SIEMENS

SIMATIC

Windows Automation Center RTX WinAC RTX 2008

Operating Instructions

This documentation is part of the WinAC RTX 2008 package with order number:
6ES7671-0RC06-0YA0 or 6ES7671-0RC06-0YE0

07/2008
A5E00486536-02

Safety Guidelines

This manual contains notices you have to observe in order to ensure your 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. These 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 information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used 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 notes 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.

Prescribed Usage

Note the following:

⚠WARNING
This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which 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

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

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
Industry Sector
Postfach 48 48
90327 NÜRNBERG
GERMANY

A5E00486536-02
Ⓟ 07/2008

Preface

Purpose of the Documentation

This documentation provides detailed information about the Windows Automation Center with Real-Time Extensions (WinAC RTX 2008) software package that includes the following components:

- Windows Logic Controller RTX (WinLC RTX V4.4)
- Ardence RTX V8.1
- WinAC Time Synchronization V4.1
- Automation License Manager V4.0
- SIMATIC NET 2007 V7.0 HF1 incl. Softnet S7 Lean V7.0 License

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 or PROFINET network to communicate with distributed I/O, such as an ET 200S 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 XP operating systems and STEP 7 programming.

Scope

This document describes the features and the operation of WinAC RTX 2008.

Changes Compared to the Previous Version

The topic "What's New? (Page 16)" in the Product Overview enumerates the new features of WinAC RTX 2008.

Location of Documentation

The WinAC RTX 2008 installation includes this documentation as both online help and a PDF online manual and also a PDF online manual for WinAC Time Synchronization V4.1. 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. http://support.automation.siemens.com/WW/view/en/18652056
- *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. http://support.automation.siemens.com/WW/view/en/1214574
- *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. http://support.automation.siemens.com/WW/view/en/18652511
- *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) http://support.automation.siemens.com/WW/view/en/13542666
- *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. http://support.automation.siemens.com/WW/view/en/22205381
- *Ardence RTX Runtime Release Notes:* These release notes include the system requirements for RTX and further information about RTX

  (on the DVD: WinAC_RTX\Ardence\RTXRuntimeRelNotes.pdf)
- *PROFINET System Description:* This manual explains installation, commissioning and operation of a PROFINET system, as well as instructions and examples for programinning diagnostics for I/O devices. http://support.automation.siemens.com/WW/view/en/19292127
- *From PROFIBUS DP to PROFINET IO:* This manual explains differences between PROFIBUS DP and PROFINET IO and provides assistance in migrating from PROFIBUS DP to PROFINET IO. http://support.automation.siemens.com/WW/view/en/19289930

- *PROFINET Getting Started Collection:* This manual collection includes information about configuring specific PROFINET interfaces and specific PROFINET IO devices. http://support.automation.siemens.com/WW/view/en/19290251
- *SIMATIC Communication:* This manual provides a general overview of communication networks and communication technologies used in the automation field with an emphasis on SIMATIC products. http://support.automation.siemens.com/WW/view/en/25074283
- *Isochrone Mode:* This manual provides a complete overview of "isochrone mode": http://support.automation.siemens.com/WW/view/en/15218045

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
For information about Ardence Real-Time Extensions (RTX):Internet: http://www.Ardence.com		

Table of contents

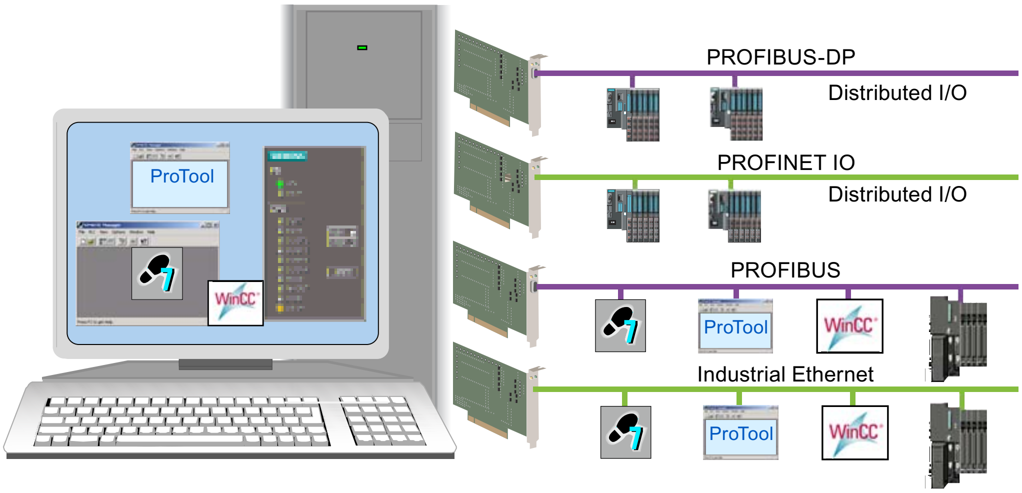
Product Overview 1

1.1 Introduction to PC-Based Control

The WinAC (Windows Automation Center) PC-based controllers provide the same functionality as SIMATIC S7 CPUs (hardware controllers). WinLC RTX is a programmable software PLC — a software application that runs on a standard computer (PC) with real-time extensions.

WinLC RTX supports multiple networks and connects to the distributed I/O, such as ET 200S, by means of PROFIBUS-DP or PROFINET communication interfaces (Page 33) 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 (Page 12) that runs on the PC. With these capabilities, you can use WinLC RTX in a typical factory automation configuration.

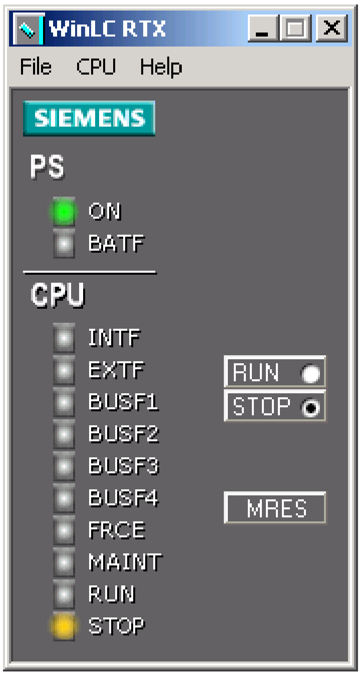
1.2 Introduction to the WinAC RTX Controller Panel

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 (Page 57) (similar to the mode selector slider on an S7 CPU front panel.)
- An MRES switch position for resetting the memory areas (Page 59)
- Status indicators (Page 60) for the controller
- Menus for controller operation

WinLC RTX Controller Panel



1.3 Relationship between the Controller and the Controller Panel

Opening and Closing 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.

WinAC RTX Icon in the Windows Taskbar

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.

The color of the background border of this icon provides some additional information about the controller:

- Yellow: The controller is in STOP mode.
- Green: The controller is in RUN mode.
- Red: The controller is in a defective state.

The color indicates the actual operating mode, not the mode selector position. The mode selector can be set to RUN, for example, but the controller could be in STOP mode due to a program execution error or an operating mode change from STEP 7.

1.4 PC-based Control Features

Real-time Process Control

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 (Ardence RTX V8.1) 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 (Page 69). 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 code blocks of SIMATIC controllers: Organization Block (OB), System Function Block (SFB), and System Function (SFC)
- Supports PROFIBUS-DP for communication with distributed I/O, including DPV0 and DPV1 (Page 120) slaves (PROFIBUS DPV1 provides enhanced alarm and status reporting, in order to communicate with intelligent slave devices)
- Supports PROFINET (Page 121) communication over Ethernet submodules: PROFINET I/O (Page 124) for communication with distributed I/O and PROFINET CBA (Page 130) (Component Based Automation) for communication with other network components
- Supports up to four separate subnets for connecting to distributed I/O
- Supports an isochronous (Page 174) mode for PROFIBUS-DP subnets, 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 (Page 119) between controllers (hardware or software) on the network
- Supports the routing of S7 communications through the submodule CP cards 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
- Supports time synchronization via NTP
- Provides ability to archive and restore (Page 68) control programs
- Allows you to control the operating mode of the controller and to view status information from the controller panel (Page 12)
- Provides a tuning panel (Page 62) 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)

Using WinLC RTX with other WinAC Products

WinLC RTX can work together with WinAC ODK and the WinAC Slot PLCs for your total automation solution. With these products (sold separately), you can perform the following tasks:

- Use WinAC ODK (Open Development Kit) to develop custom PC applications in a high-level programming language that exchange data with WinLC RTX using one of two programming interfaces to the STEP 7 user program: CCX (Custom Code Extension) and SMX (Shared Memory Exchange.)
- Use WinAC ODK to develop user interface software such as a custom controller panel to display status information and perform controller operations using the CMI programming interface (Controller Management Interface.)
- Control your process with WinLC RTX and 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 (Page 18) (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 (Page 97) (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.

1.5 What's New?

New Features: WinAC RTX 2008:

The following features are new for WinAC RTX 2008:

- Support of PROFINET IO (Page 124) in addition to PROFIBUS-DP for communication with distributed I/O
- Support of Open User Communication blocks (Page 119) using PROFINET interfaces
- Support of PROFINET CBA (Page 130)
- Support of time synchronization (Page 13) via NTP
- Support of Embedded Controller (Page 99) (S7-mEC)
- Simpler installation: WinLC RTX installs Ardence RTX (Page 23) directly.
- Maintenance LED (Page 12) on the WinLC RTX controller panel
- Configuration of WinAC Data Storage and usage of PLC Operating Mode LEDs from the controller panel options
- PC load display on tuning panel (Page 62)
- Support for SIMATIC Software Redundancy

1.6 System Requirements

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

Category	Requirement
Operating System	Microsoft Windows XP Professional Service Pack 2 Ardence RTX version 8.1 (included with WinAC RTX) **Note:** Some hardware configurations (Page 89) 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 DVD for hardware and software system requirements for RTX.
Processor and memory	Pentium multiprocessor system: • 900 MHz • 1 Gbyte or more of RAM • BIOS must support plug-and-play (ACPI, Advanced Configuration and Power Interface) **Note:** Multi-core and hyperthreading systems are also supported.
Hard drive	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 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)
Operator interface	A color monitor, keyboard, and mouse or other pointing device (optional) that are supported by Windows
Communication interface	One or more communication interfaces for communications with STEP 7 or other S7 applications or for communications with distributed I/O: See the topic What Is a Communication Interface? (Page 33) for a complete list of communication interfaces that WinLC RTX supports.
Siemens software	Programming and configuration software: STEP 7 V5.4 SP4 with the installed hardware update for WinLC RTX .
SIMATIC NET (optional)	You must install SIMATIC NET from the WinAC RTX installation DVD if you need features such as use of the OPC Server. The SIMATIC Softnet-S7 Lean (6GK1 704-1LW63-3AA0) license is included with your WinAC RTX installation package. For more information about SIMATIC NET products for PC-based automation refer to the SIEMENS Mall or the ST PC catalog.

1.7 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 (Page 57) of the controller, modifying the sleep time or the minimum scan time (Page 159) of the controller, archiving or restoring (Page 68) control programs, or setting the security options (Page 94).

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

1.8 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

Accessing Help from the Controller Panel

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

- Click an entry on the Help menu.
- 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).

The menu commands available from the controller panel Help menu are listed below:

- 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.
- 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.

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 "Hide" button on the browser to close the table of contents.
- Click the "Show" button on the browser and select the Contents tab to display the table of contents.

Using the Index

The index provides access to information about a specific subject. Select the Index tab to access the index. (If the Index tab is not visible, click the "Show" button on the browser.)

Using Full-Text Search

To use the full-text search capabilities of the online help select the Search tab. (If the Search tab is not visible, click the "Show" button on the browser.)

The full-text search supports the Boolean operators AND (&), OR (|) and NOT (!), expressions within quotation marks, nesting of expressions with parentheses, and the wildcards * and ? in your search expression.

Printing Help Topics

To print all or part of the online manual that corresponds to the help system, follow these steps:

1. Use the **Start > SIMATIC > Documentation** menu command to open the PDF file.
2. Use the **File > Print** menu command of Adobe Reader to print all or part of the manual.

Installation 2

2.1 Overview of the Installation Tasks

To install WinAC RTX, your computer must meet the system requirements (Page 17).
You must have Windows administrator (ADMIN) privileges and complete the following tasks:

- Make an archive of your existing STEP 7 user program and note your custom settings (optional).
- Uninstall any of the following software packages if they exist on your computer in the order listed, rebooting when finished:
  - WinLC Basis or WinLC Basis Demo
  - WinLC RTX
  - Ardence or VenturCom RTX
- Install the WinAC RTX software.

For communication with distributed I/O, your computer must have one or more communication interfaces (Page 33).

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

Following installation, license the WinAC RTX installation (Page 25) using the Automation License Manager.

The succeeding topics contain the installation and licensing procedures.

2.2 Preparing for Installation

A new installation of WinAC RTX removes any existing STEP 7 user program and configuration that you have, as well as the retentive data and any prior settings for the tuning panel, Station Configuration Editor, WinAC Data Storage, and other WinAC RTX options.

Procedure

To preserve your STEP 7 user program and other settings for restoration after installation, follow these steps:

1. Archive (Page 68) the STEP 7 user program and configuration.
2. Record your individual settings in these areas:
   - Station Configuration Editor
   - WinLC Properties
   - Tuning Panel
   - Data Storage
   - Autostart and Password

Following installation, you can restore (Page 68) your STEP 7 user program and configuration and re-enter the settings that you noted.

2.3 Installing the WinAC RTX Software

To install the WinAC RTX software including the Ardence RTX extensions, remove in order the software named in "Overview of the Installation Tasks" (Page 21) . Insert the WinAC RTX DVD and follow the instructions of the setup. If the setup program is not running, double-click the setup.exe file on the installation DVD.

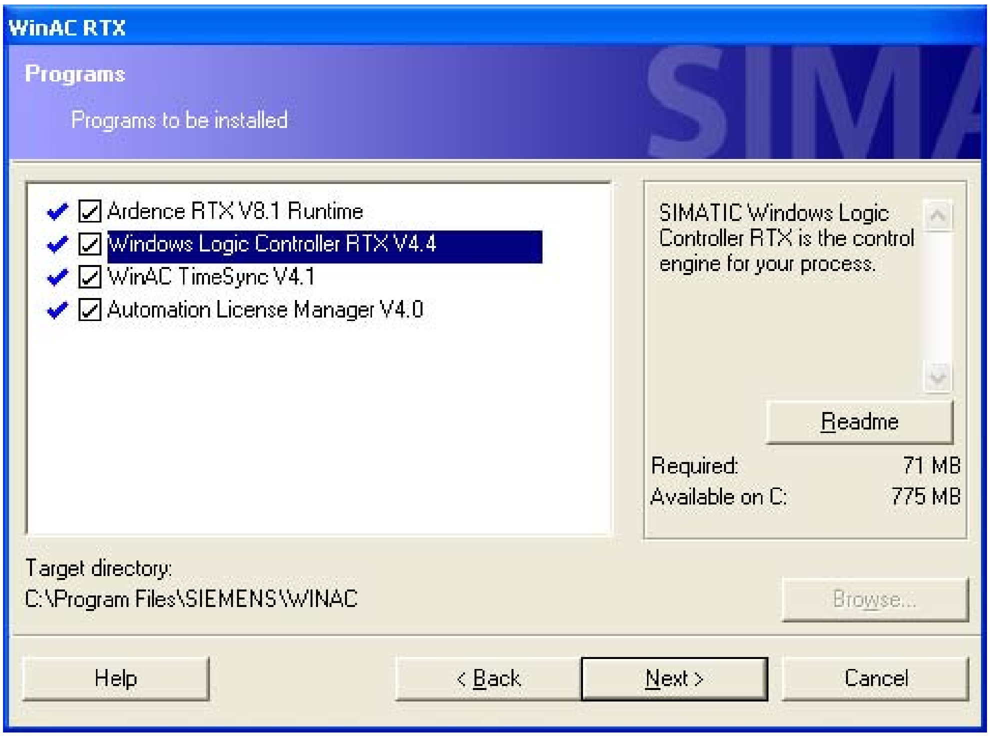
Note

You must remove any previous versions (Page 21) of WinAC or RTX from your computer and you must have Windows administrator (ADMIN) privileges to install the WinAC RTX software.

Procedure

To install the WinAC RTX software, follow these steps:

1. Select the language for performing the installation.
2. From the list of WinAC RTX components, select the components to be installed:



WinAC RTX

Programs

Programs to be installed

✓ ☑ Ardence RTX V8.1 Runtime
✓ ☑ Windows Logic Controller RTX V4.4
✓ ☑ WinAC TimeSync V4.1
✓ ☑ Automation License Manager V4.0

SIMATIC Windows Logic Controller RTX is the control engine for your process.

Required: 71 MB
Available on C: 775 MB

Target directory:
C:\Program Files\SIEMENS\WINAC

After you make your selections and click the Next button, the Setup program continues and displays the progress of the installation.

3. Proceed through the setup 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 specify on subsequent dialogs

You can also choose to license WinAC RTX (Page 25) 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.

Result

The setup program notifies you when the installation is complete.

2.4 Installing SIMATIC NET

You must install SIMATIC NET to configure a communication interface in the PC Station, to use the OPC server, or to use other SIMATIC NET features. You do not need to install SIMATIC NET to configure communication interfaces as WinLC RTX submodules.

SIMATIC NET including the Softnet-S7 Lean V7.0 license is included on your WinAC RTX installation DVD. The license keys for SIMATIC NET along with the WinLC RTX license key is on the USB stick that is included with your WinLC RTX installation package.

To install SIMATIC NET and the Softnet-S7 Lean V7.0 license, follow the instructions in the SIMATIC NET readme.

2.5 Licensing the WinAC RTX Software

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

Certificate of License

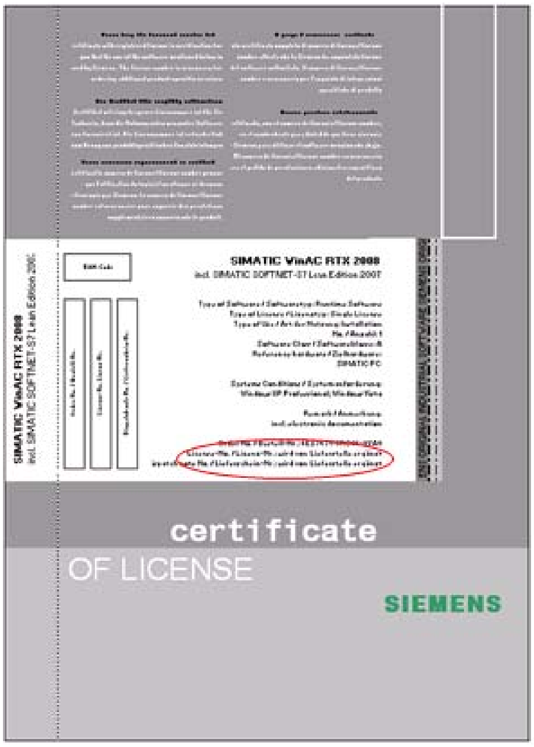
Your release includes a paper Certificate of License that shows your unique license number. This is proof that you have a valid license for WinLC RTX. Store this certificate in a safe place so that it easily accessible from the location of the computer on which WinLC RTX executes. In the event you need to replace your license key due to a lost or damaged license key, contact the Siemens hotline (http://www.siemens.com/automation/service&support) with your Certificate of License in hand. You must have a valid Certificate of License to get a replacement license key.

License Key

The license key for WinAC RTX is on a USB stick that is included with your release.

In the event that the USB license stick is lost or damaged, you can call the hotline to recover your license key. You need the Certificate of License to get a replacement license key from Siemens.

The picture below shows a Certificate of License that corresponds to a specific license key. The license key for some SIMATIC products is on a diskette; the license key for WinAC RTX is on a USB stick:


SIMATIC WinAC RTX 2008
incl. SIMATIC SOFTNET-S7 Lean Edition 2007
SIMATIC PC
certificate
OF LICENSE
SIEMENS

Installing the License Key during Installation

When you install WinAC RTX, the Automation License Manager is part of the installation set. If it is not already on your computer, select the Automation License Manager check box during installation to install it on your computer during the WinAC RTX installation.
A subsequent dialog allows you to choose whether to install the license key during installation. Refer to the Automation License Manager online help for help with installing a license key.
If you do not install the license key, WinLC RTX will display a message periodically that no license key exists.

Installing the License Key at a Later Date

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

4. Insert the WinAC RTX installation DVD and start the setup.
5. When the Components dialog appears, select the Automation License Manager checkbox.
6. 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.
7. Proceed with license key installation according to the instructions in the Automation License Manager online help.

If the Automation License Manager is already installed on your computer, follow the last two steps of the preceding procedure to license WinLC RTX.

Transferring an Installed License Key

The Automation License Manager provides steps to transfer a license key 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 Key

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

Recovering the License Key in Case of a Defective Hard Drive

If a problem occurs with the license key file on your hard disk or the USB stick, contact your local Siemens representative (Page 3). You will need your paper Certificate of License in hand.

2.6 Uninstalling Ardence RTX or WinLC RTX

Procedure

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

1. Double-click the Add/Remove Programs icon in the Windows Control Panel.
2. Select the Ardence RTX, RTX, or SIMATIC Windows Logic Controller RTX component entry in the displayed list of installed software.
3. Click the Change button to remove WinLC RTX or the Remove button to remove Ardence RTX.
4. For WinLC RTX, select Remove in the appropriate dialog to remove the software.

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

Note

Uninstalling WinLC RTX automatically closes the controller panel and WinLC RTX if they are executing. You do not see a message first.

Getting Started 3

3.1 Getting Started Overview

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 (Page 38) of WinLC RTX.
- Use STEP 7 to configure (Page 47) 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 (Page 29), Communication Interface (Page 33), Index (Page 33), Submodule (Page 35), and Interface (IF) Slot (Page 37).

