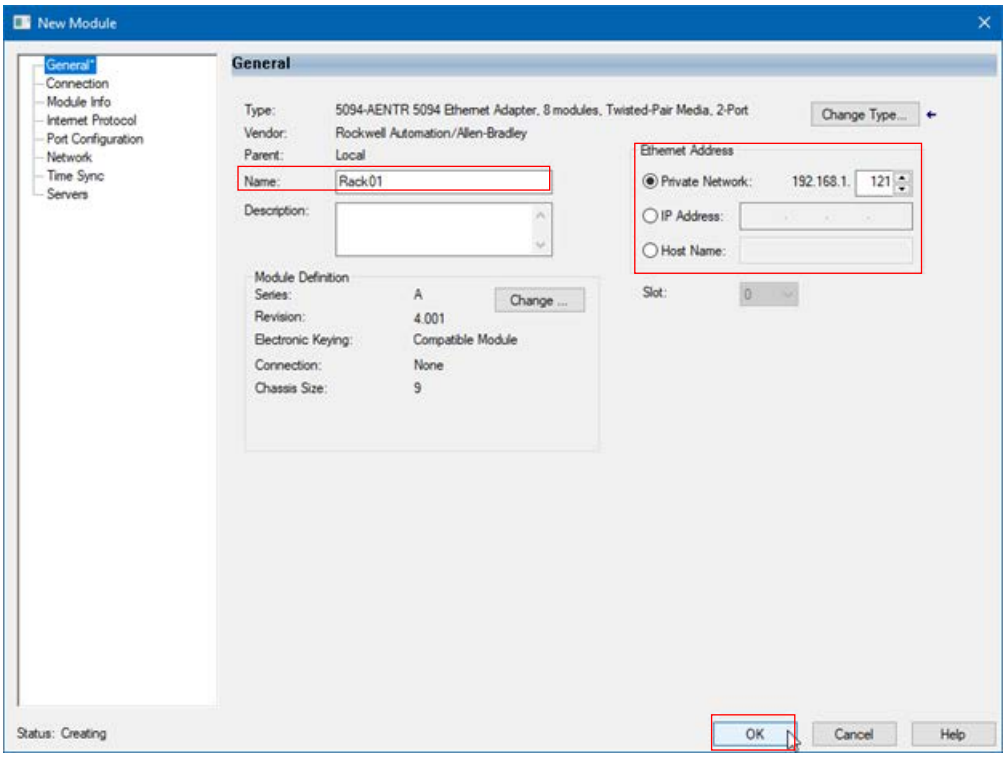
1. In the controller Organizer for your project, select the Ethernet network to be used to communicate with the 5094 I/O. Right-click and select "New Module..."



2. Select the catalog number of the 5094 adapter that you are using and Create.

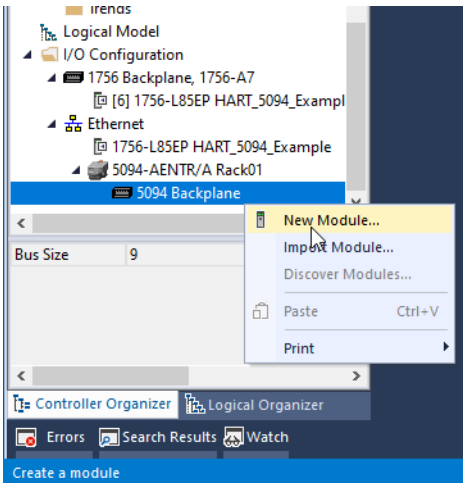


3. Enter the name and IP address for the adapter.

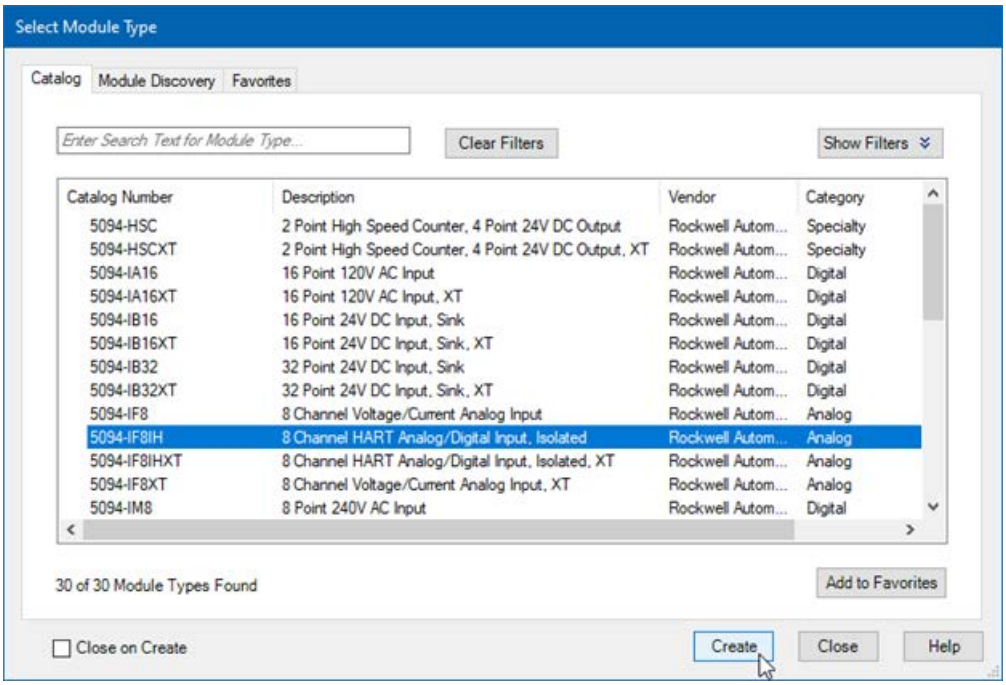


Add the 5094-IF8IH Module to the Project I/O Configuration

1. In the controller Organizer for your project, select the 5094 Backplane. Right-click and select "New Module..."



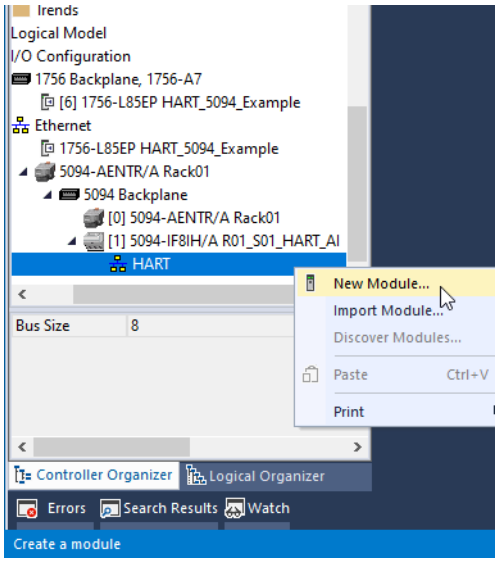
2. Select the 5094-IF8IH module and “Create”.



3. To accept the module defaults, Select OK.

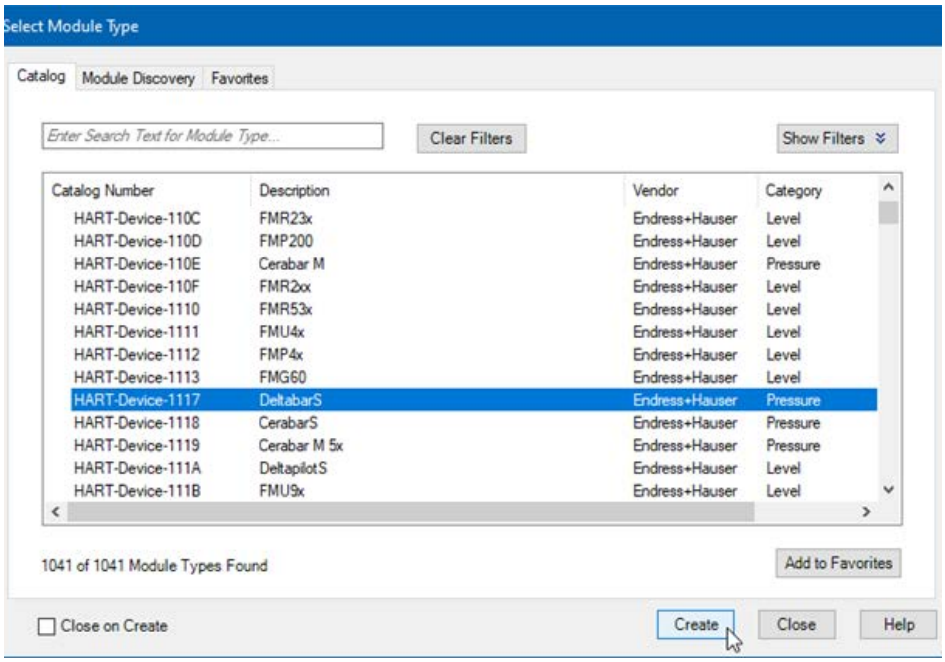
Add the HART Device to the Project I/O Configuration

1. In the controller Organizer for your project, select the HART network. Right-click and select “New Module...”.

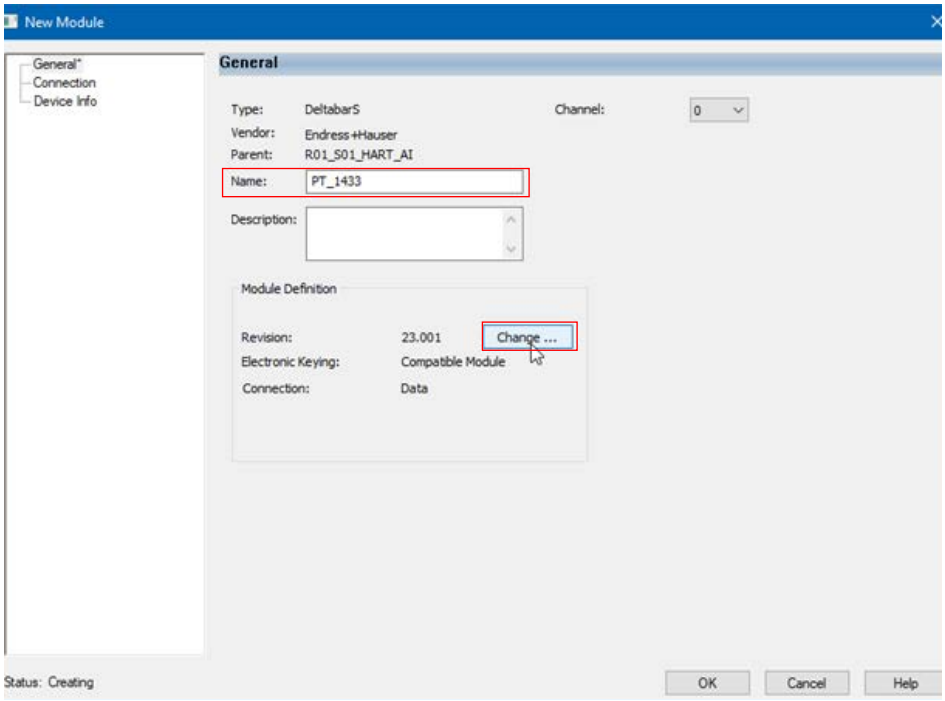


2. Select the type of HART transmitter and “Create”.

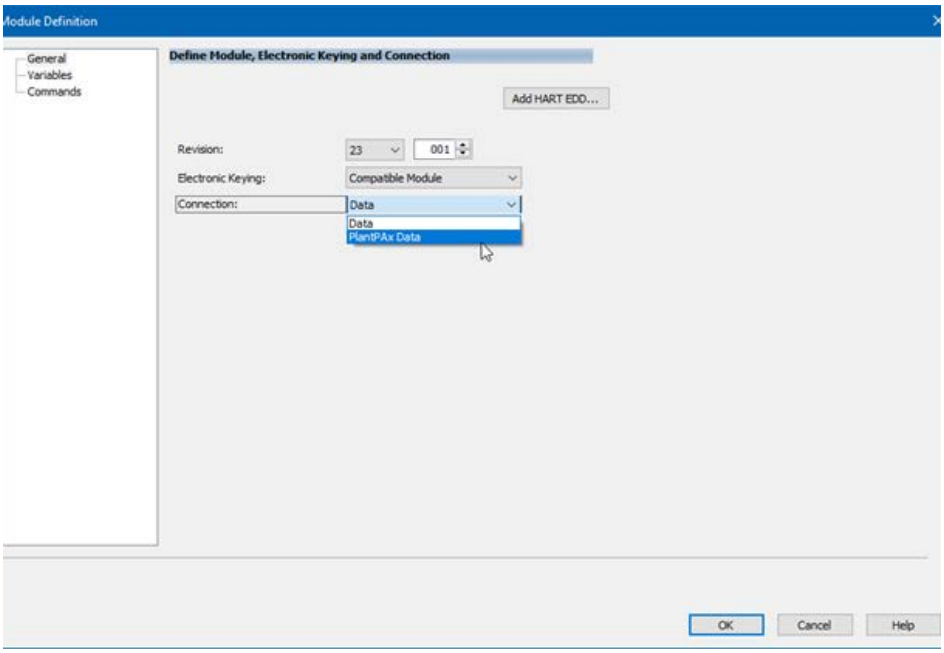
In this example, we are using an Endress+Hauser Deltabar-S device.



