

# SIEMENS

## SIMATIC

### Windows Automation Center RTX WinAC RTX 2005

#### Manual

Preface, Contents

---

Product Overview **1**

---

Installation **2**

---

Getting Started **3**

---

Controller  
Operations **4**

---

STEP 7 Operations  
and Components **5**

---

Tuning the  
Performance of the  
Controller **6**

---

Connecting the  
Controller to the  
SIMATIC NET OPC  
Server **7**

---

Reference  
Information **8**

---

Glossary, Index

This documentation is part of the WinAC RTX 2005 package with  
order number:  
6ES7 671-0RC04-0YA0

**Edition: 08/2005**  
A5E00486536-01

# Copyright and Safety Notification

This manual contains notices that you must observe to ensure your own personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol; notices referring only to property damage have no safety alert symbol. The notices shown below are graded according to the degree of danger.



## Danger

Indicates that death or severe personal injury will result if proper precautions are not taken.



## Warning

Indicates that death or severe personal injury may result if proper precautions are not taken.



## Caution

With a safety alert symbol indicates that minor personal injury can result if proper precautions are not taken.

## Caution

Without a safety alert symbol indicates that property damage can result if proper precautions are not taken.

## Notice

Indicates that an unintended result or situation can occur if the corresponding notice is not taken into account.

## Qualified Personnel

The device/system may only be set up and operated in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by qualified personnel. Within the context of the safety notices in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Correct Usage

Note the following:



## Caution

This device and its components may only be used for the applications described in the catalog or the technical descriptions and only in connection with devices or components from other manufacturers that have been approved or recommended by Siemens.

Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

## Trademarks

Siemens® and SIMATIC® are registered trademarks of SIEMENS AG.

STEP 7™ and S7™ are trademarks of SIEMENS AG.

Microsoft®, Windows®, Windows XP Professional®, Windows 2000®, and Internet Explorer® are registered trademarks of Microsoft Corporation.

Adobe® and Acrobat® are registered trademarks of Adobe Systems Incorporated.

RTX™ is a trademark of Ardence, Inc.

## Copyright Siemens Energy & Automation, Inc., 2005

### All rights reserved

The distribution and duplication of this document or the utilization and transmission of its contents are not permitted without express written permission. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Siemens AG  
Automation and Drives  
Postfach 4848, D-90327 Nuernberg

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

© Siemens AG 2005  
Technical data subject to change.

# Preface

## Purpose of the Documentation

This documentation provides detailed information about the Windows Automation Center with Real-Time Extensions (WinAC RTX 2005) software package that includes the following components:

- Windows Logic Controller RTX (WinLC RTX V4.2)
- Ardence RTX V6.1
- WinAC Time Synchronization V4.0
- SIMATIC NET OPC Server V6.3 (on SIMATIC NET CD)

You install WinAC RTX and the documentation from the installation CDs included with your release.

The Windows Logic Controller with Real-Time Extensions (WinLC RTX) provides the functionality of a programmable logic controller (PLC) in a real-time, PC-based environment. WinLC RTX uses the Ardence (formerly VenturCom) Real-Time Extensions (RTX) to Windows and is fully code-compatible with the SIMATIC product family. WinLC RTX is part of the WinAC family of PC-based controllers. You can use many of the SIMATIC products, such as WinCC flexible, with the WinAC PC-based controllers.

The PC-based controllers use a PROFIBUS-DP network to communicate with distributed I/O, such as an ET 200M device. They use PG/OP communications (PROFIBUS or Industrial Ethernet) for connecting to STEP 7 or other programming software on another computer.

## Prerequisites

This documentation is intended for engineers, programmers, and maintenance personnel who have a general knowledge of programmable logic controllers. Persons using this documentation also need knowledge of Windows 2000/XP operating systems and STEP 7 programming.

## Scope

This document describes the features and the operation of WinAC RTX 2005.

## Changes Compared to the Previous Version

The topic "What's New?" in the Product Overview enumerates the new features of WinAC RTX 2005.

## Location of Documentation

The WinAC RTX 2005 installation includes this documentation as both online help and a PDF online manual and also a PDF online manual for WinAC Time Synchronization V4.0. Installation of documentation is optional from the setup. If installed, the online help is accessible from the controller panel and all applicable PDF files are accessible from the **Start > Simatic > Documentation** menu command.

## Other Manuals

You can find additional information in the online help for STEP 7 and in the following documents:

- *STEP 7 - Programming with STEP 7*. This manual provides basic information on designing and programming a WinLC RTX STEP 7 user program.
- *STEP 7 - System and Standard Functions for S7-300 and S7-400*. WinLC RTX includes integrated system functions and organization blocks, which you can use when programming. This manual provides you with descriptions of the system functions, organization blocks, and loadable standard functions.
- *STEP 7 - Working with STEP 7*. This manual explains the usage and the functions of the STEP 7 automation software. This manual provides you with an overview of the procedures used to configure WinLC RTX and to develop STEP 7 user programs.
- *SIMATIC NET - Commissioning PC Stations*: This manual supports you when commissioning your SIMATIC NET PC modules in a PC Station, introduces all SIMATIC NET software tools, and helps you use them successfully (available if you install SIMATIC NET)
- *SIMATIC NET - Industrial Communication with PG/PC, Parts 1 and 2*. This manual helps you with setting up industrial communications over PROFIBUS and Industrial Ethernet communications networks (available if you install SIMATIC NET)
- *WinAC Time Synchronization*. This manual describes the configuration and operation of WinAC Time Synchronization.
- *Ardence RTX Runtime Release Notes*. These release notes include the system requirements for RTX and further information about RTX

To find the SIMATIC manuals, select the **Start > Simatic > Documentation** menu command from the Start menu of the computer where the SIMATIC software is installed. The Ardence RTX Runtime Release Notes are installed by default at C:\Program Files\Ardence\RTX\RTXruntimeReleaseNotes.pdf.

## Further Support

If you have any technical questions, please get in touch with your Siemens representative or agent.

You can find your contact person at:

<http://www.siemens.com/automation/partner>

You can find a guide to the technical documentation offered for the individual SIMATIC Products and Systems here at:

<http://www.siemens.com/simatic-tech-doku-portal>

The online catalog and order system is found under:

<http://mall.automation.siemens.com/>

For additional assistance in answering technical questions, for training on this product, or for ordering, contact your Siemens distributor or sales office.

North America and South America	Europe and Africa	Asia and Pacific region
Telephone: +1 (800) 333-7421	Telephone: +49 (0) 180 5050 222	Telephone: +86 10 64 75 75 75
Fax: +1 (423) 262-2200	Fax: +49 (0) 180 5050 223	Fax: +86 10 64 74 74 74
<a href="mailto:simatic.hotline@siemens.com">simatic.hotline@siemens.com</a>	<a href="mailto:adsupport@siemens.com">adsupport@siemens.com</a>	<a href="mailto:adsupport.asia@siemens.com">adsupport.asia@siemens.com</a>
For information about Ardence Real-Time Extensions (RTX):Internet: <a href="http://www.Ardence.com">http://www.Ardence.com</a>		

# Contents

<b>Product Overview</b> .....	<b>1</b>
Introduction to PC-Based Control .....	1
Important Information about Industrial Ethernet Support.....	1
Introduction to the Controller Panel .....	2
PC-based Control Features .....	3
Advantages of Real-time Extensions (RTX) .....	3
SIMATIC Functionality Supported by WinLC RTX .....	4
Windows Functionality Supported by WinLC RTX .....	4
What's New?.....	5
System Requirements .....	6
Windows User Privileges.....	7
Using Help .....	8
Accessing Help from the Controller Panel.....	8
Using the Table of Contents .....	9
Using the Index.....	9
Using Full-Text Search .....	9
Printing Help Topics.....	9
Changing the Language of a Help Topic.....	9
Differences between WinLC RTX and WinLC Basis .....	10
<b>Installation</b> .....	<b>11</b>
Overview of the Installation Tasks .....	11
Installing the Ardence RTX Extensions .....	12
Step 1: Install the Ardence RTX Extensions.....	12
Step 2: Verify that the Ardence RTX Extensions Are Operational .....	13
Step 3: Installing the WinAC RTX Software.....	13
Installing the WinAC RTX Software .....	14
Licensing the WinAC RTX Software .....	15
Installing the License during Installation.....	15
Installing the License at a Later Date .....	15
Transferring an Installed License.....	15
Running the WinLC RTX Controller without a License.....	15
Recovering the License in Case of a Defective Hard Drive.....	15
Uninstalling Ardence RTX or WinAC RTX .....	15
<b>Getting Started</b> .....	<b>17</b>
Understanding the Concepts .....	18
What Is a PC Station?.....	18
What Is a Communication Interface?.....	20
What Is an Index? .....	21
What Is a Submodule?.....	22
What Is an IF Slot? .....	24
Configuring Communication Cards.....	25
Designating a Communication Interface as a Submodule.....	25
Creating an INI File for an Industrial Ethernet Interface .....	27

Testing a CP 5613 Configuration.....	28
Removing a Submodule.....	29
<b>Configuring the Controller in STEP 7.....</b>	<b>30</b>
Connecting STEP 7 to the Controller.....	30
Hardware Configuration in STEP 7.....	32
Invalid Characters prior to STEP 7 V5.3 SP1.....	35
<b>Verifying the Configuration.....</b>	<b>37</b>
Example PC Station Configuration.....	37
Station Configuration Editor and WinLC Properties.....	37
STEP 7 PG/PC Interface.....	38
STEP 7 Hardware Configuration.....	38
<b>Controller Operations.....</b>	<b>39</b>
<b>Starting and Shutting Down the Controller.....</b>	<b>39</b>
Starting WinLC RTX.....	39
Shutting Down WinLC RTX.....	39
<b>Changing the Operating Mode of the Controller.....</b>	<b>40</b>
Operating Mode (RUN/STOP) and Status Indicators.....	40
Allowed and Prohibited Actions for each Operating Mode.....	41
<b>Resetting the Memory Areas: MRES Command (CPU Menu).....</b>	<b>42</b>
<b>Using the Status Indicators.....</b>	<b>43</b>
Flashing Indicators.....	44
Corrective Action If the STOP Indicator is Flashing Slowly.....	44
Corrective Action If All Status Indicators Are Flashing.....	44
<b>Using the Tuning Panel.....</b>	<b>45</b>
<b>Using the Diagnostic Buffer.....</b>	<b>47</b>
Sorting Events (upper panel).....	48
Choosing Format (lower panel).....	48
Saving the Diagnostic Buffer.....	48
Displaying Help.....	48
<b>Archiving and Restoring STEP 7 User Programs.....</b>	<b>49</b>
Creating an Archive File.....	49
Restoring an Archive File.....	49
<b>Closing the Controller Panel.....</b>	<b>50</b>
<b>WinLC RTX Operation following a Windows Blue Screen.....</b>	<b>50</b>
Considerations for SFC 22, SFC 23 and SFC 82 to 85.....	51
WinLC RTX Restart Behavior after a Blue Screen.....	51
<b>Storing Retentive Data.....</b>	<b>52</b>
Reloading Memory Areas on Startup.....	54
Using SFCs to Retain Data.....	56
Uninterruptible Power Supply (UPS).....	56
<b>Customizing and Security Options.....</b>	<b>57</b>
Selecting the Language.....	57
Selecting the Autostart Feature.....	57
Setting the Security Options.....	58
Changing the Password.....	59

---

Startup Options for the Controller .....	60
Starting the Controller at PC Boot .....	60
Setting the Restart Method .....	61
<b>STEP 7 Operations and Components.....</b>	<b>63</b>
Using STEP 7 with the Controller .....	63
Configuring the Operational Parameters for the Controller .....	63
Accessing Operational Parameters .....	63
Logic Blocks Supported by WinLC RTX .....	64
Additional S7 Blocks .....	64
S7 Communication Functions.....	65
PROFIBUS DPV1 .....	66
Communication Blocks .....	67
Organization Blocks (OBs) .....	68
OBs for the Free Scan Cycle, Cold Restart, and Warm Restart.....	69
Interrupt OBs.....	70
Considerations for Cyclic Interrupt OBs.....	71
Error OBs .....	71
System Functions (SFCs).....	73
Running Asynchronous SFCs Concurrently .....	76
SFCs That Can Cause the Scan Cycle to Vary.....	77
Notes for SFC 82, SFC 83, and SFC 84.....	77
System Function Blocks (SFBs) .....	78
System Clock and Run-Time Meter.....	79
<b>Tuning the Performance of the Controller .....</b>	<b>81</b>
Scan Cycle for a PC-Based Controller .....	81
Tasks Performed during the Scan Cycle .....	81
Methods for Managing the Performance of WinLC RTX .....	83
What Causes Jitter? .....	84
Priority Settings for Competing RTX Applications Can Cause Jitter .....	84
Priorities among the WinLC RTX Threads Can Cause Jitter.....	86
The Sleep Interval Forced by the Execution Monitor Can Cause Jitter.....	87
Adjusting the Priority of the Controller .....	88
Real-Time Subsystem Priorities .....	88
Managing the Sleep Time.....	89
Sleep Management Techniques .....	89
Adjusting the Minimum Sleep Time and Cycle Time .....	93
Using SFC 47 to Add Sleep Time in the STEP 7 User Program.....	96
Adjusting the Sleep-Monitoring Algorithm of the Execution Monitor .....	97
Example: Avoiding Jitter in the Start Time of an OB .....	103
Isochronous Mode for a Constant Bus Cycle .....	107
System Requirements for an Isochronous DP Cycle .....	107
<b>Connecting the Controller to the SIMATIC NET OPC Server.....</b>	<b>109</b>
Task Overview .....	109
Step 1: Add the OPC Server to the PC Station .....	110

Step 2: Add the OPC Server to the Hardware Configuration.....	112
Creating the STEP 7 Project.....	112
Adding the OPC Server to the Hardware Configuration .....	113
Configuring the OPC Server .....	114
Step 3: Add an S7 Connection for the OPC Server in NetPro.....	115
Configuring an OPC Server Connection in NetPro.....	115
Assigning a Local ID for the OPC Server Connection .....	117
Step 4: Download the Configuration to the Controller .....	118
Step 5: Connect the Controller to the OPC server.....	119
Creating an OPC Project .....	119
Adding a Connection (Group) for the OPC Server .....	119
Configuring the Items to be Accessed (Using Absolute Addressing) .....	121
Configuring the Items to be Accessed (Using the STEP 7 Symbol Table).....	124
<b>Reference Information .....</b>	<b>127</b>
Technical Data.....	127
Order Number .....	127
Technical Specifications .....	127
Execution Times .....	130
Troubleshooting.....	131
Relevant Information for Ardence RTX.....	131
Troubleshooting Network Problems.....	133
Improving the Performance of a DP Interface .....	134
Responding to Diagnostic Events.....	135
Cross-Module Access Errors .....	135
System Status List (SSL).....	136
Using SFC 51 to Read the SSL .....	136
SSL_ID Descriptions.....	137
<b>Glossary.....</b>	<b>Glossary-1</b>
<b>Index.....</b>	<b>Index-1</b>



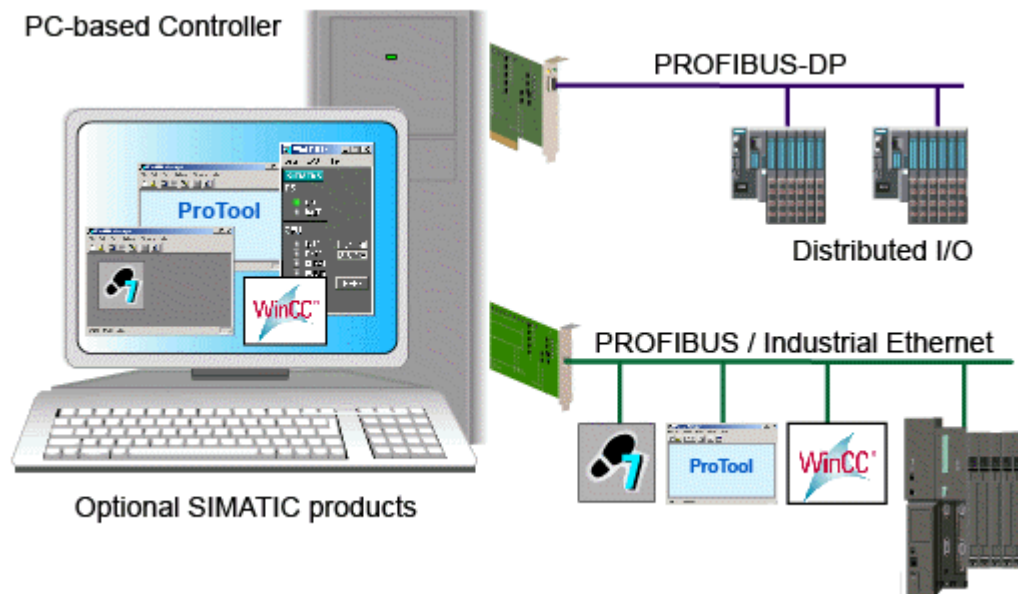
# Product Overview

## Introduction to PC-Based Control

The WinAC (Windows Automation Center) PC-based controllers execute within a standard PC and provide the same functionality as SIMATIC S7 CPUs (hardware controllers). They include WinLC Basis, WinLC RTX, and the WinAC Slot PLC. WinLC RTX is a programmable software PLC — a software application that runs on a standard computer (PC).

WinLC RTX supports multiple networks and connects to the distributed I/O, such as ET 200M, by means of DP interfaces that reside in your computer.

As part of the SIMATIC family of automation products, WinLC RTX can also communicate with STEP 7 or other SIMATIC products, such as WinCC Flexible, ProTool Pro, or other SIMATIC S7 controllers, including any of the PC-based controllers over PROFIBUS or Industrial Ethernet networks.



You can use the same programming languages, program structure and programming user interface (STEP 7) as for hardware PLCs to develop your process control solution. Programs designed for S7 controllers can run on PC-based controllers and vice versa. The PC-based controllers also include a controller panel that runs on the PC. With these capabilities, you can use WinLC RTX in a typical factory automation configuration.

## Important Information about Industrial Ethernet Support

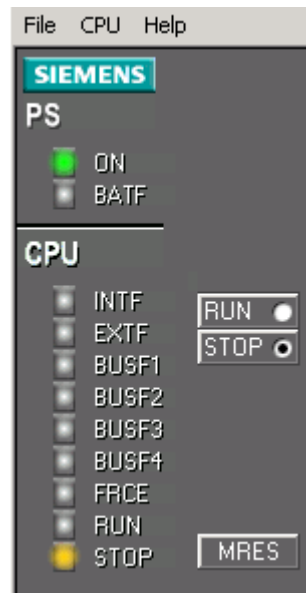
WinAC RTX 2005 does not support configuration of an Industrial Ethernet communications interface as a submodule of WinLC RTX. Although this documentation describes the capabilities of an Industrial Ethernet submodule and how to configure one, this feature is not available.


## Introduction to the Controller Panel

The controller panel corresponds to the faceplate of the SIMATIC S7 CPUs. It enables you to start or shut down the controller and to perform other operations.

The controller panel is a display window on your PC that contains the following elements for working with the controller:

- Two operating mode selector switch positions for changing the operating mode of the controller (similar to the mode selector switch on an S7 CPU front panel.)
- Status indicators for the controller
- An MRES switch position for resetting the memory areas
- Menus for controller operations



An icon  is displayed in the Windows taskbar whenever the controller is operating. When the controller is operating and the controller panel is closed, you can double-click this icon to open the controller panel.

Opening or closing the controller panel does not influence the state of the controller. The status of the operator switches and the LEDs are stored in the controller.

## PC-based Control Features

WinLC RTX is a software version of an S7 controller that adds real-time control provided by a real-time subsystem for the Windows operating system. It executes STEP 7 user programs as do other S7 controllers and allows for easy integration with STEP 7 and standard Windows applications. WinLC RTX executes in two separate environments, including processes that run in the real-time subsystem and processes that run in the Windows environment.

- The processes that run in the real-time subsystem execute the STEP 7 user program for WinLC RTX, giving process control the highest priority.
- The processes that run in the Windows environment handle other operations, such as communication and interfaces to Windows systems and applications.

## Advantages of Real-time Extensions (RTX)

WinLC RTX uses real-time extensions (RTX) to provide the following features:

- Deterministic operation ensures that response is predictable. Execution of the STEP 7 user program occurs entirely in the real-time subsystem, thus reducing "jitter."
- The control process is protected from hard disk crash and Windows system failure. WinLC RTX is notified of all Windows shutdowns (including the "blue screen") in order to programmatically shut down in an orderly fashion. You can configure Windows to reboot automatically after a system failure. This option is accessed by the Startup and Recovery button under the Advanced tab of System Properties in the Windows Control Panel.

## SIMATIC Functionality Supported by WinLC RTX

WinLC RTX provides the following features:

- Implements a substantial subset of the S7 logic blocks of SIMATIC controllers: Organization Block (OB), System Function Block (SFB), and System Function (SFC)
- Uses PROFIBUS-DP as its I/O subsystem, supporting DPV0 and DPV1 slaves (PROFIBUS DPV1 provides enhanced alarm and status reporting, in order to communicate with intelligent slave devices)
- Supports up to 4 separate PROFIBUS-DP subnets for connecting to distributed I/O
- Supports an isochronous mode, which allows WinLC RTX to operate in constant bus cycle mode to help eliminate jitter
- Uses S7 communication services, offering compatibility with SIMATIC applications such as STEP 7, WinCC, and ProTool/Pro for tasks such as programming, debugging, monitoring or visualization
- Allows peer-to-peer communications between controllers (hardware or software) on the network
- Supports the routing of S7 communications through the submodule CP cards or Industrial Ethernet card of WinLC RTX, allowing STEP 7 on one subnet to connect to an S7 station (such as an S7-400 controller) on a different subnet
- Provides ability to archive and restore control programs
- Allows you to control the operating mode of the controller and to view status information from the controller panel
- Provides a tuning panel for optimizing system performance
- Provides time synchronization as either a time master or slave
- Provides connectivity to the SIMATIC NET OPC Server, which enables OPC client applications to access process data (requires installation of SIMATIC NET, a separate product)
- Ability to exchange data with custom PC applications developed with WinAC ODK: WinAC Open Development Kit is a software package (sold separately) that supports two programming interfaces to the STEP 7 user program: CCX (Custom Code Extension) and SMX (Shared Memory Exchange.)
- Ability to develop software such as a control panel to perform controller operations and display controller status information using the Controller Management Interface (CMI) of WinAC ODK. (sold separately)
- Ability to use WinAC RTX with up to three CPU 41x-2 PCI (WinAC Slot PLCs) that are installed in your computer.

## Windows Functionality Supported by WinLC RTX

Windows Administrator privileges (ADMIN) are not required in order to operate the WinLC RTX controller. With Power User, User or even with Guest privileges, you can perform operational tasks, such as changing the operating mode of the controller from RUN to STOP, modifying the sleep time, or restoring an archived control program.

If you configured WinLC RTX to start at PC boot (and if the controller was consequently started by rebooting the computer), one user can log off and another user can log on without affecting the operation of the controller.

**Note:** Although WinLC RTX supports logging off and logging on as a Windows user, the Windows XP "Switch User" function is **not** supported by WinLC RTX.

## What's New?

The following features are new for WinAC RTX 2005:

### Improved STEP 7 User Program Performance

STEP 7 user programs running on WinLC RTX V4.2 have execution times up to twelve times as fast as the same program running on WinLC RTX V4.0. The STEP 7 user program, however, requires up to 14 times more memory than it used in previous releases.

### CP 5611 Card Support

WinAC RTX 2005 supports a CP 5611 card for communication to STEP 7 or to DP I/O. The CP 5611 card has reduced functionality from a CP 5613 card (for example, no isochronous mode, no time synchronization capability, fewer S7 connections, fewer slaves and a limit of one as a submodule), but offers lower costs than the CP 5613.

### Ability to use built-in CP 5611 PROFIBUS interface of Siemens Box, Rack, and Panel PCs

For Siemens Box, Rack, and Panel PCs, WinAC RTX 2005 supports the use of the built-in CP 5611 PROFIBUS interface for communication to STEP 7 or DP I/O. This capability means that you do not have to buy a separate communications card.

### Removal of SIMATIC NET Installation Requirement

WinAC RTX 2005 no longer requires you to install SIMATIC NET to be able to configure, program, and use WinLC RTX. SIMATIC NET is included with the installation, however, in case you need it for other purposes such as configuring additional components in the PC Station or using the OPC Server.

### Windows XPe Support

WinAC RTX pre-loaded on a Siemens Microbox PC 420 with the Windows XPe operating system extends the use of WinAC RTX to diskless computer systems, as well as lower-cost, smaller systems. Contact your local sales representative for more information about WinAC on embedded systems (separate product from WinAC RTX 2005).

### Time Sync Feature

WinAC RTX 2005 includes a Time Synchronization service as a component of the installation.

WinLC RTX can function as a time synchronization slave to the Time Synchronization service, or as a time master to devices on submodule DP-subnets. WinLC RTX can also serve as a time master to devices on the submodule DP-subnets. See your Time Synchronization service documentation for detailed information.

### Increase in Flag Memory

WinAC RTX 2005 provides 16 Kbytes of flag memory, an increase from 2 Kbytes in WinAC RTX V4.1.

### Additional Program Blocks

WinAC RTX 2005 adds the following blocks to the set that WinLC RTX supports:

- OB 88
- all OB 3x blocks
- SFB 31, SFB 33 - SFB 36
- SFC 9, SFC 10
- FB 63, FB 64, FB 65, and FB 66

To use these features, your PC must meet the system requirements.

**Note:** WinAC RTX 2005 does not support configuration of an Industrial Ethernet communications interface as a submodule of WinLC RTX. Although this documentation describes the capabilities of an Industrial Ethernet submodule and how to configure one, this feature is not available.

## System Requirements

To use WinLC RTX, your personal computer (PC) must meet the following criteria:

Category	Requirement
Operating System	<p>One of the following operating systems:</p> <ul style="list-style-type: none"> <li>• Microsoft Windows 2000 Professional Service Pack 3</li> <li>• Microsoft Windows XP Professional Service Pack 1 or Service Pack 2</li> </ul> <p>Ardence (formerly VenturCom) RTX version 6.1 (included with WinAC RTX)</p> <p><b>Note:</b> Some hardware configurations that are not SIMATIC industrial PCs do not support installation or operation of Ardence RTX. Refer to your RTX Runtime Release Notes on your installation CD for hardware and software system requirements for RTX.</p>
Processor and memory	<p>Pentium uniprocessor or multiprocessor system:</p> <ul style="list-style-type: none"> <li>• 400 MHz (800 MHz or faster recommended)</li> <li>• 256 Mbytes or more of RAM</li> <li>• BIOS must support plug-and-play (ACPI, Advanced Configuration and Power Interface)</li> </ul>
Hard drive	<p>A hard drive with 125 Mbytes of free space for the complete installation; setup options offer choice to not install some components such as documentation thereby reducing disk space requirements</p> <p>The setup program uses at least 1 Mbyte additional free space on drive C for the WinLC Setup program (Setup files are deleted when the installation is complete)</p>
Operator interface	<p>A color monitor, keyboard, and mouse or other pointing device (optional) that are supported by Windows</p>
Network interface	<p>One or more of the following communication interfaces for communications with STEP 7 or other S7 applications or for communications with distributed I/O:</p> <ul style="list-style-type: none"> <li>• CP 5611 hardware revision 5 or later</li> <li>• CP 5613 V3 or CP 5613 V6 or later</li> <li>• SIEMENS PC with built-in CP 5611 PROFIBUS interface: ASPC2 STEP E2 or ASPC2 STEP R ASIC</li> <li>• Industrial Ethernet interface supported by WinLC RTX: The Product Information document included with your installation lists the supported Industrial Ethernet interfaces. Industrial Ethernet interfaces cannot be used for DP I/O communication.</li> </ul> <p><b>Note:</b> WinLC RTX supports as submodules a maximum of one CP 5611 card or built-in CP 5611 PROFIBUS interface and a maximum of one Industrial Ethernet interface. The maximum number of submodules that WinLC RTX supports is four.</p>

Category	Requirement
Siemens software	<p>Programming and configuration software: STEP 7 V5.3 SP2 with the installed hardware update for WinLC RTX V4.2.</p> <p><b>Note:</b> You can use STEP 7 V5.2 or STEP 7 V5.3 if you do not need features dependent on later STEP 7 releases. Minimum STEP 7 requirements for these features are listed below:</p> <ul style="list-style-type: none"> <li>• Data exchange with broadcast between DP slave devices (STEP 7 V5.3)</li> <li>• Enhanced support for positioning and motion applications (STEP 7 V5.3)</li> <li>• Non-retentive DBs (STEP 7 V5.3)</li> <li>• Multi-PLC support (STEP 7 V5.3)</li> <li>• SSL_ID 0x1C (STEP 7 V5.3 SP2 + WinLC RTX V4.2 hardware update)</li> <li>• Time Synchronization (STEP 7 V5.3 SP2 + WinLC RTX V4.2 hardware update)</li> <li>• CP 5611 or Industrial Ethernet cards as WinLC RTX submodules (STEP 7 V5.3 SP2 + WinLC RTX V4.2 hardware update)</li> </ul>

## Windows User Privileges

You are not required to have Windows Administrator (ADMIN) privileges in order to perform WinAC RTX operations, such as changing the operating mode of the controller, modifying the sleep time or the minimum scan time of the controller, archiving or restoring control programs, or setting the security options.

With Power User, User or even with Guest privileges, you can perform any operation from the WinLC RTX controller panel. This allows you to manage the network privileges for the PC station within your application and to avoid conflicts during installation, commissioning and operation of a PC-based automation solution that is part of a larger system.

As shown in the following table, some operations are restricted to certain Windows User privilege classes.

Operation	Administrator	Power User	User	Guest
Installing WinAC RTX software	Allowed	Not allowed	Not allowed	Not allowed
Configuring or modifying the PC Station	Allowed	Allowed	Not allowed	Not allowed
Performing WinAC RTX operations	Allowed	Allowed	Allowed	Allowed

## Using Help

The online help system provides information about the controller panel and the controller. This topic provides information about using online help:

- Accessing Help from the Controller Panel
- Using the Table of Contents
- Using the Index
- Using Full-Text Search
- Printing Help Topics
- Changing the Language of a Help Topic

**Note:** If you have Windows XP SP 2 or later, you must allow blocked content in order to view all of the online help. The increased security features of Windows XP SP 2 block some features and controls used in the implementation of the online help. To enable blocked content when displaying the online help, click the security menu bar from your web browser and select "Allow Blocked Content", answering any prompts generated by this selection.

### Accessing Help from the Controller Panel

To access online help from the controller panel, use one of the following methods:

- Click one of the following entries on the Help menu to display the electronic manual that provides the online help:
  - **Help on Panel**

The **Help > Help on Panel** command opens the initial page of the online help for the controller panel. This help system describes the controller panel when it is not connected to a controller. The table of contents is in the left navigation pane.
  - **Help on Controller**

The **Help > Help on Controller** command displays the initial page of the online help for the controller that is connected to the controller panel. It describes controller and controller panel operations. The table of contents is in the left navigation pane.
  - **Introduction**

The **Help > Introduction** command displays a topic that provides an introduction to PC-based control and the capabilities of the controller.
  - **Getting Started**

The **Help > Getting Started** command displays a topic that helps you get started when you begin using the controller panel to work with the controller for the first time.
- Click the Help button in a dialog or message box to view information about that specific dialog or message box.
- Press the F1 key to view context-sensitive help on the currently selected item (for example, a window, dialog, or menu).



## Using the Table of Contents

The table of contents is in the left pane of the web browser and provides navigation within the online help system:

- Click a book to open it and display the books and topics that it contains.
- Click the book again to close it.
- Click any topic within the table of contents to display that topic.

The topic you are currently viewing is highlighted in the table of contents.

The table of contents can be either hidden or displayed:

- Click the "x" in the left navigation pane to close the table of contents.
- Select the Contents button on the browser or the "Show" link in a topic to display it. (The "Show" link appears only when you have displayed a context-sensitive topic from the application.)

## Using the Index

The index provides access to information about a specific subject. Use one of the following methods to access the index:

- Select the Index button on the browser. (If the Index button is not visible, click the "Show" link at the top of the topic. The "Show" link appears only when you have displayed a context-sensitive topic from the application.)
- Click the Index button in any help topic.

## Using Full-Text Search

To use the full-text search capabilities of the online help, use the search field that is displayed above the topic, or select the Search button on the browser. (If the Search field and Search button are not visible, click the "Show" link at the top of the topic. The "Show" link appears only when you have displayed a context-sensitive topic from the application.)

The full-text search supports the Boolean operators AND, OR, and NOT and parentheses in your search expression. Wildcards ("\*") are not supported.

## Printing Help Topics

To print a single topic that is displayed in your browser, right-click in the topic pane and select Print from the context menu. Select the print options of your choice.

## Changing the Language of a Help Topic

The browser contains language buttons for each of the supported languages. To see the current help topic in another language, click the language button of your choice. The current topic is displayed in the language you selected, but the contents, index, and search features of the online help system remain in the original language. This may be helpful if a topic is unclear and you want to read it in another language.

If you select a language that you did not install, the online help system cannot display the topic in that language and displays a "Page not found" error. Changing the language of an online help topic does not change the display language of the controller panel.

## Differences between WinLC RTX and WinLC Basis

Some of the operations of WinLC RTX differ from WinLC Basis:

- WinLC RTX provides a way to shut your process down in an orderly manner in the case of a Windows Blue Screen.
- WinLC RTX provides deterministic operation, ensuring predictable response time and reduced "jitter".
- WinLC RTX provides superior timing resolution in microseconds; whereas WinLC Basis provides timing resolution in milliseconds. For example, the times returned by SFC 78 and the sleep time in SFC 47 have a higher resolution in WinLC RTX.
- WinLC RTX supports an isochronous mode (constant bus cycle time).

WinLC RTX and WinLC Basis differ in how the blocks of the STEP 7 user program are stored in the memory of your computer:

- WinLC RTX uses only the non-paged RAM memory. All blocks of the STEP 7 user program and all of the process data must fit in the available non-paged memory that is also shared with other applications (such as device drivers) that are running on the computer.
- WinLC Basis runs in virtual memory that may be swapped to disk by the Windows operating system.

In addition, all new features introduced with WinLC RTX V4.2 are not available with WinLC Basis V4.1.



### Caution

Downloading a STEP 7 user program that is too large for the memory of the computer can lock up the computer or cause the operation of WinLC RTX to become unstable, possibly causing damage to equipment and/or injury to personnel.

Although STEP 7 and WinLC RTX do not limit the number of blocks or the size of the STEP 7 user program, your computer does have a limit, based on the available disk space and RAM memory. The limit for the size of the STEP 7 user program and number of blocks for your computer can only be determined by testing a configured system against the requirements of your control application.

# Installation

## Overview of the Installation Tasks

To install WinAC RTX, your computer must meet the system requirements and you must have Windows administrator (ADMIN) privileges and complete the following tasks:

- Uninstall any of the following software packages if they exist on your computer in the order listed, rebooting when finished:
  - WinAC Basis or WinAC Basis Demo
  - WinAC RTX
  - Ardence or VenturCom RTX
  - WinAC Slot release prior to WinAC Slot V3.4, or upgrade your existing to release to V3.4
- Install the Ardence RTX extensions.
- Verify that the Ardence RTX extensions are working.
- Install the WinAC RTX software.

For communication with distributed I/O over a PROFIBUS-DP network, your computer must have one or more DP interfaces.

To use the OPC Server or other SIMATIC NET features, you must install SIMATIC NET from the CD included with your WinAC RTX installation; otherwise, you do not need to install the SIMATIC NET CD.

Following installation, license the WinAC RTX installation using the Automation License Manager.

The succeeding topics contain the installation and licensing procedures.

## Installing the Ardence RTX Extensions

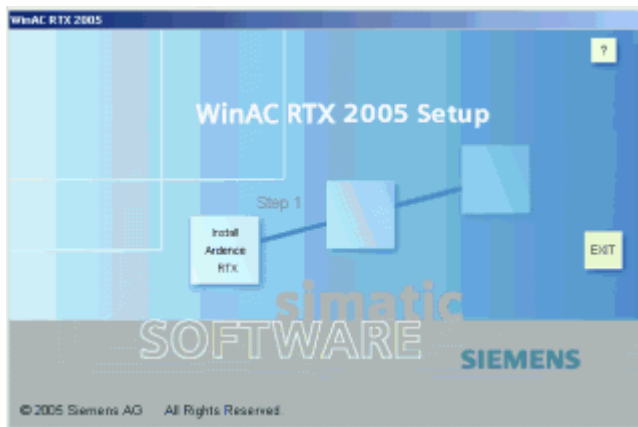
To install the WinAC RTX software including the Ardence RTX extensions, remove in order the software named in "Overview of the Installation Tasks" and perform the following tasks as directed from the WinAC RTX 2005 Setup screen:

- Install the Ardence RTX extensions.
- Verify that the Ardence RTX extensions are working.

**Note:** You must have Windows administrator (ADMIN) privileges to install the Ardence RTX extensions.

- Install the WinAC RTX software.

When you insert the installation CD, the Setup program starts automatically. If the Setup program does not automatically start, browse the CD and double-click the Setup.exe file. After you select the language for the Setup program, the Setup program displays a dialog that guides you through the installation tasks:



### Step 1: Install the Ardence RTX Extensions

Click the square under Step 1 to start the Install Wizard for installing the Ardence RTX extensions. The Install Wizard requires you to enter an Ardence PAC number and email address. The PAC number is on the back of the WinAC RTX CD case. The email address to enter is winac@siemens.com.

**Note:** If you have previously installed Ardence (Venturcom) RTX Version 6.1, the installation software detects its presence and the first square is inactive.

## Step 2: Verify that the Ardence RTX Extensions Are Operational

The Install Wizard restarts your computer after installing the RTX extensions. Click the square under Step 2 and follow the instructions to verify that the RTX extensions are working. The Install Wizard displays the following instructions:

1. Open the Windows Control Panel.
2. Double-click the RTX Properties icon to display the RTX Properties dialog.
3. Click the Control tab to view the status of the RTX extensions. The Status field lists the state of the RTX extensions (either running or stopped).
4. Click the Start RTX button to start the RTX extensions. Wait a few seconds to see that the status of the RTX extensions changes from Stopped to Running.

### Caution

Clicking the Stop RTX button before the RTX extensions have completely switched to Running can cause the computer to become unresponsive. If this happens, you must restart your computer to recover.

Always wait a few seconds until the requested action is complete and the display shows current status before clicking the Start RTX or Stop RTX buttons.

5. Wait a few seconds and then click the Stop RTX button. Verify that the RTX extensions in the Status field change to Stopped. If the Start and Stop buttons work correctly, your RTX extensions are installed and functioning. However, this procedure verifies only a minimum level of platform suitability for RTX.
6. If the RTX extensions are working correctly, close the RTX Properties dialog.

After you have installed and verified the Ardence RTX installation, you can evaluate your platform and DP interfaces for use with RTX.

## Step 3: Installing the WinAC RTX Software

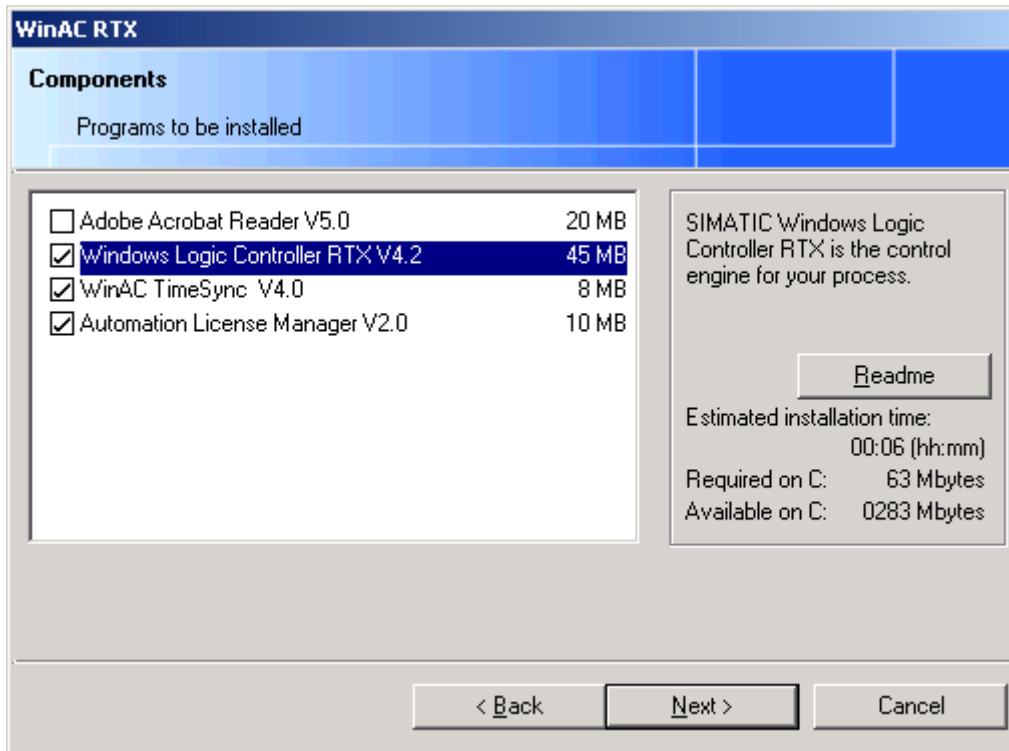
When you complete the verification of the RTX extensions on your platform, proceed with the installation of WinAC RTX.

## Installing the WinAC RTX Software

The WinAC RTX installation begins when you click the square under Step 3 of the setup screen. If the Setup program is not running, double-click the setup.exe file on the CD.

**Note:** You must remove any previous versions of WinAC or RTX from your computer and you must have Windows administrator (ADMIN) privileges to install the WinAC RTX software.

From the list of WinAC RTX components, select the components to be installed:



After you make your selections and click the Next button, the Setup program continues. Proceed through the installation dialogs.

During the installation process, you choose which setup type you prefer:

- **Typical:** installs all software and by default, all documentation in all supported languages
- **Minimal:** installs WinLC RTX in only one language and with no documentation, requiring the least amount of disk space
- **Custom:** installs the languages, online help, and manuals that you select on subsequent dialogs

You can also choose to license WinAC RTX during the installation process or at a later time.

**Note:** If you have SIMATIC NET installed on your computer, and you get messages from SIMATIC NET that say the CP card is configured for use with SIMATIC NET and STEP 7, click OK. This is a normal part of the installation process.

The Setup program notifies you when the installation is complete.

## Licensing the WinAC RTX Software

The WinAC RTX software requires a product-specific license that you install using the Automation License Manager. Each SIMATIC automation software product (for example, STEP 7) has a separate license. You must install the license for each product.

### Installing the License during Installation

When you install the software for the first time, the Automation License Manager is part of the installation set. If it is not already on your computer, select the Automation License Manager checkbox during installation. A subsequent dialog allows you choose whether to install the license during installation. Refer to the Automation License Manager online help for help with installing a license.

### Installing the License at a Later Date

If you attempt to start the WinAC RTX software and no license is found, a prompt appears on the screen. You use the Automation License Manager to install the license. If the Automation License Manager is not installed on your computer, follow these steps to install the Automation License Manager and license WinLC RTX:

- Insert the WinAC RTX installation CD.
- From the Welcome screen, click the square for Step 3.
- When the Components dialog appears, select the Automation License Manager checkbox.
- After the installation is complete, select the **Start > SIMATIC > License Management > Automation License Manager** menu command or launch the Automation License Manager from the desktop.
- Proceed with license installation according to the instructions in the Automation License Manager online help.

### Transferring an Installed License

The Automation License Manager provides steps to transfer a license from one computer to another computer. The two computers do not have to be connected over a network to perform an offline transfer of license keys. Refer to your Automation License Manager online help for assistance.

### Running the WinLC RTX Controller without a License

If no license for WinAC RTX exists on your computer, the WinLC RTX controller continues to operate; however, a notification message appears every six minutes to alert you that the license is missing.

### Recovering the License in Case of a Defective Hard Drive

If a fault occurs with the license file on your hard disk, contact your local Siemens representative.

