

Logic Developer – PLC PAC Machine Edition Getting Started

Version 9.80

ver. 9.8



PAC Machine Edition

- Install Machine Edition
- Install Change Management
- Important Product Information
- Browse the DVD
- Exit

This program is protected by U.S. and international copyright laws as indicated in "About" and/or other content or materials provided with the program.



Contents

Section 1: Welcome	1
1.1 Processor Speed and Memory	2
1.2 Extra Requirements When Executing Multiple Instances of Engineering Workstation	2
1.3 VMWare Support.....	2
1.4 Installation	3
1.4.1 To Install Logic Developer – PLC	3
1.4.2 Microsoft Visual C++	3
1.5 Product Authorization	4
Automatic.....	4
1.5.1 License Activation	4
Section 2: PAC Machine Edition	6
2.1 Quick Start	6
2.1.1 To Start Machine Edition	6
2.2 Projects	7
2.2.1 To Create a New Project Using a Template	7
2.2.2 To Open an Existing Project for Editing to Open an Existing Project for Editing	7
2.2.3 To Import a Folder.....	8
2.3 Interface Overview	8
2.3.1 Ribbon Bar	8
2.3.2 Utility Panes	9
2.3.3 Using Docking Markers.....	9
2.4 Variables	10
2.4.1 User-Defined Data Types	10
2.4.2 Variable Properties in the Inspector	10
2.4.3 To Map a Variable to Controller Memory or Alias a Variable	11
2.4.3.1 First Method.....	11
2.4.3.2 Second Method.....	11
2.5 Options	12
2.5.1 To Set Options and Preferences	12
2.5.2 To Reset an Entire Page of Options to their Default Settings	12
2.6 Machine Edition Help.....	12
2.6.1 Companion Help	12

- 2.6.1.1 To Use Companion Help 13
- 2.6.2 InfoViewer Help 13
- 2.6.2.1 To Get Help with Machine Edition..... 13
- 2.6.2.2 To Use the Full-Text Search 13
- 2.6.2.3 To Bookmark Favorite Topics 14
- 2.6.2.4 To Look Up Topics in the Help Table of Contents 14

Section 3: PACSystems Targets 15

- 3.1 Adding, Configuring, and Converting Targets 15
- 3.1.1 Adding Targets 15
- 3.1.2 Configuring Controller Targets 16
- 3.2 Converting Targets..... 17
- 3.2.1 To Convert a Target..... 17
- 3.3 Configuring Communication 18
- 3.3.1 To Configure an Ethernet, Mode, or Serial Connection with any PACSystems Controller..... 18
- 3.3.2 To Set a Permanent IP Address for a PACSystems..... 18
- 3.3.3 To Download an IP Address via a Controller Serial Port..... 19
- 3.4 Interacting with a Controller 19
- 3.4.1 Validating a Target..... 19
- 3.4.2 To Validate a Target 20
- 3.4.3 Offline, Online: Monitor Mode, Programmer Mode..... 20
- 3.4.4 To Go Online to an Emerson Controller..... 21
- 3.4.5 To Change the Online Mode 21
- 3.4.6 To Go Offline from an Emerson Controller 21
- 3.4.6.1 Upload/Download..... 21
- 3.4.7 To Download to a PACSystems Controller..... 21
- 3.4.8 To Upload Files from an PACSystems Controller..... 22
- 3.4.8.1 Run/Stop..... 22
- 3.4.9 To Start a PACSystems Controller 22
- 3.4.10 To Stop a PACSystems Controller 22
- 3.4.11 Fault Tables 23
- 3.4.11.1 To View the Fault Table Reports 23
- 3.4.11.2 Reference View Tables..... 23
- 3.4.12 To Create a User-Defined Reference View Table 23

3.4.13 To Work with a User-Defined Reference View Table.....	24
3.4.14 Reports	24
3.4.15 To Generate Reports	24
3.4.16 To Redisplay a Previously Generated Report	24
3.4.17 To Print a Report Displayed in the InfoViewer	25
3.4.18 To Print LD Blocks.....	25
3.4.19 To Print ST Blocks	25

Section 4: Hardware Configuration 26

4.1 PACSystems RXi	26
4.2 PACSystems RX3i	26
4.2.1 Configuring PACSystems.....	27
4.2.2 Configuring Controller Hardware	27
4.2.3 To Replace a CPU (Not Applicable for PACSystems RXi)	28
4.2.4 To Configure a CPU	28
4.2.5 To Add an Expansion Rack (PACSystems RX3i Only).....	28
4.2.6 To Replace a Rack (Not Applicable for PACSystems RXi).....	28
4.2.7 To Add an Ethernet module (PACSystems RX3i Only)	29
4.2.8 To Configure the Ethernet Daughterboard (RX3i Only)	29
4.2.9 To Move a Module (Not Applicable for PACSystems RXi).....	29
4.3 I/O Variables.....	30
4.3.1 To Enable I/O Variables for a Module	30
4.3.2 To Map a Variable to a Terminal in the Terminals Tab of a Module or Genius Device	31
4.3.3 Hot Redundancy Systems.....	31
4.3.3.1 Genius Redundancy	32
4.3.3.2 Hot CPU Redundancy Over Genius	32
4.3.3.3 Configuring Hot Redundancy Systems	32
4.3.3.3.1 To Setup the Primary Hardware Configuration for Hot CPU Redundancy	33
4.3.3.3.2 To Configure the Secondary Hardware Configuration (RX3i and Series 90-70)	34
4.3.3.3.3 DSM324I and Motion Mate DSM31 4 Motion Modules.....	35
4.3.3.3.4 To Add a DSM324i or Motion Mate DSM314 module.....	35
4.3.3.3.5 To Configure a DSM32i or a Motion Mate DSM314	36
4.3.3.4 Remote I/O	36

4.3.3.4.1	VersaMax Remote I/O.....	36
4.3.3.5	To Create a Project Containing a Remote I/O Target from a Template.....	37
4.3.3.6	To Add a Remote I/O target to an Existing Project.....	37
4.3.3.7	To Replace the Power Supply in your Remote I/O Configuration	38
4.3.3.7.1	To Add a New Carrier/Base to Your VersaMax Remote I/O	39
4.3.3.7.2	To Add a Module to a Carrier/Base.....	39

Section 5: Logic Programs and Blocks..... 41

5.1	Program Types	41
5.1.1	Main Program	42
5.1.2	C Programs	42
5.1.3	Motion Program.....	43
5.1.4	Number of Blocks in the Main Program.....	43
5.1.5	Scheduling Programs	43
5.1.6	To Create a User-Defined Folder	44
5.1.7	To Schedule the Execution of a Block of Logic	44
5.1.8	To Control Access to a Block.....	45
5.1.9	To Search/Replace in One Block.....	45
5.1.10	Indirect References	46
5.1.11	To Assign an Indirect Reference	46
5.2	LD Editor	46
5.2.1	To Customize the LD Editor	47
5.2.2	To Create an LD Block.....	47
5.2.3	To Open an LD Block for Editing.....	47
5.2.4	Working with the LD Editor Offline	47
5.2.4.1	To Insert an Instruction	48
5.2.4.2	To Assign Instance Data to a Built-in Function Block Instance and Assign a Length to an Instruction	48
5.2.5	To Assign Variables to Instruction Operands.....	49
5.2.6	To Check (Validate) a Single LD Block	50
5.2.7	Editing Logic as Text.....	50
5.2.8	To Copy an Entire LD Block as Text.....	51
5.2.9	To Copy a Section of LD Logic as Text	51
5.2.10	To Copy Text into the LD Editor	51
5.2.11	To Move or Duplicate LD Logic	52

5.2.12	Working with the LD Editor Online.....	52
5.2.13	To Begin Editing in Test Edit	53
5.2.14	To Begin Testing the Modified Logic.....	53
5.2.15	To Cancel the Test and Continue Editing the Logic While in Test Edit Mode	53
5.2.16	To Cancel Test Edit Mode and Restore the Original Logic in the PACSystems	53
5.2.17	To Accept the Changes you Tested and Commit Them to the PACSystems Controller.....	54
5.2.18	Go Not Equal, Keep Working, and Download Changes	54
5.2.19	Affecting BOOL Variables	54
5.2.19.1	To Turn On/Off or Force a Variable	54
5.2.19.2	LD Instructions	55
5.3	FBD Editor	60
5.3.1	To Customize the FBD Editor	61
5.3.2	To Create an FBD Block.....	61
5.3.3	To Open an FBD Block for Editing.....	61
5.3.4	To Insert an Instruction.....	62
5.3.5	To Assign a Parameter Beside an Instruction	62
5.3.6	To Assign a Parameter Above an FBD Instruction or Function Block Instance	64
5.3.7	To Check (Validate) a Single FBD Block	65
5.3.8	To Change the Number of Inputs for FBD instructions (ADD, AND, MUL, OR, SUB, XOR)	65
5.3.9	To Negate an FBD Parameter.....	67
5.3.10	To Negate an FBD Wire.....	67
5.3.11	To Move or Duplicate FBD Logic	67
5.3.12	To Zoom in or Zoom Out an FBD Block	68
5.3.13	Working with the FBD Editor Online	68
5.3.14	To Turn On/Off or Force a Variable	69
5.3.15	FBD Instructions, Functions, and Function Blocks	69
5.4	IL Editor	72
5.4.1	To Configure Accumulators.....	72
5.4.2	To Create an IL Block	72
5.4.3	To Open an IL Block Editing	72
5.4.4	Working with the IL Editor Offline	73
5.4.5	To Insert an Instruction.....	73
5.4.6	To Assign Operands to an Instruction	73

5.4.7	To Create a Variable from a Reference Address	73
5.4.8	To Create a Variable from a Name	74
5.4.9	To Move or Duplicate IL Logic	74
5.4.10	To Insert an Inline Comment	74
5.4.11	To Insert a Block Comment.....	74
5.4.12	To Reformat IL Logic.....	74
5.4.13	Working with the IL Editor Online	74
5.4.13.1	To Monitor a Data Value.....	75
5.4.13.2	To Change a BOOL Variable’s State	75
5.4.13.3	To Remove the Force from a BOOL Variable.....	75
5.4.13.4	To Make Changes to IL logic and Download them to a Running Target Controller (if the Target Controller Supports it).....	75
5.4.14	IL Instructions.....	76
5.5	ST Editor.....	79
5.5.1	To Customize the ST Editor.....	80
5.5.2	To Create an ST Block	80
5.5.3	To Create a Parameterized ST Block.....	80
5.5.4	To Open an ST Block for Editing	81
5.5.5	Working with the ST Editor Offline.....	81
5.5.6	To Insert an ST Variable or Keyword.....	81
5.5.7	To Create a Variable from a Name	82
5.5.8	To Insert a Line Comment.....	82
5.5.9	To Insert a Block Comment.....	82
5.5.10	To Select a Range of ST Logic.....	82
5.5.11	To Move or Duplicate ST Logic.....	82
5.5.12	To Locate all Occurrences of a Variable	83
5.5.13	Working with the ST Editor Online	83
5.5.14	To View a Variable Value	83
5.5.15	To View the Value of an ST Parameterized Block Parameter	83
5.5.16	To Change a BOOL Variable’s State.....	84
5.5.17	To Force a BOOL Variable’s State	84
5.5.18	To Remove the Force from a BOOL Variable.....	84
5.5.19	ST Statements, Functions, and Function Blocks	84
5.5.20	C Blocks	87

5.5.20.1	Working with C Blocks.....	87
5.5.20.1.1	To Import C Blocks	87
5.5.20.1.2	To Set a C Block’s Parameters.....	87
5.5.20.2	C Programs	88
5.5.21	Working with C Programs.....	88
5.5.21.1	Setting a C Program’s Parameters	88
Section 6:	LD Diagnostic Logic Blocks	89
6.1.1	To Create an Active LD Diagnostic Logic Block	90
Section 7:	PROFINET Support	91
7.1	Ethernet Global Data (EGD)	92
7.1.1	Exchanges vs. Pages	92
7.1.2	Integration with the EGD Configuration Server	93
7.1.3	Integration with the EGD Management Tool.....	93
7.1.4	Logic Developer – PLC implementation of EGD: the EGD Component.....	94
7.1.4.1	To Add the EGD Component	94
7.1.4.2	To Install the EGD Configuration Server	94
7.1.4.3	To Install the EGD Management Tool (EMT) on your Computer	94
7.1.4.4	To Configure Communications with the EGD Configuration Server..	94
7.1.4.5	To Configure a Logic Developer – PLC Target to use the EGD Configuration Server	95
7.1.4.6	To Add a New Produced Exchange and Configure it	95
7.1.4.7	To Publish a Target’s Produced Exchanges to the EGD Configuration Server by using the Validate Method.....	96
7.1.4.8	To Publish a Target’s Produced Exchanges to the EGD Configuration Server by using the Bind and Build Method	96
7.1.4.9	To Synchronize a Consumed Exchange on your Computer with the Corresponding Produced Exchange Published on the EGD Configuration Server	97
Section 8:	PACMotion	98
8.1	To Locate a Target’s PACMotion Node	98
8.2	To Add the PACMotion component to a PACSystems Rx3i Target	98
8.3	CAM Editor	98
8.4	Working with the CAM Editor	99
8.5	To Create a CAM Profile	99
8.6	Data Logging.....	99

8.6.1	To Generate .dlog Files	99
8.7	To Add a Data Logging Window (DLW)	100
Section 9: Motion Programming		101
9.1	To Add a Motion Component to a Target	101
9.2	Motion Editor	102
9.2.1	To Add a Motion Block.....	102
9.2.2	To Open a Motion Block for Editing.....	102
9.2.3	Working with the Motion Editor	103
9.2.3.1	To Insert a Command	103
9.3	Local Logic.....	105
9.3.1	To Create a Local Logic Block.....	105
9.3.2	To Open a Local Logic Block for Editing	106
9.3.2.1	Working with the Local Logic Editor.....	106
9.3.2.2	To Insert a Local Logic Command	107
9.3.3	Local Logic Variables	107
9.3.3.1	To View the LLVT	107
9.3.4	To Insert a Local Logic Variable	108
9.3.4.1	Local Logic Commands and Operators	108
9.3.5	CAM Editor.....	109
9.3.5.1	To Create a CAM Block	110
9.3.5.2	To Import CAM Blocks	110
9.3.5.3	To Open a Block for Editing	110
9.3.5.4	Working with the CAM Editor	110
9.3.5.5	To Configure a CAM Profile.....	111
9.3.5.6	To Edit a CAM Profile	111
9.3.5.7	To Add a CAM Block	112

Warnings and Caution Notes as Used in this Publication

WARNING

Warning notices are used in this publication to emphasize that hazardous voltages, currents, temperatures, or other conditions that could cause personal injury exist in this equipment or may be associated with its use.

In situations where inattention could cause either personal injury or damage to equipment, a Warning notice is used.

CAUTION

Caution notices are used where equipment might be damaged if care is not taken.

Note: Notes merely call attention to information that is especially significant to understanding and operating the equipment.

These instructions do not purport to cover all details or variations in equipment, nor to provide for every possible contingency to be met during installation, operation, and maintenance. The information is supplied for informational purposes only, and Emerson makes no warranty as to the accuracy of the information included herein. Changes, modifications, and/or improvements to equipment and specifications are made periodically and these changes may or may not be reflected herein. It is understood that Emerson may make changes, modifications, or improvements to the equipment referenced herein or to the document itself at any time. This document is intended for trained personnel familiar with the Emerson products referenced herein.

Emerson may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not provide any license whatsoever to any of these patents.

Emerson provides the following document and the information included therein as-is and without warranty of any kind, expressed or implied, including but not limited to any implied statutory warranty of merchantability or fitness for particular purpose.

© 2019 Emerson. All rights reserved.

Emerson Terms and Conditions of Sale are available upon request. The Emerson logo is a trademark and service mark of Emerson Electric Co. All other marks are the property of their respective owners.

Section 1: Welcome

Congratulations on your purchase of PAC Logic Developer - PLC™, the programming component of PAC Machine Edition™ automation software for PACSystems™, Series 90™, and Versamax™ controllers.

This software package provides all the tools necessary to create powerful control applications. Logic Developer - PLC provides a way to configure your Controller hardware or remote I/O, create and edit logic, upload and download projects, and monitor and debug the execution of control programs. Projects can be imported from Logicmaster™, VersaPro™, and Control folders.

Hosted in the Machine Edition environment, Logic Developer - PLC takes advantage of a powerful set of common programming tools. The same tools can be applied to Logic Developer - PC (PC Control) and View, providing a single programming environment. The Machine Edition environment unites and organizes components, providing data sharing and networked operation.

The following features are included in this version of Logic Developer - PLC:

- Hardware Configuration
- LD Editor
- FBD Editor
- IL Editor
- ST Editor
- C Blocks
- C Programs
- DLBs
- PACMotion
- Motion Editor
- Local Logic Editor
- CAM Editor

To use Logic Developer - PLC and its tools, you require the following minimum requirements:

- 32-bit or 64-bit Windows 7 SP1 Ultimate, Windows 7 SP1 Enterprise, or Windows 7 SP1 Professional.
- Must be part of the Administrators group.
- Windows regional settings must be set to English.

CAUTION

On 64-bit Windows 7 platforms, the property column display in the Variables tab of the Navigator is not supported. For more cautions and limitations when using Logic Developer on Windows 7, refer to the Important Product Information (IPI) available on the support site. Links to the support site are located at the end of this document.

1.1 Processor Speed and Memory

Minimum (for small to medium-sized projects):

- Intel Core i5 with 4 GB RAM
- AMD FX or Phenom II X6 with 4 GB RAM
- Strongly recommended for large projects to see multi-threading performance gains:
- Intel quad-core Core i7 with 8 GB RAM and 64-bit Windows 7 SP1
- AMD higher FX or Phenom II X6 models with 8 GB RAM and 64-bit Windows 7 SP1

1.2 Extra Requirements When Executing Multiple Instances of Engineering Workstation

Minimum requirement for very large projects and strongly recommended for any project:

- 64-bit versions of Windows 7 SP1
- Possible for small or average-sized projects, but not recommended: 32-bit versions of Windows 7 SP1
- Average-sized projects: each instance uses approximately 500 MB of RAM.
- Very large projects:
- 8 GB of RAM in total
- Limit the number of instances to 2 or 3

1.3 VMWare Support

The Development Environment is supported on VMware with the following VMware requirements:

Minimum requirement

- Player 2.0 or greater, or
- VMware Workstation 6.0 or greater.
- For multiple processor cores (required to leverage multi-threading performance gains):
- Player 4.0 or greater, or
- VMware Workstation 8.0 or greater.

CAUTION

After powering down the guest operating system (O/S) of a VMware virtual machine, configure the number of cores to use for that virtual machine: match or come as close as possible to the number of cores in your computer's processor. You cannot configure the number of cores for a virtual machine if its guest O/S is running or paused.

The APM Motion Programmer does not function if PAC Machine Edition is executed from within VMware.

Other Requirements

- Internet Explorer 10.0, 9.0, or 8.0; with current updates.
- You must install Internet Explorer before installing Machine Edition.
- TCP/IP network protocol (if you use an Ethernet connection).
- .NET Framework 4.5 Full. If the Microsoft .NET Framework is not yet installed, it is automatically included during the installation of Machine Edition, and a reboot may be required to complete the installation.
- 2 GB hard disk space. Additional space is required for your projects and temporary files.

1.4 Installation

For last-minute information, release notes, and supported hardware lists for Machine Edition products, see the Important Product Information (IPI) document on the install disk. There are several ways to view this document:

- When installing Machine Edition, select Important Product Information on the initial Launcher screen.
- When running Machine Edition, the IPI is accessed by the App tab on the File menu.

If you have a previous version of Machine Edition installed on your computer, you must uninstall it before installing the latest version. All your existing projects, settings, and authorizations are preserved following an uninstall operation.

1.4.1 To Install Logic Developer – PLC

1. Insert the Machine Edition install disk into your CD-ROM drive.

By default, the setup program automatically starts. If the setup program does not automatically start, run *PACSetup.exe* in the root directory of the install disk.

2. Click *Install* to start the install process.
3. Follow the instructions as they appear on the screen.

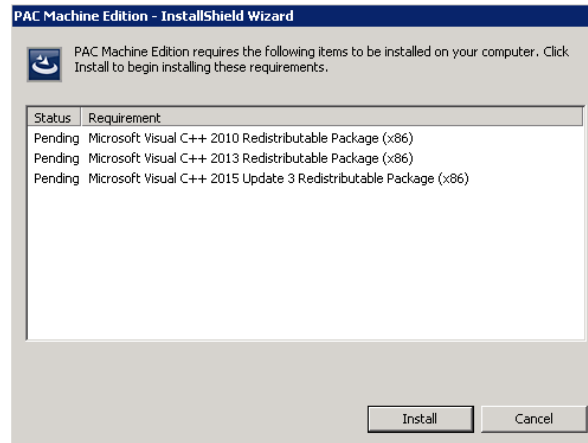
For information on troubleshooting installation problems, see the IPI Release Notes.

1.4.2 Microsoft Visual C++

PAC Machine Edition requires the latest Microsoft Visual C++ packages to be installed before continuing (Figure 1).

If you have a newer version of the redistributable already installed, a message may appear that the installing packages has failed. Choose *Yes* to continue with the installation.

Figure 1: Microsoft Visual C ++



1.5 Product Authorization

A new installation of Logic Developer - PLC provides a 4-day trial license with full access to Logic Developer - PLC features. This license overrides all other licensing and cannot be removed. Any licensing added is apparent on the fifth day.

1.5.1 Automatic License Activation

Customers can license their copy of Logic Developer – PLC using Emerson’s Entitlement and License Manager software. There are two methods to initiate the activation wizard: from within the application (PME) or from the Entitlement and License Manager itself.

During or following the trial period of PAC Machine Edition, customers can select the *Activate a License* button to initialize the License Activation screen from the Entitlement and License Manager and follow the activation wizard’s prompts. This section will demonstrate how to license PME from within the application.

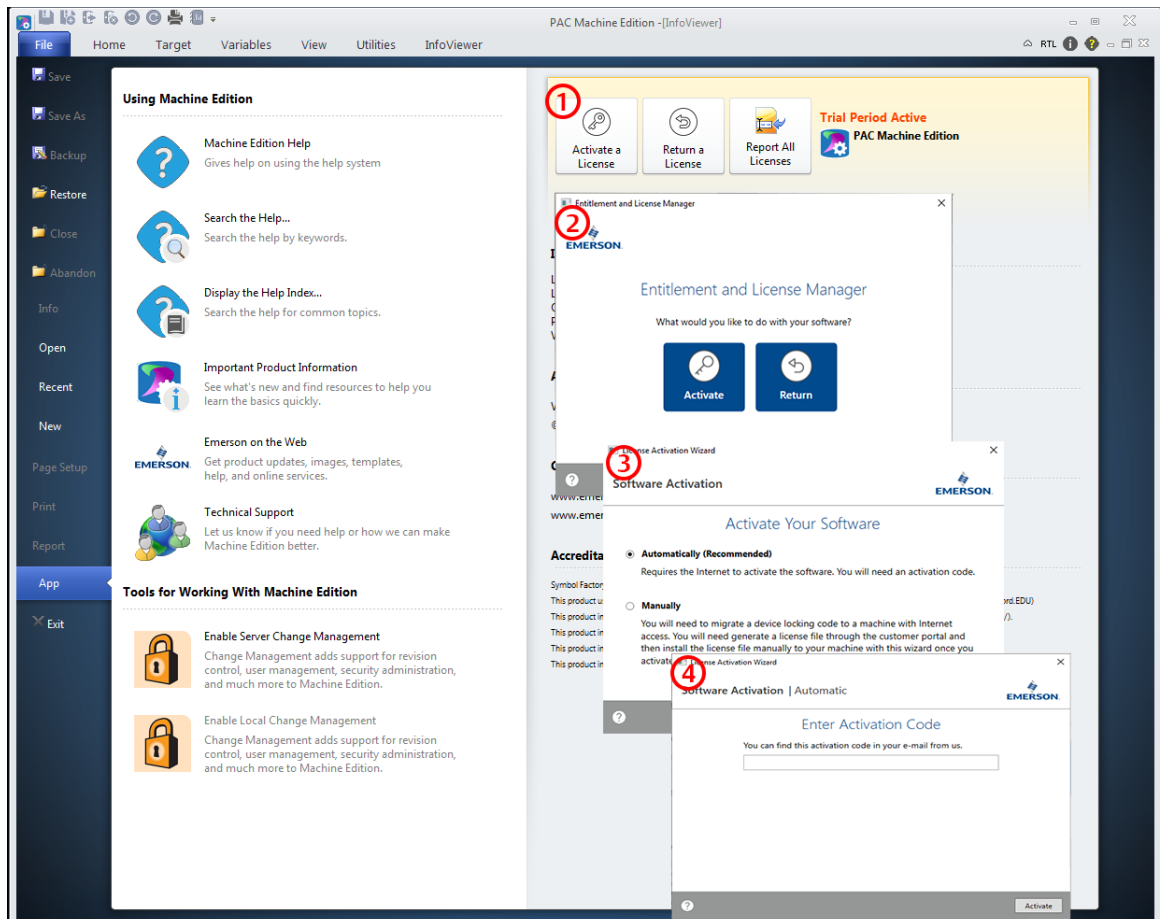
Alternatively, customers can open Entitlement and License Manager directly and activate their copy of PAC Machine Edition. For more information on using Entitlement and License Manager, please refer to GFK-3104.

Automatic Activation Steps:

- 1) Navigate to the *File* menu on the ribbon bar.
- 2) Select the *App* tab from the menu.
- 3) Click the *Activate a License* button located on the upper-right side of the menu (Figure 2, Item 1). A new window will open to the Entitlement and License Manager.
- 4) Click the *Activate* button and select the Activation Method (Figure 2, Item2).
- 5) Select *Automatically* to automatically license your product (Figure 2, Item 3).
- 6) Enter the activation code on the next screen (Figure 2, Item 4)

Note: For more information on the manual activation of your software, please refer to GFK-3104.

Figure 2: Activate a License Button



Section 2: PAC Machine Edition

PAC Machine Edition offers you a complete solution for the development of automation applications, in one package. Machine Edition features an integrated development environment and tools that enable you to spend more time building applications and less time learning the software. All Machine Edition products are fully integrated into the environment and interact with each other.

- They share the same set of tools providing a consistent interface throughout the development process
- Full drag-and-drop capabilities between tools and editors
- A true scalable solution: you choose the type of Controller your projects run on

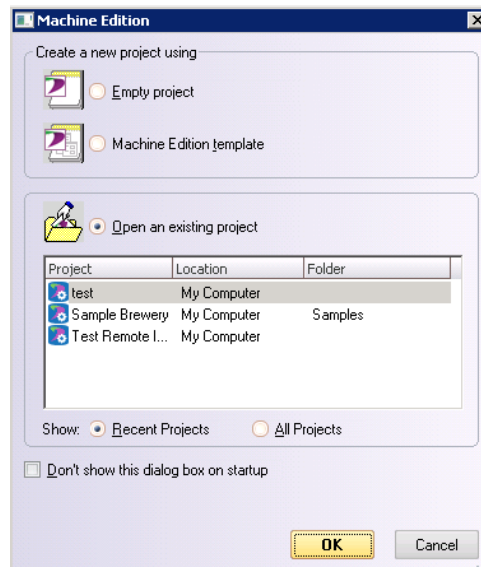
2.1 Quick Start

2.1.1 To Start Machine Edition

Click the Start Button, point to *Programs, Emerson*, and then choose *PAC Machine Edition*.