3.2 Understanding the Concepts

3.2.1 What Is a PC Station?

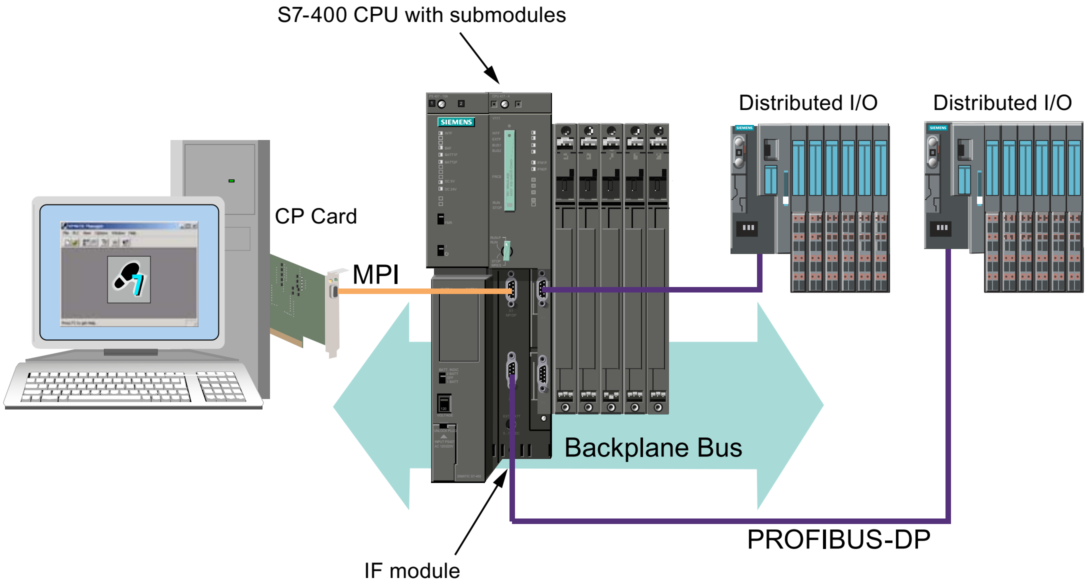
The PC station is a software-based virtual rack, represented in the Station Configuration Editor, for creating a PC-based automation system. Like a hardware rack of an S7 CPU-based automation system, it contains space for several modules required for the PC-based automation system.

When you install the WinAC RTX software, the controller appears by default in the second slot (index (Page 33)) of this virtual rack in the Station Configuration Editor. The PC Station is also represented in the STEP 7 Hardware Configuration editor (Page 47). The controller in the PC Station contains four configurable IF Slots for designating communication interfaces (Page 33) as submodules (Page 35) 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 uses a built-in submodule interface or an IF module to communicate with distributed I/O (in this example, over a PROFIBUS-DP subnet).



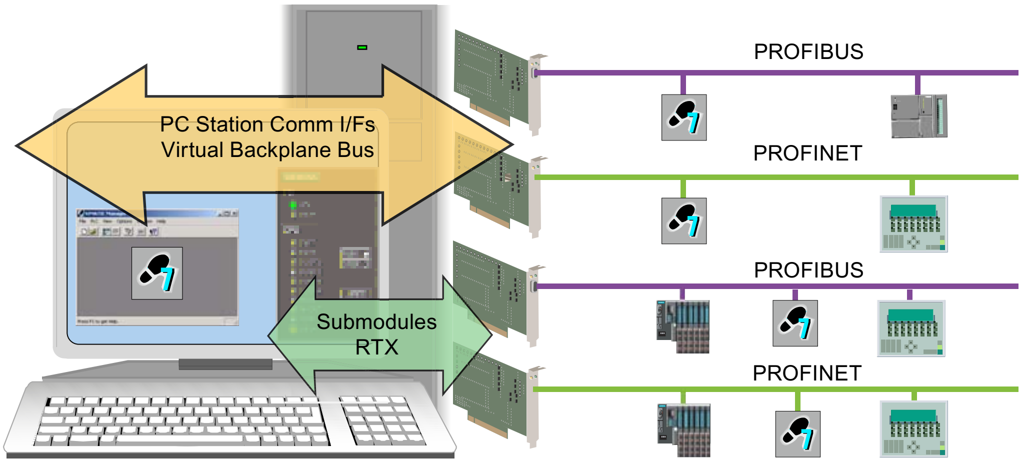
In an S7-400 station, the following types of communications are possible:

Onboard Interfaces	CP Modules used over Backplane Bus
Operation of distributed I/O Supported interfaces: • MPI • PROFIBUS • PROFINET	Operation of central I/O Supported communication processors: • PROFIBUS • PROFINET • Industrial Ethernet

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 distributed I/O Supported protocols/communication types: **PROFIBUS** • PG/OP communication • S7 communication • S7 routing • PROFIBUS-DP I/O **PROFINET** • PG/OP communication • S7 communication • S7 routing • Open User Communication (TSEND/TRCV) • PROFINET IO • PROFINET CBA	Operates in the Windows environment No access to distributed I/O or central I/O Supported protocols/communication types: **PROFIBUS** • PG/OP communication • S7 communication • S7 routing **Industrial Ethernet** • PG/OP communication • S7 communication • S7 routing
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 (Page 38).

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 (Page 47) tool to configure the components that you installed in the PC station.

After you complete hardware configuration in STEP 7 and submodule configuration (Page 38) 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.

3.2.2 What Is a Communication Interface?

A communication interface is a CP card, Industrial Ethernet card, integrated PROFIBUS or PROFINET interface on a Siemens Box, Rack, or Panel PC, or any card or service supported by SIMATIC NET for the purpose of communication. Communication interfaces enable communication between WinLC RTX and STEP 7 or other S7 applications.

What Is a PC Station Communication Interface?

With SIMATIC NET, you can configure Industrial Ethernet or other communication interfaces in the PC Station. You can use these communication interfaces for S7 communication, but they cannot be used to communicate with distributed I/O.

See also

What Is a Submodule? (Page 35)

3.2.3 What Is an Index?

An index is a numbered slot in the virtual rack of the PC station. The PC Station (Page 29) provides slots for WinLC RTX and the SIMATIC components of a PC-based automation solution. The following list shows some (but not all) of the typical SIMATIC components that can occupy an index:

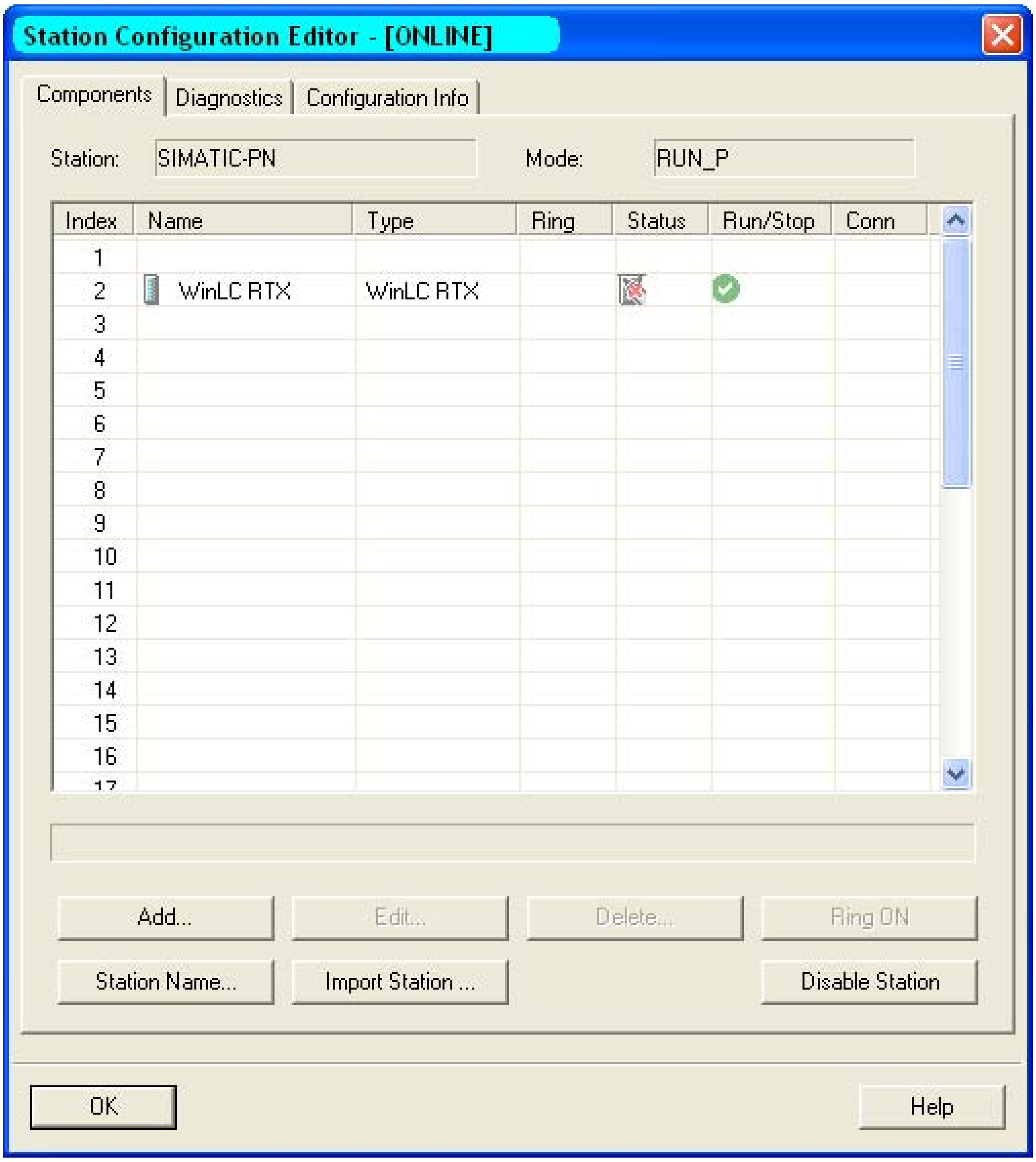
- CP card(s) (requires installation of SIMATIC NET)
- SIMATIC HMI
- SIMATIC NET OPC Server (requires installation of SIMATIC NET)
- CPU 41x-2 PCI (WinAC Slot PLC)

Each slot in the PC station corresponds to a number or index. When you install WinLC RTX, the setup 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.



Station: SIMATIC-PN

Mode: RUN_P

Index	Name	Type	Ring	Status	Run/Stop	Conn
1						
2	WinLC RTX	WinLC RTX			✓	
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

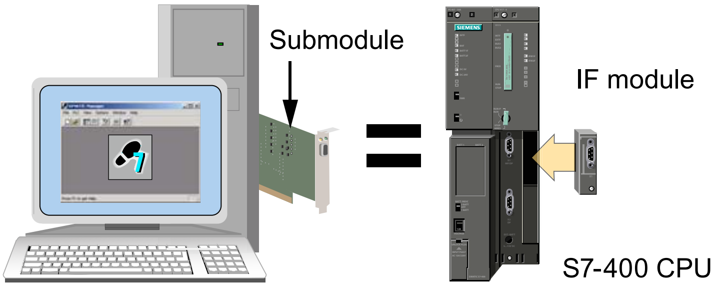
3.2.4 What Is a Submodule?

Definition

A submodule is a configured communication interface (Page 33) 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 distributed I/O devices on a PROFIBUS-DP or PROFINET I/O network, you must designate a communication interface as a submodule of the controller (Page 38). With this submodule approach, WinLC RTX has full control over the distributed 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 (Page 37).

Configuring a communication interface (Page 33) as a submodule of WinLC RTX is like installing an IF module into a slot of an S7-400 CPU:



If you use a CP 5611/21 card or an integrated CP 5611/21 communication interface as a submodule of WinLC RTX, note that you can insert only one as a submodule. Likewise you can only insert one Ethernet submodule (CP 1604, CP 1616, or IE General). You can, however, configure up to four CP 5613 communication interfaces, the total number of submodules that WinLC RTX supports.

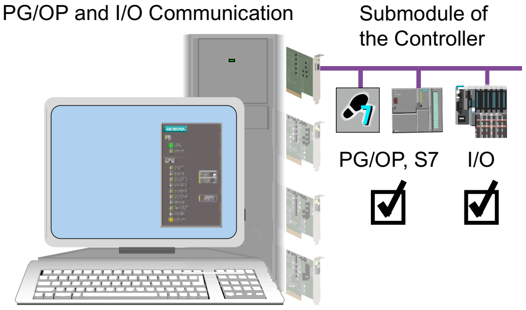
In order for a submodule to be used for SIMATIC communications with an application other than WinLC RTX (on the PC station), the second application must be a configured part of the PC station.

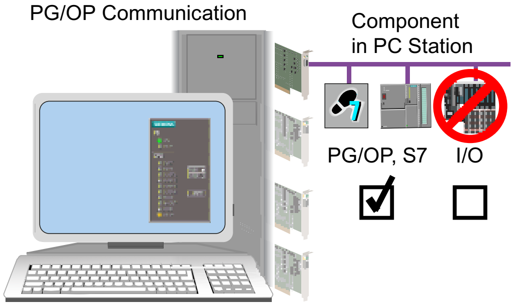
Note

Configuring a communication interface as a component of the PC Station requires the installation of SIMATIC NET. As a component of the PC station, you can use the communication interface only for SIMATIC communications with STEP 7, SIMATIC HMI, or other SIMATIC controllers. For example, you can download a program from STEP 7 to WinLC RTX. A communication interface configured as a component of the PC Station cannot be used for WinLC RTX communications with distributed I/O.

The comparison below shows the difference between a communication interface (in this case a CP card) as a submodule of WinLC RTX and as a component of the PC station:

WinLC RTX Submodule Communication	PC Station Communication
With a communication interface configured as a submodule, WinLC RTX can communicate with both STEP 7 on a remote computer (using PG/OP communication) and with the distributed 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 distributed I/O.





Communication Interfaces That You Can Configure as WinLC RTX Submodules

- CP 5613 V3 or CP 5613 V6 or later
- CP 5613 A2
- CP 5611 A2
- CP 5621
- SIEMENS PC with integrated CP 5611 PROFIBUS interface: ASPC2 STEP E2 or ASPC2 STEP R ASIC
- CP 1616, hardware revision 8 or later
- CP 1604, hardware revision 7 or later
- S7-mEC CP1616/ERTEC400_EC integrated interface
- SIMATIC PC 427B/477B integrated CP 1616 interface
- SIMATIC PC 627B/677B integrated CP 1616 interface
- SIMATIC Microbox PC 427B / Panel PC 477B integrated Intel PRO/1000 PL interface
- SIMATIC Box PC 627B / Panel PC 677B integrated Intel PRO/1000 PL interface
- SIMATIC Rack PC 847B integrated Intel PRO/1000 PL interface
- Intel PRO/1000 GT (PCI), Intel 82541PI chipset
- Intel PRO/1000 PL (integrated), Intel 82573L chipset

See also

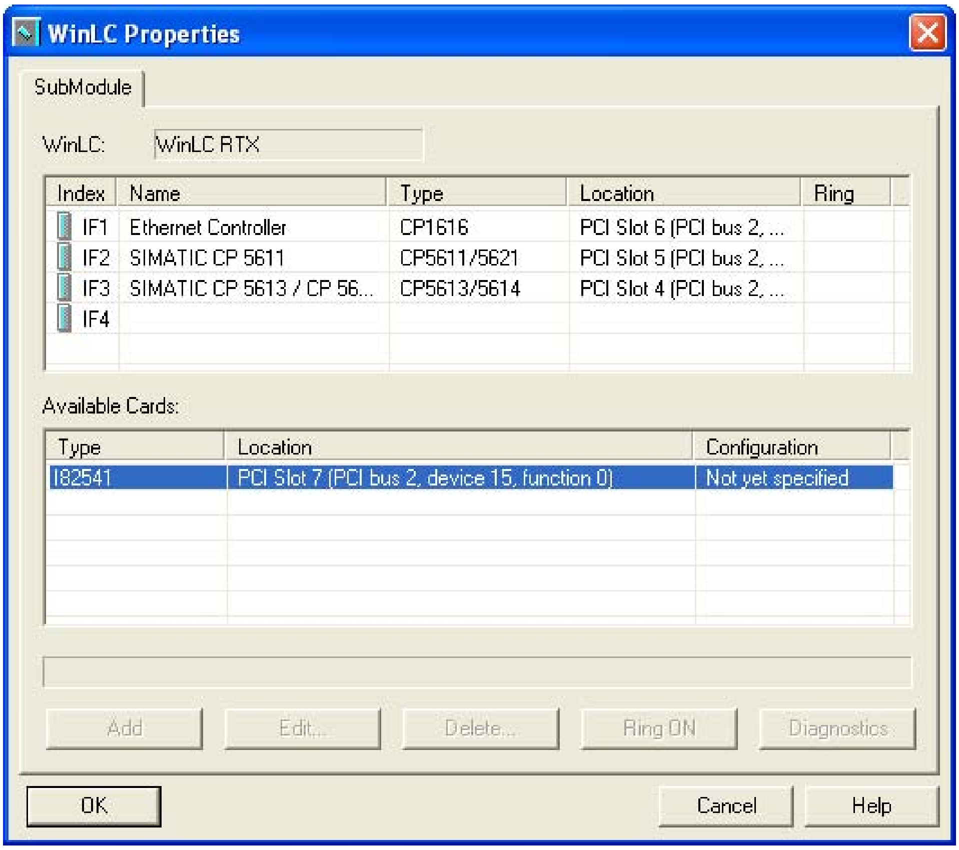
What Is a PC Station? (Page 29)

3.2.5 What Is an IF Slot?

Definition

WinLC RTX provides four interface (IF) slots for designating communication interfaces (Page 33) as submodules (Page 35). 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.

For WinLC RTX to communicate with I/O, you must configure at least one communication interface (Page 33) 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:



WinLC Properties

SubModule

WinLC: WinLC RTX

Index	Name	Type	Location	Ring
IF1	Ethernet Controller	CP1616	PCI Slot 6 (PCI bus 2, ...	
IF2	SIMATIC CP 5611	CP5611/5621	PCI Slot 5 (PCI bus 2, ...	
IF3	SIMATIC CP 5613 / CP 56...	CP5613/5614	PCI Slot 4 (PCI bus 2, ...	
IF4				

Available Cards:

Type	Location	Configuration
I82541	PCI Slot 7 (PCI bus 2, device 15, function 0)	Not yet specified

If you use a CP 5611/21 card or an integrated CP 5611/21 communication interface as a submodule of WinLC RTX, note that you can insert only one as a submodule. Likewise you can only insert one Ethernet submodule (CP 1604, CP 1616, I82541 or integrated CP1616). You can, however, configure any number of CP 5613 communication interfaces up to the total of four submodules that WinLC RTX supports.

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 (Page 38).

3.3 Configuring Communication Interfaces

3.3.1 Designating a Communication Interface as a Submodule

You configure a communication interface (Page 33) as a submodule of WinLC RTX by inserting it into an IF slot (Page 37) of the controller. Submodule communication interfaces enable WinLC RTX to communicate with distributed I/O as well as STEP 7 or other S7 applications.

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

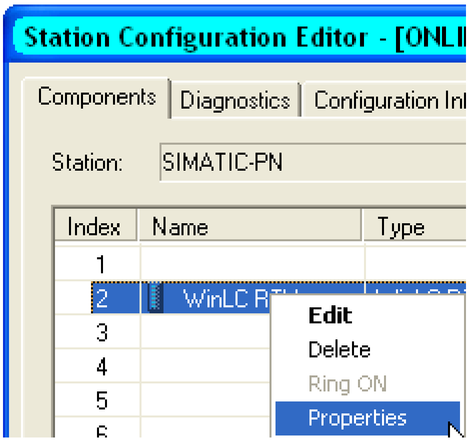
You must also configure WinLC RTX (Page 47), the submodules, and all other components of the PC station in STEP 7.

Procedure

To configure your communication Interface as a submodule (Page 35), 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.

   Right-click WinLC RTX and select Properties from the context menu to display the WinLC Properties dialog.

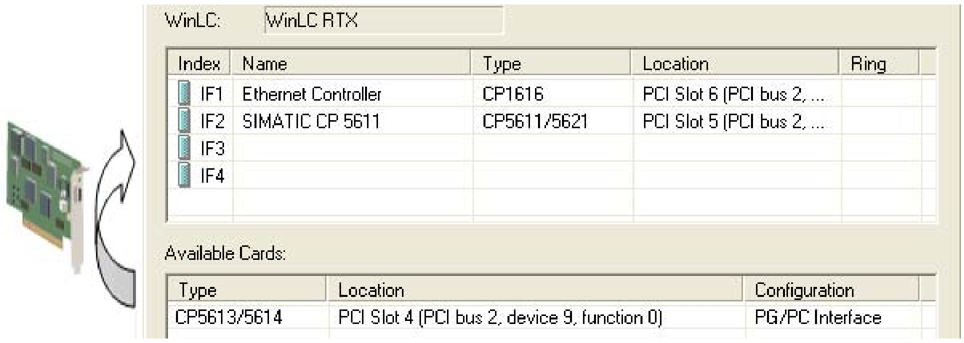


Station: SIMATIC-PN

   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 a CP1616 card and a CP 5611/21 card already configured in IF Slots 1 and 2 and a CP5613/CP5614 card being added as a submodule to IF Slot 3.

2. In the lower panel, select the communication interface that you want to configure as a submodule. (An integrated PROFIBUS or PROFINET interface is represented as a CP card with a location of "System Board.")

   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.



WinLC: WinLC RTX

Index	Name	Type	Location	Ring
IF1	Ethernet Controller	CP1616	PCI Slot 6 (PCI bus 2, ...	
IF2	SIMATIC CP 5611	CP5611/5621	PCI Slot 5 (PCI bus 2, ...	
IF3				
IF4				

Available Cards:

Type	Location	Configuration
CP5613/5614	PCI Slot 4 (PCI bus 2, device 9, function 0)	PG/PC Interface

   For multiple cards, repeat the above steps as needed.

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

Result

You have configured a submodule communication interface.

Note

WinLC RTX supports a maximum of one CP 5611/21 card or integrated CP 5611/21 interface as a submodule, a maximum of one PROFINET interface (CP 1616, CP 1604, or other) as a submodule and a maximum of four total submodules. Any number of the four submodules can be CP 5613 communication interfaces. 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 communication 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.

WinLC RTX Response to Submodule Changes

WinLC RTX can detect when a configured submodule is no longer accessible, for example, when it has been physically removed from the computer or it has failed.

Prior to the WinAC RTX 2005 SP2 release, WinLC RTX deleted the STEP 7 user program and configuration in this case. With the WinAC RTX 2005 SP2 and later releases, WinLC RTX informs you of the detected change and continues to operate without that submodule; however, you cannot set the controller to RUN mode. The diagnostic buffer also includes a "STOP caused by I/O error", which indicates the deleted or failed submodule.

The next time you attempt to access WinLC Properties, WinLC RTX informs you that a submodule is no longer accessible and prompts you to confirm the removal of that submodule. If you click OK, WinLC RTX deletes that submodule from WinLC Properties. If you cancel, WinLC RTX leaves the submodule configured and retains the current STEP 7 user program and configuration.

3.3.2 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 a CP 5611/21 card.

Procedure

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.