## Uninstalling Ardence RTX or WinAC RTX

Use the following procedure to remove Ardence RTX or WinAC RTX from your computer. If you are uninstalling both, ensure that you uninstall WinAC RTX first:

1. Double-click the Add/Remove Programs icon in the Windows Control Panel.
2. Select the Ardence RTX or WinAC RTX component entry in the displayed list of installed software. Click Add/Remove to uninstall the software.

If the Remove Shared File dialog boxes appear, click No if you are unsure how to respond.





# Getting Started

The Getting Started section helps you to establish communications between the controller, STEP 7, and I/O devices. You must perform the following tasks:

- Use the Station Configuration Editor to designate a communication interface as a submodule of WinLC RTX.
- Use STEP 7 to configure the hardware and STEP 7 user program and to download the system blocks.

The Getting Started section also helps you understand the basic concepts for setting up a PC-based controller: PC station, Communication Interface, Index, Submodule, and Interface (IF) Slot.

## Understanding the Concepts

### What Is a PC Station?

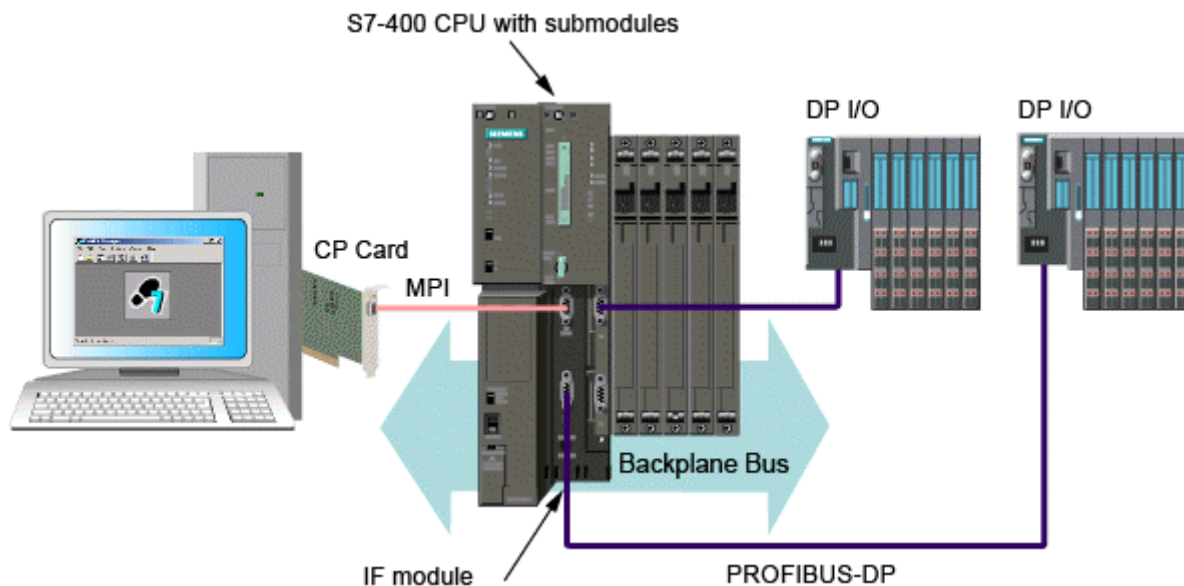
The PC station is a software-based virtual rack, represented in the Station Configuration Editor, for creating a PC-based automation system. Like a hardware rack of an S7 CPU-based automation system, it contains space for several modules required for the PC-based automation system.

When you install the WinAC RTX software, the controller appears by default in the second slot (index) of this virtual rack in the Station Configuration Editor. The PC Station is also represented in the STEP 7 Hardware Configuration editor. The controller in the PC Station contains four configurable IF Slots for designating communication interfaces as submodules to be used for communication with distributed I/O, STEP 7, or other S7 applications.

### Communication Model with S7-400

A PC-based controller is similar to an S7-400 hardware controller. The S7-400 controller consists of modules in a rack that communicate over the backplane bus of the rack. Communications for an S7-400 are defined as follows:

- STEP 7 communicates with the controller (in this example, an S7-400 CPU module) over an MPI subnet, using a CP card that is installed in the computer.
- The controller communicates with expansion modules over the backplane bus of the rack.
- The S7-400 CPU communicates with distributed I/O over a PROFIBUS-DP subnet using a built-in submodule interface or an IF module.



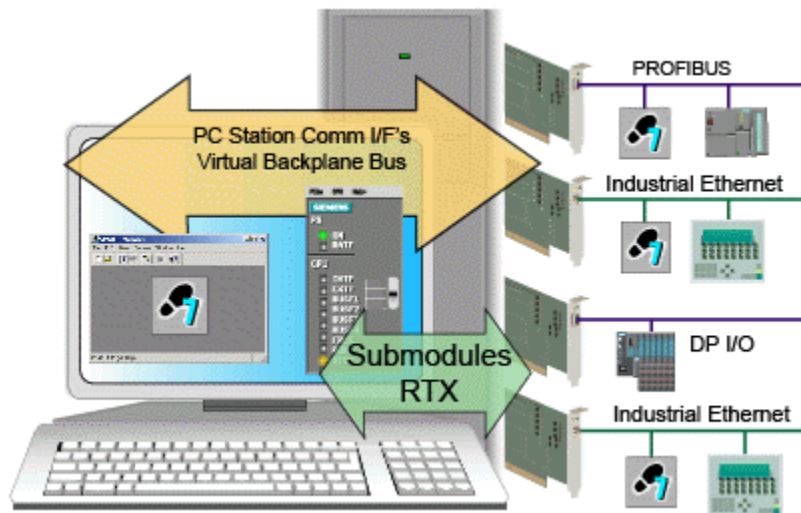
In an S7-400 station, the following types of communications are possible:

Onboard Interfaces	CP Modules used over Backplane Bus
Operation of PROFIBUS DP/IO Supported interfaces: <ul style="list-style-type: none"> <li>• MPI</li> <li>• PROFIBUS</li> </ul>	Operation of central I/O Supported communication processors: <ul style="list-style-type: none"> <li>• PROFIBUS</li> <li>• Industrial Ethernet</li> </ul>

## Communication Model with PC Station and PC-based Controller

WinLC RTX uses communication interfaces such as CP 5613 cards for communication tasks and access to distributed I/O. You can configure and use communication interfaces in WinLC RTX in one of two ways:

- Submodule configuration:** A communication interface configured as a submodule operates in the real-time subsystem and provides optimal performance and stability for communication with distributed I/O. Submodules of WinLC RTX are similar to the onboard communication interfaces of an S7-400 controller.
- PC Station configuration:** A communication interface configured in the PC Station operates in the Windows operating system and is available for a variety of communication tasks. It cannot, however, be used for WinLC RTX communication with distributed I/O. PC Station communication interfaces are similar to CP modules installed in the rack of an S7-400 controller. WinLC RTX uses a virtual backplane bus that is similar to the S7 CPU backplane bus for communication with components in the PC Station and with other PC applications on the computer with WinLC RTX.



**Note:** Configuring communication interfaces as submodules of WinLC RTX requires no additional software; configuration in the PC Station requires the installation of SIMATIC NET, a separate software package.

The following table lists the characteristics of the two types of communication:

Submodule Communication (similar to onboard interface on an S7 CPU)	PC Station Communication (similar to a CP module communicating over the backplane bus of an S7-400 station)
Operates exclusively in the real-time subsystem Access to PROFIBUS-DP I/O Supported protocols/communication types: <ul style="list-style-type: none"> <li>PROFIBUS               <ul style="list-style-type: none"> <li>PG/OP communication</li> <li>S7 communication</li> <li>S7 routing</li> <li>DP I/O</li> </ul> </li> <li>Industrial Ethernet               <ul style="list-style-type: none"> <li>PG/OP communication</li> <li>S7 communication</li> <li>S7 routing</li> </ul> </li> </ul>	Operates in the Windows environment No access to PROFIBUS-DP I/O (or central I/O) Supported protocols/communication types: <ul style="list-style-type: none"> <li>PROFIBUS               <ul style="list-style-type: none"> <li>PG/OP communication</li> <li>S7 communication</li> <li>S7 routing</li> </ul> </li> <li>Industrial Ethernet               <ul style="list-style-type: none"> <li>PG/OP communication</li> <li>S7 communication</li> <li>S7 routing</li> </ul> </li> </ul>

Submodule Communication (similar to onboard interface on an S7 CPU)	PC Station Communication (similar to a CP module communicating over the backplane bus of an S7-400 station)
Does not require SIMATIC NET installation	Requires SIMATIC NET installation
Refer to the topic "What Is a Communication Interface?" for a list of communication interfaces that WinLC RTX supports.	Refer to the SIMATIC NET documentation for a list of supported communication interfaces.

### Configuration for a PC-based Controller

You use the Station Configuration editor to configure components of the PC Station. You edit the properties of WinLC RTX in the Station Configuration Editor to configure submodules.

In the same way that you use STEP 7 to create the system and program blocks for an S7-400, you use the STEP 7 Hardware Configuration tool to configure the components that you installed in the PC station.

After you complete hardware configuration in STEP 7 and submodule configuration with the Station Configuration Editor, you can download your STEP 7 user program to the controller.

**Note:** To use the CP card for communicating with both STEP 7 and with the distributed I/O may require an additional software license. See your Siemens sales representative or distributor for more information.

### What Is a Communication Interface?

A communication interface is a CP card, built-in PROFIBUS interface on a Siemens Box, Rack, or Panel PC, or an Industrial Ethernet card that can be configured as a submodule of WinLC RTX. Communication interfaces configured as WinLC RTX submodules enable communications between STEP 7 or other S7 applications and WinLC RTX, and except for Industrial Ethernet interfaces enable communications between WinLC RTX and distributed I/O over a DP-subnet.

You must configure a communication interface as a submodule of WinLC RTX in order to use it for DP I/O communications. WinLC RTX supports the following communication interfaces and communication types:

Communication Interface	DP I/O	PG/OP	S7	S7 Routing
CP 5613 V3 or CP 5613 V6 or later	☑	☑	☑	☑
CP 5611 hardware revision 5 or later	☑	☑	☑	☑
SIEMENS PC with built-in CP 5611 PROFIBUS interface: ASPC2 STEP E2 or ASPC2 STEP R ASIC	☑	☑	☑	☑
Industrial Ethernet interface supported by WinLC RTX: The Product Information document included with your installation lists the supported Industrial Ethernet interfaces. Industrial Ethernet interfaces can be used only for S7 communications, not DP I/O communications.	N/A	☑	☑	☑

### What is a DP Interface?

A DP interface is a communication interface that can be used to communicate with distributed I/O over PROFIBUS-DP.

## What Is an Index?

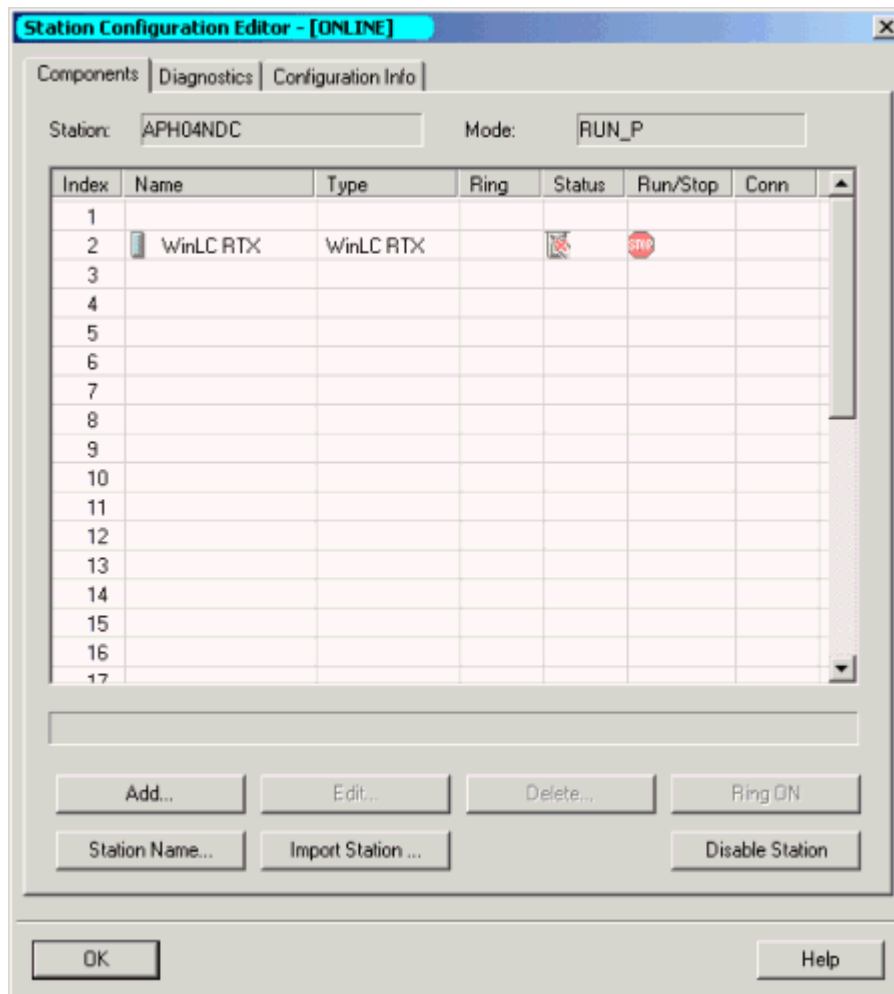
An index is a numbered slot in the virtual rack of the PC station. The PC Station provides slots for the SIMATIC components of a PC-based automation solution, including not only the PC-based controller, but potentially other components according to your application:

- CP card(s) (requires installation of SIMATIC NET)
- SIMATIC HMI
- SIMATIC NET OPC Server (requires installation of SIMATIC NET)
- CPU 41x-2 PCI (WinAC Slot PLC)

Each slot in the PC station corresponds to a number or index. When you install WinLC RTX, the installation configures the controller in the second index slot by default. The Station Configuration Editor shows the configuration of your PC station.

The index number for a component can be any index number you choose; however, the index number in the Station Configuration Editor must be the same as the slot number in the STEP 7 Hardware Configuration tool for the same component.

**Note:** If you have deleted WinLC RTX from the Station Configuration Editor, the **Start > Simatic > PC-based Control** menu does not have an entry for WinLC RTX. To restore this menu selection, you must configure WinLC RTX in an index of the Station Configuration Editor.



## What Is a Submodule?

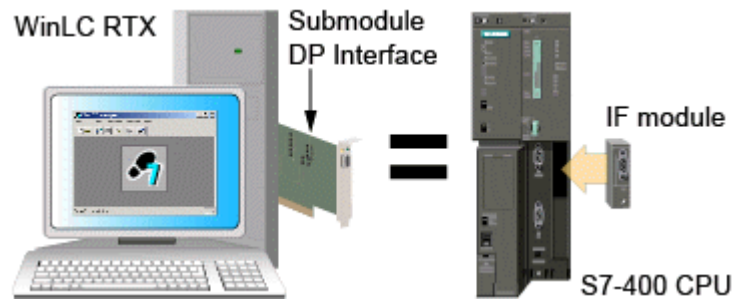
A submodule is a configured communication interface that enables communication between WinLC RTX and distributed I/O or between WinLC RTX and STEP 7 or other S7 applications.

In order for WinLC RTX to communicate with DP I/O devices on a PROFIBUS-DP network, you must designate a DP interface as a submodule for the controller. With this submodule approach, WinLC RTX has full control over the DP I/O communications, providing optimum performance and determinism for operating the I/O. WinLC RTX supports up to four submodules configured in any of the four IF slots. WinLC RTX does not support the use of an Industrial Ethernet interface for DP I/O communication; it can be used for PG/OP or S7 communications only.

If you use a CP 5611 card or built-in CP 5611 PROFIBUS interface as a submodule of WinLC, note that you can insert only one as a submodule. Similarly, you can insert only one Industrial Ethernet interface as a submodule.

Configuring a DP interface as a submodule of WinLC RTX is like installing an IF module into a slot of an S7-400 CPU.

Designating a DP interface as a submodule of WinLC RTX not only allows WinLC RTX to use that interface for PG/OP communication, but also allows WinLC RTX to use that interface for communicating with the DP I/O.



In order for a submodule to be used for SIMATIC communications with an application other than WinLC RTX (on the PC station), the second application must be a configured part of the PC station.

### Notice

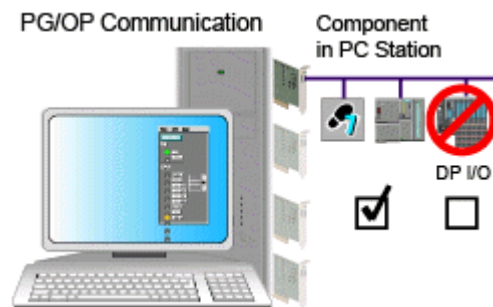
Any change to the configuration of a submodule causes WinLC RTX to delete the STEP 7 user program. You must then download a new STEP 7 user program and configuration.

For example: If you move the submodule to a different slot, WinLC RTX recognizes that the hardware configuration has changed and deletes the STEP 7 user program, even if you had reconfigured the card for the same IF slot from the STEP 7 Hardware Configuration tool. You must still reload the STEP 7 user program from STEP 7.

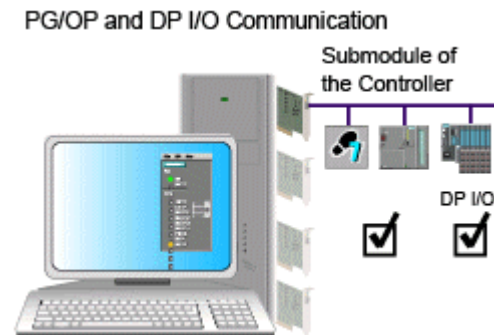
The figure shows the difference between a DP interface (in this case, a CP card) as a component of the PC station and the DP interface as a submodule of WinLC RTX.

**Note:** Configuring a DP interface as a component of the PC Station requires the installation of SIMATIC NET. As a component of the PC station, you can use it only for SIMATIC communications (PROFIBUS or Industrial Ethernet) with STEP 7, SIMATIC HMI, or other SIMATIC controllers. For example, you can download a program from STEP 7 to WinLC RTX. A DP interface configured as a component of the PC Station cannot be used for WinLC RTX communications with DP I/O.

With a communications interface configured as a component of the PC station, WinLC RTX can communicate with STEP 7 on a remote computer, but cannot communicate with the DP I/O.



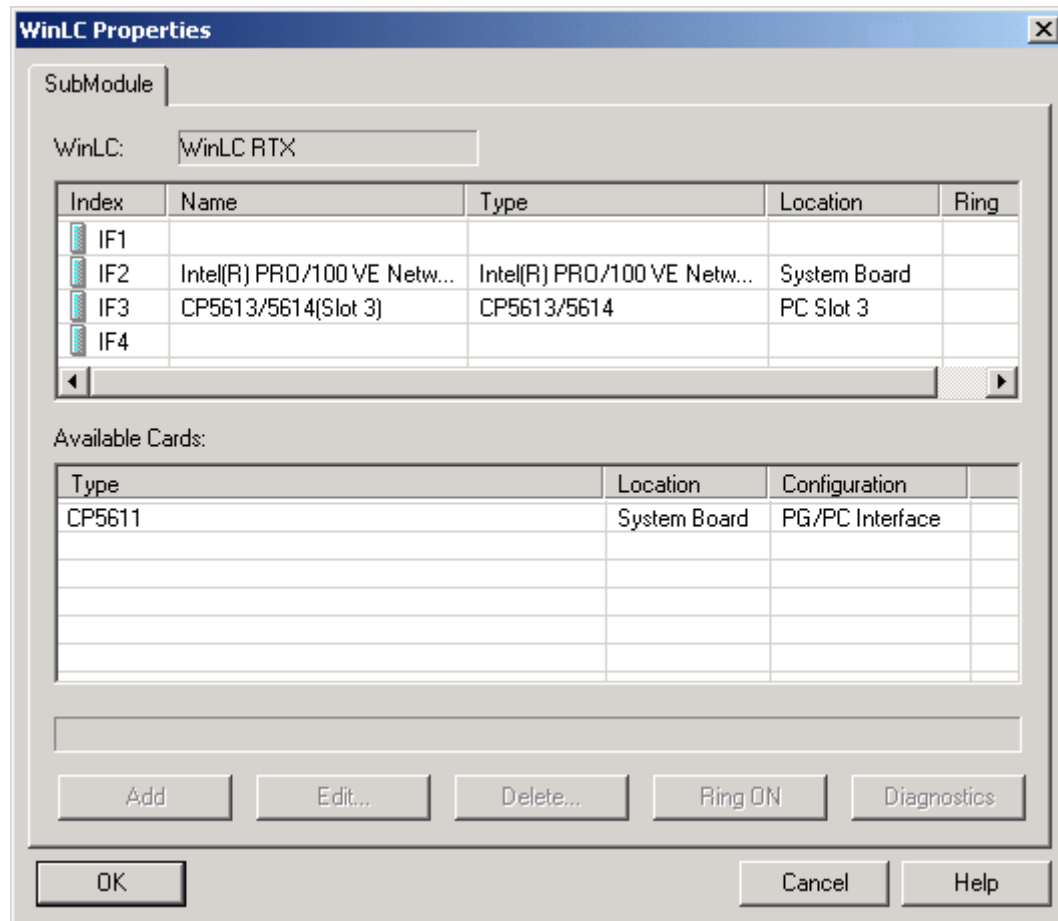
With a DP interface configured as a submodule, WinLC RTX can communicate with both STEP 7 on a remote computer (using PG/OP communication) and with the DP I/O on the PROFIBUS-DP subnet.



## What Is an IF Slot?

WinLC RTX provides four interface (IF) slots for designating communication interfaces as submodules. WinLC RTX has exclusive control of any card configured in an IF slot. The submodules enable the controller to communicate with distributed I/O, or with STEP 7 or other S7 applications.

To use the PROFIBUS-DP network to communicate with I/O, you must configure at least one DP interface to be a submodule of WinLC RTX. You use the WinLC Properties dialog to assign a communication interface to one of four interface slots, IF1 through IF4:



If you use a CP 5611 card or built-in CP 5611 PROFIBUS interface as a submodule of WinLC, note that you can insert only one as a submodule. Similarly, you can insert only one Industrial Ethernet interface as a submodule.

For CP cards and Industrial Ethernet cards, the IF slot number of the submodule is independent of the PCI hardware slot. However, the IF slot number for the submodule in WinLC Properties must match the IF slot number in the STEP 7 Hardware Configuration tool.

For information on submodule configuration, refer to the following topic: Designating a Communication Interface as a Submodule



## Configuring Communication Cards


### Designating a Communication Interface as a Submodule

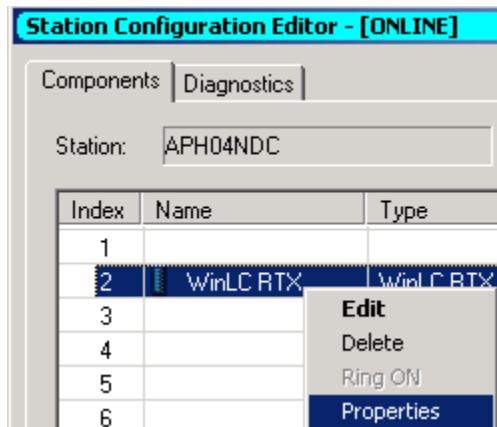
You configure a communication interface as a submodule of WinLC RTX by inserting it into an IF slot of the controller. Submodule communication interfaces enable WinLC RTX to communicate with STEP 7 or other S7 applications and submodule DP interfaces also enable WinLC RTX to communicate with distributed I/O over the PROFIBUS-DP network.

You use the Station Configuration Editor to insert a communication interface into an IF slot of WinLC RTX.

You must also configure WinLC RTX, the submodules, and all other components of the PC station in STEP 7.

To configure your communication Interface as a submodule, ensure that the controller is shut down and follow these steps:

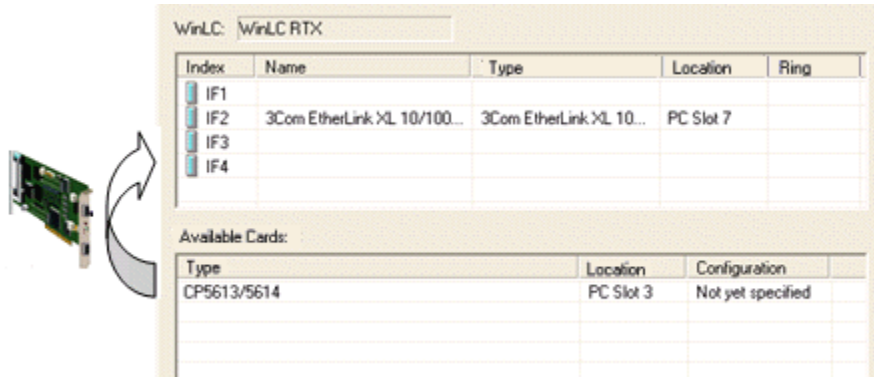
1. Double-click the computer icon  on the Windows taskbar to open the Station Configuration Editor.
2. Right-click WinLC RTX and select Properties from the context menu to display the WinLC Properties dialog.



The WinLC Properties dialog displays the four submodule interfaces (IF1 to IF4) in the upper panel and a list of available communication interfaces in the lower panel. The following example shows an Industrial Ethernet card already configured in IF Slot 2 and a CP5613/CP5614 card being added as a submodule.

3. In the lower panel, select the CP card or Industrial Ethernet card that you want to configure as a submodule. (A built-in PROFIBUS interface is represented as a CP5611 card with a location of "System Board.")

4. Drag the selected device to an empty interface slot (IF slot) in the upper panel or click the Add button to add the card to the first available interface slot.



For multiple cards, repeat the above steps as needed.

5. Click OK on the WinLC Properties dialog to accept your changes and to configure the submodule(s). This configuration may take several seconds.

**Note:** WinLC RTX supports a maximum of one CP 5611 card or built-in CP 5611 PROFIBUS interface as a submodule, a maximum of one Industrial Ethernet interface as a submodule, and a maximum of four CP 5613 cards as submodules. You can configure any combination of communication interfaces in the four interface slots as long as you observe these maximums.

You can select an occupied interface slot (IF slot) and click the Edit button to change the interface slot assignment for a configured DP interface, or to change its name. You can also use the up/down arrow keys on the keyboard to move a submodule to a different interface slot.

**Notice**

Any change to the configuration of a submodule causes WinLC RTX to delete the STEP 7 user program. You must then download a new STEP 7 user program and configuration.

For example: If you move the submodule to a different slot, WinLC RTX recognizes that the hardware configuration has changed and deletes the STEP 7 user program, even if you had reconfigured the card for the same IF slot from the STEP 7 Hardware Configuration tool. You must still reload the STEP 7 user program from STEP 7.

## Creating an INI File for an Industrial Ethernet Interface

An INI file is a configuration file for an Industrial Ethernet interface that RTX supports. The Product Information included with your WinAC RTX release lists the supported Industrial Ethernet interfaces.

When you close the WinLC Properties dialog after you designate an Industrial Ethernet card as a submodule of WinLC RTX or change the configuration of an Ethernet submodule, Notepad opens with the INI file for the submodule Industrial Ethernet interface. You must complete the INI file configuration for your specific Industrial Ethernet interface by following these steps:

1. For the `IPAddr` line, replace the displayed address with the address of your Industrial Ethernet interface.
2. For the `Netmask` line, replace the displayed value with the subnet mask of your Industrial Ethernet interface.
3. Delete lines for cards other than the Industrial Ethernet interface that you are configuring. At a minimum the INI file contains a section for an RTNE2K driver. Delete these lines:

```
[rtnd0]
Driver=RTNE2K
IPAddr=192.168.4.8
Netmask=255.255.0.0
InterruptPriority=63
ReceivePriority=64
IRQ=5
IOAddr=0240
NumRecvBuffers=48
NumXmitBuffers=48
LatencyRecvUpdated=0
```

4. Ensure that the line following `[WinLC]` is correct. This line must display `IFx = rtndy` where `x` is the IF slot in your configuration and `rtndy` corresponds to the `[rtndy]` line preceding the driver assignment for your Industrial Ethernet interface.
5. Save the file when you are sure it is correct.

The following listing is a completed INI file for a system with an Intel(R) PRO/100 VE 5 card configured as a submodule in IF slot 4. The highlighted lines indicate the lines in the file that you edit or verify:


```
[TCP/IP]
MemoryInK=256
TickInterval=200
MaxSockets=30
TimerPriority=61
NumStartupEvents=50
[rtnd1]
Driver=Rt8255x
IPAddr=192.168.4.2
Netmask=255.255.0.0
InterruptPriority=63
ReceivePriority=64
LatencyRecvUpdated=0
NumRecvBuffers=48
NumXmitBuffers=48
[WinLC]
IF4=rtnd1
```

At the completion of these steps, the Ethernet interface is configured to use a valid INI file.

## Testing a CP 5613 Configuration

The WinLC Properties ring test allows you to verify whether a submodule CP 5613 card is configured correctly. This test is especially important if you have installed more than one CP 5613 card in your computer. This test is not available for CP 5611 or Ethernet cards.

To check the operation of the submodule CP card, follow these steps:

1. Start WinLC RTX if it is not already started. (The ring test is only available when WinLC RTX is operating.)
2. Double-click the computer icon  on the Windows taskbar to open the Station Configuration Editor.
3. Double-click the WinLC RTX index entry to display the WinLC Properties dialog.
4. Select the interface slot (IF slot) containing the CP card to be tested.
5. Click the Ring ON button.


The LEDs on the CP card at the back of your computer flash in an alternating pattern so you can verify that you have configured the correct CP card. The computer also emits an audible beep if the CP card is functioning.

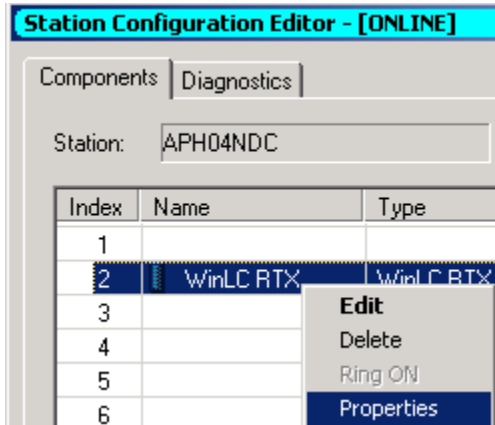
6. Click the Ring OFF button to end the test of the CP card.

## Removing a Submodule

From the WinLC Properties dialog, you can move a DP interface that is configured as a submodule of WinLC RTX to the set of available cards in your computer.

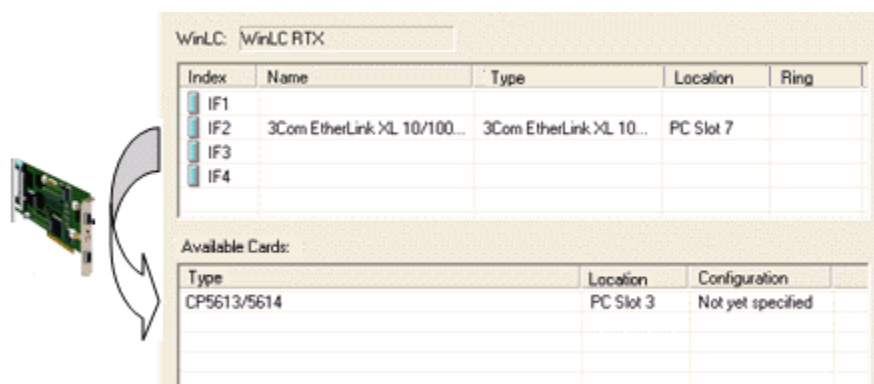
To remove a DP interface from the WinLC RTX submodule configuration, ensure that the controller is shut down and follow these steps:

1. Double-click the computer icon  on the Windows taskbar to open the Station Configuration Editor.
2. Right-click WinLC RTX and select Properties from the context menu to display the WinLC Properties dialog.



The WinLC Properties dialog displays the four submodule interfaces (IF1 to IF4) in the upper panel and a list of available communications cards in the lower panel. The following example shows a CP5613/CP5614 card being removed as a submodule, leaving an Industrial Ethernet interface in IF Slot 2.

3. In the upper panel, select the CP card or Industrial Ethernet interface that you want to remove as a submodule.
4. Drag the selected card to an available position in the lower panel or right-click the selected card and click the Delete button or Delete key from the keyboard. After you confirm your action, the card is removed as a submodule and returned to the list of available cards.



5. Click OK on the WinLC Properties dialog to accept your changes.

## Configuring the Controller in STEP 7

### Connecting STEP 7 to the Controller

You must establish a connection from STEP 7 to the controller to download the configuration and blocks of the STEP 7 user program. This type of communication is called PG/OP communication. The controller can connect to STEP 7 in any of the following ways:

- Through the virtual backplane bus to STEP 7 on the same computer as the controller
- Through a submodule communications interface to STEP 7 on a different computer
- Through a PC Station communications interface to STEP 7 on a different computer

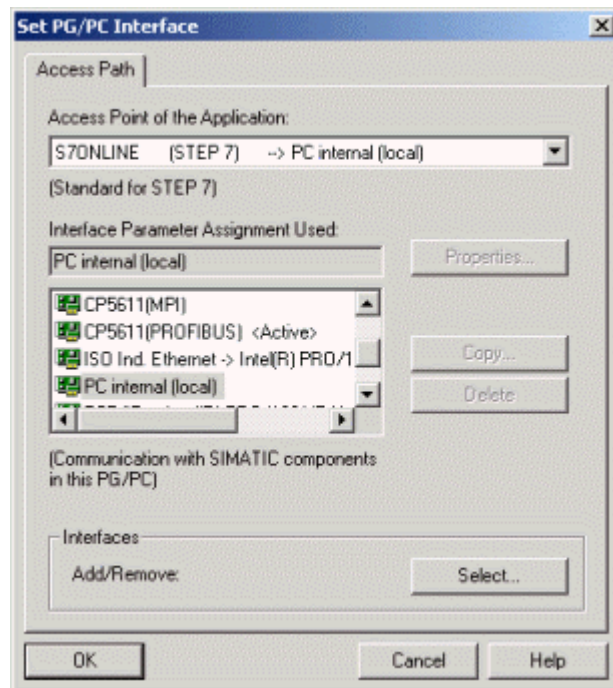
**Note:** Configuring a communications interface in the PC Station and not as a submodule requires the installation of SIMATIC NET, an additional software package.

### Connecting STEP 7 to the Controller on the Same Computer

On the same computer, STEP 7 and the controller communicate across the virtual backplane bus:



To configure communications between the controller and STEP 7 on the same computer, use the PC internal access point:

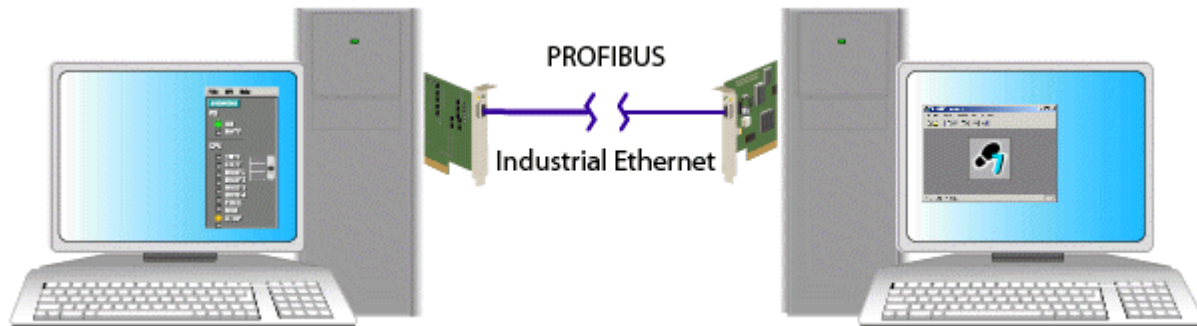


## Connecting STEP 7 to the Controller on a Different Computer

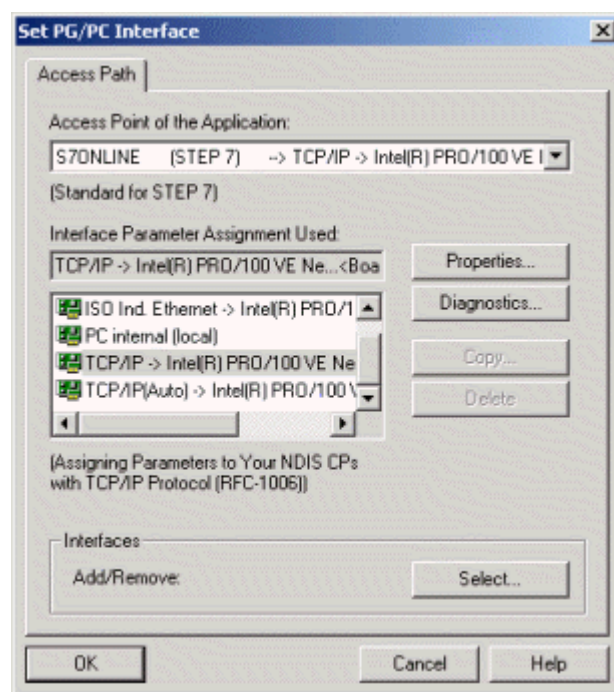
STEP 7 can communicate to WinAC RTX on a different computer or programming device through a communications interface that is configured as a submodule of the controller or through a communications interface that is configured in the PC Station. To configure a communications interface in the PC Station requires SIMATIC NET to be installed on your computer.

The following communications types are supported:

- PROFIBUS: for a CP 5613, CP 5611, or built-in PROFIBUS interface configured as a submodule
- Industrial Ethernet: for an IE card configured as a submodule or in the PC Station



To configure communications between the controller and STEP 7 on a different computer or programming device, set the PG/PC interface to the access point for the specific communications interface and the type of communications, for example an Industrial Ethernet card using the TCP/IP protocol:



## Connecting STEP 7 to a Networked Controller on a Different Computer within the STEP 7 Project

STEP 7 uses the PC internal interface to communicate to WinLC RTX on another PC Station within the same STEP 7 project. With the PC internal interface, STEP 7 communicates to all controllers (S7 CPUs or PC-based controllers) that have been configured with NetPro in PC Stations in the STEP 7 project. To use this method of PG/OP communication, set the PG/PC interface for S7ONLINE to PC internal.

## Hardware Configuration in STEP 7

You configure the STEP 7 project for a PC station with a PC-based controller in STEP 7 as you would for any S7 hardware controller. Refer to the STEP 7 help system and documentation for detailed information.

### Creating the Project and PC Station with the SIMATIC Manager

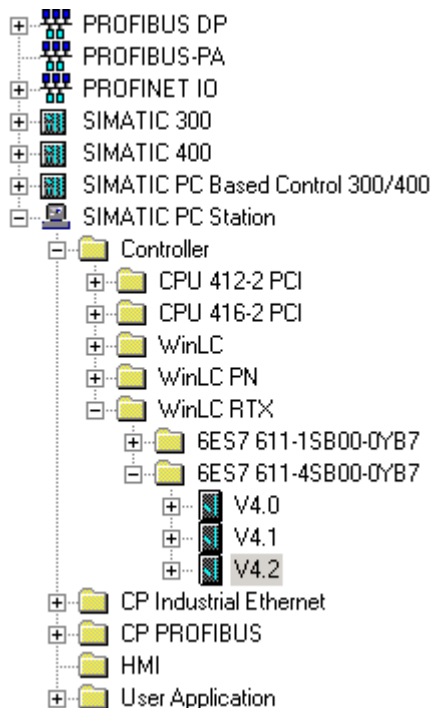
To create the project and PC station, follow these steps:

1. Select the **File > New** menu command from the SIMATIC Manager to create a new project.
2. Select the **Insert > Station > SIMATIC PC Station** to insert a PC station into the project.
3. Change the name of the PC station to match the name of the PC station defined in the Station Configuration Editor on the computer where WinLC RTX resides. To find the station name, open the Station Configuration Editor and click the Station Name button.

### Configuring the PC Station Hardware with the STEP 7 Hardware Configuration Application

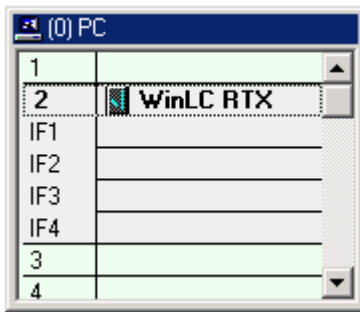
To configure the PC-based controller and DP I/O for the PC Station, follow these steps:

1. Open the PC Station folder in the project and double-click the Configuration icon to invoke the STEP 7 Hardware Configuration application.
2. Navigate to your specific controller under SIMATIC PC Station.





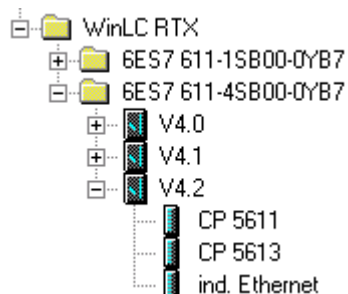
- Drag the controller into the same index it occupies in the Station Configuration Editor on the target computer.



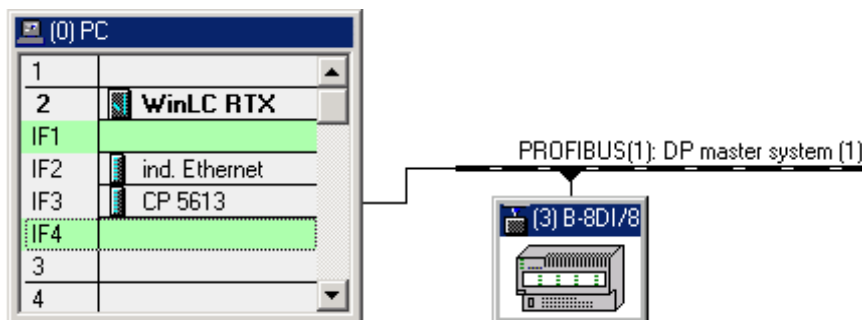
- Verify that the name of the controller matches the name of the controller configured in the Station Configuration Editor.
- Drag the submodule CP or Industrial Ethernet cards from the Hardware Catalog into interface slots (IF slots) for the WinLC RTX controller. The WinLC RTX folder in the Hardware Catalog lists the CP card types that are available. (For built-in PROFIBUS interfaces on SIEMENS PCs, you select a CP5611 card.)

The submodule cards do not have to have the same name as in the PC Station configuration; however, this is recommended. They must have the same type and interface (IF) number as specified in the Station Configuration Editor.

If you use a CP 5611 card or built-in CP 5611 PROFIBUS interface as a submodule of WinLC, note that you can insert only one as a submodule. Similarly, you can insert only one Industrial Ethernet interface as a submodule.



- Configure the distributed I/O for each of the submodule DP networks. Industrial Ethernet cards cannot be connected to a DP network.



## Additional Hardware Configuration Options

The following tasks are optional, depending on your specific application:

1. Insert any CP PROFIBUS or CP Industrial Ethernet cards that are not used as submodules for the controller into the PC Station.  
**Note:** You must have installed SIMATIC NET to configure CP cards or Ethernet cards in the Station Configuration Editor that are not submodules of WinLC RTX.
2. Insert any HMI devices, for example, text displays or operator panels.
3. Configure WinLC RTX for peer-to-peer communications:
  - a. Select the controller name in the SIMATIC Manager.
  - b. Double-click the Connections icon in the right-hand pane.
  - c. Use NetPro to describe the network.

After you have configured WinLC RTX you can use SIMATIC Manager to develop and to download your STEP 7 user program.



### Caution

Downloading a STEP 7 user program that is too large for the memory of the computer can lock up the computer or cause the operation of WinLC RTX to become unstable, possibly causing damage to equipment and/or injury to personnel.

Although STEP 7 and WinLC RTX do not limit the number of blocks or the size of the STEP 7 user program, your computer does have a limit, based on the available disk space and RAM memory. The limit for the size of the STEP 7 user program and number of blocks for your computer can only be determined by testing a configured system against the requirements of your control application.

After you have downloaded your program to the controller, you can start the controller and use STEP 7 to monitor and modify the process variables.

### Notice

Any change to the configuration of a submodule causes WinLC RTX to delete the STEP 7 user program. You must then download a new STEP 7 user program and configuration.