The Machine Edition dialog box (Figure 3).

Figure 3: Machine Edition Dialog



1. Select the appropriate option to open a project. The *Open an Existing Project* option is selected by default

Notes: If you select either the *Empty project* option or the *Machine Edition Template* option, the *New Project* dialog box appears and you can continue creating a new project.

2. If you select the *Open an Existing Project* option, select from the list the project that you want to open.
3. Optionally, select the *Don't show this dialog box on startup* option.
4. Click *OK*.

Your project opens in the Machine Edition environment.

2.2 Projects

You can create and edit Machine Edition projects by using PAC View, Logic Developer - PC, and Logic Developer - PLC. These products share Machine Edition tools to provide a high level of integration between the various parts of your project.

You can import folders created with Logicmaster, Control, or VersaPro.

With Logic Developer - PLC, you can build multiple projects to suit your specific requirements.

2.2.1 To Create a New Project Using a Template

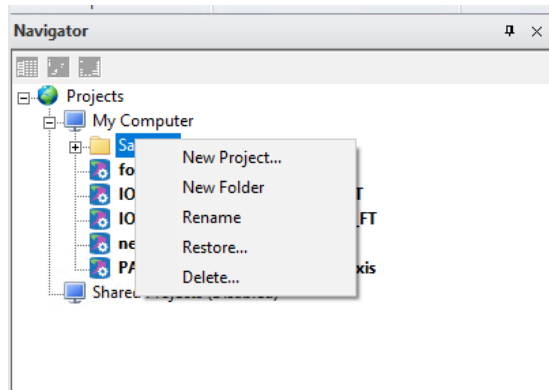
Before creating a project, there are some things you should know:

- The primary components your project will contain.
 - The PACSystems controller your project will run on.
1. From the *File* menu, select the *New* tab to reveal available templates
 2. Enter a project name in the text field above.
 3. Select a save location from the drop down menu.
 4. Click to choose from one of the available templates.
 5. Click *Create* located on the far-right to begin a new project.

2.2.2 To Open an Existing Project for Editing to Open an Existing Project for Editing

1. Open the Navigator and select the Manager tab. A list of projects appears.
2. Select a project to open by double-click the project or right-clicking and selecting *Open*.
3. If the project is not visible, right-click *My Computer* in the Manager tab (Figure 4).
4. Select *Restore* to browse for the project file location on your PC.
5. Navigate and select the required project file and select *Open*.

Figure 4: Opening Existing Projects



2.2.3 To Import a Folder

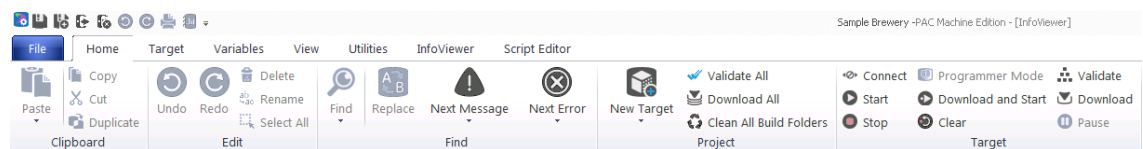
1. Open the Navigator and select the Project tab.
2. Select the target that you want to import the folder into.
3. Right-click the target, point to *import*, and choose the folder type.
4. In the dialog box that appears, navigate to and double-click the folder you want to import.

2.3 Interface Overview

2.3.1 Ribbon Bar

PAC Machine Edition uses a ribbon bar (Figure 5) for its interface. The interface commands are categorized by tabs and further organized by sub-groups for ease of use. Descriptions of each button functionality is available in the Companion pane.

Figure 5: Ribbon Bar



2.3.2 Utility Panes

The Navigator contains a set of tabs. Each tab displays information about your development system in a hierarchical tree structure like Windows Explorer. The available tabs depend on which Machine Edition products you have installed and what kind of work you are developing or managing. The Project tab shows the overall organization of your application.

The Feedback Zone window is used to display several types of output information generated by Machine Edition components. This interactive window uses category tabs to organize the output generated from the Machine Edition products you have installed.

The Inspector window lists the properties and current settings for a selected object or element. You can edit these properties directly in the Inspector. When you select several objects, the Inspector window lists the properties common to all of them. The Inspector provides a simple method of viewing and setting properties for all Machine Edition objects.

The *Data Watch* tool is a run-time debugging tool that enables you to monitor and edit the values of variables. This tool is useful while working online to a target. With the Data Watch tool, you can monitor individual variables or user-defined lists of variables. Data Watch lists can be imported, exported, or saved with a project.

The Toolchest is a powerful storehouse of objects you can add to your project. You can drag most items directly from the Toolchest into Machine Edition editors. You can choose from predefined objects or create your own reusable fxClasses. The Toolchest adds true object-oriented capability to Machine Edition.

The Companion provides useful tips and information while you work. While the Companion is open, it tracks your moves and displays help on whatever item is currently selected in the Machine Edition environment. It is context-sensitive and displays a description of whatever you click in Machine Edition.

The InfoViewer is an embedded Web browser used mainly to display the following:

- Machine Edition help
- Machine Edition reports
- The documentation associated with a project or target

If you are familiar with Internet Explorer, then you are already familiar with the basic InfoViewer interface. Like the Companion, the InfoViewer is context-sensitive. Press F1 to get help on any item you select in the Machine Edition environment.

A table of contents is found in the InfoView tab of the Navigator.

2.3.3 Using Docking Markers

If the Show Docking Markers option is enabled, you can use docking markers to help dock a tool window to a desired location.

As you drag a dockable window, a series of docking markers appear, indicating valid docking locations. Docking markers appear as a series of blue arrows.

Move the mouse over a docking marker to preview how the window would be docked when using that marker. Release the mouse button while hovering over a marker to use that location.

2.4 Variables

A variable is a named storage space for data. A variable name represents a memory location in the target Controller. The way in which the variable represents a memory location is determined by the value of the variable's Ref Address property. A variable's Ref Address property can be set to any of the following:

A reference address, for example, %R00001. This is the PACSystems Controller memory location that contains the variable's value. This can be an I/O register or an internal memory register on the Controller.

A blank. On a PACSystems, if you leave a variable's Ref Address property blank, the variable is a symbolic variable. Machine Edition handles the mapping for you in a special portion of PACSystems user memory space. On Series 90 and Versamax Controllers, you cannot leave the Ref Address property blank.

An I/O variable address, for example %IX0.6.0.1. (PACSystems only.) This represents the PACSystems Hardware Configuration terminal that contains an I/O variable's value. For example, this can be a physical discrete or analog I/O point on a module or Genius device, a discrete or analog status returned by the module, or global data.

- A name. This makes the current variable an alias variable of the name. An alias variable will point to the same memory location as the name. The following names may be valid:
- The name of a parameterized LD block parameter (provided the alias variable is local to the same block)
- The name and index of an array element, for example, MyComplexArray[255,3]
- The name of a structure element, for example, MyStructure.MyElement
- The name of another variable that is not used as an alias variable.

Arrays and compound structure variables are supported by Machine Edition. Variable definitions can be imported from and exported to a variety of file types. You can edit your variables in a spreadsheet and then import them.

Variables used on PACSystems targets must be externally published in order to be viewed in PAC View.

2.4.1 User-Defined Data Types

(PACSystems only.) A User-defined Data Type (UDT) is a structured data type consisting of various elements of various other data types. You can define a UDT to reside entirely in discrete memory or non-discrete memory. After defining a UDT, you can create structure variables of that data type and use them in logic.

2.4.2 Variable Properties in the Inspector

The Ref Address and other properties of a variable, such as Data Type, are configured in the Inspector. The following is an illustration of the Inspector displaying a typical set of variable properties.

1. In the Variables table of the Navigator, right-click *Variable List*, point to *New Variable* and then choose the data type of the variable. If only one target is in your project, then Machine Edition creates a new variable with a default name.

If more than one target is in the project, then the New Variable dialog box appears with a unique default name for the new variable.

2. (Optional) Enter a unique name for the variable.
Variable names can range from 1 through 32 characters, begin with a letter or the \$ character, contain upper or lower case letters, use numbers between zero and nine, and use the underscore character.
3. Select the target in which the variable will be used.
4. If the new variable is an array, select the Array check box and enter the size.
5. Click *Ok*.
The new variable appears in the list in the Variables tab.

2.4.3 To Map a Variable to Controller Memory or Alias a Variable

There are two ways to map a variable to Controller memory or alias a variable

2.4.3.1 First Method

1. In the Variables tab of the Navigator, right-click a variable and choose *Properties*.
2. In the inspector that appears, in the *Ref Address* field, enter a reference address to map the variable or enter the name of another existing variable, a parameterized LD block parameter, an array element, or a structure element to alias the variable.
 - Spell out the exact reference address, for example, %R00123 or 123R (in either case this maps the variable to %R00123), or enter only the memory area, for example %R. This maps the variable to the next available address in that memory area. For example, if %R00122 is the last address used by a 16-bit variable, entering %R maps the variable to %R00123.
 - Spell out the name of another variable, or of a parameterized LD block parameter, for example, MyVarWithALongName, or ABC. Or provide the array's name and the element's number, for example, MyArray[5]. Or provide the structure element's name, for example, MyStructure.MyElement.
 - Notes: (PACSystems only)
 - If you leave the Ref Address property of a variable blank, the variable is a symbolic variable. You can enter an I/O variable address to make the variable an I/O variable, but it is easier to map an I/O variable in the terminals tab of a module.

2.4.3.2 Second Method

1. In the Variables tab of the Navigator, right-click a variable and choose *Properties*.
2. In the Inspector that appears, click the *Ref Address* field and then click the ... button.

The reference Address Wizard appears.

Do one of the following

To map a variable to Controller memory: From the *Memory Area* list, choose a region of Controller memory. In the *Index* box, enter an index from the beginning of the region. Select a bit reference. The *Bit Reference* box is available only if you are mapping a BOOL variable to 16-bit memory area on a PACSystems. Click *OK*.

- or -

To alias a variable: In the *Variable Aliasing Filters* section, select the appropriate check boxes for the filters you want to use. In the *Alias variable to* box, select what you want to alias the variable to. Click *OK*.

2.5 Options

The Options tab of the Navigator contains option and preference settings. Options are organized into folders and pages. Click or to expand or collapse folders. Examples:

Controller > General > Duplicate Addresses: Indicates whether to prevent, warn about, or ignore mapping two variables of the same data size (1, 8, 16, 32, or 64 bits) and length (array size) to the same reference address.

Controller > Hardware Configuration > New Reference Assignment: The way in which default reference addresses are assigned when modules are added to the Hardware Configuration.

Editors > Ladder > View > Coil Justification: The default column in which coils are placed; also the column where the right power rail resides. The default is 10.

Machine Edition > Preferences > Visual Style: The visual style used for the Machine Edition environment, including the appearance of toolbars, tool windows, and menus. You can choose from several visual styles based on different versions of Microsoft products and the Windows operating system: Classic (based on Windows 2000), XP, 2003, and 2005.

2.5.1 To Set Options and Preferences

1. In the Options tab of the Navigator, expand an options folder and right-click a page within the folder to display the options in the inspector.
2. In the inspector, edit the option's settings as needed.

Tip: When you click an option, the Companion automatically displays help on that option. If the *Companion* is not already open, press Shift +F11 to open it.

2.5.2 To Reset an Entire Page of Options to their Default Settings

1. In the Options tab of the Navigator, expand an options folder
2. Right-click one of the options pages, and then choose Reset.

Tip: To reset only one option, look up its default value in the Companion and set the option to that value.

2.6 Machine Edition Help

Machine Edition includes a comprehensive online help system that enables you to access specific help topics while working with Machine Edition Environment.

2.6.1 Companion Help

The companion is a Machine Edition help system that provides useful tips and information while you work. While the Companion is open, it displays help on whatever item is currently selected in the Machine Edition environment.

2.6.1.1 To Use Companion Help

1. Ensure that the Companion is open. To Open it, press Shift +F11.
2. Click an item in Machine Edition located outside the Companion panel. A description of the item you clicked appears in the Companion.

2.6.2 InfoViewer Help

The InfoViewer provides detail help information. The InfoViewer has the following features:

- Table of Contents
- Index
- Full-Search Engine
- Toolbar for Navigating the Help System

InfoViewer help is context-sensitive. Click an item on the screen and press F1 to display the appropriate topic in the InfoViewer.

2.6.2.1 To Get Help with Machine Edition

There are three main ways to get help.

1. From the File menu, select the App Tab to display information about the App..
 - a. To search for help by keyword, select *Search the Help...*
 - b. To search for help on common topics, select *Display the Help Index...*
 - c. For further information on using the help system, select *Machine Edition Help*.

2.6.2.2 To Use the Full-Text Search

You can use the standard HTML Help Viewer to perform a full-text search of Help. Advanced Boolean search expressions can be applied.

1. From the Help menu, choose *Search*.

The Machine Edition Help Search dialog box appears in a separate window.

2. Enter a search word or phrase in the top text box.

Note: Be sure to surround a phrase with quotation marks.

3. Click enter to perform a Boolean search. Use AND, OR, NEAR, or NOT to create a Boolean search expression.
4. Click *List Topics* to display available topics. The topics are sorted by their rating or likelihood of containing the search term, terms, or phrase.
5. In the Select topic box, select a topic to display.
6. Click *Display*
The selected Help topic is displayed in the Help Viewer
7. (Optional) Click the Info Viewer button to display the current topic in the InfoViewer.

Tips: To narrow the search results, click the Location heading. The listed topics are sorted by location and then by rating.

- When using the HTML Viewer window, you may get better results if you select the *Select Titles Only* checkbox and/or clear the *Match Similar Words* checkbox.

2.6.2.3 To Bookmark Favorite Topics

1. After to perform a full-text search, select a topic you want to add to your list of favorite topics and then click the *Favorites* tab.

The machine Edition Help Search dialog box displays the selected topic title in the Current topic text box (bottom-left corner).

2. Click *Add to add the topic to the Topics* list.
3. (Optional) Click the InfoViewer button to display the current topic in the InfoViewer.
4. To display a favorite topic in the Help Viewer, select it and click *Display*.
5. To remove a topic from the Favorites list, select and click *Remove*.

2.6.2.4 To Look Up Topics in the Help Table of Contents

1. Click the InfoView tab of the Navigator
A table of contents for the entire help system appears.
2. Expand Libraries and Books to locate a topic of interest.
3. Double-click a topic. The topic is displayed in the InfoViewer

Section 3: PACSystems Targets

A target represents a run-time destination of the program or programs you develop with Logic Developer - PLC. Each target contains all of the components associated with that target. Logic Developer - PLC supports the following PACSystems families of Controllers:

- PACSystems RXi
- PACSystems RX3i
- PAC Systems RSTi-EP Standalone
- PACSystems RX3i Rackless
- Versamax
- Versamax Nano/Micro
- Series 90 Micro

and the following PACSystems remote I/O interface targets:

- PACSystems RX3i Ethernet
- Versamax Ethernet
- Versamax Genius
- Versamax Profibus

3.1 Adding, Configuring, and Converting Targets

3.1.1 Adding Targets

A target will be present in a project when you use a template to create the project. However, a project can contain multiple targets. Before grouping targets into a project or isolating each target in its own project, consider the following:

- How large are the targets? For example, a target that contains 200,000 variables should probably be by itself in a project.
- Are the targets related? For example, a PACSystems RX3i with a relatively small program is connected to twenty PACSystems Remote I/Os (RIOs). Size is not an issue because RIO targets use little memory in a project. Grouping all the targets into one project is appealing because it makes navigation much easier from one target to another than if the PACSystems RX3i target and RIOs are isolated in 21 separate projects. For another example, if you have various targets that do not communicate with one another or do not serve a common purpose, there may be little value in grouping them in one project.
- One target is required for each Controller or remote I/O adapter your project accesses, except when you are using CPU redundancy. In this case, one target contains a Primary Hardware Configuration and a Secondary Hardware Configuration, which correspond to the primary Controller and the redundancy Controller, respectively.
- Existing targets can be converted from one Controller family to another.

3.1.2 Configuring Controller Targets

The properties of a target specify the Controller family, the communication connections between your computer and the Controller, and various other settings. All properties are edited in the Inspector. The following table describes common Controller target properties:

Table 1: Controller Target Properties

Name	Edit the name for your target in this field.
Type	The type of target is set by default to PACSystems Controller.
Description	Enter a description of your project in this field. The maximum number of characters is 255.
Documentation Address	Enter the URL where your project documentation is stored.
Family	By changing the Controller type in this field, you initiate a target conversion. Caution: Changes are irreversible.
Controller Target Name	The name of the target as stored on the Controller.
Update Rate (ms)	Set the rate at which the screen is updated while online to the target.
Sweep Time (ms)	The sweep time of the Controller when online. This value is also displayed on the status bar. (Read-only.)
Controller Status	The online/offline, run/stop status of the Controller. (Read-only.)
Online Project Synchronization	When Enabled, Machine Edition performs project synchronization checks with an online Controller.
Physical Port	Choose the type of connection to the Controller (Ethernet, COM, or modem).
IP Address	(Ethernet protocol only.) Set the IP address of the Controller.
Additional Configuration	Group of properties used for the detailed configuration of your communication connection.
Modem Communication	Group of properties to configure the modem communications link between your computer and a remote GE target.

Other properties are available depending on the value of the Family property and the CPU model.

3.2 Converting Targets

With Logic Developer – PLC, users can convert targets from one Emerson controller family type to another. For example, the Hardware Configuration and logic written for a Series 90-70 Controller and convert them for use on a PACSystems RX3i. However, target conversions are irreversible; when logic blocks are deleted during a conversion, they cannot be restored. Emerson recommends that you make a backup of your project before converting a target.

3.2.1 To Convert a Target

1. In the Project tab of the Navigator, right-click a target and choose *Properties*.
The inspector displays the target's properties.
2. In the Inspector panel, click *Family*.
3. From the list, choose the new Emerson controller family you want to convert the target to.

There are two types of target conversions.

A basic conversion does the following:

- Strips all the configured modules from the original Hardware Configuration (HWC) and sets up the destination family's default HWC, with a single power supply and a CPU.
- Deletes or adds target components, logic programs, or blocks of logic.
- Updates the system variables including the fault locating references if applicable

An enhanced conversion does the following:

- Replaces a power supply with the destination family's default power supply.
- Retains the settings of CPU parameters common to the original and destination CPUs when the settings are supported in the destination family; otherwise, replaces them with the default settings.
- Retains all the module parameter settings that are supported by the destination family; otherwise, the default settings are used.
- Ethernet module settings of non-supported Ethernet modules are used to configure the destination's Ethernet daughterboard or default Ethernet modules.
- Converts unsupported expansion racks to the nearest equivalents.
- Deletes or adds target components, logic programs, or blocks of logic.
- Updates the system variables, including the fault locating references if applicable.
- Displays a conversion report in the InfoViewer.

CAUTION

The capabilities of a destination target may be different from those of the original target. Carefully examine the conversion report when it is available. Validate the project and test it thoroughly before deploying it in production.

Note: The hardware must be configured before it is operational.

3.3 Configuring Communication

For Logic Developer – PLC to communicate with a target Controller, a connection must be properly configured. The properties of a target are adjustable to accommodate your connection (s).

3.3.1 To Configure an Ethernet, Modem, or Serial Connection with any PACSystems Controller.

1. The Project tab of the Navigator, right-click a target and choose *Properties*.
The Inspector displays the target's properties.
2. In the Inspector, set the Physical Port property to Ethernet or a COM port (or a modem if one is installed).
3. If the Physical Port is Ethernet, enter the IP Address of the target Controller.
4. Double-click *Additional Configuration* to access the detailed settings for your connection
Note: An IP address must be set in the Controller before an Ethernet connection can be established.

3.3.2 To Set a Permanent IP Address for a PACSystems

1. In the Project tab of the Navigator, right-click the target and choose *Properties*.
The inspector displays the target's properties.
2. In the Ip Address property, enter an IP address.
3. Expand the Hardware Configuration.
4. Do one of the following:

- For a PACSystems RXi, expand the PACSystems RXi node, and double-click the Ethernet node, or
 - For a PACSystems Rx3i, expand the main rack, double-click every IC695ETM001 Ethernet module and repeat step 5 for each one of them.
5. In the Settings tab that appears in the Parameter editor, set the IP Address, the Subnet Mask, and the Gateway UP Address
 6. Download the Hardware Configuration to the PACSystems.

3.3.3 To Download an IP Address via a Controller Serial Port

1. Configure a CPU or Ethernet communications module with an IP Address using the Hardware Configuration.
2. Right-click the target and choose *Properties*. The Inspector displays the target's properties.
3. In the Physical Port property, select a serial connection.
4. Right-click the target and choose *Go Online*.
5. Right-click the target and choose *Download to Controller*.
The Download to Controller dialog box appears.
6. Select *Hardware Configuration* and click *OK*.
The IP address is assigned to the controller and the rest of the Hardware Configuration is downloaded to the Controller.
7. Right-click the target and choose *Go Offline*.
8. Right-click the target and choose *Properties*.
9. In the Physical Port property, select *Ethernet*.
The next time you go online, Machine Edition will use an Ethernet connection with the specified IP address.

3.4 Interacting with a Controller

Communicating with a PACSystems or VersaMax Controller is necessary to perform such operations as storing and loading or monitoring data values and Controller Status information. You can to a Controller from Logic Developer – PLC over a serial, Ethernet, or modem connection, depending on the controller's capabilities.

All interactions with a target are available from the target's right-click menu.

3.4.1 Validating a Target

Validating your target detects syntax and configuration errors on the target. Error messages are generated for each error and displayed in the Feedback Zone.

Tip: Double-click an error message to locate the noted error in your project. The appropriate editor or tool opens automatically with the item in question selected. Press the F4 key to locate the next

error warning in your project. Tips show you how to proceed are displayed in the Companion. To open the Companion press SHIFT +F11.

3.4.2 To Validate a Target

In the Project tab of the Navigator, right-click a target and choose *Validate*.

Logic Developer – PLC checks all items under the target for syntax errors. Any errors detected are noted in the Build tab of the Feedback Zone.

Tip: Double-click an error message to locate the noted error in your project. The appropriate editor or tool opens automatically with the item in question selected. Press the F4 key to locate the next error or warning in your project. Tips showing you how to proceed are displayed in the Companion, press SHIFT + F11.

3.4.3 Offline, Online: Monitor Mode, Programmer Mode

When offline from a PACSystems or VersaMax, there is no ongoing communication between the Controller and your development computer. A physical communication link is not required as long as you only edit logic; it is required only when you want to communicate with the Controller.

The only Controller operations you can perform while offline are to go online or to set up the temporary IP address.

When online with a target Controller, a communication link exists and is active and you have an ongoing real-time communication with the Controller.

When online in monitor mode, you can monitor the Controller while it is executing. You cannot edit logic on your computer. You cannot change any values on the Controller. Depending on your level access on the Controller and your Change Management permission levels, you can upload from the Controller.

When online in programmer mode, you can make changes on your computer and the Controller and can monitor the Controller while it is executing. You can edit any type of Controller logic on your computer. Depending on your level of access on the Controller and your Change Management permission levels, you can:

- Upload from the Controller
- Control the Controller while it is executing
- Control any change values on the Controller
- Download to the Controller

When editing LD logic while online, you can make word-for-word changes; on PACSystems, you can also use the test edit feature.

When online in either mode, if the project stored on the Controller is equal to the current project in Logic Developer - PLC, the LD editor displays a graphical representation of LD logic as it executes.

3.4.4 To Go Online to an Emerson Controller

1. In the Project tab of the Navigator, ensure that no target controller is already online.
You can be online to only one target at a time. When a target is offline, its icon is grey.
2. If the name of the target you want to go online to does not appear in bold characters, right-click the target and choose *Set as Active Target*.
3. Do one of the following:
 - Right-click the active target and choose *Go Online*, or
 - Click the thunderbolt on the Online toolbar.

Logic Developer – PLC connects your project to the Emerson controller. The online status is indicated by the target con in the Project tab and on the status bar.

Note: When online to an Emerson controller, the target icon in the Project tab of the Navigator appears as equal, not equal, or stop failed.

3.4.5 To Change the Online Mode

In the Project tab of the Navigator:

1. Right-click the target controller
2. Point to *Online Commands*
3. Choose *Set Programmer Mode* or *Set Monitor Mode*

3.4.6 To Go Offline from an Emerson Controller

Do one of the following:

- In the Project tab of the Navigator, right-click a target controller and choose *Go Offline*; or
- Click the thunderbolt on the Online toolbar.

3.4.6.1 Upload/Download

The download process builds and validates all run-time files necessary for a target to perform its role in a completed project. The compiled project is then transferred to a target hardware over the communication connection previously configured.

The upload process acquires a project from the active PACSystems controller target and transfers it to the Logic Developer – PLC for editing.

3.4.7 To Download to a PACSystems Controller

1. Ensure that you are online in programmer mode to the target Controller.
2. In the Project tab of the Navigator, right-click the target to which you want to download files and choose *Download to Controller*.

The Download to controller dialog box appears, displaying these or other options. *Note:* If the Controller is running, you can download only logic that is not equal to the Controller's current logic and the Download to Controller dialog box does not appear. On a PACSystems, some source files that do not affect equality are also downloaded.

3. Choose the items you want to download and click *OK*.

Note: Only one project can be downloaded to a target at a time. If you download to a target controller that already has a project on it, the existing project is overwritten.

For each target that you download, Machine Edition performs a validation. Any errors that occur are displayed in the Build tab of the Feedback Zone. If there are no errors, Machine Edition builds and sends all the necessary run-time files to the Controller.

3.4.8 To Upload Files from an PACSystems Controller

1. Ensure that you are online to the target Controller.
2. In the Project tab of the Navigator, right-click the target Controller from which you want to upload information and choose *Upload from Controller*.

The Upload from controller dialog box appears, displaying these or other options.

3. Choose the item(s) you want to upload and click *OK*.

The selected items are uploaded to Logic Developer - PLC. If you already had a version of the project open, the uploaded project merges with the existing project. Because variable names are not stored on Series 90 or VersaMax Controllers, if you upload to an empty target, all variables are assigned default names. For example, %R00001 is named R00001.

3.4.8.1 Run/Stop

When you are online in programmer mode, you can set a target Emerson control to Run or Stop mode. In Stop mode, you can choose to enable or disable the outputs.

3.4.9 To Start a PACSystems Controller

Do one of the following:

- In the Project tab of the Navigator, right-click a target, point to *Online Commands*, and then choose *Start Controller*; or
- Click on the Online toolbar.

The target controller begins executing its program.

Note: If you are starting a PACSystems, you can choose to have outputs enabled or disabled.

3.4.10 To Stop a PACSystems Controller

Do one of the following:

1. In the Project tab of the Navigator, right-click a target
2. Point to Online Commands
 - Choose *Stop Controller*Alternatively:
 1. Click on the Online toolbar. The Stop Controller dialog box appears, prompting you to enable or disable the controller's outputs.
2. Select an option.
3. Click *OK*. The target controller stops executing its program.

3.4.11 Fault Tables

The controller and I/O fault tables display fault information logged by the CU or modules in the controller. This information is used to determine if there are problems with the controller hardware or software running in the controller's CPU.