Result

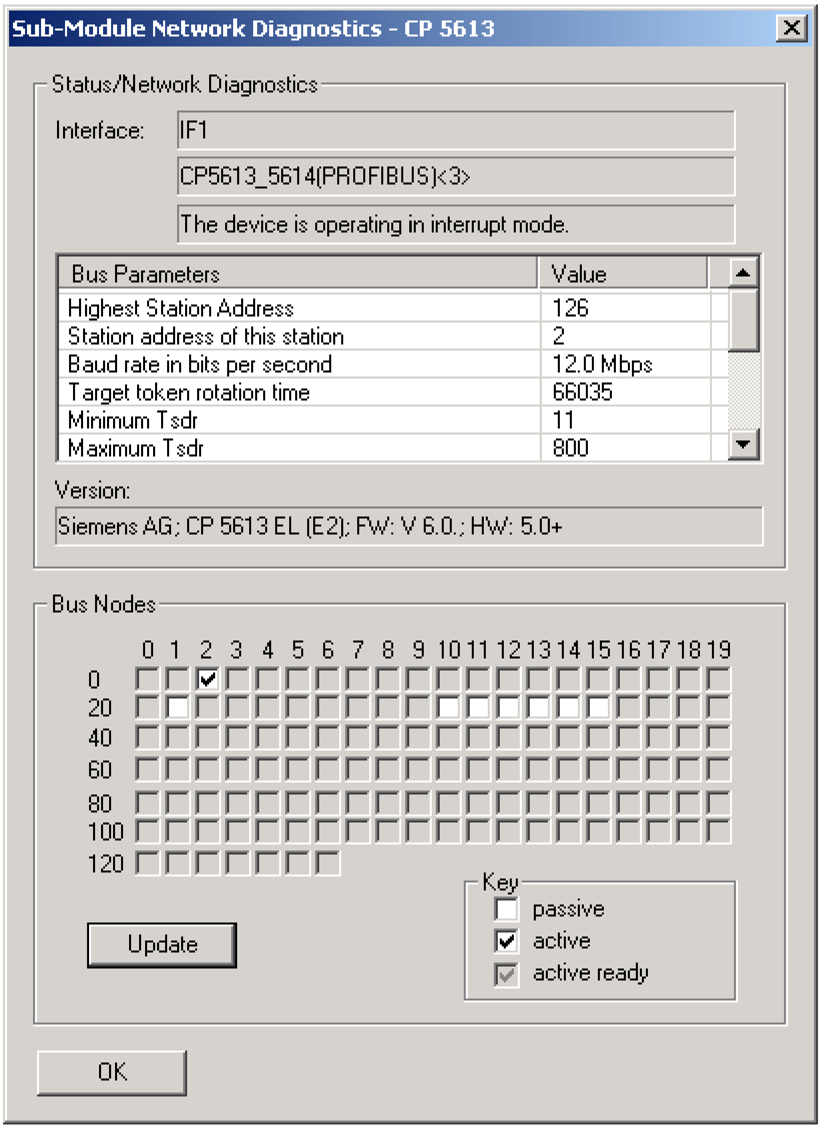
The LEDs on the CP card at the back of your computer flash in an alternating pattern so you can verify that you have configured the correct CP card. The computer also emits an audible beep if the CP card is functioning.

Ending the Test

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

3.3.3 Viewing PROFIBUS Submodule Diagnostics

You can view communication information for a PROFIBUS submodule. The Submodule Network Diagnostics dialog displays the current version of the selected CP card and the bus parameters. This dialog also displays all of the nodes on the communication network, and the status of each node.



Sub-Module Network Diagnostics - CP 5613

Status/Network Diagnostics

Interface: IF1

CP5613_5614(PROFIBUS)<3>

The device is operating in interrupt mode.

Bus Parameters	Value
Highest Station Address	126
Station address of this station	2
Baud rate in bits per second	12.0 Mbps
Target token rotation time	66035
Minimum Tsdr	11
Maximum Tsdr	800

Version:

Siemens AG; CP 5613 EL (E2); FW: V 6.0.; HW: 5.0+

Bus Nodes

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	□	□	☑	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
20	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
40	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
60	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
80	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
100	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
120	□	□	□	□	□	□	□													

Key
□ passive
☑ active
☑ active ready

Procedure

To view submodule diagnostics, follow these steps:

1. From the Station Configuration Editor, double-click WinLC RTX to open the WinLC Properties dialog.
2. Select an interface slot (IF slot) that a communication interface occupies.
3. Click the Diagnostics button.
4. From the Submodule Network Diagnostics dialog, click the Update button. This action builds the node status display. WinLC RTX does not build this display until you click the Update button. Querying each node puts an additional load on the communication network.

Result

You are able to monitor diagnostics for a PROFIBUS submodule and view the status of each node on the subnet.

Interrupt Mode for a Communication Interface

For some operations, such as isochronous mode, the communication interface (Page 33) must operate in interrupt mode. You can improve the performance of the communication interface by changing the IRQ settings. See the following topic: Improving the Performance of a Communication Interface (Page 197) in the Troubleshooting section of the Reference Information.

Note

Submodule diagnostics are available for CP 5613 and CP 5611/21 cards, including integrated PROFIBUS interfaces on Siemens PCs. Submodule diagnostics are not available for CP 1604 and CP 1616 cards, including integrated PROFINET interfaces on Siemens PCs.

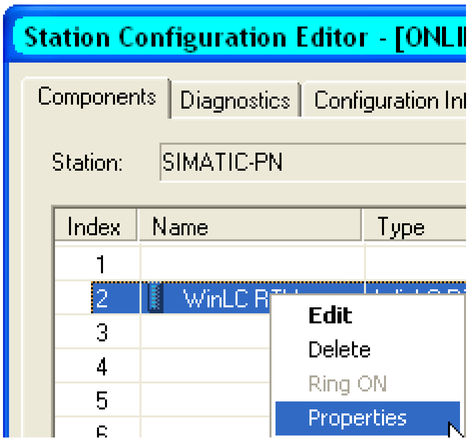
3.3.4 Removing a Submodule

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

Procedure

To remove a communication 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.

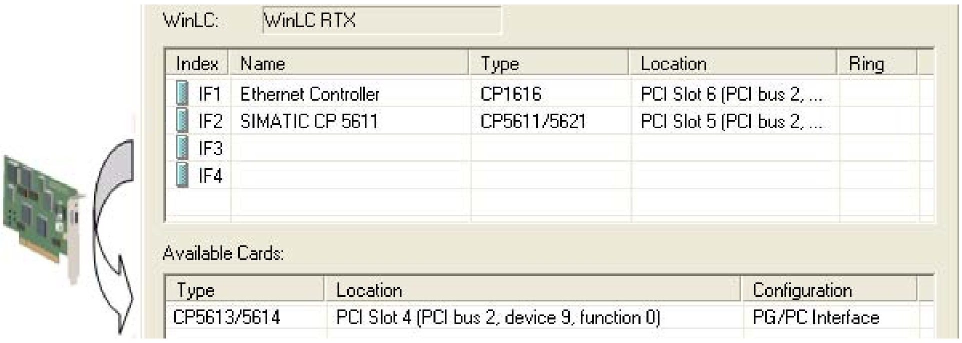


Station: SIMATIC-PN

Index	Name	Type
1		
2	WinLC R	
3		
4		
5		
6		

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 a CP1616 card and a CP 5611/21 card in IF Slots 1 and 2.

3. In the upper panel, select the communication 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.



WinLC: WinLC RTX

Index	Name	Type	Location	Ring
IF1	Ethernet Controller	CP1616	PCI Slot 6 (PCI bus 2, ...	
IF2	SIMATIC CP 5611	CP5611/5621	PCI Slot 5 (PCI bus 2, ...	
IF3				
IF4				

Available Cards:

Type	Location	Configuration
CP5613/5614	PCI Slot 4 (PCI bus 2, device 9, function 0)	PG/PC Interface

5. Click OK on the WinLC Properties dialog.
6. Restart the computer.

Result

WinLC RTX removes the communication interface from the configured submodules and returns it to the list of available cards.

The new configuration becomes effective after the next restart of the computer.

Note

Modifying the configuration of a communication interface

After removing a communication interface either from an IF slot of WinLC RTX or from a slot of the PC Station you have to restart the computer for the change to take effect and for the communication interface to become available for a new configuration.

3.4 Configuring the Controller in STEP 7

3.4.1 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 through any of the following interfaces:

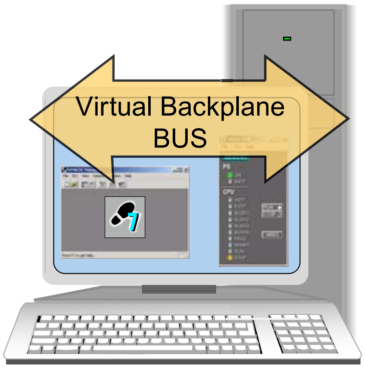
- Virtual backplane bus to STEP 7 on the same computer as the controller
- Submodule (Page 35) communications interface (Page 33) to STEP 7 on a different computer
- PC Station (Page 29) communications interface to STEP 7 on a different computer

Note

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

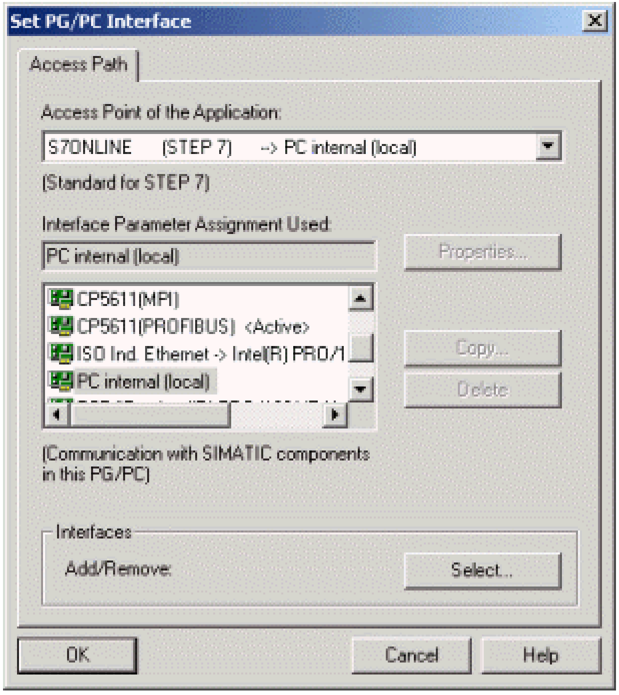
Connecting STEP 7 to the Controller on the Same Computer

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, follow these steps:

7. Open the Set PG/PC Interface dialog.
8. Select the PC internal access point:



Access Point of the Application:
S7ONLINE (STEP 7) --> PC internal (local)
(Standard for STEP 7)

Interface Parameter Assignment Used:
PC internal (local)

- CP5611(MPI)
- CP5611(PROFIBUS) <Active>
- ISO Ind. Ethernet -> Intel(R) PRO/1
- PC internal (local)

(Communication with SIMATIC components in this PG/PC)

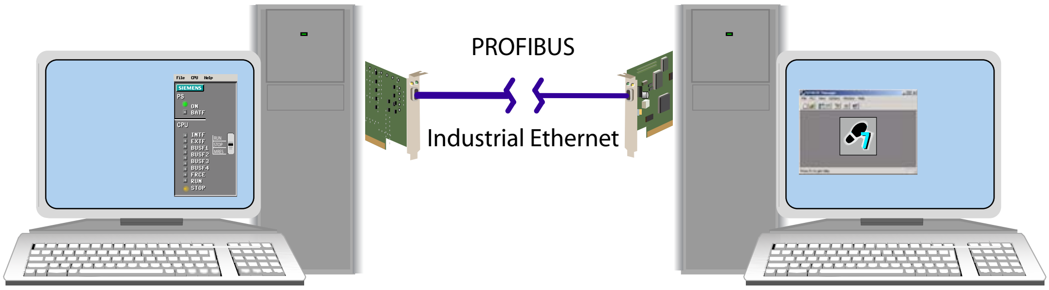
Add/Remove:

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.

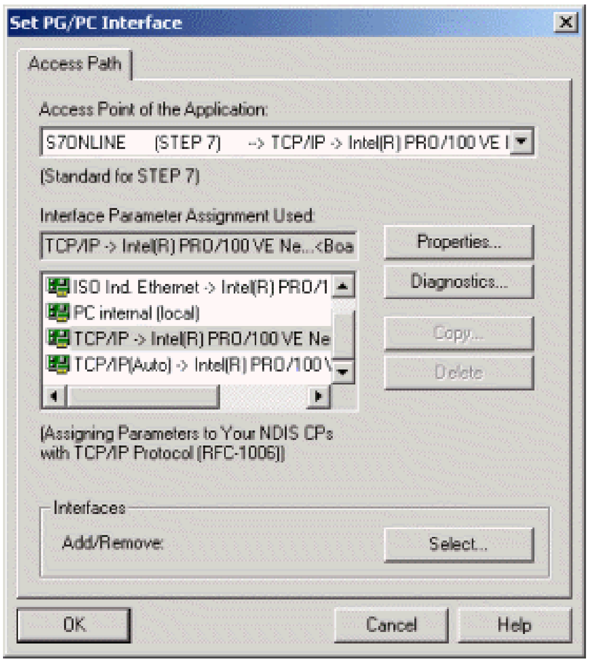
The following communications types are supported:

- PROFIBUS: for a CP 5613, CP 5611/21 or integrated PROFIBUS interface configured as a submodule
- PROFINET: for a CP 1616, CP 1604, IE General or integrated PROFINET interface configured as a submodule
- Industrial Ethernet: for an IE card configured in the PC Station



To configure communications between the controller and STEP 7 on a different computer or programming device, follow these steps:

1. Open the Set PG/PC Interface dialog
2. 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:



Set PG/PC Interface

Access Path

Access Point of the Application:
S7ONLINE (STEP 7) --> TCP/IP -> Intel(R) PRO/100 VE I
(Standard for STEP 7)

Interface Parameter Assignment Used:
TCP/IP -> Intel(R) PRO/100 VE Ne...<Boa

ISO Ind. Ethernet -> Intel(R) PRO/1
PC internal (local)
TCP/IP -> Intel(R) PRO/100 VE Ne
TCP/IP(Auto) -> Intel(R) PRO/100 \

(Assigning Parameters to Your NDIS CPs with TCP/IP Protocol (RFC-1006))

Interfaces
Add/Remove:

Alternative connection method

You can connect STEP 7 to a WinAC RTX on a different computer though the PC Internal interface if the following conditions are met:

- WinAC RTX is installed on the computer where STEP 7 is installed.
- The computer with STEP 7 and a local WinAC RTX installation is connected to a network to which WinAC RTX on the other computer is connected

If these conditions are met, you can set the S7ONLINE access point to PC Internal on the computer with STEP 7 and connect to the remote WinAC RTX through the PC Internal interface. This connection method is an alternative to connecting STEP 7 to a remote WinAC RTX through a PROFIBUS, PROFINET or Industrial Ethernet interface, as described earlier.

3.4.2 Configuring the Hardware in STEP 7

You configure the STEP 7 project for a PC station (Page 29) 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

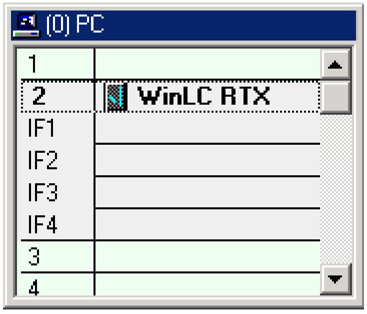
To create the project and PC station, follow these steps:

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 (Page 50) 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.

Using the STEP 7 Hardware Configuration Application

To configure the PC-based controller for the PC Station, follow these steps:

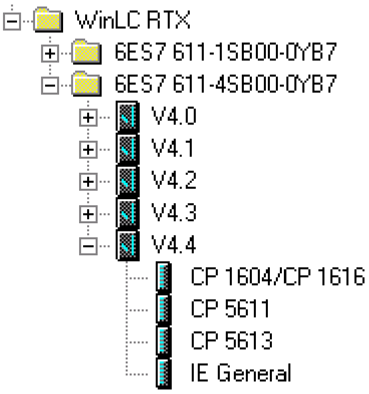
1. Open the PC Station folder in the project and double-click the Configuration icon to invoke the STEP 7 Hardware Configuration application.
2. Navigate to your specific controller under SIMATIC PC Station.
3. Drag the controller into the same index it occupies in the Station Configuration Editor on the target computer.



(0) PC

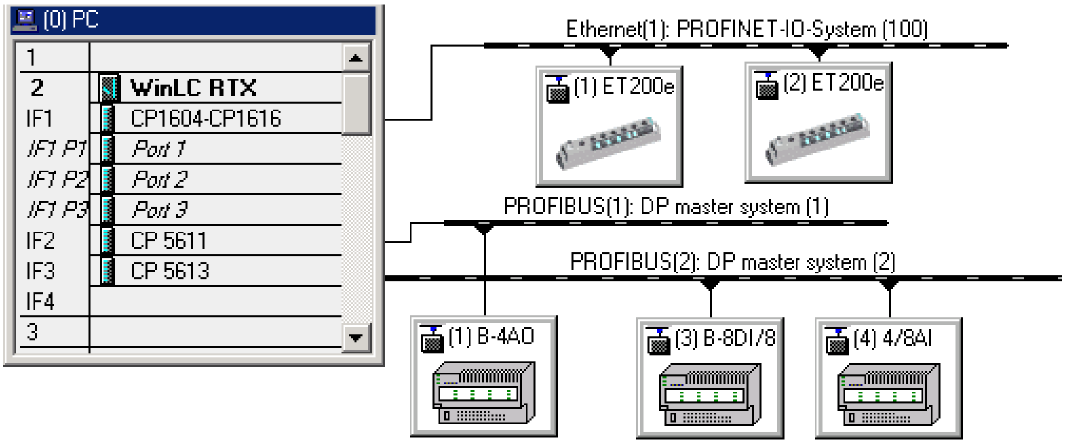
1	
2	WinLC RTX
IF1	
IF2	
IF3	
IF4	
3	
4	

4. Verify that the name of the controller matches the name of the controller configured in the Station Configuration Editor.
5. Drag the submodule communication interfaces from the Hardware Catalog into interface slots (IF slots (Page 37)) for the WinLC RTX controller. The WinLC RTX folder in the Hardware Catalog lists the available choices. (For integrated PROFIBUS interfaces on SIEMENS PCs, select a CP5611/21 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 5611/21 card or integrated CP 5611/21 PROFIBUS interface as a submodule of WinLC, note that you can insert only one as a submodule. Likewise you can only insert one CP 1604, CP 1616, or Industrial Ethernet interface as a submodule.



Note: Standard Ethernet network cards must be configured in the STEP 7 Hardware Configuration application using IE General. In the Station Configuration Editor they appear with their normal name, for example "Intel PRO/1000 GT (PCI) ".

6. Configure the distributed I/O for each of the submodule networks:



Additional Hardware Configuration Options

Note

You must have installed SIMATIC NET to configure communication interfaces in the PC Station with the Station Configuration Editor. You do not need SIMATIC NET to configure CP cards as submodules of WinLC RTX.

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

1. Insert any CP cards that your application requires into the PC Station.
2. Insert any HMI devices, for example, text displays or operator panels.
3. Configure WinLC RTX for peer-to-peer communications:
   – Select the controller name in the SIMATIC Manager.
   – Double-click the Connections icon in the right-hand pane.
   – Use NetPro to describe the network.

Result

You have configured WinLC RTX in STEP 7 and 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 to become unstable, possibly causing damage to equipment and/or injury to personnel.

Although STEP 7 and WinLC 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 drive space and available RAM. 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.

3.4.3 Correcting invalid Characters prior to STEP 7 V5.3 SP1

You can use STEP 7 to create a name for the controller and to download the configuration with the new name to the controller. However, some characters that might have been used in controller names in versions of STEP 7 prior to V5.3 SP1 are invalid. Change these controller names to valid controller names prior to downloading.

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.

Invalid Charaters

The following table describes invalid characters for controller names prior to STEP 7 V5.3 SP1 or SP2:

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. **Valid:** Pump-1: Using a hyphen in the middle of the name is valid. Pump1-: Using a hyphen at the end of the name is valid. **Invalid:** -Pump1: Using a hyphen at the beginning of the name is invalid. -: Using a hyphen as a one-character name is invalid.

Procedure

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).

Result

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.

3.5 Verifying the Configuration

A complete configuration for a PC-based automation project includes a configuration in the Station Configuration Editor and WinLC Properties (Page 38) that matches the configuration in STEP 7 (Page 47).