For example: If you move the submodule to a different slot, WinLC RTX recognizes that the hardware configuration has changed and deletes the STEP 7 user program, even if you had reconfigured the card for the same IF slot from the STEP 7 Hardware Configuration tool. You must still reload the STEP 7 user program from STEP 7.

## Invalid Characters prior to STEP 7 V5.3 SP1

You can use STEP 7 to create a name for the controller and download the configuration with the new name to the controller in the Station Manager. However, you cannot use invalid characters for the name of the controller if you are using a version of STEP 7 prior to V5.3 SP 1.

### Caution

Prior to STEP 7 V5.3 SP1, using an invalid character in the controller name creates an instance of the controller that cannot be restarted.

Downloading a configuration that uses an invalid character in the controller name creates an invalid instance of the controller. This invalid instance will continue to run and will remain connected to STEP 7 until you shut down the controller. However, the desktop icon and the Start menu command will be removed. Without the desktop icon or Start menu command, you cannot restart the controller after it has been shut down.

Avoid the use of the invalid characters in controller names.

If you inadvertently downloaded a name that contains an invalid character, follow these steps to correct the problem:

1. Using the STEP 7 Hardware Configuration application, rename the controller to the previous valid name (the name prior to downloading the invalid name).
2. Download the configuration with the previous valid name to the PC station (even if the controller is not running).

After downloading the valid name for the controller, the desktop icon and the Start menu command reappear. You can now rename the controller to a new name that does not use invalid characters.

Character	Name															
/	Forward slash (Problematic in versions prior to STEP 7 V5.3 SP1)															
.	Period (Problematic in versions prior to STEP 7 V5.3 SP2)															
-	Hyphen (also called a dash or a minus sign) (Problematic in versions prior to STEP 7 V5.3 SP1) You cannot create a name that begins with a hyphen (-). You can, however, use a hyphen within the name of the controller. <table border="1" data-bbox="360 1507 1344 1829"> <thead> <tr> <th>Name</th> <th>Valid?</th> <th>Explanation</th> </tr> </thead> <tbody> <tr> <td>Pump-1</td> <td>Valid</td> <td>Using a hyphen in the middle of the name is valid.</td> </tr> <tr> <td>Pump1-</td> <td>Valid</td> <td>Using a hyphen at the end of the name is valid.</td> </tr> <tr> <td>-Pump1</td> <td><b>Invalid</b></td> <td>Starting a name with a hyphen is not allowed.</td> </tr> <tr> <td>-</td> <td><b>Invalid</b></td> <td>Using a hyphen as a one-character name is not allowed.</td> </tr> </tbody> </table>	Name	Valid?	Explanation	Pump-1	Valid	Using a hyphen in the middle of the name is valid.	Pump1-	Valid	Using a hyphen at the end of the name is valid.	-Pump1	<b>Invalid</b>	Starting a name with a hyphen is not allowed.	-	<b>Invalid</b>	Using a hyphen as a one-character name is not allowed.
Name	Valid?	Explanation														
Pump-1	Valid	Using a hyphen in the middle of the name is valid.														
Pump1-	Valid	Using a hyphen at the end of the name is valid.														
-Pump1	<b>Invalid</b>	Starting a name with a hyphen is not allowed.														
-	<b>Invalid</b>	Using a hyphen as a one-character name is not allowed.														



## Verifying the Configuration

A complete configuration for a PC-based automation project includes a configuration in the Station Configuration Editor and WinLC Properties that matches the configuration in STEP 7.

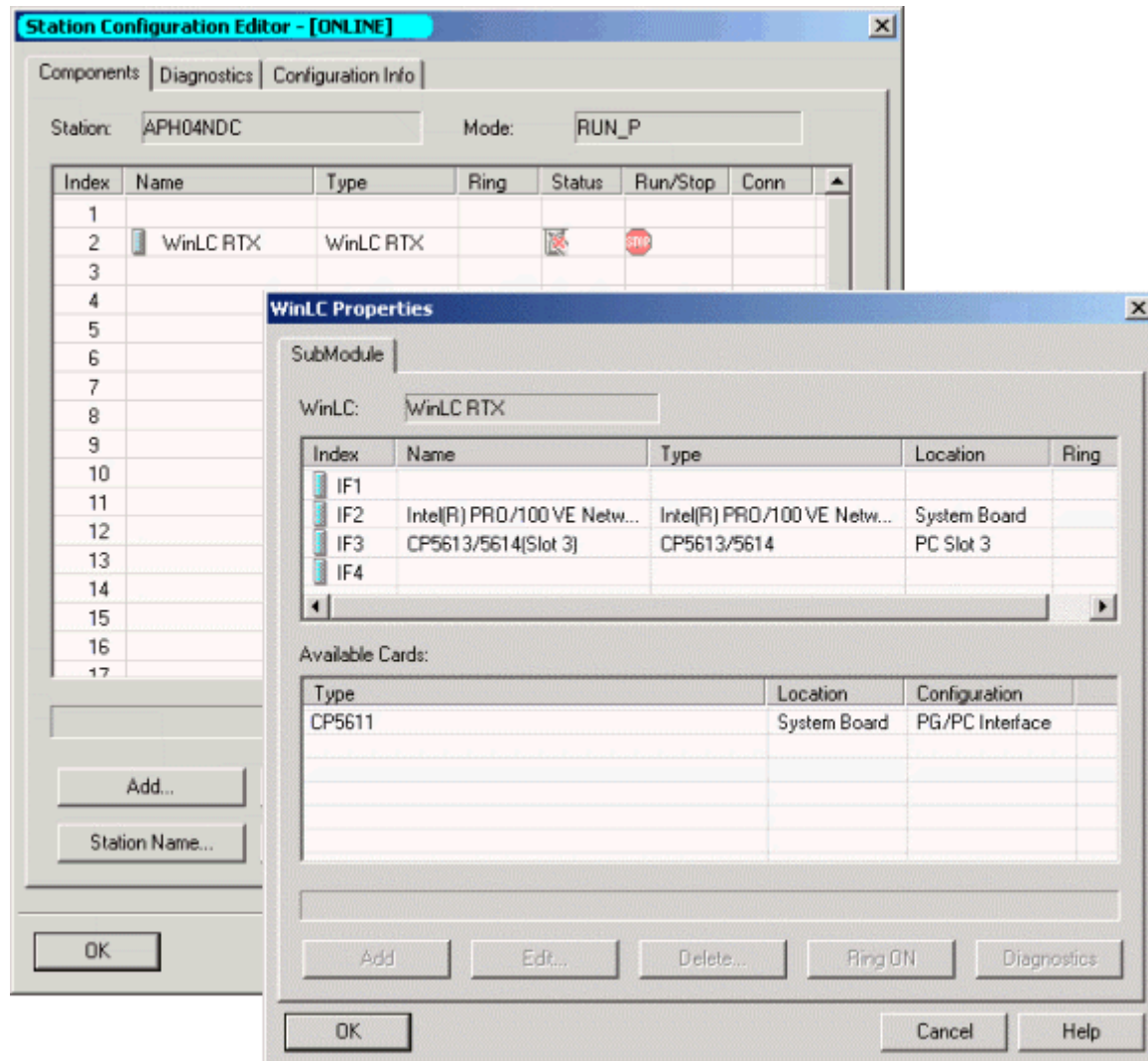
### Example PC Station Configuration

The following configuration is an example of a PC-based automation project:

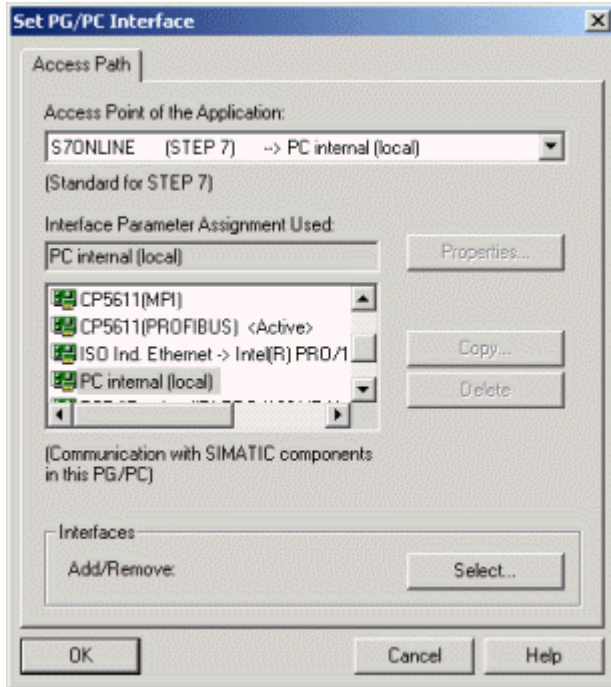
- WinLC RTX controller in index 2 of the PC station
- Industrial Ethernet card configured as WinLC RTX submodule in IF slot 2
- CP5613/CP5614 card configured as WinLC RTX submodule in IF slot 3 connected to PROFIBUS-DP I/O
- STEP 7 on same computer as WinLC RTX

The following pictures show the result of this configuration:

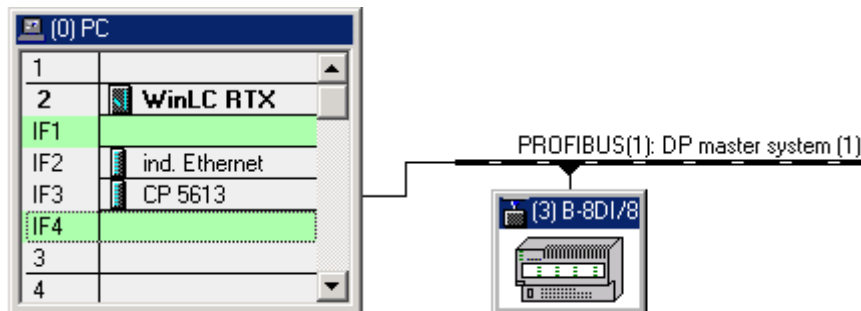
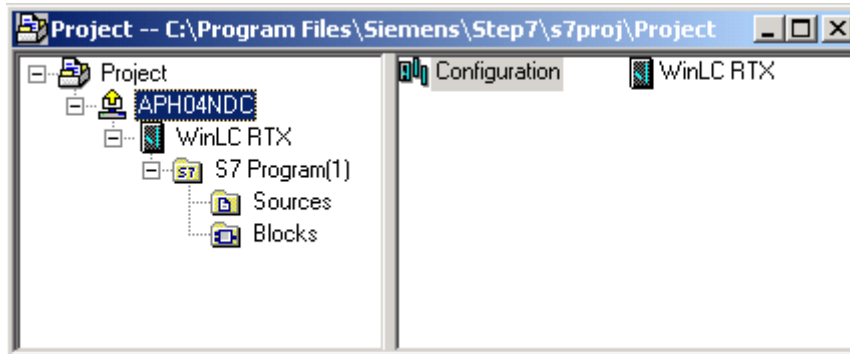
### Station Configuration Editor and WinLC Properties



## STEP 7 PG/PC Interface



## STEP 7 Hardware Configuration



# Controller Operations

## Starting and Shutting Down the Controller

The controller operates independently from the controller panel:

- Closing the panel (menu command **File > Exit**) does **not** shut down the controller.
- Shutting down the controller does **not** close the panel.

The following settings affect the starting or shutting down of the controller:

- Selecting the Autostart option
- Configuring the controller for start at PC boot

## Starting WinLC RTX

If the controller panel is not open, use one of the following methods to start the controller panel and WinLC RTX:

- Select the **Start > Simatic > PC Based Control** menu command. Then select the name of your WinLC RTX controller. (After you have downloaded the STEP 7 user program to your WinLC RTX, the name in the menu matches the name in STEP 7.)



- Double-click the desktop icon for WinLC RTX: 

If the controller panel is open, but WinLC RTX is shut down, select the **CPU > Start Controller** menu command.

## Shutting Down WinLC RTX

Select the **CPU > Shut Down Controller** menu command to shut down the WinLC RTX controller. This action does not close the controller panel. This command is only available from the controller panel when the controller is operating. After you shut down the controller, you can still change customization options.

**Note:** To shut down the Ardence RTX extensions after you shut down WinLC RTX, you must either reboot your computer or else manually stop a WinAC service that is still running. Use the **Start > Control Panel > Service** menu command to display the Windows Services dialog, and then stop the SIMATIC WinAC FeatureServer service. You can then use the RTX Properties dialog to stop the RTX extensions.

## Changing the Operating Mode of the Controller

The controller panel provides a mode selector switch that allows you to change the operating mode of the controller. Set the mode selector switch to RUN or STOP (or select the appropriate command from the **CPU** menu) to change the operating mode of the controller either to RUN mode or to STOP mode.

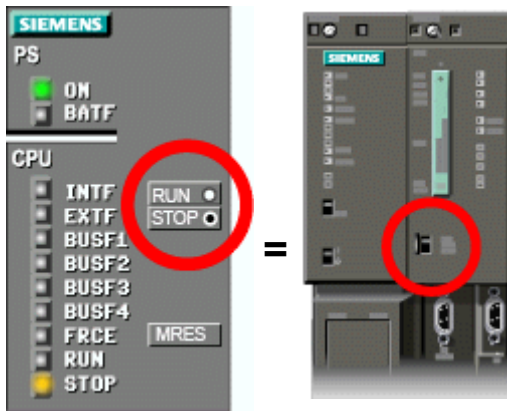
The mode selector switch positions on the controller panel correspond to the mode selector switch positions of an S7 hardware controller:

- RUN: The controller executes the STEP 7 user program.
- STOP: The controller does not execute the STEP 7 user program. Outputs are set to their safe states.

Specific controller actions are allowed or prohibited based on the operating mode.

### Operating Mode (RUN/STOP) and Status Indicators

The mode selector switch on the controller panel functions like the manual mode selector switch on a hardware S7 controller allowing you to switch between RUN and STOP mode.



For both hardware controllers and PC-based controllers, the RUN and STOP status indicators show the current operating mode of the controller. If the status indicator shows a different operating mode than the mode selector switch position, the controller has changed operating mode, possibly due to some error in the program or because you used STEP 7 to change the operating mode.



## Allowed and Prohibited Actions for each Operating Mode

The operating mode allows or prohibits access to the controller for some types of operations as shown in the following table:

Operating Mode	Description
RUN	<p>Allowed:</p> <ul style="list-style-type: none"> <li>• Uploading a program from the controller to your computer</li> <li>• Downloading a program to the controller</li> <li>• Downloading individual blocks to the controller</li> <li>• Using STEP 7 to modify program variables and to change the operating mode of the controller</li> <li>• Performing a memory reset from either the controller panel or STEP 7</li> </ul> <p>The controller automatically goes to STOP mode when you reset the memory from the controller panel. To perform a memory reset from STEP 7, you must first change the controller to STOP mode.</p> <p>Not Allowed:</p> <ul style="list-style-type: none"> <li>• Archiving and restoring a STEP 7 user program</li> </ul>
STOP	<p>Allowed:</p> <ul style="list-style-type: none"> <li>• Uploading a program from the controller to your computer or programming device</li> <li>• Downloading a program or individual blocks to the controller</li> <li>• Using STEP 7 to modify program variables</li> <li>• Performing a memory reset from either the controller panel or STEP 7</li> <li>• Archiving and restoring a STEP 7 user program</li> </ul> <p>Not Allowed:</p> <ul style="list-style-type: none"> <li>• Using STEP 7 to change the operating mode</li> </ul>

## Resetting the Memory Areas: MRES Command (CPU Menu)

The MRES (memory reset) command functions like a master reset of the controller by resetting the controller to its initial (default) state. A memory reset deletes the STEP 7 user program and the system data (configuration), and also disconnects any online communications, for example STEP 7, WinCC Flexible, PROFIBUS, or S7 communications.

Use one of the following methods to reset the memory:

- Click the MRES button on the control panel
- Select the **CPU > MRES** menu command
- Press the ALT+C+M keys

You can also use STEP 7 to perform a memory reset.

The MRES command changes the controller to STOP mode, if necessary, and then performs the following tasks:

- Deletes the entire STEP 7 user program (OBs, DBs, FCs, FBs, and the system data) from both the work memory area and the load memory area
- Resets the memory areas (I, Q, M, T, and C) to 0
- Reloads the default system configuration (for example, the size of the process-image areas, and the size of the diagnostic buffer)
- Deletes all active communications jobs (for example, TIS) and all open communications

The MRES command does not affect the submodule network addresses and does not affect the contents of the diagnostic buffer.

The STOP indicator flashes while the memory reset is in progress. After the memory has been reset, the diagnostics buffer is resized to its default size. Input (I) and output (Q) memory areas are also resized to their default sizes. After a memory reset, you may need to reconfigure these values to your own specifications.

You typically perform an MRES before downloading a new program to the controller. You **must** perform an MRES if the STOP indicator on the controller panel is flashing slowly to alert you to one of the following conditions:

- Errors were detected in the work memory area, for example, the size of the user program exceeds the work memory area
- A power cycle followed a defective state of the controller

## Using the Status Indicators

The status indicators on the control panel display the current operating mode and are helpful in troubleshooting an error condition. These indicators correspond to the LED indicators on a hardware S7 PLC.

You cannot change the status of the controller by clicking the status indicators.

If the control program reaches a break point set by the STEP 7 Program Editor, both the RUN and STOP indicators turn on while the breakpoint is active: the RUN indicator flashes, and the STOP indicator is on.

During a change from STOP mode to RUN mode, the RUN indicator flashes, and the STOP indicator is on. When the STOP indicator turns off, the outputs are enabled.

The table below describes the different status indicators for the control panel:

Indicator	Description
ON	Power supply. Lights up (solid) when you start the controller. Turns off when you shut the controller down.
BATF	Battery fault. Always off for the controller.
INTF	This indicator lights up (solid) to show error conditions within the controller, such as programming errors, arithmetic errors, timer errors, and counter errors.  If the STEP 7 user program handles the error by executing OB 80 or OB 121, the INTF indicator goes off after 3 seconds if there is no subsequent error condition.
EXTF	This indicator lights up (solid) to show error conditions that exist outside of the controller, such as hardware faults, parameter assignment errors, loss of communication or other communication errors, and I/O faults.  If the STEP 7 user program handles the error by executing OB 122, the EXTF indicator goes off after 3 seconds if there is no subsequent error condition.
BUSF1 BUSF2 BUSF3 BUSF4	These indicators light up (flashing) to identify fault conditions in the communication with the distributed I/O.  The number of the BUSF indicator corresponds to the IF number of the submodule that has a fault condition.
FRCE	This indicator is never lit. WinLC RTX does not support force jobs.
RUN STOP	Lights up (solid) to show the operating mode (RUN or STOP).  When RUN is flashing and STOP is lighted (solid), the control program has reached a breakpoint. (RUN light blinks with 0.5 Hz.)  <b>Note:</b> The RUN and STOP indicators show the actual operating mode of the controller. The RUN and STOP mode selector switch positions show the selected mode (similar to the mode selector switch position on an S7 CPU front panel), which can differ from the operating mode. For example: Changing the operating mode with STEP 7 causes the status indicators to change, but the mode selector switch does not change.

## Flashing Indicators

Flashing patterns of the RUN and STOP indicators provide additional information about the controller or the STEP 7 user program:

Indicator		Description
RUN	STOP	
flashing 2 Hz	flashing 2 Hz	The controller is in DEFECT mode. All status indicators flash.
flashing 0.5 Hz	on	The STEP 7 user program halted at a breakpoint.
flashing 2 Hz	on	A cold or warm restart is in progress. The RUN indicator continues to flash until the restart completes. The time required for the restart operation depends on the time required to execute the startup OB.
off	flashing 0.5 Hz	The controller requires a memory reset (MRES).
off	flashing 2 Hz	A memory reset (MRES) is in progress.

### Corrective Action If the STOP Indicator is Flashing Slowly

If the STOP indicator flashes slowly, the controller requires a memory reset (MRES). To recover from this condition, you must use the MRES command to reset the controller.

### Corrective Action If All Status Indicators Are Flashing

If all of the status indicators are flashing at the same time, the controller is in a defective state and has encountered an error condition that cannot be fixed by resetting the memory with the MRES command. To recover from this condition, you must perform the following tasks:

1. Shut down the controller.
2. Restart the controller. The STOP indicator flashes with the RUN indicator off.
3. Use the MRES command to reset the memory.
4. Use STEP 7 to download the control program and system configuration, or to restore an archived control program.

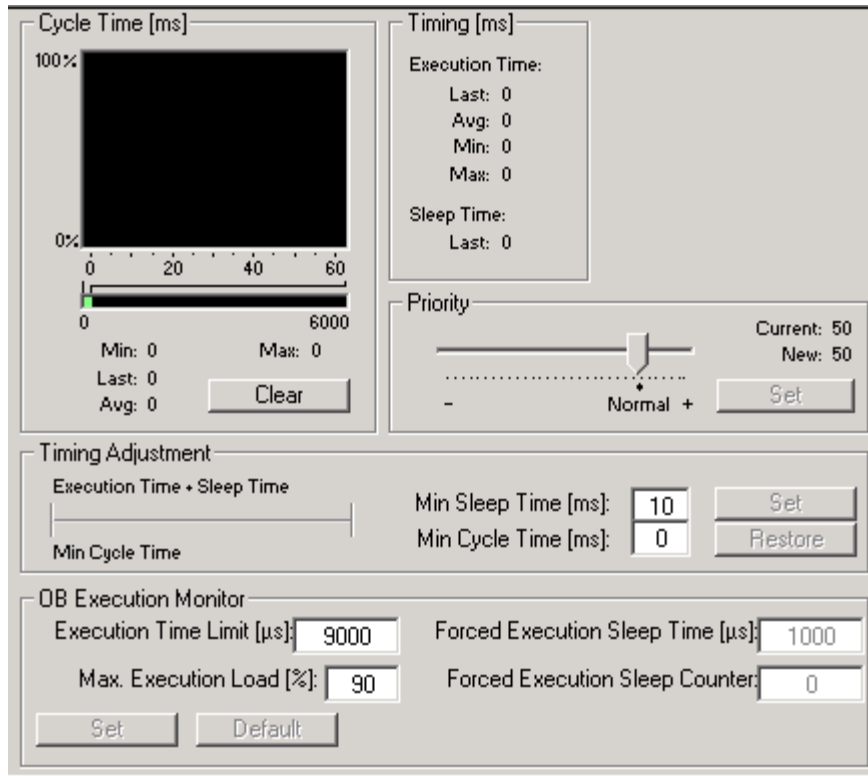
If either shutting down or restarting the controller does not resolve the problem, you may need to reboot your computer.

## Using the Tuning Panel

You can use the tuning panel to view and adjust the current performance of the controller. The tuning panel displays information about the scan cycle, such as the execution time and the sleep time. By adjusting these values, you can tune the performance of the controller.

**Note:** The tuning panel is designed for adjusting the parameters and verifying the performance for WinLC RTX. Because the tuning panel causes an additional load on the computer resources, do not leave the tuning panel open during normal operation of WinLC RTX.

To open or close the tuning panel, select the **CPU > Tuning Panel** menu command. WinLC RTX opens the tuning panel, as shown below. Click a region on the tuning panel picture for more information about its functionality.



Values other than the minimum cycle time are unique to WinLC RTX and are not stored in the system configuration. Using the tuning panel to enter a value for minimum cycle time does not change the configuration of the controller.

Changing the controller from STOP mode to RUN mode resets the minimum scan cycle time parameter to the value that you configured in STEP 7. To make any changes made with the tuning panel permanent, you must use the STEP 7 Hardware Configuration tool.



**Warning**

Variation in the execution time or response time of the STEP 7 user program could potentially create a situation where the application being controlled can operate erratically and possibly cause damage to equipment or injury to personnel.

If the controller does not provide sufficient sleep time for the other applications to run, the computer can become unresponsive to operator input, or the controller and other applications can operate erratically. In addition, the execution of the STEP 7 user program can experience non-deterministic behavior (jitter) such that execution times can vary and start events can be delayed.

Always provide an external emergency stop circuit. In addition, always tune the sleep time and manage the performance of the controller so that your STEP 7 user program executes consistently.

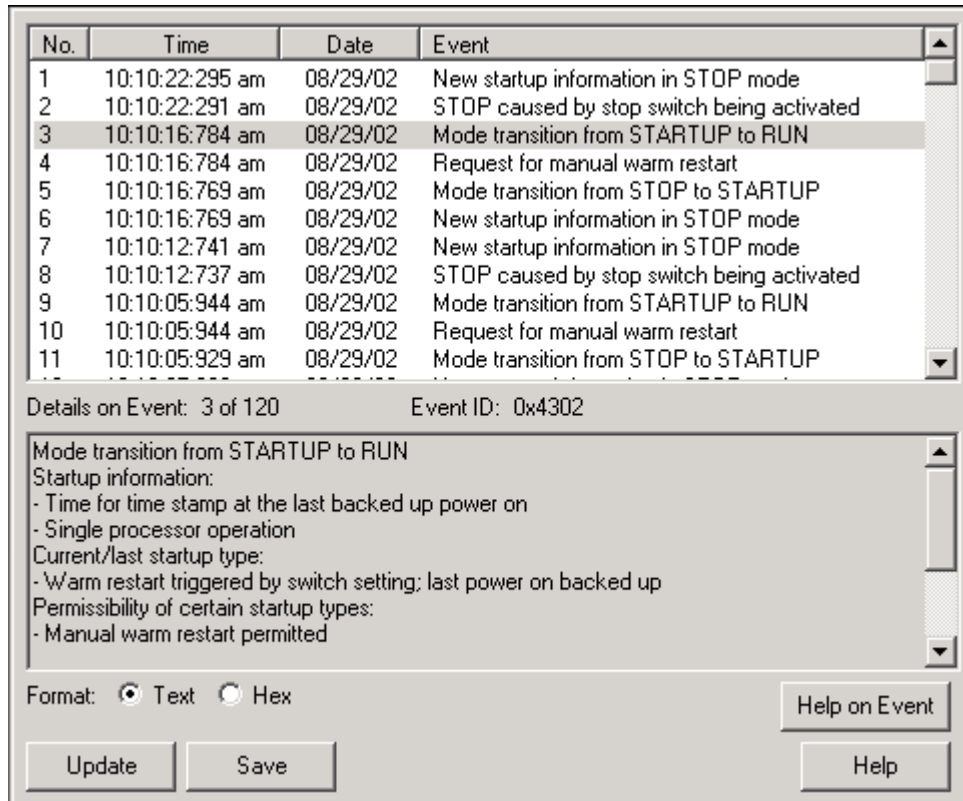
The tuning panel contains the following functional areas:

Area	Description
Cycle Time	<p>This area provides a histogram of execution times of the scan cycle over a 60-ms range. This histogram tracks minimum (shortest) and maximum (longest) scan times, as well as the percentage of scans that fall in various ranges of scan times. To delete the historical data and start a new histogram, click Clear. A STOP-to-RUN transition also resets the Cycle Time display, as does closing and reopening the tuning panel.</p>
Timing	<p>This read-only field displays the following information about the scan cycle:</p> <p><b>Execution Time</b> displays the execution time for the last (most current) scan, the average scan cycle time, the shortest (minimum) scan cycle time, and the longest (maximum) scan cycle time.</p> <p><b>Sleep Time</b> displays the amount of sleep time for the last (most current) scan.</p>
Priority	<p>Use this slider to set the priority level for the execution of WinLC RTX relative to other applications running on your computer.</p> <p>Because WinLC RTX runs at a higher priority than any Windows application, you change the priority for WinLC RTX only if you are running other RTX applications.</p> <p>Setting the priority higher means that the operating system responds to WinLC RTX before executing lower-priority tasks. This results in less jitter in the start times and execution time of the OBs in your program.</p>
Timing Adjustment	<p>Use these fields to tune the scan cycle by entering values for the minimum sleep time and the minimum cycle time. These parameters determine the amount of sleep time that is added at the end of the free cycle.</p> <p>Click the Set button to apply these values. Click the Restore button to reset the values to those currently being used by the controller. After you apply new values, the panel stores these values for the controller and you can monitor the effect on the execution of your control program.</p> <p>To ensure that the minimum cycle time controls the sleep time for the controller, you must configure the scan cycle monitoring time and minimum scan cycle time parameters on the Cycle/Clock Memory tab of the Properties dialog box in STEP 7. Set the minimum scan cycle time to a value less than the value for scan cycle monitoring time. (The default scan cycle time is 6 seconds.)</p>

## Using the Diagnostic Buffer

Select the **CPU > Diagnostic Buffer** menu command to display the SIMATIC Diagnostic Buffer.

The Diagnostic Buffer allows you to view system diagnostic information without using the SIMATIC STEP 7 programming software. It consists of an upper panel that displays an event list and a lower panel that displays specific event details.



The diagnostic buffer is implemented as a ring buffer that contains single event entries. The events are displayed in descending order by time, with the most recent event at the top. If the ring buffer is full, a new event overwrites the oldest entry in the buffer.

The Diagnostic Buffer displays the following information:

**Event List** (upper panel): A list of all the events in the diagnostic buffer. The following information is shown for each diagnostic event:

- The number of the entry
- The date and time of the event
- A brief description of the event

**Event ID** (between the upper and the lower panels): Displays the ID number of a selected event.

**Event Details** (lower panel): Displays the event details in either text or hexadecimal format.

If you have chosen Text format, the following details about a selected event appear in the lower panel:

- A brief description
- Additional information, depending on the event, such as the address of the instruction that caused the diagnostic event and the mode transition that was caused by the event
- The event state (incoming or outgoing)

If a single parameter of text cannot be identified, the diagnostic buffer displays the string "###". If no text exists for new modules or new events, the event numbers and the single parameters are displayed as hexadecimal values.

If you have chosen Hexadecimal format, the hexadecimal values of the selected event appear in the lower panel.

### **Sorting Events (upper panel)**

You can sort the events listed in the upper panel by clicking the specific column:

- Number (determined by time and date)
- Event description

### **Choosing Format (lower panel)**

You can display the diagnostic information in the lower panel in text or hexadecimal (Hex) format. In Hex format, the hexadecimal values of the 20 bytes of the selected event are displayed. To select the format:

- Click Text to display the event details in text format.
- Click Hex to display the hexadecimal values of the event.

### **Saving the Diagnostic Buffer**

To save a text file containing the event list and the detailed information for every event, click the Save button. The text file contains the information either in text or in hexadecimal format.

### **Displaying Help**

To display help on the diagnostic buffer, click the Help button. To display help on a specific event:

1. Select the event in the upper panel.
2. Click the Help on Event button.



## Archiving and Restoring STEP 7 User Programs

The Archive command enables you to save the configuration and STEP 7 user program to an archive file (\*.wld). The archive file allows you to easily restore the configuration and STEP 7 user program for the controller.

**Note:** You must update a project or program from a release of WinLC V3 or earlier to a WinLC RTX V4 project. Use STEP 7 to create a new project and to create a new configuration for the WinLC RTX V4 controller and any submodules.

You can only archive or restore a STEP 7 user program when the controller is in STOP mode. You cannot archive or restore a STEP 7 user program when the controller is in RUN mode or is shut down.

The archive file functions like the removable memory cartridge (EEPROM cartridge) of an S7 CPU; however, it differs in that the controller does not automatically restore the archive file after a memory reset (MRES). You must manually restore the archive file.

### Creating an Archive File

An Archive file stores the current STEP 7 user program, the current system configuration, and the current values of the DBs. The Archive file does **not** store the configuration of the PC station.

To create an Archive file, select the **File > Archive** menu command. This command displays the Save As dialog, which allows you to give a name to the file. The controller then creates the archive file with the extension .wld.

You can also use the SIMATIC Manager of STEP 7 to create an Archive file. Select the **File > Memory Card File > New** menu command.

### Restoring an Archive File

When you restore an archive file, you reload the STEP 7 user program and the configuration for the controller. You can only restore archive files of extension .wld.


Before you can restore an archive file, you must set the controller to STOP mode. Use the following procedure to load an archived configuration and STEP 7 user program:

1. Click the STOP button to place the controller in STOP mode.
2. Select the **File > Restore** menu command.
3. Select the specific archive file to restore and click OK.

## Closing the Controller Panel

Select the **File > Exit** menu command to close the control panel.

**Note:** Closing the controller panel does not shut down the controller or affect the operating mode.

An icon  is displayed in the Windows taskbar whenever the controller is operating. When the controller is operating and the controller panel is closed, you can double-click this icon to open the controller panel.

## WinLC RTX Operation following a Windows Blue Screen

WinLC RTX supports OB 84 (CPU Hardware Fault), which allows you to initiate orderly shutdown of your process in case a Windows Blue Screen occurs while WinLC RTX is operating. During a blue screen, communication interfaces configured as submodules continue to function. If WinLC RTX can still operate after Windows has initiated the system shutdown procedure, and the memory used by the real-time subsystem is not corrupted, one of the following occurs:

- If WinLC RTX is in RUN mode and the control program includes OB 84, WinLC RTX starts OB 84 and continues in RUN mode until the STEP 7 user program calls SFC 46 (STP) to place the controller in STOP mode. Windows does not complete its system shutdown until after WinLC RTX transitions to STOP mode, either from the SFC 46 call or from a change to STOP mode initiated from a programming device or communication partner accessing WinLC RTX through a submodule communication interface.
- If WinLC RTX is in RUN mode and the control program does not include OB 84, WinLC RTX transitions to STOP mode and then Windows completes its system shutdown.
- If WinLC RTX is not in RUN mode, Windows completes its system shutdown.

The operation of WinLC RTX during a blue screen can be affected by SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, or SFC 85.

You can configure both Windows and WinLC RTX to automatically restart following a blue screen.

The following restrictions apply when Windows is shutting down:

- The WinLC RTX controller panel is unavailable.
- Some system functions are disabled, including SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, and SFC 85.
- Block operations fail, returning an error code.
- Communication with Windows applications is unavailable; however, communication with the submodules of WinLC RTX is not affected.
- Communication with external systems (such as HMI devices or programming devices) is only available if the network is connected to a configured submodule of WinLC RTX.
- Restarting the computer followed by restarting WinLC RTX initializes all of the program variables to their default values and empties the diagnostic buffer.

## Considerations for SFC 22, SFC 23 and SFC 82 to 85

If a Windows Blue Screen occurs when WinLC RTX is in RUN mode, WinLC RTX attempts to stay in RUN mode and initiates OB 84; however, the operation of WinLC RTX during a blue screen can be adversely affected by SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, or SFC 85.

Under most circumstances, SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, and SFC 85 return error code 8092 in the event of a Windows Blue Screen. Applications that need to continue operating after a blue screen can check for this error code. If however, one of these SFCs is in a Windows call at the time of the blue screen, the SFC is not able to return the 8092 error code and WinLC RTX cannot initiate OB 84.



### Warning

Certain SFCs, if active at the time of a Windows failure, can cause either WinLC RTX or other functions to become unresponsive and lock up:

- If either SFC 22, SFC 23 or SFC 85 is in a call to a Windows function at the time of the blue screen, the SFC cannot return from the SFC call and WinLC RTX fails to maintain control of the process. If this occurs, the I/O watchdog operation disables the inputs and outputs.
- If SFC 82, SFC 83, or SFC 84 is in a call to a Windows function at the time of the blue screen, WinLC RTX attempts to stay in RUN mode (continuing to control the process), but background operations including some communication functions can lock up. Setting WinLC RTX to STOP mode, whether by program action or by user intervention from a remote system, can affect the shut-down sequence of the computer.

A blue screen that results in locking up either the controller or background functions can cause damage to process equipment or injury to personnel if you do not take proper precautions in designing your STEP 7 user program.

If your process application needs to survive a Windows failure, call these SFCs (SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, or SFC 85) only when initializing (during the execution of OB 100 or OB 102) or during non-critical parts of the control process.

## WinLC RTX Restart Behavior after a Blue Screen

If Windows is configured to reboot automatically after a Windows Blue Screen, WinLC RTX starts if it is configured to start at PC boot.

When WinLC RTX restarts, it uses the program as it was last downloaded and executes OB 100 (not OB 102) if it is present. WinLC RTX always executes OB 100 after a blue screen, even if OB 102 "Cold Start" is configured in the STEP 7 Hardware Configuration. WinLC RTX starts OB 100 with event 1382 (hex). The current/last startup type is shown in the diagnostic buffer as "automatic warm reboot after non-backup power on with system memory reset".

You can program OB 100 to respond to event 1382. For more information, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

To configure automatic reboot for Windows, follow these steps:

1. Open the Windows Control Panel and double-click System.
2. From the Advanced tab of the System Properties dialog, click the Settings button for Startup and Recovery.
3. Select the "automatically restart" checkbox.
4. Click OK on the Startup and Recovery dialog and the Systems Properties Dialog.

## Storing Retentive Data

WinLC RTX stores the following operational information on your hard disk:

- Load memory: contains the system data (configuration of the controller) and the initial values of the blocks of the STEP 7 user program.
- State of the controller: includes the last transition of the operating mode (STOP, RUN, or STARTUP) for the controller and the setting for the mode selector switch (STOP or RUN).
- Power-down state of the controller: as saved during the shutdown process, includes the contents of the diagnostic buffer, the current values for the retentive memory areas of the controller (such as timers, counters and bit memory), and the current values for the data blocks (Work memory).

WinLC RTX saves this information on your hard disk during operation and uses this information when starting up the controller.

### Load Memory

When you download the STEP 7 user program, WinLC RTX saves the program blocks and system data in the Load memory area. These blocks include the initial values for the process variables used by the STEP 7 user program.

SFC 82 (CREA\_DBL) allows you to create new blocks in Load memory during the execution of the STEP 7 user program. You can use SFC 84 (WRIT\_DBL) to modify these blocks. The blocks that SFC 82 creates are stored in the Load memory at the time that SFC 82 runs.

**Note:** Data blocks (DBs) created by SFC 22 (CREAT\_DB) and SFC 85 (CREA\_DB) are not saved in the Load memory. These DBs are stored only in the Work memory.

### Retentive Data

When you configure WinLC RTX in STEP 7, you can determine the ranges of retentive data for the timers (T memory), counters (C memory), bit memory (M memory), and data blocks (DBs).

WinLC RTX saves the retentive memory areas as part of the power-down state. If the power-down state was saved and the controller performs a warm restart (OB 100), WinLC RTX loads the retentive memory areas, including the values of the DBs stored in Work memory.

### State of the Controller

WinLC RTX stores the current operational status of the controller on the hard disk and the status of the following events:

- Whenever the controller changes operating mode (RUN to STOP, STOP to STARTUP, or STARTUP to RUN), WinLC RTX stores the state of the controller on the hard disk to show the latest transition.
- Whenever the mode selector switch on the controller panel changes (STOP or RUN), WinLC RTX stores the state of the mode selector switch on the hard disk to show the latest action.

## Power-Down State

Under normal circumstances, WinLC RTX saves the current state of the controller and the retentive data on your hard disk, including the DBs from Work memory, when you shut down the controller. (The shutdown process can be initiated by the **CPU > Shut Down Controller** menu command or by shutting down the Windows operating system, either by user action or by UPS signal.) WinLC RTX stores the following information from the Work memory:

- Values for the retentive data in the S7 memory areas (such as T, C, M, and DB)
- Diagnostic buffer

**Note:** The power-down state (which includes the diagnostic buffer) is **not** saved in cases when WinLC RTX terminates abnormally, such as when the computer loses power (either by turning off the power or by power failure), or when WinLC RTX is not able to write to the hard disk, such as following a Windows crash ("Blue Screen").

WinLC RTX saves the power-down state during shutdown. After WinLC RTX starts up, it loads the power-down state from the hard disk and then deletes it from hard disk (to avoid problems in case of an abnormal termination of the controller).

## Shutting Down the Controller Saves the Retentive Data

The following table shows the actions that cause WinLC RTX to save the retentive data.

Retentive Data	Action That Causes WinLC RTX to Save This Data
Load memory (blocks and initial values of the STEP 7 user program, DBs, and system data)	Downloading the STEP 7 user program from STEP 7
State of the controller	Changing the operating mode (RUN to STOP, or STOP to RUN), either from STEP 7 or from the controller panel Successfully shutting down WinLC RTX
Power-down state	Successfully shutting down WinLC RTX
Work memory	Successfully shutting down WinLC RTX
Retentive memory areas (T, C, M, and DBs)	Successfully shutting down WinLC RTX
Diagnostics buffer	Successfully shutting down WinLC RTX

## Reloading Memory Areas on Startup

Upon startup, WinLC RTX searches for the stored power-down state to determine whether the controller had been shut down correctly and performs the following tasks:

- Reloads the downloaded blocks of the STEP 7 user program from the Load memory.
- If the power-down state **is** found (showing that the controller **had** been shut down correctly):  
Updates the work memory from the power-down state and loads the retentive data with the values stored at the time that the controller was shut down
- If **no** power-down state is found (showing that the controller had **not** been shut down correctly) :  
Restores the work memory to its initial state from Load memory (as downloaded from STEP 7)
- Restores the state of the controller, based on the saved operating mode and the Autostart configuration, and resets the mode selector switch setting on the controller panel.

If WinLC RTX cannot read some element of the Load memory, state of the controller, or the power-down state, WinLC RTX starts an unconfigured (empty) controller.

**Note:** WinLC RTX V4.2 cannot read the stored load memory or power-down state saved from a previous release of WinAC RTX or WinAC Basis. You can restore a STEP 7 user program and configuration that was archived from a previous release, but WinLC RTX V4.2 cannot access retentive data from a previous release.

## Reloading Memory Following a Successful Shutdown of the Controller

If the power-down status was successfully saved when shutting down the controller, WinLC RTX reloads the operational data for the controller:

- WinLC RTX reloads data stored in the power-down state during startup. This includes the retentive S7 memory areas, the current values of the process blocks (work memory), and the contents of the diagnostic buffer.  
**Note:** If you configured the controller for a cold restart (OB 102), WinLC RTX resets the process variables and S7 memory areas to the initial values from the Load memory.
- Based on the Autostart settings, WinLC RTX sets the state of the controller to either STOP mode or RUN mode. In case of a Windows Blue Screen, WinLC RTX sets the state of the controller to the state before the Windows Blue Screen occurred. Although the controller performed a "normal" RUN-to-STOP transition, WinLC RTX is unable to save the state of the controller during a Windows Blue Screen.  
**Note:** If the controller was configured for Autostart, WinLC RTX generates a startup event that identifies the type of startup: buffered or unbuffered. (An unbuffered startup is like reloading the control program from an EPROM file.) You can program OB 100 to read this start event. For an unbuffered startup, the variable OB100\_STOP at address LW6 is set to W#16#4309.
- WinLC RTX sets the mode selector switch to the setting when the state of the controller was last saved.

After resetting these values and completing startup, WinLC RTX deletes the power-down state from the retentive memory file.

## Reloading Memory if the Controller Was Not Shut Down Properly

If the controller was not shut down properly, WinLC RTX does not create the power-down state.

**Note:** If the power-down state was not created, the diagnostic buffer is not saved. When you restart the controller, the diagnostic buffer is empty.

If the power-down state was **not** saved when shutting down the controller, WinLC RTX performs the following tasks when restarting the controller:

- Reads the Load memory and reloads the system configuration, the process variables and S7 memory areas to the initial values configured in STEP 7.
- Reads the state of the controller and performs an unbuffered startup. (An unbuffered startup is like reloading the control program from an EPROM file. WinLC RTX generates a startup event that you can program the OB 100 to read.) Based on the Autostart settings, WinLC RTX sets the controller to either STOP mode or RUN mode.
- Sets the mode selector switch to the setting when the controller was shut down.

## Encountering Problems When Starting the Controller

If WinLC RTX cannot read (or encounters an error in reading) some element of the retentive memory area (Load memory, state of the controller, or power-down state), WinLC RTX starts with an unconfigured (empty) controller. In this case, the controller is set to STOP mode with the mode selector switch set to STOP, and the system data and control program are deleted.

Possible causes for this problem include a hardware error in your computer, or a partial block in the Load memory area caused by an error that occurred when WinLC RTX was writing a block to the Load memory.

To recover from this condition, you must reload your control program and system data from STEP 7.

**Note:** The mode selector switch of the controller is set to STOP mode. You can download the control program and system data from a remote computer, but you cannot use the remote computer to set the controller to RUN mode. You must go to the local computer for WinLC RTX and set the mode selector switch to RUN to place the controller in RUN mode.