3. In the New Module dialog box, enter a name for the transmitter then select Change in the Module Definition section.

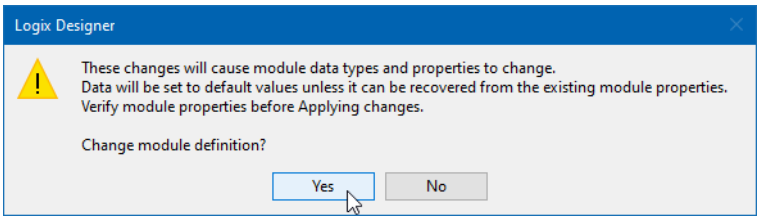


4. Change the Connection type to PlantPAx Data.

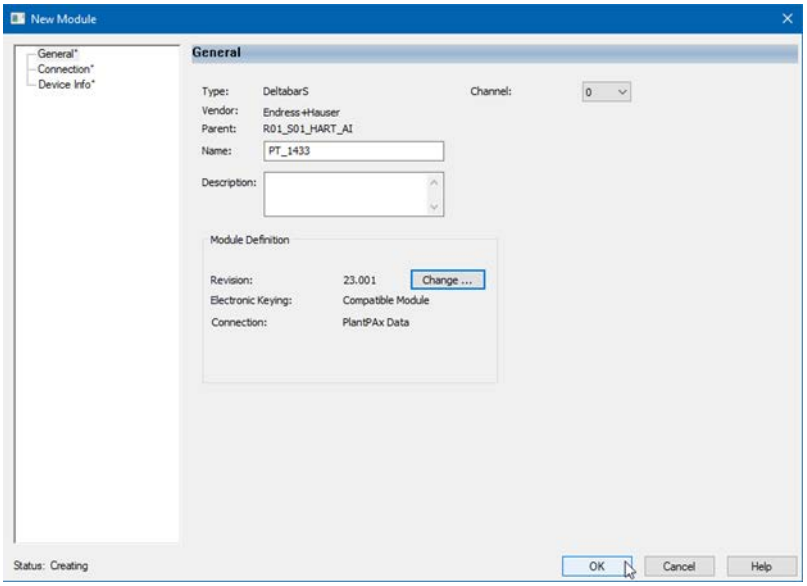


5. Changing the connection type causes a change in data types for the input and output data.

   Select Yes to change the module definition.

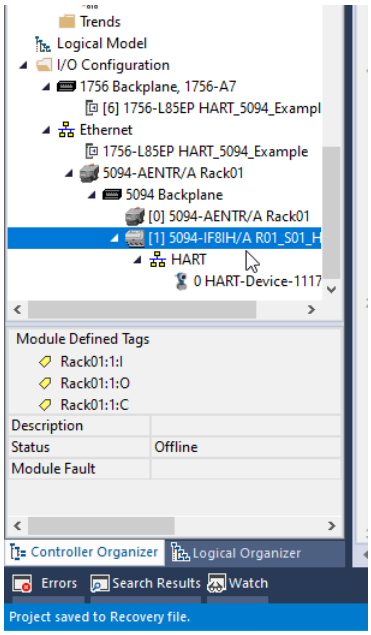


6. Verify the information and Select OK.

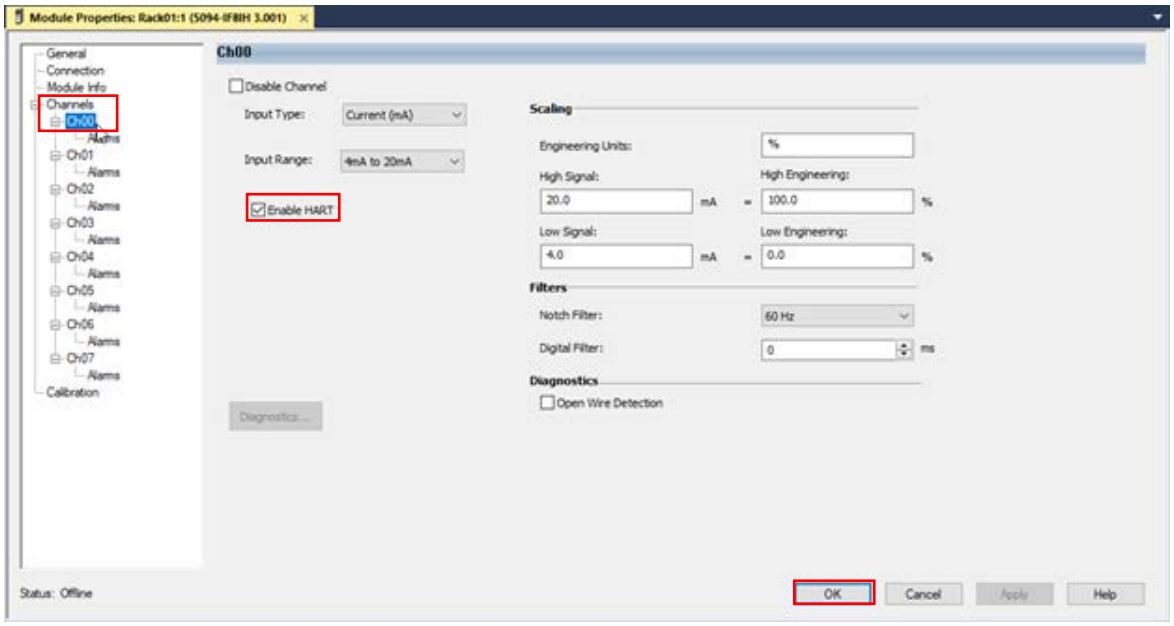


Configure the Analog Input Channel

1. In the controller Organizer for your project, select the 5094-IF8IH module created. Double-click to open the Properties dialog.



2. Select the channel where the transmitter is connected. In this example, it is Channel 00. Select the box to Enable HART communication on this channel.

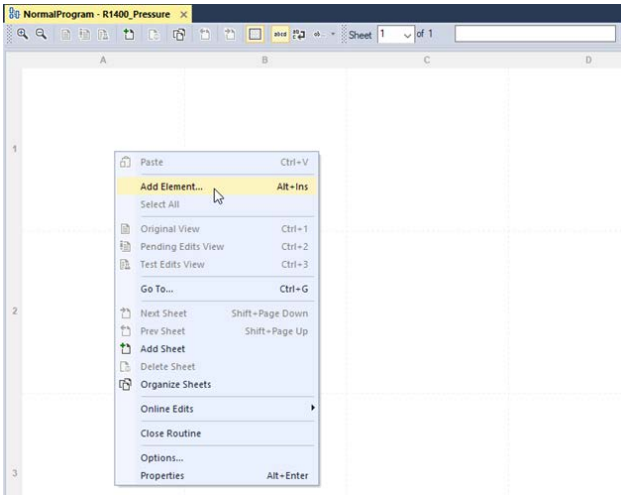


Add the PAH (Process Analog HART) and PAI (Process Analog Input) Instruction Instances to the Project

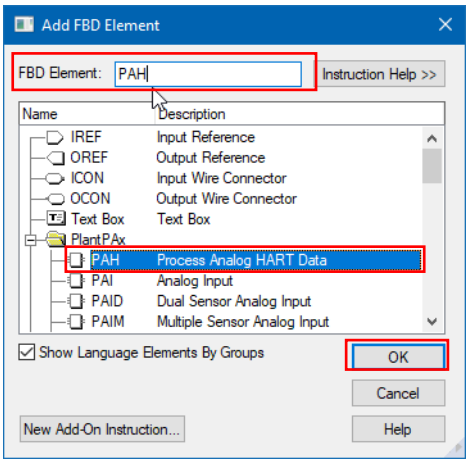
In this example, we are using a Function Block routine. Ladder Diagram or Structured Text could be used.

Add the PAH Instruction Instance

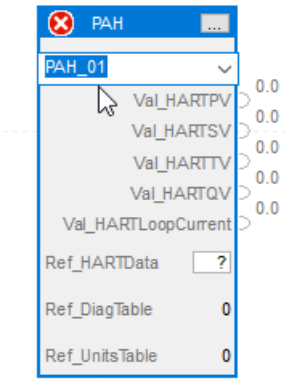
1. Right-click a blank area on the sheet and select "Add Element...".



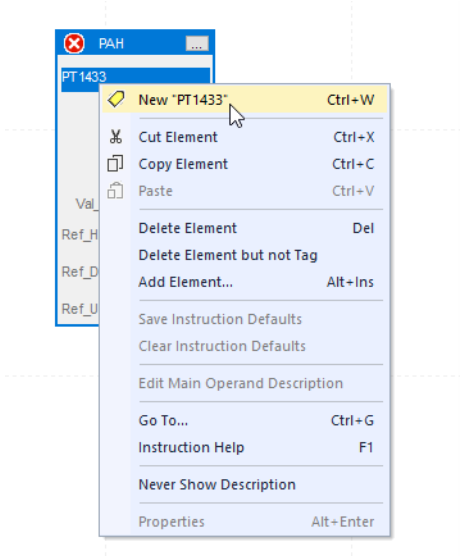
2. Enter PAH for the FBD Element.



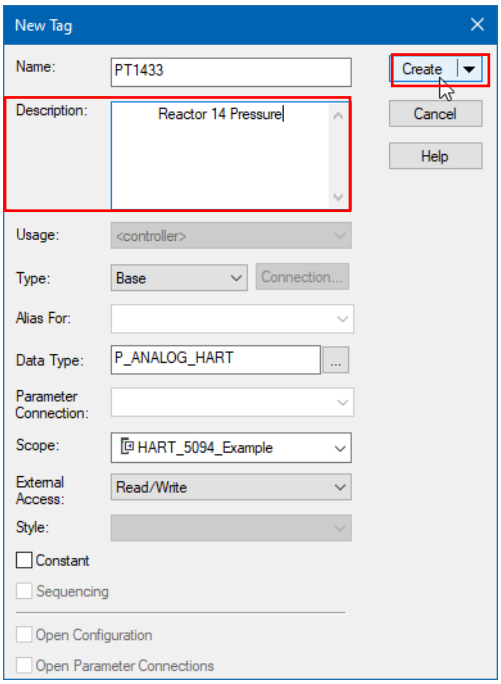
3. Enter the desired tag name of the backing tag for the PAH block.