3.4.11.1 To View the Fault Table Reports

Do one of the following:

1. In the Project tab of the Navigator, double-click the target you want a fault table report on, or right-click it and choose *Diagnostics*.

-or-

2. In the status bar at the bottom of Machine Edition, double-click the target.

Note: To view the controller and I/O fault tables, your computer must be online to the controller. To clear faults, you must be in *Online Programmer* mode.

3.4.11.2 Reference View Tables

In Reference View Tables (RVTs), you can monitor reference data when you are online to a PACSystems Controller. If you are in online programmer mode, you can also use RVTs to change the values of reference data. In the Project tab of the Navigator, the Reference View Tables folder contains a Default Tables folder. You can add user-defined tables to the Reference ViewTables folder. A target can have zero or more user-defined RVTs.

You can configure the default appearance of your RVTs in the Options tab of the Navigator. For more information, see section *Options* on page 12.

Data values at sequential addresses are displayed from right to left, by default, starting at the reference address specified in the Address column. Both default and user-defined RVTs display rows of 8 cells for discrete memory (each cell corresponding to 8 bits) and rows of 10 cells for register memory (each cell corresponding to one 16-bit register). The amount of data displayed in the columns depends on the data display format.

3.4.12 To Create a User-Defined Reference View Table

In the Project ab of the Navigator, right-click the Reference View Tables folder and choose *New*.

A new reference View Table with a default name is added to the folder.

3.4.13 To Work with a User-Defined Reference View Table.

1. In the Project tab of the Navigator, expand the Reference View Tables folder and double-click the table you want to view.

The Reference View Table appears in the main Machine Edition window.

2. Add reference addresses to the table as required.
Note: You cannot add reference addresses to a default RVT.
3. Format the table entries as desired.

3.4.14 Reports

Reports provide summaries and tales of information about your project. Most reports are displayed in the InfoViewer. The Reports tab of the FeedbackZone contains a list of all reports generated since the last Machine Edition project was opened. The following table shows the types of reports and logic printouts available in Logic Developer – PLC.

Table 2: Report Types Available

Address Use Report	Hardware Configuration Report	IL Block Report ^L
Application Structure	Initial Force States n Project Report	LD Block Report ^L
CAM Profile Report	Modbus Address Report	Local Logic Block Report ^L
EGD Reports	Variable Difference Report	Motion Block Report ^L
Forces in Controller Report	Variables Reports	Structured Text Blocks ^L

Note: An L indicates a logic printout.

3.4.15 To Generate Reports

In the Project tab of the Navigator, right-click a node and choose *Report* to generate a report on that node.

A report is automatically generated and displayed in the InfoViewer.

Note: To generate a Forces in Controller report for a target, you must be online to the target. Right-click the target and choose *Report*. In the *Select a Report* dialog box, select *Forces in Controller Report* and click *OK*.

3.4.16 To Redisplay a Previously Generated Report

1. In the Feedback Zone, double-click the Reports tab. A list of previously generated reports appears in the Feedback Zone.

2. In the list, click the report you want to view. The report appears in the InfoViewer. Many items in a report contain hyperlinks. Click a hyperlinked item to jump to that item in the project. For example, if a variable's name appears hyperlinked in a report, clicking it selects that variable in the Variables tab of the Navigator. Large reports are often separated into several pages. To view a different page of the report, scroll to the bottom of the report in the InfoViewer and click the number of the page you want to view.

3.4.17 To Print a Report Displayed in the InfoViewer

1. Generate the report you want to print, or redisplay a previously generated report.
2. When the report is displayed, right-click the InfoViewer window and choose *Print*.

3.4.18 To Print LD Blocks

1. In the Project tab of the Navigator, expand the Logic node.
2. Right-click the Program Blocks node and choose *Print LD Blocks*.
The Print dialog box appears.
3. Select the blocks to print.
 - a. To print all of the target's LD blocks in alphabetical order, select the *All* option.
 - b. To print only some of the target's LD blocks, select the *Selection* option, then select the check box in front of each block you want to print. To determine in which order to print the blocks, select them one at a time and click *Up* or *Down* until the selected blocks appear in the desired order.
4. Select options as required and click *OK*.

3.4.19 To Print ST Blocks

1. In the Project tab of the Navigator, expand the Logic node.
2. Right-click the Program Blocks node and choose *Report of ST Blocks*.
The Structured Text Blocks logic printout is automatically generated and displayed in the InfoViewer.

Section 4: Hardware Configuration

Logic Developer - PLC supports several PACSystems Controller families and various remote I/O interfaces with a variety of CPUs, racks, and modules for each. To operate, PACSystems Controller hardware must be configured with Logic Developer - PLC or some other Emerson software tool. The HWC component of Logic Developer - PLC provides a way to configure your target equipment. This chapter provides specifics on configuring Controller hardware for your operational needs. The first step in configuring Controller hardware is to select the Controller you want to configure. When creating a new project, you can use a project template containing a default Hardware Configuration, or you can create an empty project and configure it manually.

4.1 PACSystems RXi

The PACSystems RXi is an advanced, high performance, small footprint, PROFINET-dedicated Controller designed for distributed applications (process or discrete) in rugged environments.

The PACSystems RXi has the following characteristics:

- 2-port (shared MAC) Gigabit Ethernet I/O PROFINET network connection
- Additional 1xGB Ethernet port Built-in cable redundancy (MRP) delivering PROFINET IO cabling redundancy with no external switches
- Dual core COMExpress ICRXICTL000 CPU with 2 GB RAM and up to 64 GB internal flash data storage
- Support for most LD, FBD, and ST instructions supported on other PACSystems platforms
- High speed Interconnect Bus enabling combinations of control and PAC (or other Microsoft® Windows® or Linux applications)
- Internal industrial grade SSD drive providing local long-term data retention
- Energy Pak supplying power to write data to NV RAM during power failures
- USB and SD interfaces enabling program loading, serial communications, and data storage via standard devices ?
- (Optional.) Intelligent Display Module, a configuration and maintenance touchscreen right on the Controller

4.2 PACSystems RX3i

The PACSystems RX3i consists of a main rack and up to seven expansion racks. Six types of Series 90-30 expansion racks are supported; they have IC693CHSnnn catalog numbers and are colored black, in keeping with the Series 90-30 color scheme. Two expansion racks have IC694CHSnnn catalog numbers. These are identical to the IC693CHSnnn expansion racks of the same numbers except that they are colored blue, in keeping with the PACSystemsRX3i color scheme.

The main rack and all expansion racks support most Series 90-30 modules; these have IC693NNNnnn catalog numbers and are colored black, in keeping with the Series 90-30 color scheme. Almost all of the IC693NNNnnn modules supported by PACSystems RX3i also come as IC694NNNnnn modules that have identical functionality. The only difference is that the IC694NNNnnn modules are colored blue, in keeping with the PACSystems RX3i color scheme. For example, the IC693DSM314 and IC694DSM314 are functionally identical and both can be used interchangeably in a Series 90-30 rack system or a PACSystems RX3i rack system. The only hardware difference is their color.

When you configure a PACSystems RX3i rack system, you can select IC693NNNnnn or IC694NNNnnn racks or modules. When you configure a Series 90-30 rack system, however, you

can select only IC693NNNnnn racks or modules. Even if you are physically using IC694NNNnnn racks or modules, selecting IC693NNNnnn racks or modules in Logic Developer - PLC inadequate.

The PACSystems RX3i default main rack, IC695CHS012, has 13 slots: Slots 0 through 12. The alternate main rack, IC695CHS016, contains 17 slots: Slots 0 through 16.

On either main rack, slot 0 is reserved for a power supply or the CPU, but the power supply or CPU does not have to be in slot 0. The last slot is reserved for the Serial Bus Transmitter module (IC695LRE001). The slot just before the last slot can contain any single-width module native to PACSystems RX3i (IC695...) except the Serial Bus Transmitter. Any other slot can contain any single-width module native to PACSystems RX3i (IC695...) except the Serial Bus Transmitter, and if the next slot is empty, they can contain any supported double-width module.

The IC695PSA040 and IC695PSA140 power supplies and the CPU are two-slot modules. The IC695PSD040 and IC695PSD140 power supplies are one-slot modules. PACSystems RX3i supports many Series 90-30 modules: each one occupies a single slot.

4.2.1 Configuring PACSystems

When you create a target with a PACSystems RX3i rack system, the default consists of the main rack, with a power supply in slots 0 and 1, and a CPU in slots 2 and 3. You can replace the default power supply. You can replace the CPU with itself to update it to the latest catalog version, that is, the latest set of parameters supported for the CPU in Logic Developer - PLC. You can move the power supply and the CPU to any empty slot in the target with an adjacent empty slot; however, the second adjacent slot cannot be the last slot. You can add up to seven expansion racks, and on each of these, you can add IC693NNNnnn modules and the IC694NNNnnn that are supported by Series 90-30.

Note: On PACSystems RX3i targets, only the main rack is added by default. On Series 90-70 and Series 90-30 Controllers, seven expansion racks are added by default to the HWC. You do not need to add them.

4.2.2 Configuring Controller Hardware

The following configuration procedures focus on the PACSystems RX3i Procedures for the other controller families supported by Logic Developer – PLC are nearly identical, when applicable.

To Replace a Power Supply (Not Applicable for PACSystems RXi)

1. In the Project tab of the Navigator, right-click the power supply slot and choose *Replace Module*.

A list of available power supplies appears.

Note: For non-PACSystems families, there is no Slot 0. Right-click the PWR slot instead.

2. Select the power supply you have installed in your rack and click *OK*.

The default CPU specified in the project template for a PACSystems RX3i is the IC695CPU320.

4.2.3 To Replace a CPU (Not Applicable for PACSystems RXi)

1. In the Project tab of the Navigator, expand the Hardware Configuration. All racks revealed.
2. Expand the main rack.
3. Right-click the CPU slot and choose *Replace Module*.
4. *Note:* An Emerson controller supports only one CPU; it can be replaced with itself to update it to the latest catalog version, that is, the latest set of parameters supported for the CPU in Logic Developer -PLC. The Catalog dialog box appears.
5. Click *OK*. A dialog box appears asking if you want to retain the setting a from the existing CPU.
6. Click *Yes* or *No*. The target is configured with the selected CPU.

4.2.4 To Configure a CPU

1. In the Project tab of the Navigator, right-click a slot containing a CPU and choose *Configure*.
The Parameter editor displays all configurable settings for the CPU.
2. Modify the settings as required. For information on any parameter of a PACSystems CPU, select the parameter.
Help topics for PACSystems CPU parameters appear in the Companion. To open the Companion, press SHIFT+F11.

4.2.5 To Add an Expansion Rack (PACSystems RX3i Only)

1. In the Project tab of the Navigator, right-click the Hardware Configuration and choose *Add Rack*
2. Select a rack and click *OK*.

4.2.6 To Replace a Rack (Not Applicable for PACSystems RXi)

1. In the Project tab of the *Navigator*, right-click a rack and choose *Replace Rack*.
The Catalog dialog box appears listing available rack types.
2. Select a rack and click *OK*.

4.2.7 To Add an Ethernet module (PACSystems RX3i Only)

1. In the Project tab of the *Navigator*, expand the *Hardware Configuration* and then *Rack 0* (the main rack).
2. Double-click the slot you want to add an Ethernet module to, or right-click it and choose *Add Module*.
3. In the Communications tab of the Catalog, select the Ethernet Module and click *OK*.
The Ethernet module is added to the slot.
4. Double-click the Ethernet module.
The Parameter editor appears.
5. Configure the Ethernet module's parameters as needed.

4.2.8 To Configure the Ethernet Daughterboard (RX3i Only)

1. In the Project tab of the Navigator, expand the Hardware Configuration, then Rack 0 (the main rack), and then the CPU.
2. Double-click *Ethernet*. The Parameter editor appears.
3. Configure the Ethernet daughterboard's parameters as needed.

4.2.9 To Move a Module (Not Applicable for PACSystems RXi)

In the project tab of the Navigator, expand the Hardware Configuration and then the rack that contains the module.

Do one of the following:

- To move the module from one target to another: press SHIFT while dragging and dropping the module onto an appropriate empty slot; or
- To move the module within a target: drag and drop the module onto an appropriate empty slot without pressing any keys. The module is removed from the original slot and inserted into the empty slot.
-
- You can drag a double-width module to a main rack slot only if both the slot and the slot after it are empty. Exceptions: The last slot on a PACSystems RX3i main rack can contain only an IC695LRE001.
-
- Only PACSystems RX3i supports moving a power supply or CPU module.
-
- In the Project tab of the Navigator, expand the Hardware Configuration and then the rack that contains the module.

Do one of the following:

- To copy the module from one target to another: drag and drop the module onto an appropriate empty slot without pressing any keys.

OR

- To copy the module within a target (not applicable for PACSystems RXi): press CTRL while dragging and dropping onto an appropriate empty slot.

You cannot copy modules between different types of Controller targets, for example from a Series 90-30 to a PACSystems RX7i. Exception: You can copy a *PROFINET* Controller from a PACSystems RX3i to a PACSystems RXi or vice versa.

A copy of the module is inserted into the empty slot.

Notes:

- Only the PROFINET Controller can be copied to or from a PACSystems RXi.
- You can drag and drop a double-width module to a main rack slot only if both the slot and the slot after it are empty. Exceptions: The last slot on a PACSystems RX7i can contain a double-width module or one single-width module. The last slot on a PACSystems RX3i can contain only an IC695LRE001.
- PACSystems RX3i can support copying a power supply.

4.3 I/O Variables

An I/O variable is a variable mapped to a terminal in the Hardware Configuration of a PACSystems. A terminal, for example, can be one of the following:

1. A physical I/O discrete or analog point on a PACSystems module or on a Genius device
2. A discrete or analog status returned from a PACSystems module
3. Genius global data

Memory required to support I/O variables counts against your user memory. When you configure the PACSystems CPU, select the Memory tab and set the I/O Discrete (# of Bits) and I/O Non-Discrete (# of Words) parameters to configure the space available for I/O variables.

4.3.1 To Enable I/O Variables for a Module

1. In the Project tab of the Navigator, expand the target that contains the module.
2. Expand the Hardware Configuration and expand the rack that contains the module.
3. Right-click the module and choose Properties. The Inspector displays the module's properties.
4. In the Inspector, set the Variable Mode property to *True*.

If the Variable Mode property is read-only and set to *False*, the module does not support I/O variables.

5. In the dialog box that appears, click *Yes*.
6. If you enable I/O variables for a Genius bus controller, I/O variables are enabled for all the Genius devices on the Genius bus.

Note: When you enable I/O variables for a module, the Hardware Configuration and logic become coupled. This means that they must be downloaded, uploaded, or cleared together. You cannot change or add I/O variables in Run Mode Store.

4.3.2 To Map a Variable to a Terminal in the Terminals Tab of a Module or Genius Device

1. In the Project tab of the Navigator, expand the target that contains the module or Genius device.
2. Expand the Hardware Configuration and expand the rack that contains the module or device.
3. Double-click the module or expand the Genius Bus Controller (GBC) and double-click the Genius device.

The Parameter editor displays the Terminals tab of the module or device.

4. In the Terminals tab, right-click a terminal node that has no I/O variable mapped to it and choose *Map Variable*.

The Variables smart list appears.

5. In the smart list, do one of the following:
 - Enter a name that is not used elsewhere in the PACSystems target. A new I/O variable by that name is created and mapped to the terminal. Its Ref Address property is set to an I/O variable address. If the terminal is discrete, the new I/O variable's data type is set to BOOL. If the terminal is analog, the data type is set to INT, but you can change it to another 16-bit data type: UINT or WORD.

OR

- Enter the name of an existing CPU-mapped variable, symbolic variable, or alias variable. (There are some mapping limitations.) The variable becomes an I/O variable mapped to the terminal and its Ref Address property is changed accordingly.

4.3.3 Hot Redundancy Systems

Hot redundancy systems are supported only on PACSystems RX7i, PACSystems RX3i, and Series 90-70 Controllers. In redundancy systems, two units are set up and configured identically. If one unit fails or is taken offline, the other unit assumes responsibility without interrupting operation of the entire system. There are three types of hot redundancy systems:

- Basic CPU Redundancy
- Genius Redundancy
- CPU Redundancy Over Genius

To implement basic hot CPU redundancy, a single Logic Developer - PLC target is associated with two physical Controllers, a Primary and a Secondary. Both Controllers share the same logic, but each has its own Hardware Configuration (HWC): Primary or Secondary. The selected HWC is the HWC that you can go online with, download to, upload from, and so on.

Three types of basic CPU redundancy are available. These redundancy systems can be used in combination with Genius redundancy schemes.

- Single Bus with Preferred Master (Series 90-70 CPUs CGR772 and CGR935 only): uses a single Genius bus with one or more bus Controllers in each Controller. The primary unit is always chosen as the active unit when the units initially synchronize.
- Single Bus with Floating Master (PACSystems RX7i, PACSystems RX3i, and Series 90-70): uses a single Genius bus with one or more bus Controllers in each Controller. No switchover occurs on initial synchronization to make the primary unit the active unit.
-
- Dual Bus with Floating Master (RX7i, RX3i, and Series 90-70): uses dual busses with one or more bus Controllers in each Controller. No switchover occurs on initial synchronization. Bus Switching Modules (BSMs) are required in accordance with configuration of a dual bus network. This option provides redundancy of both the CPU and the Genius bus.

4.3.3.1 Genius Redundancy

A Genius redundancy system contains duplicate components that are configured to keep the Genius system operating properly even if one of the duplicate components fails or is taken out of service. Genius redundancy systems can be used in combination with PACSystems RX7i, PACSystems RX3i, and Series 90-70 CPU redundancy systems.

- You can configure five types of Genius redundancy systems:
- Genius dual bus redundancy (paired GBC internal)
- Genius dual bus redundancy (paired GBC external)
- Genius dual GBC redundancy (paired GBC internal)
- Genius dual GBC redundancy (paired GBC external)
- Genius dual bus & dual GBC redundancy

4.3.3.2 Hot CPU Redundancy Over Genius

A CPU Redundancy Over Genius system contains duplicate components that are configured to keep the system operating properly if one of the duplicate components fails or is taken out of service

Five types of CPU Redundancy Over Genius systems can be built upon the various types of redundancy Genius systems combined with the basic CPU Redundancy schemes.

- CPU redundancy (GHS) using Genius dual GBC redundancy (paired *GBC external*): Single bus with preferred master (Series 90-70 only)
-
- CPU redundancy (HSB/GDB) using Genius dual GBC redundancy (*paired GBC external*): Single bus with floating master
-
- CPU redundancy (HSB/GDB) using Genius dual bus redundancy (paired *GBC external*): Single bus with floating master
-
- CPU redundancy (HSB/GDB) using Genius dual bus and dual GBC *redundancy*: Dual bus with floating master
- CPU redundancy (HSB/GDB) using a mixed Genius redundancy scheme

4.3.3.3 Configuring Hot Redundancy Systems

The procedures below illustrate how to configure hot redundancy systems. For the procedures on configuring specific hot redundancy systems, see online help.

4.3.3.3.1 To Setup the Primary Hardware Configuration for Hot CPU Redundancy

1. In the Project tab of the Navigator, expand the target for which you want to set up CPU redundancy.

2. Right-click Hardware Configuration, point to *Redundancy*, and choose *Wizard*.

The Redundancy Wizards dialog box appears, with the *Set Up Primary Hardware Configuration for CPU Redundancy* option selected by default.

3. Click *Next* and follow the wizard to the end.

The wizard adds a redundancy CPU and other redundancy modules to the configuration.

The target property, *Dual HWC*, is now available in the Inspector and is set to False by default.

To add Genius Bus Controllers (GBC) to your system (PACSystems RX3i)

1. In the Project tab of the Navigator, expand the target for which you want to set up Genius redundancy.

2. Right-click Hardware Configuration, point to Redundancy, and choose Wizard. The Redundancy Wizards dialog box appears.

3. Select Add GBCs for Genius Redundancy.

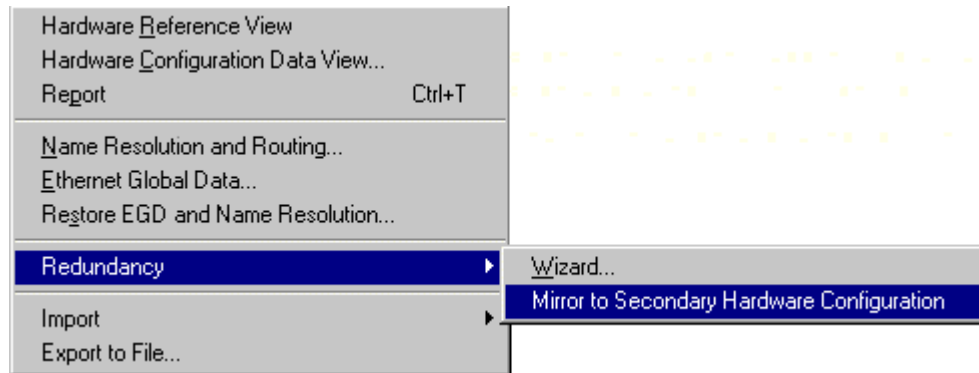
4. Click *Next* and follow the wizard to the end. The wizard enables you to select a Genius redundancy system and the location of the GBC modules. You can run this wizard multiple times to configure additional pairs of redundancy busses in the same system.

Note: When you add a Genius Bus Controller (GBC) to a PACSystems, PACSystems RX3i, or Series 90-70 rack, a new Genius bus network is automatically created and associated with that slot and GBC module. Up to 31 Genius I/O devices (blocks) can be connected to a GBC through its Genius bus.

4.3.3.3.2 To Configure the Secondary Hardware Configuration (RX3i and Series 90-70)

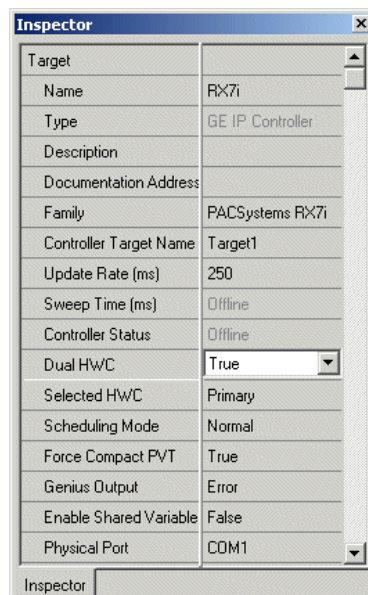
1. In the Project tab of the Navigator, select the Hardware Configuration. Right-click Hardware Configuration, point to Redundancy, and choose Mirror to Secondary Hardware Configuration (Figure 6).

Figure 6: Mirror to Secondary Configuration



A secondary rack system that is a copy of the primary rack system is generated. The target now displays two Hardware Configurations (HWCs), one labeled [Primary] and the other, [Secondary]. The Primary HWC is bolded because it is currently selected. The target property *Dual HWC* is now set to True. To select which Controller to interact with, set the Selected HWC property, just below the Dual HWC property (Figure 7)

Figure 7: Hardware Configuration



Note: You can mirror as many times as necessary to synchronize the two HWCs after modifying the primary HWC. Each time you mirror the primary HWC, the secondary HWC is updated to reflect those changes.

4.3.3.3.3 **DSM324i and Motion Mate DSM31 4 Motion Modules**

The Series 90-30 Controller family supports various I/O modules (discrete input, discrete output, discrete mixed, analog input, analog output, and analog mixed), communication modules, intelligent modules, bus Controllers, and motion modules. You configure Series 90-30 hardware as described in the section “Configuring Controller Hardware.”

PACSystems RX3i and Series 90-30 support some Motion modules that no other Emerson controller family supports: the DSM324i and the Motion Mate DSM314. Both are high performance, easy-to-use multi-axis motion control modules. Compatible with Controller logic solving and communications functions, the DSM314 supports the following servo types:

- Digital – Emerson digital servo amplifiers and motors.
- Analog – Emerson SL Series analog servos and third-party servos.

The DSM324i supports only the Digital servo types.

Both the DSM324i and the Motion Mate DSM314 modules have four axes that can be individually configured in Standard or Follower mode.

In order to perform motion programming in Logic Developer - PLC, you must program a DSM324i or a Motion Mate DSM314 module.

4.3.3.3.4 **To Add a DSM324i or Motion Mate DSM314 module**

1. With a rack of the Hardware Configuration node expanded, right-click the empty slot you want to add a DSM324i or a Motion Mate DSM314 module to and choose Add Module.

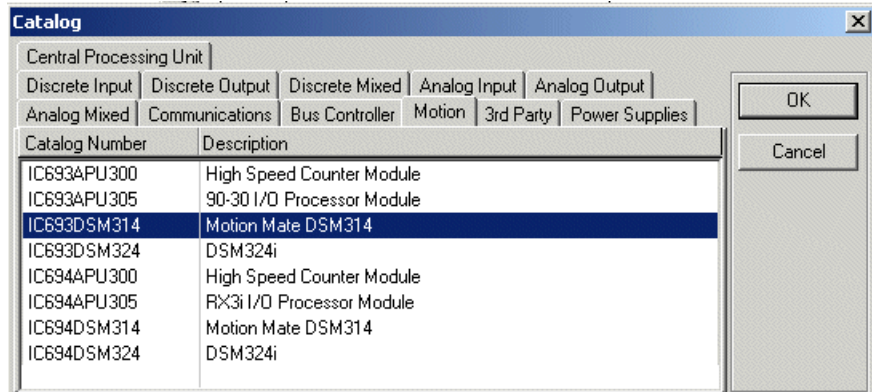
The Catalog dialog box appears.

2. In the Catalog dialog box, click the Motion tab.

A selection of motion modules appears in the Catalog dialog box.3. From the list, select DSM324i or Motion Mate DSM314.

The following picture displays some of the selections available for a PACSystems RX3i (Figure 8).

Figure 8: Catalog



3. Click OK. AS per the example illustration, an IC693DSM314 is added to the Hardware Configuration of your project.

4.3.3.3.5 To Configure a DSM32i or a Motion Mate DSM314

1. In the Project tab of the Navigator, double-click the slot containing a DSM324i or Motion Mate DSM314. The Parameter editor appears.
2. Configure the DSM324i or Motion Mate DSM314 by using the Parameter editor.

4.3.3.4 Remote I/O

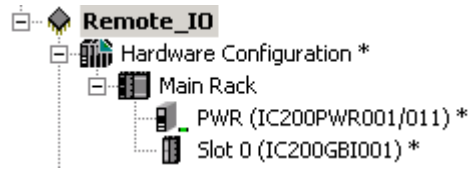
4.3.3.4.1 VersaMax Remote I/O

VersaMax remote I/O consists of a Network Interface Unit (NIU) (Ethernet, Genius, or Profibus), and one or more I/O modules. You can use Logic Developer - PLC to configure this hardware. Once configured, the remote I/O can be controlled by a VersaMax Controller or a PC Controller. The differentiating factor between a VersaMax remote I/O and a Controller is that a remote I/O is simply an input/output device with a communication interface.

Unlike a Controller, a remote I/O has no CPU.

With Logic Developer - PLC, you can add racks, configure the power supply, and configure modules in the VersaMax Remote I/O Hardware Configuration. Remote I/O targets are generally added to a project when you create a project from a template. You can also add a Remote I/O target to an existing project (Figure 9).