## Starting the Controller After a Windows Blue Screen

If the controller was in RUN mode at the time of the shutdown and is configured for Autostart, WinLC RTX restarts in RUN mode. If OB 84 (CPU Hardware Fault) had responded to a Windows Blue Screen and had placed the controller in STOP mode before shutting down, WinLC RTX still starts in RUN mode because WinLC RTX **could not** save the state setting for the controller to the hard disk during the shutdown caused by the Windows crash.

If you do not want the controller to restart in RUN mode after a Windows Blue Screen, you must include code in the startup OB (OB 100 or OB 102) to detect that WinLC RTX terminated without saving the power-down state, and to set the controller to STOP mode when restarted. A value of 0010 xxxx in bits 7 - 0 of the startup OB variable OB\_STR\_INFO indicates that the power-down state is not available on disk.

## Using SFCs to Retain Data

You can use SFC 82 (CREA\_DBL), SFC 83 (READ\_DBL), and SFC 84 (WRIT\_DBL) to save data at significant events in your process. For example, you may want to store the recipe values into Load memory when changing a recipe without downloading new blocks for the STEP 7 user program.

SFC 82 and SFC 84 modify the data for the STEP 7 user program that is stored in the Load memory. Saving the blocks to Load memory (instead of keeping the values in Work memory) ensures that these blocks are available even if WinLC RTX cannot save the power-down state when shutting down the controller.

**Note:** You must consider the possibility of a Windows Blue Screen when using SFC 22, SFC 23, SFC 82, SFC 83, SFC 84, or SFC 85.

When executed from the STEP 7 user program, SFC 82 (CREA\_DBL), SFC 83 (READ\_DBL), and SFC 84 (WRIT\_DBL) create and update blocks that are stored as part of your STEP 7 user program in Load memory.

SFC 82, SFC 83, and SFC 84 are asynchronous SFCs that run in the background.

**Note:** If you call SFC 82, SFC 83, or SFC 84 from the startup OB (OB 100 or OB 102), WinLC RTX executes these SFCs synchronously. This differs from the operation of a hardware PLC.

Like the other asynchronous SFCs, SFC 82, SFC 83, and SFC 84 are typically long-running SFCs that can require a relatively long time to complete. (The time for the SFC call itself will be short, but the actual operation for the SFC will be executing in the background.) In order to use asynchronous SFCs, you must allow sufficient sleep time to allow WinLC RTX to process the SFCs without encountering jitter.

**Note:** Do not use a polling loop that looks for the completion of an asynchronous SFC, especially for SFC 82, SFC 83, or SFC 84. Because the asynchronous SFC is being executed in the background, having your control program loop until the SFC finishes will extend the execution of the OB that is performing the polling loop and can cause jitter.

### Caution

Whenever your control program calls SFC 82, SFC 83, or SFC 84, the SFC reads or writes data to the disk. If you call these SFCs every scan (such as from OB 1) or from a cyclical OB that is executing rapidly, the constant reading or writing to the disk can cause the disk to fail or can add jitter.

You should only call SFC 82, SFC 83, or SFC 84 to record a significant process event, such as a change of recipe.

## Uninterruptible Power Supply (UPS)

You can use a UPS system to provide emergency power for your computer. A UPS system helps ensure that WinLC RTX shuts down correctly and saves the power-down state in case of a power failure.

Refer to the manufacturer's documentation for your computer and your UPS system.

Microsoft Windows provides a dialog for configuring the UPS for your computer:

1. Select the **Start > Settings > Control Panel** menu command to display the control panel.
2. Double-click the Power Options icon to display the Power Options Properties dialog.
3. Click the UPS tab and configure the parameters for your UPS system.
4. Click Apply or OK to set the UPS properties.



## Customizing and Security Options

To open the Customize dialog box, select the **CPU > Options > Customize** menu command. The tabs of the dialog box allow you to customize the controller panel as follows:

### General

Select **Always On Top** to display the controller panel on top of all other open windows.

### Language

The language field displays the current display language for the controller panel.

The language select list displays the installed languages for the controller panel. Click a language selection to change the controller panel display language.

**Note:** To install the languages that are available for the controller panel, run the setup program and select the languages from the dialog.

### AutoStart

Select **Autostart CPU** to set the autostart feature. The autostart feature allows the controller to start automatically in RUN mode under the conditions described in *Selecting the Autostart Feature*.

### Selecting the Language

You can change the display language for the controller panel menus and online help.

To change the display language, follow these steps:

1. Select the **CPU > Options > Customize** menu command to display the Customize dialog.
2. In the Customize dialog, select the Language tab.
3. Select the language for the controller panel.
4. Click Apply to change the language.
5. Click OK to close the Customize dialog.

The controller panel automatically changes to the selected language.

### Selecting the Autostart Feature

The panel provides an Autostart feature that when enabled causes the controller to start in the same operating mode as when previously shut down:

- If the controller was in RUN mode when shut down, the controller restarts in RUN mode.
- If the controller was in STOP mode when shut down, the controller restarts in STOP mode.

If the Autostart feature is **not** enabled, the controller always starts in STOP mode.

Use the following procedure to enable the Autostart feature:

1. Select the **CPU > Options > Customize** menu command to display the Customize dialog.
2. In the Customize dialog, select the Autostart tab.
3. Select the Autostart CPU option for the Startup Mode.
4. Click Apply to enable the Autostart feature and click OK to close the Customize dialog.

## Setting the Security Options

Select the **CPU > Options > Security** menu command to change security options. The controller panel displays the Access Verification dialog. You must enter your password in this dialog in order to make any changes to the security settings for the controller.

**Note:** The default password is an empty field containing no characters. To enter the default password, press the Enter key.

### Security Level

The Security dialog allows you to set levels of password security that limit access to the controller. The following security access options are provided:

- **Password:** When you select Password, certain controller panel operations such as changing the operating mode, opening the tuning panel, and archiving and restoring a STEP 7 user program, require that the user enter a password.
- **Confirmation:** When you select Confirmation, operating mode changes require that the user acknowledge a confirmation dialog box.
- **None:** When you select None, no confirmation or password is required.

### Password Prompt Interval

You can set the Password Prompt Interval to a time interval of your choice, from 0 to a maximum of 23 hours, 59 minutes. After you have entered your password, you are not prompted for it again until this time interval has expired. The default setting of 0 means that you must enter a password for each protected operation.

Shutting down and starting the controller does not affect the expiration of the Password Prompt Interval; however, it is reset whenever you shut down the controller panel. The next time you start the controller panel and access a password-protected function, you will be prompted for password entry.

### Change Password

Click the Change Password button to display the Change Password dialog.

**Note:** If you create a password, but set the security level to None (disabling the password), you still need to enter the configured password in the Access Verification dialog before you can access the Security dialog box again.



#### Warning

Running the controller without confirmation or password protection increases the risk that an operator may change the controller mode inadvertently, which could cause the process or equipment to operate unpredictably, resulting in potential damage to equipment and/or death or serious injury to personnel.

Exercise caution to ensure that you do not inadvertently change the operating mode, or permit unauthorized persons to access the machine or process. Always install a physical emergency stop circuit for your machine or process.

## Changing the Password

The Change Password dialog allows you to change the current password.

**Note:** The default password is an empty field containing no characters. To enter the default password, press the Enter key.

Use the following procedure to change the password:

1. In the Old Password field, enter the old password.
2. In the New Password field, enter the new password (maximum length 12 characters).
3. In the Confirm New Password field, enter the new password again.
4. Click OK to apply all the changes made in this dialog.

To subsequently access the security options, you must enter the password at the Access Verification dialog.

## Startup Options for the Controller

### Starting the Controller at PC Boot

By default, you must start the controller manually after the computer reboots. You can, however, register the controller to start during the Windows boot sequence prior to user login.

**Note:** To configure the controller for starting in the same operating mode (STOP or RUN) as when previously shut down, use the Autostart feature.

### Registering the Controller for Start at PC Boot

To register the controller to start during the Windows boot sequence, follow these steps:

1. Shut down the controller.
2. Select the **CPU > Register Controller for Start at PC Boot** menu command.

WinLC RTX will now start whenever you boot your computer.

### Unregistering the Controller for Start at PC Boot

To unregister the controller for starting automatically, follow these steps:

1. Shut down the controller.
2. Select the **CPU > Unregister Controller for Start at PC Boot** menu command.

WinLC RTX will now **not** start during the boot sequence. To start WinLC RTX, you must manually start the controller.

## Setting the Restart Method

The restart method determines which startup OB the controller executes whenever a change from STOP mode to RUN mode occurs. The startup OB allows you to initialize your STEP 7 user program and variables. WinLC RTX supports two restart methods:

- **Warm restart:** The controller executes OB 100 before starting the free cycle (OB 1). A warm restart resets the peripheral inputs (PI) and changes the peripheral outputs (PQ) to a pre-defined safe state (default is 0). The warm restart also saves the current value for the retentive memory areas for the memory bits (M), timers (T), counters (C), and data blocks (DBs).
- **Cold restart:** The controller executes OB 102 before starting the free cycle (OB 1). Like a warm restart, a cold restart resets the peripheral inputs (PI) and changes the peripheral outputs (PQ) to a pre-defined safe state (default is 0). However, a cold restart does **not** save the retentive memory (M, T, C, or DB), but sets these areas to their default (initial) values.

You use STEP 7 to configure the default restart method for the controller. The default restart method is stored in the configuration (system data) for the controller that you download with your STEP 7 user program. WinLC RTX uses this restart method when WinLC RTX is configured for Autostart and returns to RUN mode following a power cycle.

Whenever you click (using the left mouse button) the RUN mode selector switch on the panel to change from STOP mode to RUN mode, WinLC RTX performs a warm restart, executing OB 100.

To select a specific restart method, choose one of the following options to change the controller from STOP mode to RUN mode:

- Select the **CPU > RUN** menu command to change the controller from STOP to RUN mode.
- Right-click (using the right mouse button) the RUN mode selector switch position.

Both of these actions display the Restart Method dialog that allows you to select either a warm or cold restart.

**Note:** If you have configured the confirmation security option, you must acknowledge a confirmation dialog before the controller panel displays the Restart Method dialog.

If you have configured the password security option and the password prompt interval is either 0 or has expired, the controller panel displays the Access Verification dialog for you to enter the password. After verifying successful password entry, the controller panel displays the Restart Method dialog.

After executing OB 100 (warm restart) or OB 102 (cold restart) according to your selection, the controller executes the free cycle (OB 1).



# STEP 7 Operations and Components

## Using STEP 7 with the Controller

STEP 7 provides programming and configuration tools for working with WinLC RTX. You perform the following tasks with STEP 7:

- Define the controller and DP I/O configuration through the STEP 7 Hardware Configuration tool
- Develop a STEP 7 user program using any of the STEP 7 control programming languages
- Configure operational parameters and I/O addresses for the controller
- Download your configuration and STEP 7 user program to the controller

Refer to your STEP 7 documentation for additional information.

## Configuring the Operational Parameters for the Controller

STEP 7 provides a Hardware Configuration application for configuring the operational parameters for the controller. This configuration is then stored in various SDBs in the System Data container.

After you download the System Data, the controller uses the configured parameters for the following events:

- Whenever you start the controller
- On the transition to RUN mode (if you modified the hardware configuration online while the controller was in STOP mode)

To configure the operational parameters from the STEP 7 Hardware Configuration application, right-click the controller entry in the station window and select Object Properties. From the Properties dialog, you configure the operational parameters.

## Accessing Operational Parameters

To configure any of these operational parameters in STEP 7, open the SIMATIC Manager and follow these steps:

1. In the SIMATIC Manager, select the PC station.
2. Click the Configuration icon.
3. Right-click the controller in the station window and select Object Properties.
4. Click the tab with the name of the parameter that you want to configure (such as Cyclic Interrupt) and enter the appropriate values in the dialog.
5. Click OK to confirm your configuration.

Refer to your STEP 7 documentation for specific information about configuring the controller properties and the operational parameters.

## Logic Blocks Supported by WinLC RTX

Like the other S7 controllers, WinLC RTX provides several types of logic blocks for processing the user program: organization blocks (OBs), system functions (SFCs), and system function blocks (SFBs). These blocks are an integral part of WinLC RTX.

Organization Block (OB)	System Function (SFC)	System Function Block (SFB)
OB 1	SFC 0 to SFC 6	SFB 0 to SFB 5
OB 10	SFC 9 to SFC 15	SFB 8 and SFB 9
OB 20	SFC 17 to SFC 24	SFB 12 to SFB 15
OB 30 to OB 38	SFC 26 to SFC 34	SFB 22 and SFB 23
OB 40	SFC 36 to SFC 44	SFB 31 to SFB 36
OB 52 to OB 57	SFC 46 and SFC 47	SFB 52 to SFB 54
OB 61 and OB 62	SFC 49 to SFC 52	SFB 65001 and SFB 65002
OB 80, OB 82 to OB 86, and OB 88	SFC 54 to SFC 59	
	SFC 62 and SFC 64	
OB 100 and OB 102	SFC 78 to SFC 80	
OB 121 and OB 122	SFC 82 to SFC 84	
	SFC 85 and SFC 87	
	SFC 126 and SFC 127	

### Additional S7 Blocks

In addition to these system blocks, you can use these other S7 blocks to create the STEP 7 user program:

- **Function (FC):** WinLC RTX supports up to 65,536 FCs (FC 0 to FC 65535). Each FC can contain up to 65,570 bytes.
- **Function block (FB):** WinLC RTX supports up to 65,536 FBs (FB 0 to FB 65535). Each FB can contain up to 65,570 bytes.
- **Data block (DB):** WinLC RTX supports up to 65,535 DBs (DB 1 to DB 65535). (DB 0 is reserved.) Each DB can contain up to 65,534 bytes.

The number and size of FCs, FBs, and DBs are also limited by the amount of available system memory. For more information about the instruction list supported by WinLC RTX, see the following topics:

- Technical data
- Organization blocks (OBs)
- System functions (SFCs)
- System function blocks (SFBs)



## S7 Communication Functions

Like other S7 controllers, WinLC RTX provides S7 communication between controllers on the network. The controllers can be either hardware or software logic controllers.

SFB or SFC	Name	Description
SFB 8 SFB 9	USEND URCV	Exchange data using a send and a receive SFB
SFB 12 SFB 13	BSEND BRCV	Exchange blocks of data of variable length between a send SFB and a receive SFB
SFB 14 SFB 15	GET PUT	Read data from a remote device Write data to a remote device
SFB 22 SFB 23	STATUS USTATUS	Specific query of the status of a remote device Receive status messages from a remote devices
SFC 62	CONTROL	Query the status of a connection
SFC 87	C_DIAG	Determines the current status of all S7 connections

Refer to your STEP 7 documentation for more information about S7 communications.

## PROFIBUS DPV1

DPV1 extensions to PROFIBUS-DP allow the enhanced communication required by complex slave devices. This enhanced communication includes acyclic data exchange, alarm and status messaging, and the transmission of complex data types. WinLC RTX provides support for the following DPV1 functionality:

- DP-Norm, DP-S7, DPV1, and DPV1 S7-compliant slaves
- Alarm and status OBs for processing DPV1-defined events, including:
  - OB 40 (process alarm)
  - OB 55 (status alarm)
  - OB 56 (update alarm)
  - OB 57 (manufacturer-specified alarm)
  - OB 82 (diagnostic alarm)
  - OB 83 (module pull/plug alarm)
- Data set read and write function blocks:
  - SFB 52 (RDREC), Read Data Set
  - SFB 53 (WRREC), Write Data Set
  - Execution of SFB 54 (RALRM), read alarm data, in the context of the triggering alarm
- Station and interface address
- Buffering of alarms received in DP mode CLEAR

For WinLC RTX to support DPV1, configure the submodule DP interface to be a DP Master. To select DP Master, follow these steps from the SIMATIC Manager:

1. Open the Hardware Configuration for your PC Station.
2. Double-click your submodule DP interface in the corresponding submodule slot of your WinLC RTX.
3. Select the Operating Mode tab of the CP card Properties dialog.
4. Select DP Master and set the DP mode to DPV1.

Refer to your STEP 7 documentation for specific information about DPV1 functionality.

## Communication Blocks

Like other S7 controllers, WinLC RTX provides communication blocks for exchanging data with other TCP/IP communications partners:

FB	Name	Description
FB 63	TSEND	Send data over a communications connection
FB 64	TRCV	Receive data over a communications connection
FB 65	TCON	Establishes a communications connection from the controller to a communications partner
FB 66	TDISCON	Terminates a communications connection from the controller to a communications partner

Refer to your STEP 7 documentation for more information about the TCP/IP communication FBs.

## Organization Blocks (OBs)

Organization blocks (OBs) are the interface between the operating system of the controller and the STEP 7 user program. You use OBs to execute specific components of your STEP 7 user program for the following events:

- When the controller starts and restarts
- Cyclically or at a specific time interval
- At certain times or on certain days
- After running for a specified period of time
- When errors occur
- When a hardware interrupt occurs

The program logic in an OB can contain up to 65,570 bytes.

OBs are processed according to the priority assigned to them.

The following table lists the OBs that WinLC RTX supports:

OB	Description	Priority Class
OB 1	Free scan cycle	1 (lowest)
OB 10	Time-of-day interrupt	0, 2 to 24
OB 20	Time-delay interrupt	0, 2 to 24
OB 30 to OB 38	Cyclic interrupt	0, 2 to 24
OB 40	Hardware (process alarm) interrupts	0, 2 to 24
OB 52 to OB 54	ODK interrupt	15
OB 55	Status alarm interrupt	0, 2 to 24
OB 56	Update alarm interrupt	0, 2 to 24
OB 57	Manufacturer-specific alarm interrupt	0, 2 to 24
OB 61 and OB 62	Synchronous cycle interrupts	0, 2 to 26 Default: 25
OB 80	Time error	26
OB 82	Diagnostic alarm interrupt	24 to 26 (or 28)*
OB 83	Module pull/plug alarm interrupt	24 to 26 (or 28)*
OB 84	CPU Hardware fault	24 to 26 (or 28)*

OB	Description	Priority Class
OB 85	Priority class error	24 to 26 (or 28)*
OB 86	Rack (DP slave) failure	24 to 26 (or 28)*
OB 88	Processing Interrupt (stop avoidance)	28
OB 100	Warm restart	27
OB 102	Cold restart	27
OB 121	Programming error	Priority class of the OB where the error occurred
OB 122	I/O access error	

\* Priority class 28 during STARTUP, user-configurable priority class (from 24 to 26) in RUN mode.

### OBs for the Free Scan Cycle, Cold Restart, and Warm Restart

The following table shows OBs for the free scan cycle and cold and warm restarts. WinLC RTX provides OB 1 (free scan cycle) for continuously executing the STEP 7 user program. On the transition from STOP mode to RUN mode, WinLC RTX executes OB 100 (warm restart) or OB 102 (cold restart), based either on the hardware configuration for WinLC RTX or which restart option was selected from a dialog displayed by the WinLC RTX panel. After OB 100 (or OB 102) has been successfully executed, WinLC RTX executes OB 1.

Organization Block (OB)		Start Event (in Hex)	Priority Class
Main program cycle	OB 1	1101, 1103, 1104	1
Warm restart	OB 100	1381, 1382	27
Cold restart	OB 102	1385, 1386	27

## Interrupt OBs

WinLC RTX provides a variety of OBs that interrupt the execution of OB 1. The following table lists the different interrupt OBs that are supported by WinLC RTX. These interrupts occur according to the type and configuration of the OB.

The priority class determines whether the controller suspends the execution of the STEP 7 user program (or other OB) and executes the interrupting OB. You can change the priority class for the interrupt OBs.

Interrupts		Start Event (in Hex)	Default Priority Class
Time-of-Day Interrupt	OB 10	1111	2
Time-Delay Interrupt Range: 1 ms to 60000 ms	OB 20	1121	3
Cyclic Interrupt Range: 1 ms to 60000 ms Recommended: > 10 ms	OB 30	1131	7
	OB 31	1132	8
	OB 32	1133	9
	OB 33	1134	10
	OB 34	1135	11
	OB 35	1136	12
	OB 36	1137	13
	OB 37	1138	14
	OB 38	1139	15
Hardware interrupt	OB 40	1141	16
Status Alarm interrupt	OB 55	1155	2
Update Alarm interrupt	OB 56	1156	2
Manufacturer-Specific Alarm interrupt	OB 57	1157	2

If WinLC RTX has been configured to execute a particular interrupt OB, but that OB has not been downloaded, WinLC RTX reacts in the following manner:

- If OB 10, OB 20, OB 40, OB 55, OB 56, or OB 57 is missing and OB 85 has not been downloaded, WinLC RTX changes operating mode (from RUN to STOP).
- WinLC RTX remains in RUN mode if a cyclic interrupt OB (OB 32 to OB 36) is missing. If these OBs cannot be executed at the specified time and OB 80 has not been downloaded, WinLC RTX changes from RUN mode to STOP mode.

## Considerations for Cyclic Interrupt OBs

Based on the time interval that you configure in the operational parameters for the cyclic interrupt, WinLC RTX starts the execution of the cyclic interrupt OB at the appropriate time. The optimum time interval for your application depends on the processing speed of your computer and the execution time of the cyclic OB. Jitter can cause an occasional overrun in the start event for a cyclic OB, which might cause WinLC RTX to go to STOP mode. Other factors that affect the execution of the OB include the following situations:

- The program in the OB takes longer to execute than the interval allows. If the execution of the program consistently overruns the start event of the cyclic OB, WinLC RTX can go to STOP mode (unless OB 80 is loaded).
- Programs in other priority classes frequently interrupt or take longer to execute, which prevents the controller from executing the cyclic OB at the scheduled time. If this occasionally causes an overrun, WinLC RTX starts the cyclic OB as soon as the first OB finishes.
- STEP 7 performs some task or function that causes the controller not to execute the cyclic OB at the scheduled time.

The sleep time of the WinLC RTX scan cycle does not affect the execution of a cyclic interrupt OB: WinLC RTX attempts to execute the OB at the appropriate interval regardless of the amount of sleep time that you configure for the scan. WinLC RTX provides several types of free cycle sleep management for managing sleep time. If a cyclic interrupt OB runs too frequently or requires too much of the time allotted for the total scan, it could cause the watchdog timer to time out (calling OB 80 or going to STOP mode).

If you schedule a cyclic interrupt OB (OB 30 to OB 38) to be executed at a specific interval, make certain that the program can be executed within the time frame and also that your STEP 7 user program can process the OB within the allotted time.

## Error OBs

WinLC RTX provides a variety of error OBs. Some of these error OBs have the configured (the user-assigned) priority class, while others (OB 121 and OB 122) inherit the priority class of the block where the error occurred.

The local variables for OB 121 and OB 122 contain the following information that can be used by the STEP 7 user program to respond to the error:

- The type of block (byte 4) and the number (bytes 8 and 9) where the error occurred
- The address within the block (bytes 10 and 11) where the error occurred

If the start event occurs for a particular error OB that has not been downloaded, WinLC RTX changes operating mode from RUN to STOP.

Error or Fault		Start Event (in Hex)	Default Priority Class
Time error	OB 80	3501, 3502, 3505, 3507	26
Diagnostic interrupt	OB 82	3842, 3942	26
Insert/remove module interrupt	OB 83	3861, 3863, 3864, 3865, 3961	26
CPU hardware fault (Windows "blue screen")	OB 84	3585	26 (or 28)
Priority class error: <ul style="list-style-type: none"> <li>Start event occurs for an OB that has not been downloaded.</li> <li>During the I/O cycle, WinLC attempts to access a module or DP slave that is defective or not plugged in.</li> </ul>	OB 85	35A1, 35A3, 39B1, 39B2	26
Rack failure (distributed I/O): a node in the PROFIBUS-DP subnet has failed or has been restored.	OB 86	38C4, 38C5, 38C7, 38C8, 39C4, 39C5	26 (or 28)
Processing interrupt: execution of a program block has been aborted	OB 88	3571, 3572, 3573, 3575, 3576, 3578, 357A	28
Programming error (For example: the user program attempts to address a timer that does not exist.)	OB 121	2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 253A; 253C, 253E	Same priority class as the OB in which the error occurred
I/O access error (For example: the user program attempts to access a module that is defective or is not plugged in.)	OB 122	2942, 2943	

For more information on the OBs, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".



## System Functions (SFCs)

WinLC RTX provides SFCs, which are system functions that perform various tasks. The STEP 7 user program calls the SFC and passes the required parameters; the SFC performs its task and returns the result. The following table lists the SFCs that WinLC RTX supports:

SFC	Name	Description
SFC 0	SET_CLK	Sets the system clock
SFC 1	READ_CLK	Reads the system clock
SFC 2	SET_RTM	Sets the run-time meter
SFC 3	CTRL_RTM	Starts or stops the run-time meter
SFC 4	READ_RTM	Reads the run-time meter
SFC 5	GADR_LGC	Queries the logical address of a channel
SFC 6	RD_SINFO	Reads the start information of an OB
SFC 9	EN_MSG	Enable Block-Related, Symbol-Related and Group Status Messages
SFC 10	DIS_MSG	Disable Block-Related, Symbol-Related and Group Status Messages
SFC 11	DPSYNC_FR	Synchronizes groups of DP slaves
SFC 12	D_ACT_DP	Deactivates and activates of DP slaves
SFC 13	DPNRM_DG	Reads the diagnostic data of a DP slave  DP configuration tested: one ET 200M slave with one 8-input/8-output module and one 16-output module
SFC 14	DPRD_DAT	Reads the consistent data from a DP slave
SFC 15	DPWR_DAT	Writes the consistent data to a DP slave
SFC 17	ALARM_SQ	Generates an acknowledgeable block-related message
SFC 18	ALARM_S	Generates a permanently acknowledgeable block-related message
SFC 19	ALARM_SC	Queries the acknowledgement status for the last message (SFC 17 or SFC 18).
SFC 20	BLKMOV	Copies variables

SFC	Name	Description
SFC 21	FILL	Initializes a memory area 1 word 50 words 100 words
SFC 22	CREAT_DB	Creates a retentive data block in Work memory The current values of the DB are retained after a warm restart
SFC 23	DEL_DB	Deletes a data block. WinLC RTX allows an application to delete a non-sequence-relevant data block.
SFC 24	TEST_DB	Provides information about a data block For WinLC RTX, SFC 24 can return the DB length and write-protection flags for non-sequence-relevant data blocks, although it returns error code 80B2 for non-sequence-relevant data blocks.
SFC 26	UPDAT_PI	Updates the process-image input table
SFC 27	UPDAT_PO	Updates the process-image output table
SFC 28	SET_TINT	Sets the time-of-day interrupt (OB 10)
SFC 29	CAN_TINT	Cancel the time-of-day interrupt (OB 10)
SFC 30	ACT_TINT	Activates the time-of-day interrupt (OB 10)
SFC 31	QRY_TINT	Queries the time-of-day interrupt (OB 10)
SFC 32	SRT_DINT	Starts the time-delay interrupt (OB 20)
SFC 33	CAN_DINT	Cancel the time-delay interrupt (OB 20)
SFC 34	QRY_DINT	Queries the time-delay interrupt (OB 20)
SFC 36	MSK_FLT	Masks synchronous errors
SFC 37	DMSK_FLT	Unmasks synchronous errors
SFC 38	READ_ERR	Reads the error register
SFC 39	DIS_IRT	Disables the processing of all new interrupts
SFC 40	EN_IRT	Enables the processing of new interrupts

SFC	Name	Description
SFC 41	DIS_AIRT	Delays higher priority interrupts and asynchronous errors
SFC 42	EN_AIRT	Enables the processing of new interrupts with higher priority than the current OB
SFC 43	RE_TRIGR	Retrigger cycle time monitoring
SFC 44	REPL_VAL	Transfers a substitute value to ACCU1 (accumulator 1)
SFC 46	STP	Changes the operating mode to STOP mode
SFC 47	WAIT	Delays the execution of the STEP 7 user program by the specified number of microseconds, rounded up to the nearest millisecond.
SFC 49	LGC_GADR	Queries the module slot belonging to a logical address
SFC 50	RD_LGADR	Queries all of the logical addresses of a module
SFC 51	RDSYSST	Reads all or part of a system status list
SFC 52	WR_USMSG	Writes a user-defined diagnostic event to the diagnostics buffer
SFC 54	RD_DPARM	Reads the defined parameter
SFC 55	WR_PARM	Writes the dynamic parameters
SFC 56	WR_DPARM	Writes the default parameters
SFC 57	PARAM_MOD	Assigns the parameters to a module
SFC 58	WR_REC	Writes a data record
SFC 59	RD_REC	Reads a data record
SFC 62	CONTROL	Checks the status of the connection belonging to an SFB instance
SFC 64	TIME_TCK	Reads the system time
SFC 78	OB_RT	Reports OB run-time information, with resolution to the nearest microsecond
SFC 79	SET	Sets a range of outputs
SFC 80	RESET	Resets a range of outputs

SFC	Name	Description
SFC 82	CREA_DBL	Creates a data block in Load memory
SFC 83	READ_DBL	Copies data from a block in Load memory
SFC 84	WRIT_DBL	Writes to a Load Memory block so that the data is saved immediately  Load memory blocks that are used to recover from an abnormal termination can be updated while the program is running. Use SFC 84 only for larger segments of a database, not for frequent variable processing.
SFC 85	CREA_DB	Creates a DB that can be either retentive or non-retentive, depending on the input parameter <ul style="list-style-type: none"> <li>If retentive, the current values of the DB are retained after a warm restart (OB 100).</li> <li>If non-retentive, the current values of the DB are not retained after a warm restart (OB 100).</li> </ul>
SFC 87	C_DIAG	Determines the current status of all S7 connections
SFC 126	SYNC_PI	Update process image partition input table in synchronous cycle
SFC 127	SYNC_PO	Update process image partition output table in synchronous cycle

For more information on the SFCs, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

**Note:** Some SFCs require special consideration regarding the possibility of a Windows Blue Screen. See the topic "Considerations for SFC 22, SFC 23, and SFC 82 - 85" for information.

## Running Asynchronous SFCs Concurrently

WinLC RTX restricts the number of asynchronous OBs that can be running concurrently according to the following rules:

- WinLC RTX allows a maximum of 5 instances of the asynchronous system function SFC 51 (index B1, B3) to be running.
- WinLC RTX allows a maximum of 20 asynchronous SFCs from the following set to be running: SFC 11, SFC 13, SFC 55, SFC 56, SFC 57, SFC 58, and SFC 59.
- WinLC RTX allows a maximum of 32 asynchronous SFCs in any combination from the following set to be running: SFC 82, SFC 83, and SFC 84.

## **SFCs That Can Cause the Scan Cycle to Vary**

The following SFCs can cause the scan cycle to vary ("jitter"):

- SFC 22 (CREAT\_DB)
- SFC 23 (DEL\_DB)
- SFC 52 (WR\_USMG)
- SFC 85 (CREA\_DB)

## **Notes for SFC 82, SFC 83, and SFC 84**

In contrast to the S7-300, WinLC RTX supports a synchronous interface for SFC 82, SFC 83, and SFC 84 in STARTUP. WinLC allows both the first call (with REQ = 1) and the second call (with REQ = 0) in STARTUP so the action can be completed in STARTUP.

The normal STEP 7 error codes apply for SFC 82, SFC 83, and SFC 84, plus an additional return error code of 80C3. These SFCs return the 80C3 return error codes if WinLC RTX exceeds a limit of 32 outstanding SFC 82, SFC 83, and SFC 84 jobs.

## System Function Blocks (SFBs)

System Function Blocks are logic blocks (similar to SFCs) that perform basic tasks when called by the STEP 7 user program. You must provide a data block (DB) when you call an SFB.

The following table lists the SFBs that WinLC RTX supports.

SFB	Name	Description
SFB 0	CTU	Provides a count-up counter
SFB 1	CTD	Provides a count-down counter
SFB 2	CTUD	Provides a count-up/down counter
SFB 3	TP	Generates a pulse
SFB 4	TON	Generates an on-delay timer
SFB 5	TOF	Generates an off-delay timer
SFB 8	USEND	Sends a data packet of CPU-specific length (two-way), uncoordinated with receiving partner
SFB 9	URCV	Asynchronously receives a data packet of CPU-specific length (two-way)
SFB 12	BSEND	Sends a segmented data block up to 64 Kbytes (two-way)
SFB 13	BRCV	Receives a segmented data block up to 64 Kbytes (two-way)
SFB 14	GET	Reads data up to a CPU-specific maximum length (one-way) from a remote CPU
SFB 15	PUT	Writes data up to a CPU-specific maximum length (one-way) to a remote CPU
SFB 22	STATUS	Query the status of a remote device
SFB 23	USTATUS	Receive the status of a remote device
SFB 31	NOTIFY8P	Generates block-related messages without acknowledgement indication for 8 signals
SFB 32	DRUM	Implements a sequencer
SFB 33	ALARM	Generates block-related messages with acknowledgment display
SFB 34	ALARM_8	Generates block-related messages without values for 8 signals

SFB	Name	Description
SFB 35	ALARM_8P	Generates block-related messages with values for 8 signals
SFB 36	NOTIFY	Generate block-related messages without acknowledgment display
SFB 52	RDREC	Read data set
SFB 53	WRREC	Write data set
SFB 54	RALRM	Receives alarm data for a DP slave
SFB 65001	CREA_COM	(WinAC ODK CCX)
SFB 65002	EXEC_COM	(WinAC ODK CCX)

For more information on the SFBs, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

## System Clock and Run-Time Meter

Like a hardware S7 controller, WinLC RTX has a system clock based on the hardware clock of your computer.

**Note:** With the WinAC TimeSync feature, the WinLC RTX clock can be either a time master or a time slave.

To adjust and read the system clock, use the following SFCs. For more information on these functions, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

SFC	Name	Description
SFC 0	SET_CLK	Allows you to set the time and the date of the system clock. The clock then runs starting from the set time and set date. Format: DT#1995-01-15-10:30:30
SFC 1	READ_CLK	Allows you to read the current date or current time of the system clock of the controller.
SFC 64	TIME_TCK	Allows you to read the system time of the controller. The system time is a "time counter" counting cyclically from 0 to a maximum of 2147483647 ms. In case of an overflow, the system time is counted again, starting at 0. The resolution is 1 ms. The system time is only influenced by the operating modes of the controller.





# Tuning the Performance of the Controller

## Scan Cycle for a PC-Based Controller

During one scan cycle, the controller updates the outputs, reads the inputs, executes the STEP 7 user program, performs communication tasks, and provides time for other applications to run. The following parameters affect the scan cycle:

- Execution Time (in milliseconds) is the actual amount of time used by the controller to update the I/O and to execute the STEP 7 user program.
- Scan Cycle Time (in milliseconds) is the number of milliseconds from the start of one cycle to the start of the next cycle. This value must be greater than the execution time of the scan to provide execution time for any application that has a lower priority than WinLC RTX.
- Sleep Time (in milliseconds) determines how much time is available during the free cycle (execution cycle for OB 1) to allow higher priority OBs and other applications to use the resources of the computer.

The Priority for the controller application also affects the scan cycle by determining when the controller runs or is interrupted by other Windows applications. You must ensure that the sleep time occurs every 50 milliseconds or less in order for other Windows applications, such as moving the mouse, to operate smoothly.

The tuning panel allows you to tune and test the performance of the controller by adjusting the parameters that affect the scan cycle (minimum cycle time, minimum sleep time, and priority) without affecting the system configuration for the controller. After testing tuning parameters, you use STEP 7 to configure the minimum scan cycle time for the controller when you create the system (hardware) configuration.

## Tasks Performed during the Scan Cycle

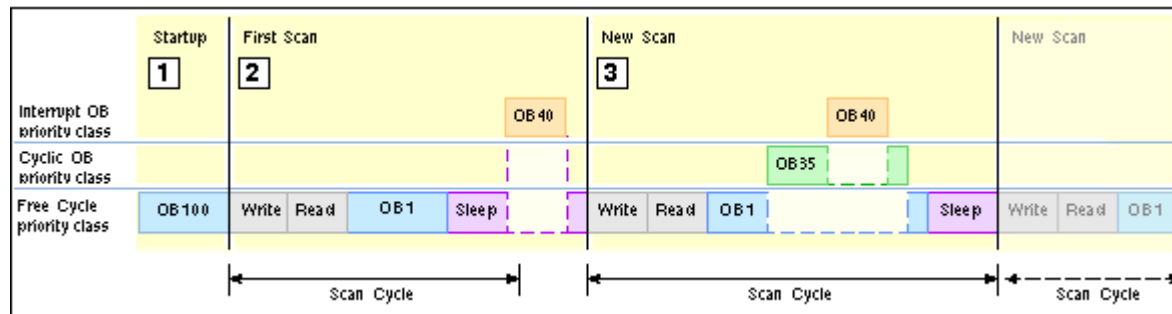
After you have used STEP 7 to create and download your control program to the controller, the controller starts executing the control program when you set the controller to RUN mode. Like any other S7 PLC, the controller executes your STEP 7 user program in a continuously repeated scan cycle.

In one scan, the controller performs the following tasks:

1.	The controller writes the status of the OB 1-assigned process-image output table (the Q memory area) to the I/O module outputs.
2.	The controller reads the states of the I/O module inputs into the OB 1-assigned process-image input table (the I memory area).
3.	The controller executes the STEP 7 user program in OB 1.
4.	OB 1 waits until the minimum sleep time requirement is met before starting another scan. Other OBs can execute at this time.

Because the PC-based controller shares the resources of your computer with other programs (including the operating system), you must ensure that the controller provides sufficient time for other Windows applications to be processed. If the actual execution time for the scan cycle is less than the minimum scan cycle time that you configured with STEP 7, the controller suspends the free cycle (OB 1) until the minimum scan cycle time is reached before starting the next scan. This waiting period, or sleep time, allows other applications to use the resources of the computer.

The following figure provides an overview of the tasks that are performed by the controller during different scan cycles.



**1** Startup  
On a transition from STOP mode to RUN mode, the controller loads the system configuration, sets the I/O to the default states, and executes the startup OB (OB 100 or OB 102).

The startup cycle is not affected by the minimum cycle time and minimum sleep time or watchdog parameters; however, it is affected by the execution time limit.

**2** First Scan  
An OB with a higher priority class can interrupt the free cycle at any time, even during the sleep time.  
In the example above, the controller handles a hardware (I/O) interrupt that occurs during the sleep time by executing OB 40. After OB 40 has finished, the controller waits for the minimum cycle time to start the next scan.

**Note:** It is possible for the controller to use all of the sleep time for processing higher-priority OBs. In this case, other Windows applications may not have sufficient time to run. Refer to the techniques for managing sleep time listed below.

**3** New Scan  
In the example above, the controller suspends the execution of OB 1 to execute a cyclic OB (OB 35), which has a higher S7 priority than OB 1. The controller also suspends the execution of OB 35 to handle another I/O interrupt (OB 40).  
After OB 40 finishes, the controller resumes the execution of OB 35, and after OB 35 finishes, the controller resumes the execution of OB 1.

The length of the scan cycle is determined by the execution time of all OBs executed during the scan, the minimum cycle time, and the minimum sleep time. If the execution time is less than the minimum cycle time that was configured in the system configuration, the controller suspends the free cycle until the minimum sleep time is met. During the sleep time, the computer runs any interrupt OBs and other Windows applications.



#### **Warning**

Variation in the execution time or response time of the STEP 7 user program could potentially create a situation where the equipment or application being controlled can operate erratically and possibly cause damage to equipment or injury to personnel.

If the controller does not provide sufficient sleep time for other applications to run, the computer can become unresponsive to operator input, or the controller and other applications can operate erratically. In addition, the execution of the STEP 7 user program can experience non-deterministic behavior (jitter) such that execution times can vary and start events can be delayed.

Always provide an external emergency stop circuit. In addition, always tune the sleep time and manage the performance of the controller so that your STEP 7 user program executes consistently.

## **Methods for Managing the Performance of WinLC RTX**

While executing the STEP 7 user program, WinLC RTX can experience a variation in the process execution time or response time that causes the scan times to vary or to exhibit non-deterministic behavior ("jitter"). You can use the following methods to manage the performance of WinLC RTX:

- Adjusting the priority for the controller: Affects the execution of WinLC RTX in relation to other RTX processes executing on your computer
- Adjusting the minimum sleep time and minimum cycle time parameters: Affects the execution of the free cycle or OB 1 (OB priority class 1)
- Inserting sleep time into the STEP 7 user program (SFC 47 "WAIT"): Affects the execution for the priority class of the OB that calls SFC 47 (and any lower priority class)
- Adjusting the sleep-monitoring algorithm of the execution monitor: Affects the execution of all OB priority classes (if the other mechanisms do not meet the requirements for sleep time)

WinLC RTX provides a tuning panel for monitoring the performance and for modifying the parameters that affect the scan cycle.

## What Causes Jitter?

Because the PC-based controller must share the computer with other running processes, the execution of the control program can experience "jitter" when a higher-priority or active process uses the CPU or system resources of the computer. Jitter is a variation in the process execution time or response time that causes the scan times to vary or to exhibit non-deterministic behavior.

Jitter occurs when there is a delay in either the start or the finish of an OB. For example: the execution time can deviate by a few milliseconds between scans, or the start of an interrupt OB can be delayed. For some control applications, such time lapses do not disturb the proper operation of the controller, but in a highly time-sensitive process, a jitter of even 1 ms can be significant.

The following settings for WinLC RTX can cause jitter in the execution of the control programs:

- Priority settings for competing RTX applications
- Priorities among the WinLC RTX threads
- Execution Monitor sleep interval

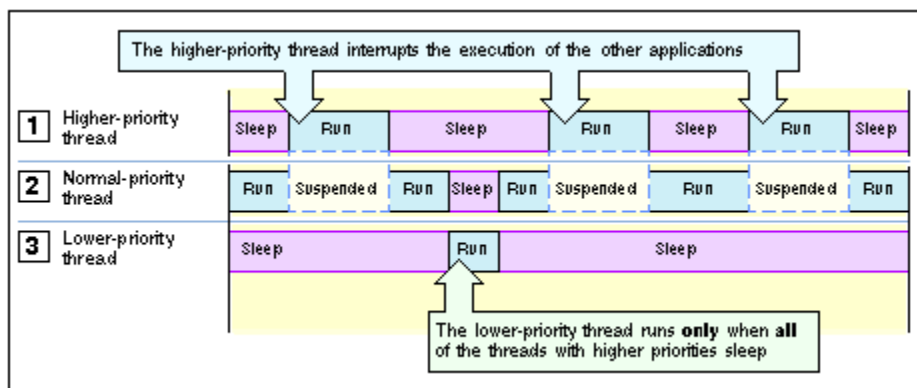
The tuning panel of WinLC RTX provides several tools for reducing jitter in the control program.

Jitter can also be caused by other sources than WinLC RTX:

- Jitter can be caused by the design of your control program. For example, different branches in the logic of the control program might cause the execution time to vary.
- Jitter can be caused by the computer hardware. For example, jitter can be caused by an operation with a long DMA cycle, such as a video card using the PCI bus. Jitter can also be caused by a driver, such as for a CD drive or a diskette drive. Hardware-induced jitter cannot be managed by software. For a uniprocessor system running WinLC RTX, Ardence provides an application to help evaluate the suitability of the computer hardware for use with the RTX extensions.
- Jitter can be caused by an application that was created with the WinAC RTX Open Development Kit (ODK), such as when a synchronous process takes too long to execute. Refer to the documentation for WinAC RTX ODK for more information.

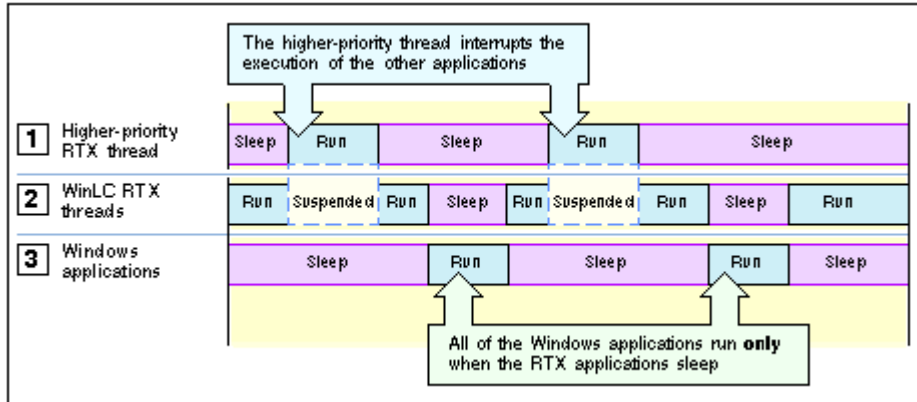
## Priority Settings for Competing RTX Applications Can Cause Jitter

Every RTX application that is running on your computer has one or more threads (or tasks), and each thread has a priority. The RTX subsystem executes the RTX application threads with the highest priority first and executes a lower-priority thread only when all of the higher priority threads are finished or suspended (for example, to wait for some other activity to complete or to "sleep" for a specified time). Threads with higher priorities interrupt and suspend the operations of threads with lower priorities. After the higher-priority thread finishes, the lower-priority thread resumes its operation.