4. Right-click the new tag name and select "New <tagname>".



5. In the New Tag dialog, enter a tag description. The required data type is automatically selected for you.

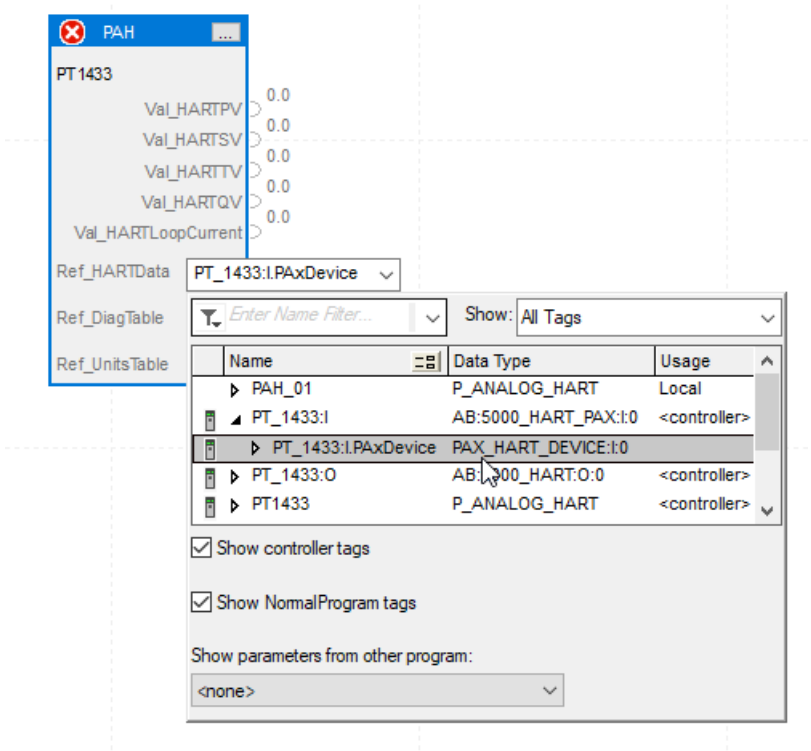


The tag can be created at Controller scope, or in the Program containing this routine. For this example, we use a Controller-scope tag.

Connect PAX_HART_DEVICE:I:0 Member from Input Assembly to Ref_HARTData InOut Parameter.

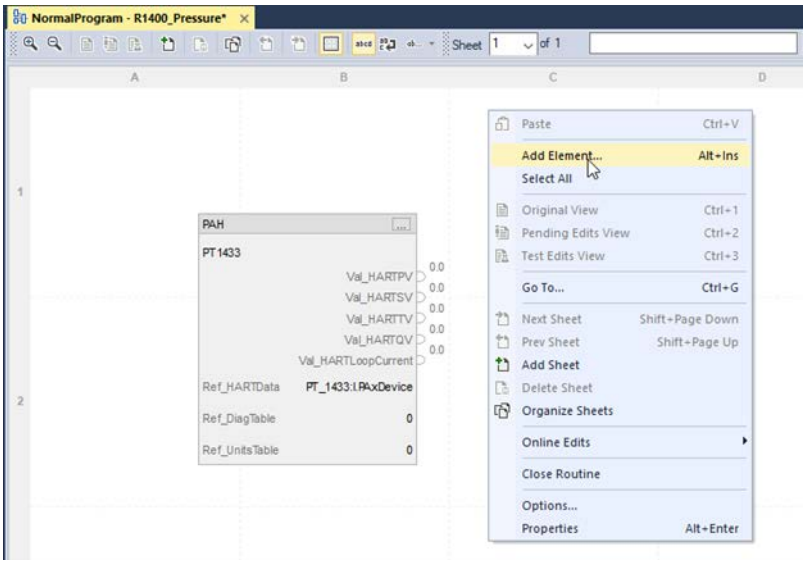
1. Select the dropdown for the Ref_HARTData InOut Parameter.
2. Navigate to the input assembly tag for the HART device, expand, and select the "PAxDevice" member.

   The data type must be "PAX_HART_DEVICE:I:O".

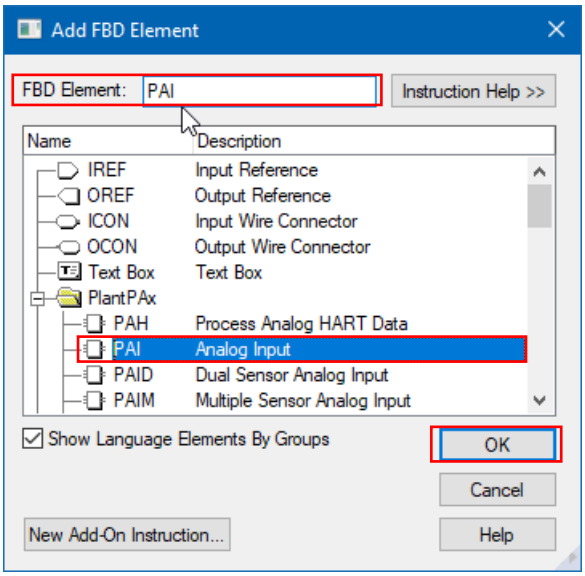


Add the PAI Instruction Instance

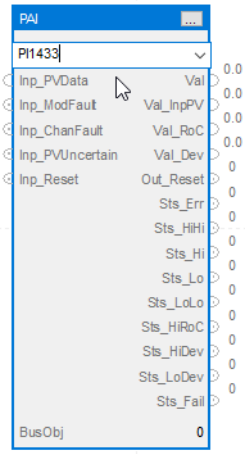
1. Right-click a blank area of the Function Block routine sheet and select "Add Element...".



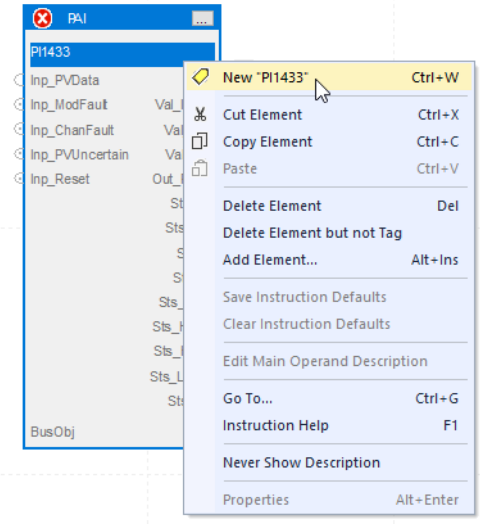
2. Enter PAI for the FBD Element.



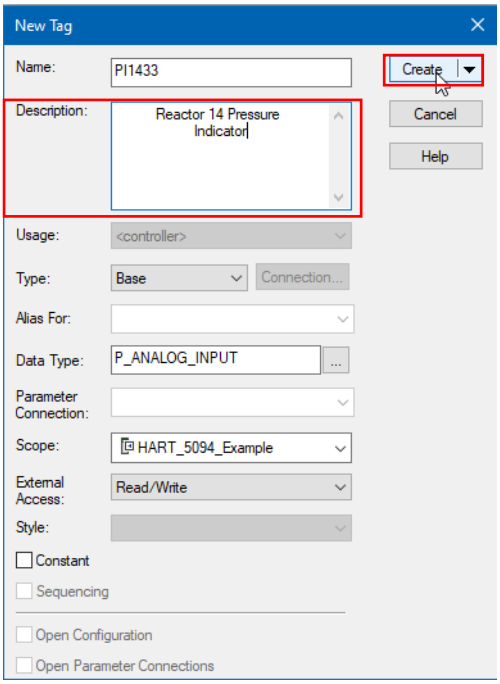
3. Enter the desired tag name of the backing tag for the PAI block. In this example, we used "PI1433" (for Pressure Indicator).



4. Right-click the new tag name and select "New <tagname>".

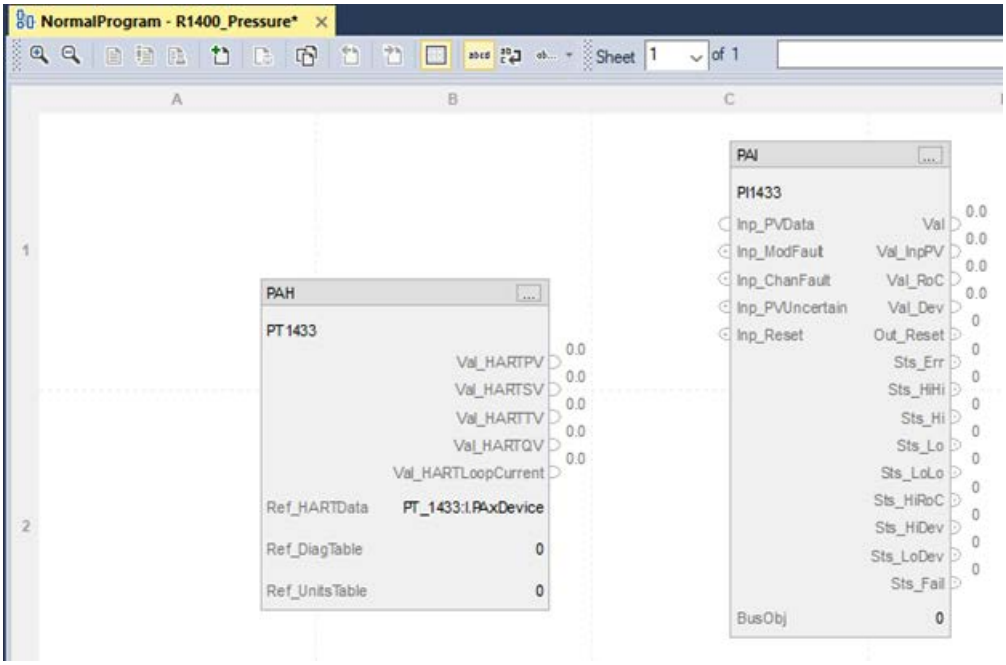


5. In the New Tag dialog, enter a tag description. The required data type is automatically selected for you.



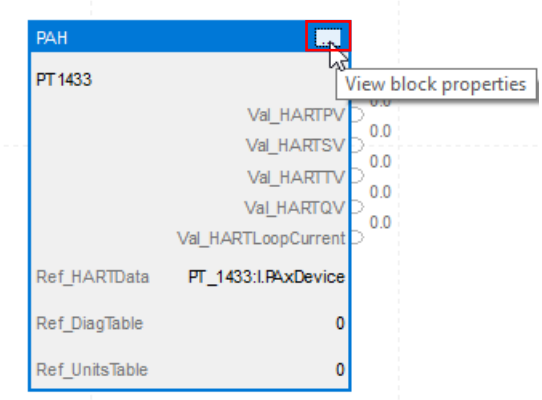
The tag can be created at Controller scope, or in the Program containing this routine. For this example, we use a Controller-scope tag. For HMI navigation to work properly, the PAH and PAI instance tags must be at the same scope.

The tag is created and the routine contains no errors.