Figure 9: Remote I/O



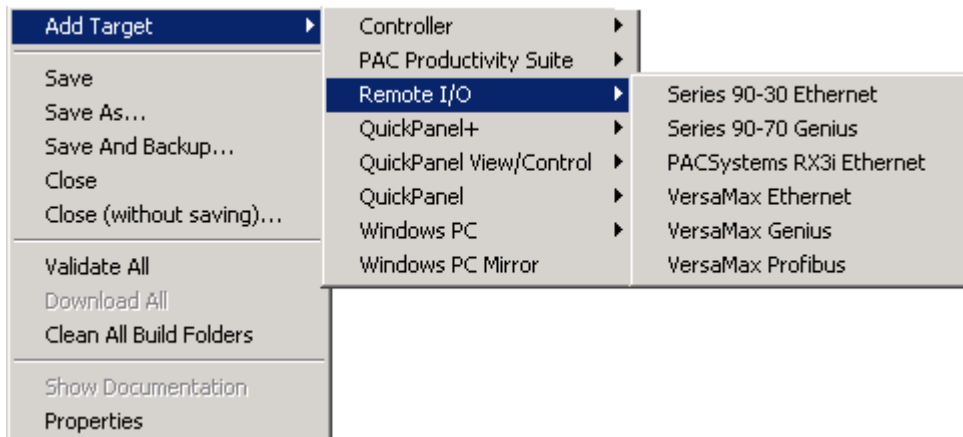
4.3.3.5 To Create a Project Containing a Remote I/O Target from a Template

1. From the File menu, choose New Project. The New Project dialog box appears.
2. From the Project Template list, choose the Remote I/O type you want to add to your project.
3. Enter a descriptive Project Name.
4. Click OK. A new Remote I/O project is started.

4.3.3.6 To Add a Remote I/O target to an Existing Project

1. In the Project tab of the Navigator, right-click the Project node.
2. Point to Add Target, then to Remote I/O, and choose a remote I/O (Figure 10).

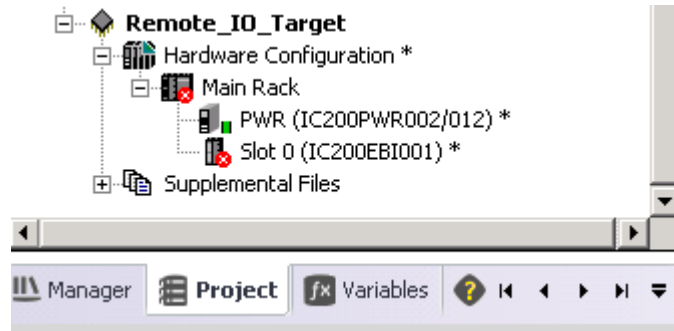
Figure 10: Adding a Remote I/O Target



4.3.3.7 To Replace the Power Supply in your Remote I/O Configuration

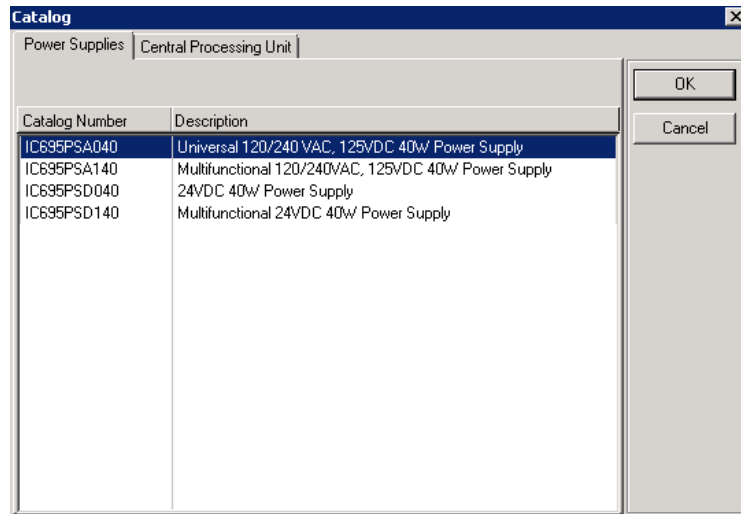
1. In the Project tab of the Navigator, expand the Hardware Configuration of the remote I/O. The Navigator displays the following (Figure 11):

Figure 11: Remote I/O Configuration



2. Right-click the PWR slot and choose Replace Module. The Module Catalog dialog box appears (Figure 12):

Figure 12: Module Catalog

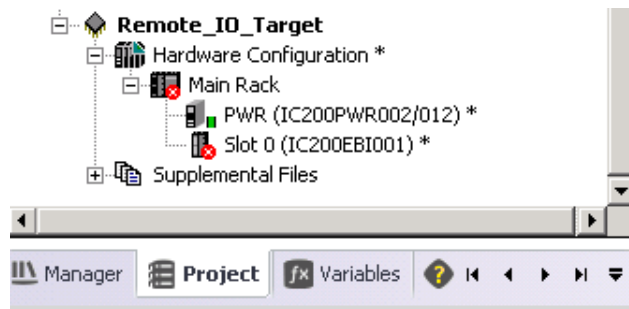


3. From the list, select the power supply that you want to configure for your system.
4. Click OK.

4.3.3.7.1 To Add a New Carrier/Base to Your VersaMax Remote I/O

1. In the Project tab of the Navigator, select a remote I/O target.
2. Expand the Hardware Configuration node and the main rack. The navigator displays the following (Figure 13):

Figure 13: VersaMax Remote

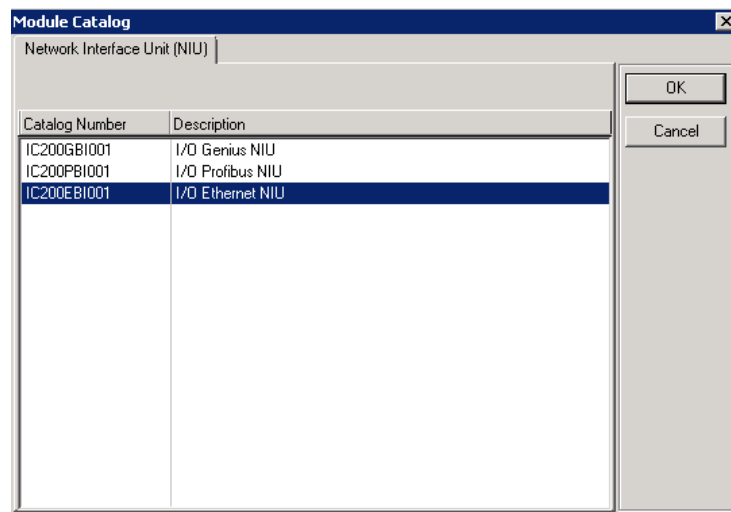


3. Right-click the Slot 0 node and choose *Add Carrier/Base*. The Module Catalog dialog box appears.
4. Select the carrier/base that you want to add to the remote I/O target.
Note: You can add a maximum of eight carrier modules to each VersaMax rack.
5. Click OK.

4.3.3.7.2 To Add a Module to a Carrier/Base

1. In the Project tab of the Navigator, expand the Remote I/O target you want to add a base to.
2. Double-click an empty carrier/base. The Module Catalog appears (Figure 14):

Figure 14: Module Catalog



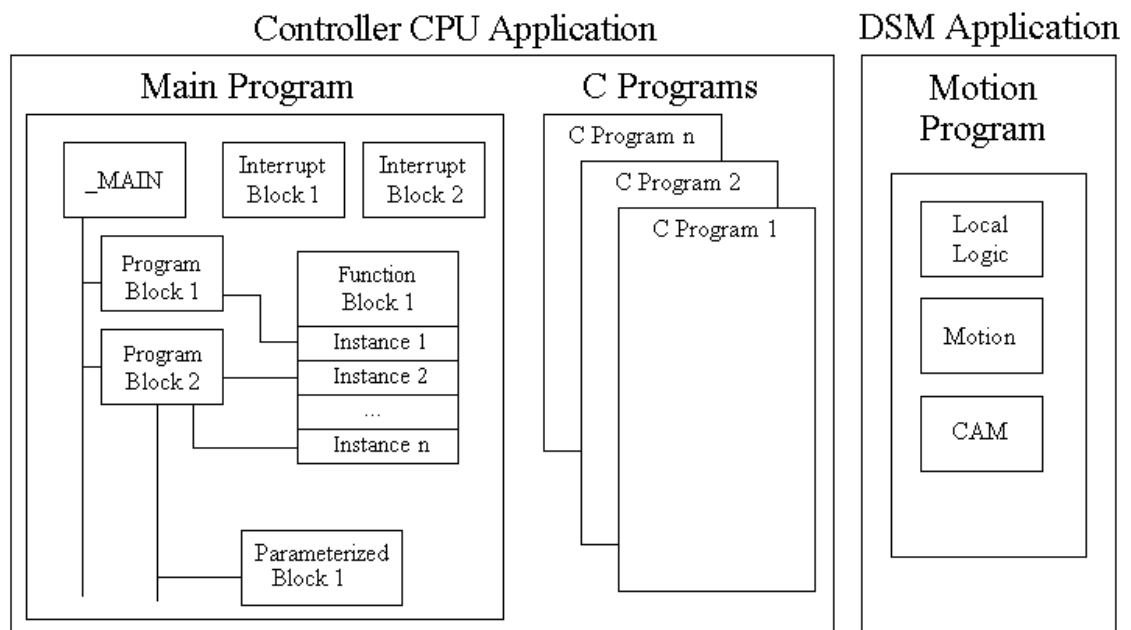
3. From the Module Catalog, select the module that you want to install.
4. Click *OK*.

Section 5: Logic Programs and Blocks

5.1 Program Types

For most Emerson controllers, all the logic that is downloaded to a Controller constitutes one program. However, there are cases when distinctions need to be made. Therefore, we can speak of a main program, a C program, and a Motion program (Figure 15).

Figure 15: Program Types



Each C program is a named section of executable code. The main program and the Motion program contain blocks; that is, named sections of executable code that can be written in various languages.

The Project tab of the Navigator contains all of the programs:

- The main program and its blocks reside in the Program Blocks folder. You can create user-defined subfolders to further organize these blocks.
- C programs reside in the Logic folder.
- The Motion program resides in the Motion Program folder.

5.1.1 Main Program

Most PACSystems Controllers support only a main program. The main program consists of a block of logic named `_MAIN` and optionally, one or more of the following kinds of blocks of logic, found under the Program Blocks folder in the Logic folder of an PACSystems controller target:

Called blocks developed in various languages:

- Ladder Diagram (LD) Blocks
- Structured Text (ST) Blocks
- Function Block Diagram (FBD) Blocks
- Instruction List (IL) Blocks
- C Blocks
- Called parameterized blocks (LD, ST, FBD, and/or C).
- Interrupt blocks (LD, ST, FBD, C, and/or IL) scheduled to execute at certain time intervals or when specific memory registers reach certain values.
- User-defined function blocks (UDFBs) in LD, ST, and FBD, of which independent instances with their own data structures and local memory can be used in logic.

Specialty function blocks, like HART utilities. Like UDFBs, you can create independent instances of specialty functions blocks with their own instance data (a structure variable). Unlike UDFBs, you cannot edit the logic of specialty function blocks because they are read-only and provided to you by Emerson.

- All PACSystems Controllers support LD blocks, but only certain Controller families or models support one or more of the other kinds of blocks.
- The `_MAIN` block can be in various languages, depending on the Controller family or model.

You download the main program to the Controller as logic; that is, after you initiate a download, a dialog box presents you with various Download to Controller options and you select the Logic option. If the Logic option is not present, select the Program option.

5.1.2 C Programs

Series 90-70 CPUs firmware version 6.00 and later support either the main program, or various C programs, or a combination of a main program and C programs. You download these various programs to the Controller together as logic; that is, after you initiate a download, a dialog box presents you with three Download to RAM options and you select the Logic option.

If there is no `_MAIN` block, then there is no main program. In this case, you would have only one or more C programs and possibly various standalone interrupt blocks (LD or C). These standalone interrupt blocks do not constitute a main program if there is no `_MAIN` block.

A C program can be named `_MAIN` only if the target contains no LD blocks.

A C program is not the same as a C block.

5.1.3 Motion Program

In addition to the main program, Series 90-30 CPUs firmware version 10.00 and later, as well as PACSystems RX3i CPUs firmware version 2.80 and later, support a Motion program for use on a DSM324i or Motion Mate DSM314 module. The Motion program consists of:

- CAM profiles
- CAM blocks
- Local Logic block
- Motion blocks

You download the Motion program to the Controller as part of its Hardware Configuration (HWC); that is, after you initiate a download, a dialog box presents you with various Download to RAM options and you select the Hardware Configuration option. If you want to download the main program at the same time, you also select the Logic option. If neither option is present, select the Program option to download both the Hardware Configuration and the Logic.

5.1.4 Number of Blocks in the Main Program

Table 3: Number of Blocks in the Main Program

CPU Type	Number of Blocks	Maximum Number of Blocks
PACSystems ¹	767 subroutine blocks plus one _MAIN block	768
VersaMax and VersaMax Micro	64 subroutine blocks plus one _MAIN block	65
VersaMax Nano	8 subroutine blocks plus one _MAIN block	9

5.1.5 Scheduling Programs

On Emerson Controllers that support only one program (the main program), the program as a whole cannot be scheduled. Every scan, logic execution begins with the _MAIN block. You can schedule individual interrupt blocks of logic but not the _MAIN block.

When you execute both a main program and a Motion program on a PACSystemsRX3i with a firmware version of 2.80 or later, you cannot schedule the programs because they are executed concurrently in two different locations.

¹ PACSystems controllers can only support up to 512 subroutine blocks depending on the firmware version.

- In the CPU, every scan, logic execution begins with the `_MAIN` block of the main program.
- In the DSM324i or Motion Mate DSM314 module, the Motion program executes independently of CPU scan times.
- You can schedule individual interrupt blocks of logic in the main program (except the `_MAIN` block).

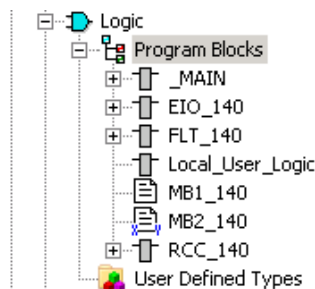
Series 90-70 CPUs firmware version 6.00 and later support scheduling programs. On targets that contain such CPUs, you can schedule any program, even the main program. That is, logic execution of any scan does not have to begin with the `_MAIN` block of the main program: you can schedule any C program to begin the logic execution part of the scan. Within the main program, you can further schedule individual interrupt blocks of logic (except the `_MAIN` block). If you have no main program, that is, if your logic has no `_MAIN` block, you can still have stand-alone interrupt blocks that you can schedule individually.

5.1.6 To Create a User-Defined Folder

In the Project tab of the Navigator, expand the target with the main program you want to organize and then expand the Logic folder.

1. Right-click the Program Blocks folder, point to *New*, and choose *Folder*.
A new user-defined folder appears with a default name.
2. Optionally enter a new folder name, which must be unique among the folders directly under the parent folder.

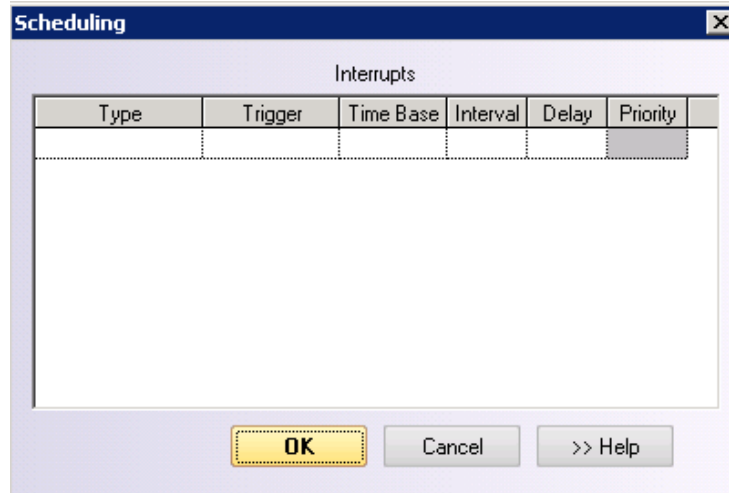
Figure 16: Logic Folder



5.1.7 To Schedule the Execution of a Block of Logic

1. In the Project tab of the Navigator, right-click an existing LD, FBD, ST, C, or IL block, and choose *Properties*.
The block's properties appear in the Inspector.
2. In the Inspector, click the ellipsis button (...) in the Scheduling property (Figure 17).

Figure 17: Scheduling Property



3. Configure scheduling by entering values in each of the fields.

5.1.8 To Control Access to a Block

1. Right-click an LD, FBD, ST, or IL block and choose *Properties*
2. In the Inspector, expand the Lock Settings property.
3. In the Lock Type property, choose a setting from the list.
4. In the Password Property, enter a password.

5.1.9 To Search/Replace in One Block

1. Double-click an LD, FBD, ST, or IL block to open it.
2. From the *Search* menu, choose *Find* or *Replace*.
A dialog box appears.
3. In the *Find what* or in the *Text to find* field, enter the text to find.
4. (Only if you want to replace text.) In the *Replace with* field or in the *New text* field, enter the text to replace the found text with.
5. Select or clear each search/replace option, as required.
6. Click *Find*, *Find next*, *Replace*, *Replace All*, *Close*, or *Cancel*, as required.

Note: With Logic Developer - PLC, you can search for some text in an entire target or portions thereof, which you can specify. Various options are available to narrow a search.

5.1.10 Indirect References

The LD, FBD, and ST editors support indirect references. An indirect reference treats the value of a variable assigned to an instruction operand as a pointer to other data, rather than as actual data. Indirect references are sometimes referred to as relative pointers. Indirect references can be used only as follows:

- With PACSystems CPUs and Series 90-70 CPUs.
- In the LD, FBD, and ST editors.
- With %R, %AI, %AQ, %P, %L, and %W memory areas.
- On instructions that support indirect references.

Notes for PACSystems

- The index for an indirect reference to %W is a 32-bit DWORD value.
- Indirect references cannot be used to address bits in 16-bit memory.
- Indirect references are not supported on symbolic variables or I/O variables.

Possible Uses

- To perform the same operation to many registers.
- To avoid repetitious logic within the application program.
- In loop situations where each register is incremented by a constant or by a value specified until a maximum is reached.

5.1.11 To Assign an Indirect Reference

Where a variable operand is expected, type the @ symbol, followed by a valid reference address or variable name. The LD, FBD, or ST editor converts a reference address to a variable name.

The LD, FBD, or ST editor converts a reference address to a variable name or the editor substitutes the name of a variable already mapped to the address.

Notes:

- The actual variable name does not contain the @ symbol. The @ symbol, when used at the start of a reference address or variable name, merely indicates that the address or variable must be treated as an indirect reference.
- The @ symbol can be thought of as an operator with the lowest precedence. In other words, MyArray[2,5] becomes a reference address that is used for the indirect reference. Another way to think of this is @(MyArray[2,5]) as opposed to (@MyArray)[2,5].

The following are valid indirect references:

- @R0001 (a variable name automatically created from a reference address)
- @MyArray[2, 5] (an array reference)
- @\$MainSwitch (a universal variable)
- @MyTimer.PV (a structure element)

5.2 LD Editor

The Ladder Diagram (LD) editor is used to create programs with the Ladder Diagram programming language. LD logic graphically represents the programmed actions performed by a Controller as it executes.

The LD editor is cell-based with rungs constructed of horizontal sequences of instructions that are wired together. A given instruction and its operands can occupy one or more cells.

You can work with the LD editor while offline to edit a disk copy of a project, or while online to monitor the execution of the logic while you fine tune the project.

You can customize the appearance and behavior of the LD editor by setting options.

An LD block is a named section of LD Logic that is compiled and downloaded to the Controller represented by the associated target.

5.2.1 To Customize the LD Editor

1. In the Options tab of the Navigator, expand the Editors folder and then the Ladder folder.
2. Right-click a page (Confirmations, Editing, Font and Colors, or View), and choose *Properties*.
The configurable options appear as properties in the Inspector
3. In the Inspector, adjust settings as required.

5.2.2 To Create an LD Block

1. In the Project tab of the Navigator, right-click Program Blocks, point to *New*, and then choose LD Block.
A new LD block with a default name is created.
2. Rename the block as desired.

5.2.3 To Open an LD Block for Editing

- In the Project tab of the Navigator, double-click an LD block.

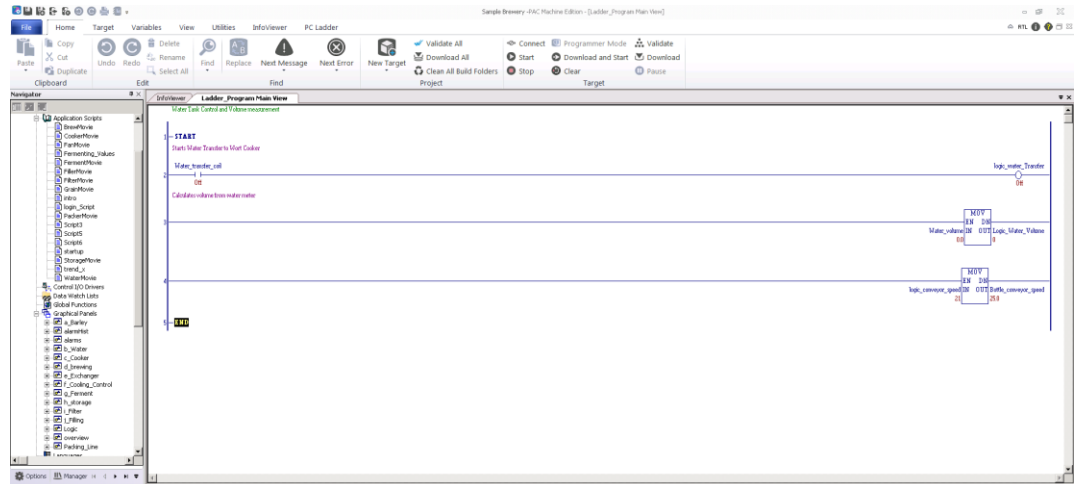
The block opens in the LD editor.

Note: You can have multiple blocks open for editing. To navigate to another open LD block, click the tab that displays its name at the top of the editor window.

5.2.4 Working with the LD Editor Offline

When you are offline, there is no live communication between the LD editor and the target. Most logic development is done while offline. The following diagram illustrates the ladder diagram interface in Machine Edition (Figure 18).

Figure 18: LD Editor Offline



5.2.4.1 To Insert an Instruction

1. In the LD Editor, right-click an empty cell and choose *Place Instruction*.

A smart list appears listing all available instruction mnemonics.

2. Choose an instruction mnemonic from the list and press enter.

Note: If you attempt to place an instruction that has inputs into the first column. It is pushed to the second column and a horizontal wire is inserted in the first column.

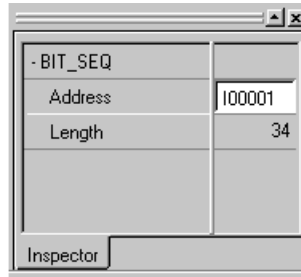
5.2.4.2 To Assign Instance Data to a Built-in Function Block Instance and Assign a Length to an Instruction

There are two methods:

First Method

1. In the LD editor, insert a built-in function block or insert an instruction that requires a Length.
In the following example, a built-in function block instance has a length. Most built-in function blocks do not require their instances to have a length.
2. Right-click the function block instance or instruction and choose *Properties*. The inspector displays the instance's or instruction's properties (Figure 19).

Figure 19: Properties



3. In the Address property, enter a variable name or a reference address to specify the start of a memory block used for the instance data of the function block instance.

Note: If you type a reference address, it is converted to a variable name automatically.

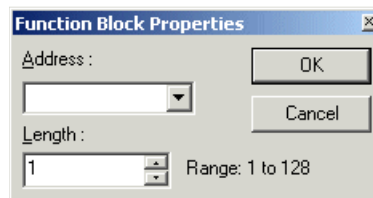
In the Length property, enter the number of bits or 16-bit registers on which the function block instance or instruction is to operate.

Second Method

1. In the LD editor, insert a built-in function block or insert an instruction that requires a Length.
2. Double-click the function block instance or instruction or select it and press ENTER.

When you double-click a function block instance, the Function Block Properties dialog box appears (Figure 20). When you double-click an instruction, the Function Properties dialog box appears, in which the Address property is unavailable.

Figure 20: Function Block Properties



3. In the Address property, enter a variable name or a reference address to specify the start of a memory block used for the instance data of the function block instance.

Note: If you type a reference address, it is converted to a variable name automatically.
4. In the Length property, enter the number of bit or 16-bit registers on which the function block instance or instruction is to operate.

5.2.5 To Assign Variables to Instruction Operands

1. In the LD editor double-click beside an operand of an instruction, or click there and press enter. A small list appears.

2. Type, or choose from the list, a variable name or reference address. If you enter a reference address, a variable name is automatically substituted.

5.2.6 To Check (Validate) a Single LD Block

1. In the Project tab of the Navigator, expand the target that contains the block to check, and then expand the Logic folder.
2. Expand the Program Blocks folder and then, if the block resides in a user defined folder, expand the user-defined folder.
3. Right-click the block and choose *Check Block*.

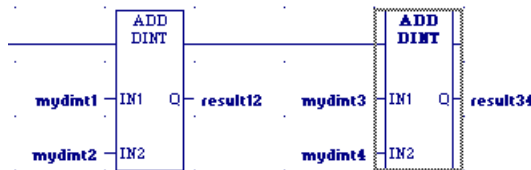
Machine Edition checks the block for errors. Any messages or errors are displayed in the Build tab of the Feedback Zone.

5.2.7 Editing Logic as Text

You can enter LD logic into the LD editor by starting to type right into a cell. A smart list appears and you can continue to type your instructions and operands.

When you press **ENTER**, the logic you typed appears in customary graphic form in the LD editor. For example, in the first cell of a new rung, if you type “AD MyDint1 MyDint2 Result12;AD MyDint3 MyDint4 Result34“, the following logic appears in the LD editor (Figure 21):

Figure 21: LD Editor



A horizontal wire was inserted in the first cell to make room for the ADD_DINT instruction’s operands and the operands were inserted in their proper places. In the keyboard entry, “AD” stood for “ADD_DINT“, because ADD_DINT is the first available instruction that begins with “AD“ in the smart list. A semicolon separates the ADD_DINT instructions. The required horizontal wires are supplied automatically.

A powerful extension of typing logic into the LD editor is the ability to write LD logic in any text editor and copy and paste it into the LD editor, or to copy and paste logic from the LD editor to a text editor. You can copy an entire block of LD logic from the Navigator to any text editor, or copy sections of LD logic as text from the LD editor to any text editor. You can then edit the logic in the text editor and copy the edited logic back into the LD editor.

When LD logic is copied as text, a verbose format is used to make it easier to read, but when you edit the logic as text you can use the same shorthand used in the keyboard method of entering LD logic in the LD editor.

5.2.8 To Copy an Entire LD Block as Text

1. In the Project tab of the Navigator, expand the target and expand the Logic folder.
2. Expand the Program Blocks folder and, if the LD block resides in a user defined folder, expand the user-defined folder.
3. Optionally right-click the LD block and choose *Check Block*. This validates the LD block. If the logic contains errors, error messages appear in the Feedback Zone. If the logic contains errors, fix them.
4. Right-click the LD block and choose *Copy*. The contents of the LD block are copied to the Windows Clipboard.
5. Paste the content of the Windows Clipboard into a text editor. The LD logic is pasted as text.