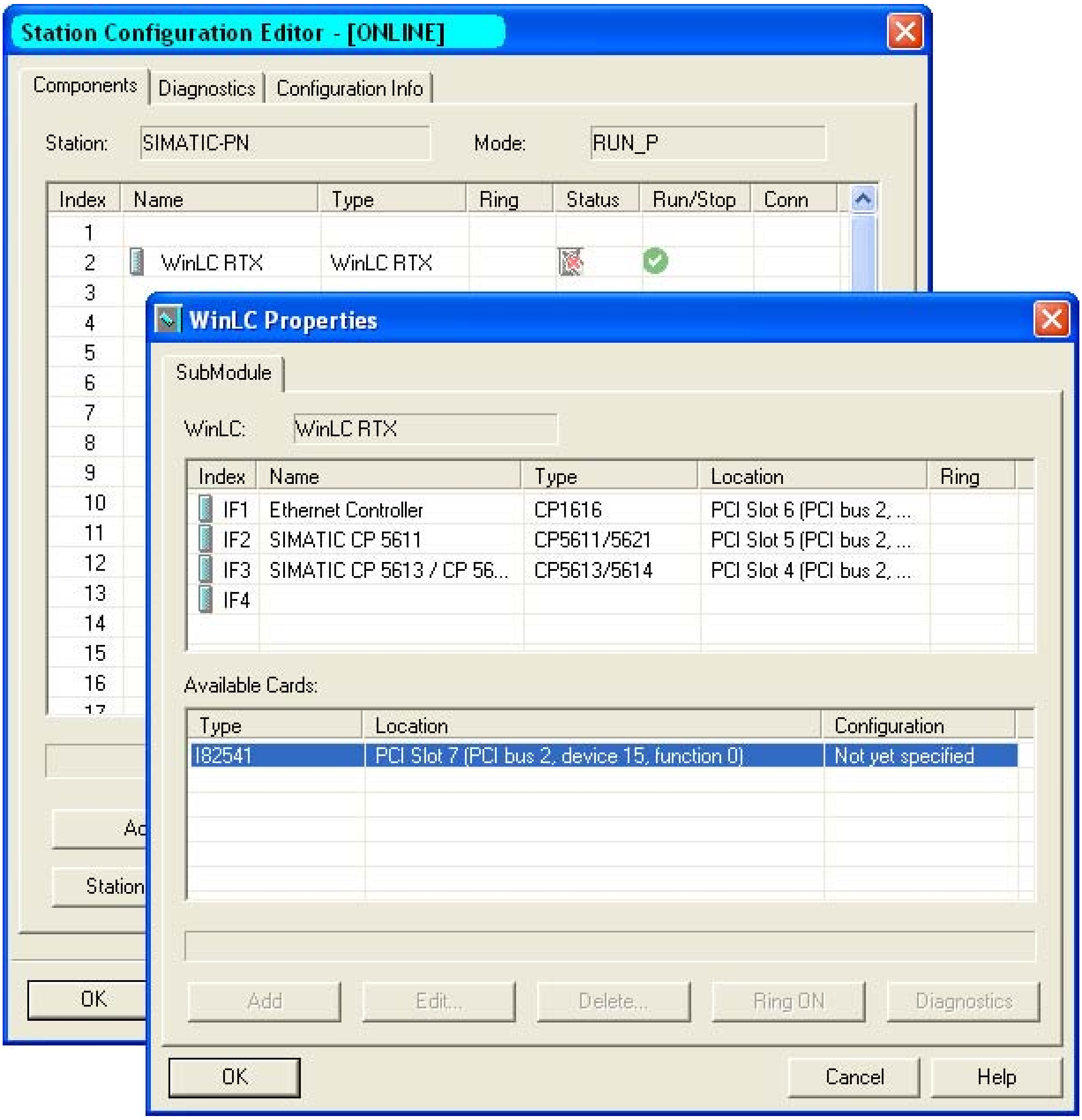
Example PC Station Configuration

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

- WinLC RTX controller in index (Page 33) 2 of the PC station (Page 29)
- CP1616 card configured as a WinLC RTX submodule (Page 35) in IF slot (Page 37) 1 connected to PROFINET IO
- CP 5611/21 card configured as a WinLC RTX submodule in IF slot 2 connected to PROFIBUS-DP I/O
- CP5613/CP5614 card configured as WinLC RTX submodule in IF slot 3 connected to PROFIBUS-DP I/O
- STEP 7 (Page 44) on same computer as WinLC RTX

Station Configuration Editor and WinLC Properties

The Station Configuration Editor and WinLC Properties dialogs show the configuration of the project:



Station: SIMATIC-PN Mode: RUN_P

Index	Name	Type	Ring	Status	Run/Stop	Conn
1						
2	WinLC RTX	WinLC RTX			✓	
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

WinLC: WinLC RTX

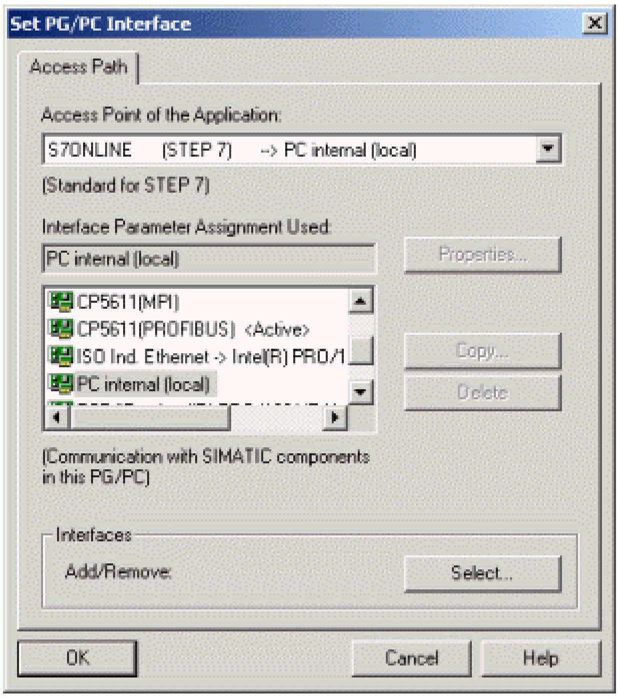
Index	Name	Type	Location	Ring
IF1	Ethernet Controller	CP1616	PCI Slot 6 (PCI bus 2, ...	
IF2	SIMATIC CP 5611	CP5611/5621	PCI Slot 5 (PCI bus 2, ...	
IF3	SIMATIC CP 5613 / CP 56...	CP5613/5614	PCI Slot 4 (PCI bus 2, ...	
IF4				

Available Cards:

Type	Location	Configuration
I82541	PCI Slot 7 (PCI bus 2, device 15, function 0)	Not yet specified

STEP 7 PG/PC Interface

The STEP 7 PG/PC interface shows the PC internal access point:



Access Point of the Application:
S7ONLINE (STEP 7) --> PC internal (local)
(Standard for STEP 7)

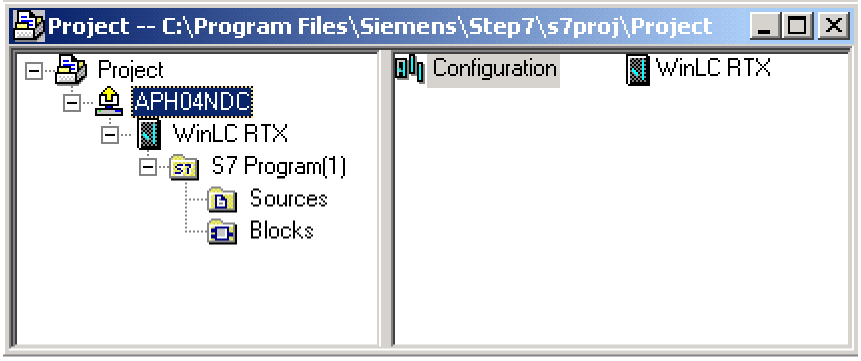
Interface Parameter Assignment Used:
PC internal (local)

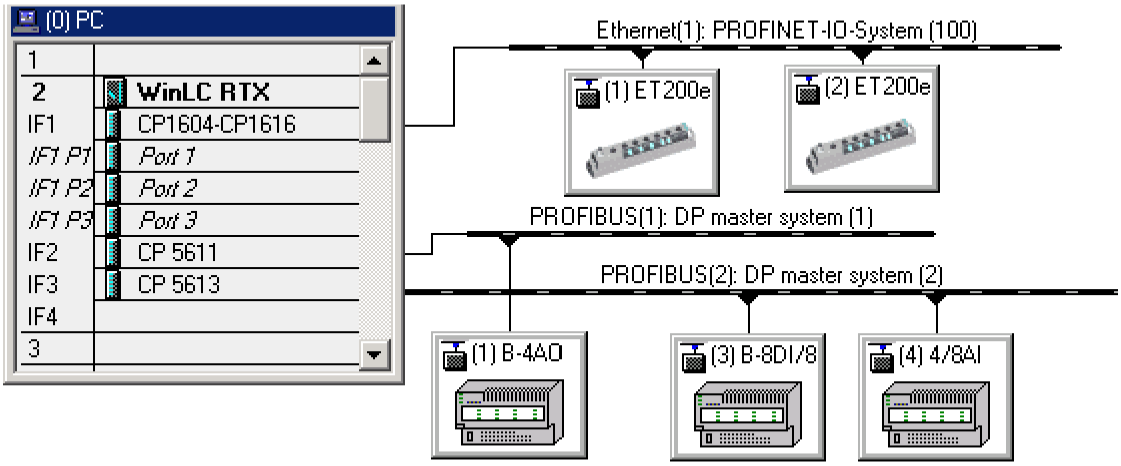
- CP5611(MPI)
- CP5611(PROFIBUS) <Active>
- ISO Ind. Ethernet -> Intel(R) PRO/1
- PC internal (local)

(Communication with SIMATIC components in this PG/PC)

STEP 7 Hardware Configuration

The STEP 7 hardware configuration shows WinLC RTX in slot 2 of the hardware configuration and the three submodules of WInLC RTX with I/O:




(0) PC

1	
2	**WinLC RTX**
IF1	CP1604-CP1616
IF1 P1	*Port 1*
IF1 P2	*Port 2*
IF1 P3	*Port 3*
IF2	CP 5611
IF3	CP 5613
IF4	
3	

Ethernet(1): PROFINET-IO-System (100)
(1) ET200e
(2) ET200e
PROFIBUS(1): DP master system (1)
PROFIBUS(2): DP master system (2)
(1) B-4AO
(3) B-8DI/8
(4) 4/8AI

Controller Operations 4

4.1 Starting and Shutting Down the Controller

The controller operates independently from the controller panel. The controller panel is the visible interface to the controller. The controller can be in operation or shut down regardless of whether the panel is visible.

- Opening the panel starts the controller if it is not already in operation.
- Closing the panel (Page 69) (menu command **File > Exit**) does **not** shut down the controller.
- Shutting down the controller does **not** close the panel.

When the controller is in operation, your Windows taskbar displays a WinLC RTX icon regardless of whether the controller panel is open. The icon will be surrounded by a yellow border when the controller is in STOP mode, and by a green border when the controller is in RUN mode.

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

- Selecting the Autostart option (Page 88)
- Configuring the controller for start at PC boot (Page 97)

Starting WinLC RTX

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

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

  Note

  If the WinLC RTX menu command or icon is missing, then WinLC RTX has been deleted from the Station Configuration Editor. If this is the case, insert WinLC RTX into slot 2 of the Station Configuration Editor. The menu command and desktop icon will then be available.

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

Result: The controller panel opens and starts the WinLC RTX controller.

Note

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

Shutting Down WinLC RTX

Select the CPU > Shut Down Controller menu command to shut down the WinLC 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 (Page 86).

4.2 Changing the Operating Mode of the Controller

The controller panel (Page 12) provides a mode selector that allows you to change the operating mode of the controller. Set the mode selector 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 positions on the controller panel correspond to the mode selector 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 on the controller panel functions like the manual mode selector 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 (Page 60) show the current operating mode of the controller. If the status indicator shows a different operating mode than the mode selector 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. Note that if you used the STOP selector on the controller panel to put the controller in STOP mode, you cannot change WinLC RTX to RUN mode from STEP 7.

Allowed and Prohibited Actions for each Operating Mode

Table 4-1 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	Allowed: • Uploading a program from the controller to your computer • Downloading a program to the controller • Downloading individual blocks to the controller • Using STEP 7 to modify program variables and to change the operating mode of the controller • Performing a memory reset from either the controller panel or STEP 7 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. Not Allowed: • Archiving and restoring a STEP 7 user program
STOP	Allowed: • Uploading a program from the controller to your computer or programming device • Downloading a program or individual blocks to the controller • Using STEP 7 to modify program variables • Performing a memory reset from either the controller panel or STEP 7 • Archiving and restoring a STEP 7 user program Not Allowed: • Using STEP 7 to change the operating mode to RUN if you had used the STOP selector on the controller panel to put WinLC RTX in STOP mode

4.3 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 (Page 12)
- 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

4.4 Using the Status Indicators

The status indicators on the controller panel (Page 12) 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 STEP 7 user program reaches a breakpoint 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.

Table 4-2 The table below describes the different status indicators for the controller 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.
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, download of a STEP 7 program with configured CP cards to a WinLC RTX with no CP cards or faulty CP cards, and I/O faults. This indicator lights up (solid) together with a BUSF indicator to show that a CP is defective. 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 (Page 37) of the submodule (Page 35) that has a fault condition.
FRCE	This indicator is never lit. WinLC RTX does not support force jobs.
MAINT	This indicator lights up (solid) when a PROFINET IO Controller or IO Device requires maintenance.
RUN STOP	Lights up (solid) to show the operating mode (RUN or STOP). When RUN is flashing and STOP is lighted (solid), the STEP 7 user program has reached a breakpoint. (RUN light blinks with 0.5 Hz.) **Note:** 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 (Page 57) with STEP 7 causes the status indicators to change, but the mode selector switch does not change.

See the topic Troubleshooting Network Problems (Page 196) for additional information about LED behavior.

Flashing Indicators

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

RUN Indicator	STOP Indicator	Description
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 (Page 98) 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 (Page 59) 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 (Page 59) the memory.
4. Use STEP 7 to download the STEP 7 user program and system configuration, or to restore an archived STEP 7 user program.

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

4.5 Using the Tuning Panel

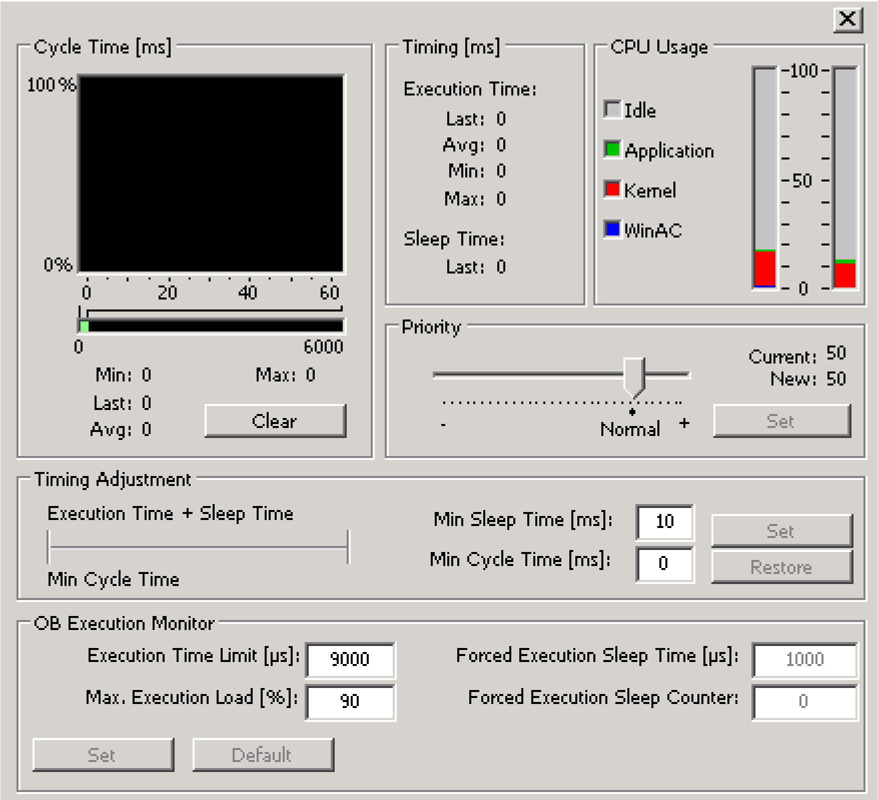
Tuning Panel Command (CPU Menu)

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 (Page 143) , 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. Because the tuning panel causes an additional load on the computer resources, do not leave the tuning panel open during normal operation of WinLC.

To open or close the tuning panel, select the CPU > Tuning Panel menu command. WinLC RTX opens the tuning panel, as shown below.



Cycle Time [ms]

Min: 0 Max: 0
Last: 0
Avg: 0

Timing [ms]

Execution Time:
Last: 0
Avg: 0
Min: 0
Max: 0

Sleep Time:
Last: 0

CPU Usage

☐ Idle
🟩 Application
🟥 Kernel
🟦 WinAC

Priority

Current: 50
New: 50

Timing Adjustment

Min Sleep Time [ms]: 10
Min Cycle Time [ms]: 0

OB Execution Monitor

Execution Time Limit [µs]: 9000
Max. Execution Load [%]: 90
Forced Execution Sleep Time [µs]: 1000
Forced Execution Sleep Counter: 0

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 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.

⚠ CAUTION

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.

Functional Areas of the Tuning Panel

The tuning panel contains the following functional areas:

Area	Description
Cycle Time (Page 143)	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.
Timing (Page 159)	This read-only field displays the following information about the scan cycle: Execution Time displays the execution time for the last (most current) scan, the average cycle time, the shortest (minimum) cycle time, and the longest (maximum) cycle time. Sleep Time displays the amount of sleep time for the last (most current) scan.
CPU Usage (Page 164)	This area shows the percentage of the computer's CPU that is idle, and the percentage this is used by applications, the kernel, or WnAC.
Priority (Page 152)	Use this slider to set the priority level for the execution of WinLC RTX relative to other RTX applications running on your computer. 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. 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.

Area	Description
Timing Adjustment (Page 159)	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. 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. To ensure that the minimum cycle time controls the sleep time for the controller, you must configure the scan cycle monitoring time and minimum cycle time parameters on the Cycle/Clock Memory tab of the Properties dialog box in STEP 7. Set the minimum cycle time to a value less than the value for scan cycle monitoring time. (The default cycle time is 6 seconds.)
OB Execution Monitor (Page 164)	Use the OB execution monitor to ensure that the controller does not exceed a configurable maximum execution load for the CPU usage within a monitor interval.

See also

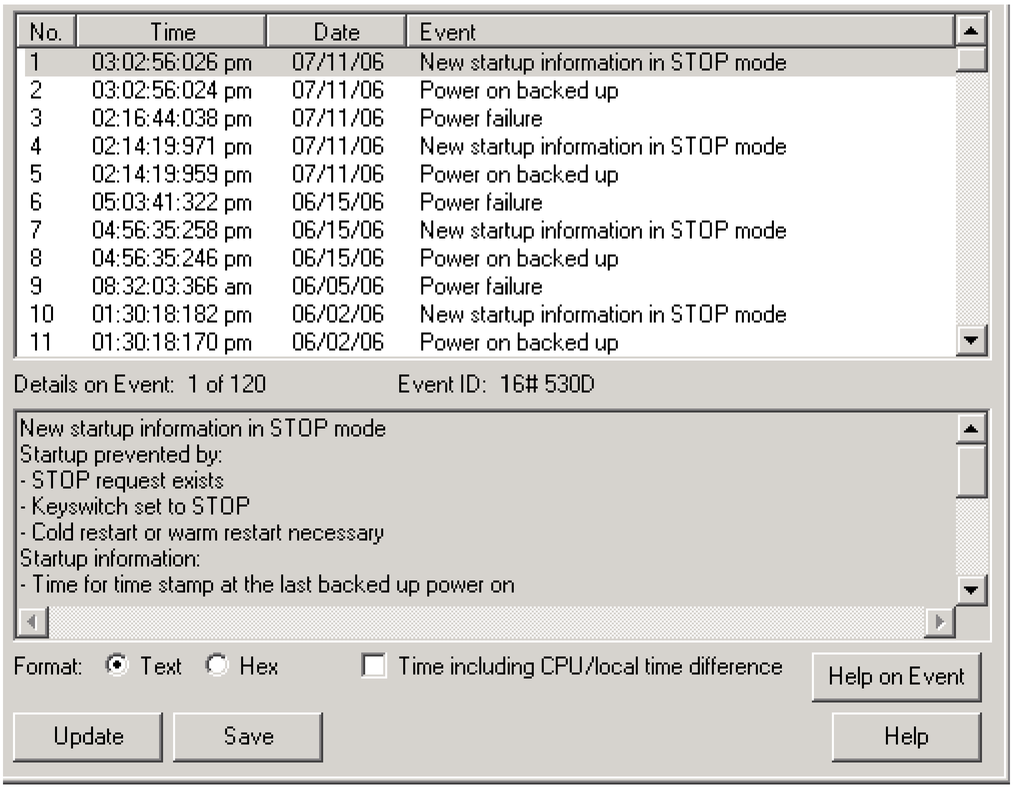
Changing the Operating Mode of the Controller (Page 57)

4.6 Using the Diagnostic Buffer

Diagnostic Buffer Command (CPU Menu)

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.



No.	Time	Date	Event
1	03:02:56:026 pm	07/11/06	New startup information in STOP mode
2	03:02:56:024 pm	07/11/06	Power on backed up
3	02:16:44:038 pm	07/11/06	Power failure
4	02:14:19:971 pm	07/11/06	New startup information in STOP mode
5	02:14:19:959 pm	07/11/06	Power on backed up
6	05:03:41:322 pm	06/15/06	Power failure
7	04:56:35:258 pm	06/15/06	New startup information in STOP mode
8	04:56:35:246 pm	06/15/06	Power on backed up
9	08:32:03:366 am	06/05/06	Power failure
10	01:30:18:182 pm	06/02/06	New startup information in STOP mode
11	01:30:18:170 pm	06/02/06	Power on backed up

Details on Event: 1 of 120   Event ID: 16# 530D

New startup information in STOP mode
Startup prevented by:
- STOP request exists
- Keyswitch set to STOP
- Cold restart or warm restart necessary
Startup information:
- Time for time stamp at the last backed up power on

Format: ● Text ○ Hex   □ Time including CPU/local time difference

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.

Selecting the Time Type

If you select the "Time including CPU/local time difference" checkbox, the diagnostic buffer applies a correction value to the time-of-day. Use this setting if the location of the diagnostic buffer reader is in a different time zone than the module. You can only select the checkbox for modules which support time-of-day status.

If you do not select the "Time including CPU/local time difference" checkbox, the diagnostic buffer displays the entries with the time of day of the module. Use this setting if this time is the same as the time at the location of the diagnostic buffer reader (the same time zone).

If you change the settings, the diagnostic buffer immediately updates the time stamps of the entries.

Updating the Diagnostic Buffer

To display the most up-to-date information in the window, select the "Update" button.

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.

4.7 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. Use STEP 7 to create a new project and to create a new configuration for WinLC RTX 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.

4.8 Closing the Controller Panel

Exit Command (File Menu)

Select the File > Exit menu command to close the control panel (Page 12).

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.

4.9 WinLC RTX Operation Following a Windows Blue Screen

4.9.1 WinLC RTX Response to a 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 (Page 33) configured as submodules (Page 35) 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 (Page 71).

You can configure both Windows and WinLC RTX to automatically restart (Page 72) 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.
- If the WinAC data storage (Page 89) is not in NVRAM, a restart of the computer followed by a restart of WinLC RTX initializes all of the program variables to their default values and empties the diagnostic buffer. If the WinAC data storage is in NVRAM, WinLC RTX can recover the retentive data when it restarts. See the topic "Available Options for WinAC Data Storage (Page 75)" for a summary of which SIMATIC PCs have NVRAM for WinAC data storage.

NOTICE

WinLC RTX cannot guarantee in all cases that it can detect a Windows Blue Screen and continue operation. Operation is only possible if the cause of the blue screen does not corrupt memory that WinLC RTX or the realtime operating system uses.

If WinLC RTX cannot detect the blue screen, it cannot call OB 84 or continue running. You must reboot your computer to recover.

If you specified NVRAM storage for the retentive data, (either a SIMATIC WinAC NV128 card or integrated PC SRAM) and an undetected Windows Blue Screen occurs, WinAC RTX will start with an unbuffered startup after the reboot. The controller panel lights up the INTF status indicator, and the diagnostic buffer contains an "unbuffered power on" error.

In actual practice, the occurrence of a blue screen is very rare, and the occurrence of a blue screen that WinLC RTX cannot detect is rarer still.

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

If a Windows Blue Screen (Page 69) 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.

4.9.3 WinLC RTX Restart Behavior after a Blue Screen

If Windows is configured to reboot automatically (Page 72) after a Windows Blue Screen, WinLC RTX starts if it is configured to start at PC boot.