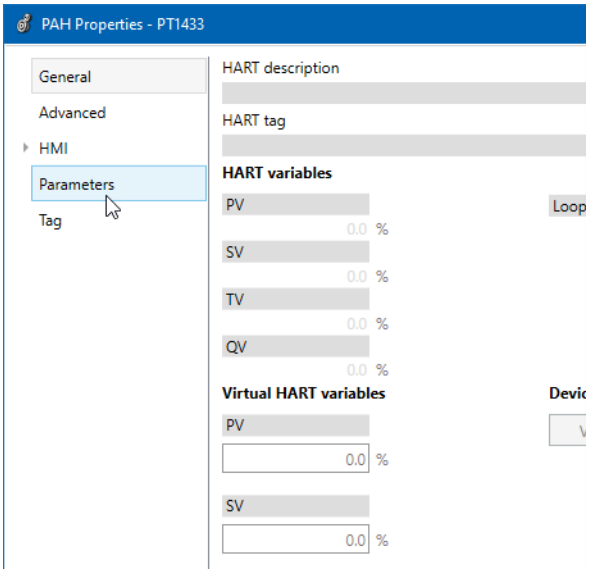


Connect the PAH Instance to the PAI Instance

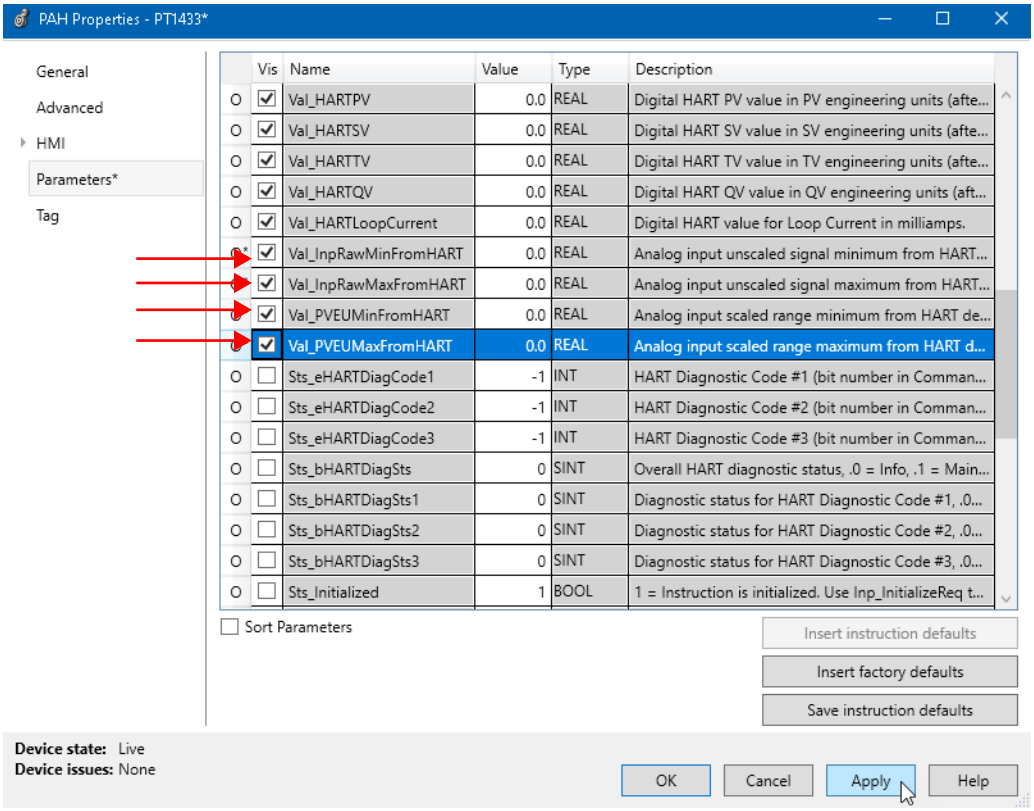
1. Select the properties of the PAH instruction.



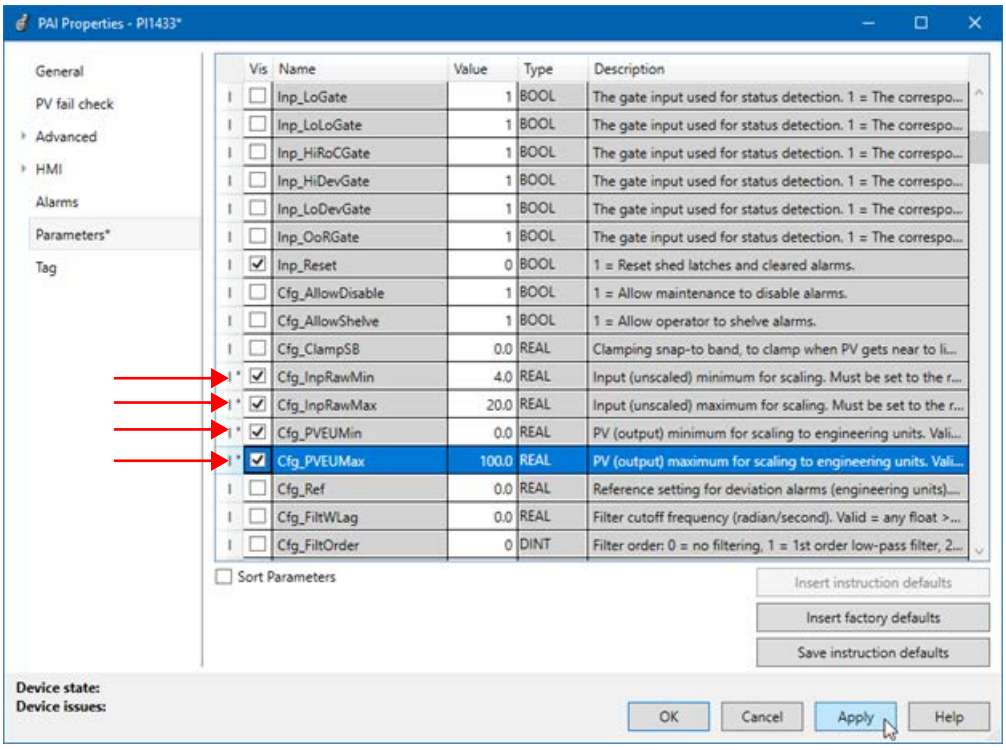
2. Select the Parameters tab.



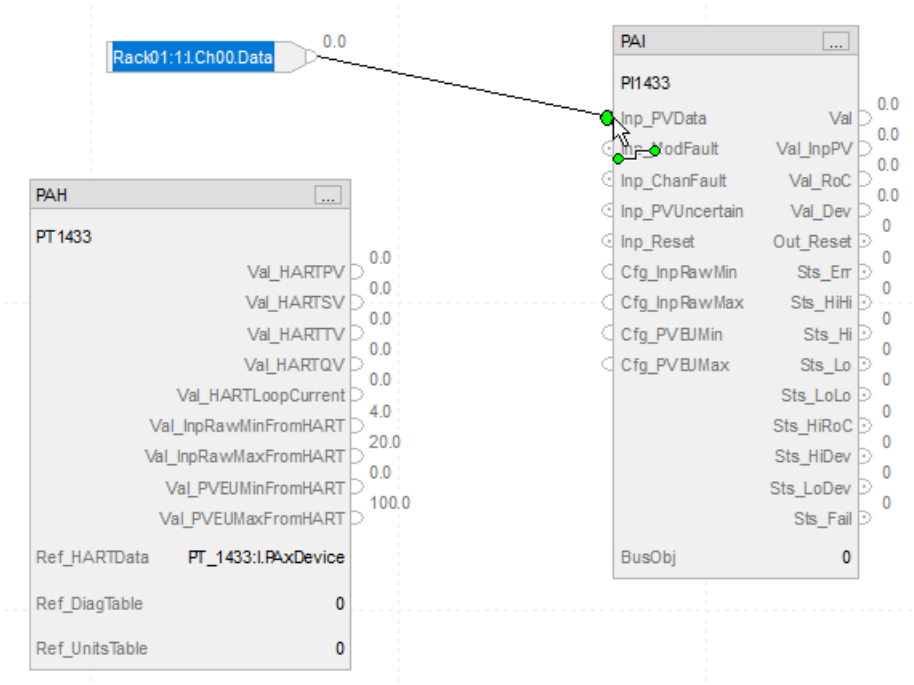
3. Select the boxes in the "Vis" column to make the Raw and EU scaling Values visible as output pins.



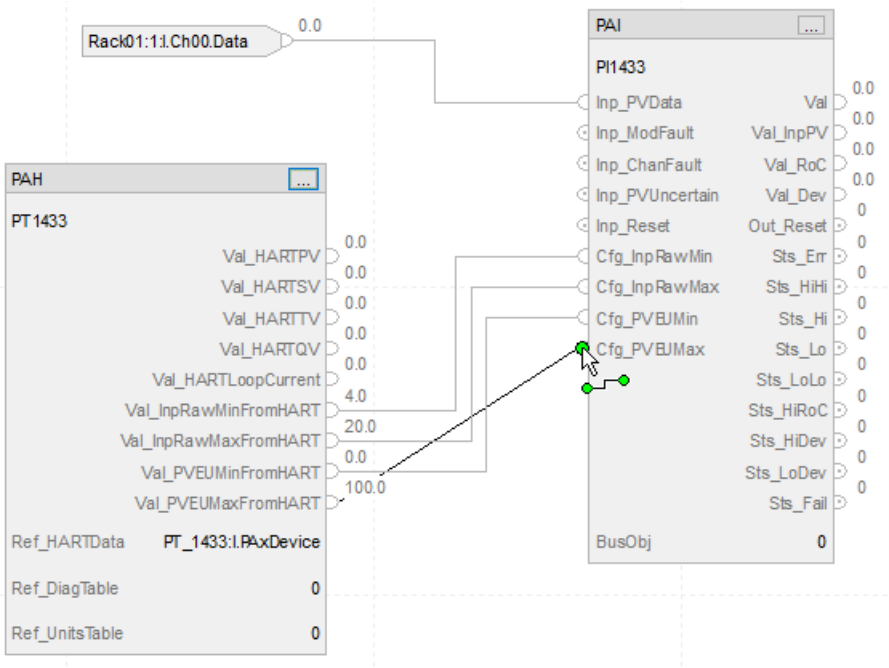
4. From the Parameters tab of the **PAI instruction** properties, Select the boxes in the "Vis" column to make the Raw and EU **configuration input** pins visible.



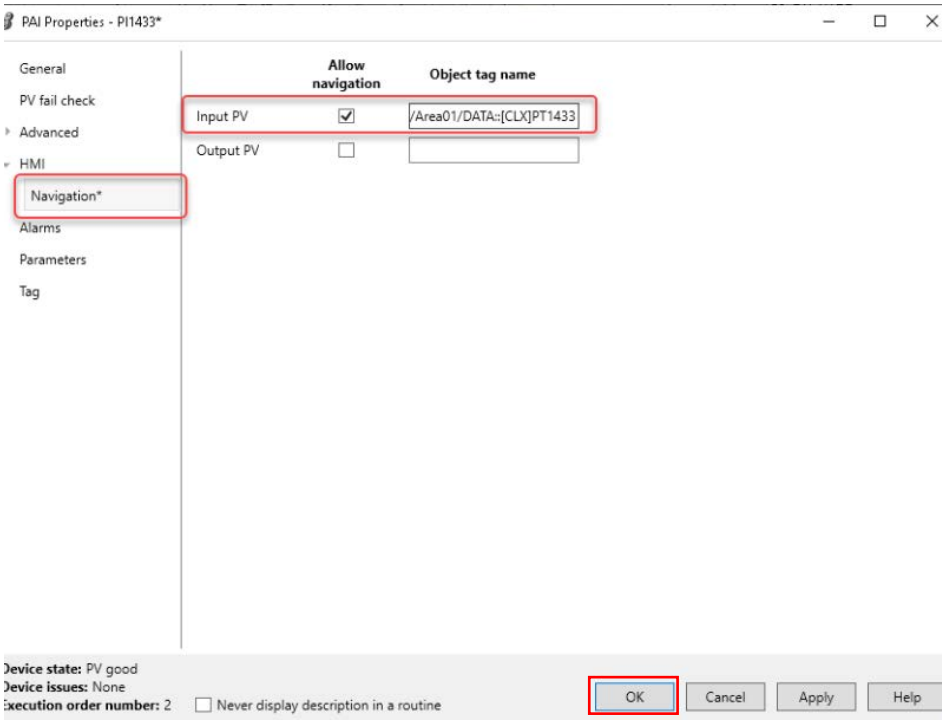
5. Wire the analog input signal to the PAI block.



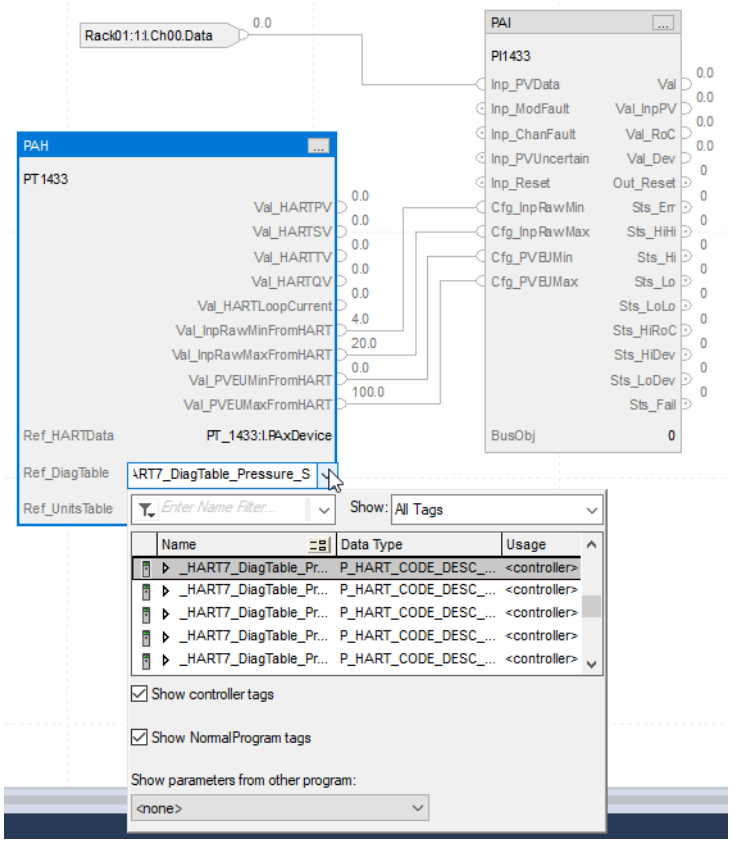
6. Wire the HART scaling data from the PAH block to the PAI block.