5.2.9 To Copy a Section of LD Logic as Text

1. Optionally right-click the LD block and choose *Check Block*. This validates the LD block. If the logic contains errors, error messages appear in the Feedback Zone. If the logic contains errors, fix them.
2. In the LD editor, select the cells that contain the logic you want to copy as text.
3. Right-click the selected logic and choose *Copy*. The LD logic is copied to the Windows Clipboard.
4. Paste the content of the Windows Clipboard into a text editor. The LD logic is pasted as text.

5.2.10 To Copy Text into the LD Editor

1. In the text editor, select and copy the text representing LD logic to the Windows clipboard.
2. In the LD editor, do one of the following:
3. Select the cells that you want to overwrite.
- or -
4. Click the cell that will be at the upper left corner of the LD logic you want to paste.
5. Right-click and choose *Paste*. Existing cells in the LD editor are overwritten with the content of the Windows clipboard.

5.2.11 To Move or Duplicate LD Logic

In the LD Editor, select a range of logic.

To move it:

- Click the selection and drag it to a new location.
-
- To duplicate it:
 - Press CTRL while clicking the selection, and drag the selection to where you want to place the duplicate.

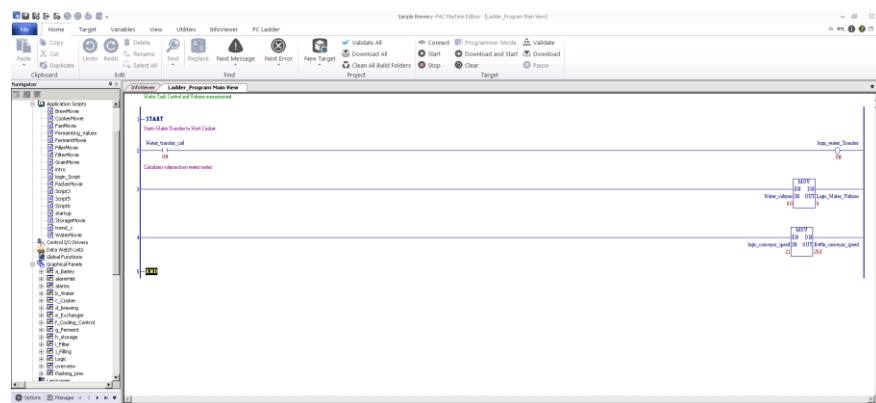
When you release the mouse button, the selection is respectively moved to the new location or a duplicate copy of the selected logic is placed in the new area.

5.2.12 Working with the LD Editor Online

When online in monitor mode, you can monitor the Controller but cannot change anything in the Controller or edit your logic. The LD editor animates LD logic to reflect program execution in the Controller. Data values change typically every 250 milliseconds while coils and contacts indicate power flow. The number of milliseconds is set in the Update Rate target property.

When you are online in programmer mode, you can edit your logic, make changes in the Controller, and monitor the Controller. Many Emerson targets also support Run Mode Store; that is, you can download logic to them when they are running (Figure 22).

Figure 22: LD Editor Online



Three methods are at your disposal to make changes in your LD logic and download the changes to an online running Controller:

Test Edit (PACSystems Only)

With Test Edit, you can perform transacted online programming. That is, you can modify an LD block in the LD editor while seeing both the original and modified logic, test the new logic's execution on the PACSystems, and then either keep the modified logic or roll back to the original logic

5.2.13 To Begin Editing in Test Edit

1. In the LD editor, start modifying the logic. The Logic Change Options dialog box appears.
2. Select Enter Test Edit Mode and click OK. The LD editor is placed in Online Edit mode, with some visual cues to remind you that a Test Edit session is in progress. Because Test Edit is an extension of the Run Mode Store (RMS), only changes supported for an RMS are supported in Test Edit mode. At any time, you can test the modified logic.

5.2.14 To Begin Testing the Modified Logic

1. Click anywhere inside the LD editor.
2. From the Debug menu, choose Begin Test. The modified logic is downloaded to the PACSystems, which then executes the modified logic. The original block of logic is retained in PACSystems memory as an inactive block.

You now have three possible courses of action:

1. Cancel the test and continue editing the logic while in Test Edit mode.
 - Cancel Test Edit mode and restore the original logic in the PACSystems.
 - Accept the changes you tested and commit them to the PACSystems.

5.2.15 To Cancel the Test and Continue Editing the Logic While in Test Edit Mode

1. Click Anywhere Inside the LD Editor

From the Debug menu, choose Cancel Test. The modified logic block is deactivated in the PACSystems. The original block is activated and executed. The modified logic block retains its changes in the LD editor and you can continue editing it and test it again later.

5.2.16 To Cancel Test Edit Mode and Restore the Original Logic in the PACSystems

1. Click anywhere inside the LD editor.
2. From the Debug menu, choose *Cancel Edit*. If you made no changes to logic, the Test Edit session ends immediately. If you made any changes to logic, the Cancel Test Edit dialog box appears.
3. In the Cancel Test Edit dialog box, select one of the following options and click OK.

- *Restore original logic in editor and maintain equality:* The PACSystems deactivates and deletes the modified block of logic, and it activates and executes the original block. The LD editor discards the modified logic from the block, retaining only the block's original logic. Your Test Edit session ends.

You are online in programmer mode and logic equal.

- *Keep modified logic in editor and lose equality:* The PACSystems deactivates and deletes the modified block of logic, and it activates and executes the original block. The LD editor retains the modified block of logic and no longer displays the original logic where it was different. Your Test Edit

5.2.17 To Accept the Changes you Tested and Commit Them to the PACSystems Controller

1. Click anywhere inside the LD editor.
2. From the Debug menu, choose Accept Edit. The modified logic is committed to the PACSystems. The original logic is completely deleted from the PACSystems; it no longer exists as an inactive block. The Test Edit session is ended. You are online in programmer mode and logic equal.

Word-for-Word Changes

A word-for-word is a small change in logic made while online that generally fits in the same amount of memory as the original logic. For example, changing the type of contact or coil or changing an operand is usually a word-for-word change.

Word-for-word changes can be completed online to PACSystems, Series 90- 70, Series 90-30, VersaMax, and Series 90 Micro Controllers. (VersaMax Nano / Micro Controllers do not support word-for-word changes.)

To make word-for-word changes

1. While online to a target Controller, make a change to LD logic that does not change the logic size.
2. Do one of the following:
 - a. If the Logic Change Options dialog box appears, select Word-for-word change and click *OK*.
- or -
 - b. If the Word for Word Change dialog box appears, click *Yes*. The change is downloaded to the Controller.

5.2.18 Go Not Equal, Keep Working, and Download Changes

If you make changes in logic that do not qualify for Test Edit or a word-for-word change, or if you choose not to enter Test Edit mode or make a word-for-word change, logic becomes not equal.

To regain equality, you can download your changes to the Controller or upload the logic from the Controller. You can generally download to the Controller whether the target is running or not. All Controller families support this Run Mode Store capability, but not every Controller in every family.

5.2.19 Affecting BOOL Variables

5.2.19.1 To Turn On/Off or Force a Variable

In the LD editor, right-click a BOOL variable anywhere in LD logic and choose *Force ON*, *Force OFF*, *Toggle Force*, *Turn ON*, *Turn OFF*, or *Toggle IO*.

Note: Forcing a variable ON or OFF overrides any actions the application may take during runtime. That is, if a variable is forced OFF (0), but LD logic is trying to set it to ON (1), it remains set to OFF.

5.2.19.2 LD Instructions

The following is a list of all LD instructions available in Logic Developer - PLC. Companion help indicates which Controllers support the instructions.

ADVANCED MATH					
ACOS	ATAN_REAL ^P	EXP	LN_REAL ^P	SIN	TAN
ACOS_REAL ^P	ATAN_REAL ^P	EXP_REAL ^P	LN_REAL ^P	SIN_REAL ^P	TAN_REAL ^P
ACOS_REAL ^P	COS	EXP_REAL ^P	LOG	SIN_REAL ^P	TAN_REAL ^P
ASIN	COS_REAL ^P	EXPT	LOG_REAL ^P	SQRT_DINT	
ASIN_REAL ^P	COS_REAL ^P	EXPT_REAL ^P	LOG_REAL ^P	SQRT_INT	
ASIN_REAL ^P	DEG_TO_RAD_READ ^P	EXPT_REAL ^P	RAD_TO_DEG_REAL ^P	SQRT_REAL ^P	
ATAN	DEG_TO_RAD_REAL ^P	LN	RAD_TO_DEG_REAL ^P	SQRT_REAL	
^P Indicates instructions exclusive to PACSystems Controllers.					

BIT OPERATIONS			
AND_DWORD [^]	BIT_SET_DWORD [^]	NOT_WORD	SHIFTL_DWORD [^]
AND_WORD	BIT_SET_WORD	OR_WORD [^]	SHIFTL_WORD
BIT_CLR_DWORD [^]	BIT_TEST_DWORD [^]	OR_WORD	SHIFTR_DWORD [^]
BIT_CLR_WORD	BIT_TEST_WORD	ROL_DWORD [^]	SHIFTR_WORD
BIT_POS_DWORD [^]	MASK_COMP_DWORD	ROL_WORD	XOR_DWORD [^]
BIT_POS_WORD	MASK_COMP_WORD	ROR_DWORD [^]	XOR_WORD
BIT_SEQ	NOT_DWORD	ROR_WORD	
[^] Indicates instructions shared by PACSystems and Series 90-70 Controllers and exclusive to them.			

COILS				
COIL	NCCOIL	NTCOIL	PTCOIL	SETCOIL
CONTCOIL	NEGCOIL	POSCOIL	RESETCOIL	
^P Indicates instructions exclusive to PACSystems Controllers.				

COMMUNICATION	
MODBUS_TCP_RW	PNIO_DEV_COMM

CONTACTS				
CONTCON	HIALR [^]	NCCON	NOCON	NTCON ^P
PTCON ^P	FAULT [^]	LOALR [^]	NEGCON [^]	NOFLT [^]
POSCON [^]				
^P Indicates instructions exclusive to PACSystems Controllers.				
[^] Indicates instructions shared by PACSystems and Series 90-70 Controllers and exclusive to them.				

CONTROL				
DO_IO	EXIT_FOR [^]	MASK_IO_INTR	R_TRIG ^P	SUS_IO [^]
SWITCH_POS ^P	DRUM	F_TRIG ^P	PID_IND	SCAN_SET_IO ^P
SUSP_IO_INTR	END_FOR [^]	FOR_LOOP [^]	PID_ISA	SER
SVC_REQ				
<p>^PIndicates instructions exclusive to PACSystems Controllers.</p> <p>[^]Indicates instructions shared by PACSystems and Series 90-70 Controllers and exclusive to them.</p>				

CONVERSIONS				
BCD4_TO_INT	DINT_TO_INT [^]	INT_TO_UINT [^]	REAL_TO_UINT [^]	UNIT_TO_REAL [^]
BCD4_TO_REAL	DINT_TO_REAL ^P	LREAL_TO_DINT ^P	REAL_TO_WORD	WORD_TO_REAL
BCD4_TO_UINT [^]	DINT_TO_REAL ^{L^P}	LREAL_TO_REAL ^{L^P}	TRUNC_DINT	
BCD8_TO_DINT [^]	DINT_TO_UINT ^T	RAD_TO_DEG	TRUNC_INT	
BCD8_TO_REAL ^{L[^]}	INT_TO_BCD4	REAL_TO_DINT	UINT_TO_BCD4 [^]	
DEG_TO_RAD	INT_TO_DINT [^]	REAL_TO_INT	UINT_TO_DINT [^]	
DINT_TO_BCD8 [^]	INT_TO_REAL	REAL_TO_LREAL ^{L^P}	UINT_TO_INT [^]	

COUNTERS	
DNCTR	UPCTR

DATA MOVE			
ARRAY_SIZE ^P	BUS_RMW_WORD ^P	DATA_INIT_REAL [^]	SHFT_DWORD [^]
ARRAY_SIZE_DIM1 ^P	BUS_TS_BYTE	DATA_INIT_WORD [^]	SHFR_WORD
ARRAY_SIZE_DIM2 ^P	BUS_TS_WORD	MOVE_BOOL	SIZE_OF ^P
BLK_CLR_WORD	BUS_WRT_BYTE	MOVE_DATA ^P	SWAP_DWORD [^]
BLKMOV_DINT [^]	BUS_WRT_DWORD ^P	MOVE_DATA_EX ^P	SWAP_WORD [^]
BLKMOV_DWORD	BUS_WRT_WORD ^P	MOVE_DINT	VME_CFG_READ ⁷⁰
BLKMOV_INT	COMM_REQ	MOVE_DWORD	VME_CFG_WRITE ⁷⁰
BLKMOVE_REAL	DATA_INIT_ASCII [^]	MOVE_FROM_FLAT ^P	VME_RD_BYTE ⁷⁰
BLKMOVE_UINT [^]	DATA_INIT_COMM [^]	MOVE_INT	VME_RD_WORD ⁷⁰
BLKMOVE_WORD	DATA_INIT_DINT [^]	MOVE_LREAL ^P	VME_RMW_BYTE ⁷⁰
BUS_RD_BYTE	DATA_INIT_DLAN [^]	MOVE_REAL	VMW_RMW_WORD ⁷⁰
BUS_RD_DWORD ^P	DATA_INIT_DWORD [^]	MOVE_TO_FLAT ^P	WME_TS_BYTE ⁷⁰
BUS_RD_WORD ^P	DATA_INIT_INT [^]	MOVE_UINT [^]	VME_TS_WORD ⁷⁰
BUS_RMW_BYTE ^P	DATA_INIT_LREAL ^P	MOVE_WORD	VMW_WRT_BYTE ⁷⁰
BUS_RMW_DWORD ^P	DATA_INIT_UINT [^]	SHFR_BIT	VME_WRT_WORD ⁷⁰
^P Indicates instructions exclusive to PACSystems Controllers ⁷⁰ Indicates instructions exclusive to Series 90-70 Controllers [^] Indicates instructions shared by PACSystems and Series 90-70 Controllers and exclusive to them.			

DATA TABLE			
ARRAY_MOVE_BOOLEAN	FIFO_WRT_WORD ^	SEARCH_GE_UINT ^	SEARCH_NE_DINT
ARRAY_MOVE_BYTE	LIFO_RD_DINT ^	SEARCH_GE_WORD	SEARCH_NE_DWORD ^
ARRAY_MOVE_DINT	LIFO_RD_DWORD ^	SEARCH_GT_BYTE	SEARCH_NE_INT
ARRAY_MOVE_DWORD ^	LIFO_RD_INT ^	SEARCH_GT_BYTE	SEARCH_NE_INT
ARRAY_MOVE_INT	LIFO_RD_UINT ^	SEARCH_GT_DWORD	SEARCH_NE_WORD ^
ARRAY_MOVE_UINT ^	LIFO_RD_WORD ^	SEARCH_GT_INT ^	SORT_INT ^
ARRAY_MOVE_WORD	LIFO_WRT_DINT ^	SEARCH_GT_UINT ^	SORT_UINT ^
ARRAY_RANGE_DINT ^	LIFO_WRT_DWORD ^	SEARCH_GT_WORD	SORT_WORD ^
ARRAY_RANGE_DWORD	LIFE_WRT_INT ^	SEARCH_LE_BYTE	TBL_RD_DINT ^
ARRAY_RANGE_INT ^	LIFO_WRT_UINT ^	SEARCH_LE_DINT	TBL_RD_DWORD ^
ARRAY_RANGE_UINT ^	LIFO_WRT_WORD ^	SEARCH_LE_DWORD ^	TBL_RD_INT ^
ARRAY_RANGE_WORD ^	SEARCH_EQ_BYTE	SEARCH_LE_WORD	TBL_RD_UINT ^
FIFO_RD_DINT ^	SEARCH_EQ_DINT	SEARCH_LE_WORD	TBL_RD_WORD ^
FIFO_RD_DWORD ^	SEARCH_EQ_DWORD ^	SEARCH_LE_DWORD ^	TBL_WRT_DINT ^
FIFO_RD_INT ^	SEARCH_EQ_INT	SEARCH_LT_BYTE	TBL_WRT_DWORD ^
FIFO_RD_UINT ^	SEARCH_EQ_UINT ^	SEARCH_LT_DINT	TBL_WRT_INT ^
FIFO_RD_WORD ^	SEARCH_EQ_WORD	SEARCH_LT_DWORD ^	TBL_WRT_UINT ^
FIFO_WRT_DINT ^	SEARCH_GE_BYTE	SEARCH_LT_INT	TBL_WRT_WORD

FIFO_WRT_DWORD ^	SEARCH_GE_DINT	SEARCH_LT_UINT ^	
FIFO_WRT_INT ^	SEARCH_GE_DWORD D ^	SEARCH_LT_WORD	
FIFO_WRT_UINT ^	SEARCH_GE_INT	SEARCH-NE_BYTE	
<i>^ Indicates instructions shared by PACSystems and Series 90-70 Controllers and exclusive to them.</i>			

MATH				
ABS_DINT ^	ADD_REAL	DIV_UINT ^	MUL_MIXED ^	SUB_DINT
ABS_INT ^	ADD_UINT ^	MOD_DINT	MUL_REAL	SUB_INT
ABS_LREAL ^	DIV_DINT	MOD_INT	MUL_UINT ^	SUB_LREAL ^P
ABS_REAL ^	DIV_INT	MOD_UINT ^	SCALE_DINT ^P	SUB_REAL
ADD_DINT	DIV_LREAL ^P	MUL_DINT	SCALE_INT	DUB_UINT ^
ADD_INT	DIV_MIXED ^	MUL_INT	SCALE_UINT ^P	
ADD_LREAL ^P	DIV_REAL	MUL_LREAL ^P	SCALE_WORD _P	

5.3 FBD Editor

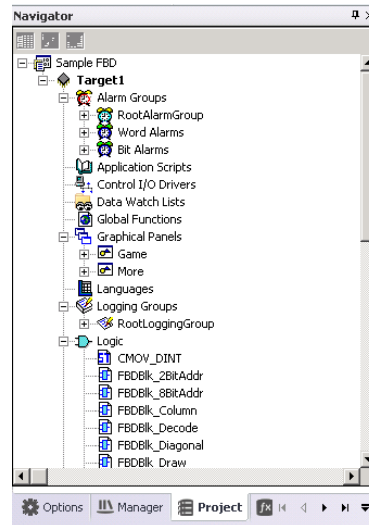
The Function Block Diagram (FBD) editor is used to create logic with the Function Block Diagram programming language. FBD is a process language logic graphically represents the programmed actions performed by a Controller as it executes.

The FBD editor is free form; that is, instructions and text boxes can be placed anywhere on the FBD editor where there is empty space. Sequences of instructions can be wired together horizontally and vertically.

You can work with the FBD editor while offline to edit a disk copy of a project, or you can edit an FBD block of logic online, but this causes the logic to become not equal until you download the FBD block.

You can customize the appearance and behavior of the FBD editor by setting options. An FBD block is a named section of FBD Logic that is compiled and downloaded to the Controller represented by the associated target.

Figure 23: FBD Editor



5.3.1 To Customize the FBD Editor

1. In the Options tab of the Navigator, expand the Editors folder, then expand the Function Block Diagram folder (Figure 23).
2. Right-click the Colors and Preferences page and choose *Properties*. The configurable settings appear as properties in the Inspector.
3. In the Inspector, adjust settings as required.

5.3.2 To Create an FBD Block

1. In the Project tab of the Navigator, right-click the Program Blocks folder, point to *New*, and then choose *FBD Block*. A new FBD block with a default name is created.
2. Rename the block as desired.

5.3.3 To Open an FBD Block for Editing

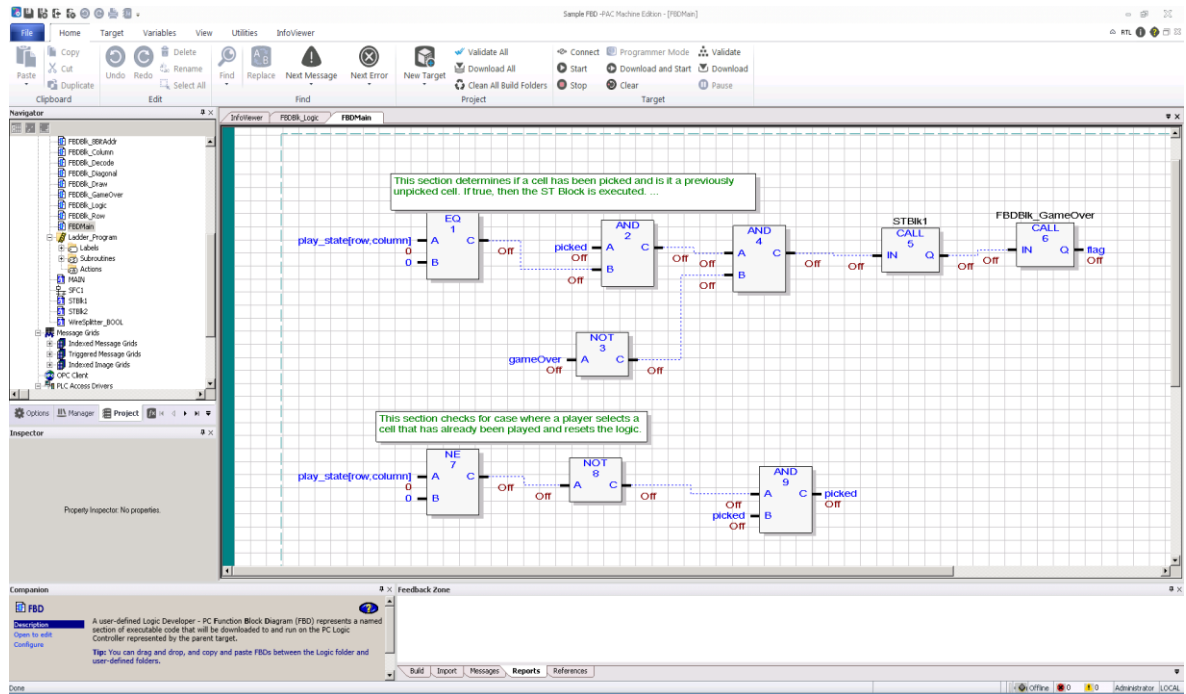
In the Project tab of the Navigator, double-click an FBD block. The block opens in the FBD editor (Figure 24).

Note: You can have multiple blocks open for editing. To navigate to another open FBD block, click the tab that displays its name at the top of the editor window.

Working with the FBD Editor Offline

While in offline mode, there is no live communication between the FBD editor and the target. Most logic development is done while offline. The following diagram illustrates some of the more common operations you can perform using the FBD editor offline.

Figure 24: FBD Editor Offline



5.3.4 To Insert an Instruction

1. In the FBD editor, right-click an empty cell and choose *Insert Instruction*. A smart list appears listing all available instruction mnemonics.
2. Choose an instruction mnemonic from the list and press ENTER.

5.3.5 To Assign a Parameter Beside an Instruction

There are two methods to assign a parameter to an instruction.

First method (from the FBD editor)

1. In FBD editor, hover the mouse pointer immediately before an input connection point or immediately beyond an output connection point of an FBD instruction, so that the mouse pointer changes as seen in
2. In the smart list that appears, enter or choose an existing variable, constant, or expression to assign to the instruction parameter. In the FBD editor, the instruction may display as seen in Figure 26:

Figure 25: Double Click the Connection Point

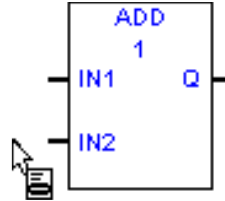
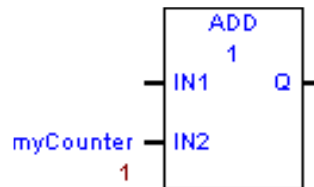


Figure 26: Assigning a Value to the Instruction Parameter



Note: Inside the rectangle of the ADD instruction, the “1” indicates the solve order of the instruction. Under the variable *myCounter*, the “1” is the initial value of *myCounter*.

Second Method (outside the FBD editor)


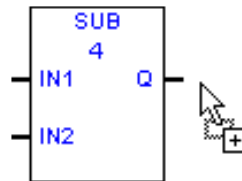
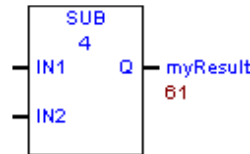
- In the Variables tab of the Navigator, or the Data Watch tool, select a variable.
- Drag the selected variable to the connection point of an FBD instruction.
- When the mouse pointer appears as , release the left mouse button to assign the variable to a parameter of the FBD instruction.
- For example, the following diagram shows that you can assign a parameter to the FBD SUBTRACT instruction at the output connection point named Q.

Figure 27: Double Click the Connection Point Q



After you have assigned the parameter, the instruction will appear as Figure 28:

Figure 28: Assigned FBD Subtract to Connection Point Q



Note: In the above diagram, “61” is the initial value of myResult. “4” inside the rectangle of the FBD instruction.

5.3.6 To Assign a Parameter Above an FBD Instruction or Function Block Instance

The FBD Call instruction and every function block instance (an instance of a counter, PID, timer, HART utility, or user-defined function block) require a parameter to be assigned immediately above the instruction or instance.

1. In the FBD editor, hover with the mouse pointer immediately above an FBD Call or function block instance so that the mouse pointer appears.


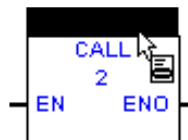
When the mouse pointer appears as , click the left mouse button. The instruction or function block instance appears in the FBD editor with the parameter selected, as shown in Figure 29.

Figure 29: Assigned Parameter Above an FBD Instruction



2. Without moving the mouse pointer, click again.

A smart list appears. If this is an FBD Call instruction, the smart list contains a list of all callable blocks of logic in your target. If this is an FBD function block instance, the smart list contains a list of variables that you can choose from to assign instance data to the function block instance. Normally you would choose or create instance data that is not used by any other function block instance, but you can assign the same instance data to multiple function block instances if you want.

3. In the smart list, enter or choose an existing block of logic, or enter a new variable, or enter or choose an existing variable. The block appears as the destination of the FBD Call or the variable is assigned as instance data to the function block instance.

5.3.7 To Check (Validate) a Single FBD Block

1. In the Project tab of the Navigator, expand the target that contains the block to check, and then expand the Logic folder.
2. Expand the Program Blocks folder and then, if the block resides in a user defined folder, expand the user-defined folder.
3. Right-click the block and choose *Check Block*. Machine Edition checks the block for errors. Any messages or errors are displayed in the Build tab of the Feedback Zone.

5.3.8 To Change the Number of Inputs for FBD instructions (ADD, AND, MUL, OR, SUB, XOR)

1. In the FBD editor, select an ADD, AND, MUL, OR, SUB, or XOR instruction.
2. In the Inspector, select the Number of Inputs property, and then enter or choose from the list the number of inputs you need.
3. If required, draw a wire or assign a variable or constant to the input and output parameters.
4. Hold down the left mouse button, and then drag the mouse pointer to a connection point of another FBD instruction (Figure 30).
5. Continue to hold down the left mouse button and drag the mouse pointer to a connection point of another FBD instruction (Figure 31).
6. Release the left mouse button. The newly drawn wire appears as a solid (analog) or a dashed (discrete) line in the FBD editor (Figure 32).