Behavior when WinAC Data Storage is in NVRAM

If your computer supports NVRAM configuration (Page 75) for WinAC data storage, and you configured the WinAC data storage (Page 89) for NVRAM prior to the blue screen event, then WinLC RTX restarts with the current STEP 7 user program and uses the retentive data stored in the NVRAM , providing that WinLC RTX was able to save the retentive data when the blue screen occurred.

Behavior when WinAC Data Storage is not in NVRAM

WinLC RTX restarts with the STEP 7 user program as it was last downloaded and executes OB 100 (not OB 102) if it is present. WinLC RTX executes OB 100 with event 1382 (hex) after a blue screen, even if OB 102 "Cold Start" is configured in the STEP 7 Hardware Configuration. The diagnostic buffer shows the current/last startup type 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"

See also the topic WinLC RTX Response to a Blue Screen (Page 69).

4.9.4 Configuring Automatic Reboot for Windows

Procedure

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.

Result

The next time the computer restarts, the Windows operating system will restart.

4.10 Storing Retentive Data

4.10.1 What Information about the Controller Does WinLC RTX Store?

WinLC RTX stores the following operational information about the controller:

- Load memory: contains the system data (configuration of the controller and STEP 7 user program) and the initial values of the data blocks of the STEP 7 user program.
- State of the controller: includes the last transition of the operating mode (Page 57) (STOP, RUN, or STARTUP) for the controller and the setting for the mode selector switch (Page 57) (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 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 data blocks and system data in the Load memory area. These data blocks include the initial values for the process variables used by the STEP 7 user program.

SFC 82 (Page 82) (CREA_DBL) allows you to create new data blocks in Load memory during the execution of the STEP 7 user program. You can use SFC 84 (Page 82) (WRIT_DBL) to modify these data blocks. SFC 82 creates and stores the new data blocks 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.

State of the Controller

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

- Whenever the controller changes operating mode (Page 57) (RUN to STOP, STOP to STARTUP, or STARTUP to RUN), WinLC RTX stores the state of the controller in a filesystem to show the latest transition.
- Whenever the mode selector switch (Page 57) on the controller panel changes (STOP or RUN), WinLC RTX stores the state of the mode selector switch in a filesystem to show the latest action.

Power-Down State

The power-down state includes the following information:

- Current operating state of the controller
- Diagnostic buffer
- Retentive data

When you configure WinLC RTX in STEP 7, you specify the ranges of retentive data for the timers (T memory), counters (C memory), bit memory (M memory), and retentive data blocks (Page 89) (DBs). When you perform a normal shutdown of WinLC RTX, the controller saves this retentive data and the diagnostic buffer in the power-down state. A normal shutdown of the Windows operating system, whether by user action or UPS signal, also causes WinLC RTX to save the power-down state.

After WinLC RTX starts up the next time, it loads the power-down state. See the topic "Loading Memory Areas on Startup (Page 79)" for a description of how WinLC RTX loads memory areas at startup.

Note

If you are not using non-volatile RAM (NVRAM) for WinAC Data Storage (Page 75) , WinLC RTX can **not** store the power-down state (which includes the diagnostic buffer) when WinLC RTX terminates abnormally. An abnormal termination can occur 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 filesystem, such as following a Windows crash ("Blue Screen"). If, however, you are using either the internal SRAM of a SIMATIC Panel PC 477, or a SIMATIC WinAC NV128 card for WinAC data storage, WinLC RTX can store the retentive data when an abnormal termination (Page 85) occurs.

When Does WinLC RTX Save 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 (data blocks and initial values of the STEP 7 user program, and system data)	Downloading the STEP 7 user program from STEP 7 Calling SFC 82, SFC 83, or SFC 84 (Page 82).
State of the controller	Changing the operating mode (RUN to STOP, or STOP to RUN), either from STEP 7 or from the controller panel Normal shutdown of WinLC RTX Abnormal termination with WinAC Data Storage in NVRAM
Power-down state: • retentive T, C, M, and DB memory areas • diagnostic buffer	Normal shutdown of WinLC RTX or Abnormal termination with WinAC Data Storage in NVRAM

4.10.2 Available Options for WinAC Data Storage

The controller can store the WinAC RTX retentive data in a filesystem on your computer as well as on non-volatile RAM (NVRAM):

- Storing retentive data in a filesystem: WinAC RTX stores the retentive data at shutdown. This shutdown can be caused by a shutdown command from the controller panel or by a Windows shutdown. In order to save retentive data in a power loss situation, protect your system with an Uninterruptable Power Supply (UPS).
- Storing retentive data in NVRAM: Some SIMATIC PCs contain NVRAM that you can use for retentive data storage. With the use of NVRAM, the retentive data can be saved even in the event of a Windows Blue Screen (Page 69) event in most cases.

NVRAM Possibilities

The following types of NVRAM are available:

- 25 KByte internal static RAM (SRAM)
- 128 KByte NVRAM of an optional plug-in SIMATIC WinAC NV128 card
- 128 KByte integrated NVRAM
- 512 KByte S7-mEC, EC31-RTX NVRAM

Note

To use a SIMATIC WinAC NV128 card, plug the card into any available PCI slot when the computer is shut down. When you power on the computer, the Windows Plug-and-Play manager detects it and allocates memory for it. Whether you install WinAC RTX before or after you install the SIMATIC WinAC NV128 card, WinAC RTX will automatically detect the card and make it available for WinAC data storage.

Note

You cannot use multiple SIMATIC WinAC NV128 cards with WinLC RTX, nor can you use a SIMATIC WinAC NV128 card in conjunction with integrated NVRAM.

The following table summarizes the available types of retentive storage solutions:

PC System	Storage	Size	UPS required	Remark
Any PC	File storage	Limited only by computer's disk space	Yes	Retentive data cannot be saved to file storage in the event of a blue screen.
Panel PC 477	Integrated SRAM	25 KB	No	Maximum 6W load over USB and PC104 devices
Box PC 627 24V Panel PC 677 24V	SIMATIC WinAC NV128 card	128 KB	No	DC Power supply only
Box PC 627 230V Panel PC 677 230V Rack PC 847 Panel PC 877 PC IL43 Other SIMATIC PC	SIMATIC WinAC NV128 card	128 KB	Yes	
Microbox PC 427B Panel PC 477B Box PC 627B 24V (with PROFIBUS option) Panel PC 677B 24V	Integrated NVRAM	128 KB	No	Panel PC 477B and Microbox PC 427B: Maximum 6W load over USB and PC104 devices
S7-mEC	Integrated NVRAM	512 KB	No	

To configure the storage of WinAC RTX retentive data, use the WinAC Data Storage (Page 89) option.

NOTICE

Power loss without a shutdown of the operating system can cause filesystems of Windows XP Professional to be corrupted. For this reason, use a UPS (Page 84) to protect filesystems with these operating systems.

In addition, some SIMATIC PCs can detect a power failure and send a signal to WinLC RTX. WinLC RTX can then initiate a fast shutdown and save retentive data to NVRAM if so configured. See the topic "Retentive Data Storage following Power Loss or Blue Screen (Page 85)" for a list of the SIMATIC PCs that support the power failure signal, and a description of how WinLC RTX responds.

Systems with Windows XP embedded that use a CompactFlash file system that is protected with the Enhanced Write filter are stable against an unexpected loss of power.

4.10.3 Managing the Enhanced Write Filter

The following Windows XP embedded systems use CompactFlash cards:

- Panel PC 477
- Box PC 627
- Microbox PC 427B
- Panel PC 477B
- Box PC 627B
- S7-mEC

The Enhanced Write Filter (EWF) is a Windows XP embedded utility for protecting a CompactFlash card. CompactFlash cards allow a limited number of write accesses. When the Enhanced Write Filter is enabled, Windows XP embedded writes no data to the CompactFlash card. Instead, the file writes are kept in virtual memory. No difference is apparent to you when you view file contents. File information appears the same whether it actually resides on the CompactFlash card or in virtual memory. The difference in file storage is evident when you reboot the computer or it loses power. All data in virtual memory is lost, and the computer restarts with the file contents of the CompactFlash card.

You can manage the EWF to maintain data that must persist after a reboot. If you disable the EWF, all file writes go to the CompactFlash card. This does result in the persistence of all data after a power loss; however, it causes the most stress over time to the CompactFlash card.

NOTICE
The Enhanced Write Filter default setting of the D:\ and C:\ drive on the Microbox PCs and Panel PCs is "Disabled". To protect the CompactFlash card from early failure due to continuous writes, enable the EWF for the C:\ drive after you have finished the development of applications on the C:\ drive.

You can also commit all data that is stored in virtual memory to the CompactFlash card at any point in time, using one of the following commands:

- commit: The "commit" EWF command writes all data that has accumulated in virtual memory at that moment in time to the CompactFlash card. The CompactFlash card is not disabled.
- commitanddisable: The "commitanddisable" EWF command disables the EWF and then writes all data that has accumulated in virtual memory to the CompactFlash card. Typically, this command is followed by an enable command to once again protect the CompactFlash card.

To use the enhanced write filter manager, follow these steps:

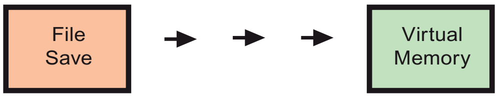
1. Open a command prompt window.
2. Enter "ewfmgr" followed by a drive designation and a command as illustrated.
3. Reboot the computer to make the command take effect.

NOTICE
After you enter commands, you must reboot the computer for the ewfmgr commands to take effect.

The ewfmgr commands that are applicable appear below, as applied to the C:\ drive:

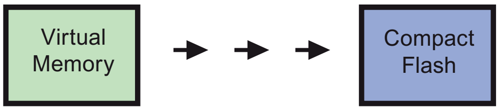
Enable the EWF:

Command: ewfmgr c: -enable



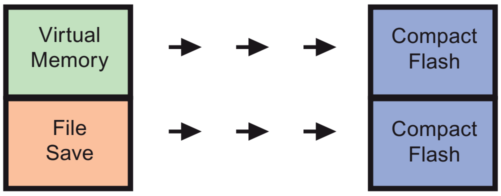
Commit Data to CompactFlash:

Command: ewfmgr c: -commit



Commit Data to CompactFlash and Disable the EWF:

Command: ewfmgr c: -commitanddisable



Note that the initial state of the Enhanced Write Filter is "disabled". Following a "commitanddisable" command, use the "enable" command to protect the CompactFlash card. All subsequent data writes will go to virtual memory.

Note

If the Enhanced Write Filter was shut down before downloading to WinLC RTX, you must click the button "Enable/Disable Station" in the Station Configuration Editor.

Replacing the CompactFlash card

In order to ensure a correct restart of WinLC RTX on a Microbox or Panel PC after replacing the CompactFlash card, perform one of the following steps:

- Disable the EWF for the CompactFlash drive before replacing the CompactFlash card and enable it after the first start of WinLC RTX.
- Store the WAF file (retentive data) on a drive on which the EWF is disabled.

Otherwise the next time you start WinAC RTX, it will start with an unbuffered startup.

Note

Flash card in Embedded Controller

Replacing the CompactFlash card is not possible with the Embedded Controller.

4.10.4 How WinLC RTX Loads Memory Areas on Startup

Upon startup, WinLC RTX searches for the stored power-down state (Page 73) to determine whether the controller had been shut down correctly and performs the following tasks:

- Loads the downloaded blocks of the STEP 7 user program from the Load memory (Page 73).
- Restores the work memory depending on whether a valid power-down state exists or not:
  - If WinLC RTX **does find** a valid power-down state, it 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. With the use of NVRAM for WinAC data storage (Page 89), the power-down state is often available even after a Windows Blue Screen event, as well as after a normal shutdown of the controller.
  - If WinLC RTX **does not find** a valid power-down state (showing that the controller had **not** been shut down correctly), it restores the work memory to its initial state from Load memory (as downloaded from STEP 7). The power-down state can be missing or unreadable if a Windows Blue Screen event occurred and WinAC Data storage is not in NVRAM.
- Restores the state (Page 73) 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 any element of the Load memory, state of the controller, or the power-down state, WinLC RTX starts an unconfigured (empty) controller.

Note

WinLC RTX cannot read the stored load memory or power-down state saved from a previous release of WinAC RTX or WinAC Basis. You can restore (Page 68) a STEP 7 user program and configuration that was archived from a previous release, but WinLC RTX cannot access retentive data from a previous release.

Loading Memory from a Valid Power-down State

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

- WinLC RTX loads data stored in the power-down state (Page 73) during startup. This includes the retentive S7 memory areas, the current values of the data blocks (work memory), and the contents of the diagnostic buffer (Page 65).

  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 (Page 73).

- Based on the Autostart (Page 88) settings, WinLC RTX sets the state (Page 73) of the controller to either STOP mode or RUN mode.

  In case of a Windows Blue Screen where the WinAC data storage (Page 89) is not in NVRAM, 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, unless you configured NVRAM for WinAC data storage.

  Note

  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 WinLC RTX last saved the state (Page 73) of the controller.

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

Loading Memory when no Valid Power-down State Exists

If the controller was not shut down properly, and WinAC data storage was not in NVRAM, WinLC RTX did 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 (Page 73) and reloads the system configuration, the process variables and S7 memory areas to the initial values configured in STEP 7.
- Reads the state (Page 73) of the controller and performs an unbuffered startup. (An unbuffered startup is like loading 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 (Page 88) settings, WinLC RTX sets the controller to either STOP mode or RUN mode.
- WinLC RTX sets the mode selector switch to the setting when WinLC RTX last saved the state (Page 73) of the controller.

Encountering Problems when Starting the Controller

If WinLC RTX cannot read (or encounters an error in reading) any 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. WinLC RTX still contains the STEP 7 user program and configuration, but no retentive data.

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 (Page 57) 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 (Page 57) to RUN to place the controller in RUN mode.

Starting the Controller After a Windows Blue Screen when WinAC Data Storage is in NVRAM

WinLC RTX is able to save the power-down state when the WinAC data storage (Page 89) is in NVRAM. When WinLC RTX starts up, it performs the actions described in "Loading Memory from a Valid Power-down State."

Starting the Controller After a Windows Blue Screen when WinAC Data Storage is not in NVRAM

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 (Page 69) and had placed the controller in STOP mode before shutting down, WinLC RTX still starts in RUN mode because without NVRAM data storage, WinLC RTX could not save the state (Page 73) setting for the controller during the shutdown caused by the Windows crash.

If you do not have NVRAM data storage, and 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.

4.10.5 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 (Page 73) 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 (Page 71).

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 (Page 147).

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 STEP 7 user 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 STEP 7 user 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.

4.10.6 Using an 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 (Page 73) in case of a power failure. Siemens strongly recommends the use of a UPS for operation with Windows XP Professional operating systems.

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.

NOTICE

Power loss without a shutdown of the operating system can cause filesystems of Windows XP Professional to be corrupted. For this reason, use a UPS system to protect filesystems with these operating systems.

In addition, some SIMATIC PCs can detect a power loss and send a power failure signal to WinLC RTX. WinLC RTX can then initiate a fast shutdown and save retentive data to NVRAM if so configured. See the topic "Retentive Data Storage following Power Loss or Blue Screen (Page 85)" for a list of the SIMATIC PCs that support the power failure signal, and a description of how WinLC RTX responds.

Systems with Windows XP embedded that use a compact flash file system that is protected with the Enhanced Write filter are stable against an unexpected loss of power.

See also

What's New? (Page 16)

4.10.7 Retentive Data Storage following Power Loss or Blue Screen

Previous topics discussed WinLC RTX response to a blue screen (Page 69), retentive data (Page 73), options for WinAC data storage (Page 75), configuring WinAC data storage (Page 89) and the possible use of an Uninterruptible Power Supply (UPS) (Page 84). Also, some SIMATIC PCs can send a signal to WinLC RTX to indicate a power failure.

All of these variations in features, configuration choices, and types of Windows or power failures will interact together. The behavior of your particular system depends on these variations in combination.

The following tables describe WinLC RTX behavior on various SIMATIC PCs under various conditions. NVRAM includes PCs with integrated 25 KByte SRAM, the SIMATIC WinAC NV128 card or integrated 128 KByte NVRAM, or the 512 Kbyte NVRAM of the S7-mEC:

SIMATIC PC with power failure detection	UPS	Retentive data storage possible in File Storage		Retentive data storage possible in NVRAM	
		In the event of power loss	In the event of Blue Screen	In the event of power loss	In the event of Blue Screen
Box PC 427B Panel PC 477B Box PC 627 (DC, basic board 4 and higher, with WinAC NV128 Card) Box PC 627B (DC, without integrated PROFIBUS) Panel PC 677 (DC, basic board 4 and higher, with WinAC NV128 Card) Panel PC 677B (DC) S7-mEC	Not required	No*	No	Yes	Yes

*The use of a UPS, though not required, would enable retentive data storage in file storage in the event of a power loss.

SIMATIC PC without power failure detection	UPS	Retentive data storage possible in File Storage		Retentive data storage possible in NVRAM	
		In the event of power loss	In the event of Blue Screen	In the event of power loss	In the event of Blue Screen
Box PC 627 (AC, DC up to basic board 4) Box PC 840 Panel PC 577 Panel PC 677 (AC, DC up to basic board 4) Panel PC 877 Rack PC 840 Rack PC 847B Rack PC IL 43	In use	Yes	No	Yes	Yes
	Not in use	No	No	No	Yes

*To store retentive data in the event of Blue Screen on PCs without Power failure detection, WinAC RTX must terminate properly by calling SFC STP (SFC 46). A power failure during a Blue Screen will lead to the loss of the retentive data.

The most reliable configuration is to use a SIMATIC PC with power failure detection and configure WinAC data storage to be in NVRAM.

See also

What's New? (Page 16)

4.11 Setting Controller Panel Options

4.11.1 Customizing Options

Customize Command (CPU Menu)

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 (Page 88).

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 (Page 88).

Data Storage

From the **Data Storage** dialog you configure various options about where WinLC stores data. See the chapter Storing Retentive Data (Page 73) for information about retentive data, and the topic Configuring WinAC Data Storage (Page 89) for how to configure your choices.

LEDs

The PLC Operating Mode LEDs selection provides an option to use the LEDs on a Microbox PC 427B or a Panel PC 477B to display the operating mode. This option is only selectable on those PCs. WinLC RTX will light the LEDs to display the RUN/STOP mode or a potential fault condition if you select this option and restart your computer. The topic Using the Status Indicators (Page 60) describes the LED status indications.

Energy Saving Function

By default, WinAC RTX disables the "SpeedStep" and "Cool'n'Quiet" energy saving functions of computers with Intel processors. These options save power but interfere with WinLC RTX operating at maximum efficiency. In general, leave these options disabled.

Note

Any changes to the WinLC RTX customizing options only take effect after WinLC RTX restarts. If WinLC RTX is not shut down, select the check box to restart the controller when you finish your changes. If WinLC RTX is shut down, the changes will take effect the next time you start the controller with the CPU > Start Controller menu command.

See also

Setting the Security Options (Page 94)

4.11.2 Selecting the Language

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

Procedure

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.

Result

The controller panel automatically changes to the selected language.

4.11.3 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.

Procedure

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.

Result

The next time the controller starts up, the operating mode will be the same as it was when the controller shut down.

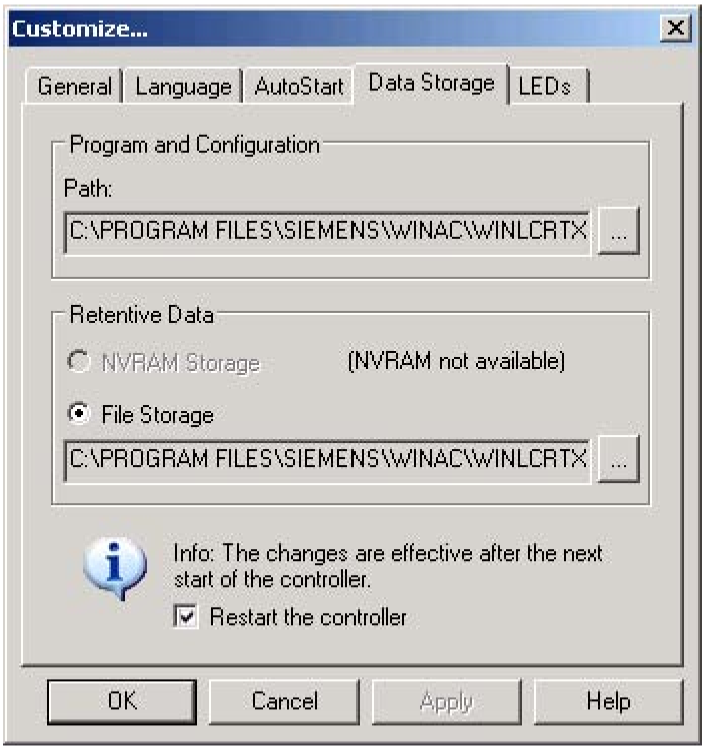
4.11.4 Configuring WinAC Data Storage

The Data Storage option dialog enables you to configure where (Page 75) WinLC RTX stores retentive data (Page 73) and the STEP 7 user program and configuration. Storing data in NVRAM, for example, helps guard against the loss of important program data following a power loss (Page 85).

NOTICE
If you change the program and configuration path, WinLC RTX can no longer access the STEP 7 user program and configuration data stored at the original location. If you change the retentive data storage settings, WinLC RTX can no longer access the existing retentive data from the original location. For this reason, archive your STEP 7 user program and data prior to changing any WinAC data storage parameters. You can restore the archive file after you restart the controller with the new WinAC data storage parameters. Alternatively, you can upload the STEP 7 user program and configuration to STEP 7 prior to making changes, and download the program to the controller after making changes.

To configure data storage options, follow these steps:

1. Archive (Page 68) your STEP 7 user program and configuration. Alternatively, you can upload the STEP 7 user program and configuration to STEP 7 using the **PLC > Upload Station to PG** menu command in STEP 7.
2. Select the **CPU > Options > Customize** menu command and the Data Storage tab of the Customize dialog:



Customize...

Program and Configuration
Path: C:\PROGRAM FILES\SIEMENS\WINAC\WINLCRTX

Retentive Data
○ NVRAM Storage (NVRAM not available)
● File Storage
C:\PROGRAM FILES\SIEMENS\WINAC\WINLCRTX

Info: The changes are effective after the next start of the controller.
☑ Restart the controller

3. In the Program and Configuration Path field, accept the default path or use the ... button to browse to a folder to use for the storage of the STEP 7 user program and configuration.
4. If your computer has NVRAM of any type, you can select either NVRAM Storage or File Storage for the Retentive Data specification. The dialog shows how much NVRAM storage is available. Otherwise, File Storage is your only choice. You can use the ... button to browse to a folder for the storage of the retentive data, or accept the default path.