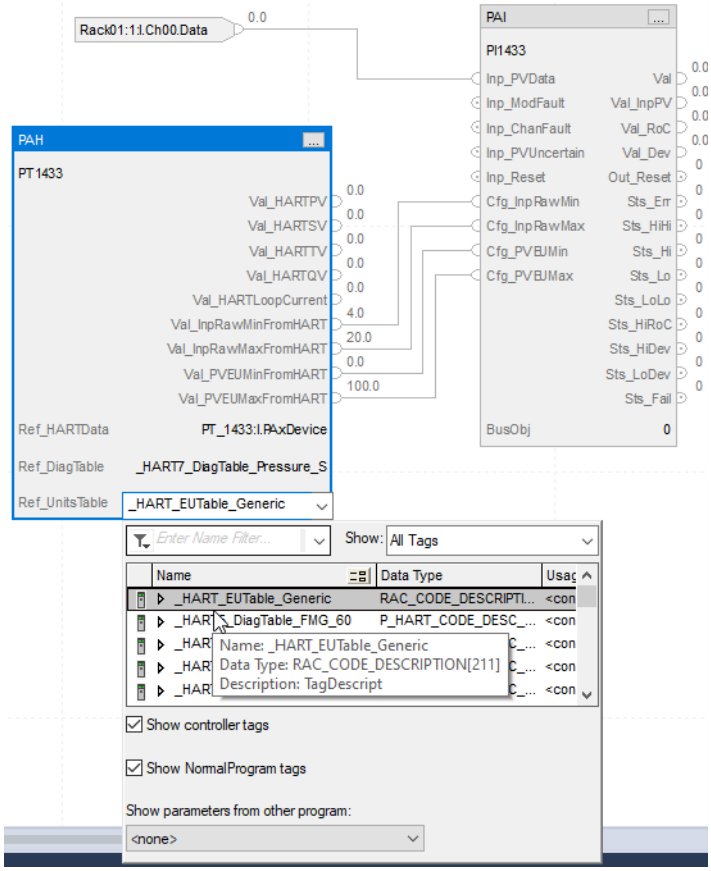
7. From the PAI Properties, navigate to the HMI>Navigation tab. Link the PAI Input PV navigation to the PAH instance.



8. On the PAH instance, link the HART Diagnostic Lookup Table for the pressure transmitter InOut parameter.



9. Link the HART engineering units lookup table to the units InOut parameter.

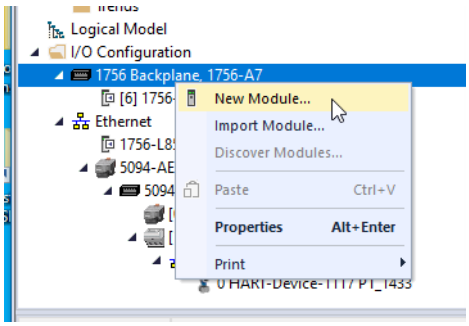


1756-IF8IH with raP_Tec_HARTChanData_to_PAH Add-On Instruction Configuration Example

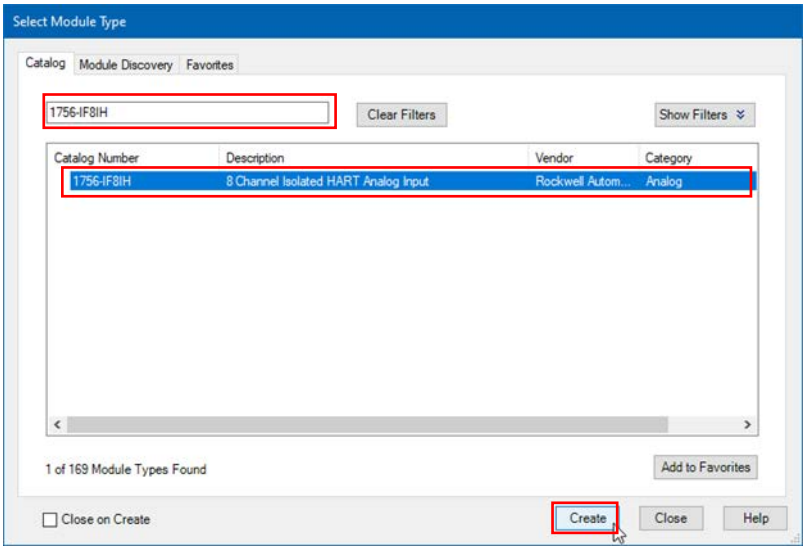
This appendix shows an example of using a 1756-IF8IH (using I_1756IF8IH 4.10) with the raP_Tec_HARTChanData_to_PAH Add-On Instruction from the 5.00 or later Library download to feed PAH and PAI instructions (5.00 or later system on L85EP).

Add the 1756-IF8IH Module to the Project I/O Configuration

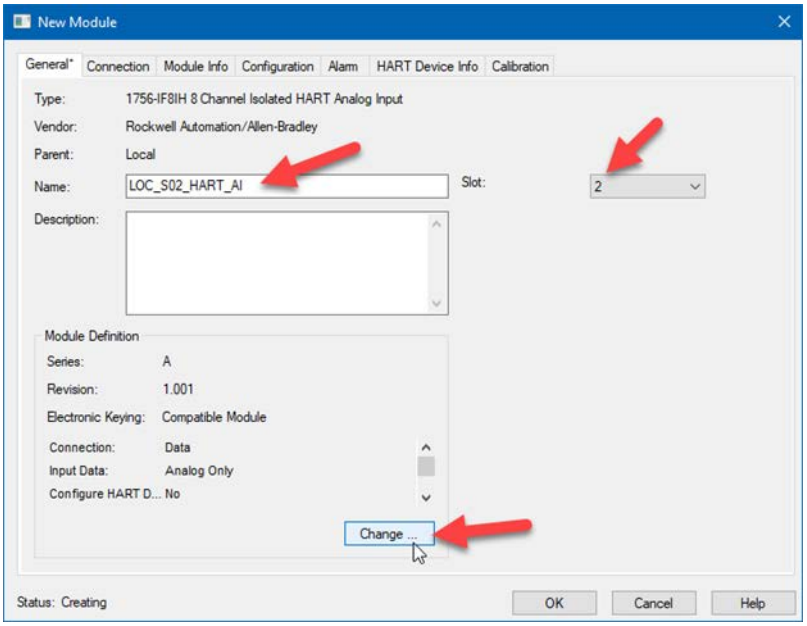
1. In the controller Organizer for your project, select the 1756 Backplane. Right-click and select "New Module...".



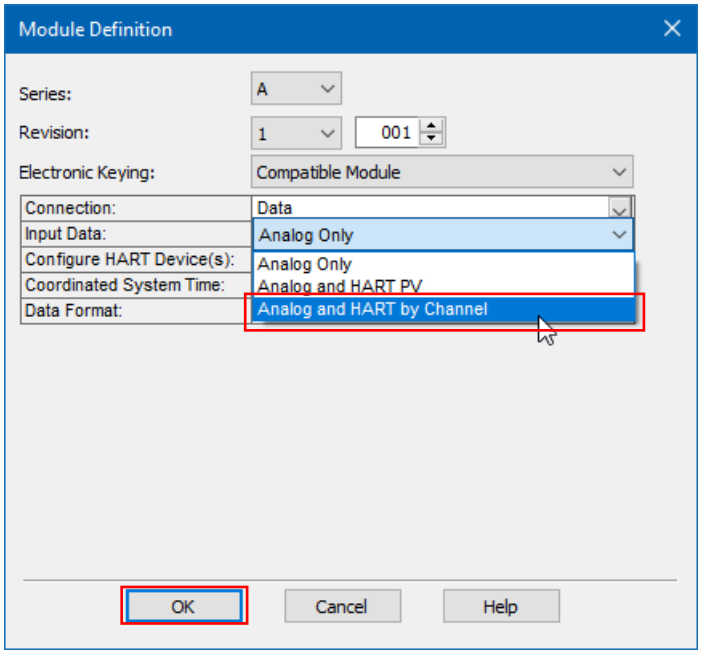
2. Select 1756-IF8IH.



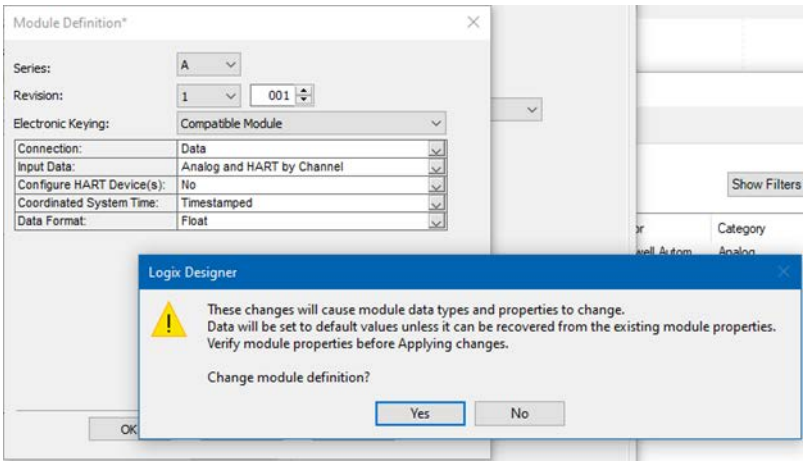
3. Enter a name for the module, Select the slot number where the module is installed, and select Change in the Module Definition.