Figure 30: Selecting the function block instance

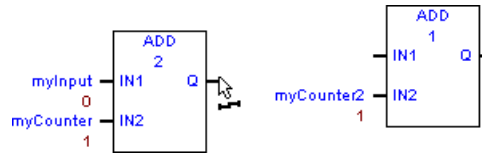


Figure 31: Changing the Number of Inputs

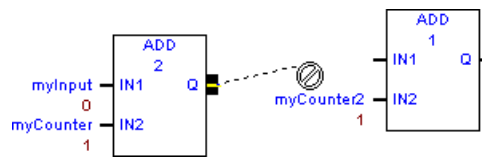
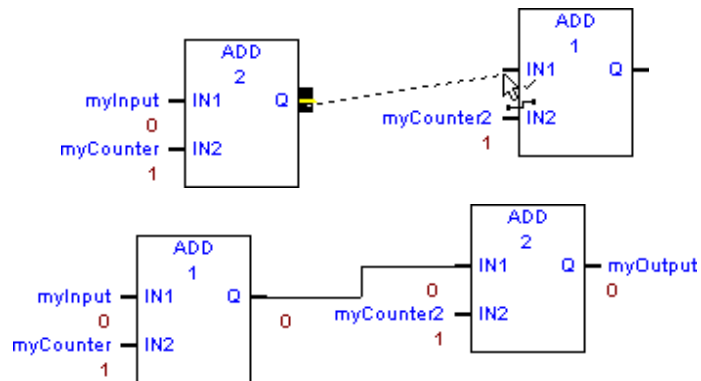



Figure 32: Newly Drawn Wire Appears



Notes:

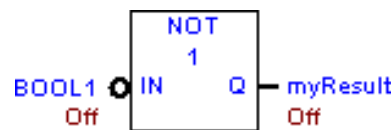
- The numbers “1” and “2” inside the rectangles have been reversed, indicating a change in the solve order.
- The zeroes under the wire are the value that is travelling over the wire. It’s the same value at both ends.
- At least two instructions must exist in the FBD editor.
- You cannot draw a wire to or from a Text box.
- In the FBD editor, hover the mouse pointer above a connection point of an FBD instruction or function block instance.
- When the mouse pointer appears as , you can start a wire.

5.3.9 To Negate an FBD Parameter

(For discrete parameters only).

1. In the FBD editor, assign a BOOL parameter to a connection point of an FBD instruction that requires a BOOL variable.
2. Right-click a BOOL variable assigned to a connection point, for example, in the diagram above, *BOOL1*.
3. In the list box that appears, choose *Negate*. The diagram now appears as shown in Figure 33.

Figure 33: Assign a BOOL Parameter to a Connection Point



Note: Negating a BOOL parameter while online causes logic to be not equal; turning a BOOL parameter ON/OFF while online does not affect logic equality.

5.3.10 To Negate an FBD Wire

(For discrete FBD wires only.)

1. If required, draw the FBD wire from a connection point that requires a BOOL variable, to a connection point that also requires a BOOL variable. If the wire is discrete, it appears as a dotted line.
2. Right-click the wire and then choose *Negate*.

5.3.11 To Move or Duplicate FBD Logic

In the FBD editor, select an FBD instruction or Text box.

To Move It

- Click the selection and drag it to a new valid location. When you move an instruction, the parameters and wires assigned to it move along with it.

To Duplicate It

- Press CTRL while clicking the selection and drag the selection to where you want to place the duplicate. When you release the mouse button, a duplicate copy of the selected instruction is placed in the new area. When you duplicate an instruction, its parameters are also duplicated; wires are not duplicated.

5.3.12 To Zoom in or Zoom Out an FBD Block

1. Click anywhere in the FBD editor.
2. Do one of the following:

- In the Inspector, expand the Diagram Settings group of properties, and then set the zoom property.

OR

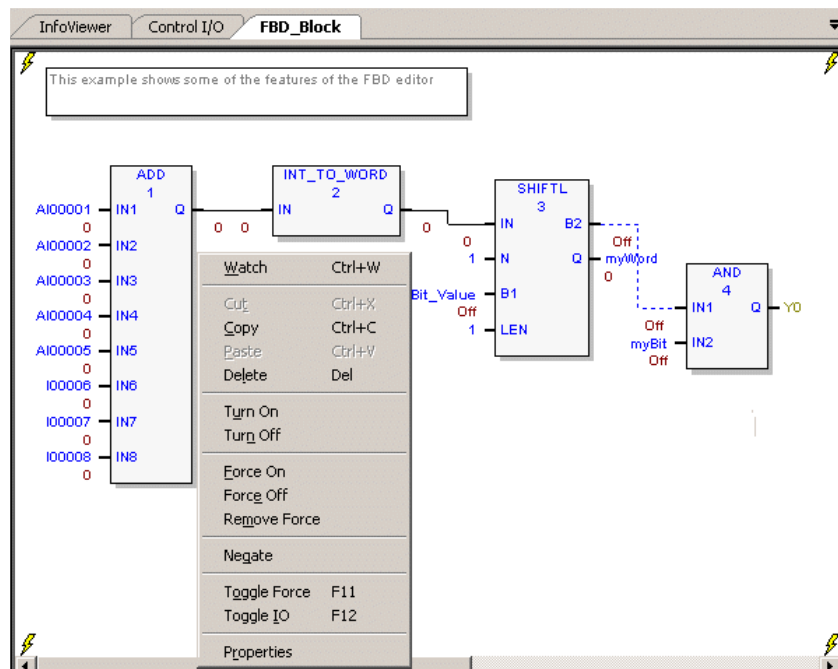
- To zoom out, press the “-“ key. To zoom in, press the “+“ key.

Note: To zoom in an FBD block display in the FBD editor makes the FBD appear larger; to zoom out an FBD block display makes the FBD appear smaller.

5.3.13 Working with the FBD Editor Online

When you are online in monitor mode you can edit your logic, make changes in the controller, and monitor the controller. PACSystems targets also support Run Mode Store; that is, you can download logic to a running PACSystems.

Figure 34: Example FBD Editor



5.3.14 To Turn On/Off or Force a Variable

In the FBD editor, right-click a BOOL variable anywhere in FBD logic and choose *Force On*, *Force Off*, *Toggle Force*, *Turn On*, *Turn Off*, or *Toggle IO*.

Note: Forcing a variable ON or OFF overrides any actions the logic may take during runtime. That is, if a variable is forced OFF (0), but FBD logic is trying to turn it ON (1), it stays OFF.

5.3.15 FBD Instructions, Functions, and Function Blocks

(PACSystems only.) The following is a list of all FBD instructions, functions, and function blocks available in Logic Developer – PLC:

ADVANCED MATH					
ABS	ASIN	COS	EXPT	LOG	SQRT
ACOS	ATAN	EXP	LN	SIN	TAN

BIT OPERATIONS			
AND	OR	ROR	SHIFTR
NOT	ROL	SHIFTL	XOR

COMPARISON			
CMO	GE	LE	NE
EQ	GT	LT	RANGE

COMMUNICATION
PINO DEV COMM

CONTROL				
DO_IO	MASK)IO_INTR	PID_ISA	SCAN_SET_IO	SUSP_IO_INTR
F_TRIG	PID_IND	R_TRIG	SUS_IO	SVC_REQ

COUNTERS	
DNCTR	UPCTR

DATA MOVE			
ARRAY_SIZE	BUS_RMW_BYTE	BUS_TS_WORD	MOVE
ARRAY_SIZE_DIM1	BUS_RMW_DWORD	BUS_WRT	MOVE_DATA_EX
ARRAY_SIZE_DIM2	BUS_RMW_WORD	COMM_REQ	MOVE_TO_FLAT
BUS_RD	BUS_TS_BYTE	FANOUT	SIZE_OF

MATH			
ADD	MOD	NEG	SUB
DIV	MUL	SCALE	

PROGRAM FLOW	
ARG_PRES	CALL

TIMER FUNCTION BLOCKS			
OFDT_HUNDS	ONDTR_HUNDS	TMR_HUNDS	TOF
OFDT_SEC	ONDTR_SEC	TMR_SEC	TON
OFDT_TENTHS	ONDTR_TENTHS	TMR_TENTHS	TP
ODFT_THOUS	ONDTR_THOUS	TMR_THOUS	

TYPE CONVERSION				
BCD4_TO_INT	DINT_TO_DWORD	INT_TO_DINT	REAL_TO_DINT	UINT_TO_DINT
BCD4_TO_REAL	DINT_TO_INT	INT_TO_REAL	REAL_TO_INT	UINT_TO_INT
BCD4_TO_UINT	DINT_TO_LREAL	INT_TO_UINT	REAL_TO_LREAL	UINT_TO_REAL
BCD8_TO_DINT	DINT_TO_REAL	INT_TO_WORD	REAL_TO_UINT	UINT_TO_WORD
BCD8_TO_REAL	DINT_TO_UINT	LREAL_TO_DINT	TRUNC_DINT	WORD_TO_INT
DEG_TO_RAD	DWORD_TO_DINT	LREAL_TO_REAL	TRUNC_INT	WORD_TO_UINT
DINT_TO_BCD8	INT_TO_BCD4	RAD_TO_DEG	UINT_TO_BCD4	

5.4 IL Editor

Instruction List (IL) is a programming language specified by the IEC 61131-3 standard. This text language is accumulator-based and much like the assembly languages used for programming microprocessors. The instructions executed by an IL block modify or use an accumulator that is located in Controller memory. Two types of accumulators are defined: one analog accumulator for numeric and

bitwise operations and eight Boolean accumulators for discrete logic to support eight levels of nested Boolean expressions. The IL editor is free-form with an option to apply a standard formatting rule. The appearance and behavior of the IL editor is user-configurable.

Note: Only Series 90-30, VersaMax Controllers, and VersaMax Nano/Micro support IL logic.

5.4.1 To Configure Accumulators

1. In the Project tab of the Navigator, right-click the Program Blocks folder and choose *Properties*.

The Inspector displays the Accumulator Address properties.

2. In the Boolean Start property, enter the reference address of the first of eight controller memory locations to use for Boolean accumulators.

The ending address is calculated automatically. The memory area must be %T,%M, or %Q.

3. In the Analog Start property, enter the Controller memory locations to use for the analog accumulator.

The ending address is calculated automatically. The memory area must be %R, %AI or %AQ.

5.4.2 To Create an IL Block

1. In the Project tab of the Navigator, right-click the Program Blocks folder, point to *New*, and choose *IL Block*.

An empty IL block with the default name "ILBkn" is added to the folder, where *n* represents a unique number.

2. Rename the block as desired.

5.4.3 To Open an IL Block Editing

In the Project tab of the Navigator, right-click an IL Block and choose *Open*. The block opens in the IL editor.

Note: You can have multiple blocks open for editing. To navigate to another open IL block, click the tab displaying its name at the top of the editor window.

5.4.4 Working with the IL Editor Offline

Most project development is carried out while offline from the target Controller. Editing while offline provides maximum flexibility and enables you to interact with the Machine Edition tools as shown in (Figure 35).

Figure 35: IL Editor Offline

```

'-----
'
' Created: Friday, February 02, 2001
'
'-----
'This is a test of Basic Boolean IL Logic. When run,
'All Test Discrete Variables should be ON.
'
'Boolean Storage Cases
'
LD_BOOL #ALW_ON
ST_BOOL Test_STBool
LD_BOOL #ALW_OFF
STN_BOOL Test_BOOL Target1.#ALW_OFF
'
'Boolean Operators TRUE Cases
'
LD_BOOL #ALW_ON
AND #ALW_ON
ST_BOOL Test_AND_T
OR #ALW_OFF
ST_BOOL Test_OR_T
LD_BOOL #ALW_OFF

```

5.4.5 To Insert an Instruction

In the IL editor, right-click and choose Insert Keyword. A smart list appears listing all available instruction mnemonics.

From the list, select an instruction and press ENTER. The instruction is inserted in your logic.

5.4.6 To Assign Operands to an Instruction

1. In the IL editor, right-click and choose *Insert Variable*. A smart list appears showing all your defined variables.
2. Type, or choose from the list, a variable name or reference address and then press ENTER. The name appears in your logic.

Note: If you entered a reference address or a new variable name, you must create a variable from it.

5.4.7 To Create a Variable from a Reference Address

In the IL editor, right-click a reference address, point to Create “name” as, and then choose a data type.

A variable is created and a default name is applied. For example, if the reference address is %R0032, the auto-created variable is named R00032.

5.4.8 To Create a Variable from a Name

1. In the IL editor, right-click a name, point to Create “name” as, and then choose a data type.
2. A variable is created with the name you right-clicked.
3. Map the variable to Controller memory.

5.4.9 To Move or Duplicate IL Logic

1. In the IL editor, select a range of logic.
2. To move it, click the selection and drag it to a new location. To duplicate it, press `CTRL` while clicking the selection, and drag the selection to where you want to place the duplicate.
3. When you release the mouse button, the selection is respectively moved to the new location or a duplicate copy of the selected logic is placed in the new area.

5.4.10 To Insert an Inline Comment

1. In the IL editor, click where you want to insert an inline comment.
2. Type an apostrophe (') followed by comment text.
3. Press `ENTER` to complete the comment.

5.4.11 To Insert a Block Comment

1. In the IL editor, click where you want to insert a block comment.
2. Type (* followed by comment text. A block comment can contain any number of characters and can span multiple lines.
3. Type *) to complete the block comment.

5.4.12 To Reformat IL Logic

In the IL editor, right-click and choose *Beautify Source*.

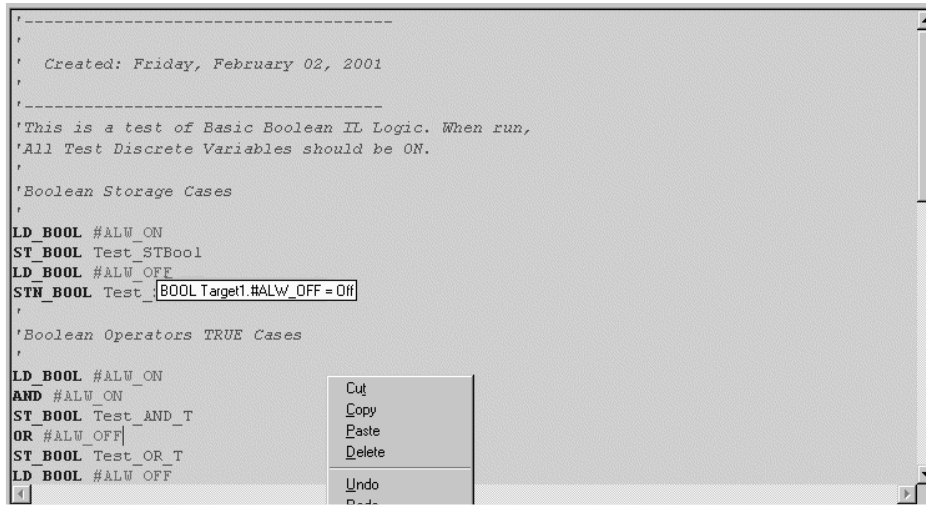
The entire content of the IL editor is reformatted according to the default formatting (indentation) rules.

5.4.13 Working with the IL Editor Online

When online in monitor mode, you can monitor the Controller but cannot change anything in the Controller or edit your logic.

When you are online in programmer mode, you can edit your logic, make changes in the Controller, and monitor the Controller. Many Emerson targets support Run Mode Store; that is, you can download logic to them when they are running.

Figure 36: RMS in IL Editor Online



```
'-----  
'  
' Created: Friday, February 02, 2001  
'  
'-----  
'This is a test of Basic Boolean IL Logic. When run,  
'All Test Discrete Variables should be ON.  
'  
'Boolean Storage Cases  
'  
LD_BOOL #ALW_ON  
ST_BOOL Test_STBool  
LD_BOOL #ALW_OFF  
STN_BOOL Test_BooleanTarget1.#ALW_OFF=Off  
'  
'Boolean Operators TRUE Cases  
'  
LD_BOOL #ALW_ON  
AND #ALW_ON  
ST_BOOL Test_AND_T  
OR #ALW_OFF  
ST_BOOL Test_OR_T  
LD_BOOL #ALW_OFF
```

5.4.13.1 To Monitor a Data Value

Click anywhere in the IL editor and hover the mouse pointer over a variable. A tooltip appears, showing the variable’s current value. This value, however, does not update automatically: you must move the mouse pointer away and back over the variable to update the value.

5.4.13.2 To Change a BOOL Variable’s State

In the IL editor, right-click the BOOL variable whose value you want to change and choose Turn On, Turn Off, or Toggle IO. The state of the variable in the target Controller changes when the command is received. It remains in that state until acted on by the Controller’s logic. To Force a BOOL variable’s state

In the IL editor, right-click a BOOL variable and choose *Force On*, *Force Off*, or Toggle Force.

The state of a forced variable remains unchanged, regardless of any actions by Controller logic.

5.4.13.3 To Remove the Force from a BOOL Variable

In the IL editor, right-click a BOOL variable and choose *Remove Forces*. The state of the forced variable is controlled by Controller logic from now on.

5.4.13.4 To Make Changes to IL logic and Download them to a Running Target Controller (if the Target Controller Supports it)

1. While online to a target Controller, make a change to IL logic.
2. In the Project tab of the Navigator, right-click the target and choose *Download to Controller*.
3. You are prompted to confirm a Run Mode Store. That is, the altered IL block will be downloaded to the running Controller without stopping it.

5.4.14 IL Instructions

The following is a list of all IL instructions available in Logic Developer - PLC. Companion help indicates which Controllers support the instructions.

BASIC INSTRUCTIONS					
ADD	GT	LT	OR	ST_DINT	XOR
AND	LD_BOOL	MOD	ORN	ST_INT	XORN
ANDN	LD_ENO	MUL	PT	ST_REAL	
DIV	LD_INT	NE	R	ST_WORD	
EQ	LDN_BOOL	NOT	S	STN_NOOL	
GE	LE	NT	ST_BOOL	SUB	

ADVANCED MATH				
ACOS	COS	LN	SQRT_DINT	TAN
ASIN	EXP	LOG	SQRT_INT	
ATAN	EXPT	SIN	SQRT_REAL	

BIT OPERATIONS				
AND-WORD	BIT-SEQ	MASK_COMP_DWORD	OR_WORD	SHIFTL_WORD
BIT_CLR_WORD	BIT_SET_WORD	MASK_COMP_WORD	ROL_WORD	SHIFTR_WORD
BIT_POS_WORD	BIT_TEST_WORD	NOT_WORD	ROR_WORD	XOR_WORD

COMMUNICATION
MODBUS_TCP_RW (VersaMax Micro CPUs with firmware version 4.00 or later)

CONTROL					
DO_IO	DRUM	PID_IND	PID_ISA	SER	SVC_REQ

CONVERSIONS				
BCD4_TO_INT	DIN_TO_REAL	_RAD_TO_DEG	REAL_TO_WORD	WORD_TO_REAL
BCD4_TO_REAL	INT_TO_BCD4	REAL_TO_DINT	TRUNC_DINT	
DEG_TO_RAD	INT_TO_REAL	REAL_TO_INT	TRUNC_INT	

COUNTERS	
DNCTR	UPCTR

DATA MOVE				
BLK_CLR_WORD	BLKMOV_REAL	COMM_REQ	MOVE_BOOL	MOVE_REAL
BLKMOVE_INT	BLKMOV_WORD	MOV_INT	MOVE_WORD	SHFT_WORD

DATA TABLE			
ARRAY_MOVE_BOOL	SEARCH_EQ_INT	SEARCH_GR_DINT	SEARCH_LT_BYTE
ARRAY_MOVE_BYTE	SEARCH_EQ_WORD	SEARCH_GT_INT	SEARCH_LT_DINT
ARRAY_MOVE_DINT	SEARCH_GE_BYTE	SEARCH_GT_WORD	SEARCH_LT_INT
ARRAY_MOVE_INT	SEARCH_GE_DINT	SEARCH_LE_BYTE	SEARCH_LT_WORD
ARRAY_MOVE_WORD	SEARCH_GE_INT	SEARCH_LE_DINT	SEARCH_NE_BYTE
SEARCH_EQ_BYTE	SEARCH_GE_WORD	SEARCH_LE_INT	SEARCH_NE_DINT
SEARCH_EQ_DINT	SEARCH_GT	SEARCH_LE_WORD	SEARCH_NEW_INT
SEARCH_NE_WORD			

MATH				
ADD_DINT	DIV_DINT	MOD_DINT	MUL_DINT	SCALE_WORD
ADD-INT	DIV_INT	MOD_INT	MUL_REAL	SUB_DINT
ADD_REAL	DIV_REAL	MUL_IN	SCALE_INT	SUB_INT
SUB_REAL				

PROGRAM FLOW				
CAL	CALCN	END_MCRN	JMPC	MCRN
RETC				

RELATIONAL				
EQ-DINT	GE_DINT	GT_DINT	LE_Dint	LT_DINT
NE_DINT	RANGE_DINT	EQ_INT	GE_INT	GT_INT
LE_INT	LT_INT	NE_INT	RANGE_INT	EQ_REAL
GE_REAL	GT_REAL	LE_REAL	LT_REAL	NE_REAL
RANGE_WORD				

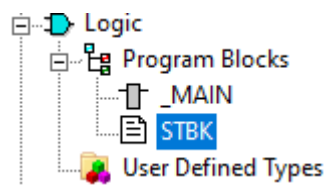
TIMER FUNCTION BLOCKS				
OFDT_HUND S	ONDTR_HUND S	TMR_HUNDS	OFDT_TENTH S	ONDTR_TENTH S
TMR_TENTHS	OFDT_THOUS	ONDTR_THOU S	TMR_THOUS	

5.5 ST Editor

Structured Text logic is a programming language specified by the IEC 61131-3 standard. The Structured Text (ST) Editor is a free-form text editor for editing Structured Text logic in the Machine Edition environment. With the ST editor, you can work on a disk copy of a structured text block (offline) or monitor the execution of an ST block running in the Controller (online). You can edit an ST block online, but this causes the logic to become not equal until you download the ST block. ST is a high-level language that uses various operators and functions. ST logic is supported by PACSystems controllers.

NOTE: As of version 9.70, the ST Editor options have been added to customize the editor. The Options are located under: *Editors > Structured Text*.

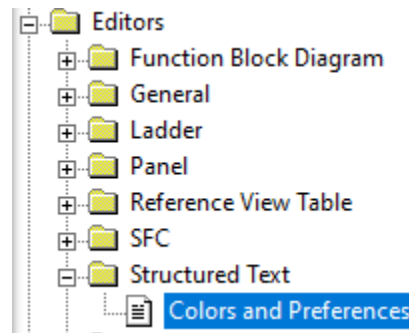
Figure 37: ST Editor



5.5.1 To Customize the ST Editor

1. In the Options tab of the Navigator, expand the Editors folder and then the Text folder (Figure 38).
2. Right-click a page (Colors or Preferences), and choose *Properties*. The configurable settings appear as properties in the Inspector.
3. In the Inspector, adjust the settings as required.

Figure 38: Customizing the ST Editor



5.5.2 To Create an ST Block

1. In the Project tab of the Navigator, expand the Logic folder, right-click the Program Blocks folder or a user-defined folder, point to *New*, and then choose *ST block*.
A new ST block with a default name is created under the Program Blocks folder or under a user-defined folder.
2. (Optional.) Rename the block.

5.5.3 To Create a Parameterized ST Block

1. Create an ST block
2. Do one of the following:
3. In the Inspector, set parameters for the block.
-or-
4. Set the block's Block Type property to *Parameterized Block*, and then optionally schedule the parameterized block.

Notes:

- By assigning parameters to the block, its Block Type property is automatically changed to Parameterized Block.
- A parameterized block cannot have both interrupt scheduling and parameters.

5.5.4 To Open an ST Block for Editing

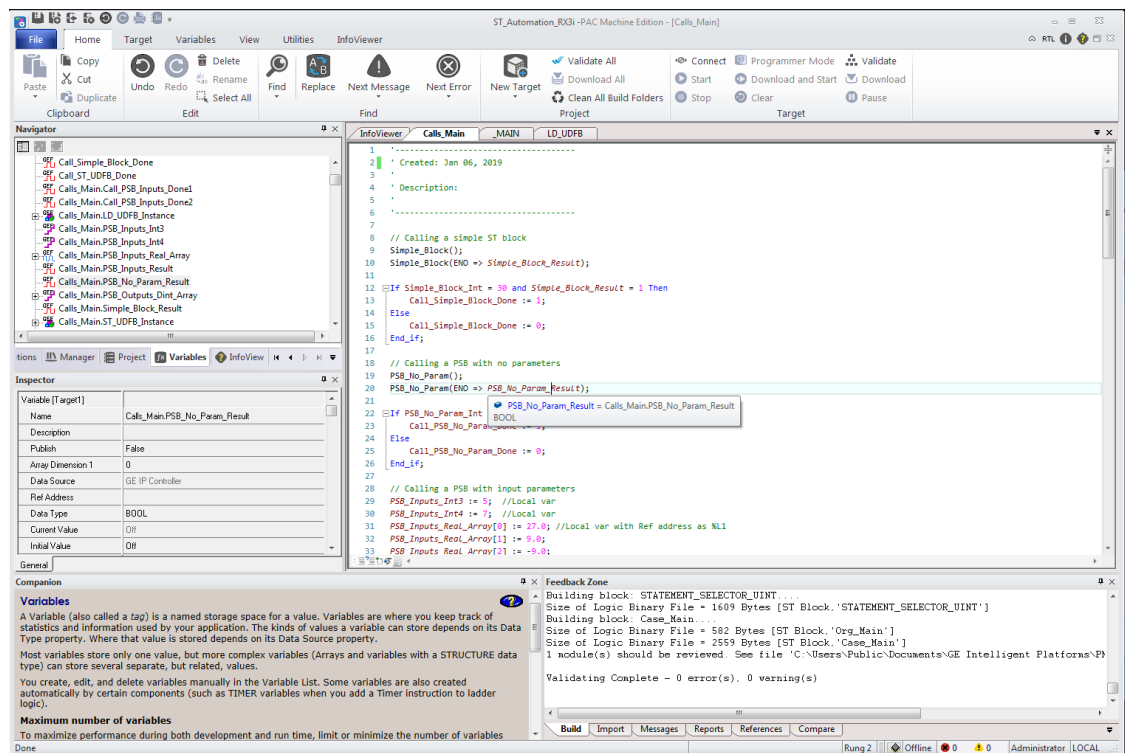
In the Project tab of the Navigator, under the Logic folder, under the Program Blocks folder, or under a user-defined folder, double-click an ST block.

The ST block opens in the ST editor.

5.5.5 Working with the ST Editor Offline

The ST editor interacts with the Machine Edition tools to provide maximum flexibility when editing a program. The following diagram illustrates some of the operations you can perform.

Figure 39: ST Editor Offline



5.5.6 To Insert an ST Variable or Keyword

1. In the ST editor, right-click and choose *Insert Variable/Keyword*. A smart list appears, prompting you to enter the name of an existing variable or keyword.

Tip: Keyboard shortcut Ctrl+Space also inserts an ST Variable or Keyword.

2. Type, or choose from the list, the item you want, and then press ENTER. The variable or keyword appears in the ST Editor.

Note:

- Constants must be manually entered.
- When entering an indirect reference as an operand, prefix the variable name with @ after selecting it from the smart list, for example, @IntVar. Indirect referencing is available for all register references (%R, %P, %L, %W, %AI, and %AQ).