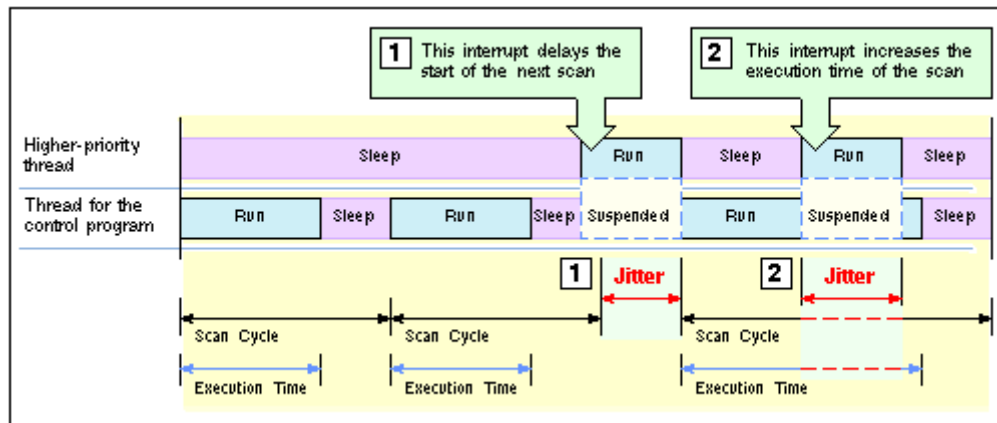


WinLC RTX operates in a real-time subsystem (RTSS) that provides a range of priorities above the typical Windows priorities. All threads of WinLC RTX execute at higher priorities than threads for Windows applications. Windows applications can not cause jitter in WinLC RTX, but another RTX thread that has a higher RTSS priority than WinLC RTX can induce jitter.

You must also ensure that WinLC RTX and any other RTX application provide sufficient sleep time to allow the Windows applications to run.



Jitter can occur when a process with a higher RTSS priority interrupts and suspends the execution of the controller. As shown in the following figure, jitter typically appears in two forms.



- 1** The higher priority threads can cause jitter by delaying the start of an OB. This could delay the start of the free cycle (OB 1) or of an interrupt OB (such as OB 35 or OB 40).
- 2** The higher priority application can cause jitter by extending the execution time for an individual scan.

You can use the tuning panel to increase or decrease the priority for the WinLC RTX threads. The higher you set the priority for the WinLC RTX threads in relation to the threads of the other RTX applications, the less jitter you typically encounter. However, you must also ensure that WinLC RTX provides enough sleep time for other RTX and Windows applications to run.

The tuning panel also provides information that allows you to monitor the amount of jitter in the scan cycle.

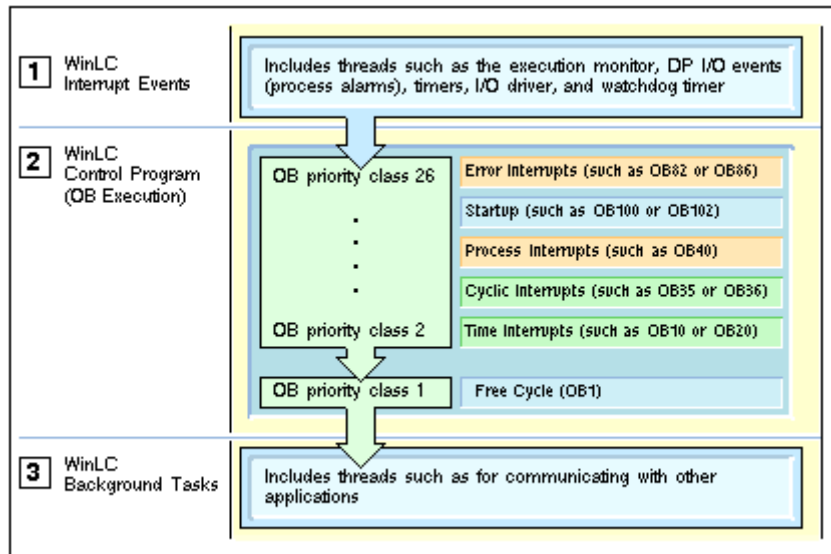
For more information about priorities, refer to the following topics:

- Adjusting the Priority
- Real-Time Subsystem Priorities

## Priorities among the WinLC RTX Threads Can Cause Jitter

In addition to the thread that executes the OBs of the control program, WinLC RTX uses other threads, including some with higher priority than the OB Execution thread. Some examples of higher-priority threads are the execution monitor, the start event for an OB, the watchdog events, the timers, communication interfaces, and the events for the DP I/O. Any of these higher-priority threads can induce jitter in the execution of the control program.

The relative priorities (priority classes) of the OBs in the control program itself can also cause jitter. For example, an error OB delays or interrupts the execution of all lower-priority OBs.

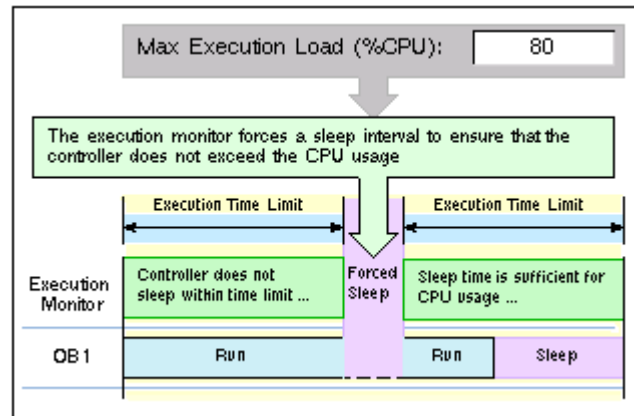


- 1** The threads of the interrupt events have a higher priority than the thread for the execution of the control program. These threads can cause jitter by interrupting the execution of the control program.
- 2** The OB Execution thread includes the different priority classes for the OBs of the control program. The interrupt OBs can cause jitter not only by interrupting the free cycle (OB 1), but also by interrupting another interrupt OB with a lower priority class.
- 3** The background tasks for WinLC RTX includes the threads used for communicating with other applications, such as STEP 7. The OB Execution thread and the higher-priority threads affect the execution of these tasks.

## The Sleep Interval Forced by the Execution Monitor Can Cause Jitter

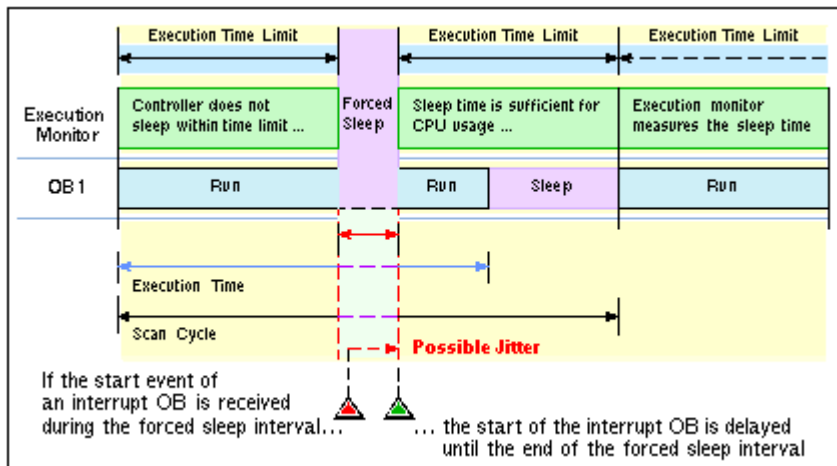
WinLC RTX must sleep (release the CPU) periodically in order for the other applications to run. The free cycle includes a sleep interval that follows the execution of OB 1. However, this sleep interval can be interrupted by higher-priority OBs. Also, a scan cycle with a relatively long execution time could cause other applications to wait too long to access the CPU.

To ensure that the controller does not exceed a specified percentage of CPU usage, an execution monitor measures the sleep time within a fixed execution time limit. If the controller does not sleep for the specified amount of time within the execution time limit, the execution monitor forces a sleep interval.



Because the execution monitor runs in a higher priority class than any OB, the controller cannot interrupt the forced sleep interval. This could delay the start of an interrupt OB, such as OB 35, until the end of the forced sleep interval. This delay in handling an interrupt OB results in jitter.

As a general rule for decreasing jitter, always design your control program to keep the execution time of the higher-priority OBs as short as possible.



WinLC RTX provides several options for managing the sleep time to avoid the uninterruptible forced sleep interval:

- You can increase the minimum sleep time parameter for managing the sleep time for the free cycle (priority class 1, or OB 1).
- You can call SFC 47 (“WAIT”) to insert an extra, interruptible sleep interval into the control program for managing the sleep time for an application-defined priority class (priority classes 2 to 24).
- You can adjust the sleep-monitoring algorithm for the execution monitor for managing sleep time at a higher priority class than any OB.

## Adjusting the Priority of the Controller

If other RTSS applications are installed in addition to WinLC RTX, you can adjust the priority of the controller to improve performance. If not, you do not need to adjust the controller priority.

The priority of the controller determines how WinLC RTX runs in relation to other RTSS applications that are running on the computer.

Adjusting the priority of the controller can reduce or increase the amount of jitter in the scan time. The tuning panel allows you to change the priority for the controller application. When you use the tuning panel to change the priority, the controller automatically ensures that its interrupt activities, such as those which schedule interrupt OBs, are also set to an appropriate priority.

WinLC RTX does not control priorities in customer software, such as asynchronous threads in custom software or other applications in the same environment.

**Note:** The CCX interface of the WinAC Open Development Kit (ODK) provides an `ODK_CreateThread` function. Calling the `ODK_CreateThread` function creates asynchronous threads with priorities that are adjusted when you change the priority of the controller.

If you do **not** use the `ODK_CreateThread` function to create threads (for example, if you use a Windows API call to create a thread), changing the priority of the controller does **not** adjust the priority of those threads.

Refer to the documentation of the WinAC Open Development Kit (ODK) for more information.

While a PC-based controller must maintain the essential features of a SIMATIC S7 PLC, the PC-based controller must also allow the other applications to run on the computer. The operating system of the computer uses a concept of execution threads (or tasks) to run or execute the applications. Each application has one or more threads, and each thread has a priority. The operating system executes the threads with the highest priority first and executes a lower-priority thread only when all of the higher priority threads are suspended (for example, to wait for some other activity to complete or to “sleep” for a specified time). Threads with higher priorities interrupt and suspend the operations of other threads that have lower priorities. After the higher-priority thread finishes, the lower-priority thread resumes its operation.

To change the priority, follow these steps:

1. Use the Priority slider to choose a priority based on the priority levels for your operating system. The new priority is displayed as you move the slider.
2. Click Set to set the priority to the new value.

## Real-Time Subsystem Priorities

WinLC RTX provides real-time priorities for the most demanding control projects that are absolutely time-critical. Because WinLC RTX competes only with other applications in the real-time subsystem, the controller provides the most deterministic behavior, with a possibility for reducing jitter in the scan cycle to less than 500 microseconds.

Because the controller runs with an RTSS priority above the Windows priorities, the sleep time for the STEP 7 user program determines the amount of time for other Windows activities and applications. Provide sleep time that allows other application to run. Use the tuning panel to monitor the variation in scan times that occurs as the controller executes your STEP 7 user program.

Although the RTSS environment allows priorities from 1 to 127, WinLC RTX only runs up to priority 62. Another RTSS application thread could have a higher or lower priority than WinLC RTX.

The controller application installs with a default RTX priority of 50, which typically delivers satisfactory performance. If the controller competes with other RTSS applications for the computer resources, set the priority for the controller application to run either above or below the priority of the other RTSS applications.



## Managing the Sleep Time

### Sleep Management Techniques

During a sleep interval, the controller allows other applications to use the resources of the computer. By managing the sleep time, you can tune the performance of the controller in order to allow all applications on the computer to run with acceptable performance. You can use a variety of techniques for managing the sleep intervals for the controller:

- Adjusting the minimum sleep time parameter. The minimum sleep time determines the amount of sleep time that is added during the execution of the free cycle (OB 1). This sleep time affects only OB priority class 1.
- Calling SFC 47 from your STEP 7 user program. SFC 47 inserts a sleep interval into the execution of your STEP 7 user program. This sleep time affects OB priority classes 2 to 24.
- Adjusting the execution monitor. The execution monitor uses a sleep-monitoring algorithm (based on the execution time limit and the maximum execution load parameters) to force a sleep interval. The execution monitor runs asynchronously to the scan cycle. This sleep time affects all OB priority classes.

#### Contents of this topic:

- Managing the Sleep Time of the Controller
- Tuning Strategy
- Sample Interaction of the Execution Monitor and the Minimum Sleep Time

### Managing the Sleep Time of the Controller

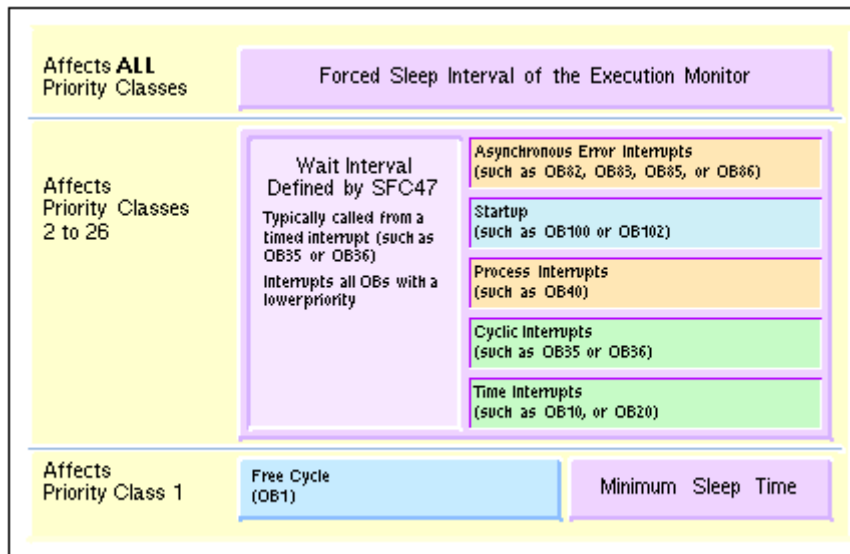
Because the controller shares the resources of your computer with other applications, you must ensure that the controller sleeps for a sufficient interval to allow the other applications to run.

#### Notice

The most effective method for granting time to other applications is to set the minimum sleep time parameter to the largest value that your control application allows. The other methods for managing the sleep time provide sufficient sleep time for the other applications to run, but may degrade the performance of the controller.

The controller provides the following techniques for managing the sleep time:

- The controller provides an execution monitor that enforces the maximum execution load on the resources of the computer. The execution monitor measures the amount of sleep time taken by the controller within an execution time limit, which is independent from the execution time of the scan cycle. If necessary, the execution monitor forces a sleep interval to achieve the specified execution load. This forced sleep interval suspends the execution of any OB and can also delay the start of an interrupt OB.
- The controller provides a minimum sleep time parameter that adds sleep time for the free cycle. This sleep interval occurs after the execution of OB 1. The minimum sleep time affects only priority class 1. An OB in a higher priority class can interrupt this sleep interval. The controller does not adjust the minimum sleep time to compensate for the execution time of interrupt OB. However, any forced sleep interval (generated by the execution monitor) is subtracted from the sleep interval generated by the minimum sleep time.
- The controller supports SFC 47 ("WAIT"), which inserts a specified sleep interval for the priority class of the OB that calls SFC 47. This sleep interval the OBs at the same or lower priority class as the OB that calls SFC 47, but an OB in a higher priority class can interrupt this sleep interval. You can use SFC 47 to create sleep time that can be interrupted so that the controller can avoid jitter when handling any interrupts that are critical for the application.



### Tuning Strategy

As you test the performance of the controller during the development phase of your project, consider the following strategy for adjusting the sleep time:

1. Set the minimum sleep time parameter to 0 and run the STEP 7 user program. This allows you to determine whether there is unacceptable jitter in the scan cycle.
2. To reduce any unacceptable jitter, first use the tuning panel to increase the minimum sleep time and observe the effect on cycle time and CPU usage.
3. If the amount of jitter is still unacceptable, review the sections of the STEP 7 user program that are being affected by the jitter. If possible, have your STEP 7 user program call SFC 47 to add sleep time.
4. To further reduce any jitter, increase the execution time limit to the maximum possible execution time for your control program.

If the sleep management techniques do not provide adequate improvement in reducing jitter, consider increasing the priority for the controller. (The priority of the controller is not the same as the priority class of an OB.)

### Sample Interaction of the Execution Monitor and the Minimum Sleep Time

To help explain the tools for managing the sleep time of the controller, the following example shows how the execution monitor and the minimum sleep time can interact:

- The first sample shows the sleep time that would be generated by the execution monitor alone, with no minimum sleep time added to the free cycle.
- The second sample shows how the execution of the free cycle is affected by adding a minimum sleep time to the scan cycle.

The following example describes the execution of a STEP 7 user program that uses OB 1 to start a 1-second timer, and then check the timer after an elapsed time of 1 second (1000 ms). The controller has been configured with the following parameters:

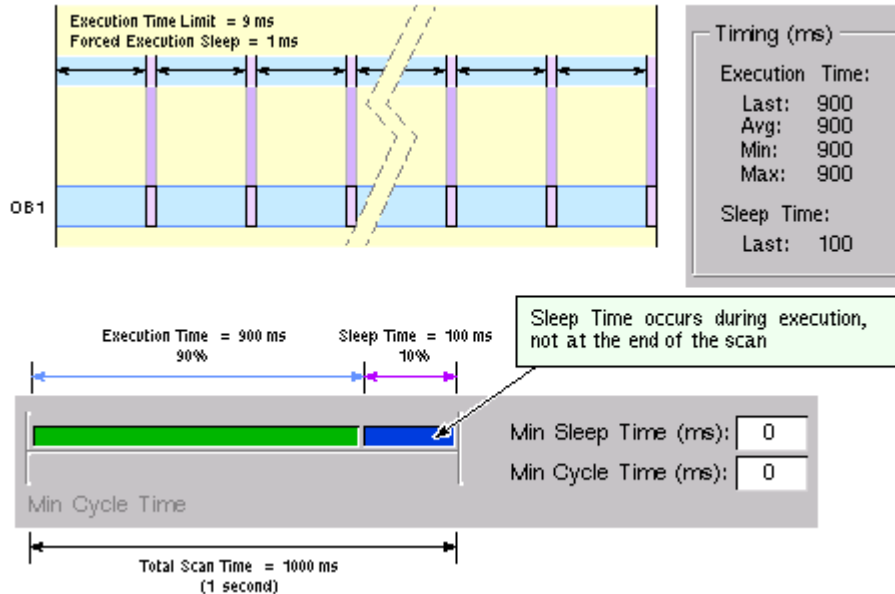
Parameter	Value
Execution Time	OB 1 takes 900 ms to execute.
Minimum Sleep Time	0 ms
Minimum Cycle Time	0 ms
Maximum Execution Load	90% (uses the default wake/sleep algorithm)
Execution Time Limit	9 ms (uses the default value)
Forced Execution Sleep	1 ms (uses the default value)

### Sleep Time Generated by the Execution Monitor (Minimum Scan Time = 0)

If you set the minimum sleep time parameter to 0, the controller uses the execution monitor alone to provide sleep time. The figure shows the operation of the execution monitor, using the default values.

The execution monitor suspends the execution of OB 1 for 1 ms after every 9 ms of execution by default in order to enforce a limit of 90% execution load (CPU usage). For every 1 second of elapsed clock time, the default execution time for OB 1 is 900 ms, with forced sleep intervals totaling 100 ms.

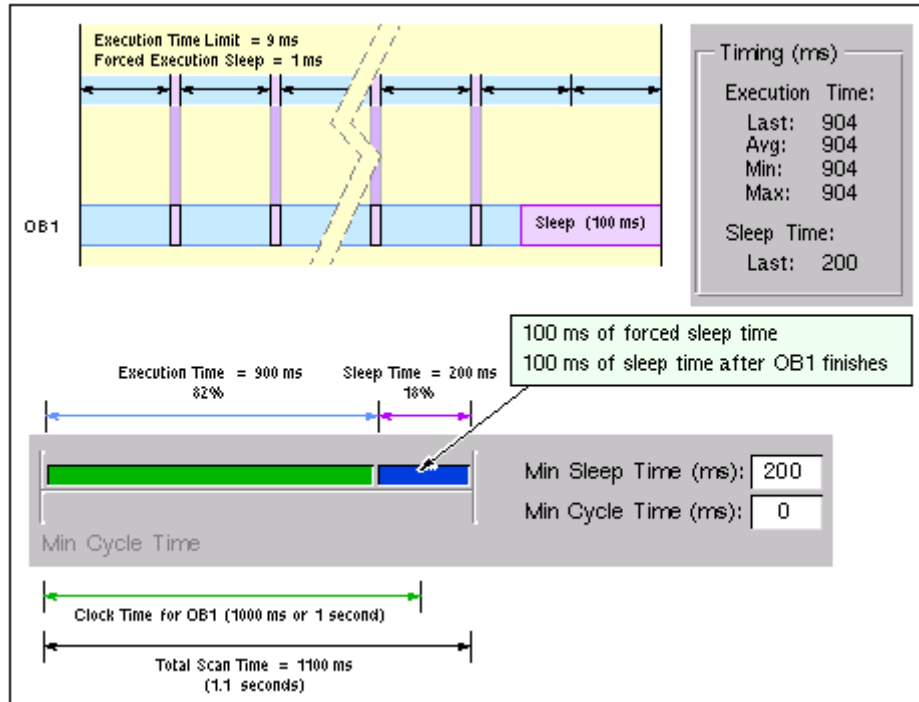
Notice that the sleep time occurs at intervals within the execution of OB 1.



### Adding a Minimum Sleep Time for the Free Cycle

This figure shows how changing the minimum sleep time from 0 to 200 affects the execution of OB 1. The execution monitor still forces 100 ms of sleep time to occur during the execution of OB 1. With the minimum scan time parameter set to 200 ms, the controller then sleeps for only another 100 ms, for a combined total of 200 ms, before starting the next free cycle.

The total scan time increases to approximately 1100 ms: the execution time (900 ms) for OB 1, the forced sleep time (100 ms), and the sleep time at the end of the scan cycle (100 ms).



## Adjusting the Minimum Sleep Time and Cycle Time

The tuning panel provides the following parameters that allow you to manage the sleep time of the free cycle (priority class 1, or OB 1):

- Minimum Cycle Time (in milliseconds) sets the minimum number of milliseconds from the start of one free cycle to the start of the next free cycle. This value must be greater than the execution time before it causes any sleep time to occur within the free cycle. You use STEP 7 to configure the minimum cycle time for the controller when you create the system (hardware) configuration. You can use the tuning panel to adjust the minimum cycle time, but any changes are discarded when you shut down the controller. However, you must use STEP 7 to make the changes permanent.
- Minimum Sleep Time (in milliseconds) determines how much sleep time is available during the free cycle (OB 1) for allowing higher priority OBs and other applications to use the resources of the computer. The controller automatically saves any changes to the minimum sleep time made with the tuning panel. You do not use STEP 7 to make any change to the minimum sleep time permanent.

The execution of the free cycle is affected by both the minimum sleep time and the minimum cycle time values.

- The minimum cycle time by itself results in a fixed scan cycle time with a variable sleep time (if the minimum cycle time is large enough to accommodate the execution time plus the sleep time).
- The minimum sleep time by itself results in a fixed sleep time with a variable scan time, depending on the length of the execution time.

The minimum sleep time value guarantees that a configured amount of sleep time occurs within each free cycle, even if the value for the minimum cycle time is too small. The controller releases control of the CPU for a sleep interval. This sleep interval is the larger of either the configured minimum sleep time value or a sleep time that is computed from the minimum cycle time parameter.



### Warning

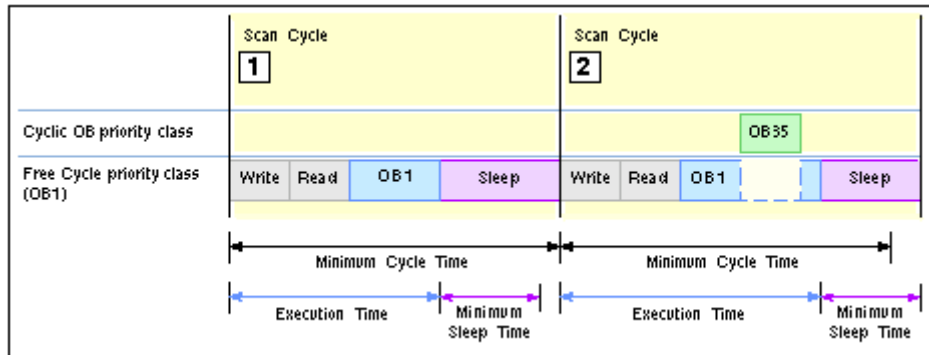
If you set the minimum scan time to a value larger than the watchdog time, WinLC goes to STOP mode during the first scan at the end of the watchdog time interval.

Causing the controller to go to STOP mode unexpectedly can cause damage to process equipment or injury to personnel.

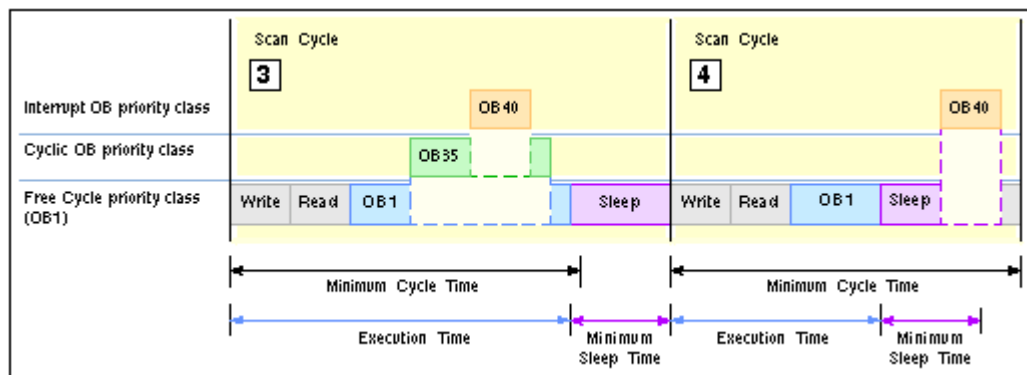
Do not set the minimum scan cycle time to be longer than the scan cycle monitoring time (the watchdog time) configured in the STEP 7 Hardware Configuration Editor.

## Parameters That Affect the Sleep Time for the Free Cycle

The following figures explain the interaction between the execution time, the minimum sleep time, and the minimum cycle time parameters.



- 1 For the first sample scan shown in the example above, the execution time plus minimum sleep time is less than the minimum cycle time. In this case, the controller increases the sleep time until the minimum cycle time is achieved.
- 2 For the second sample scan shown in the example above, the execution of OB 35 increases the execution time, and the execution time plus the minimum sleep time is greater than the minimum cycle time. In this case, the controller waits the minimum sleep time before starting the next scan.



- 3 For the third sample scan shown in the example above, the controller executes both a cyclic interrupt (OB 35) and an I/O interrupt (OB 40). The execution time exceeds the minimum cycle time, and the controller waits the minimum sleep time before executing the next scan.
- 4 For the fourth sample scan shown in the example above, the controller executes OB 40 during the sleep time after OB 1 has finished. In this case, the controller waits until the minimum cycle time before starting the next scan.

Because the execution of OB 40 does not reset the minimum sleep time counter, it is possible that the controller does not provide sufficient sleep time to allow other Windows applications to be processed. You must then use other methods for ensuring that the controller provides a sufficient amount of sleep time.

## Hints

You can use the following techniques to adjust controller performance using the minimum sleep time and minimum cycle time parameters:

- Use the tuning panel to test values for the minimum cycle time. After you have determined the optimum value for the minimum cycle time, use STEP 7 to update and download the system configuration for the controller.

Changing the operating mode from STOP to RUN deletes any value entered by the tuning panel and resets the minimum cycle time to the value stored in the system configuration.

- To ensure that the controller executes the scan cycle on a fixed schedule, use the minimum cycle time parameter.
- To ensure that there is always a sleep interval even if the execution time changes, set the minimum cycle time to 0 (the default value) and modify the minimum sleep time as needed. Modifying the minimum sleep time is especially useful during the development of your STEP 7 user program.

When you are tuning the operation of the controller, be aware that the following situations can increase the time required to complete the scan cycle:

- The controller executes other OBs (such as OB 40 and OB 35) with higher priorities than OB 1.
- You use STEP 7 to monitor and debug the STEP 7 user program.
- You use a variable table (VAT) with STEP 7 to display the status of the STEP 7 user program.
- An application with a higher priority is running on your computer.
- The controller interacts with an HMI interface, such as WinCC.

## Additional Methods for Managing the Sleep Time

- Using SFC 47 to add sleep time in the STEP 7 user program
- Adjusting the sleep-monitoring algorithm of the execution monitor

## Using SFC 47 to Add Sleep Time in the STEP 7 User Program

SFC 47 (WAIT) inserts sleep time into the execution of the STEP 7 user program, allowing you to manage the sleep time for a control program by inserting the sleep time in a specific priority class. When the STEP 7 user program calls SFC 47, the controller suspends the execution of the OB for a specified number of microseconds and sleeps. During this sleep period, the controller can interrupt this sleep period to execute an interrupt OB. Because an OB with a higher priority class can interrupt the sleep time, higher priority OBs execute with less chance of jitter.

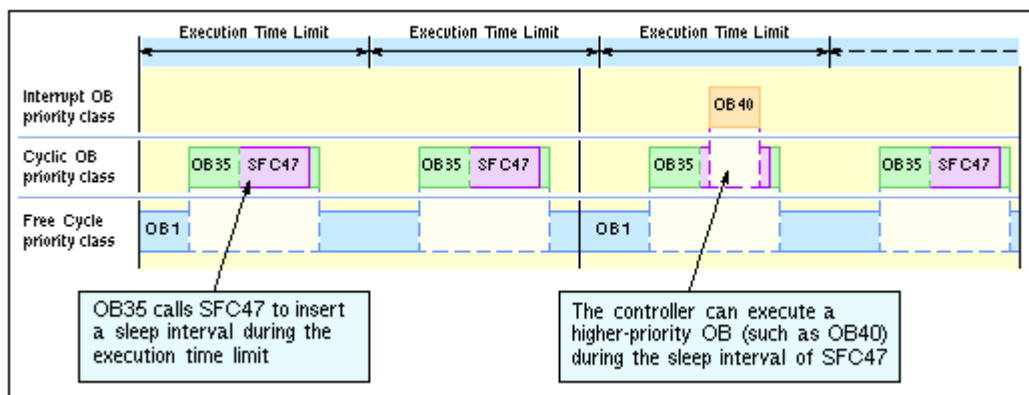
Typically, you call SFC 47 from a cyclic OB (such as OB 35) that starts within the execution time limit of the execution monitor.

For more information, refer to the example: Avoiding Jitter in the Start Time of an OB

To provide greater control over when the sleep time occurs, you can use SFC 47 to insert sleep time into your control program. Calling SFC 47 in the control program also allows you to define which OBs are affected by setting the priority class of the OB that calls SFC 47.

As shown in the following figure, you can use SFC 47 to insert a sleep interval that can satisfy the execution monitor and still allow the controller to handle an interrupt OB. By using a cyclic OB (such as OB 35) to call SFC 47, you can ensure that the sleep interval occurs within the execution time limit of the execution monitor.

The sleep time parameter is rounded up to the nearest multiple of the HAL timer period defined in the RTX Properties dialog. For example, if the HAL timer period is 500 microseconds (the default), and the sleep time parameter is 1200 microseconds, WinLC RTX rounds up the sleep time to 1500 microseconds.



### Additional Methods for Managing the Sleep Time

- Adjusting the Minimum Sleep Time and Cycle Time
- Adjusting the Sleep-Monitoring Algorithm of the Execution Monitor



## Adjusting the Sleep-Monitoring Algorithm of the Execution Monitor

The execution monitor uses a sleep-monitoring algorithm to ensure that the controller does not exceed a configurable maximum execution load for the CPU usage within a monitor interval.

The monitor interval is calculated as the amount of time such that the maximum load percentage of the monitor interval equals the entered execution time limit. The execution monitor calculates the forced execution sleep time as the difference between the monitor interval and the execution time limit.

The execution monitor determines whether to insert a forced execution sleep time if the OB execution exceeds the execution time limit.

If there is sufficient sleep time within the monitor interval, the execution monitor does not affect the execution of the program. Otherwise, the execution monitor forces a sleep interval. The default execution load is 90%, and the default execution time limit is 9 ms. For the default settings, the execution monitor calculates a monitor interval of 10 ms and a forced sleep interval of 1 ms.

The execution monitor runs asynchronous to the scan cycle and measures the amount of sleep time that occurs within the monitor interval and enforces a minimum sleep interval.

- If the scan cycle (execution time plus sleep time) is shorter than the monitor interval and the sleep time is greater than or equal to the forced sleep value: The execution monitor does not force a sleep interval.
- If the scan cycle is longer than the monitor interval: The execution monitor forces the controller to sleep for the required amount of time. Because the execution monitor runs in a higher priority class than any OB, the controller cannot interrupt the forced sleep interval. This could delay the start of an interrupt OB, such as OB 35 or OB 40.

Use the tuning panel to configure the parameters for the sleep-monitoring algorithm of the execution monitor.

For more information, see the example: [Avoiding Jitter in the Start Time of an OB](#)

### Contents of this topic:

- Operation of the Execution Monitor
- Parameters of the Sleep-monitoring Algorithm
- Configuring the Parameters of the Sleep-Monitoring Algorithm
- Situations that Cause the Execution Monitor to Force a Sleep Interval
- Situations that Prevent the Execution Monitor from Providing Sufficient Sleep Time

In addition to the sleep time that is added to the scan cycle (based on the minimum sleep time and minimum cycle time parameters), the execution monitor uses a sleep-monitoring algorithm that is based on a maximum execution load (percentage of CPU usage). For the default execution load (90% CPU usage), the execution monitor measures the length of time that the controller sleeps during the monitor interval of 10 ms and ensures that the controller sleeps for at least 1 ms.

By measuring the sleep time, the execution monitor ensures that the controller allows the other applications to access the computer resources while the controller sleeps. The execution monitor also provides the safety net in cases where there are programming errors (for example, an infinite loop in OB 100) that are not handled with other mechanisms.

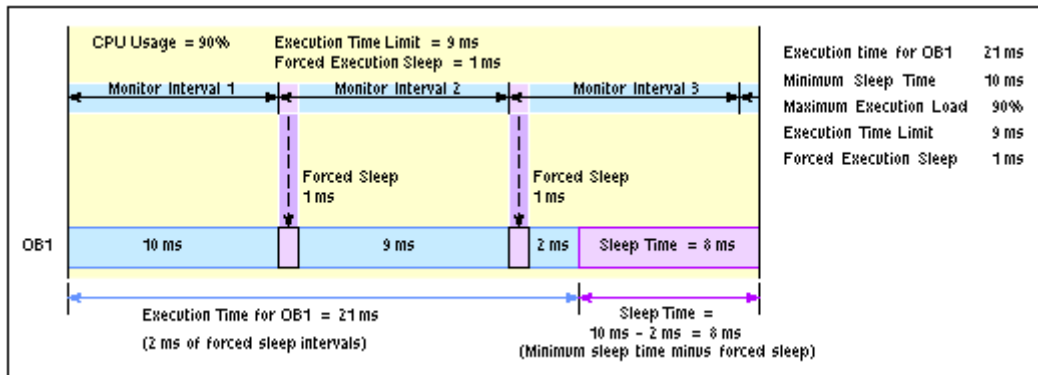
The difference between the forced sleep intervals and the minimum sleep time is that the controller can interrupt the minimum sleep time to handle interrupts (such as OB 35 or OB 40), but cannot interrupt the forced execution sleep time.

When the execution monitor forces a sleep interval, the following actions occur:

- The controller immediately suspends the execution of the OB for the forced sleep interval. By forcing a sleep interval, the execution monitor increases the actual time between starting and finishing the OB being executed.
- The controller cannot respond to the start event for any interrupt OB until the end of the forced sleep interval. Delaying the start of the OB (for example, OB 35 or OB 40) until the end of the forced sleep interval creates jitter or latency in the actual start time for the OB.

### Operation of the Execution Monitor

The following figure shows how the execution monitor might affect a control program. Because the execution time for OB 1 in this example is greater than the execution time limit, the execution monitor inserts a 1-ms sleep interval after the first two monitor intervals. However, the execution monitor does not insert a forced sleep interval in the third monitor interval because the controller sleeps longer than the required forced sleep interval as required by the configured minimum sleep time.



#### Note

The execution monitor runs asynchronous to the scan cycle. The example above shows the execution monitor measuring time from the beginning of the scan cycle, but because the execution monitor runs asynchronous to the control program, the beginning of the execution time limit of the execution monitor does not necessarily coincide with the beginning of the scan cycle.

## Parameters of the Sleep-Monitoring Algorithm

The sleep-monitoring algorithm of the execution monitor uses the following parameters:

Parameter	Description
Execution Time Limit	<p>This value defines the maximum time (in microseconds) that the execution monitor allows for OB execution before exceeding the configured maximum execution load (CPU usage) of the monitor interval.</p> <p>To determine the CPU load caused by the execution of the control program, the execution monitor measures the time that the controller sleeps during the monitor interval. If the controller does not use a sufficient amount of sleep time (indicating that the CPU load exceeds the maximum execution load), the execution monitor forces the controller to sleep for the remainder of the required forced execution sleep time.</p> <p>The default value is 9000 microseconds (9 ms).</p> <p><b>Note:</b> If you set this value greater than approximately 50000 (50 ms), you may observe jitter in Windows applications and in response to the mouse or keyboard. Test that the execution time limit you choose is appropriate for your application.</p>
Maximum Execution Load	<p>This value defines the maximum percentage of CPU usage that is allowed for the controller to execute OBs during each monitor interval.</p> <p>The default value is 90%.</p>
Forced Execution Sleep	<p>This read-only field shows how much sleep time (in microseconds) the execution monitor requires during the monitor interval to satisfy the requirement for the maximum execution load. The execution monitor subtracts any controller sleep time that occurs during a monitor interval from the forced execution sleep time to determine how much sleep time (if any) to force.</p> <p>The forced execution sleep time is a calculated number based on the execution time limit and the maximum execution load. The execution monitor corrects this value as required, depending on the capability of the operating system configuration to have timers operate at the specified intervals. For example, if the HAL timer period (in the RTX Properties dialog) is set to 500 microseconds, you cannot have a forced execution sleep time of 1200 microseconds. It would be rounded up to 1500 microseconds.</p> <p>The default value is 1000 microseconds (or 1 ms).</p>

The execution monitor uses the execution time limit and the maximum execution load to calculate the forced execution sleep. For example, the execution monitor uses the 90% usage rate and the 9-ms execution time limit to calculate a 1-ms sleep interval. In this case, the monitor interval is 10 ms such that 90% of the monitor interval corresponds to the entered execution time limit (9 ms).

During the monitor interval, the execution monitor measures the actual amount of time that no OBs are executing (the sleep time).

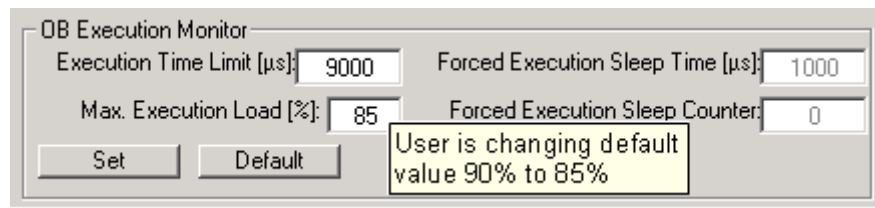
- If the controller sleeps **longer** than the sleep interval (forced execution sleep time), then the execution monitor restarts another monitor interval and does not affect the control program.
- If the controller sleeps **less** than the sleep interval (forced execution sleep time), then the execution monitor blocks the execution of any OBs for the **remainder** of the sleep interval.

Any control program sleep time imposed because of the sleep-monitoring algorithm is subtracted from the sleep time configured for the end of the free cycle as defined by the minimum sleep time parameter.

The default value for the “Execution Time Limit” interval is 9000 microseconds (or 9 milliseconds) and the default value for the “Forced Execution Sleep” interval is 1000 microseconds (or 1 millisecond). This ratio ensures that the control program execution cannot use more than 90% of the CPU time in any of the worst case situations described above.

### Configuring the Parameters of the Sleep-Monitoring Algorithm

The parameters of the sleep-monitoring algorithm of the execution monitor are configurable from the tuning panel:



To change the sleep-monitoring parameters, follow these steps:

1. Enter values in the Execution Time Limit and the Max. Execution Load fields. You can change one of the fields or both.
2. Click Set to set the parameters.

To restore the default sleep-monitoring parameters, follow these steps:

1. Click Default to display the default parameters.
2. Click Set to set the default parameters.

Changes to the sleep-monitoring parameter take effect when the controller is in RUN mode.

## Situations that Cause the Execution Monitor to Force a Sleep Interval

The controller must relinquish control of the CPU long enough to satisfy the maximum execution load. Typically, the sleep time that is added to the end of the scan cycle allows sufficient time for the operating system to process the other Windows applications. However, some situations may require that the execution monitor force a sleep interval.

Condition	Description
Execution time for the control program exceeds the execution time limit	The configured minimum sleep time for the free cycle occurs after OB 1 finishes. If the execution time is longer than the execution time limit, the execution monitor forces a sleep interval because the controller did not sleep for the required amount within the monitor interval.
Minimum sleep time is insufficient for the maximum execution load	Even when the scan cycle is less than the execution time, the minimum sleep time may not provide enough sleep time. In this case, the controller would exceed the maximum execution load. The execution monitor forces an additional sleep interval to ensure that the operating system can run the other applications.
Interrupt OBs reduce the sleep time	<p>To process an interrupt OB (such as OB 35, OB 40, or OB 85), the controller can interrupt the sleep time for the scan cycle. This reduces the time that the controller actually sleeps and can cause the controller to exceed the maximum execution load, which affects the performance of the other Windows applications.</p> <p>By forcing a sleep interval, the execution monitor ensures that the other Windows application can be processed.</p>

### Situations that Prevent the Execution Monitor from Providing Sufficient Sleep Time

In some cases, a high execution time limit can prevent the execution monitor from managing the sleep time of the control program adequately. Under the following conditions, the control program utilizes too much CPU time, which can result in jitter in Windows response time to the mouse, keyboard, or other applications. For either case, the problem can be resolved by lowering the execution time limit.

Condition	Description
Execution time for the startup OB (OB 100 or OB 102) and the configured execution time limit exceed approximately 50 ms	<p>During startup, the controller turns the watchdog timer off and cannot handle a program error, such as a loop in the logic of the OB or an excessively long initialization routine.</p> <p>Because the scan cycle does not provide any sleep time for the startup OB (such as OB 100), the execution monitor cannot relinquish CPU time for other applications. If the startup OB executes for more than approximately 50 ms, jitter can occur in Windows response time to the mouse, keyboard, or other applications.</p>
Execution time for the control program and the configured execution time limit exceed approximately 50 ms	<p>Whenever the operating system has to wait more than approximately 50 ms to process the other Windows applications, the performance of those applications can be noticeably affected. This can be a problem for an OB 1 with a long execution time, especially if other OBs (such as OB 35 or OB 40) extend the execution of OB 1.</p> <p>Because the sleep time is added at the end of the scan cycle, and the execution time limit is set to a high value, the sleep intervals are then spaced too far apart for the other Windows applications to perform naturally.</p>

### Additional Methods for Managing Sleep Time

- Adjusting the minimum sleep time and minimum cycle time parameters
- Inserting sleep time into the control program (SFC 47 "WAIT")

### Example: Avoiding Jitter in the Start Time of an OB

The following example discusses two possible solutions for a program that experiences jitter in the start of a cyclic interrupt (OB 32 to OB 36).