4. In the Module Definition dialog, change the Input Data selection to "Analog and HART by Channel".

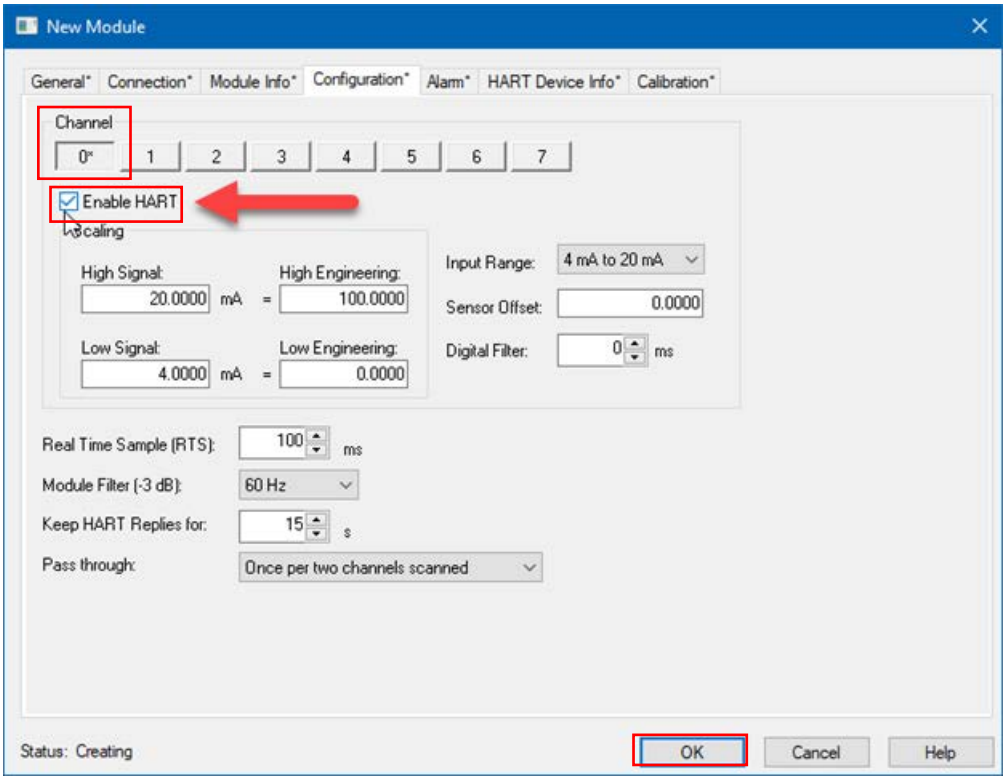


5. Select YES to change the module definition.



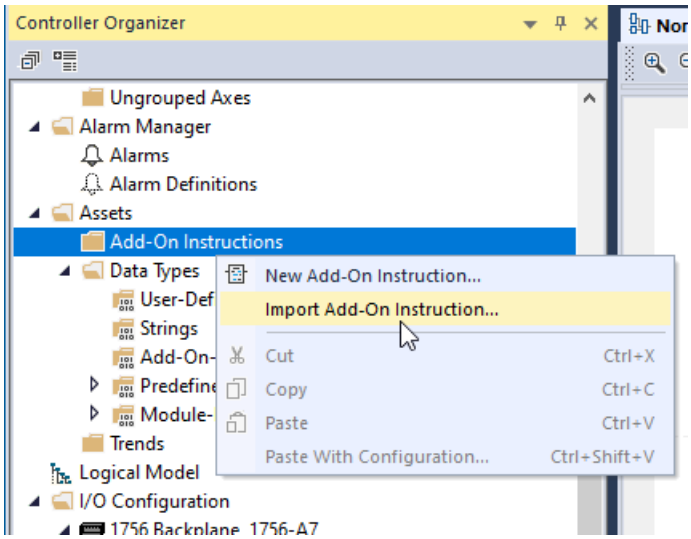
Configure the Channel for the HART Device

1. From the New Module dialog box, Select the configuration tab.
2. Select the Channel where the transmitter is installed and Enable HART on the channel.

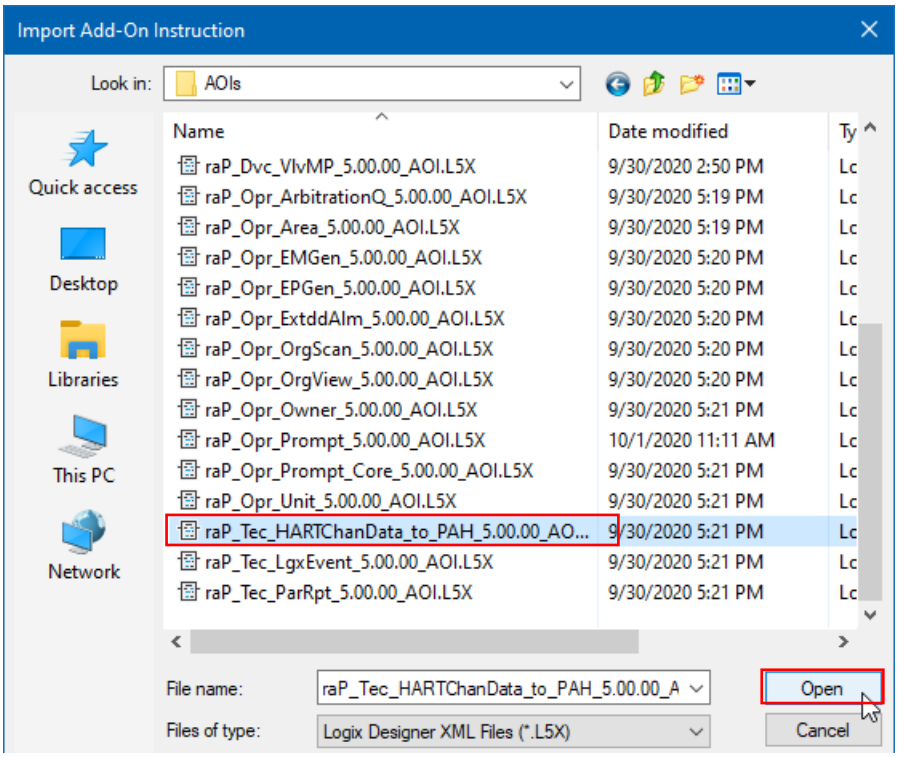


Import the raP_Tec_HARTChanData_to_PAH Add-On Instruction

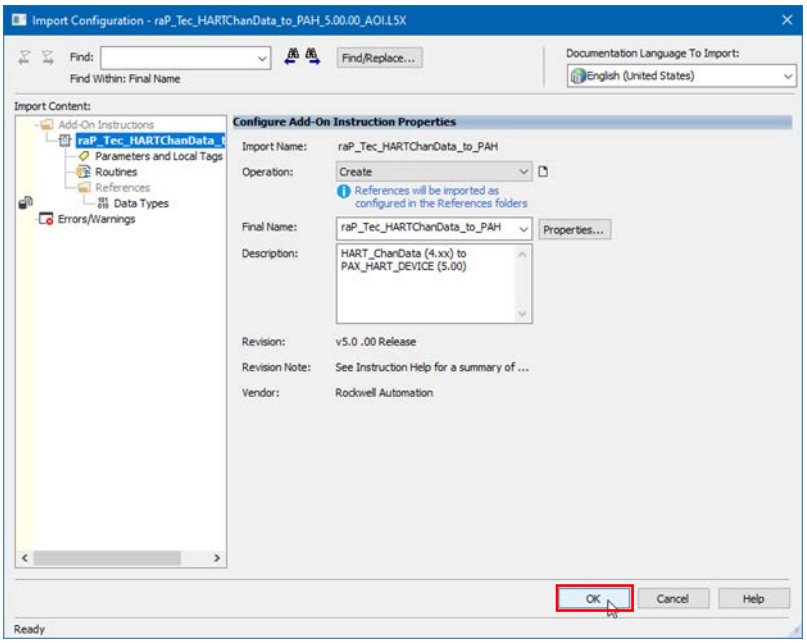
1. In the Controller Organizer, expand "Assets" to show the "Add-On Instructions" folder.
2. Right-click the Add-On Instructions folder and select "Import Add-On Instruction...".



3. Navigate to the location where you downloaded the Library of Process Objects version 5.x.
4. Navigate to the Logix Add-On Instructions. Select the "raP_Tec_HARTChanData_to_PAH" Add-On Instruction import L5X file.

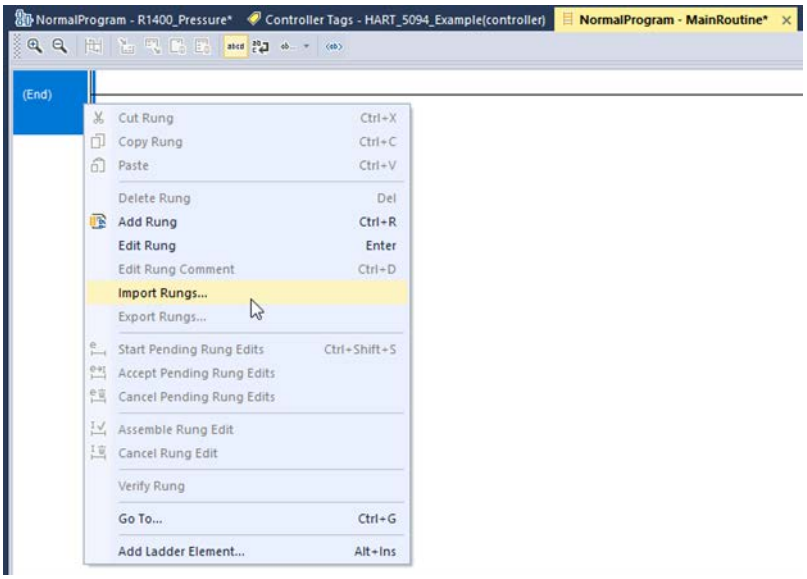


5. Select OK in the Import Configuration dialog box to accept the default values.



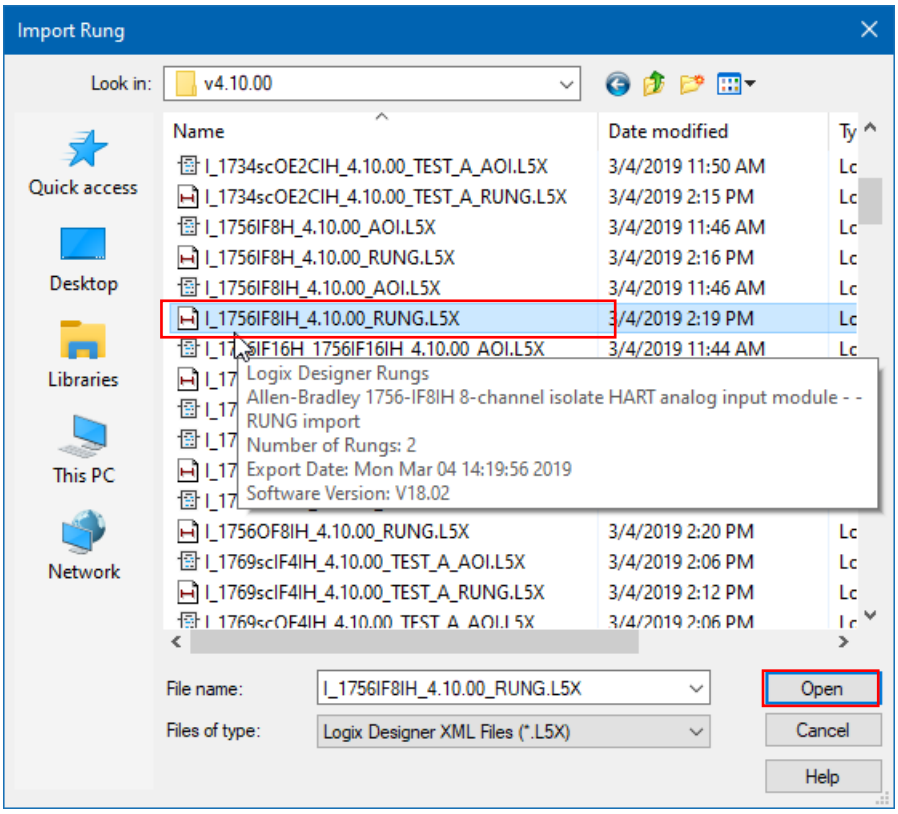
Import the I_1756IF8IH Rung into the Project

1. Open a Ladder Diagram routine in your project.
2. Click in the left margin where you want to insert the rung for the 1756-IF8IH module. Right-click and select "Import Rungs...".