5.5.7 To Create a Variable from a Name

In the ST editor, right-click a name that was entered as an operand, point to *Create "[name]" as*, and then choose a data type from the list that appears. A new variable of the specified data type is created and substituted for the name at every occurrence in the ST block. The name of the new variable is identical with the original name.

5.5.8 To Insert a Line Comment

1. In the ST editor, click where you want a line comment to begin.
2. Type an apostrophe (') or // followed by comment text.
3. Press *Enter* to complete the line comment. The line comment ends when a new line starts.

5.5.9 To Insert a Block Comment

1. In the ST editor, click where you want a block comment to begin.
2. Type (* followed by comment text). A block comment can contain any number of characters and can span multiple lines.
3. Type *) to complete the block comment.

5.5.10 To Select a Range of ST Logic

1. In the ST editor, click at the beginning of the range.
2. Press and hold shift and then click at the end of the range. All logic within the range is selected.

Tips:

- You can also click and drag from the beginning of the range to the end.
- After selecting a range of logic, you can click anywhere in it and drag it to another area in the ST editor.

5.5.11 To Move or Duplicate ST Logic

1. In the ST editor, select a range of logic.
2. To move it, click the selection and drag it to the new location. To duplicate it, press CTRL while clicking the selection, and then drag the selection to where you want to place the duplicate.

When you release the mouse button, the selection is respectively moved to the new location or a duplicate copy of the selected logic is placed in the new area.

5.5.12 To Locate all Occurrences of a Variable

1. In the Feedback Zone, click the References tab.
2. Click a variable anywhere it appears in ST logic, in the Variables tab of the Navigator, or in the Data Watch tool.

Each occurrence of the variable in your project (for a given target) is listed in the *References* tab of the Feedback Zone.

5.5.13 Working with the ST Editor Online

When online in monitor mode, you can monitor the Controller but cannot change anything in the Controller or edit your logic.

When you are online in programmer mode, you can edit your logic, make changes in the Controller, and monitor the Controller. PACSystems targets also support Run Mode Store; that is, you can download logic to a running PACSystems.

5.5.14 To View a Variable Value

In the ST editor, hover the mouse pointer over a variable to see its value. A tool tip displays the variable name, address, description, data type, and current value. If this is a BOOL variable, the tool tip also displays if it has been forced On (1) or Off (0).

5.5.15 To View the Value of an ST Parameterized Block Parameter

1. In the Project tab of the Navigator, expand the Program Blocks folder.
2. If the block that contains the call to the ST parameterized block whose parameter you want to monitor resides in a user-defined folder, expand the user-defined folder.
3. Right-click the block that contains the call and choose *Open*. The block appears in the appropriate editor.
4. In the editor, right-click the block call to the ST parameterized block and choose *Open Block*. The called ST parameterized block appears in the ST editor.
5. In the ST editor, hover the mouse pointer over the parameterized block parameter (variable) to see its value.

Depending on the context of the parameter, the tool tip displays the following items:

6. If the text refers to a parameter instead of a variable, and the block was opened from a CALL instruction (in an LD, ST, or FBD block), then if possible, the parameter is associated with the variable used in the call the block was opened from. In this case, the tool tip displays the parameter name, address of the variable, description of the parameter, data type, and value of the variable.
7. If the text refers to a parameter instead of a variable, and the block was not opened from a CALL instruction (in an LD, ST, or FBD block), then the tool tip displays the parameter name, description of the parameter, and data type only.

In either case, if this is a forced BOOL variable, the tool tip also displays if the variable has been forced On (1) or Off (0).

Note: The tool tip displays a variable's current value only if logic is equal; the tool tip does not display the current value if logic is not equal.

5.5.16 To Change a BOOL Variable's State

In the ST editor, right-click the BOOL variable whose value you want to change and choose *Turn On*, *Turn Off*, or *Toggle IO*.

The state of the variable in the target Controller changes when the Controller receives the command.

5.5.17 To Force a BOOL Variable's State

In the ST editor, right-click a BOOL variable and choose *Force On*, *Force Off*, or *Toggle Force*.

The state of a forced variable remains unchanged, regardless of any actions by Controller logic.

5.5.18 To Remove the Force from a BOOL Variable

In the ST editor, right-click a BOOL variable and choose *Remove Forces*.

The state of the unforced variable is controlled by Controller logic from now on.

5.5.19 ST Statements, Functions, and Function Blocks

The following is a list of all ST statements, functions, and function blocks available in Logic Developer - PLC. InfoViewer Help indicates which firmware version of PACSystems supports the statements, functions, or function blocks.

STATEMENTS				
:= (ASSIGNMENT)	COMMENT	FUNCTION BLOCK INVOCATION	REPEAT...UNTIL	BLOCK CALL
EXIT	FUNCTION CALL	RETURN	CASE	FOR...DO
IF	WHILE...DO	VAR CONSTANT END_VAR		

ADVANCED MATH				
ACOS	ATAN	EXP	LN	SIN
SQRT_REAL	ACOS_LREAL	ATAN_LREAL	EXP_LREAL	SIN_LREAL
TANACOS_REAL	ATAN_REAL	EXP_REAL	LN_REAL	SIN_REAL
TAN_LREAL	ASIN	COS	EXPT, “”, OR ^	LOG
SQRT_DINT	TAN_REAL	ASIN_LREAL	COS_LREAL	EXPT_REAL
LOG_LREAL	SQRT_INT	ASIN_REAL	COS_REAL	EXPT_LREAL
LOG_REAL	SQRT_LREAL			

BITWISE OPERATORS				
AND	NOTE	OR	XOR	

COMMUNICATIONS				
PNIO_DEV_COMM				

CONTROL				
DO_IO	MASK_IO	SCAN_SET	SUSP_IO_INTR	SWITCH_POS
F_TRIG	R_TRIG	SUS_IO	SVC_REQ	

CONVERSIONS				
ANGLES	BCD4 to INT, REAL, or UINT	REAL to DINT, INT, LREAL, or UINT	BCD8 to DINT or REAL	TRUNC_DINT, TRUNC_INT
DINT to BCD8, DWORD, INT, LREAL, REAL, or UINT	UINT to BCD4, DINT, INT, REAL, or WORD	DWORD to DINT	WORD to INT or UINT	INT to BCD4, DINT, REAL, UINT, or WORD

DATA MOVE				
ARRAY_SIZE	ARRAY_SIZE_DIM2	MOVE_DATA_EX	SIZE_OF	ARRAY_SIZE_DIM1
COMM_REQ	MOVE_TO_FLAT			

PROGRAM FLOW
ARG_PRES

MATH FUNCTIONS				
ABS_DINT	ABS_LREAL	SCALE_DINT	SCALE_UINT	ABS_INT
ABS_REAL	SCALE_INT			

MATH OPERATORS				
+ (addition)	- (subtraction)	- (negation)	* (multiplication)	/ (division)
Mod (modulo)				

RELATIONAL OPERATORS				
= (equal)	>= (greater than or equal)	> (greater than)	<= (less than or equal)	< (less than)
<>, != (not equal)				

TIMER FUNCTION BLOCKS		
TOF	TON	TP

5.5.20 C Blocks

A C block is an independent section of executable code written in the C programming language that is downloaded to and executed on the target Controller. C blocks are created externally using Emerson’s C Programming Toolkit and then imported into a project. A C block compiled for PACSystems has a *.gefelf* extension. A C block compiled for Series 90-70 or Series 90-30 has a *.exe* extension. C Blocks can be called as a subroutine from another block (LD, ST, FBD, or IL) but cannot call another block.

Note: There are four different extensions for C block depending on the target controller. For more information on developing C Blocks, refer to *C Programmer’s Toolkit for Series 90 Controllers* (GFK-0646) and *PACSystems C Toolkit User’s Guide* (GFK-2259). More information on C Block extensions is also available in PME Help.

5.5.20.1 Working with C Blocks

5.5.20.1.1 To Import C Blocks

1. In the Project tab of the Navigator, right-click the Program Blocks folder and choose *Add C block*.

The Open dialog box appears.

2. Browse to the *.exe* or *.gefelf* file you want to import and click *Open*.

The selected file is added to the Program Blocks folder with the same name as the *.exe* or *.gefelf* file.

Note: You can then move the C Block to a user-defined folder

5.5.20.1.2 To Set a C Block’s Parameters

Note: This procedure applies only to C blocks that were written to require parameters for use of PACSystems or Series 90-70 targets.

In the Project tab of the Navigator, right-click the C block and choose Properties.

1. The Inspector displays the block’s properties.
2. In the Inspector, select the Parameters property and click. The Parameters dialog box appears.

3. Type a name and description for each required input and output parameter. For information on the required parameters, refer to the written documentation for the C block. The names you enter will display in the CALL instruction that calls the C block. The names and descriptions will display in a tooltip when you hover the mouse pointer over the CALL instruction.

5.5.20.2 C Programs

Note: C programs are supported only on Series 90-70 CPUs, firmware release 6.00 and later.

A C program is an independent section of executable code, written in C language, that is downloaded to and executed on the associated target Controller. To develop a C program, use Emerson's *C Programmer's Toolkit for Series 90 Controllers User's Manual* (GFK-0646). You then import it into a project.

A C program has access to all the % reference tables of the Controller except for the _MAIN LD block's %P memory and the %L memory of any other LD block. A C program can also call any of the numerous Controller-embedded functions that are included in the C Programmer's Toolkit. A C program cannot be called as a subroutine. Execution is controlled only through scheduling. A C program cannot call a block as a subroutine. By setting the parameters of a C program, you enable it to access memory directly. When a C program begins to execute, it reads the data for all the parameters and makes a copy of the data. If the C program's execution is interrupted or time-sliced over multiple scans, the C program, when it resumes execution, uses the copy of the data that it made when it began to execute. C programs can coexist with a main program on a Series 90-70 Controller.

5.5.21 Working with C Programs

5.5.21.1 Setting a C Program's Parameters

Note: You must provide a list of all the input and output parameters that the C program requires. The main program does not use parameters. In the Project tab of the Navigator, right-click a C program and choose Properties.

The Inspector displays the C program's properties.

1. In the Inspector, select the Parameters property and click the ellipsis button. The Parameters dialog box appears.
2. On the Input and Output tabs, enter up to 8 input and up to 8 output parameters.

Each parameter has its own row on the tab. For each parameter, double-click the following cells and enter the required data:

- **Name:** The parameter's name.
- **Type:** The parameter's data type.
- **Length:** The length of the input or output reference.
- **Variable:** The first data item associated with the parameter.
- **Description:** (Optional.) The parameter's description.

Section 6: LD Diagnostic Logic Blocks

(PACSystems RX3i firmware version 5.60 and later.) A PACSystems RX3i target contains the Diagnostic Logic Blocks folder, which contains two empty subfolders: the Active Blocks folder and the Inactive Blocks folder.

Active Blocks folder

The Active Blocks Folder can contain the following:

- One or more active LD DLBs.
- Zero through three Cam profiles associated with an LD DLB.
- One or more user-defined folders.

Inactive Blocks Folder

The Inactive Blocks Folder can contain the following:

- One or more inactive LD DLBs.
- One or more user-defined folders.

Active LD Diagnostic Logic Blocks

An active LD Diagnostic Logic Block (DLB) is a named section of ladder logic that is compiled and downloaded to the Controller.

Cam profiles associated with an active LD DLB are stored in the Cam profiles folder and are also downloaded to the Controller.

The contents of an active LD DLB are edited with the LD editor. Immediately before an upload from the Controller is performed, all active LD DLBs and their associated Cam profiles are moved to the Inactive Blocks folder. This causes all active LD DLBs to become inactive LD DLBs prior to the upload, because the variables being used by the active LD DLBs may not exist in your project after the upload completes.

After an upload from the Controller, you can activate an LD DLB. Do this in the Navigator by right-clicking an inactive LD DLB and choosing *Activate*.

Every LD DLB, whether active or inactive, has an associated Published Variable Table (PVT) that contains the local published variables of the LD DLB. If an LD DLB has no local variables, then its PVT is empty.

6.1.1 To Create an Active LD Diagnostic Logic Block

1. In the Project tab of the Navigator, expand the PACSystems RX3i target where you want to create the active LD Diagnostic Logic Block (DLB), expand the Diagnostic Logic Blocks folder, and then expand the Active Blocks folder.
2. If the active LD DLB is to reside in a user-defined folder, then, if required, add the user-defined folder.
3. Right-click the folder where you want to create the active LD DLB, point to *New*, and then choose *Block*. The active LD DLB is created and its unique default name is highlighted. Each active LD DLB contains a Cam Profiles folder, where you can add, edit, or import from one through three Cam profiles.
4. (Optional) Rename the new active LD DLB. All LD DLBs in the parent folder appear in alphabetical order.

Section 7: PROFINET Support

(PACSystems RXi; PACSystems RX3i with firmware version 7.00 and later.)

Hardware

PACSystems PROFINET Controllers (PNCs) support the following PROFINET devices:

A PNC is connected by Ethernet to PROFINET devices, each of which has its own modules and possibly submodules.

Table 4: PROFINET-Enabled Devices

VersaMax	VersaMax IP	VersaPoint
RESti	PAC8000	General Electric AF6
Third-Party		

Software

In Machine Edition, the PROFINET devices connected to a PNC are shown as nodes under the PNC in the Hardware Configuration of a PACSystems RXi or PACSystems RX3i target in a Machine Edition project. The illustration on the next page highlights the target names and the PNCs.

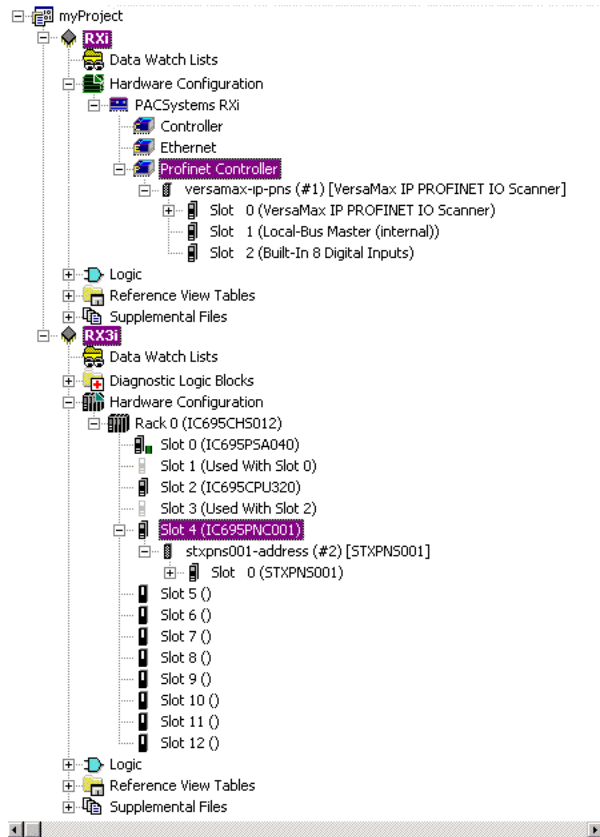
Configuration Data

The data required to configure the PROFINET devices connected to the PROFINET Controller is contained in GSDML files provided by the vendors in accordance with PROFINET standards.

Redundancy

PACSystems RX3i Controllers support Hot Standby CPU redundancy with PROFINET I/O. In that setup, two RX3i Controllers are capable of controlling the same set of PROFINET devices. (For the minimum PACSystems RX3i firmware version requirement, refer to online help.)

Figure 40: Redundant Configuration



7.1 Ethernet Global Data (EGD)

Ethernet Global Data is a mechanism that enables one CPU, referred to as a *producer*, to share a portion of its internal memory with one or more other CPUs, referred to as *consumers*, at a scheduled periodic rate. Such a snapshot of internal memory, mediated by an Ethernet interface, is referred to as an exchange. An exchange is identified by a unique combination of three identifiers

- The Producer ID (the producer’s IP address)
- The Exchange ID (the exchange’s identifier)
- The Adapter Name (the Ethernet interface identifier)

7.1.1 Exchanges vs. Pages

Some EGD tools group multiple produced exchanges into a page, and some EGD tools can consume such pages. Logic Developer – PLC; however, does not consume or produce pages. It consumes and produces exchanges.

8. When you add a consumed exchange in Logic Developer - PLC, if an EGD tool has published produced pages that contain multiple exchanges to the EGD Configuration Server, the produced exchanges are extracted from the pages and presented for your selection as exchanges.
9. Logic Developer - PLC does not group multiple produced exchanges into a produced page. EGD tools that consume pages see Logic Developer - PLC produced exchanges as pages with only one exchange each.

7.1.2 Integration with the EGD Configuration Server

Logic Developer - PLC is integrated with the EGD Configuration Server. The EGD Configuration Server is a central repository of EGD configuration information used to facilitate the sharing of information between EGD tools in order to assist with the configuration of EGD.

After configuring a produced exchange on any EGD tool (such as the EGD component in Logic Developer - PLC), you can publish the configuration of the produced exchanges to the EGD Configuration Server. After this, Logic Developer - PLC, when connected to the server, can obtain the produced exchange configuration from the server and automatically create a consumed exchange that matches the produced exchange exactly. At this point, the only thing left to do is to map the variables of the exchange to memory on the EGD consumer, or to replace some or all of the unmapped variables with mapped variables that already exist on the EGD consumer. The EGD Configuration Server provides extra EGD validation that is unavailable without the server.

The EGD Configuration Server supports EGD Signatures. When a PACSystems RX7i or PACSystems RX3i is configured to use signatures, then every scan, it compares the signature of every consumed exchange with the signature of its corresponding produced exchange to determine whether the configuration of the consumed exchange matches the configuration of the produced exchange; if not, the exchange is not consumed. The EGD Configuration Server tracks which produced exchanges published on the server are consumed by EGD devices connected to the server. You can obtain an Unconsumed Data report. Other reports compare the produced or consumed exchange information on your computer with that on the server. The EGD Configuration Server is not the authoritative source of the EGD configuration. It does not store past versions of EGD configurations. It contains only the latest snapshots of the EGD configurations published to it. The EGD tools are the authoritative source, while the server enables the tools to share information. It is possible to delete the server contents completely and restore them again by using the tools in a two-step process: publish the authoritative information from all tools to the server, and have each tool read the information it needs from the server and confirm to the server that it has received it.

7.1.3 Integration with the EGD Management Tool

Logic Developer - PLC is integrated with the EGD Management Tool (EMT). You can open the EMT from within Logic Developer - PLC and vice-versa. The EMT provides a high-level view of the EGD system published on the EGD Configuration Server, including a graphical display of individual EGD systems.

The EMT provides validation of the EGD configuration. The EMT enables the EGD configuration of devices by launching the appropriate EGD Configuration Tool from the graphical display.

7.1.4 Logic Developer – PLC implementation of EGD: the EGD Component

The first step in using Ethernet Global Data (EGD) in a PACSystems Controller or ENIU target consists in adding the EGD component. In a Controller target, you can do so even if the CPU presently configured in the target does not support EGD, as long as the target belongs to one of the Controller families that support EGD. This flexibility enables you to start configuring EGD even if the hardware specifications of your system have not been determined.

7.1.4.1 To Add the EGD Component

In the Project tab of the Navigator, right-click a PACSystems Controller target that belongs to one of the Controller or ENIU family types that support Ethernet Global Data (EGD), point to *Add Component*, and choose *Ethernet Global Data*.

The Ethernet Global Data folder is added to the target. The folder contains two empty folders: Consumed Exchanges and Produced Exchanges.

Note: For Ethernet modules that support EGD uploads, the EGD component is automatically added when you upload the Hardware Configuration and EGD configuration. The EGD component is also automatically added to a target when you import a folder that contains an Ethernet Global Data (EGD) configuration or import a Hardware Configuration that contains an EGD configuration. In such cases, the Consumed Exchanges and Produced Exchanges folders are automatically populated.

7.1.4.2 To Install the EGD Configuration Server

1. Insert the Machine Edition 8.50 install disk into the computer on which you want to install the EGD Configuration Server. The computer must be connected to an Ethernet network. It may be local or remote.
2. On the Machine Edition 8.50 install disk, navigate to the Install folder.
3. Double-click the EgdCfgServer Setup.msi file.

The EGD Configuration Server is installed on the computer.

Note: The client library required for your copy of Logic Developer - PLC to communicate with the EGD Configuration Server is automatically installed as part of the installation of Logic Developer - PLC.

7.1.4.3 To Install the EGD Management Tool (EMT) on your Computer

1. Insert the Machine Edition 8.50 install disk into your computer.
2. On the Machine Edition 8.50 install disk, navigate to the Install folder.
3. Double-click the EgdManagementTool Setup.msi file. The EGD Management Tool is installed on your computer.

7.1.4.4 To Configure Communications with the EGD Configuration Server

1. In the Options tab of the Navigator, expand the Machine Edition folder.

2. Right-click the EGD page and choose *Properties*. The Inspector displays the EGD options.
3. In the Inspector, configure the Host Name option. For help on this or any option, select the option and look up the help that automatically appears in the Companion. To open the Companion, press SHIFT+F11.
4. (Optional) Configure the Local Server Cache Path and Timeout options.

7.1.4.5 To Configure a Logic Developer – PLC Target to use the EGD Configuration Server

1. In the Project tab of the Navigator, expand the target.
2. Right-click the Ethernet Global Data folder and choose *Properties*. The Inspector displays the folder's properties.
3. Ensure that the Use Configuration Server property is set to *True* and configure the properties below it.

7.1.4.6 To Add a New Produced Exchange and Configure it

1. In the Project tab of the Navigator, expand the target's Ethernet Global Data folder.
2. Right-click the Produced Exchanges folder and choose *New*. A new produced exchange appears with a default name. The new produced exchange is invalid.
3. (Optional.) Enter a name that is more meaningful than the default name.
4. Right-click the produced exchange and choose *Properties*.
The Inspector displays the exchange's properties (Figure 41).
5. In the Inspector, configure the Destination property and other properties. The new produced exchange is now valid.
6. In the Project tab of the Navigator, right-click the produced exchange and choose *Configure*.

Figure 41: EGD Variable Editor

Offset (Byte.Bit)	Variable	Ref Address
Status		%I00113

7. In the editor, use the buttons to add, insert, or delete rows.
8. For each row, define the Variable, Ref Address, Length, and/or Description parameters. When you have configured the target to use the EGD Configuration Server, you must specify a Variable for each row; you cannot use memory ranges that have no variables mapped to them. The Ignore parameter is available only for the Status. The Type parameter is read-only. For help on the editor, click inside the grey space at the top and press F1.

7.1.4.7 To Publish a Target's Produced Exchanges to the EGD Configuration Server by using the Validate Method

1. In the Project tab of the Navigator, right-click the target and choose *Set as active target*.
If the option is unavailable, the target has already been set as the active target.
2. Right-click the target and choose *Validate*.
Any errors found in the target's EGD configuration, Hardware Configuration, or logic are listed in the Build tab of the Feedback Zone.
3. Correct any errors found and repeat step 2 of this procedure.
4. When the Build tab of the Feedback Zone displays no errors, click the Messages tab of the Feedback Zone. If the message "EGD Produced Data published" appears, the target's produced exchanges have been updated (published) from your computer to the EGD Configuration Server.

7.1.4.8 To Publish a Target's Produced Exchanges to the EGD Configuration Server by using the Bind and Build Method

1. In the Project tab of the Navigator, right-click the target and choose *Set as active target*.
If the option is unavailable, the target has already been set as the active target.
2. Expand the target.
3. Right-click the Ethernet Global Data folder and choose *Bind and Build*. Any errors found in the target's EGD configuration are listed in the Messages tab of the Feedback Zone. Some of these errors may pertain to consumed exchanges. If you want context-sensitive help on the errors, we recommend that you use the Validation method instead of the Bind and Build method.
4. If errors are found pertaining to produced exchanges, correct them and repeat step 3 of this procedure. The message "EGD Produced Data published" appears when the target's produced exchanges have been updated (published) from your computer to the EGD Configuration Server.

If an error is found pertaining to a consumed exchange, you typically need to synchronize the consumed exchange.

7.1.4.9 To Synchronize a Consumed Exchange on your Computer with the Corresponding Produced Exchange Published on the EGD Configuration Server

1. If any of the following conditions is true, validate the target that contains the producer. (See Section 3.4.1, *Validating a Target*.)
 - You have uploaded the Hardware Configuration and EGD configuration from the producer Controller to your computer
 - You have converted the target that contains the producer
You have deleted the Ethernet Global Folder from the target that contains the producer and added it again.
2. Validating the target that contains the producer updates the EGD Configuration Server with the produced exchange that this consumed exchange consumes.
3. In the Project tab of the Navigator, expand the target.
4. Expand the Ethernet Global Data folder and then expand the Consumed Exchanges folder.
5. If you have changed the Local Producer ID property of the producer that the consumed exchange consumes from, right-click the consumed exchange and choose *Properties*. Then, in the Inspector, set the Producer ID property on the consumed exchange to the same value as the new Local Producer ID.
6. In the Project tab of the Navigator, right-click the consumed exchange and choose *Synchronize to Server*.

Section 8: PACMotion

(PACSystems RX3i with firmware version 5.60 or later.) The PACMotion Multi-axis Motion controller (PMM) is a high performance, easy-to-use servo motion control module that is closely integrated with the PACSystems RX3i CPU logic solving and communications functions. This versatile motion controller combines the benefits of highly integrated motion and machine logic with the performance, flexibility, and scalability required for advanced machine automation. The open programming environment simplifies motion and machine logic synchronization, and enables real-time performance required for high-speed motion applications.

Expanding the PACMotion node reveals the following PACMotion components used with a PACMotion Digital Motion Control Module (IC695PMM335):

- Cam Profile Library: Contains all of the PACMotion Cam profiles for this target, which cannot be downloaded to a PACMotion module.
- Active Profiles: Contains all of the PACMotion Cam profile aliases for this target, which can be downloaded to a PACMotion module.
- Data Logging Windows: Contains all of the Data Logging Windows for this target. Each window is used to view and print graphs of the values of various IC695PMM335 parameters over time.

PACMotion function blocks and instructions controlling or interacting with a PMM module are supported in LD, FBD, and ST logic. You can drag them into logic from the Toolchest LD Instructions drawer (or FBD Instructions drawer) PACMotion folder.

The following can be used with PACMotion function blocks only:

- Enumerated data types: An enumerated variable stores a value belonging to an enumeration, list, or set of possible values. The value of an enumerated variable can be selected from a drop-down list in the Inspector.
- Reference ID variables (RIVs): A variable of a reference ID data type is used to identify something on which a PACMotion function block instance is used to perform an operation.
- This chapter outlines basic procedures that will get the user started in Logic Developer – PLC to work with PACMotion.

8.1 To Locate a Target's PACMotion Node

1. Expand the target.
2. By default, the PACMotion node does not appear in the Navigator. You must add the PACMotion component.

8.2 To Add the PACMotion component to a PACSystems Rx3i Target

In the Project tab of the Navigator, right-click the target, point to *Add Component*, and choose PACMotion. The PACMotion node is added to the target.