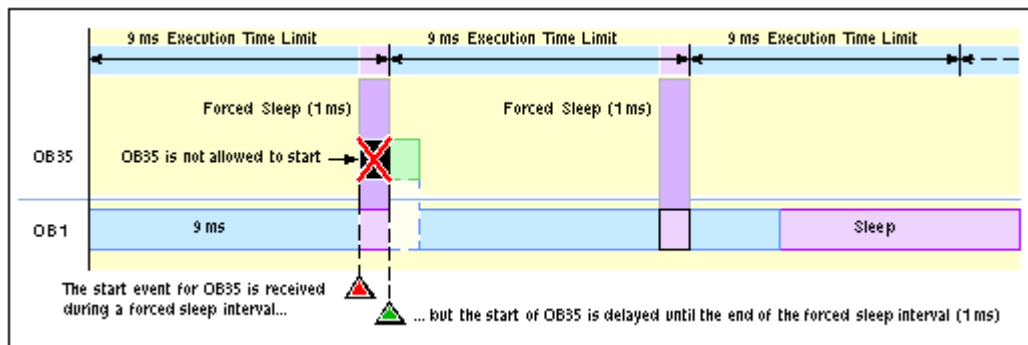
- Inserting a sleep interval into the execution of your STEP 7 user program. For this solution, you call SFC 47 ("WAIT") and specify the length of time to sleep. The controller can interrupt this sleep interval to process other OBs.
- Changing the sleep-monitoring algorithm of the execution monitor. For this solution, you use the tuning panel to change the execution time limit.

#### Scenario

The following example describes the execution of a STEP 7 user program that consists of OB 1 and OB 35. OB 1 takes 20 ms to execute, and OB 35 starts every 100 ms and takes 1 ms to execute. The controller has been configured with the following parameters:

Parameter	Value
Execution Time for the STEP 7 user program	OB 1: 20 ms, and OB 35: 1 ms
Minimum Sleep Time	10 ms (uses the default value)
Minimum Cycle Time	0 ms (uses the default value)
Maximum Execution Load	90% (uses the default wake/sleep algorithm)
Execution Time Limit	9 ms (uses the default value)
Forced Execution Sleep	1 ms (uses the default value)

The sleep time (10 ms) is added to the scan cycle after OB 1 has finished. However, because the execution time for OB 1 (20 ms) exceeds the execution time limit (9 ms), the controller exceeds the configured maximum execution load (90%) by not sleeping during the execution time limit. Therefore, the sleep-monitoring algorithm forces the controller to sleep for 1 ms after every 9 ms that OB 1 executes. As shown in the following figure, this forced sleep can cause a variance or jitter in the time that the start event and the time that the controller starts to execute OB 35. This jitter happens because all controller operations are suspended during a forced sleep interval. Similarly, OB 35 could be suspended for 1 millisecond if the end of the execution time limit interval occurs while OB 35 is executing.



For many applications, a 1-ms jitter might be acceptable. However, you have several options for removing this jitter:

- You can modify the STEP 7 user program to call SFC 47 and insert sleep time that can be interrupted by OB 35.
- You can adjust the parameters for the sleep-monitoring algorithm to avoid the jitter caused by the execution monitor.

### Solution 1: Insert a sleep interval into the execution of your STEP 7 user program

You could avoid the forced sleep interval by using SFC 47 to add a periodic sleep interval that occurs within the execution time limit (for this example, 9 ms). This sleep interval not only ensures that the sleep-monitoring algorithm does not force the controller to sleep, but also allows the controller to suspend this sleep interval and execute any OB that has a higher priority than the OB that called SFC 47.

For this example, you can use SFC 47 to remove the jitter in OB 35:

- By ensuring that SFC 47 executes at a specified time. The STEP 7 user program calls SFC 47 from an OB (such as OB 36) that has a priority greater than OB 1.
- By ensuring that OB 35 executes as scheduled. You configure OB 36 to have a lower priority than OB 35.
- By ensuring a sufficient sleep interval during the execution time limit. You configure SFC 47 to wait for 3 ms, which ensures a sleep interval of at least 2 ms.

To maintain a 50% ratio for CPU usage (20 ms execution time for OB 1 with a 10 ms minimum sleep time), configure OB 36 to run every 6 ms (so that OB 1 executes for 6 ms, then sleeps for 3 ms). You can then change the minimum sleep time to 0 ms, unless you want to decrease the ratio for CPU usage.

To create an OB 36 that calls SFC 47 to create a 3 ms sleep interval:

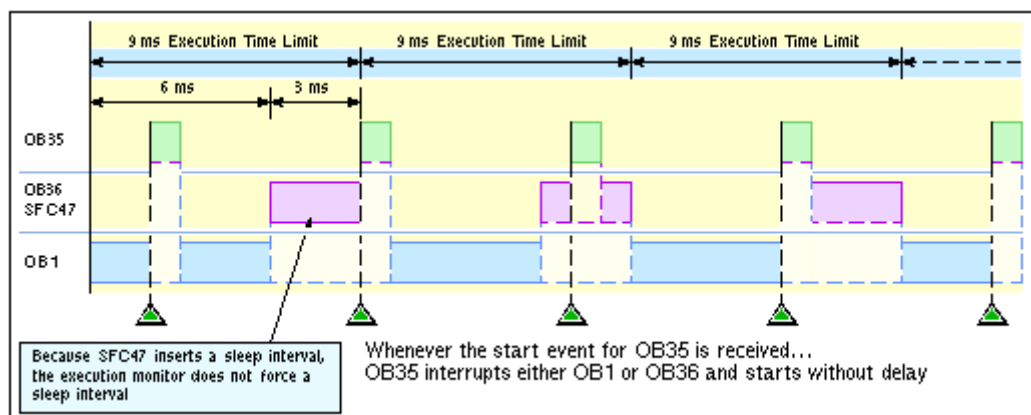
1. Using the STEP 7 Program Editor: Create an OB 36 for your STEP 7 user program, and enter the following program:

```
CALL "WAIT" // SFC 47 wait function
WT: 3000 // 3000 microseconds or 3 milliseconds
```

2. Using the STEP 7 Hardware Configuration tool, configure the priority level and execution time for OB 36:

- Open the WinLC Properties dialog box and select the Cyclic Interrupt tab.
- Set the priority for OB 36 to 2 (or any other priority lower than the priority for OB 35).
- Configure OB 36 to execute every 6 ms (by entering 6 in the Execution field).

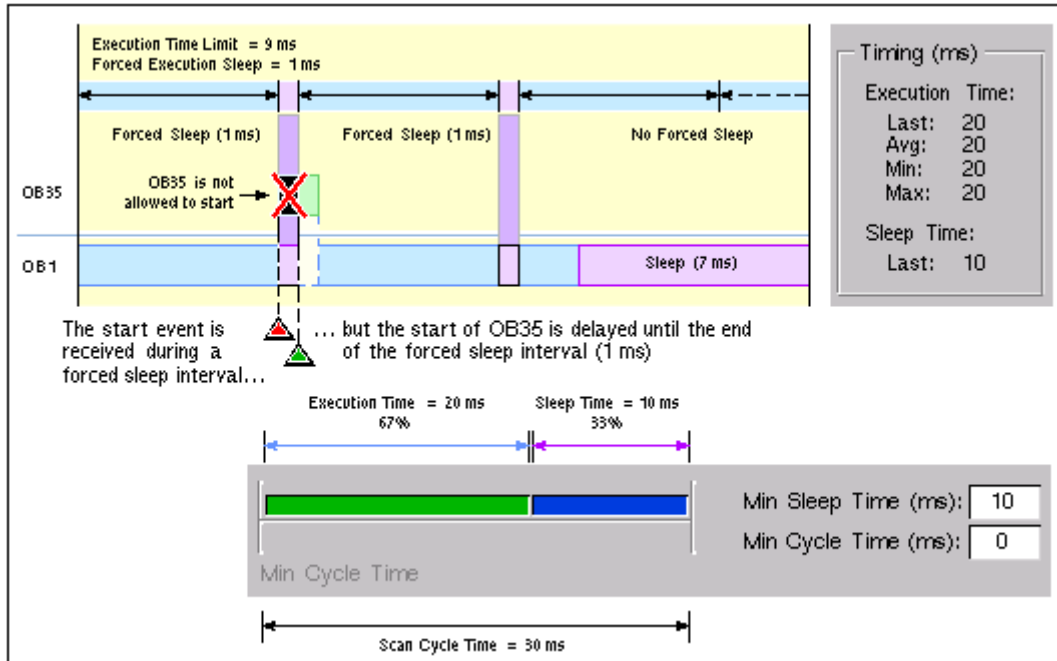
The following figure shows how SFC 47 affects the execution of the STEP 7 user program. Because OB 36 ensures that the controller sleeps at least 1 ms within the 90% wake interval, the execution monitor does not insert a forced sleep interval. Therefore, OB 35 executes without any delay or jitter.





### Solution 2: Change the sleep-monitoring algorithm to eliminate the forced sleep interval

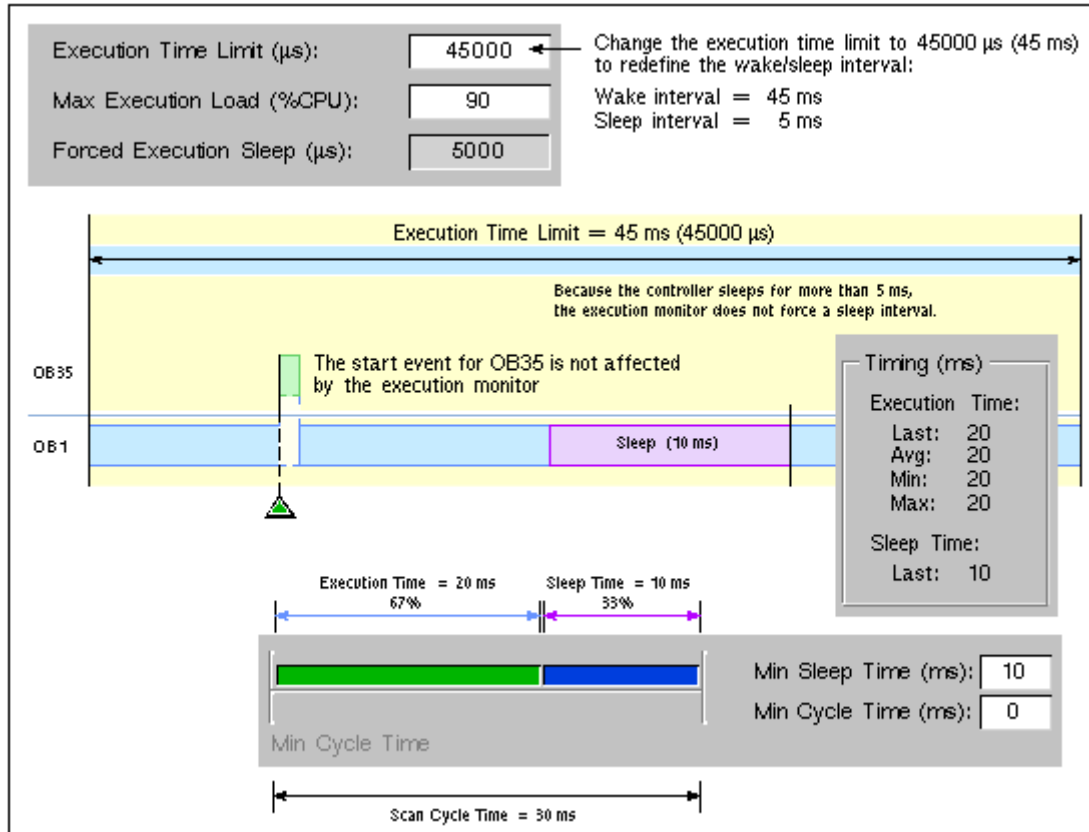
The following figure shows the jitter in the start time of OB 35 and also shows the values displayed by the tuning panel. Notice that the tuning panel shows only the information about OB 1. The tuning panel does not display information about OB 35. For this example, the execution time for OB 1 is 20 ms. With the minimum sleep time of 10 ms, the total free cycle time is 30 ms. OB 35 and other interrupt OBs can make the total scan time more than this, depending on how fast the interrupt OBs execute.



By changing the parameters of the sleep-monitoring algorithm, you can configure the execution monitor to use the minimum sleep time in the free cycle. For example: if the longest total scan time for this example is less than 45 ms, change the execution time limit to 45000 microseconds (45 ms):

1. Open the tuning panel.
2. Change the execution time limit to 45000 (microseconds). For this example, do not change the value for the maximum execution load.
3. Apply the new value.

The following figure shows the effect of the changed execution time limit.



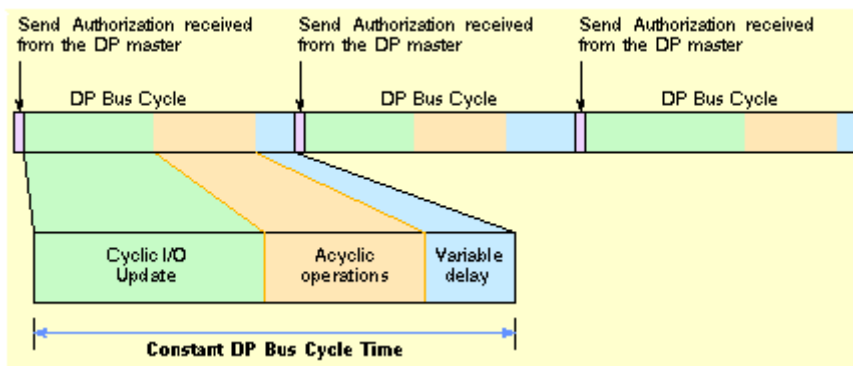
## Isochronous Mode for a Constant Bus Cycle

With WinLC RTX, you can operate the DP Master in an isochronous mode to maintain a constant bus cycle time.

**Note:** WinLC RTX allows you to use isochronous mode on more than one PROFIBUS-DP subnet; however, your computer must not share the interrupt (IRQ) of the PCI slots used by the DP interfaces with any device operating in the Windows operating system (for example, a video card). For example, the SIMATIC Box PC 627 provides two PCI slots that can be used for isochronous mode on two different PROFIBUS-DP subnets.

To implement an isochronous DP cycle, you assign a synchronous interrupt OB (OB 61 or OB 62) with an associated process image partition to the DP master for synchronous update. Each isochronous DP cycle contains the following elements:

- A global control command (Send Authorization) notifies the slave devices of the start of the bus cycle.
- The cyclic inputs and outputs are updated.
- Any acyclic operations are performed.
- A variable delay allows the next DP cycle to start on the next multiple of the configured cycle time.



During the bus cycle, two events signal the STEP 7 user program:

- At the end of the I/O update, an interrupt schedules the synchronous OB for execution.
- At the start of the succeeding cycle (when the Send Authorization command is being transmitted to the slave devices), an event signals WinLC RTX that further execution of SFC 126 and SFC 127 should return an error.

Between the two events (between the interrupt and the transmission of the global control command), the synchronous OB can call SFC 126 and SFC 127 to execute the synchronous updating of the process image partition that was assigned to the synchronous OB. If these SFC calls execute without error, the I/O update is synchronized to the process image partition update and occurs at a constant interval between updates.

You configure the DP bus cycle when you configure network properties for the DP master.

## System Requirements for an Isochronous DP Cycle

For an isochronous DP cycle, your system requires a CP 5613 card, hardware revision 6 or higher operating in interrupt mode.

To determine the revision level, you can use either the Set PG/PC Interface utility of STEP 7 or you can view the submodule diagnostics.



# Connecting the Controller to the SIMATIC NET OPC Server






WinAC RTX can use the SIMATIC NET OPC server to read and write data over the network. You use the following tools to configure the OPC connection:

- OPC Scout for configuring the connection to the SIMATIC NET OPC server
- STEP 7 (HW Config and NetPro) for configuring the WinAC RTX controller
- Station Configuration Editor for configuring the PC station

Configuring an OPC Server connection requires the installation of SIMATIC NET.

**Note:** The critical step most frequently overlooked is Step 3: Configuring the S7 connection for the OPC server in NetPro. After adding the connection for the OPC server, you must set the connection type to "S7 connection" and enter a Local ID for the connection.

## Task Overview

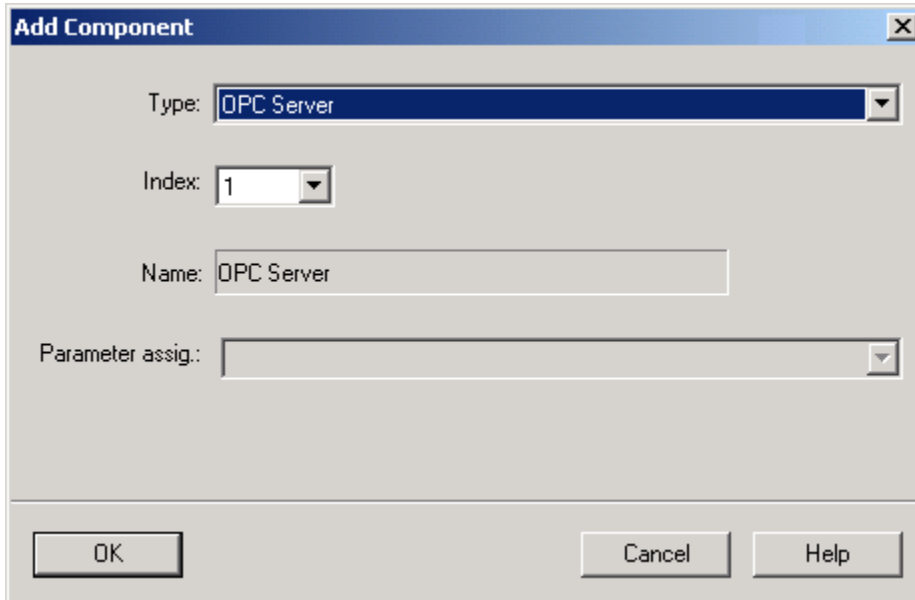
	<b>Step 1: Station Configuration Editor (SIMATIC NET)</b> Add the OPC server to the PC station.
	<b>Step 2: HW-Config (STEP 7)</b> Add the OPC server to the hardware configuration in STEP 7.
	<b>Step 3: NetPro (STEP 7)</b> Add an S7 connection for the OPC server to the configuration of WinLC RTX.
	<b>Step 4: SIMATIC Manager (STEP 7)</b> Download the configuration to the controller.
	<b>Step 5: OPC Scout (SIMATIC NET)</b> Connect the controller to the OPC server.

## Step 1: Add the OPC Server to the PC Station

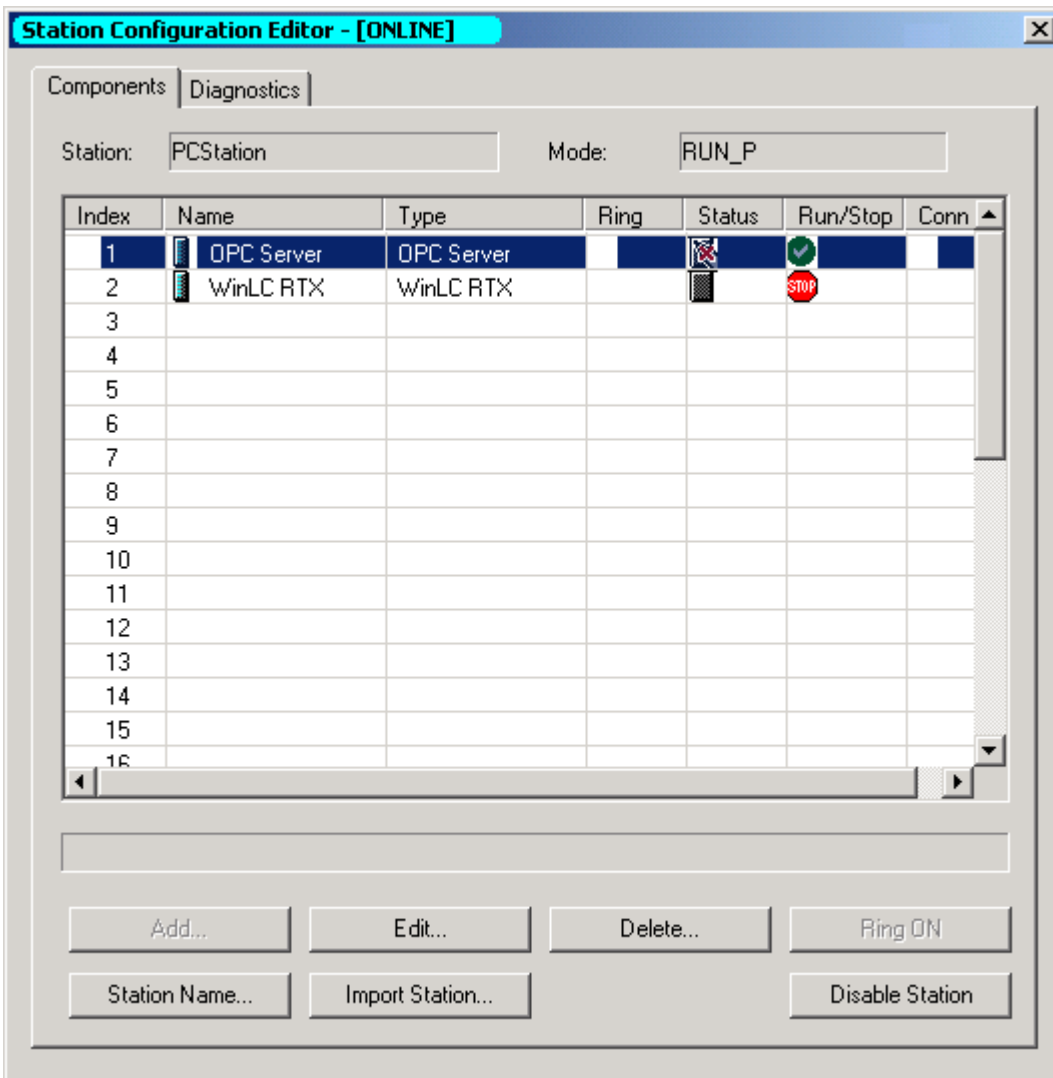
Tool:  Station Configuration Editor (SIMATIC NET)

To configure the OPC server in the PC Station, follow these steps:

1. Open the Station Configuration Editor and select any index in the Station Configuration Editor.
2. Right-click the mouse to display the Add button. Click the Add button to display the Add Component dialog.
3. Select "OPC Server" from the drop-down list of component types:



4. Click OK to add the OPC server to the station configuration. The Station Configuration Editor displays the OPC Server in the index selected. (For this example, the OPC server is configured for Index 1.)
5. Click OK to save the PC station configuration and to close the Station Configuration Editor dialog.



## Step 2: Add the OPC Server to the Hardware Configuration

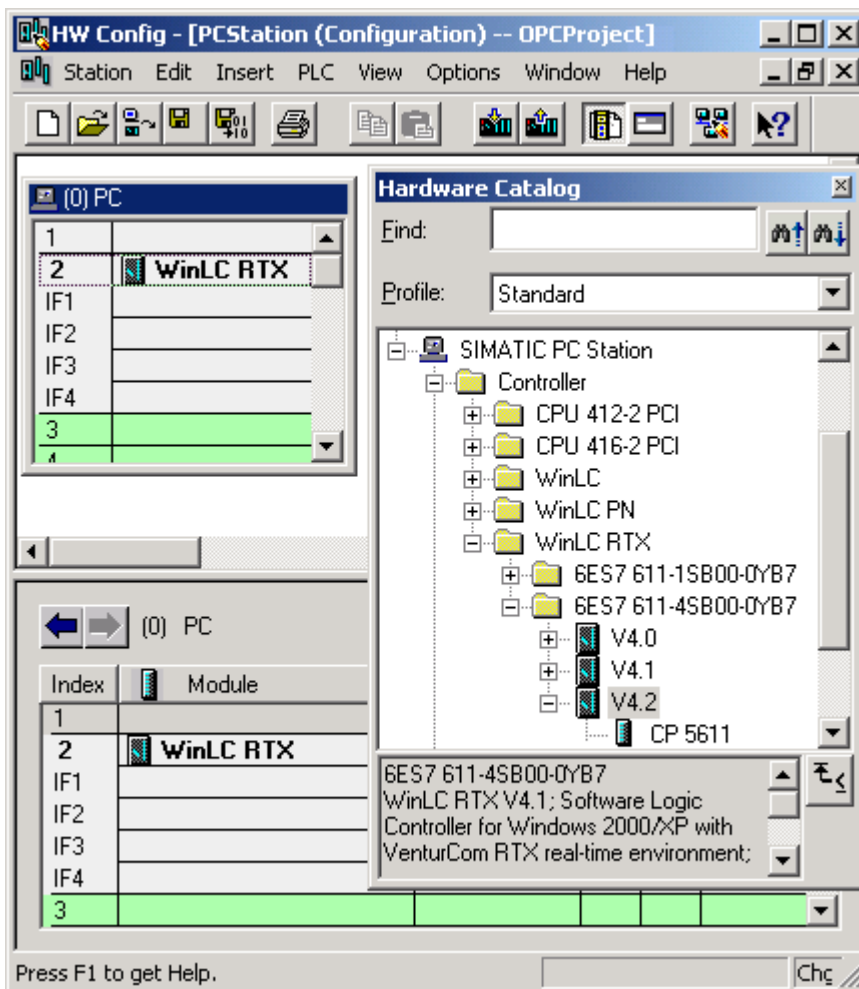
 **HW Config (STEP 7)**

### Task Summary:

- Create a STEP 7 project for a PC station with WinAC RTX.
- Insert the OPC server into the hardware configuration.
- Configure the OPC server.

### Creating the STEP 7 Project

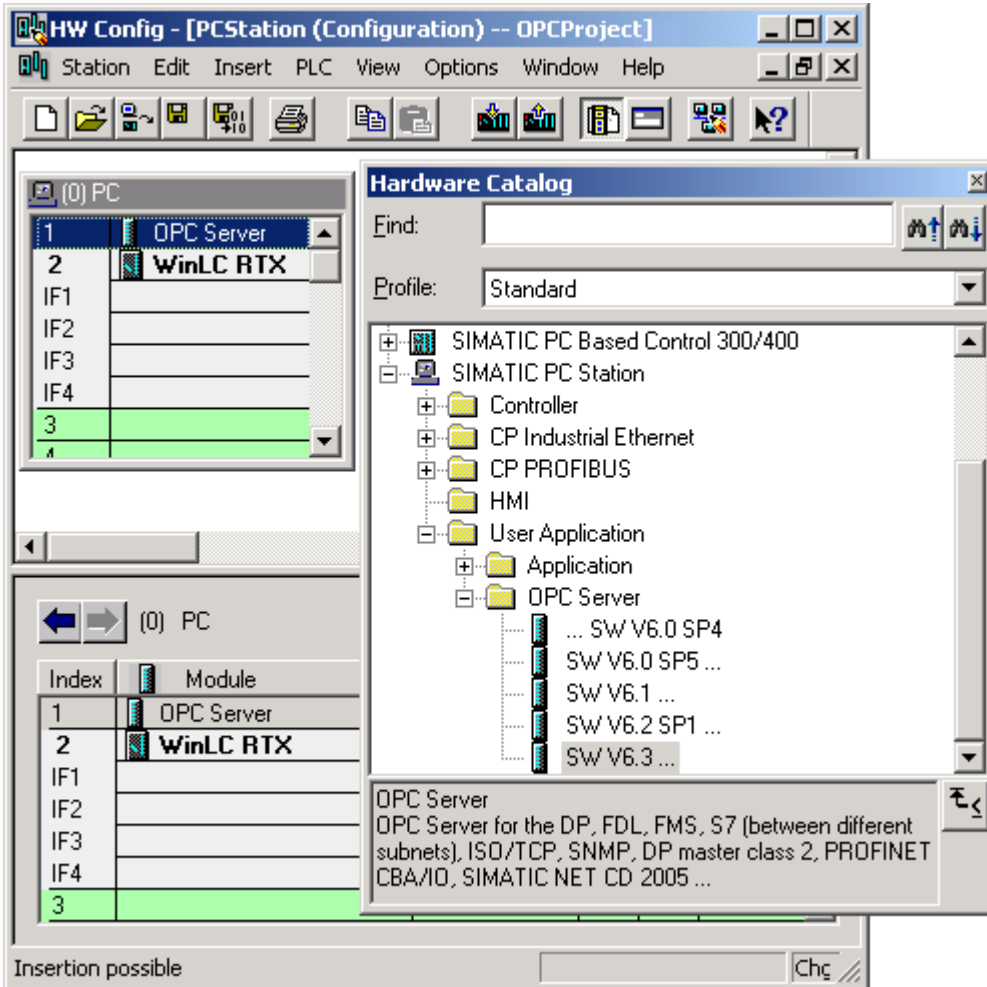
1. Open STEP 7 and create a project (for example, OPCProject).
2. Insert a SIMATIC PC Station with the same name as entered in the Station Configuration Editor.
3. Double-click the Configuration icon for the PC Station to open the STEP 7 Hardware Configuration utility.
4. Insert the WinAC RTX controller in the same index as configured in the Station Configuration Editor.





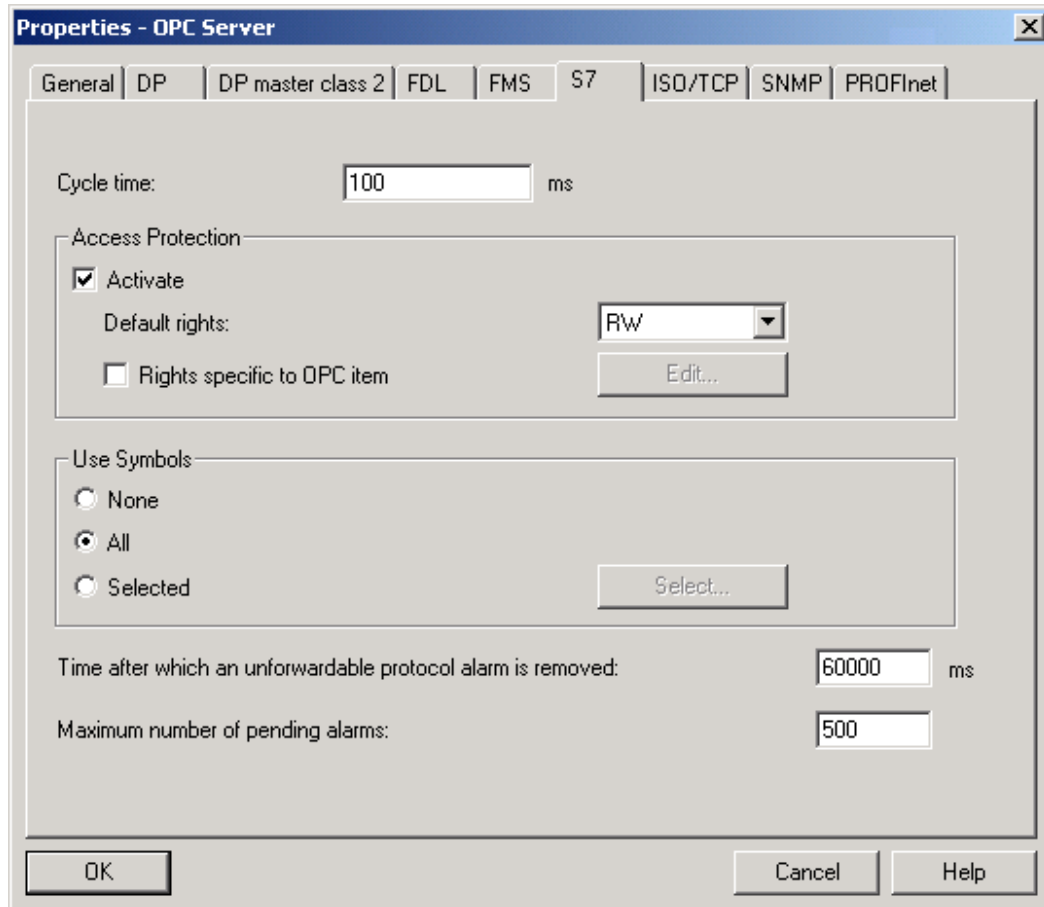
## Adding the OPC Server to the Hardware Configuration

1. Expand the User Application folder in the Hardware Catalog.
2. Expand the OPC Server folder and select the following component:  
SW V6.3
3. Drag and drop the SW V6.3 component to in the same index as configured in the Station Configuration Editor. (For this example, the OPC server is configured for Index 1.)



## Configuring the OPC Server

1. Double-click the OPC Server entry (Index 1) to open the Properties dialog.
2. Click the S7 tab and select the Activate option (under Access Protection).
3. To use the STEP 7 symbols for accessing controller data from the OPC Server, select the option for All (or for Selected, to specify specific entries in the symbol table) under the Use Symbols field.
4. Click OK to close the Properties dialog.
5. Click the Save and Compile icon to create the hardware configuration for the PC station.



After you have compiled the configuration into the STEP 7 project, you can close HW Config and return to SIMATIC Manager.

### Step 3: Add an S7 Connection for the OPC Server in NetPro

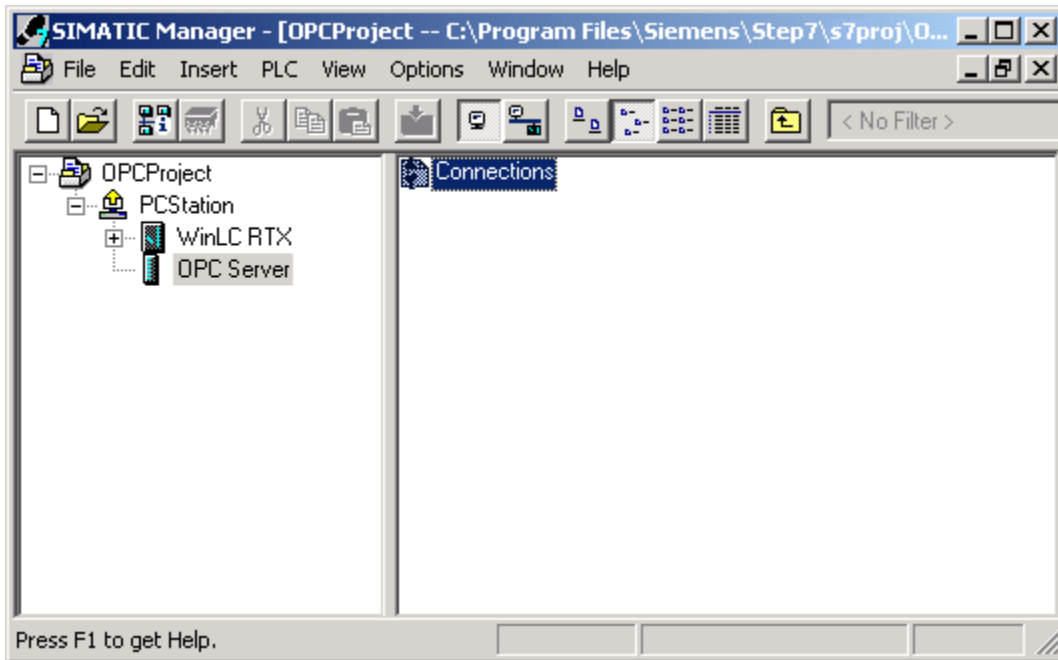
Tool:  NetPro (STEP 7)

**Task Summary:**

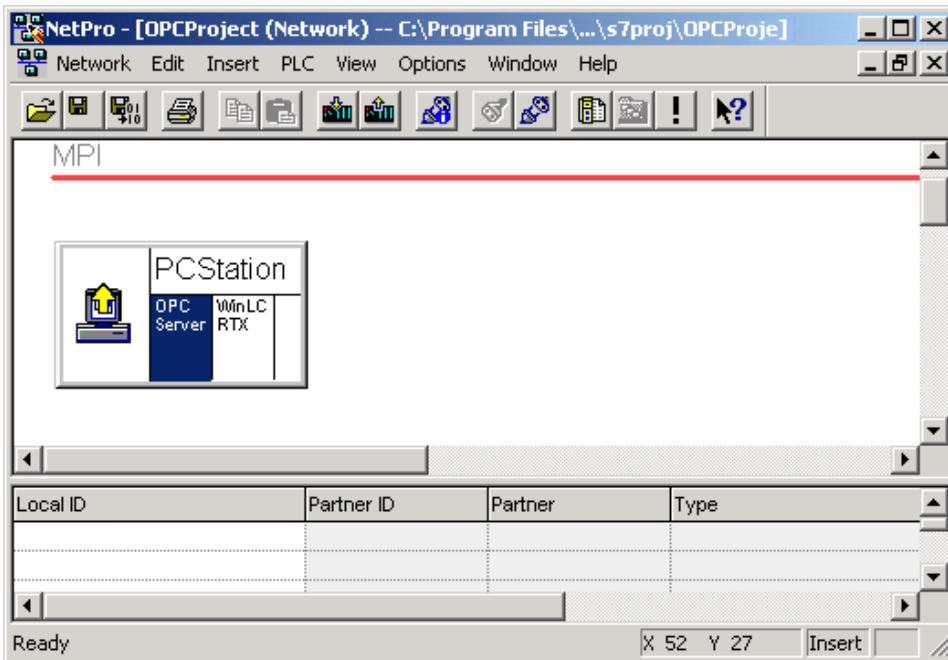
- Configure an S7 connection for the OPC server to the PC Station configuration.
- Assign a Local ID for the OPC server connection.

### Configuring an OPC Server Connection in NetPro

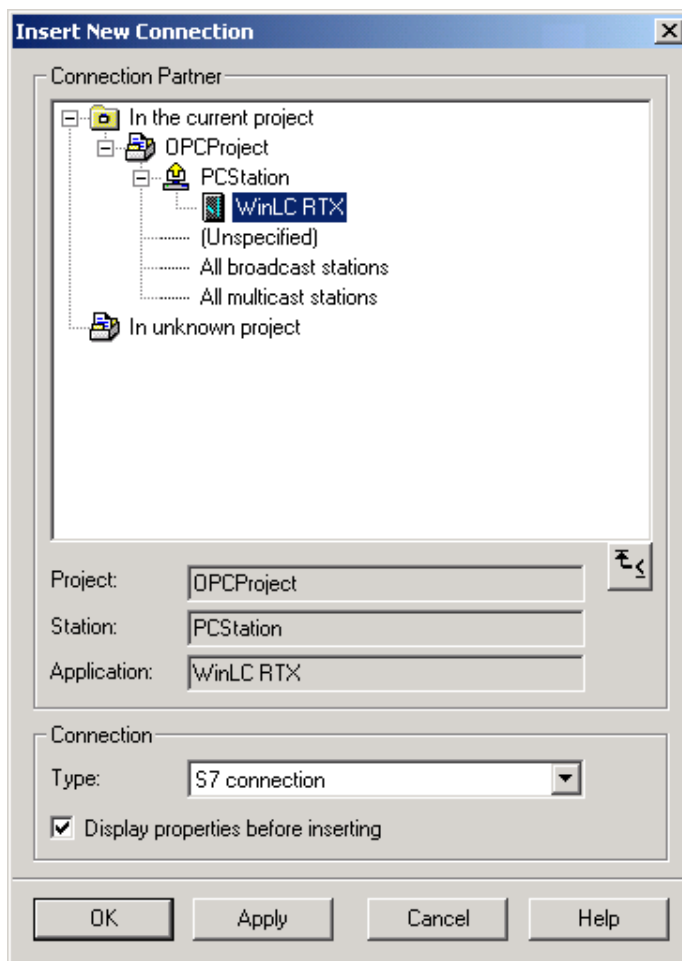
1. In SIMATIC Manager, browse to the OPC server and double-click the Connections icon to open NetPro.



2. Select the OPC Server in the PC station.



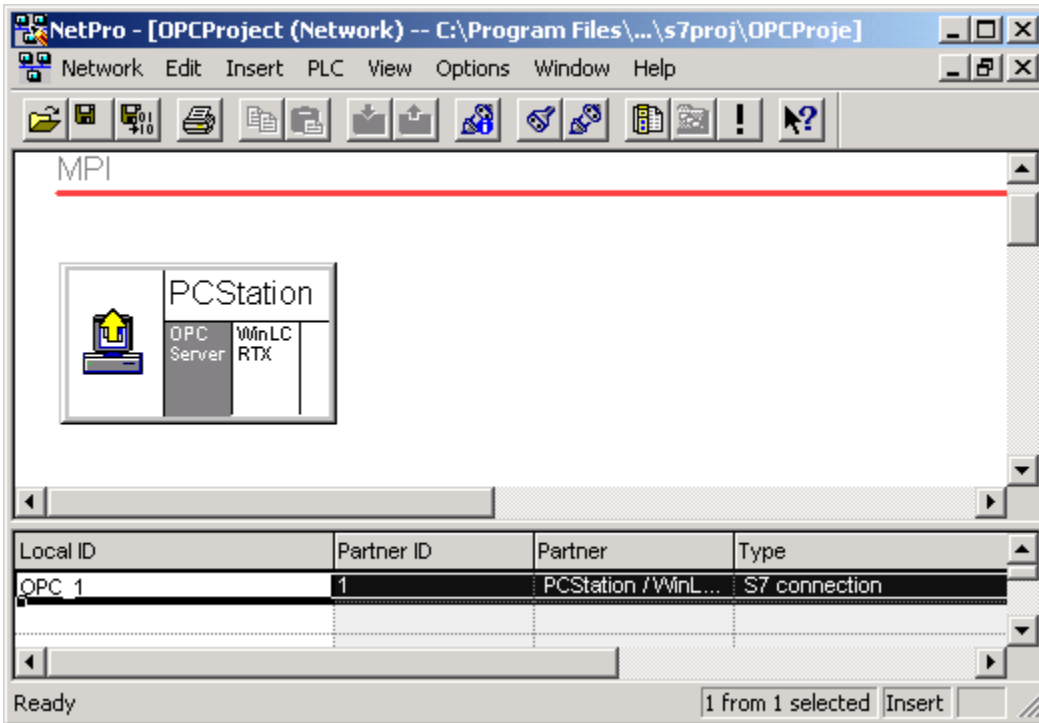
3. Right-click the OPC server to display the context menu. Select the **Insert New Connection** menu command to open the Insert New Connection dialog.



4. Set the connection type to S7 connection and click OK to add the S7 connection for the OPC server. The Properties dialog for the S7 connection opens automatically.

### Assigning a Local ID for the OPC Server Connection

1. In the Properties dialog, enter the Local ID for the S7 connection (such as OPC\_1).
2. Click OK to add the S7 connection to NetPro.
3. Click the Save and Compile icon to save and compile your changes into the STEP 7 project.



After you have compiled the S7 connection for the OPC server into the STEP 7 project, you can close NetPro and return to SIMATIC Manager.