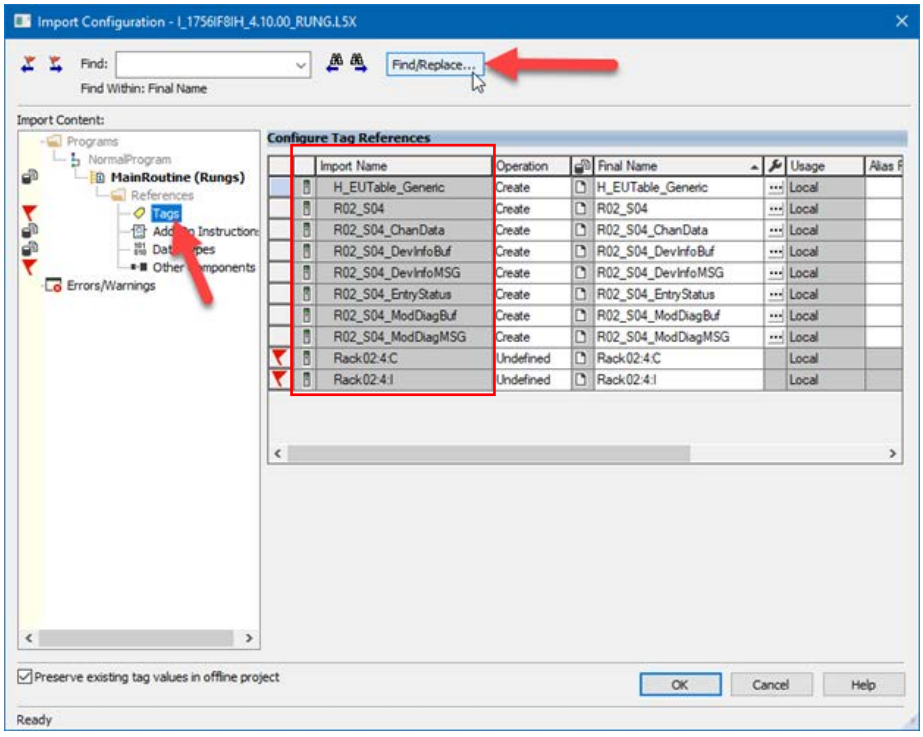


3. Navigate to the location where you downloaded the Library of Process Objects version 4.10.xx.

4. Navigate to the Logix Add-On Instructions. Select the I_1756IF8IH_4.10.00_RUNG.L5X file.

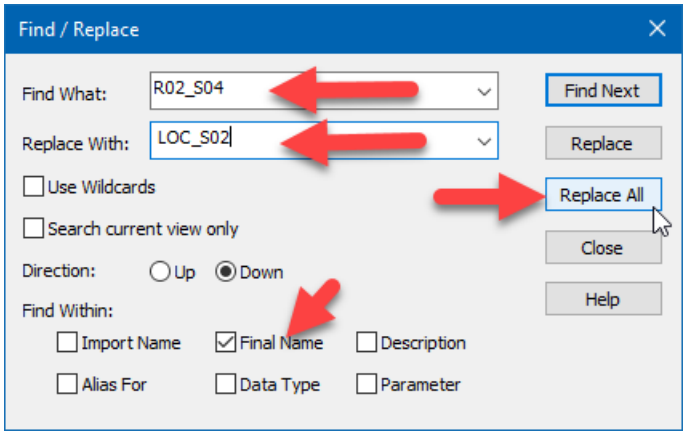


5. In the Import Configuration window, select the "Tags" item in the "Import Content" tree on the left. Note the names of tags in the import file. Select "Find/Replace...".

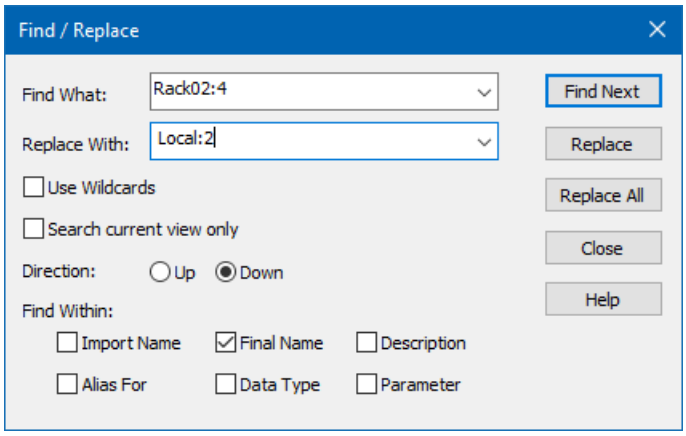


6. Change the base that you want to use for the tag names.
   - In the "Find What" box, enter "R02_S04", which is the base name for the tags in the import file.

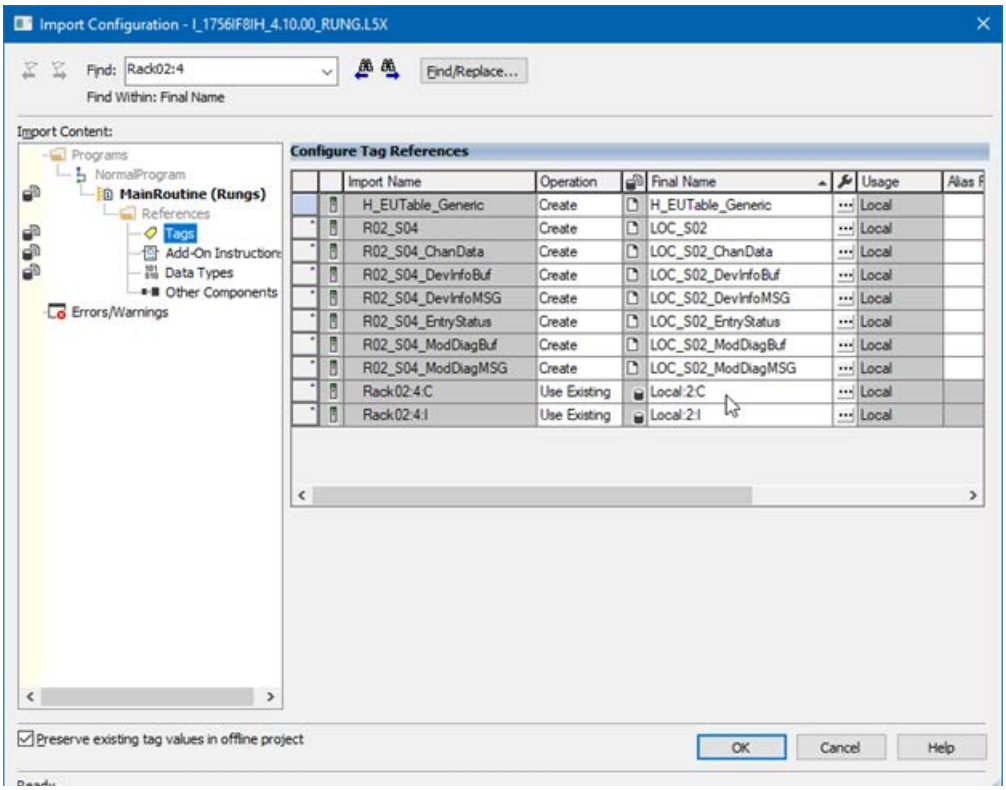
- In the "Replace With" box, enter the base that you want to use for tag names for this rung. Since we created the module in local chassis slot 2, for this example we use "LOC_S02".
- Select "Replace All"



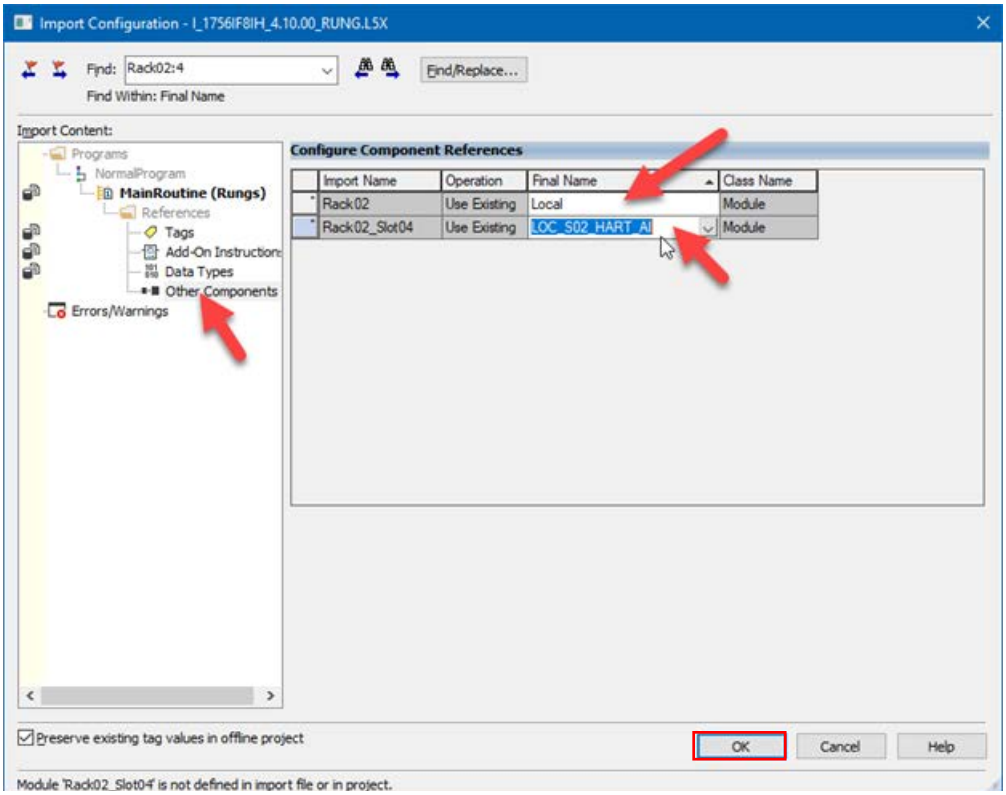
7. Use the same process to replace the text "Rack02:4" in the import with "Local:2", for the tag names assigned to the module I/O data.



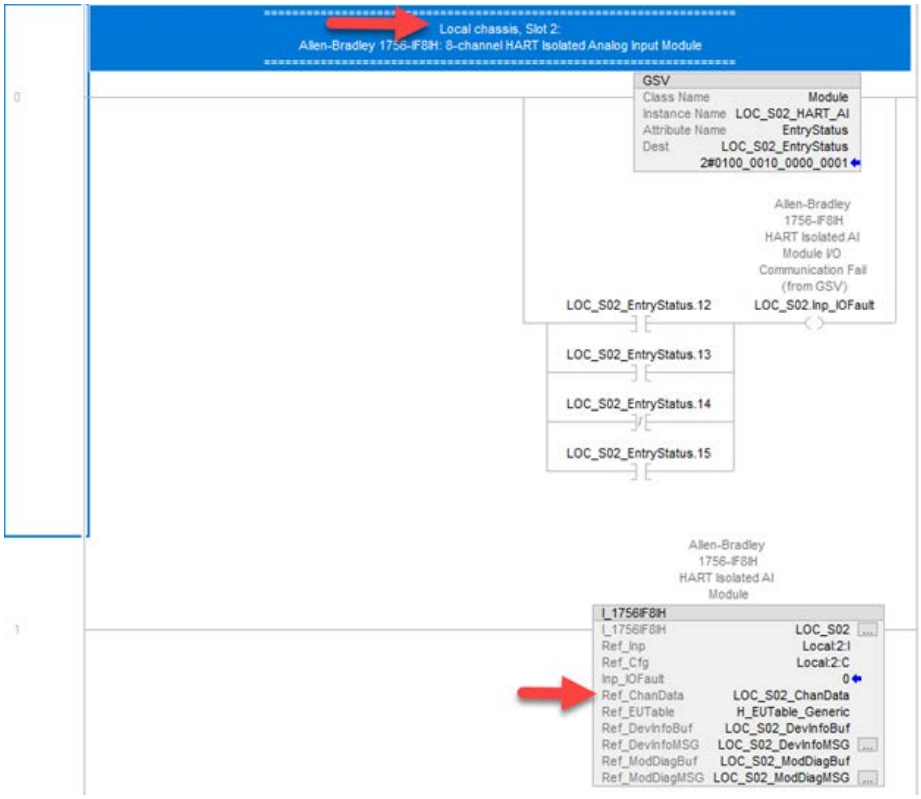
The Final Name column shows the tags to be created or used.



8. Select the Other Components item in the Import Contents tree.
9. Change the Final Name items to align with the Rack name and the Module name you gave the 1756-IF8IH module when you created it. Select OK to import the rung.



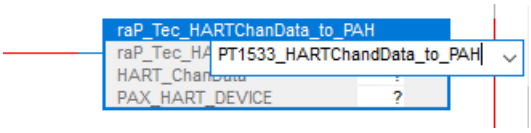
10. Two rungs are imported. On the first rung, change the Rung Comment to reflect the location of the module created. Note the tag of the Ref_ChanData InOut parameter in the second rung. This tag name is used in the following steps.