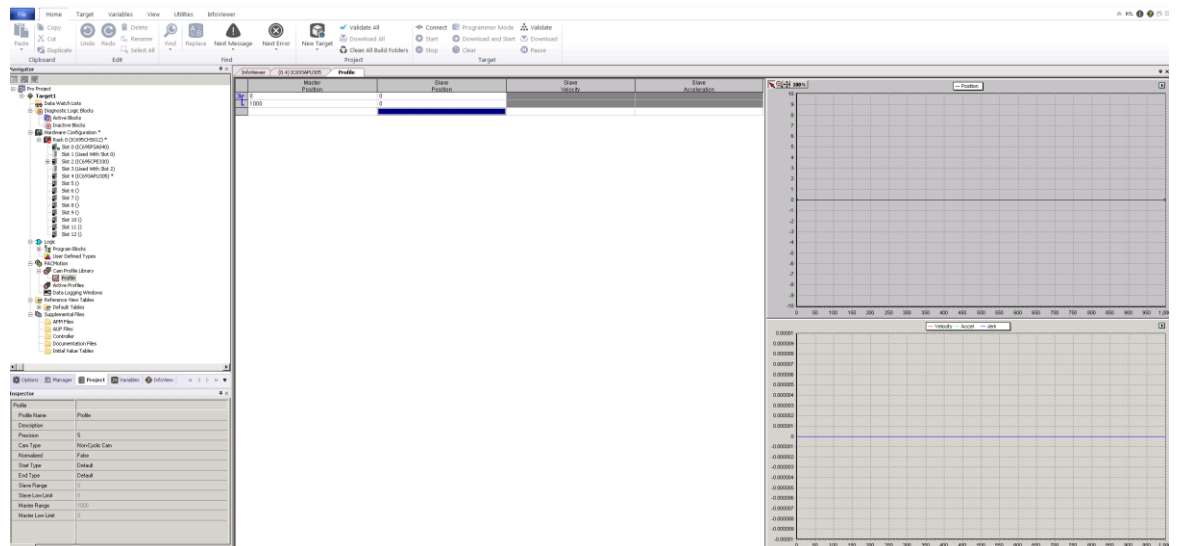
8.3 CAM Editor

The Cam editor is an accessory for PACSystems RX3i PACMotion programming that enables you to create, edit, and manage Cam profiles. Each Cam profile is a user-defined curve that specifies the response of a slave servo to a master position index. Cam profiles are referenced by name and grouped into the Cam Profile Library and the Active Profiles folder. Each active profile is intended for use on a specific PACMotion module. The hardware components are specified in the Hardware Configuration (HWC) of the parent target. You can reuse a Cam profile by right-clicking it and choosing Copy Profile.

8.4 Working with the CAM Editor

You can adjust the curves of your CAM profile to suit the specific needs of your project. With the CAM editor, you create profiles by defining points on a master/slave position curve. Groups of adjoining points are allocated to sectors. Each sector is assigned a polynomial curve fit order (1,2,3,5) that specifies how the curve will be interpolated between points.

Figure 42: CAM Editor



8.5 To Create a CAM Profile

1. In the Project tab of the Navigator expand the PACMotion node.
2. Right-click CAM Profile Library and choose New Profile.

A new CAM profile with a default name is added to your project.

8.6 Data Logging

8.6.1 To Generate .dlog Files

1. In logic, create an instance of the MC_DL_CONFIGURE PACMotion function block. To specify which parameters you want to log in the .dlog file, edit the properties of the variable (of the DATA_LOG_PARAM_CONFIG data type) that is assigned to the ParameterConfig input. When the instance executes, data logging is configured.
2. In logic, create an instance of the MC_DL_ACTIVATE PACMotion function block. When the Enable input to the instance is set to On, data logging occurs.
3. In logic, create an instance of the MC_DL_GET PACMotion function block.
When the instance executes, it writes the data logged to the *.dlog file specified by the DataLogFile

8.7 To Add a Data Logging Window (DLW)

1. In the Project tab of the Navigator, locate the Data Logging Windows node, right-click it, and choose *Create New*.
2. If you are online to the PACSystems RX3i, the Create dialog box appears.

Select one of the following:

- Controller: The DLW is a .dlog file stored on the PACSystems RX3i; or
- PC: The DLW is a .dlw or .csv file stored on your computer.

If you Selected Controller

If You Selected PC or are Offline

The Controller File Explorer window appears and displays the .dlog files found on the PACSystems RX3i.	The Select DLW to Load dialog box appears.
--	--

3. Do as follows, depending on which window appeared.

In the Controller File Explorer window

In the Select DLW to Load dialog box

Select a .dlog and click Open	Navigate to a folder on your computer, select a .dlw or .csv file, and click Open
-------------------------------	---

A Data Logging Window (DLW) appears, in alphabetical order, under the Data Logging Windows node with a default name that begins with “Sessions.” It contains the following nodes.

Views:

4. If you selected a .dlw file, this contains one or more views.
5. If you selected a .dlog or .csv file, there are no views. You need to add one.

Data Sources, whose child node contains the data snapshot contained in the .dlog, .dlw, or .csv file. If the .csv file was exported from a data source, it is a complete snapshot. If it was exported from a view, it

contains only the data required for the traces configured to appear in that view.

Section 9: Motion Programming

Logic Developer - PLC supports motion programming for the DSM324i and Motion Mate DSM314 motion control modules. High performance, easy-to-use, these multi-axis motion control modules are highly integrated with the PACSystems RX3i and the Series 90-30 Controller logic solving and communication functions.

Both the DSM324i and the DSM314 support 10 motion blocks, 40 subroutines, and a maximum total of 1000 motion program statements. Logic Developer - PLC, making motion programming possible, supports the following motion editors:

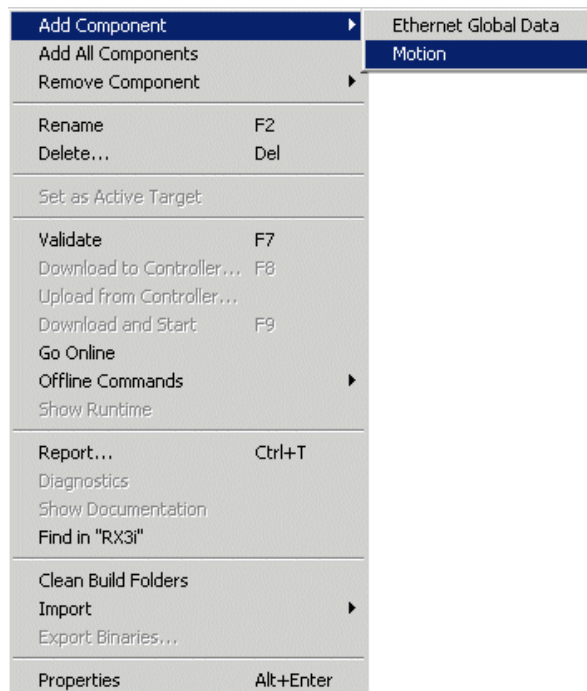
- Motion editor
- Local Logic editor
- Cam editor

This chapter outlines basic procedures that will get you started with Logic Developer – PLC to create motion programs with these motion editors.

9.1 To Add a Motion Component to a Target

In the Project tab of the Navigator, right-click the target, point to *Add Component*, and then choose *Motion* (Figure 43). A Motion Program node is added to your project. Included are empty Motion Blocks, Local Logic, CAM Profiles and CAM Blocks folders.

Figure 43: Motion Component



9.2 Motion Editor

Logic Developer - PLC includes a Motion editor, which enables you to create Motion blocks for the DSM324i and the DSM314. This text-based editor is configurable as to its appearance and behavior. Comments and white space are not considered to be Motion block statements. The Motion block programming syntax is different from Local Logic syntax.

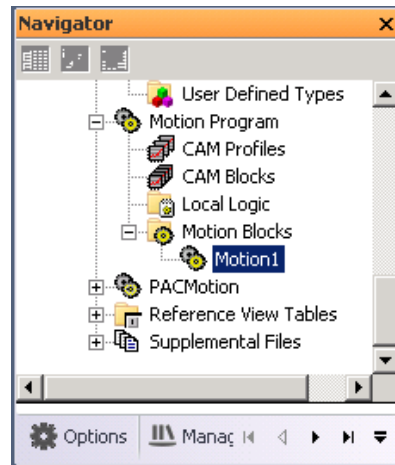
9.2.1 To Add a Motion Block

1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Motion Blocks and choose *New*.

An empty Motion block with a default name is added to your project (Figure 44).

3. Rename the block as desired.

Figure 44: To Add a Motion Block



9.2.2 To Open a Motion Block for Editing

1. In the Project tab of the Navigator, expand the Motion Program.
2. Expand Motion Blocks and double-click the Motion block you want to open.

The block opens for editing in the Motion editor.

9.2.3 Working with the Motion Editor

9.2.3.1 To Insert a Command

1. In the Motion Editor, right-click and choose Insert Keyword. A smart list appears showing all available motion commands.
2. Select the appropriate command from the smart list and press *enter*. The command is placed in the motion editor.

Figure 45: Motion Editor

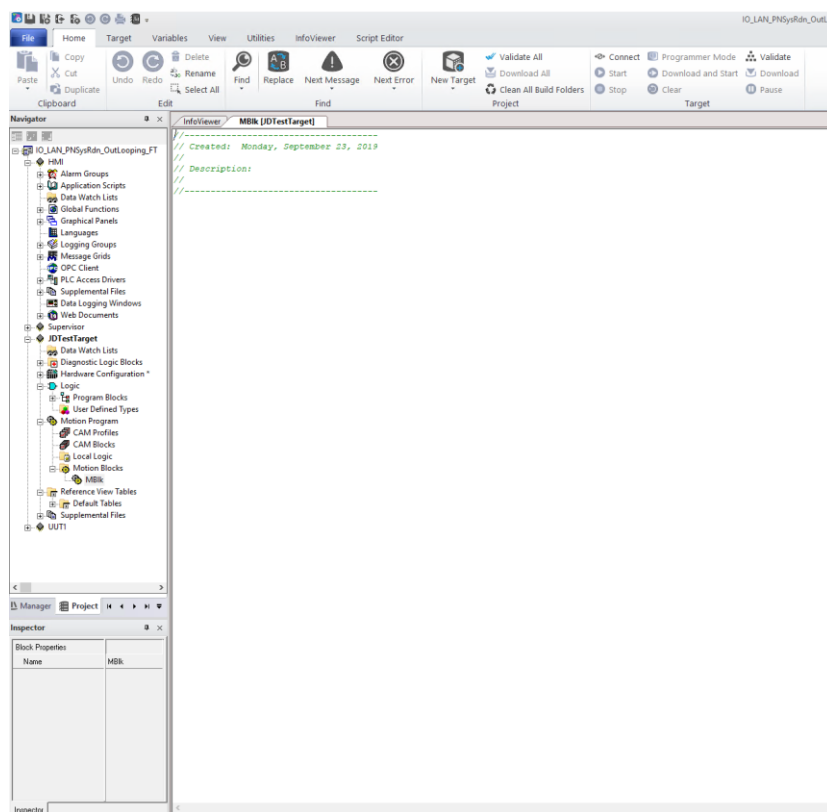


Table 5: Motion Commands

ACCEL	The ACCEL statement sets the axis acceleration for subsequent moves and remains in effect in a given block unless changed. Note: If a move instruction is executed before ACCEL, the tag Acceleration is used.
BLOCK NUMBER	Block numbers can be used as the destination of JUMP commands. Block numbers must be unique and can be between 1 and 65535.
CAM	The CAM statement starts CAM motion and specifies exit conditions.
CAM-LOAD	CAM-LOAD loads a parameter register with the starting location for a CAM slave axis.
CAM-PHASE	CAM-PHASE sets the phase for CAM comm
CALL	The CALL command executes another block as a subroutine.
CMOVE	The CMOVE command programs a continuous move using the specified position and acceleration mode.
DWELL	DWELL causes motion to cease for a specified time period before processing the next command.
ENDPROG	The ENDPROG statement terminates a Motion program definition.
ENDSUB	The ENDSUB statement terminates a Motion subroutine definition
JUMP	Jump to a block number or a sync block within the current program or subroutine. The jump may be unconditional or conditional based on the status of a CTL bit.
LOAD	Initializes or changes a parameter data register with a 32-bit twos-complement integer value.
PMOVE	The PMOVE Command programs a positioning move using the specified position and accelerator mode.
PROGRAM	The PROGRAM statement is the first statement in a motion program. The program statement identifies the program number (valid range: 1

	through 10) and the axis configuration. Program definitions cannot nest.
SUBROUTINE	The SUBROUTINE statement is the first statement in a motion subroutine. The subroutine statement identifies the subroutine number.
SYNC BLOCK	A sync block is a special case of a block number. A sync block can be used only in multi-axis programs.
VELOC	Set the process VELOCITY used by subsequent motion program move commands and remains in effect until changed by another VELOC statement.

9.3 Local Logic

A Local Logic block runs synchronously with the Motion block, but is independent of the Controller's CPU scan. This enables the DSM324i or DSM314 to interact much more quickly with motion I/O signals on its faceplate connectors than would be possible if the logic for the signals were handled in the _MAIN program running on the Controller.

Local Logic language uses free-form, text-based circuits and contains basic mathematical and logical constructs. The Local Logic syntax enables you to assign a variety of logic tasks to your motion programs while working in conjunction with Controller Logic programs and motion blocks to yield a flexible programming environment. Because it uses straightforward, understandable syntax, it is easy to gain proficiency with this editor.

The Local Logic programming language supports assignments, conditional statements, arithmetic, logical, and relational statements. Local Logic provides you with access to motion Controller data, parameters using a fixed set of variables, control bits, and status bits:

- Parameter data - accessible from Local Logic host Controller and motion blocks. The parameter data are similar to variables in a program.
- CTL bits - enable the Local Logic block or host Controller to signal the motion block to start an event.
- Motion block numbers - the current block number can be used within the Local Logic block or host Controller to make an action occur only during a specific motion programming section.

9.3.1 To Create a Local Logic Block

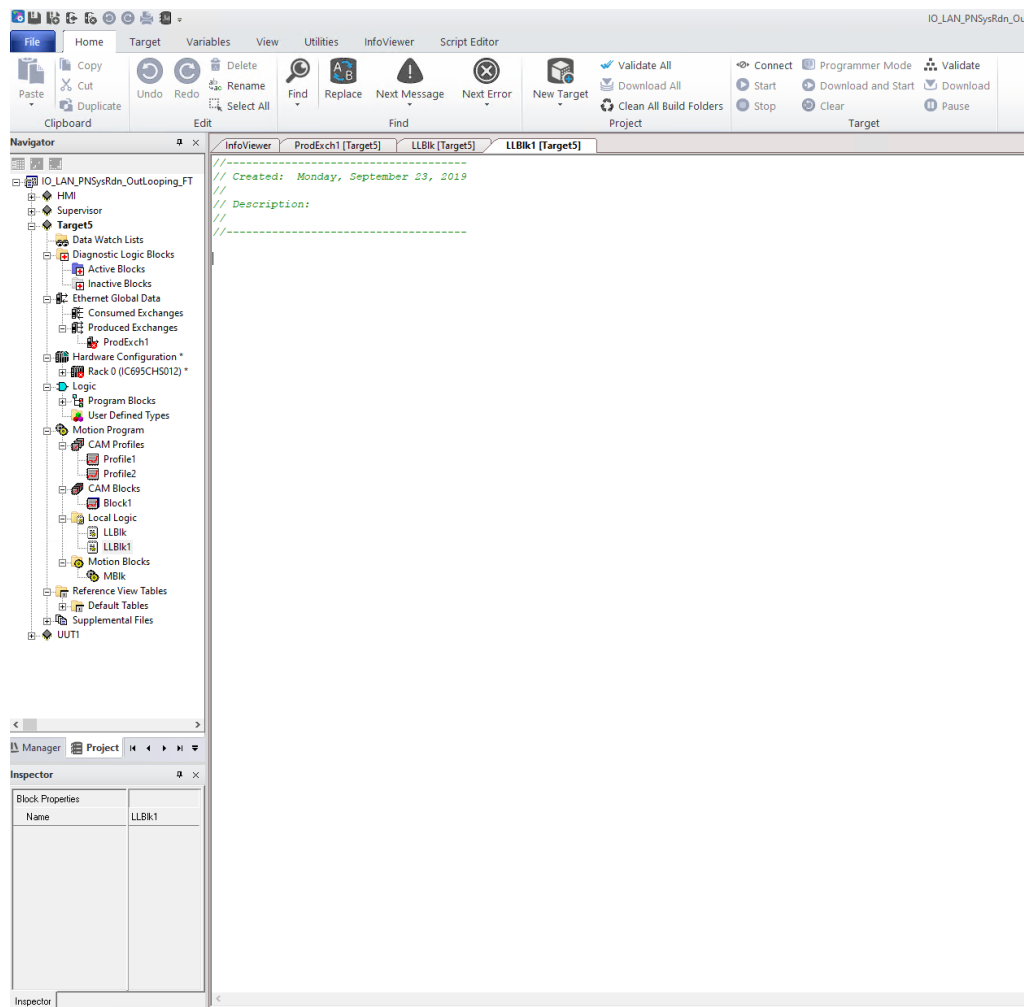
1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Local Logic and choose *New*.
A new Local Logic block with a default name is created.
3. Rename the block as required.

9.3.2 To Open a Local Logic Block for Editing

1. In the Project tab of the Navigator, expand the Motion Program and double-click the Local Logic block. The Local Logic block opens for editing in the Local Logic editor.

9.3.2.1 Working with the Local Logic Editor

Figure 46: Local Logic Editor



9.3.2.2 To Insert a Local Logic Command

1. In the Local Logic editor, right-click and choose *Insert Keyword*. A smart list displays all available Local Logic commands.
2. Select the desired command in the smart list and press ENTER
The command is inserted.

Note: You can drag variables from the Local Logic Variable Table (LLVT) to the Local Logic editor.

9.3.3 Local Logic Variables

Local Logic is designed to complement a Controller’s logic and mathematical capabilities. Solving small Local Logic and mathematical sets requires a tight synchronization with the controlled motion.

Logic Developer - PLC includes a table containing Local Logic variables, the Local Logic Variable Table (LLVT), that you can drag into your Local Logic blocks. The LLVT has several tabs that organize the variables by category.

9.3.3.1 To View the LLVT

1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Local Logic and choose *Local Logic Variable Table*.
The “Which LLVT do you want?” help topic appears.
3. Select Motion Mate DSM314 or DSM324i.
The LLVT appears in the InfoViewer, displaying variables or data on each tab:

Table 6: LLVT Variables

AXIS 1	Variables specific to axis 1
AXIS 2	Variables specific to axis 2
AXIS 3	Variables specific to axis 3
AXIS 4	Variables specific to axis 4
GLOBAL	Global data such as module status code
CTL BITS	DSM general Control/Status bits
PARAMETER REGISTERS	DSM parameter data

The table has six columns:

NAME	Contains the variable name that is to be used within a Local Logic block
TYPE	The data type for this variable. For example, 32-bit means that this variable is a 32-bit variable.
GROUP	The group this variable is placed in. For example, Faceplate I/O means that this variable refers to a point on the module faceplate.
DESCRIPTION	This column contains a textual description of the variable. When you hover the mouse pointer over the description, a tool tip displays the description.
R	This column indicates if the variable can be read by a Local Logic block .
W	This column indicates if the variable can be written by a Local Logic block.

9.3.4 To Insert a Local Logic Variable

With Local Logic, you can execute basic logic and mathematical functions on the DSM324i or Motion Mate DSM314 module. Commands use upper case characters only and are case sensitive.

1. In the Local Logic editor right-click and choose *Insert Variable*.
A smart list appears prompting you to choose a Local logic variable name.
2. Select a variable in the list and press enter.
The variable is inserted in your Local Logic.

9.3.4.1 Local Logic Commands and Operators

With Local Logic, you can execute basic logic and mathematical functions on the DSM324i or Motion Mate DSM314 module. Commands use upper case characters only and are case sensitive.

Table 7: Local Logic Commands and Operators

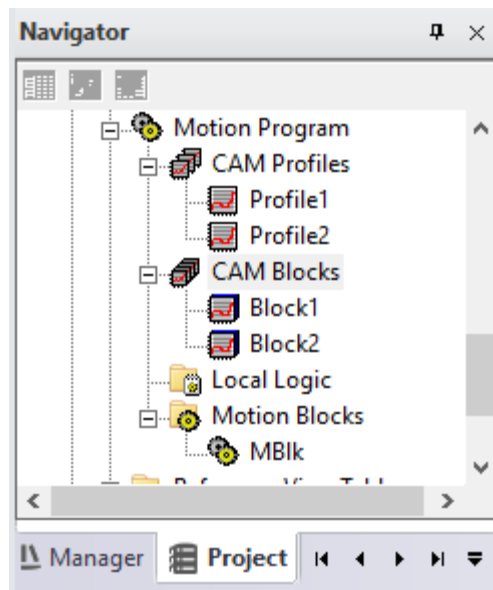
ABS	TRUE	-(minus)	<= (less than or equal to)
BWAND	FALSE	/(divide)	<> (not equal)

BWOR	IF	*(multiply)	
BWXOR	THEN	:=(assign)	
BWNOT	END_IF	>(greater than)	
ON	MOD	< (less than)	
OFF	+ (plus)	>= (greater than or equal to)	

9.3.5 CAM Editor

The Cam editor is an accessory for Logic Developer – PLC motion programming that provides a means to create, edit, and manage electronic Cam profiles. Each Cam profile is a user-defined curve that specifies the response of a slave servo to a master position index. Cam profiles are referenced by name in the parent motion program and grouped into Cam blocks. Each block is intended for download to a specific motion module via its Controller. The hardware components are specified in the Hardware Configuration (HWC) of the parent target. You can reuse a Cam profile by including it in multiple Cam blocks.

Figure 47: CAM Profiles and CAM Blocks



9.3.5.1 To Create a CAM Block

1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Cam blocks and choose *New*.
A new Cam block with a default name is created.
3. Rename the block as desired.

9.3.5.2 To Import CAM Blocks

1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Cam blocks and choose *Import from File*.
The Open dialog box appears.

Browse to the Cam block you want to import (.csv or .txt file). Click *Open*. The imported block appears in your project.

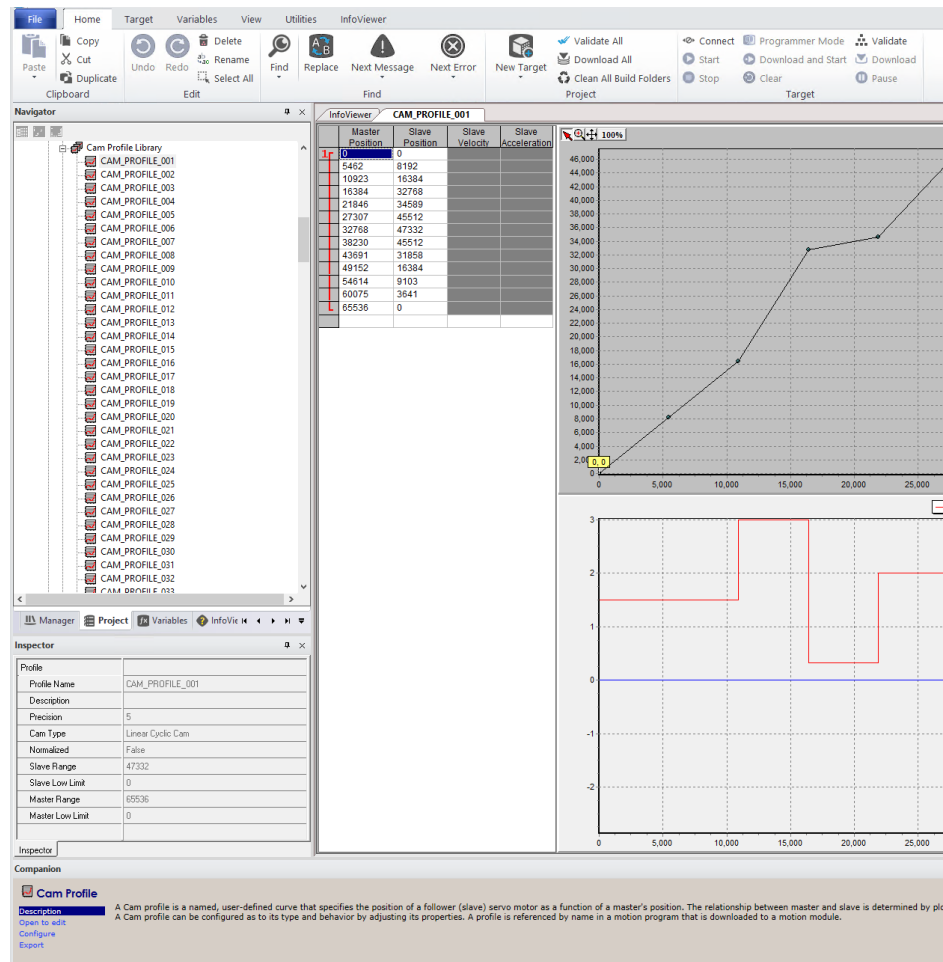
9.3.5.3 To Open a Block for Editing

1. In the Project tab of the Navigator, expand the Motion Program.
2. Expand Cam Blocks and double-click the Cam block you want to open.
The block opens for editing in the Motion editor.

9.3.5.4 Working with the CAM Editor

You can adjust the curves of your Cam profile to suit the specific needs of your project. With the Cam editor you create profiles by defining points on a master/slave position curve. Groups of adjoining points are allocated to sectors. Each sector is assigned a polynomial curve fit order (1,2,3) that specifies how the curve will be interpolated between points.

Figure 48: Adjusting Curves of the CAM Profiles



9.3.5.5 To Configure a CAM Profile

1. In the Project tab of the Navigator, expand the Motion Program.
2. Right-click Cam Profiles and choose *New*. A new Cam profile with a default name is added to your project
3. Adjust the properties of the CAM profile in the Inspector

9.3.5.6 To Edit a CAM Profile

1. In the Project tab of the Navigator, expand the Motion Program
2. Right-click Cam Profiles and double-click a Cam profile.

A graphical representation of your profile appears in the profile editor and a numeric representation appears in the profile table.

3. Insert and move points in the profile editor or table.
4. Group points into sectors in the profile table and assign curve fit order to each sector.

9.3.5.7 To Add a CAM Block

Right-click method

1. In the Project tab of the Navigator, expand the Motion Program and expand Cam blocks.
2. Right-click a Cam block, point to *Add Alias to*, and then choose a profile.

Note: Aliases correspond to Cam profiles within Cam blocks. In order to create aliases for Cam blocks, you must have previously created Cam profiles.

Drag and Drop Method

1. In the Project tab of the Navigator, expand the Motion Program.
2. Expand Cam Profiles and expand Cam blocks.
3. Drag a Cam profile and drop it onto a Cam block. The Cam profile is added to the Cam block.

Contact Information

Home link: <http://www.Emerson.com/Industrial-Automation-Controls>

Knowledge Base: <https://www.emerson.com/Industrial-Automation-Controls/support>

Note: If the product is purchased through an Authorized Channel Partner, please contact the seller directly for any support.

Emerson reserves the right to modify or improve the designs or specifications of the products mentioned in this manual at any time without notice. Emerson does not assume responsibility for the selection, use or maintenance of any product. Responsibility for proper selection, use and maintenance of any Emerson product remains solely with the purchaser.

© 2019 Emerson. All rights reserved.

Emerson Terms and Conditions of Sale are available upon request. The Emerson logo is a trademark and service mark of Emerson Electric Co. All other marks are the property of their respective owners.