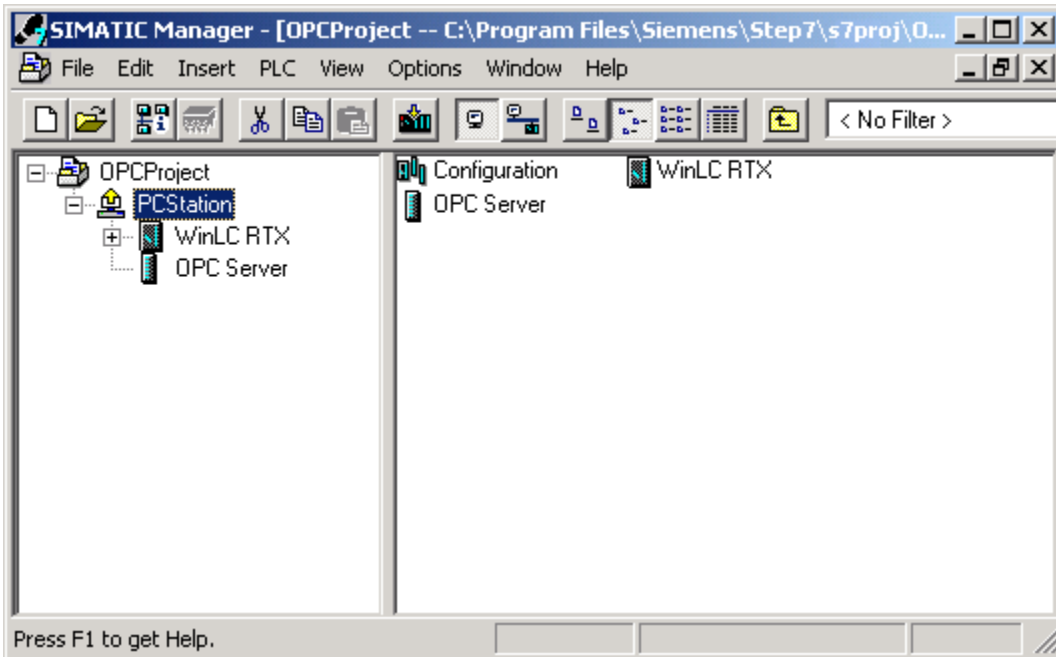
## Step 4: Download the Configuration to the Controller

Tool:  SIMATIC Manager (STEP 7)

**Note:** The controller must be executing to download the configuration from STEP 7.

To download the configuration, follow these steps:

1. If the controller is not executing, start it executing.
2. In SIMATIC Manager, select the SIMATIC PC Station icon.
3. Select the **PLC > Download** menu command or click the Download icon on the toolbar.



## Step 5: Connect the Controller to the OPC server

Tool:  OPC Scout

### Task Summary:

- Create an OPC Project.
- Add the connection to the SIMATIC NET OPC server.
- Define the items to be accessed through the OPC server.

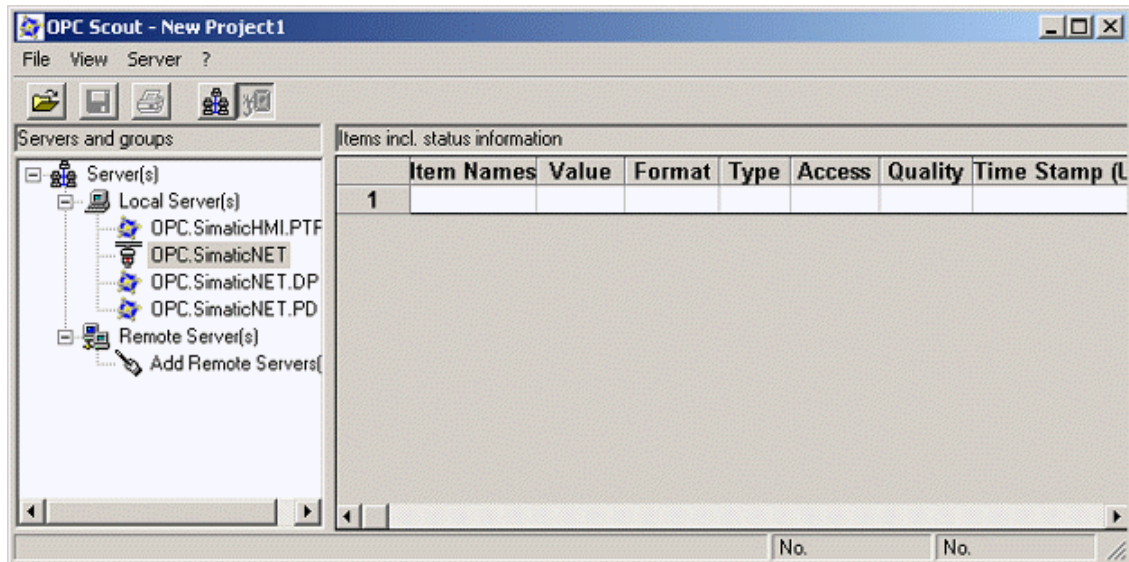
### Creating an OPC Project

Select the **Start > SIMATIC > SIMATIC NET > OPC SCOUT** menu command to create a new project in OPC Scout.

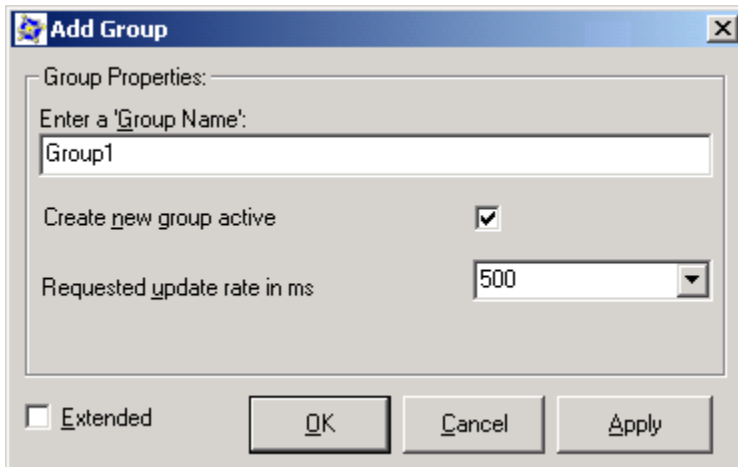
### Adding a Connection (Group) for the OPC Server

To add a connection to the SIMATIC NET OPC server, follow these steps:

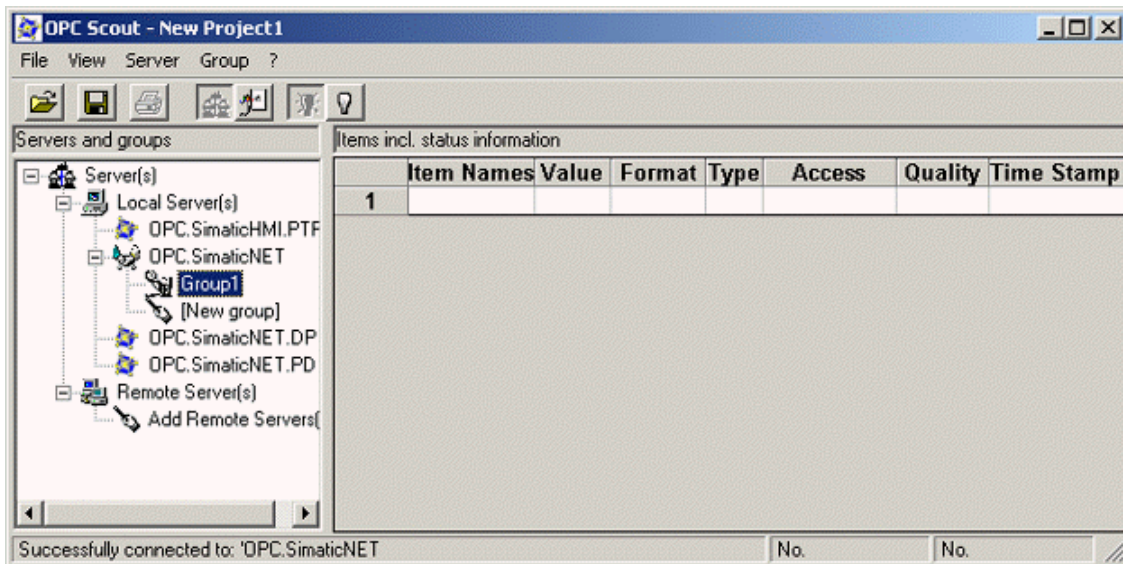
1. Expand the Local Server(s) directory in the Servers and Groups for the project.
2. Double-click the OPC.SimaticNet element to add a connection (or group) for the SIMATIC NET OPC server.



3. In the Add Group dialog, enter the Group Name for the connection (for example, Group1).



4. Click OK to add the group to the OPC server. OPC Scout adds the connection (Group1) to the OPC server.



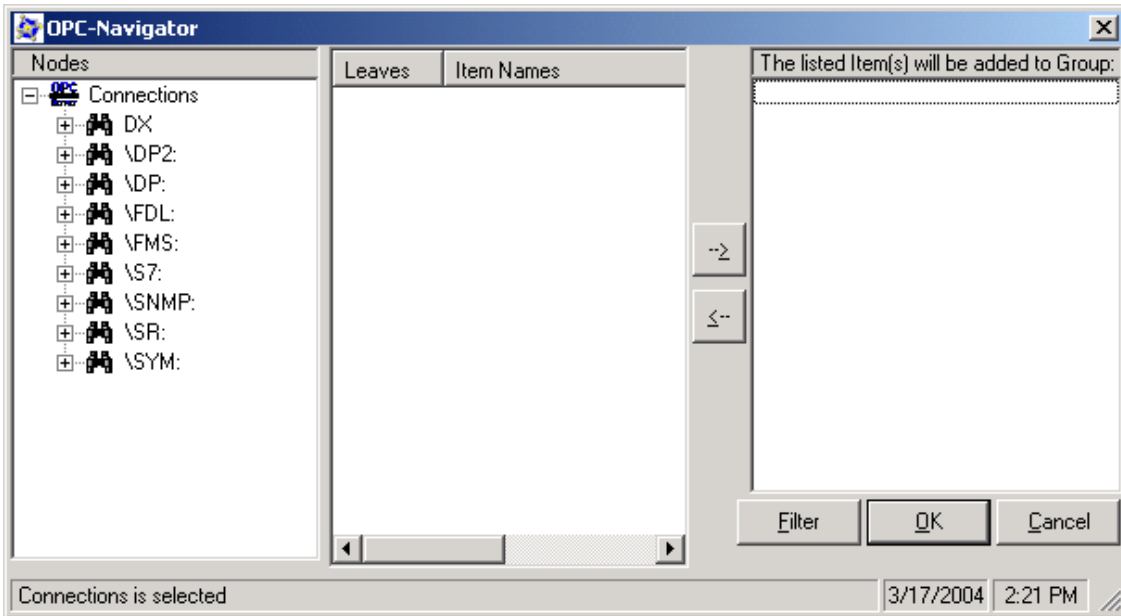


## Configuring the Items to be Accessed (Using Absolute Addressing)

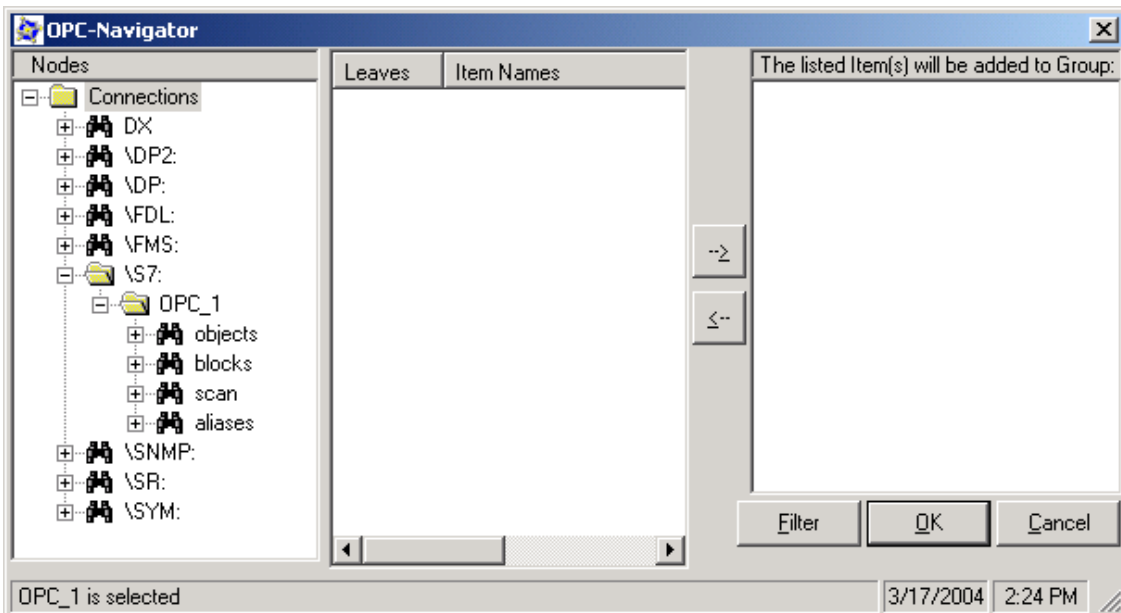
**Note:** This procedure describes how to use absolute addressing when configuring the OPC server. You can also use the STEP 7 symbol table for connecting the OPC server, as described in the section "Configuring the Items to be Accessed (Using the STEP 7 Symbol Table)".

Use the following procedure to configure the OPC server to use an absolute address for accessing data in the controller:

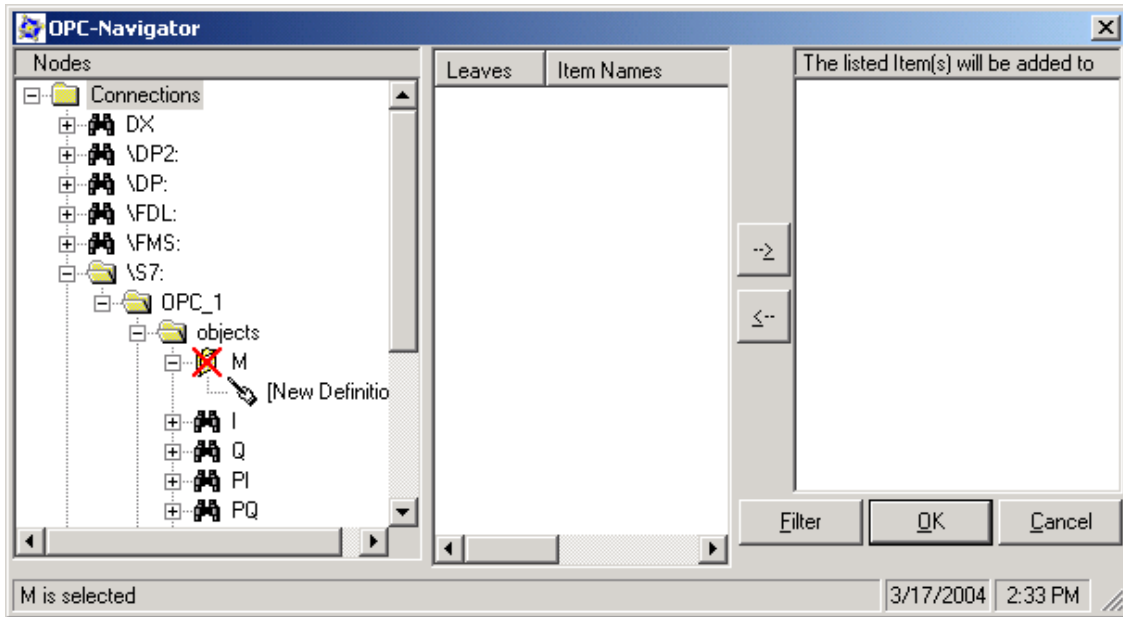
1. Open the OPC Navigator by double-clicking the connection (Group1) for the OPC server.



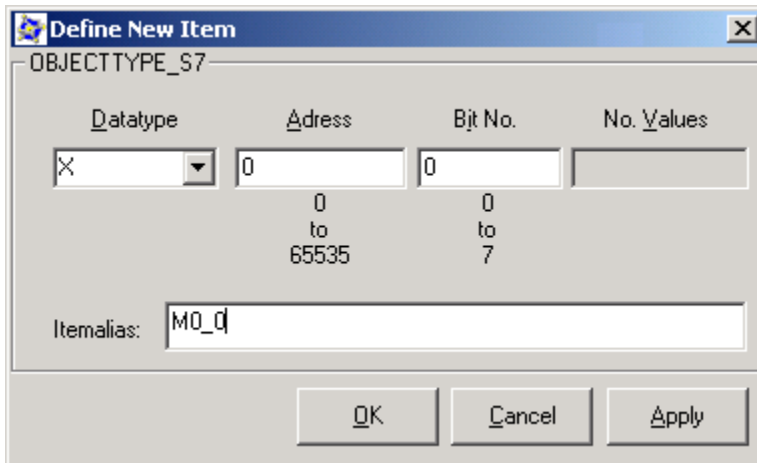
2. To add an item to be accessed, expand the \S7: folder and select OPC\_1.



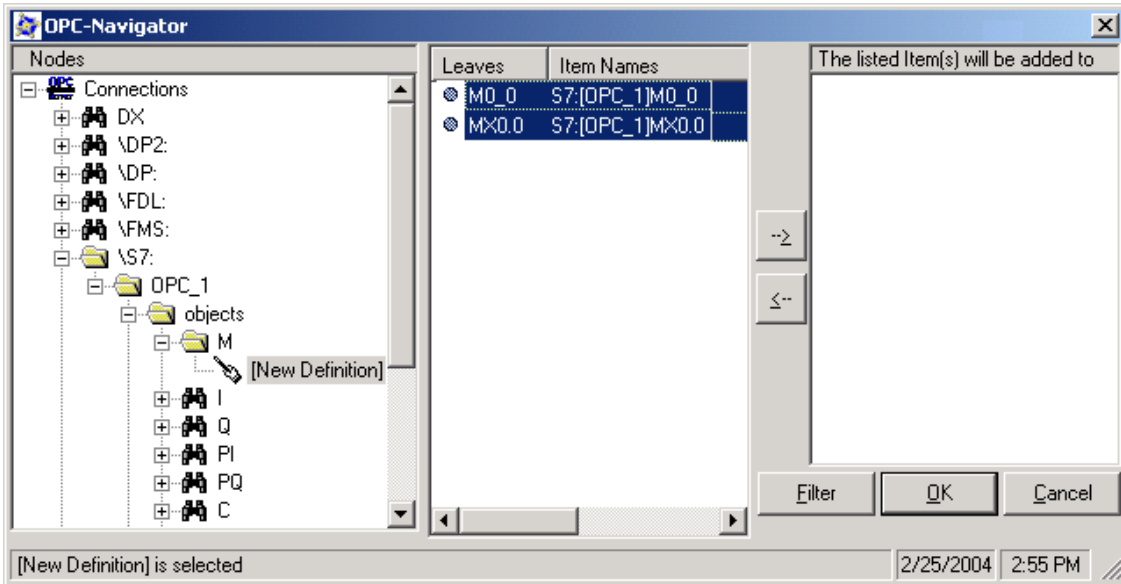
- To configure access to M 0.0, expand the Objects folder and expand the M folder (for the bit memory area).



- Double-click the New Definition icon to open the Define New Item dialog.
- To define a connection for M0.0, select X (for bit) field from the drop-down list in the Data Type and enter the byte address (0) and bit number (0). (You can also enter an alias for the item.)



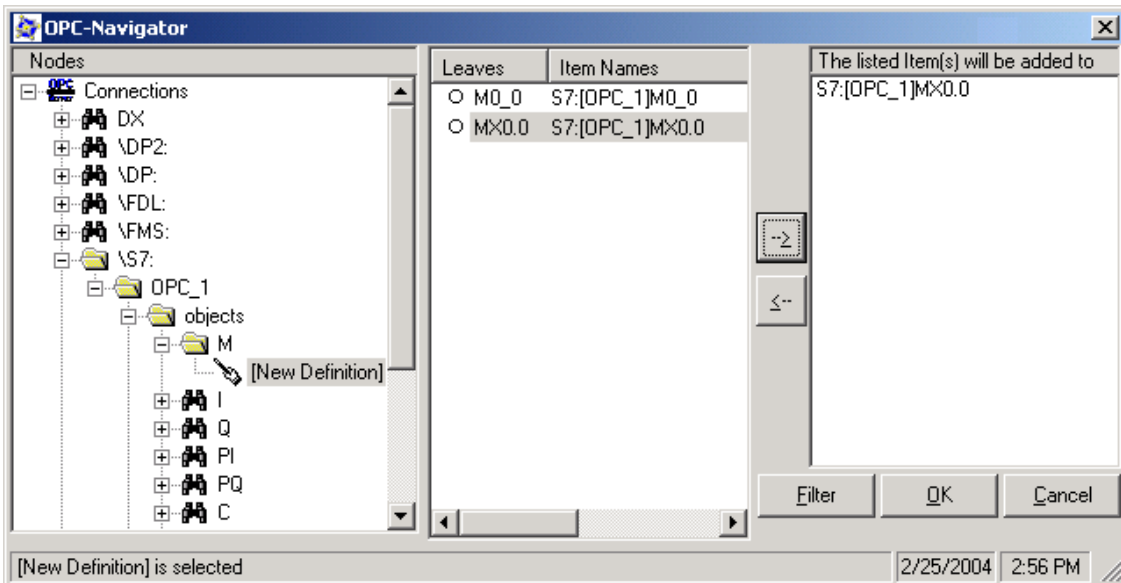
- Click OK to define an item for M0.0.



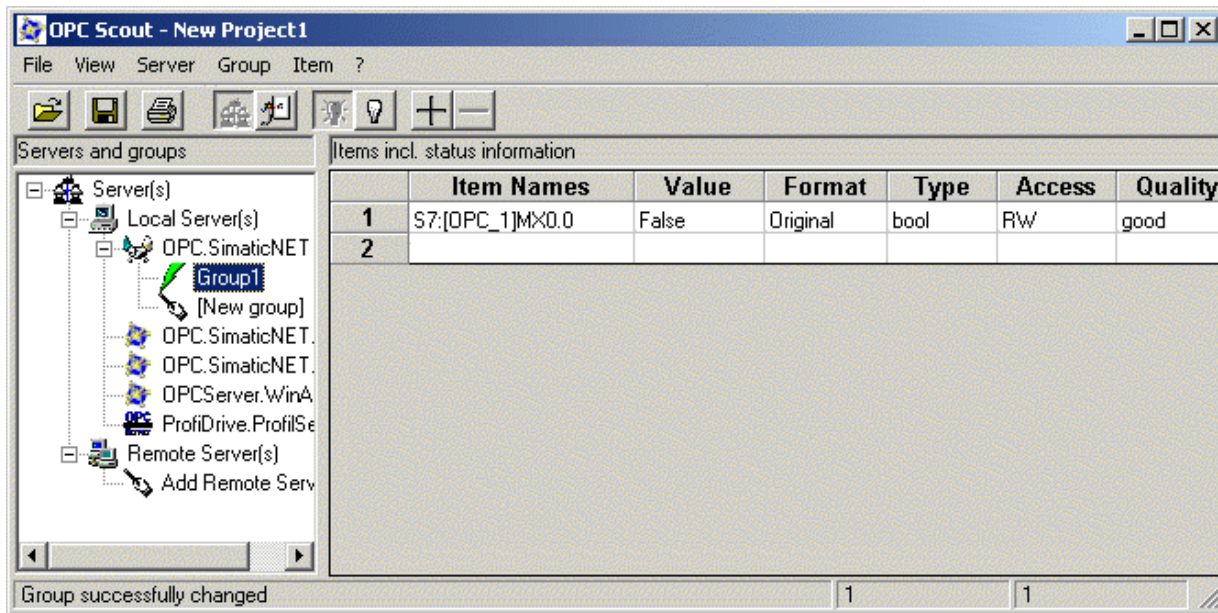
- Select the MX0.0 entry and click the Add arrow (-->) to enter the following syntax that defines a connection for MX0.0:

S7:[OPC\_1]MX0.0

- Select the entry (S7:[OPC\_1]MX0.0) and click OK to add the connection for MX0.0 to Group1.



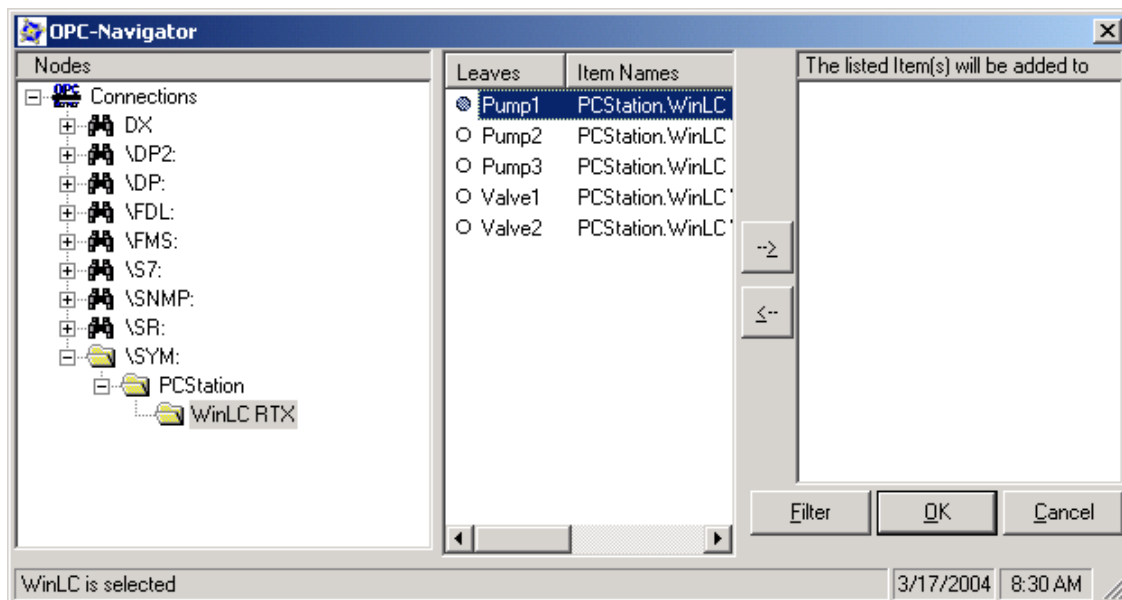
After adding the item to Group1, OPC Scout displays name and other parameters for the item. You can now use any of the methods supported by SIMATIC NET OPC server.



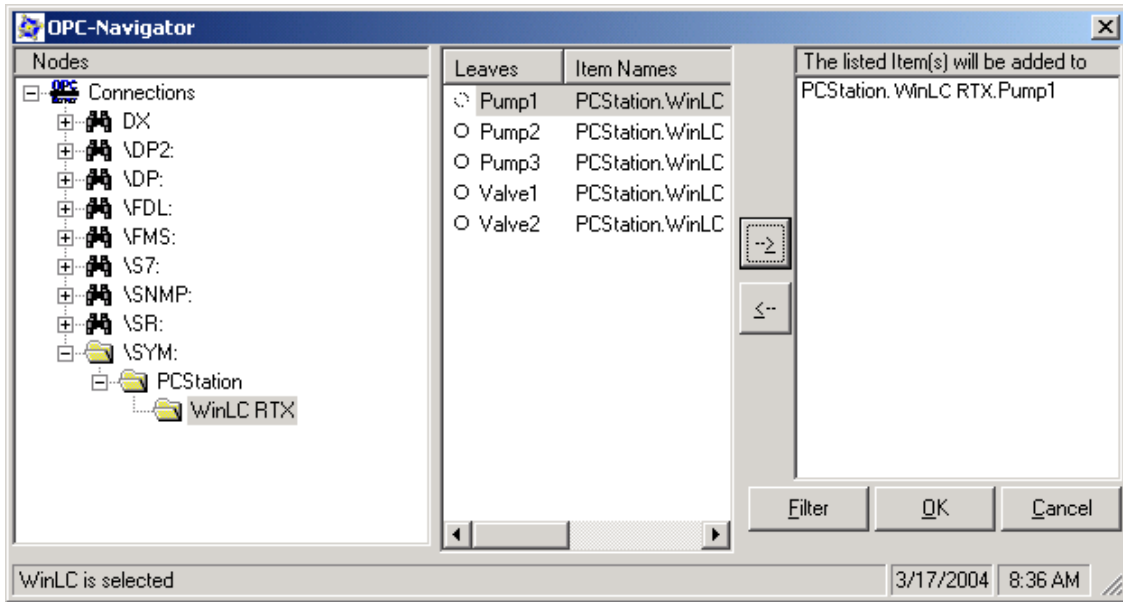
### Configuring the Items to be Accessed (Using the STEP 7 Symbol Table)

If you created a symbol table for the STEP 7 program that you downloaded, you can use the symbols for connecting the OPC server to the data in the controller. To configure the items to be accessed using the STEP 7 symbol table, follow these steps:

1. Open the OPC Navigator by double-clicking the connection (Group1) for the OPC server.
2. Browse to the folder for the controller to display the symbols that have been downloaded to the controller.

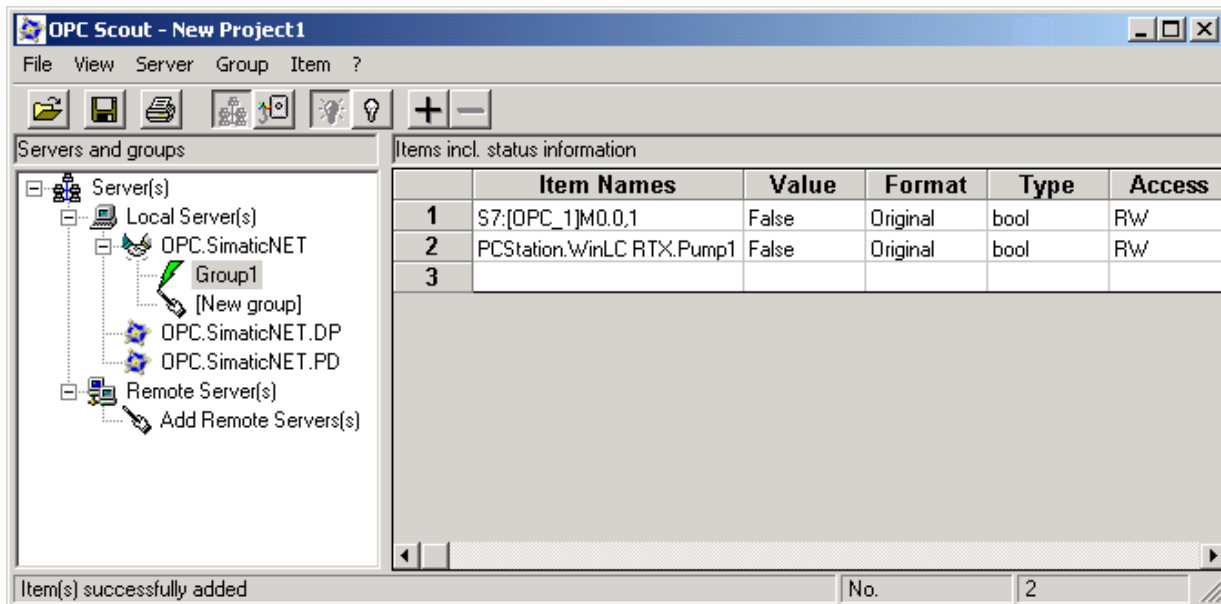


- After selecting the symbols for the data to be connected to the OPC server, click the Add button (-->).



- Click the OK button to add the symbol to Group1.

After adding the item to the group, OPC Scout displays symbol name and other parameters for the STEP 7 symbol.





# Reference Information

## Technical Data

### Order Number

WinLC RTX V4.2 is a component of the WinAC RTX 2005 package: 6ES7 671-0RC05-0YA0

WinLC RTX communicates with the distributed I/O as a PROFIBUS-DP master device. WinLC RTX does not support local I/O.

### Technical Specifications

The following table lists the technical information about WinLC RTX.

WinLC RTX	Description
Work memory Load memory	<p>Limited by the amount of non-paging memory configured in Windows. The following factors affect this amount:</p> <ul style="list-style-type: none"><li>• Amount of physical memory (RAM) installed in the computer</li><li>• Other Windows drivers and RTSS programs being executed at the same time as WinLC RTX</li><li>• Virtual memory configuration in Windows</li></ul> <p>To change the virtual memory paging configuration, follow these steps:</p> <ol style="list-style-type: none"><li>1. Select System from the Windows Control Panel.</li><li>2. From the Advanced tab of the System Properties dialog, click the Settings button for Performance.</li><li>3. From the Advanced tab of the Performance Options dialog, click the Change button for Virtual memory.</li><li>4. Make any changes you need, and click OK on the dialogs to complete your configuration.</li></ol>
Accumulators	4 (ACCU 1 to ACCU 4)
Local data	16 Kbytes per priority class (determined by HW Config, Memory tab)
Clock	<p>Real-time system clock, based on the hardware clock of the computer</p> <p>With the WinAC Time Sync feature, WinLC RTX can function as a time synchronization slave to the Time Synchronization service, or as a time slave to devices on submodule DP-subnets. WinLC RTX can also serve as a time master to devices on the submodule DP-subnets.</p>

WinLC RTX	Description
I/O (digital and analog)	16384 bytes total I/O, addressable over a range of 0 to 16383 You can freely assign the I/O between digital and analog inputs and outputs. For example, you can assign all of 16384 bytes to the inputs or all of the 16384 bytes to the outputs. However, the total amount allocated to all of the inputs and outputs cannot exceed the maximum of 16384 bytes.
Process image I/O (user configurable)	Inputs: 512 bytes (default) or configurable from 0 bytes to 8192 bytes (I 0.0 to I 8191.7) Outputs: 512 bytes (default) or configurable from 0 bytes to 8192 bytes (Q 0.0 to Q 8191.7)
Memory bytes	16 Kbytes
Retentive range (configurable)	Up to 16384 bytes (MB0 to MB16383)
Preset as retentive	16 bytes (MB0 to MB15)
Counters	512
Retentive range (configurable)	C0 to C511
Preset as retentive	8 (C0 to C7)
Timers	512
Retentive range (configurable)	T0 to T511
Preset as retentive	None
Clock memory	8 bits of clock memory (1 byte) 8 frequencies within 1 byte of bit memory (M): address is configurable
Address ranges for logic blocks (FB, FC, and DB):	FB0 to FB65535 FC0 to FC65535 DB1 to DB65535 (DB0 is reserved)
Number of S7 connections	CP 5611 and built-in PROFIBUS interfaces: 8 CP 5613: 50 Industrial Ethernet: 8 Total for all submodules: 64
Nesting depth	24 per OB in a priority class (sequence layer) At any one time, a priority class can have one OB and up to two synchronous OBs (OB 121 and OB 122). Each OB in the priority class can have a nesting depth of 24.



WinLC RTX	Description
Total number of blocks that can be downloaded to WinLC RTX	No fixed limit: The number of blocks that can be downloaded is based on the memory requirements and the number of blocks in the program
Number of TCP connections supported by IE card submodule	TCP: 8



### Caution

Downloading a STEP 7 user program that is too large for the memory of the computer can lock up the computer or cause the operation of WinLC RTX to become unstable, possibly causing damage to equipment and/or injury to personnel.

Although STEP 7 and WinLC RTX do not limit the number of blocks or the size of the STEP 7 user program, your computer does have a limit, based on the available disk space and RAM memory. The limit for the size of the STEP 7 user program and number of blocks for your computer can only be determined by testing a configured system against the requirements of your control application.

The following table lists specific information about the PROFIBUS-DP interface, as supported by WinLC RTX.

PROFIBUS-DP interface	Description
DP address area	16384 bytes (inputs) and 16384 bytes (outputs)
Number of DP slaves supported for each submodule DP interface	Dependent on the DP interface <ul style="list-style-type: none"> <li>• CP 5611 and built-in PROFIBUS interfaces: 64</li> <li>• CP 5613: 125</li> </ul>
Baud rate	Up to 12 Mbaud: 9.6 KBPS, 19.2 KBPS, 45.45 (31.25) KBaud, 93.75 KBPS, 187.5 KBPS, 500 KBPS, 1.5 MBPS, 3 MBPS, 6 MBPS, 12 MBPS
Baud rate search (as a DP slave)	Not applicable
Transfer memory (as a DP slave)	Not applicable
Maximum distance	Dependent on the baud rate

## Execution Times

Execution times for instructions in your user program vary depending on how the code is designed and implemented. The execution times listed below reflect typical execution times for STEP 7 user programs running on WinLC RTX.

Math Operation	Average Time
Bit operations	0.004 $\mu$ s
Integer Math	0.003 $\mu$ s
Floating-point math	0.004 $\mu$ s

## Troubleshooting

### Relevant Information for Ardence RTX

The RTX real-time extensions provide the determinism and performance of a real-time operating system within the Windows 2000 or Windows XP environment. However, not all computer configurations (hardware and software) support the installation and operation of Ardence RTX. When testing the operation of Ardence RTX and WinLC RTX on your computer, check the following items:

- Ardence RTX installs and runs. Make certain that you have administrator (ADMIN) privileges for the computer. Make certain that your computer meets the hardware and software requirements as described in the RTX Runtime Release Notes and that the installed Hardware Abstraction Layer (HAL) is one that RTX supports.
- Ardence RTX allows a free interrupt in order to operate a DP interface in interrupt mode (varies for different computer manufacturers). If a free interrupt is not available, the DP interface operates only in polled mode and not in interrupt mode.
- Ardence RTX is able to operate without interference from hardware components installed in the computer. Some components (such as the video card) can cause problems that affect the performance of real-time control with Ardence RTX.

### Setting the HAL Timer Period

The HAL timer period sets a number of microseconds as the basis for RTX timers. The default value is 500 microseconds. WinLC RTX uses the RTX timers for starting certain OBs, for SFC 47 (WAIT), and for other internal events. Changing the HAL timer period may provide more deterministic behavior for some applications that require accuracy of less than 1 millisecond. However, decreasing the HAL timer period also increases the CPU load, with no benefit for most applications.

#### Notice

Changing the HAL timer period to a value lower than the default value can increase the load on the CPU of your computer. This increased CPU usage could affect the operation of your application.

If you change the HAL timer period, always test your application to ensure that the increased CPU load does not adversely affect the operation of WinLC RTX.

To change the value for the HAL timer period, follow these steps:

1. Use the **Start** menu to open the Windows Control Panel.
2. Double-click the RTX Properties icon to display the RTX Properties dialog.
3. Click the Setting tab to display the parameters for the HAL timer.
4. Adjust the value for the HAL timer period (in microseconds) and click OK.

### Running the DP Interface in Interrupt Mode

On some computers, Ardence RTX allows a free interrupt for the DP interface. (This varies for different computer manufacturers.) If a free interrupt is not available, CP cards (including built-in PROFIBUS adapters on Siemens PCs) operate only in polled mode and not in interrupt mode, which can affect the performance of the CP card. An Industrial Ethernet interface cannot function without a free interrupt.

Refer to the topic on improving the performance of a DP interface.

## Using the RTX Platform Evaluator to Check Performance

On some computers, some components of the computer (such as a video card) can cause problems with Ardence RTX that affect the performance of real-time control.

For a uniprocessor system (single processor), you can use the RTX Platform Evaluation utility to determine if your computer has any hardware installed (such as a video card) that may introduce jitter or latencies.

The RTX Platform Evaluator is not included with WinAC RTX. Contact Ardence to obtain the RTX Platform Evaluator and information about how to install and use it.

## Changing the HAL Type for the Computer

### Caution

Changing the HAL type can create a situation where the computer cannot be booted. You must then recover by using an Emergency Repair disk.

Changing the HAL type changes the entry in the Windows registry. Errors in the registry can keep the computer from rebooting.

Before you make any changes to the Windows registry (such as changing the HAL type), always create an Emergency Repair disk. Select the **Start > Programs > Accessories > System Tools > Backup** menu command to create an Emergency Repair disk.

Refer to the Product Information document provided with your installation for specific directions regarding changing the HAL type.

## Troubleshooting Network Problems

The controller panel provides the EXTF and BUSF status indicators that can be used to diagnose problems with the PROFIBUS-DP network. The table below describes the activity of the EXTF and BUSF indicators to help you determine the type of problem and a possible solution.

EXTF	BUSF	Description	Action
Off	Off	No configuration	Ensure that the DP configuration has been entered into your STEP 7 project. Download the project's System Data container to the controller.
		Normal operation	The configured DP slaves are responding. No action is required.
On	Flashing	Station failure	Check to see that the bus cable is connected to WinLC RTX (the CP card) and that all segments are correctly terminated at powered nodes. Check to see that the bus is not interrupted.
		At least one of the DP slaves could not be accessed	Wait for completion of the power-on cycle. If the indicator continues to flash, check the DP slaves or evaluate the diagnostic data for the DP slaves.
—	On	Bus fault (hardware failure)	Check the bus cable for an electrical short, or a broken wire or connection.
On	Off	Diagnostic error	Indicates that a fault condition has not been cleared or that a DP module with diagnostic capability has initiated OB 82.

In addition to these visual indicators, you can use the Diagnose Hardware feature of the STEP 7 programming software to determine which nodes are experiencing problems and to determine the nature of the problem.

## Improving the Performance of a DP Interface

To use a DP interface in isochronous mode, the DP interface must operate in interrupt mode.

**Note:** WinLC RTX allows you to use isochronous mode on more than one PROFIBUS-DP subnet; however, your computer must not share the interrupt (IRQ) of the PCI slots used by the DP interfaces with any device operating in the Windows operating system (for example, a video card). For example, the SIMATIC Box PC 627 provides two PCI slots that can be used for isochronous mode on two different PROFIBUS-DP subnets.

**Tool:** You use the Windows Device Manager.

WinLC RTX accesses communications interfaces in either interrupt mode or polled mode. Interrupt mode provides improved performance over polled mode.

In order for WinLC RTX to use interrupt mode for accessing a communication interface, you must configure your computer so that the communication interface does not share an IRQ (interrupt request) with a Windows-controlled device.

Use the following procedure to determine whether the IRQ assignment for a communication interface is shared with an IRQ assignment for a Windows-controlled device:

1. Right-click the My Computer icon and select the Manage menu command.
2. Click Device Manager, and then select the **View > Resources by Type** menu command.
3. Expand the Interrupt request (IRQ) folder. The numerical values shown beside each entry indicate the IRQ assignment.
4. Locate the entry for the communication card in the device list. If the IRQ assigned to this entry is assigned to any other device, the card is sharing an interrupt with that device. If this other device is Windows-controlled, the communication card will be operated in polled mode if it is configured as a submodule of WinLC RTX. Otherwise, the communication card operates in interrupt mode.

To determine whether a device is Windows-controlled (as opposed to being RTX-controlled), use the following procedure:

1. Right-click the device entry for the communication card in the Device Manager list, and select Properties.
2. Select the General tab on the Properties dialog, and check the Device type value. If it displays "RTX Drivers", the device is RTX-controlled. Otherwise, it is Windows-controlled.

If the communication card shares the IRQ number with a Windows-controlled device, use one of the following methods to change the system configuration for your computer and to assign a different IRQ number to the communication interface:

- Use the BIOS setup utility for your computer to manipulate IRQ assignments and remove IRQ conflicts.
- Install the communication card in a different PCI expansion slot of your computer. Because the PCI slots are often assigned different IRQ numbers, installing the card in a different slot might eliminate the conflict; however, changing the slot can also result in a new conflict.
- If the IRQ conflict is due to a built-in device (for example, an Ethernet or SCSI controller), consider using the BIOS setup utility to disable the conflicting built-in device, if possible. In this case, you might have to use an equivalent expansion card to replace the functionality for the disabled device.

Using these methods can be an iterative process, and you might not find a solution that assigns a suitable IRQ number to the communication interface. If no configuration can be found that eliminates the IRQ conflict, you must either select a different PC platform or you must use the polled mode of operation for the communication interface.

For multiple cards, repeat this process as necessary to resolve all interrupt conflicts.

## Responding to Diagnostic Events

If an error is detected by the controller, the error condition is logged in the diagnostic buffer as a diagnostic event. The diagnostic events that are typically associated with distributed I/O can cause the controller to execute the following OBs:

- OB 40 responds to hardware interrupts (process alarms) generated by an I/O module with configured interrupt capability.
- OB 82 responds to diagnostic interrupts generated by an I/O module with configured diagnostic interrupt capability.
- OB 83 responds to module removal/insertion at a DP Slave, (for example, ET200M), which has been configured for module pull/plug support.
- OB 85 responds to a priority class error. There are multiple causes for OB 85 relating to the DP I/O system. If the controller attempts to copy a module's inputs to (or outputs from) the process image during the I/O cycle, and the module is not operational, then an OB 85 is executed.
- OB 86 responds to a station failure or some other interruption of the physical network (such as a short circuit).
- OB 122 responds to an I/O access error by the user program. If OB 122 is not programmed, the controller goes to STOP mode.

You can use SFC 39 to SFC 42 to disable, delay, or re-enable any of these OBs. If an OB is requested and the OB has not been downloaded to the controller, the controller goes to STOP mode.

The local variables for these OBs contain startup information indicating the cause for executing the OB. The program for the OB can use this information for responding to the event. You can also use SFC 13 (DPNRM\_DG) to read the diagnostic information from a DP slave.