   See the topic "Available Options for WinAC Data Storage" for a summary of the NVRAM possiblities (Page 75) and the computers that have NVRAM. In a few cases, the Data Storage tab will indicate that NVRAM is an available option, when in fact it is not. Verify the NVRAM availability on your specific computer before selecting the NVRAM data storage option.
5. Select the option "Restart the controller".
6. Click OK.
7. Restore (Page 68) the STEP 7 user program and configuration that you archived in Step 1. Alternatively, you can download the program and configuration from STEP 7 to the controller.

NOTICE

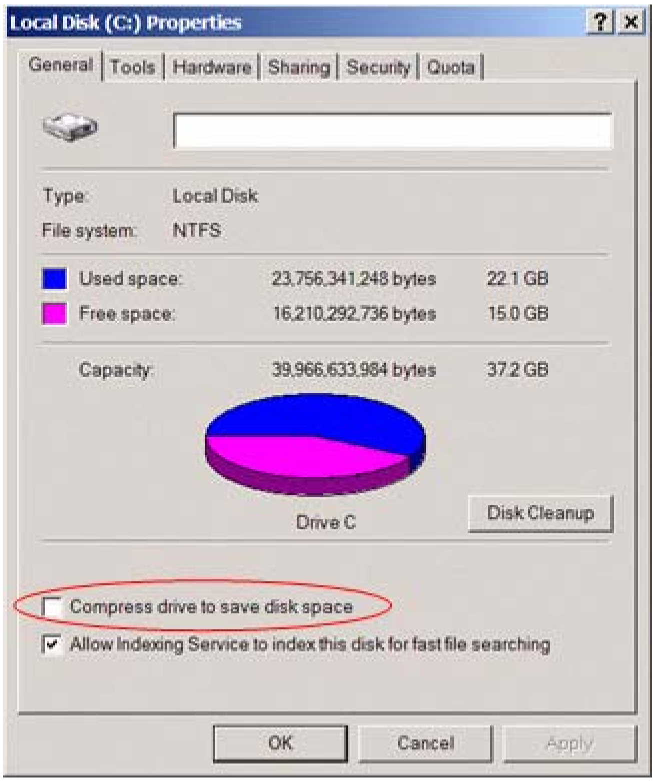
For SIMATIC PCs with compact flash cards an the EC31-RTX, you typically enable the Enhanced Write Filter (EWF) for the C:\ drive and leave the Enhanced Write filter disabled for the D:\ drive. With this usage, if you specify a file storage location on the D:\ drive for the STEP 7 user program and configuration and for the retentive data, WinAC RTX always recovers these files after a reboot. For this reason, specify a D:\ drive location for STEP 7 files and retentive data.

If however, you choose to store either program data or retentive data on the C:\ drive, and you have enabled the Enhanced Write Filter for the C:\ drive, you must manage the Enhanced Write Filter (Page 77) to commit data to the flash card. If you have files on the C:\ drive and you have not committed these files to the flash card, WinAC RTX cannot recover these files following a reboot. You must commit data on the C:\ drive to the flash card for it to be available after a reboot.

If you choose "File Storage" for retentive data, you must use an Uninterruptible Power Supply (UPS) to maintain data following a power loss. Without a UPS, retentive data in "File Storage" is lost after a power failure. If you choose "NVRAM Storage", Siemens recommends the use of a UPS, but a UPS is not a requirement.

File Storage Restrictions

Do not use a compressed file system for the Program and Configuration path or for the Retentive Data File Storage path. To determine whether a file system is compressed, examine the disk properties for the drive that you specified. Ensure that the "Compress drive to save disk space" checkbox is not selected.



Local Disk (C:) Properties

Type: Local Disk
File system: NTFS

🟦	Used space:	23,756,341,248 bytes	22.1 GB
🟪	Free space:	16,210,292,736 bytes	15.0 GB
	Capacity:	39,966,633,984 bytes	37.2 GB

□ Compress drive to save disk space
☑ Allow Indexing Service to index this disk for fast file searching

NVRAM Memory Allocation

The following types of information share the available NVRAM:

Item	Memory Consumption	Default	
System Startup Information	1 KByte	1 KByte	
Diagnostic Buffer	Number of entries * 20 bytes	2400 bytes	120 entries
Flag (M) Memory	Number of flag memory bytes	16 bytes	MB0 – MB15
S7-Timers	Number of timers * 2 bytes	0 bytes	No timers are retentive by default
S7-Counters	Number of counters * 2 bytes	16 bytes	C0 - C7
Retentive DBs configured with STEP 7 or created by SFC 85 with ATTRIB=0x00	Number of KBytes in retentive DBs	User program configuration	
Overhead for DBs created by SFC 85	Number of DBs * 45 bytes	0 bytes	

NOTICE

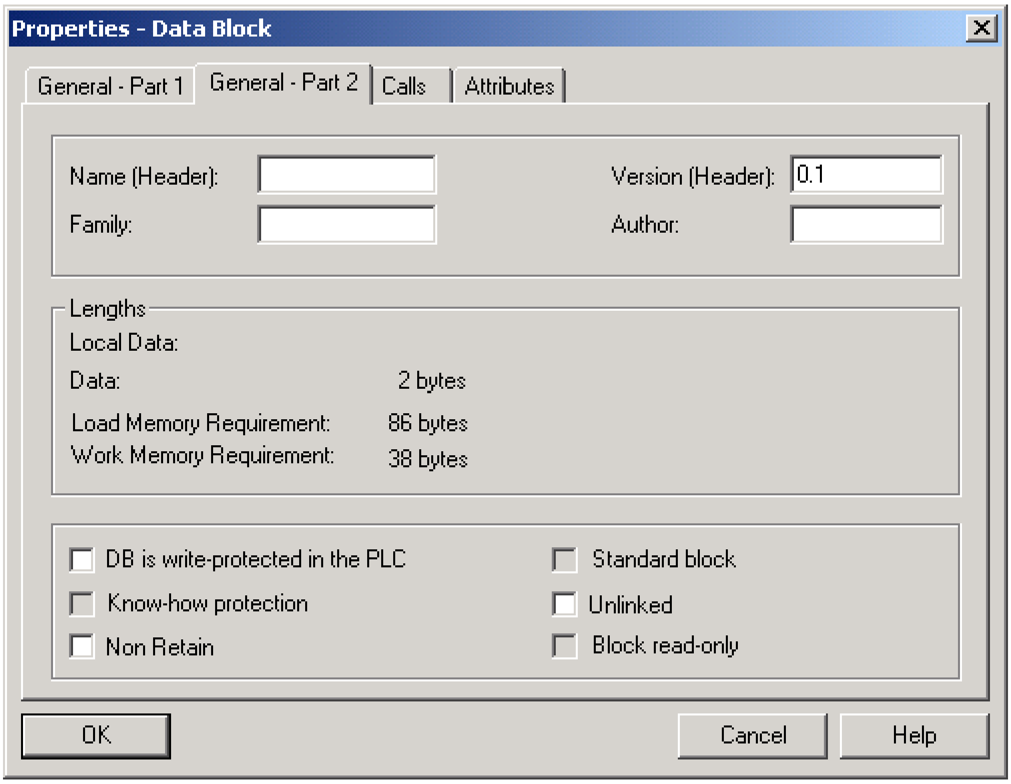
If you have stored retentive data on a SIMATIC WinAC NV128 card and you remove the card when your computer is off, the next time you start WinAC RTX, it will start with an unbuffered startup. The controller panel lights up the INTF status indicator, and the diagnostic buffer contains an "unbuffered power on" error.

To recover, you must either shut down WinAC RTX and your computer and reinstall the SIMATIC WinAC NV128 card, or you must switch to File Storage for your retentive data.

Retentive Data Indication for Data Blocks

By default, STEP 7 configures all data blocks as retentive. As shown below, the Properties dialog for a retentive data block displays all three of the following checkboxes as not selected (not checked):

- DB is write-protected in the PLC
- Non-retain
- Unlinked



Name (Header):
Version (Header): 0.1
Family:
Author:

Lengths
Local Data:
Data: 2 bytes
Load Memory Requirement: 86 bytes
Work Memory Requirement: 38 bytes

□ DB is write-protected in the PLC
□ Know-how protection
□ Non Retain
□ Standard block
□ Unlinked
□ Block read-only

If you select (check) any of these three checkboxes, the data block is non-retentive and not affected by the NVRAM restrictions.

Data Blocks Created by SFC 85

A data block created by SFC 85 is retentive if the ATTRIB parameter is 0x00. When considering NVRAM usage, these data blocks require memory for the overhead and for the retentive data. For data blocks created by SFC 85 with the ATTRIB parameter not equal to 0x00, the data block requires memory only for the overhead.

Exceeding NVRAM Storage

NOTICE
If you switch from File Storage to NVRAM Storage for your retentive data, and the retentive data in your STEP 7 user program requires more memory than the NVRAM supports, no retentive data can be reloaded after a startup. A message in the Diagnostic Buffer indicates that an unbuffered startup occurred. You must either reduce the size of the retentive data in your STEP 7 user program, or use File Storage instead of NVRAM Storage for the retentive data. The STEP 7 Hardware Configuration tool displays the current memory usage in the Module Information dialog.

See also

Using an Uninterruptible Power Supply (UPS) (Page 84)

4.11.5 Setting the Security Options

Security Command (CPU Menu)

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 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 (Page 96) 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.

4.11.6 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.

Procedure

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 (Page 94), you must enter the password at the Access Verification dialog.

Result

You have configured the password for accessing controller panel operations such as changing the operating mode and archiving and restoring STEP 7 user programs.

4.12 Startup Options for the Controller

4.12.1 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 (Page 55) the controller.
2. Select the CPU > Register Controller for Start at PC Boot menu command.

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 will now not start during the boot sequence. To start WinLC, you must manually start the controller (Page 55).

4.12.2 Setting the Restart Method

The restart method determines which startup OB (Page 108) 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 (Page 88) 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:

Option 1

1. Select the **CPU > RUN** menu command to change the controller from STOP to RUN mode.
2. If required, confirm your selection or enter the password.
3. Select either warm or cold restart from the Restart Method dialog.

Option 2

1. Right-click (using the right mouse button) the RUN mode selector switch position.
2. If required, confirm your selection or enter the password.
3. Select either warm or cold restart from the Restart Method dialog.

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 (Page 94) 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.

Result

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

4.13 Operating WinAC RTX 2008 with S7-modular Embedded Controller

4.13.1 Overview

Introduction

S7-modular Embedded Controller (S7-mEC) is a PC based Automation System with S7-300 design that runs on a preinstalled Windows XP Embedded operating system.

S7-mEC can be mounted in horizontal position onto a mounting rail. It can be expanded into a modular S7-mEC system by installing signal modules and standard PC expansion units which are commonly available on the market. A flash memory is used to backup the data.

S7-mEC is an open platform for user specific applications and provides access to S7-300 I/O.

EC31-RTX

EC31-RTX is the controller module within the S7-mEC preinstalled with the operating system Windows XP Embedded SP2 Feature Pack 2007, WinAC RTX 2008 and SIMATIC NET 2007.

Compared to WinAC RTX on other platforms, there are certain differences in the handling of EC31 with WinAC RTX.

- Layout and appearance of the panel
- The operation
- The hardware configuration

Note

New System status lists

The system status lists have been enhanced with S7-mEC functionality.

A listing of all systems status lists is available in chapter "System status list (Page 200)".

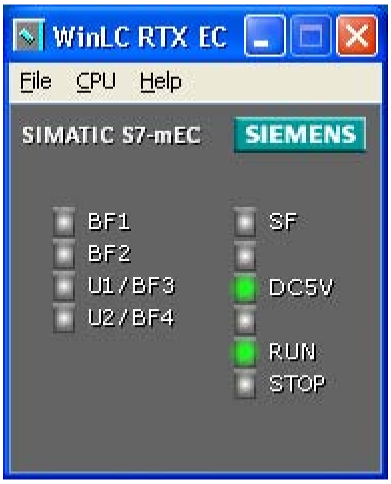
Reference

For further information on WinAC RTX within S7-mEC, refer to the *Embedded Automation, S7-modular Embedded Controller RTX* operating instructions.

4.13.2 Panel

Status and error displays of the panel

The picture below shows the WinAC RTX panel within S7-mEC:



The panel of the Embedded Controller is equipped with the following LEDs:

LED designation	Color	Meaning
BF1	red	Bus error at interface X1 PN LAN P1 or X1 PN LAN P2
BF2	red	Reserved
U1/BF3	red	Reserved
U2/BF4	red	Reserved
SF	red	Group errors
5 VDC	green	5V supply for the P bus
RUN	green	EC31 in RUN.
STOP	Yellow	EC31 in STOP.

LEDs BF1 to BF4 correspond to the IF slots of WinLC RTX. These are preconfigured in the component configurator.

Interfaces of the EC31-RTX

EC31-RTX comes with two Industrial Ethernet interfaces:

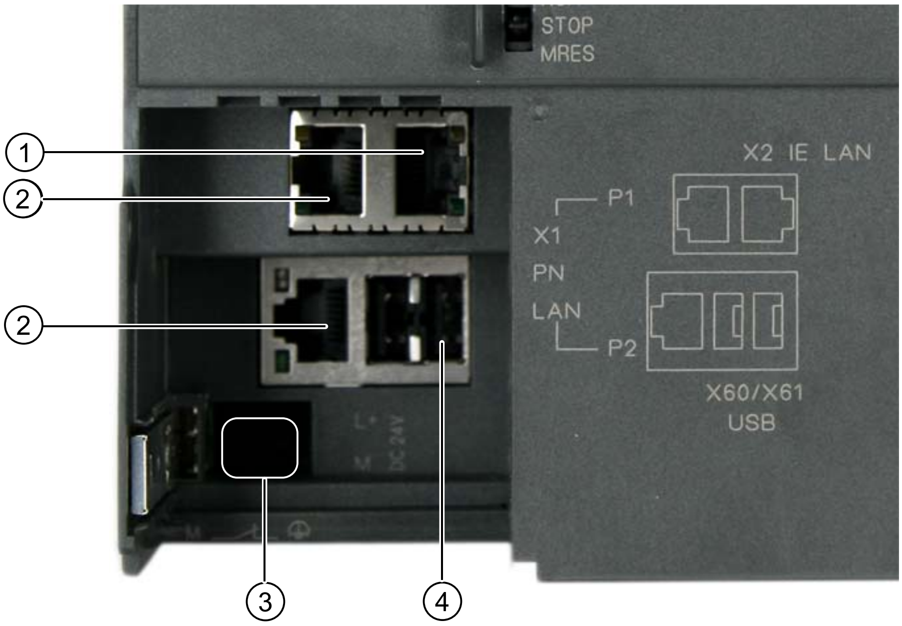
- X1 PN LAN P1 / X1 PN LAN P2

  This interface is used as PROFINET interface.

- X2 IE LAN

  The "X2 IE LAN" interface is is assigned to the PC station in index 3 at IE_General and is preconfigured for Industrial Ethernet communication.

The picture below shows the interfaces and connections of the EC31-RTX.



Number	Element
1	Ethernet port
2	PROFINET port
3	Power supply connection
4	USB 2.0 ports

4.13.3 Operation

Mode selector switch

You cannot change the operating state using the panel. You must use the mode selector switch on the module to set the EC31 to RUN, STOP or MRES state.

MRES function

A CPU memory reset (MRES) on the EC31 functions similar to the reset on an S7-300.

Data memory

Flash memory

The EC31 is equipped with a flash memory that can be used for storing data and to archive configuration data. This memory is split up into two default partitions.

NVRAM

The EC31 is equipped with an NVRAM area. This 512 KB memory is located on a separate chip.

4.13.4 Configuration

Communication

Communication can be implemented on the EC31 by means of the Ethernet interface.

The hardware is pre-configured. For further information, refer to the "Pre-configured system" chapter in the *S7-modular Embedded Controller RTX* operating instructions.

WinAC RTX within S7-mEC does not support the following communication interfaces:

- PROFIBUS DP
- MPI

Hardware configuration

By contrast to WinLC RTX, note that you can connect central I/O to the EC31 when configuring your hardware. The connection is implemented by means of backplane bus similar to the setup for an S7-300 CPU.

Archiving

Configuration data can be archived in *.wld files for reuse and transfer.

NOTICE
The wld files of WinAC RTX and of WinAC RTX within S7-mEC are incompatible.

STEP 7 Operations and Components 5

5.1 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 distributed I/O configuration through the STEP 7 Hardware Configuration (Page 47) tool
- Develop a STEP 7 user program using any of the STEP 7 control programming languages
- Configure operational parameters (Page 106) 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.

5.2 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.

5.3 Logic Blocks Supported by WinLC RTX

Table 5-1 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 OB 10 OB 20 OB 30 to OB 38 OB 40 OB 52 to OB 57 OB 61 and OB 62 OB 80, OB 82 to OB 86, and OB 88 OB 100 and OB 102 OB 121 and OB 122	SFC 0 to SFC 6 SFC 9 to SFC 15 SFC 17 to SFC 24 SFC 26 to SFC 34 SFC 36 to SFC 44 SFC 46 and SFC 47 SFC 49 to SFC 52 SFC 54 to SFC 59 SFC 62 and SFC 64 SFC 70 to SFC 71 SFC 78 to SFC 80 SFC 82 to SFC 84 SFC 85 and SFC 87 SFC 112 to SFC 114 SFC 126 and SFC 127	SFB 0 to SFB 5 SFB 8 and SFB 9 SFB 12 to SFB 15 SFB 19 to SFB 20 SFB 22 and SFB 23 SFB 31 to SFB 36 SFB 52 to SFB 54 SFB 81 SFB 65001, SFB 65002 and SFB 65003

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 65536 FCs (FC 0 to FC 65535). Each FC can contain up to 65570 bytes.
- Function block (FB): WinLC RTX supports up to 65536 FBs (FB 0 to FB 65535). Each FB can contain up to 65570 bytes.
- Data block (DB): WinLC RTX supports up to 65535 DBs (DB 1 to DB 65535). (DB 0 is reserved.) Each DB can contain up to 65534 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:

- Organization Blocks (OBs) (Page 108)
- System Functions (SFCs) (Page 112)
- System Function Blocks (SFBs) (Page 116)
- Communication Blocks (Page 119)

5.4 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 interrupt	0, 2 to 24
OB 56	Update interrupt	0, 2 to 24
OB 57	Manufacturer-specific 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 interrupt	24 to 26 (or 28)**
OB 83	Insert/remove module interrupt t	24 to 26 (or 28)**
OB 84	CPU Hardware fault	24 to 26 (or 28)**
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 Iiterrupt Range: 1 ms to 60000 ms	OB 20	1121	3
Cyclic interrupt Range: 1 ms to 60000 ms Recommended: > 10 ms	OB 30 OB 31 OB 32 OB 33 OB 34 OB 35 OB 36 OB 37 OB 38	1131 1132 1133 1134 1135 1136 1137 1138 1139	7 8 9 10 11 12 13 14 15
Hardware interrupt	OB 40	1141	16
Status interrupt	OB 55	1155	2
Update interrupt	OB 56	1156	2
Manufacturer-specific 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 (Page 106) 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 (Page 147) 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 (Page 154) 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" (Page 69))	OB 84	3585	26 (or 28)
Priority class error: Start event occurs for an OB that has not been downloaded. During the I/O cycle, WinLC attempts to access a module or slave that is defective or not plugged in.	OB 85	35A1, 35A3, 39B1, 39B2	26
Rack failure (distributed I/O): a node in the 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".

5.5 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 (PROFIBUS-DP only; see also SFC 70)
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 (not available for PROFINET I/O)
SFC 12	D_ACT_DP	Deactivates and activates slaves (PROFIBUS-DP or PROFINET I/O)
SFC 13	DPNRM_DG	Reads the diagnostic data of a DP slave (PROFIBUS-DP only; for similar functionality for PROFINET I/O, see SFB 54 and SFB 52) 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 21	FILL	Initializes a memory area 1 word 50 words 100 words
SFC 22 (Page 71)	CREAT_DB	Creates a retentive data block in Work memory The current values of the DB are retained after a warm restart
SFC 23 (Page 71)	DEL_DB	Deletes a data block. WinLC RTX allows an application to delete a non-sequence-relevant data block.

SFC	Name	Description
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	Cancels 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	Cancels 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 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	Retriggers 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 (Page 163)	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 (PROFIBUS-DP only, see also SFC 71)
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 (PROFIBUS-DP only, see also SFB 81)
SFC 55	WR_PARM	Writes the dynamic parameters (PROFIBUS-DP only, see also SFB 53)
SFC 56	WR_DPARM	Writes the default parameters (PROFIBUS-DP only, see also SFB 53 and SFB 81)
SFC 57	PARM_MOD	Assigns the parameters to a module (PROFIBUS-DP only, see also SFB 53 and SFB 81)
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	Name	Description
SFC 64	TIME_TCK	Reads the system time
SFC 70	GEO_LOG	Determines the start address of a module
SFC 71	LOG_GEO	Determines the slot belonging to a logical address
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 82 (Page 71)	CREA_DBL	Creates a data block in Load memory
SFC 83 (Page 71)	READ_DBL	Copies data from a block in Load memory
SFC 84 (Page 71)	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 (Page 71)	CREA_DB	Creates a DB that can be either retentive or non-retentive, depending on the input parameter If retentive, the current values of the DB are retained after a warm restart (OB 100 (Page 108)). If non-retentive, the current values of the DB are not retained after a warm restart (OB 100 (Page 108)).
SFC 87	C_DIAG	Determines the current status of all S7 connections
SFC 112	PN_IN	Copies input data from shadow memory of PROFINET CBA component to the associated interface DB
SFC 113	PN_OUT	Copies output data to shadow memory of PROFINET CBA component from the associated interface DB
SFC 114	PN_DP	Update interconnections between PROFINET CBA components on local PROFIBUS, and update interconnections between PROFINET CBA components on the local PROFIBUS and external PROFINET CBA components.
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 (Page 71)" 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 (Page 147)"):

- 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.

5.6 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.

Table 5-2 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 19	START	Initiate a warm or cold restart on a remote device
SFB 20	STOP	Change a remote device to STOP mode
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 35	ALARM_8P	Generates block-related messages with values for 8 signals
SFB 36	NOTIFY	Generates block-related messages without acknowledgment display
SFB 52	RDREC	Reads data set
SFB 53	WRREC	Writes data set
SFB 54	RALRM	Receives alarm data for a PROFIBUS-DP slave or PROFINET I/O device
SFB 81	RD_DPAR	Reads predefined parameters

SFB	Name	Description
SFB 65001	CREA_COM	(WinAC ODK CCX)
SFB 65002	EXEC_COM	(WinAC ODK CCX)
SFB 65003	ASYNC	(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".