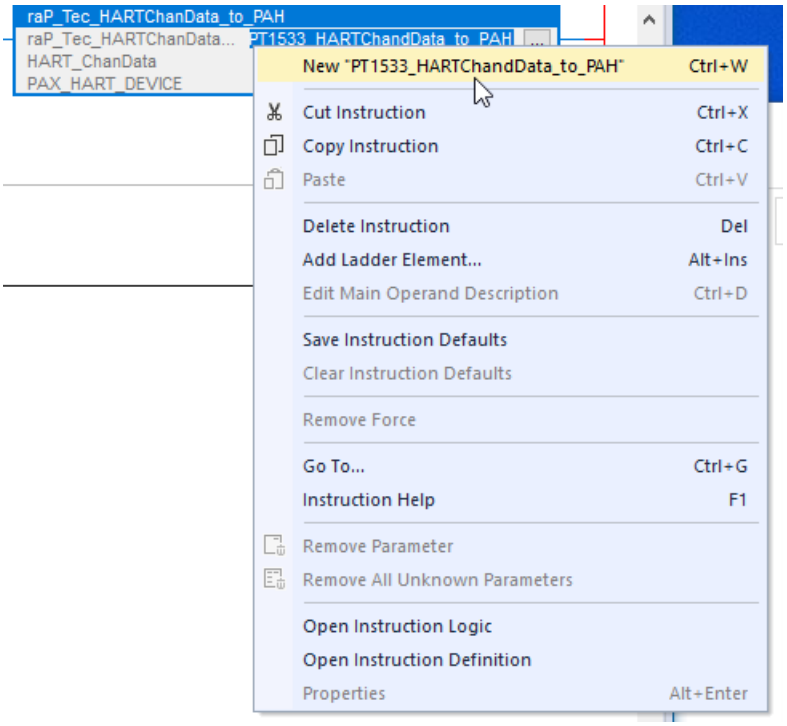
Add the raP_Tec_HARTChanData_to_PAH Instance to the Project

1. Add a rung after the I_1756IF8IH rung. On that rung, place an instance of the raP_Tec_HARTChanData_to_PAH instruction.

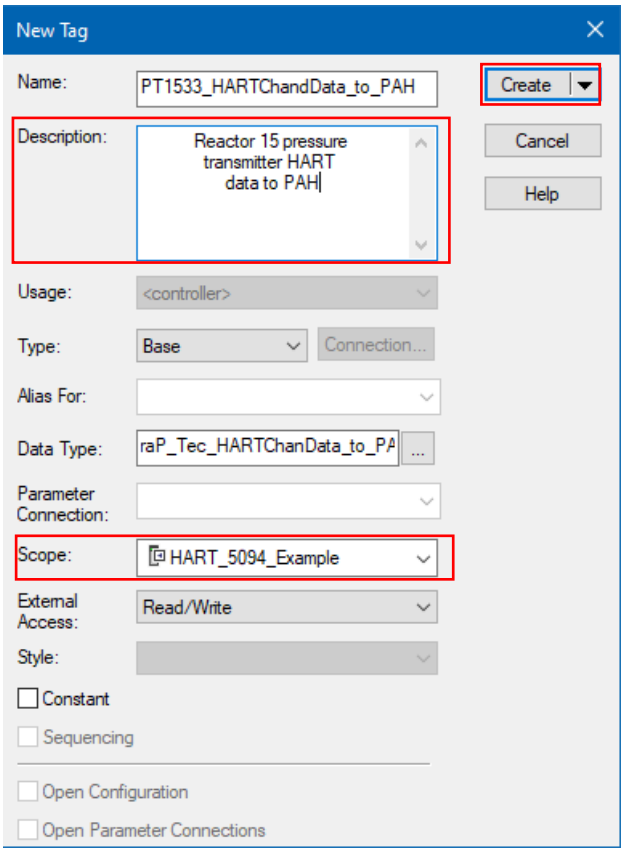
2. The first operand is the backing tag for the instruction. Enter a suitable name.



3. Right-click and select "New (tag name)".



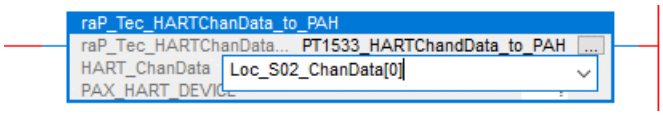
4. Enter a description and select the tag scope. The tag Data Type is set for you automatically.



5. The second operand is a HART Channel Data member from the I_1756IF8IH instruction.

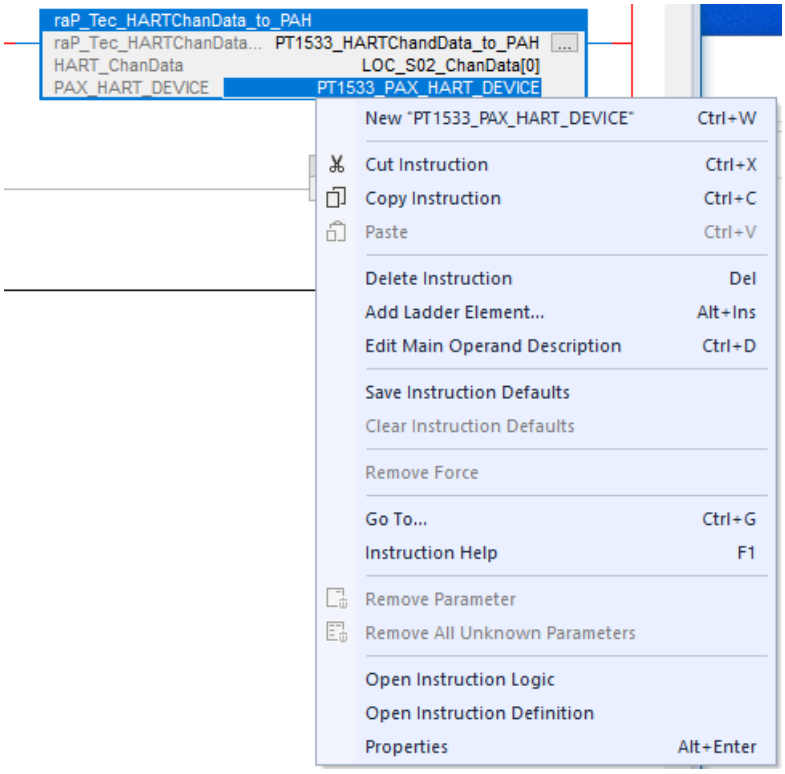
The I_1756IF8IH instruction creates an array of 8 channels' data. Previously we configured Channel 0 on the 1756-IF8IH for this device.

Select element [0] of that array for this operand.

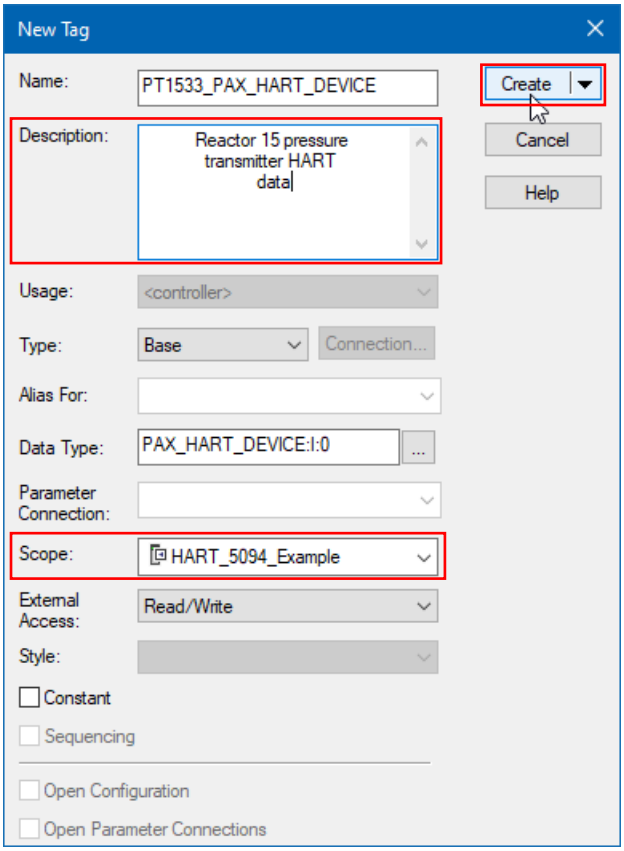


6. The third operand is a tag that you create that is the same data type as used by newer HART I/O modules, such as the 5094-IF8IH. This tag contains the HART data coming out of the raP_Tec_HARTChandData_to_PAH instruction and going to the PAH instruction.

   Enter a suitable tag name, then right-click and select "New (tag name)".



7. Enter a description and select the tag scope. The tag Data Type is set for you automatically.

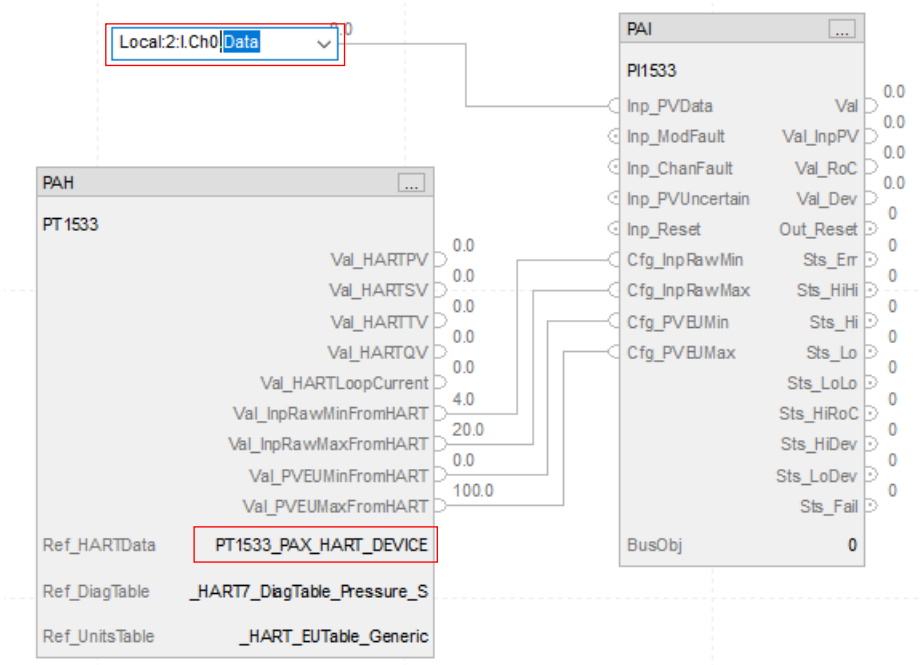


Add the PAH and PAI Instances to the Project and Connect PAH and PAI Instances

Continue as in the example documented in Appendix E, Add the PAH (Process Analog HART) and PAI (Process Analog Input) Instruction Instances to the Project, creating the PAH and PAI instances and linking them together.

The "Ref_HARTData" operand on the PAH instruction is the tag that you just created above, PT1533_PAX_HART_DEVICE. The analog input to the PAI instruction comes from the input data value from Channel 0 of the 1756-IF8IH, which is in the Local chassis, slot 2. In this example, the tag is Local:2:I.Ch0.Data.

The following diagram shows the final configuration for this example.



Notes:

Rockwell Automation Support

Use these resources to access support information.

Technical Support Center	Find help with how-to videos, FAQs, chat, user forums, Knowledgebase, and product notification updates.	rok.auto/support
Local Technical Support Phone Numbers	Locate the telephone number for your country.	rok.auto/phonesupport
Technical Documentation Center	Quickly access and download technical specifications, installation instructions, and user manuals.	rok.auto/techdocs
Literature Library	Find installation instructions, manuals, brochures, and technical data publications.	rok.auto/literature
Product Compatibility and Download Center (PCDC)	Download firmware, associated files (such as AOP, EDS, and DTM), and access product release notes.	rok.auto/pcdc

Documentation Feedback

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

Waste Electrical and Electronic Equipment (WEEE)

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

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

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

Connect with us.

rockwellautomation.com — expanding **human possibility**®

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2663 0600, Fax: (32) 2 663 0640
ASIA PACIFIC: Rockwell Automation SEA Pte Ltd, 2 Corporation Road, #04-05, Main Lobby, Corporation Place, Singapore 618494, Tel: (65) 6510 6608, FAX: (65) 6510 6699
UNITED KINGDOM: Rockwell Automation Ltd., Pitfield, Kiln Farm, Milton Keynes, MK11 3DR, United Kingdom, Tel: (44)(1908) 838-800, Fax: (44)(1908) 261-917

Publication PROCES-RM200J-EN-P - December 2025
Supersedes Publication PROCES-RM200I-EN-P - October 2025