For information about using OBs and SFC 13, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

## Cross-Module Access Errors

Unlike hardware PLCs, PC-based controllers do not allow a Load (L) or Transfer (T) instruction to access bytes of more than one module. Consider a configuration of two output modules, each containing five bytes. Module 1 is addressed from 10 to 14, and Module 2 is addressed from 15 to 19. OB 1 contains the instructions shown below:

```
L 5
T PAW 14
```

In this example, OB 122 is called because of an attempt to access bytes across a module boundary. A word instruction at address 14 attempts to access address 14 and 15, which is prevented because the addresses are not in the same module.

## System Status List (SSL)

### Using SFC 51 to Read the SSL

STEP 7 stores read-only information about the controller in the system status list (SSL) as a set of sublists.

You use SFC 51 (RDSYSST) to access the entries in the SSL. You supply the input parameters SSL\_ID and Index to access the records stored in the sublist. SFC 51 returns a two-word header and a sublist or partial sublist. The header provides the following information about the sublist:

- The first word defines the length (size in bytes) of a record for the sublist.
- The second word defines the number of records contained in the sublist.

The requested information follows the header. The size of the sublist in bytes is the record length times the number of records.

**Note:** The SSL\_ID and Index values are represented as hexadecimal (16#) numbers.

For more information about the system status list, see the online help for STEP 7 or the *System Software for S7-300/400 System and Standard Functions Reference Manual*. To view this manual from a computer where STEP 7 is installed, select the **Start > Simatic > Documentation > English** menu command and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

WinLC RTX supports the following SSL entries. Some are available only when WinLC RTX has at least one DP interface configured as a submodule:

<b>Module Identification:</b> 0111	<b>Communications Status:</b> 0132, 0232
<b>CPU Characteristics:</b> 0012, 0112, 0F12	<b>LED Status:</b> 0174
<b>Memory Areas:</b> 0113	<b>DP Master System:</b> 0090, 0190, 0F90
<b>System Areas:</b> 0014, 0F14	<b>Module Status:</b> 0591, 0991, 0C91, 0D91, 0E91
<b>Block Types:</b> 0015	<b>Rack and Station Status:</b> 0092, 0192, 0292, 0692
<b>Local Module LED Status:</b> 0019, 0F19	<b>Expanded DP Master:</b> 0195, 0F95
<b>Component Identification:</b> 001C, 011C, 0F1C	<b>Diagnostic Buffer:</b> 00A0, 01A0, 0FA0
<b>Interrupt Status:</b> 0222	<b>Module Diagnostics:</b> 00B1, 00B3, 00B4
<b>Process Image Partitions:</b> 0025, 0125, 0225, 0F25	



## SSL\_ID Descriptions

### SSL\_ID 0x11 (Module Identification)

0111 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0111	Specific information for a module	0001: Order number, module type, and version 0007: Firmware version

### SSL\_ID 0x12 (CPU Characteristics)

0012, 0112, 0F12 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0012	All characteristics for a module	MC7 processing unit, time system, system response, and MC7 language description
0112	One specific group of characteristics	0000: MC7 processing unit 0100: Time system 0200: System response 0300: MC7 language description
0F12	Header information only	

### SSL\_ID 0x13 (Memory Areas)

0113 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0113	Specific memory area	0001: User memory 0002: Load memory integrated 0003: Load memory inserted 0004: Maximum insertable Load memory 0005: Backup memory 0006: Peer-to-peer memory (shadow memory)

**SSL\_ID 0x14 (System Areas)**

0014, 0F14 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0014	All system memory areas for a module	Size and other parameters for each area of system memory
0F14	Header information only	

**SSL\_ID 0x15 (Block Types)**

0015 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0015	All block types for a module	Maximum number and size for each type of block

**SSL\_ID 0x19 (Local Module LED Status)**

0019, 0F19 (hexadecimal)

**Note:** SSL\_ID 0x19 supports local, non-redundant CPUs. You can use SSL\_ID 0x19 with a redundant H CPU only when the H CPU is in a non-redundant operating mode. Use SSL\_ID 0x74 to access information for a redundant H CPU.

SSL_ID	Sublist	Index and Contents of the Record
0019	All of the LEDs for the local module	Status for all of the LEDs
0F19	Header information only	

**SSL\_ID 0x1C (Component Identification)**

001C, 011C, 0F1C (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
001C	All of the information for a component	Controller name, module name, module tag, copyright, serial number, project ID, module type, and manufacturer information
011C	Specific element for the component	0001: Name of the controller 0002: Name of the module 0003: Module tag 0004: Copyright entry 0005: Serial number 0007: Module type 0009: Manufacturer and profile identification 000B: Location designation (OKZ) of a module
0F1C	Header information only	

**SSL\_ID 0x22 (Interrupt Status)**

0222 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0222	Start event for a specific OB	OB number: Start event and time for the requested OB

**Note:** For a list of the OBs supported by WinLC RTX, refer to the following topics: Logic Blocks Supported by WinLC and Organization Blocks (OBs).

**SSL\_ID 0x25 (Process Image Partitions)**

0025, 0125, 0225, 0F25 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0025	All process image partitions	Process image partitions for all of the OBs that have been downloaded to the module
0125	Process image partition for a specific OB	Partition number: OB configured for that partition
0225	OBs assigned for a specific process image partition	OB number: Partition assigned for that OB
0F25	Header information only	

**SSL\_ID 0x32 (Communications Status)**

0132, 0232 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0132	Specific set of parameters	0001: Number and type of connections 0002: Connections configured 0003: Operator interface 0004: Protection level and mode switch selection 0005: Diagnostics 0006: Peer-to-peer status data 0008: Time system 000A: Baud rate
0232	Parameters for a redundant system (H CPU)	0004: Protection level and mode switch selection

**SSL\_ID 0x74 (LED Status)**

0174 (hexadecimal)

**Note:** Use SSL\_ID 0x74 to access information about LEDs for any module, including a redundant H CPU module. See also SSL\_ID 0x19.

SSL_ID	Sublist	Index and Contents of the Record
0174	Specific LED	0002: INTF (Internal failure) 0003: EXTF (External failure) 0004: RUN (Run) 0005: STOP (Stop) 0006: FRCE (Force) 0008: BATF (Battery failure) 000B: BUSF1 (submodule 1 fault) 000C: BUSF2 (submodule 2 fault) 0012: BUSF3 (submodule 3 fault) 0013: BUSF4 (submodule 4 fault)

**SSL\_ID 0x90 DP Master System**

0090, 0190, 0F90 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0090	All DP masters configured on the network and downloaded to the module	DP master identifier, address, and attributes for all DP masters
0190	Specific DP master	DP master identifier: Address and attributes
0F90	Header information only	

**SSL\_ID 0x91 (Module Status)**

0591, 0991, 0C91, 0D91, 0E91 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0591	Module status information of all submodules of the host module	Irrelevant
0991	Module status information of all submodules of the host module in the rack specified	Rack or DP master system ID
0C91	Specific module, identified by the logical base address	Logical base address: Features and parameters of the specified module
0D91	Specific station, identified either by rack/station, by DP master identifier, or by DP master identifier with station number	Station identifier: Features and parameters for all the modules of the specified station
0E91	Module status information of all assigned modules	Irrelevant

**SSL\_ID 0x92 (Rack and Station Status)**

0092, 0192, 0292, 0692 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0092	Expected status of the stations of a DP master	0: Local DP master DP master identifier: Specific DP master
0192	Configuration and activation status for the stations of a DP master	0: Local DP master DP master identifier: Specific DP master
0292	Actual status for the stations of a DP master	0: Local DP master DP master identifier: Specific DP master
0692	OK state for the stations of a DP master	0: Local DP master DP master identifier: Specific DP master

**SSL\_ID 0x95 (Expanded DP Master System)**

0195, 0F95 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0195	Specific DP master	DP master identifier: Properties for the stations of the specified DP master (such as DP mode, equidistant mode and cycle, clock synchronization, and transmission rate)
0F95	Header information only	

**SSL\_ID 0xA0 (Diagnostic Buffer)**

00A0, 01A0, 0FA0 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
00A0	All of the entries in the diagnostics buffer	Event information for every event listed in the diagnostics buffer
01A0	Most recent entries in the diagnostics buffer	Number: Event information for the specified number of entries in the diagnostics buffer
0FA0	Header information only	

**SSL\_ID 00B1, 00B3, and 00B4 (Module Diagnostics)**

00B1, 00B2, 00B4 (hexadecimal)

**Note:** The information varies according to the type of module specified.

SSL_ID	Sublist	Index and Contents of the Record
00B1	Diagnostic information (4 bytes) for a specific module, identified by the logical base address	Logical base address: First 4 bytes of the diagnostic information
00B3	All of the diagnostic information for a specific module, identified by the logical base address	Logical base address: Complete diagnostic information
00B4	Specific DP slave, identified by the configured diagnostic address	Diagnostic address: Standard diagnostic information for a DP station





# Glossary

## B

**Backplane bus:** For hardware controllers such as the S7-300 or S7-400, the backplane bus is the printed circuit board on the inside panel of the rack into which modules are inserted. (See topic "What Is a PC Station?")

**Blue Screen:** Termination of the Windows operating system, resulting in a display on the monitor of the fatal error on a blue background. A blue screen is also known as a Windows Stop Error.

## C

**Cold Restart:** The controller executes OB 102 before starting the free cycle (OB 1). Like a warm restart, a cold restart resets the peripheral inputs (PI) and changes the peripheral outputs (PQ) to a pre-defined safe state (default is 0). However, a cold restart does not save the retentive memory (M, T, C, or DB), but sets these areas to their default (initial) values.

**Communications Interface:** CP cards, Siemens PC built-in PROFIBUS interface, or Industrial Ethernet interface that WinLC RTX uses for communications.

**CP Card:** Communications Processor card: for WinLC RTX either a CP 5611 (including built-in PROFIBUS interface on Siemens PCs), a CP 5613, or an Industrial Ethernet interface (built-in or installed card).

## D

**Deterministic behavior:** Predictability of execution time and response time

**DP interface:** A Siemens CP card or Siemens PC built-in PROFIBUS interface used for PROFIBUS-DP communications. (See topic "What Is a Communication Interface?")

## E

**Execution Load:** The percentage of CPU time used by the controller.

**Execution Monitor:** The execution monitor of the controller measures the time that the controller sleeps and ensures that the controller does not exceed the maximum execution load. The execution monitor uses the maximum execution load and the execution time limit to calculate the forced execution sleep time.

**Execution Time:** The execution time is the actual time the controller takes to complete one pass through the instructions of the STEP 7 user program. This includes executing OB 1 and updating the I/O.

**Execution Time Limit:** Defined maximum amount of time allowed for the controller to execute the STEP 7 user program. The execution monitor uses this value and the maximum execution load to calculate the forced execution sleep time.

## F

**Forced Execution Sleep Time:** This read-only field shows how much sleep time (in microseconds) is required during the monitor interval to meet the maximum execution load requirement.

**Free Cycle:** The free cycle consists of the basic tasks for priority class 1: writing to the outputs, reading the inputs, executing OB 1, and completing the sleep time requirement before triggering the next free cycle. The controller executes these tasks at the base, or lowest, internal priority level for executing the OBs. (Priority level in this context refers to OB priority classes, not the operating system priority level.)

## H

**HAL:** Hardware Abstraction Layer. (See your Product Information for the relationship between HAL configuration and WinLC RTX performance.)

## I

**IF Slot:** Interface Slot. One of four slots allocated for communication interfaces configured as submodules of the controller. (See topic "What Is an IF Slot?")

**Index:** A numbered slot in the PC Station, or virtual rack that represents a PC-based automation system. The controller occupies one index; other components can occupy other index slots. (See topic "What is an Index?")

**Industrial Ethernet:** Physical communications layer that supports communication to STEP 7, S7 CPUs, PGs, OPs, and S7 applications

**Isochronous Mode:** Configuration of DP cycle that yields a constant bus cycle time. (See topic "Isochronous Mode for a Constant Bus Cycle".)

## J

**Jitter:** Difference in the actual scan cycle time from the configured minimum scan time.

## L

**Load memory:** Memory area (RAM) allocated for all of the blocks downloaded from STEP 7 excluding the symbol table and comments

## M

**Maximum Execution Load:** Maximum percentage of CPU usage that is allocated for the controller. The execution monitor uses this value and the execution time limit to calculate the forced execution sleep time.

**Minimum Cycle Time:** Minimum number of milliseconds from the start of one cycle to the start of the next cycle. You enter a value for the minimum cycle time when you use STEP 7 to configure the system data for the controller. You can use the tuning panel to adjust this value as you test the performance of the controller. After you have tuned the performance of the controller, use STEP 7 to enter the optimum cycle time value and download the new system data. Any value for the cycle time that you enter with the tuning panel is overwritten by the value in the system data when the controller changes from STOP mode to RUN mode.

**Minimum Sleep Time:** Specific amount of time that the controller must wait before starting the next scan cycle. You use the tuning panel to configure this parameter. The controller uses the minimum sleep time and the minimum cycle time parameters to calculate the start of the next scan cycle.

**Monitor Interval:** Length of time used by the execution monitor in determining whether to add a forced sleep time. The monitor interval is the sum of the execution time limit and the forced execution sleep time that is calculated based on the maximum execution load percentage.

**MPI:** Multi-point interface: physical communications layer that can be used for S7 communications to STEP 7, S7 CPUs, and S7 applications

## N

**Non-deterministic behavior:** Lack of predictability of execution time and response time associated with "jitter." (See topic "What Causes Jitter?")

---

## O

**OP:** Operator panel

**Organization Block (OB):** Interface between the operating system and the STEP 7 user program. Called by the operating system, they control cyclic and interrupt-driven program execution, startup behavior of the controller and error handling.

## P

**PC Station:** Representation of a software-based virtual rack that defines a PC-based automation system. (See topic "What is a PC Station?")

**PG:** Programming device

**PG/OP communication:** Communication between WinLC RTX and other S7 applications such as programming devices, operator panels, and S7 controllers. WinLC RTX supports PROFIBUS and Industrial Ethernet for PG/OP communication.

**Priority:** The priority of an application determines the order in which the operating system executes or interrupts an application in relation to the other applications that are running on the computer. An application with a higher priority interrupts and suspends the execution of an application with a lower priority. After the application with the higher priority finishes, the application with the lower priority resumes. A higher number indicates a higher priority.

**Priority Class:** The priority class determines the order in which the controller executes the individual sections of the STEP 7 user program. Organization blocks (OBs) are ranked by priority class. Higher priority OBs interrupt lower priority OBs. The free cycle (OB 1) has the lowest priority. You can use STEP 7 to change the priority class for an OB. A higher number indicates a higher priority class.

**PROFIBUS:** Physical communications layer that can be used for PROFIBUS-DP communications to I/O or S7 communications to STEP 7, S7 CPUs, and S7 applications.

**PROFIBUS-DP:** Communications network protocol used to communicate to DP I/O

## R

**Restart Method:** The restart method determines which startup OB is executed whenever the controller changes from STOP mode to RUN mode. The startup OB allows you to initialize your STEP 7 user program and variables. The two restart methods are Cold Restart (OB 102) and Warm Restart (OB 100).

**RTX:** Real-time extensions: Ardence real-time extensions to the Windows Operating system extensions that allow processes to run in a real-time environment providing more deterministic execution and protection from Windows operating system crashes.

## S

**S7 communication:** Communication between controllers on the network, hardware or software, using the S7 communication functions. (See topic "S7 Communication Functions.")

**S7 routing:** Communications between S7 controllers, S7 applications or PC Stations across different subnets through one or more network nodes acting as routers, configured with NetPro

**Scan Cycle:** The scan cycle includes writing to the outputs, reading the inputs, executing OB 1 and all other OBs, and completing the sleep time requirement.

**Scan Cycle Time:** Time required to execute the complete scan cycle, which includes the execution of OB1 and the minimum sleep time.

**Sleep Time:** Difference between the execution time of the free cycle and the total scan time. Sleep time measures the time between the completion of OB1 and the start of the next scan cycle, and ensures that the next scan cycle does not start until the end of the sleep interval. However, if the start event for an interrupt OB (such as OB40) occurs during the sleep time, the controller executes that interrupt OB.

**Station Configuration Editor:** Tool, accessible from taskbar, for configuring the PC Station: for WinLC RTX this includes WinLC Properties, submodule assignments, and submodule diagnostics for some DP interfaces.

**STEP 7 User Program:** Application program created with STEP 7 and downloaded to the controller for execution. It includes all organization blocks (such as OB 1 or OB 35) and the other logic blocks that they call, including functions (FCs), system functions (SFCs), function blocks (FBs), and system function blocks (SFBs).

**Submodule:** Communication interface in the PC that is designated for exclusive use by WinLC RTX. (See topic "What Is a Submodule?")

**System Function (SFC):** Preprogrammed function that is integrated as a part of the operating system of the controller and is not downloaded as part of the STEP 7 user program. You can call an SFC in your STEP 7 user program. Like a function (FC), an SFC is a block "without memory."

**System Function Block (SFB):** Function block that is integrated as a part of the operating system of the controller and is not downloaded as part of the STEP 7 user program. Like a function block (FB), an SFB is a block "with memory." You must also create an instance data block (DB) for the SFB. The instance DB is then downloaded to controller as part of the STEP 7 user program.

## T

**Time Synchronization:** The ability to broadcast a system standard time from a single source to all devices within the system so that they may set their own clocks to the standard time.

**Time Synchronization Service:** Software component of WinAC RTX that provides the capability to synchronize time between components in the PC Station. (See the documentation for the WinAC Time Synchronization Service.)

## V

**Virtual backplane bus:** For PC-based controllers, the virtual backplane bus is a software-based, virtual "rack" that enables communications between the controller and other PC Station components. (See topic "What Is a PC Station?")

## W

**Wait Time:** The wait time, or sleep time, is the time that the controller is not using the CPU. During this time, the operating system can run other applications.

**Warm Restart:** Type of restart where the controller executes OB 100 before starting the free cycle (OB 1). A warm restart resets the peripheral inputs (PI) and changes the peripheral outputs (PQ) to a pre-defined safe state (default is 0). The warm restart also saves the current value for the retentive memory areas for the memory bits (M), timers (T), counters (C), and data blocks (DBs).

**Windows Stop Error:** Termination of the Windows operating system, resulting in a display on the monitor of the fatal error on a blue background. A Windows Stop Error is also known as a blue screen.

**Work memory:** Memory area (RAM) allocated for the blocks used at runtime

# Index

## A

Absolute addressing, OPC server, 119

Access points, 30

Access verification, 58

Accumulators, 127

Add/Remove Programs, 15

Adding blocks to Load memory, SFC 82 and SFC 84, 56

Adding S7 connection for OPC Server in NetPro, 115

Adding sleep time, 96, 97, 103

Address ranges, logic blocks, 127

Addressing errors, 135

Adjusting, 79

- minimum scan cycle time, 93
- minimum sleep time, 93
- priority, 81, 88
- sleep-monitoring algorithm, 93, 97
- system clock, 79

Administrator user privileges, 7

Africa, customer support, iii

Alarms, 135

All indicators flashing, 43

ALT+C+M, resetting memory, 42

Analog I/O, 127

Archiving, 49

Ardence RTX, 12

- advantages, 3
- HAL (hardware abstraction layer), 131
- installation, 12
- requirements, 6

Asia, customer support, iii

Asynchronous SFC considerations, 56

Asynchronous threads, 88

Automatic reboot for Windows, 51

Automation Licence Manager, 14, 15

Autostart

configuring, 57

effect of setting on startup, 54

Avoiding jitter, 96, 97, 103

## B

Backplane bus, 18

BATF status indicator, 43

Battery fault, 43

Baud rate, 127

Blocks, 64

- creating, reading, and writing, 56
- new, 5
- OBS, 68
- SFBs, 78
- SFCs, 73

Blue Screen (unrecoverable fault in Windows), 3, 10, 50, 51, 52, 54, 68

BRCV (SFB 13), 65

BSEND (SFB 12), 65

Bus cycle time, 107

BUSF status indicators, 43, 133

## C

C\_DIAG, 65

Change Password dialog, 59

Changing

- HAL type or timer, 131
- mode selector switch, 40
- operating mode, 40
- password, 59
- priority, 88
- sleep-monitoring algorithm, 97, 103

Characters (invalid in controller name), 35

Clock, 79

Clock memory, 127

Closing the control panel, 39

Cold restart, 68

- configuring, 61

- effect on startup, 54
- Commands
  - Diagnostic buffer, 47
  - Exit, 50
  - MRES (memory reset), 42
  - Options
    - Security, 58
  - Tuning panel, 45
- Communication
  - comparison to S7-400, 18
  - configuring, overview, 17
  - DPV1 extensions, 66
  - getting started overview, 17
  - PC Station vs. submodules, 18
  - PC-based control, 1
  - S7 communication functions, 65
  - STEP 7 and controller, 30
  - TCP/IP communication blocks, 67
  - with DP I/O, 22
- Communication interfaces
  - configuration options, 22
  - configuring as submodules, 25
  - definition, 20
  - IF slots, 24
  - supported, 20
- Comparison, S7-400 to PC-based controller, 18
- Computer requirements, 6
- Configuring
  - communication between STEP 7 and controller, 30
  - communication interface as submodule, 25
  - controller communications, overview, 17
  - INI file for Industrial Ethernet interface, 27
  - language for controller panel and help, 57
  - Local ID for OPC server connection, 115
  - OPC server, 112
    - connection, 119
    - hardware configuration, 112
    - in Station Configuration Editor, 110
    - items to be accessed, 119
    - overview, 109
    - operational parameters, 63
    - project in STEP 7, 32
    - S7 connection for OPC server in NetPro, 115
    - submodules, 25
    - verification, 37
- Confirmation, mode changes, 58
- Connecting
  - controller to the OPC server, 109
  - STEP 7 to controller, 30
- Connection
  - adding for OPC server with OPC Scout, 119
  - configuring for OPC server with NetPro, 115
- Constant bus cycle time, 107
- Contact information, iii
- Context-sensitive help, 8
- CONTROL (SFC 62), 65
- Controller memory reset, 42
- Controller name, invalid characters, 35
- Controller panel
  - introduction, 2
  - opening and closing, 39
  - status indicators, 43
- Controller state, saving, 52
- Copyright information**, 2
- Counters supported, 127
- CP 5611
  - available functionality, 5
  - communications supported, 20
  - improving performance, 134
  - operating in interrupt mode, 134
  - submodule limit, 24
- CP 5613
  - communications supported, 20
  - improving performance, 134
  - operating in interrupt mode, 134

- testing configuration, 28
- CP Cards
  - configuration options, 22
  - configuring as submodule, 25
  - configuring in STEP 7, 32
- CPU hardware fault, 68
- CPU indicators, 43
- CPU menu
  - diagnostic buffer, 47
  - MRES (memory reset), 42
  - options
    - security, 58
  - tuning panel, 45
- CPU usage, 45
  - jitter, 84
- Crash operations (OB 84), 50
- Creating
  - archive file, 49
  - password, 58
- Creating data blocks, 56
- Cross-module access errors, 135
- Customer service, iii
- Cycle time, 45, 81, 93, 97, 103
- Cycle/clock memory, 45
- Cyclic input/output update, 107
- Cyclic interrupt, 68
- Cyclic OBs
  - calling SFC 47, 96
- D**
- Dashes (in controller name), 35
- Data blocks
  - creating, reading, and writing, 56
- Data retention, 54
- Data set read and write SFBs, 66
- DBs, 64
- Default sleep monitor parameters, 97
- Defective state, 43
- Deleting
  - installed software, 11
  - submodules, 29
  - WinAC RTX software, 15
- Deterministic scan cycle, 88
- Device Manager, 134
- Diagnosing hardware, STEP 7, 133
- Diagnostic alarm interrupts, 68
- Diagnostic buffer, 47, 135
  - saving the contents, 52
- Diagnostic events, 47, 135
- Differences between submodules and PC station communication interfaces, 18
- Differences between WinLC RTX and WinLC Basis, 10
- Digital I/O, 127
- Disk space, 6
- Display language, 57
- Distributed I/O
  - access error, 68, 135
  - available, 127
  - communication, 22
  - configuration, 32
  - overview, 1
  - supported communication interfaces, 20
  - technical data, 127
- Documentation, iii
- Downloading OPC server configuration, 118
- DP address area, 127
- DP bus cycle, 107
- DP I/O, 22
  - access error, 68, 135
  - communication, 22
  - configuration, 32
  - overview, 1
  - supported communication interfaces, 20
  - technical data, 127
- DP interface, 22
  - configuration options, 22
  - configuring as a submodule, 25

- definition, 20
- improving performance, 134
- operating in interrupt mode, 134
- DP Master, isochronous mode, 107
- DP network, 133
  - subnet configuration, STEP 7, 32
  - troubleshooting, 133
- DP slave failure, 68
- DP slaves supported, 127
- DPV1 event OBs, 66
- DPV1 extensions, 66

## E

- Eliminating forced sleep interval, 103
- E-mail addresses (Siemens), iii
- English language option, 57
- Equidistant DP, 107, 134
- Error OBs, 68
- Errors, 43, 135
- Europe, customer support, iii
- Events, diagnostic, 47
- Execution monitor, 45, 81, 84, 89, 93, 97, 103
- Execution priorities, 88
  - adjusting, 88
- Execution time jitter, 84
- Execution time limit, 97
- Execution times of instructions, 130
- Exit command (File menu), 50
- External power supply, 56
- EXTF status indicator, 43, 133

## F

- FBs, 64
- FCs, 64
- Features of PC-based control, 3
- File menu
  - Archive, 49
  - Exit command, 50
  - Restore, 49
- Flag memory size increase, 5

- Flashing indicators, 43
- Forced execution sleep time, 89, 97
  - eliminating, 103
- Forced execution sleep time/counter, 45
- Forced sleep, 96
- Format for diagnostic buffer, 47
- Forward slash (in controller name), 35
- FRCE status indicator, 43
- Free cycle, 68, 81, 84, 93, 97, 103
- French language option, 57
- Full-text search, 8

## G

- German language option, 57
- GET (SFB 14), 65
- Getting started, 17
- Group name, OPC server connection, 119
- Guest user privileges, 7

## H

- Hardware abstraction layer (HAL)
  - Changing the HAL type, 131
  - HAL timer period, effect on sleep time parameter, 96
  - Setting the HAL timer, 131
- Hardware configuration
  - adding OPC server, 112
  - operational parameters, 63
  - STEP 7, 32
- Hardware interrupts, 68, 135
- Help menu
  - Using help, 8
- Help on Event, diagnostic buffer, 47
- HMI, effect on cycle time, 93
- Hotline (Siemens), iii
- Hyphen in controller name, 35

## I

- I/O
  - access error, 68, 135
  - available, 127



- communication, 22
  - configuration, 32
  - overview, 1
  - supported communication interfaces, 20
  - technical data, 127
- IF slots, 24
- configuration, STEP 7, 32
  - configuring, 25
- Index, definition, 21
- Indicator LEDs, 43
- Industrial Ethernet
- communication supported, 5, 20
  - INI file, 27
  - STEP 7 communication, 30
  - submodule limit, 24
  - TCP connections supported, 127
- INI File configuration, 27
- Insert New Connection dialog, 115
- Inserting sleep time, 96, 97, 103
- Installation
- Automation License Manager, 14
  - licensing, 15
  - overview, 11
  - prerequisite software deletion, 11
  - removing, 15
  - requirements, 6
  - WinAC RTX software, 14
  - WinAC RTX TimeSync, 14
  - Windows user privileges, 7
- Interface slots, 24
- configuration, STEP 7, 32
  - configuring, 25
- Internet web sites (Siemens), iii
- Interrupt assignments, 107
- Interrupt mode, 134
- Interrupt OBs, 68, 84
- Interrupts, 81, 135
- INTF status indicator, 43
- Invalid characters, controller names, 35
- IP address assignment, 27
- IRQ assignments, 107, 134
- Isochronous Mode, 107
- interrupt mode requirements for DP interface, 134
- J**
- Jitter, 84, 88, 89
- reducing, 45, 88, 89, 96, 97, 103
  - retentive data SFCs, 56
  - tuning panel, 45
- K**
- Keyswitch position, 40
- L**
- Language selection, 57
- LED indicators, 43
- LEDs, CP 5613, 28
- Licensing the software, 15
- Load instructions, 135
- Load memory, 127
- Adding blocks with SFC82 and SFC84, 56
  - Creating, reading, and writing blocks, 56
  - reloading, 54
  - saving, 52
- Local data space, 127
- Local ID, OPC server connection, 115
- Logic blocks, 64
- OBs, 68
  - SFBs, 78
  - SFCs, 73
- Logic blocks, address ranges, 127
- M**
- Main program cycle, 68
- Managing sleep time, 89
- Manuals, iii
- Manufacturer-specific interrupt, 68
- Math operations, execution times, 130
- Maximum execution load, 45, 89, 97, 103

- Memory, 52
  - requirements, 6
  - saving, 52
  - specifications, 127
- Memory areas, reloading on startup, 54
- Memory problems, STEP 7 user program, 10, 32, 127
- Memory reset, 42
- Meter, 79
- Meter, run-time, 79
- Minimum scan cycle time, 45, 81, 89, 93
- Minimum sleep time, 45, 81, 89, 93, 96, 97, 103
- Minus sign (in controller name), 35
- Mode, operating
  - allowed and prohibited actions, 40
  - changing, 40
  - saving switch position, 52
  - switch position on startup, 54
- Module access errors, 135
- Module pull/plug interrupts, 68
- Module removal/insertion, 135
- MPI, 30
- MRES, 42
  - effect on status indicators, 43
- N**
- Naming controller (invalid characters), 35
- Nesting depth, 127
- Netmask assignment, 27
- NetPro, adding S7 connection for OPC server, 115
- Network, 133
  - protocols, 30
  - STEP 7 communications, 30, 32
  - troubleshooting, 133
- North America, customer support, iii
- O**
- OB 1, 61, 81
  - example, avoiding jitter in starting cyclic interrupt, 103
  - jitter, 84
  - managing sleep time, 93
- OB 100 and OB 102, 54, 61, 81
  - startup after blue screen, 51
- OB 122, 135
- OB 20, effect on cycle time, 93
- OB 32 to OB 36, jitter, 103
- OB 35, effect on scan cycle, 81, 93
- OB 40, 135
  - effect on scan cycle, 81, 93
  - jitter, 84
- OB 61/OB 62
  - isochronous mode usage, 107
- OB 82, 135
- OB 83, 135
- OB 84, blue screen occurrence, 50
- OB 85, 135
- OB 86, 135
- OB\_STR\_INFO, 54
- Object properties, STEP 7, 63
- OBs, 68
  - diagnostic events, 135
  - execution, 84
  - interrupting sleep time, 97
  - supported by controller, 68
- ODK interrupt, 68
- ON status indicator, 43
- OPC Navigator, 119
- OPC Project
  - adding OPC server connection, 119
  - creating, 119
  - defining items to access, 119
- OPC Scout, 119
- OPC server
  - adding S7 connection in NetPro, 115
  - adding to STEP 7 hardware configuration, 112
  - addition in Station Configuration Editor, 110
  - configuration overview, 109

- configuring, 112
- downloading configuration to controller, 118
- properties, 112
- required SIMATIC NET installation, 11
- Opening the control panel, 39
- Operating mode, 40
  - allowed and prohibited actions, 40
  - changing, 40
  - saving switch position, 52
  - status indicators, 43
  - switch position at startup, 57
- Operating system requirements, 6
- Operating system threads, 88
- Operational parameters, 63
- Optimizing performance, 45
- Order number, 127
- Organization blocks, 68
- Overview
  - getting started, 17
  - OPC server configuration, 109
  - PC-Based Control, 1
- P**
- Pacific Region, customer support, iii
- Panel
  - opening and closing, 39
  - status indicators, 43
- Parameters, sleep monitor, 97
- Part number, 127
- Password
  - changing, 59
  - creating/changing, 58
  - prompt interval, 58
- PC Boot, starting controller, 60
- PC requirements, 6
- PC station, 25
  - communication interface capabilities, 18
  - comparison to S7-400, 18
  - component difference from WinLC RTX
    - submodule, 22
  - configuring with STEP 7, 32
  - definition, 18
  - index, 21
  - names, 32
  - OPC Server component addition, 110
  - STEP 7 communication interface, 30
- PC-based control
  - features, 3
  - introduction, 1
- PCI slot
  - independence from configured IF slot, 24
  - IRQ assignments, 134
- Performance, 89
  - DP interface, 134
  - improvements, 5
  - tuning, 45, 81, 89, 93, 97
- Period (in controller name), 35
- PG/OP communication, 22, 30
  - supported communication interfaces, 20
- Polled mode, 134
- Position of mode selector switch on startup, 54
- Power failure, 56
- Power user privileges, 7
- Power-Down State, 52, 54
  - using UPS to save, 56
- Priority, 45, 68, 89, 97, 103
  - adjusting, 45, 81, 88, 89
  - effect of other applications on cycle time, 84, 93
  - OBS, 68
  - RTSS values, 88
  - setting, 103
- Priority class, 68
  - error, 68
  - values, 68
- Priority class error, 135
- Privileges, 7

Process image, 127  
Process image partition, 107  
Processing interrupt (stop avoidance), 68  
PROFIBUS-DP  
    communicating with I/O, 22  
    CP card submodule, 25  
    DPV1 extensions, 66  
    equidistant DP, 107  
    isochronous mode, 107  
    network troubleshooting, 133  
    specifications, 127  
    supported communication interfaces, 20  
Program blocks, new, 5  
Programming error, 68  
Properties - OPC Server dialog, 112  
Protocols, 30  
PUT (SFB 15), 65

**R**

Rack failure, 68  
RAM requirements, 6  
RDSYSST (SFC 51), 136  
Reading data blocks, 56  
Real-time subsystem priority, 88  
Real-time system clock, 79  
Recovering from a blue screen, 50  
Recovering from a defective state, 43  
Recovering license, 15  
Registering controller for start at PC boot, 60  
Reloading memory areas on startup, 54  
Remote STEP 7 connection, 30  
Removing  
    submodules, 29  
    WinAC RTX software, 15  
Renaming the controller (invalid characters), 35  
Requirements, 6  
Resetting memory areas, 42  
Resolving IRQ conflicts, 134  
Resources (computer), 93

Responding to diagnostic events, 135  
Restart, 54  
    autostart feature, 57  
    following blue screen, 50  
    restart method, 61  
Restoring, 49  
Retentive data, 52  
    SFCs, 56  
Ring on test, 28  
RTSS priority, 88  
RTX extensions (Ardence), 12  
    advantages, 3  
    HAL (hardware abstraction layer), 131  
    installation, 12  
RTX-controlled devices, 134  
RUN  
    operating mode, 40  
    status indicator, 43  
Running controller without a license, 15  
Run-time meter, 79

**S**

S7 Blocks, 64  
S7 Communication  
    functions, 65  
    supported communication interfaces, 20  
S7 connections  
    adding for OPC server in NetPro, 115  
    number supported, 127  
S7 memory areas, saving, 52  
S7 routing, supported communication interfaces,  
    20  
S7-400 communication, 18  
    communication model, 18  
    PC-based control comparison, 18, 22

**Safety notification, 2**  
Scan cycle, 45, 81, 89, 93, 97, 103  
    adjusting priorities, 88  
    jitter, 84

- monitoring, 45
- Security
  - setting level, 58
- Security command (CPU menu), 58
- Selecting, 57
  - autostart, 57
  - language for controller panel and help, 57
  - restart method, 61
- Send Authorization command, 107
- Service, 60
  - starting the controller after reboot, 60
- Setting
  - HAL timer/HAL type, 131
  - PG/PC Interface, 30
  - priority, 88, 103
  - restart method, 61
  - security options, 58
  - sleep monitor parameters, 97
- Setup, 11, 14
  - Ardence RTX, 12
- SFBs
  - S7 communications, 65
  - supported by controller, 78
  - supported by WinLC RTX, 78
- SFC 126/SFC 127
  - isochronous mode usage, 107
- SFC 13, 135
- SFC 39, 135
- SFC 42, 135
- SFC 47 (WAIT), 81, 89, 96, 103
- SFC 51, 136
- SFC 62, 65
- SFC 87, 65
- SFCs, 73
  - reading the system status list, 136
  - retaining data, 56
  - running asynchronous SFCs concurrently, 73
  - S7 communications, 65
  - scan cycle jitter, 73
  - supported by controller, 73
  - Windows blue screen, 50, 51
- Shutting down, 39, 52, 54, 56
- Siemens contact information, iii
- SIEMENS PCs, built-in PROFIBUS interfaces, 20
- SIMATIC NET
  - optional installation, 11
  - removal of requirement for WinAC RTX, 5
  - requirement for PC Station component configuration, 18
  - requirement for using OPC server, 109
- Slash (in controller name), 35
- Sleep interval, 84
  - inserting, 103
- Sleep management techniques, 89
- Sleep time
  - adding with SFC 47, 96
  - adjusting, 45, 93, 97
  - jitter, 84, 103
  - relationship to other applications, 88
  - scan cycle, 81
  - techniques for sleep management, 89
- Sleep time parameter, SFC 47
  - relationship to HAL timer period, 96
- Sleep-monitoring algorithm, 81, 89, 93, 97, 103
- Sorting, diagnostic buffer events, 47
- South America, customer support, iii
- Specifications, 127
- SSL ID, 136
  - 0x11, 137
  - 0x12, 137
  - 0x13, 137
  - 0x14, 138
  - 0x15, 138
  - 0x19, 138
  - 0x1C, 139
  - 0x22, 139

- 0x25, 139
- 0x32, 140
- 0x74, 140
- 0x90, 141
- 0x91, 141
- 0x92, 142
- 0x95, 142
- 0xA0, 142
- 0xB1, 143
- 0xB3, 143
- 0xB4, 143
- backup memory (0x13), 137
- block types (0x15), 138
- C memory size (0x14), 138
- communications status (0x32), 140
- component identification (0x1C), 139
- contents (SSL IDs supported), 136
- CPU characteristics (0x12), 137
- CPU LED status (0x74), 140
- CPU LED status, local only (0x19), 138
- DB number and size (0x15), 138
- diagnostic buffer (0xA0), 142
- DP master system (0x90), 141
- DP master system, expanded (0x95), 142
- DP module diagnostics (00B1, 00B3, 00B4), 143
- DP module status (0x91), 141
- DP rack/station status (0x92), 142
- expanded DP master system (0x95), 142
- FB and FC number and size (0x15), 138
- H CPU LED status (0x74), 140
- I memory size (0x14), 138
- identification, module (0x11), 137
- interrupt status (0x22), 139
- L memory size (0x14), 138
- LED status for redundant modules (0x74), 140
- LED status, local only (0x19), 138
- Load memory (0x13), 137
- local module LED status (0x19), 138
- M memory size (0x14), 138
- master system (0x90), 141
- maximum number and size of blocks (0x15), 138
- memory area sizes (0x14), 138
- memory areas (0x13), 137
- module diagnostics (00B1, 00B3, 00B4), 143
- module identification (0x11), 137
- module LED status, local and redundant CPU (0x74), 140
- module LED status, local only (0x19), 138
- module status (0x91), 141
- OB number and size (0x15), 138
- order number (0x11), 137
- peer-to-peer memory (0x13), 137
- PII memory size (0x14), 138
- PIQ memory size (0x14), 138
- process image partitions (0x25), 139
- Q memory size (0x14), 138
- rack/station status (0x92), 142
- redundant CPU LED status (0x74), 140
- SDB number and size (0x15), 138
- SFC51, 136
- shadow memory (0x13), 137
- size of blocks (0x15), 138
- size of the memory areas (0x14), 138
- size of user memory areas (0x13), 137
- SSL IDs supported, 136
- station, module status (0x91), 141
- status, communications (0x32), 140
- supported, 136
- system areas (0x14), 138
- T memory size (0x14), 138
- types of blocks (0x15), 138
- User memory (0x13), 137
- version number (0x11), 137
- Start events, 68
- Starting the controller, 39

- after blue screen, 51
  - at PC boot, 60
  - autostart, 57
  - effect on execution monitor, 97
  - programming blue screen detection, 54
  - reloading memory areas, 54
  - restart method, 61
  - State, saving, 52
  - Station configuration editor
    - index in PC station, 21
    - OPC Server component addition, 110
    - using to configure submodules, 25
  - Station failure, 135
  - STATUS (SFB 22), 65
  - Status alarm interrupt, 68
  - Status indicators, 43
  - STEP 7
    - connecting to WinLC RTX, 30
    - diagnose hardware feature, 133
    - hardware configuration, 32
    - OPC server, 112
    - renaming the controller (invalid characters), 35
    - requirements, 6
    - symbol table usage, OPC server, 119
    - system status list (SSL), 136
    - task summary, 63
  - STEP 7 user program
    - actions affected by operating mode, 40
    - adding sleep time with SFC 47, 96
    - archiving, 49
    - debugging, effect on cycle time, 93
    - deleting, 42
    - inserting sleep interval, 103
    - jitter, 84
    - memory problems, 32
    - restoring, 49
  - STOP
    - operating mode, 40
    - status indicator, 43
  - Submodules
    - capabilities, 18
    - configuration in WinLC RTX Properties, 25
    - configuring in STEP 7, 32
    - connecting through to STEP 7, 30
    - definition, 22
    - difference from PC station components, 22
    - IF slots, 24
    - removing, 29
  - Support, iii
  - Symbols
    - invalid characters in controller names, 35
    - OPC server data, 119
  - Synchronous cycle interrupts, 68
  - Synchronous interrupt OBs, 107
  - System clock, 79
  - System function blocks (SFBs), 78
  - System functions (SFCs), 73
  - System requirements, 6
  - System status list (see SSL\_ID), 136
- ## T
- Task summary, STEP 7, 63
  - TCON (FB 65), 67
  - TCP connections supported, 127
  - TCP/IP communication blocks, 67
  - TDISCON (FB 66), 67
  - Technical specifications, 127
  - Technical support, iii
  - Telephone numbers (Siemens), iii
  - Testing
    - CP 5613 configuration, 28
    - jitter, 84
  - Threads, 84, 88
  - Time (system clock), 79
  - Time error, 68
  - Time Sync, 5
  - Time-delay interrupt, 68

Time-of-day interrupt, 68  
Timer, HAL (hardware abstraction layer), 131  
Timers supported, 127  
TimeSync  
    feature description, 5  
    installation, 14  
Timing adjustment, 45  
Timing resolution, 10  
Tools for OPC Server configuration, 109  
Transfer instructions, 135  
Transferring installed license, 15  
TRCV (FB 64), 67  
Troubleshooting  
    defective state, 43  
    error conditions, 43  
    network problems, 133  
TSEND (FB 63), 67  
Tuning panel, 45, 81, 84, 88, 93, 97, 103  
Tuning performance, 81, 89, 93  
Type, HAL (hardware abstraction layer), 131

## U

Unbuffered startup, 54  
Uninstalling  
    software prior to WinAC RTX installation, 11  
    WinAC RTX, 15  
Uninterruptible power supply (UPS), 56  
Unrecoverable fault in Windows, 50  
Update alarm interrupt, 68  
UPS, 56  
URCV (SFB 9), 65  
USEND (SFB 8), 65  
User privileges, 7  
USTATUS (SFB 23), 65

## V

Valid characters, controller name, 35  
VAT, effect on cycle time, 93

Venturcom RTX (now Ardenze RTX), 12  
    advantages, 3  
    HAL (hardware abstraction layer), 131  
    installation, 12

Verifying  
    configuration, 37  
Virtual backplane bus, 18

## W

WAIT (SFC 47), 96  
Warm restart, 68  
    configuring, 61  
Watchdog timer, 93  
Web sites (Siemens), iii  
WinAC RTX installation  
    Automation License Manager, 14  
    licensing, 15  
    overview, 11  
    prerequisite software deletion, 11  
    removing, 15  
    WinAC RTX software, 14  
    WinAC RTX TimeSync, 14  
    Windows user privileges, 7  
Windows  
    automatic reboot, 51  
    blue screen (unrecoverable fault), 50  
    UPS settings, 56  
    user privileges, 7  
    Windows-controlled devices, 134  
WinLC RTX differences from WinLC Basis, 10  
WinLC RTX Properties, 24  
    configuring a submodule, 25  
    removing submodules, 29  
Work memory, 127  
    reloading, 54  
    saving, 52  
Writing data blocks, 56