Communication 6

6.1 Communication Blocks

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 19 SFB 20	START STOP	Initiate a warm or cold restart on a remote device Change a remote device to STOP mode
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

Like other S7 controllers, WinLC RTX provides Open User Communication blocks for exchanging data with other TCP/IP communications partners on a PROFINET submodule network:

FB	Name	Description
FB 63	TSEND	Send data over a communications connection
FB 64	TRCV	Receive data over a communications connection
FB 65	TCON	Establish a communications connection from the controller to a communications partner
FB 66	TDISCON	Terminate a communications connection from the controller to a communications partner
FB 67	TUSEND	Send data to a remote partner using UDP
FB 68	TURVC	Receive data from a remote partner using UDP

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

6.2 Using PROFIBUS DPV1

6.2.1 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 communication interface (Page 33) to be a DP Master (Page 121) in STEP 7.

6.2.2 Selecting a DP Master

Procedure

To select a 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 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.

Result

You have configured a submodule to be a DP Master.

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

Refer also to the following topics in this documentation:

- What Is a Communication Interface? (Page 33)
- Designating a Communication Interface as a Submodule (Page 38)
- Configuring the Hardware in STEP 7 (Page 47)

6.3 Using PROFINET

What is PROFINET?

Within the context of Totally Integrated Automation (TIA), PROFINET is the systematic development of the following systems:

- PROFIBUS DP, the well-established field bus
- Industrial Ethernet, the communication bus for the cell level.

Experiences from both systems have been and are being integrated in PROFINET.

PROFINET as an Ethernet-based automation standard from PROFIBUS International (PROFIBUS Nutzerorganisation e.V.) thereby defines a cross-vendor communication and engineering model.

Aims and Advantages of PROFINET

The objectives in PROFINET are:

- Open Ethernet Standard for automation based on Industrial Ethernet.
  Although you can use Industrial Ethernet and Standard Ethernet components together, the Industrial Ethernet devices are more sturdy and therefore better suited for industrial environments (temperature, immunity to interference, etc.)
- Use of TCP/IP and IT standards
- Automation with real-time Ethernet
- Total integration of field bus systems

PROFINET specifies functions for implementing an integrated automation solution from network installation through to web-based diagnostics.

With its modular structure, PROFINET can be easily upgraded to incorporate additional functions in the future.

This has the following benefits for you, the user:

- Flexibility due to the use of Ethernet and proven IT standards
- Less configuration and commissioning required due to modular structure
- Guaranteed return on investment for PROFIBUS devices and applications
- Faster than today's special buses in motion control
- Large range of products available on the market

Implementation of PROFINET by Siemens

We have implemented the aspects of the PROFINET architecture as follows:

- Communication between controllers and field devices is achieved with PROFINET IO.
- Communication between controllers as components in distributed systems is achieved with PROFINET CBA (Component Based Automation).
- Installation engineering and network components are available in SIMATIC NET.
- The proven IT standards from the field of office automation are used for remote maintenance and network diagnostics.

What is PROFINET IO?

As part of PROFINET, PROFINET IO is a communication concept that is used to implement modular, distributed applications.

PROFINET IO allows you to create automation solutions that are familiar to you from PROFIBUS.

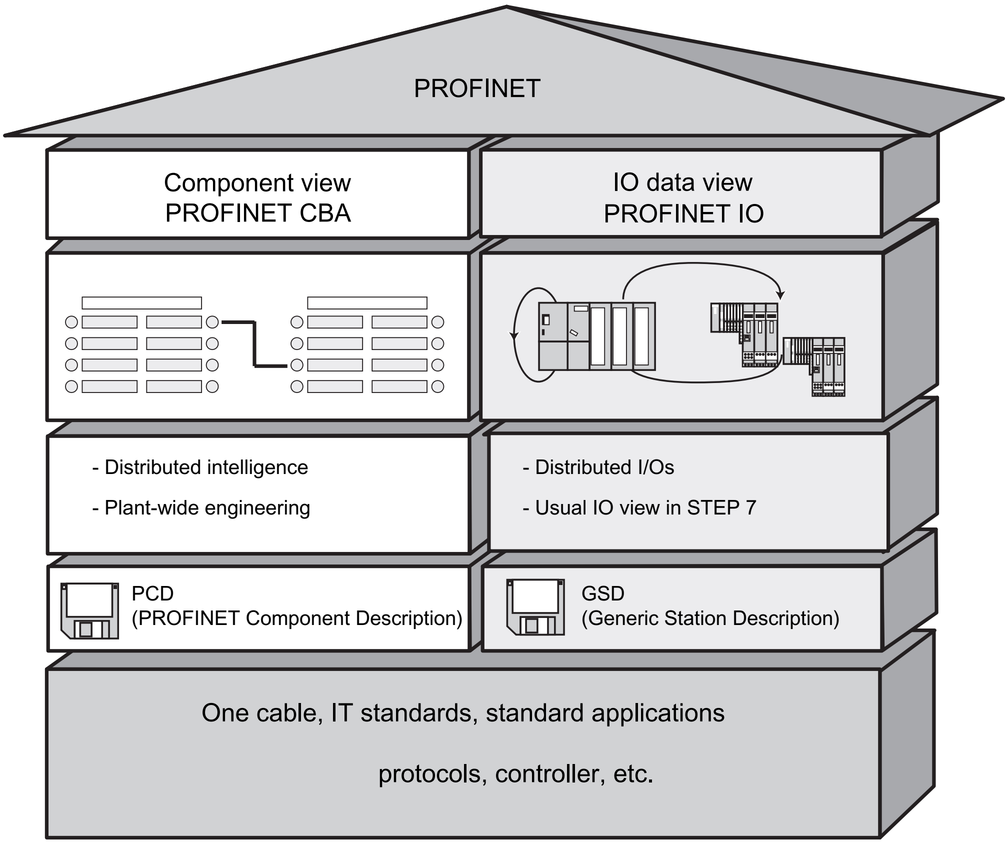
WinLC is used as an IO controller for communication via PROFINET IO.

What is PROFINET CBA?

Within the framework of PROFINET, PROFINET CBA is an automation concept for the implementation of applications with distributed intelligence. Component Based Automation allows you to use complete technological modules as standardized components in large systems.

PROFINET CBA lets you create distributed automation solutions, based on default components and partial solutions.

The WinLC controller acts like a DP master with proxy functionality during communication via PROFINET CBA.


PROFINET
Component view
PROFINET CBA
IO data view
PROFINET IO
- Distributed intelligence
- Plant-wide engineering
- Distributed I/Os
- Usual IO view in STEP 7
PCD
(PROFINET Component Description)
GSD
(Generic Station Description)
One cable, IT standards, standard applications
protocols, controller, etc.

Only special features of PROFINET if used with WinAC RTX 2008 are described in the next chapters.

Reference

Further information on the topic of PROFINET can be found

- in the "PROFINET System Description" system manual
- in the "PROFINET IO, Getting Started: Collection" manual.

Additional information on system migration from PROFIBUS DP to PROFINET IO can be found

- in the "PROFINET IO, From PROFIBUS DP to PROFINET IO" programming manual.

This manual also provides a clear overview of the new PROFINET blocks and system status lists.

6.3.1 PROFINET IO

6.3.1.1 Using PROFINET IO

As part of PROFINET, PROFINET IO is a communication concept that is used to implement modular, distributed applications.

PROFINET IO allows you to create automation solutions that are familiar to you from PROFIBUS.

The STEP 7 engineering tool helps you to structure and configure an automation solution. In STEP 7 you have the same application view, regardless of whether you are configuring PROFINET devices or PROFIBUS devices. You will program your user program in the same way for both PROFINET IO and PROFIBUS DP since you will use the extended blocks and system status lists for PROFINET IO.

PROFINET uses TCP/IP and IT standards as the Ethernet-based automation standard. PROFINET IO ensures communication between IO controllers and IO devices.

Devices capable of PROFINET (PN) communication

- PROFINET IO devices (for example, interface module IM 151-3 PN in an ET 200S)
- PROFINET CBA components
- S7-300 / S7-400 with PROFINET interface (for example, CPU 317-2 PN/DP or CP 343-1)
- Active network components (a switch, for example)
- PG/PC with Ethernet card
- IE/PB-Link

Properties of the PROFINET interface

IEEE standard	802.3
Connector design	RJ45
Transmission rate	Up to 100 Mbps
Protocols and communication functions	PROFINET IO PROFINET CBA PROFINET-Standard to IEC 61784-2, Conformance Class A and B Open block communication over TCP and UDP S7-Communication PG functions SNMP LLDP time synchronization based on the NTP method as client

Converting a user program to PROFINET IO

You can convert PROFIBUS DP user programs to PROFINET IO. This requires only a few changes to be made. The program does not have to be created from scratch.

If the OPC interface is used, the dynamics of the PROFINET IO OPC server and handling of items for services remains the same.

Only conversion tasks typical for SIMATIC CPUs such as the S7-300 and the S7-400 need to be performed.

6.3.1.2 Overview - PROFINET IO and WinAC RTX

The communications model of PROFINET IO was developed in order to benefit from the communication system Industrial Ethernet without redesigning the user programs and without losses in performance and deterministics.

With this cyclic data exchange, the WinLC controller is the master (IO controller) and the distributed I/Os take on the role of the slave (IO device).

PROFINET devices can be integrated in a PC-based automation system if the following requirements are met:

- a WinAC RTX software package with a WinLC RTX V4.4 controller
- STEP 7 V5.4 Service Pack 4 or higher version

CPs for PROFINET IO

WinAC RTX 2008 supports the following Ethernet cards:

Network adapter	Chipset	Designation in STEP 7 HW Config
CP 1616 hardware revision 8 or higher	ERTEC 400-1	CP1616-CP1604
CP 1604 hardware revision 7 or higher	ERTEC 400-1	CP1616-CP1604
S7-mEC CP1616/ERTEC400_EC (integrated)	ERTEC 400-1	PN-IO
SIMATIC PC 427B/477B (integrated) 1616	ERTEC 400-1	CP1616-CP1604
SIMATIC PC 627B/677B (integrated) 1616	ERTEC 400-1	CP1616-CP1604
SIMATIC Microbox PC 427B/Panel PC 477B integrated Intel PRO/1000 PL	Intel 82573L	IE general
SIMATIC Box PC 627B / Panel PC 677B integrated Intel PRO/1000 PL	Intel 82573L	IE general
SIMATIC Rack PC 847B integrated Intel PRO/1000 PL	Intel 82573L	IE general
Intel PRO/1000 GT (PCI)	Intel 82541PI	IE general
Intel PRO/1000 PL (integrated)	Intel 82573L	IE general

Use depends on the required quantity structure and possible fields of application. Only one of the network adapters can be configured as a submodule of the WinLC.

Note

Inoperative port of the CP 1604 and 1616

Port 4 on the hardware of the CP 1604 and 1616 is inoperative.

Only 3 ports are available for use in the hardware configuration of STEP 7.

Topology

PROFINET devices in industrial plants are linked via wired or wireless components. An overview of the network components can be found in the "PROFINET System Description" user manual in chapter 3 "Setting up PROFINET".

PROFINET IO supports the established network structures.

- Linear bus topology
- Star topology
- Tree topology

Different topologies can be combined with switches and routers. Switches with up to 10 ports (8 electrical and 2 optical) are available. A redundancy manager closes the open ends of the linear bus topology to form a ring topology.

6.3.1.3 Configuring submodules of WinLC RTX

An installation is local if WinAC RTX is installed on the same computer as STEP 7. WinLC RTX is then also configured locally. WinLC RTX can also be used as a remote controller from a different computer. The configuration operation is the same except the setting of the interface parameters.

Requirement

For local configuration of WinLC RTX, you need the Station Configuration Editor. The Station Configuration Editor shows the configuration of your PC station.

WinLC RTX supports a maximum of four submodules. They may consist of an Ethernet card, a CP5611/21 or up to four CP5613.

Procedure

1. Configure the submodules of WinLC RTX as described in chapter "Configuring communication interfaces (Page 38)".
2. Ensure that you assign a PROFINET communication interface to an interface slot.

6.3.1.4 Configuring WinLC RTX as a PN IO controller in STEP 7

This section describes how to configure a WinLC RTX controller as a PN IO controller in STEP 7.

Configuration in the hardware configuration of STEP 7

1. Set up a PROFINET IO system in the STEP 7 hardware configuration as described in chapter "Hardware configuration in STEP 7 (Page 47)".
2. Ensure that you add a PROFINET-capable communications processor on the same IF slot as in the Station Configuration Editor.

   The "Properties" dialog box opens.
3. Set the options of the PROFINET interface.
4. Verify the default IP address, device name and subnet mask.
5. Add the IO devices to the IO system.
6. Configure the IO system in the Topology Editor.

   Further steps are described in the help for STEP 7.

Assigning an IP address and a device name for identification on PROFINET

During communication with PROFINET IO the individual system nodes are not identified by an integer address as with PROFIBUS. As PROFINET is based on the TCP/IP and IT standards, all PROFINET devices are identified and addressed by means of a unique IP address. The IP address is assigned using the STEP 7 software.

The IP address is made up of 4 decimal numbers with a range of values from 0 through 255. The decimal numbers are separated by periods. The IP address is made up of:

- The address of the (subnet) network
- The address of the node (generally called the host or network node)

The IP address is assigned permanently to the device name. Therefore an IO device must have a device name before it can be addressed by an IO controller. This procedure was selected for PROFINET because names are easier to handle than complex IP addresses. Assignment of a device name for a concrete IO device can be compared with the setting of the PROFIBUS address for a DP slave.

Rules for the assignment of device names

You must conform to the following rules when assigning names to PROFINET IO devices:

- The PROFINET IO Controller, the PROFINET IO devices and the Engineering System are located on the same subnet.
- The PROFINET IO Controller is interconnected with the Engineering System via PROFINET IO interface.

PG/PC interface settings

Set up the PG/PC interface as described in chapter "Connecting STEP 7 to the Controller (Page 44)".

6.3.1.5 Loading configuration data in WinLC RTX

Requirement

The following actions must be completed before the configuration data can be loaded in the target device:

- Configuring submodules of the WinLC RTX
- Configuring WinLC RTX as a PN IO controller in STEP 7

Procedure

1. Set the PG/PC interface.
2. Start WinLC RTX.
3. Load the configuration data in WinLC RTX.

Reference

Further information on the topic:

- Loading configuration data
- PG/PC interface settings

can be found in chapter Connecting STEP 7 to the Controller. (Page 44)

6.3.1.6 New blocks

Comparison of the Blocks and System and Standard Functions of PROFINET IO and PROFIBUS DP

If a system is migrated from PROFIBUS DP to PROFINET IO, there are some blocks and system and standard functions which are new or which have to be replaced.

For additional information on this topic, refer to chapter 3 "Blocks in PROFINET IO and PROFIBUS DP" in the user manual "PROFINET IO, From PROFIBUS DP to PROFINET IO".

The table below provides an overview of the functions and blocks which have to be replaced by newer ones or which can be emulated in PROFINET IO.

Blocks/functions	PROFINET IO	PROFIBUS DP
SFC5 (query start address of a module)	No Replacement: SFC 70	Yes
SFC 13 (read diagnostic data of a DP slave)	No Replacement: • event-driven: SFB 54 • status-driven: SFB 52	Yes
SFC49 (query the slot at a logical address)	No Replacement: SFC 71	Yes
SFC54 (read default parameters - S7-400 CPU only)	No Replacement: SFB 81	Yes
SFC 55 (write dynamic parameters)	No Emulate using SFB53	Yes
SFC56 (write predefined parameters)	No Emulate using SFB81 and SFB53	Yes
SFC 57 (assign module parameters)	No Emulate using SFB81 and SFB53	Yes
SFC58/59 (write/read record in I/O)	No Replacement: SFB 53/52	Yes should already have been replaced by SFB 53/52 in DPV1
OB83 (hot swapping of modules/submodules)	Yes	Yes
OB86 (rack failure)	New error information	Unchanged

The following SIMATIC system functions are not supported for PROFINET IO:

- SFC11 (synchronize groups of DP slaves)

Note

New system status lists

New system status lists are available to cover PROFINET IO functionality.

A listing of all systems status lists is provided in chapter "System status list (Page 200)".

6.3.2 PROFINET CBA

6.3.2.1 Using PROFINET CBA

PROFINET CBA supports real-time communication in WinAC RTX 2008.

PROFINET IO and CBA can be operated simultaneously via an Ethernet network. Communication resources must be specified appropriately as they they share the same network. For information, refer to chapter "Hardware configuration (Page 132)".

Basic procedure for use in component-based automation

To be able to integrate WinLC RTX in PROFINET communication, you must perform the following tasks:

1. Set up the Ethernet submodule in the Station Configuration Editor.
2. Configure the hardware in HW Config of STEP 7.
3. Create a PROFINET interface DB in STEP 7.
4. Create a PROFINET component in STEP 7.
5. Configure and generate PROFINET communication in SIMATIC iMap.
6. Load the configuration data in the WinLC RTX controller.

Reference

The individual steps will now be briefly described.

They are described in detail in the "SIMATIC iMap" and "WinAC Basis" manuals.

6.3.2.2 Possible Configurations

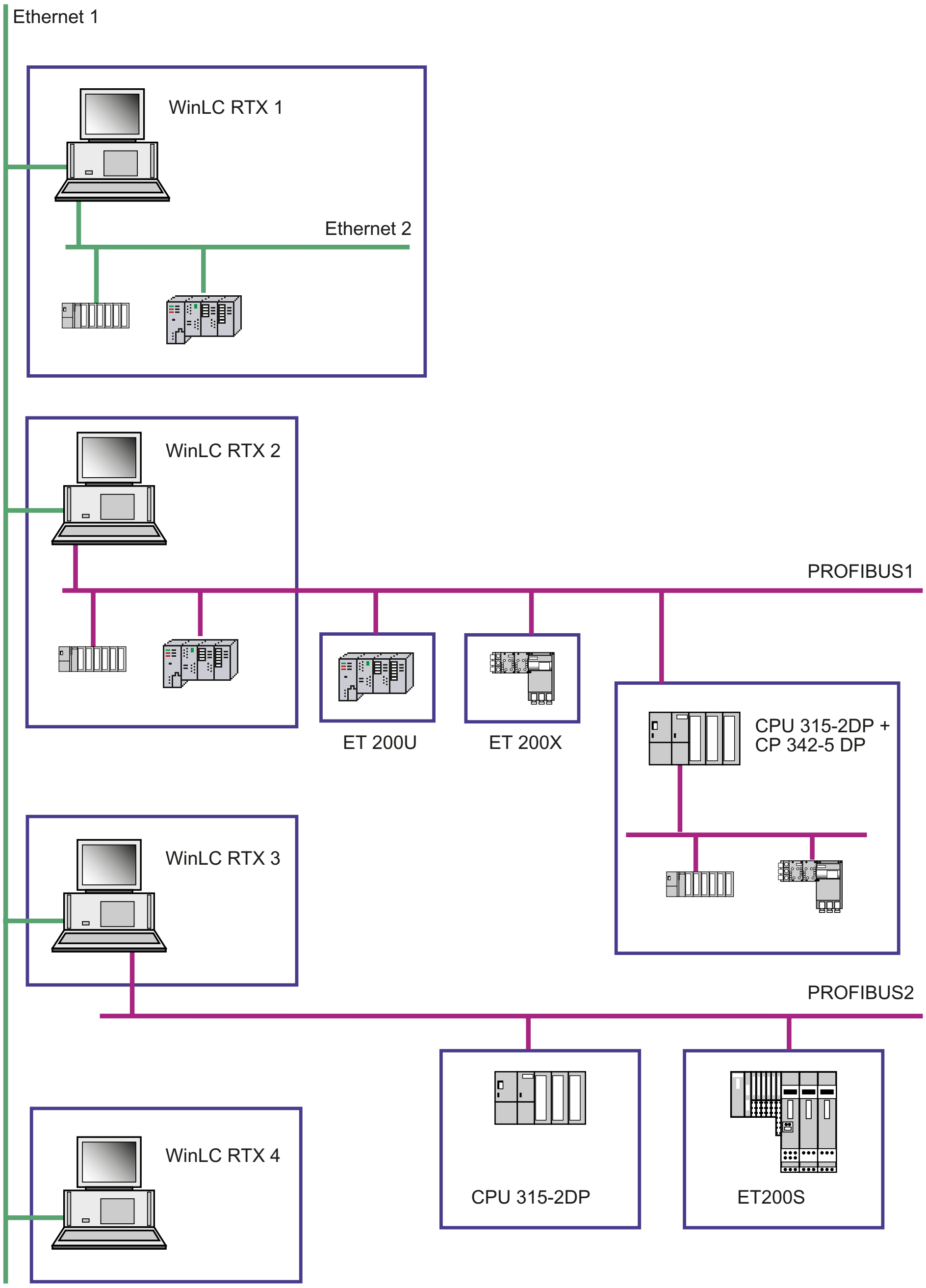
Examples of PROFINET components

There are four ways of carrying out hardware configurations for WinLC RTX as a PROFINET component.

- WinLC RTX 1 with proxy functionality as an IO controller
- WinLC RTX 2 with proxy functionality and local DP master system
- WinLC RTX 3 with proxy functionality
- WinLC RTX 4 without proxy functionality

Example Facility Configuration with WinLC RTX

The figure below schematically illustrates an example system containing the four possible configurations for WinLC RTX acting as a PROFINET component. Each frame in the figure designates one PROFINET component.



6.3.2.3 Configuring Hardware

Introduction

The hardware must be configured in the Station Configuration Editor and in HW Config of STEP 7. The station name, index and IF slots must be identical.

The configuration must contain the following modules:

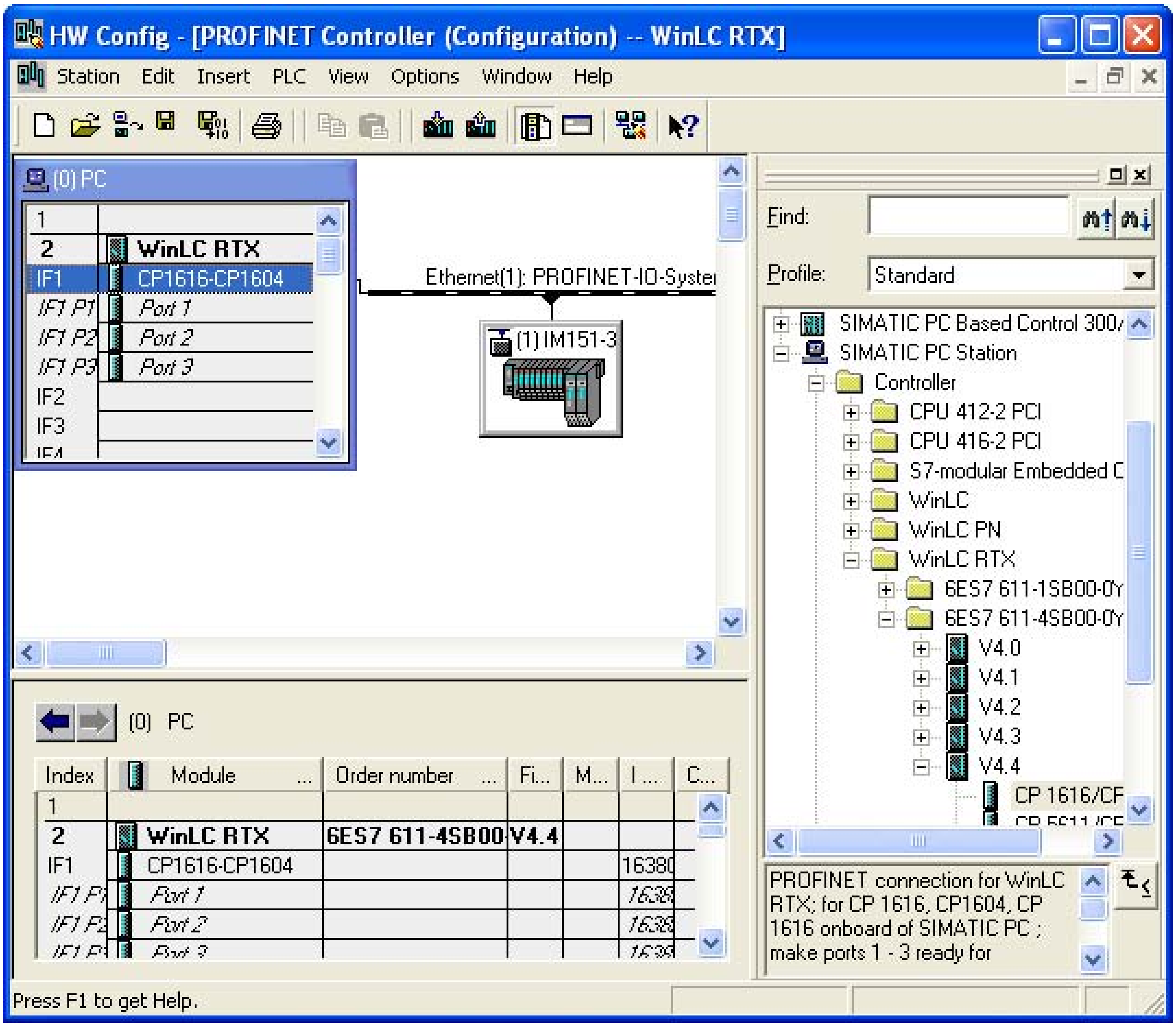
- WinLC RTX V4.4
- CP 1604/1616 or IE General for Industrial Ethernet – mandatory
- Optionally a CP 5611/21 or 5613 for PROFIBUS DP as a submodule on an IF slot The PROFIBUS-CP is required for the configuration as a DP master with proxy functionality.

Set up Ethernet submodules in the Station Configuration Editor

1. Configure the submodules of WinLC RTX as described in chapter "Configuring communication interfaces (Page 38)".
2. Ensure that you assign a PROFINET communication interface to an interface slot.

Configuring hardware in HW Config of STEP 7

1. Create a project in SIMATIC Manager and insert a SIMATIC PC station. The station name must be identical to that in the Station Configuration Editor.
2. Configure the hardware based on the following diagram:



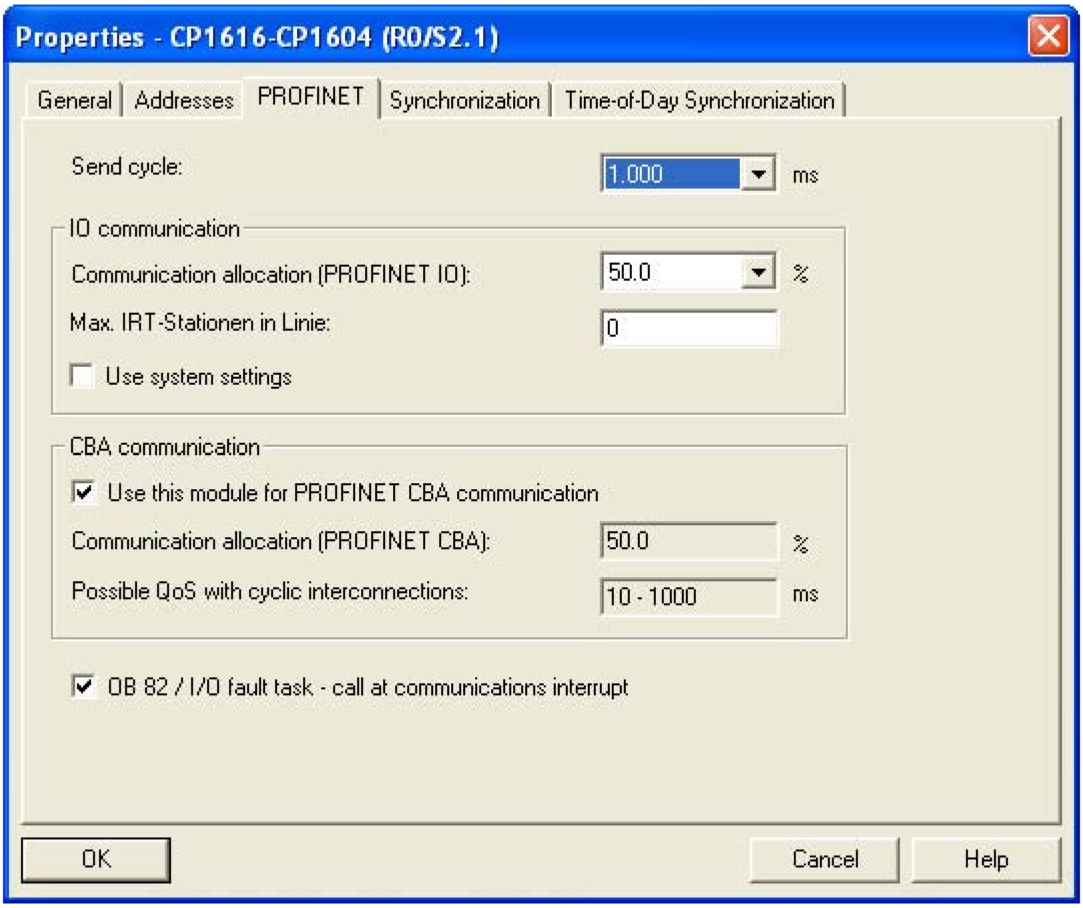
1	
2	WinLC RTX
IF1	CP1616-CP1604
IF1 P1	Port 1
IF1 P2	Port 2
IF1 P3	Port 3
IF2	
IF3	
IF4	

Index	Module ...	Order number ...	Fi...	M...	I ...	C...
1						
2	WinLC RTX	6ES7 611-4SB00	V4.4			
IF1	CP1616-CP1604				16380	
IF1 P1	Port 1				1638	
IF1 P2	Port 2				1638	
IF1 P3	Port 3				1638	

PROFINET connection for WinLC RTX; for CP 1616, CP1604, CP 1616 onboard of SIMATIC PC ; make ports 1 - 3 ready for

3. Configure a PROFINET IO system for the PROFINET-CP and link it.
4. In the dialog field "Properties CP 1604-1616" in the field "CBA communication" activate the option "Use this module for PROFINET CBA communication".

5. Assign PROFINET IO and PROFINET CBA to the appropriate volume of communication traffic.



Properties - CP1616-CP1604 (R0/S2.1)

Send cycle: 1.000 ms

IO communication

Communication allocation (PROFINET IO): 50.0 %

Max. IRT-Stationen in Linie: 0

□ Use system settings

CBA communication

☑ Use this module for PROFINET CBA communication

Communication allocation (PROFINET CBA): 50.0 %

Possible QoS with cyclic interconnections: 10 - 1000 ms

☑ OB 82 / I/O fault task - call at communications interrupt

6. Save and compile the HW configuration and close HW Config.

6.3.2.4 Creating the PROFINET interface DB

Prerequisites

Before you can create an interface DB, the hardware configuration of the PC station must be completed.

Procedure

1. In SIMATIC Manager, select the SIMATIC PC station and select **Create PROFINET interface** in the shortcut menu.

   The "New/Open PROFINET Interface" dialog box is opened.
2. Select WinLC RTX in the left window of the "New/Open PROFINET Interface" dialog. Activate the "New" option and confirm by clicking the "OK" button.

   The next dialog box shows the properties of the new block to create.
3. In the "Name and type" field, enter the desired block number, DB100 for example, and select the block type, "Global DB".
4. Click on the "OK" button.

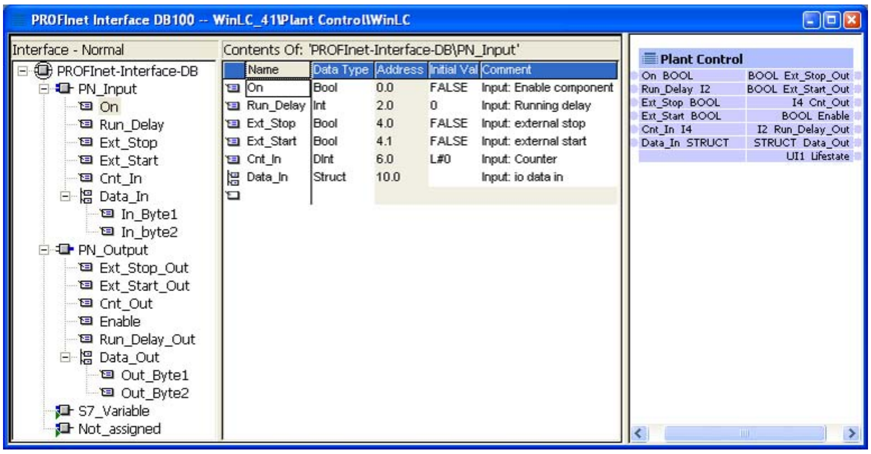
   The interface DB is opened in the PROFINET interface editor.
5. Enter the inputs of the technological function in the PN_Input section and the outputs of the technological function in the PN_Output section, and assign the entries the required properties: name, data type, interconnectable, HMI/MES
6. Save the PROFINET interface DB.

Result

The inputs and outputs for PROFINET communication are defined.

The interface DB is saved in the block folder of the PC station.

The following picture shows an example interface DB in the PROFINET interface editor.


PROFInet Interface DB100 -- WinLC_41\Plant Control\WinLC
Interface - Normal
Contents Of: 'PROFInet-Interface-DB\PN_Input'

```
PROFInet-Interface-DB
  PN_Input
    On
    Run_Delay
    Ext_Stop
    Ext_Start
    Cnt_In
    Data_In
      In_Byte1
      In_byte2
  PN_Output
    Ext_Stop_Out
    Ext_Start_Out
    Cnt_Out
    Enable
    Run_Delay_Out
    Data_Out
      Out_Byte1
      Out_Byte2
  S7_Variable
  Not_assigned
```

Name	Data Type	Address	Initial Val	Comment
On	Bool	0.0	FALSE	Input: Enable component
Run_Delay	Int	2.0	0	Input: Running delay
Ext_Stop	Bool	4.0	FALSE	Input: external stop
Ext_Start	Bool	4.1	FALSE	Input: external start
Cnt_In	DInt	6.0	L#0	Input: Counter
Data_In	Struct	10.0		Input: io data in

Plant Control	
On BOOL	BOOL Ext_Stop_Out
Run_Delay I2	BOOL Ext_Start_Out
Ext_Stop BOOL	I4 Cnt_Out
Ext_Start BOOL	BOOL Enable
Cnt_In I4	I2 Run_Delay_Out
Data_In STRUCT	STRUCT Data_Out
	UI1 Lifestate

Reference

Additional information about creating an interface DB can be found under "Properties of the PROFINET interface" in the SIMATIC iMap or SIMATIC Manager basic help.

6.3.2.5 Creating PROFINET components

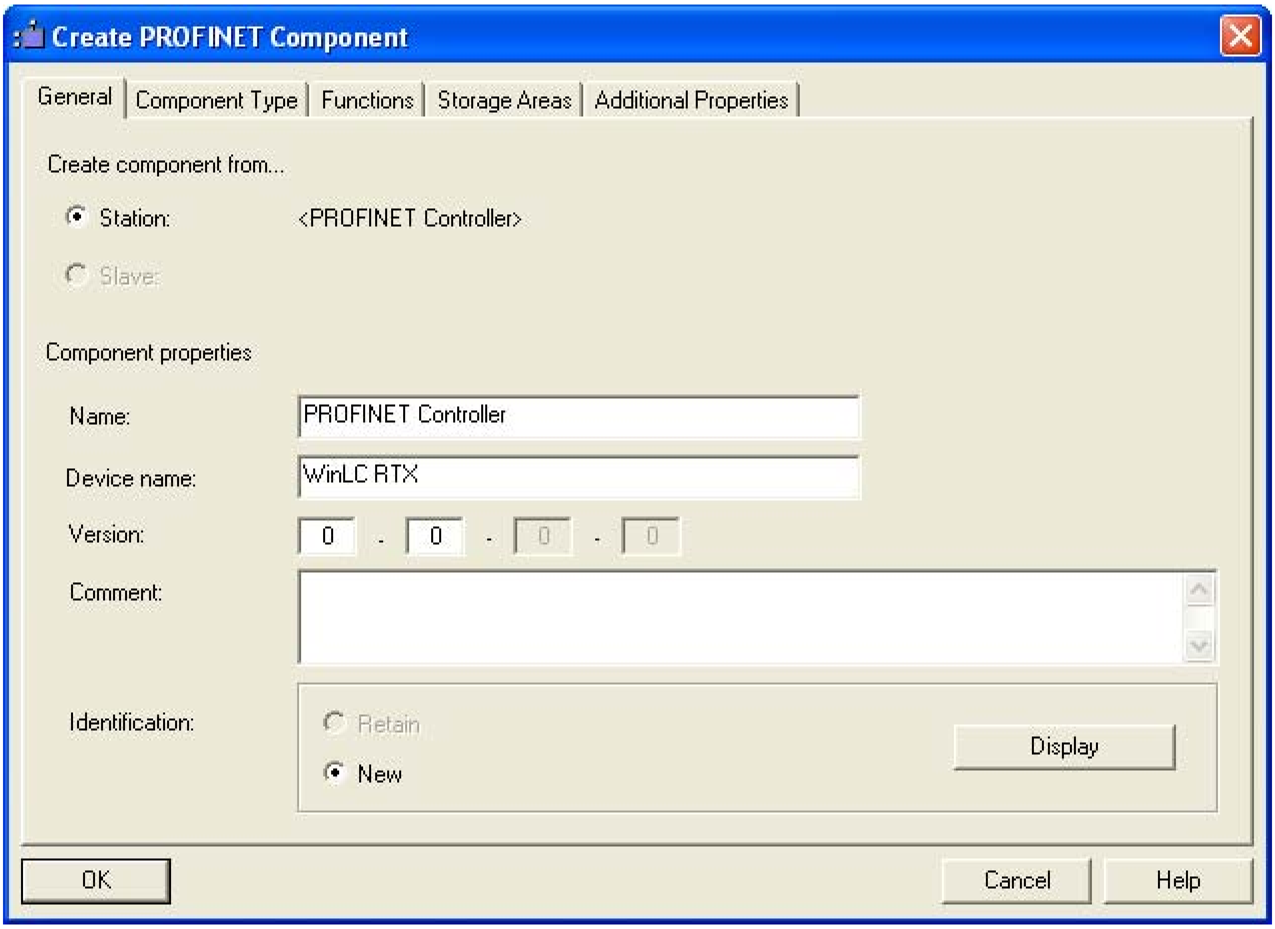
Prerequisites

- Hardware configuration of the PC station is completed.
- The interface DB has been created.

Procedure

1. In SIMATIC Manager, select the SIMATIC PC station and select **Create PROFINET component** in the shortcut menu.

   The "Create PROFINET component" dialog box opens.
2. Activate the option "Identification: New" in the "General" tab and enter a name.



Create PROFINET Component

General | Component Type | Functions | Storage Areas | Additional Properties

Create component from...

● Station: <PROFINET Controller>

○ Slave:

Component properties

Name: PROFINET Controller

Device name: WinLC RTX

Version: 0 . 0 . 0 . 0

Comment:

Identification: ○ Retain ● New

3. In the "Component type" tab, select:
   - Standard component with proxy functionality
   - Update of the PN interface - automatically (at the cycle checkpoint)
4. The "Functions" tab contains information about the technological function or subfunctions of the PROFINET component and which interface blocks are assigned.
5. Enter a storage location in the file system in the "File locations" tab.
6. In the "Additional properties" tab, enter the path of the icon files, and the path of the documentation link. You can also use the included icons (default path: Step7\s7data\s7cbac1x).

Result

The PROFINET component will be stored as an XML file at the specified file location along with the archived component project.

6.3.2.6 Configuring PROFINET communication in SIMATIC iMap

Prerequisites

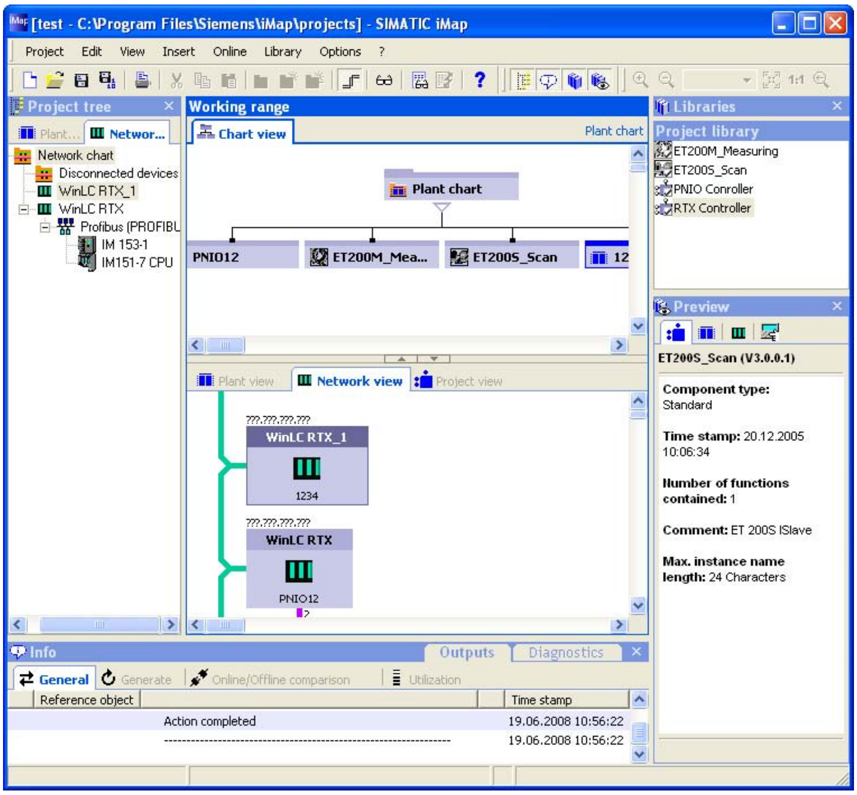
The following actions must have been completed prior to configuration in SIMATIC iMAP:

- Hardware Configuration
- Creating the PROFINET interface DB
- Creating a PROFINET component

Procedure

1. Start SIMATIC iMAP.
2. Import the required components in the project library and move them to the "plant tree" or "plan view" using a drag & drop operation.
3. Configure the communication connection in the "network" view of the work area.
4. In the "network" view of the work areas, assign IP addresses and the subnet mask to all devices.

   The project cannot be generated without assigning an IP address.
5. Interconnect the appropriate inputs and outputs in the "plant view".



ET200S_Scan (V3.0.0.1)

Component type: Standard

Time stamp: 20.12.2005 10:06:34

Number of functions contained: 1

Comment: ET 200S ISlave

Max. instance name length: 24 Characters

	Reference object			Time stamp
		Action completed		19.06.2008 10:56:22
		--		19.06.2008 10:56:22

6. Save the project.
7. Generate the project.

6.3.2.7 Loading configuration data in WinLC RTX

Prerequisites

The following actions must be completed before the configuration data can be loaded in the target device:

- Hardware Configuration
- Creating the PROFINET interface DB
- Creating a PROFINET component
- Configuring PROFINET communication in SIMATIC iMap.

Procedure

1. Set the PG/PC interface.
2. Start WinLC RTX.
3. Select the WinLC RTX target device and select the shortcut menu command **Download selected instances > All**

Result

The configuration data are loaded in WinLC RTX.

WinLC RTX is operational and can be viewed and diagnosed online with SIMATIC iMap.

Reference

Further information on the topic:

- Loading configuration data
- PG/PC interface settings

can be found in chapter Connecting STEP 7 to the Controller. (Page 44)

6.3.3 SNMP Communication Service

Availability

WinAC RTX 2008 supports the SNMP V1 (MIB-II) network protocol. Applications based on SNMP can be operated on the same network in parallel with PROFINET applications.

Properties

SNMP (Simple Network Management Protocol) is a standard protocol for TCP/IP networks.

Reference

For further information on the SNMP communication service and diagnostics with SNMP, refer to the *PROFINET System Description*.

6.4 Using open communication via Industrial Ethernet

6.4.1 Overview - Open communication via Industrial Ethernet

Introduction

Open User Communication via Industrial Ethernet supports the protocol variants "connection-oriented" and "connectionless" for data communication.

How the function blocks actually function depends on the protocol variant being used.

The blocks for Open User Communication via Industrial Ethernet

STEP 7 provides the following FBs and UDTs under "Communication blocks" in the "Standard Library" for data exchange with communication partners:

Block	Name	Description
FB 63	TSEND	Sending data
FB 64	TRCV	Receiving data
FB 65	TCON	Establish connection
FB 66	TDISCON	Disconnect
FB 67	TUSEND	Send data via UDP.
FB 68	TURVC	Receive data via UDP
UDT 65	TCON_PAR	Connection-oriented protocols: with the data structure for assigning connection parameters Connectionless protocol: with the data structure for assigning parameters for the local communications access point
UDT 66	TADDR_PAR	Connectionless protocol: with the data structure for assigning addressing parameters for the remote partner
UDT 651	TCP_conn_active	Connection-oriented protocols: with protocol-specific preassignments
UDT 652	TCP_conn_passive	Connection-oriented protocols: with protocol-specific preassignments
UDT 657	UDP_local_open	Connectionless protocol: with protocol-specific preassignments
UDT 661	UDP_rem_address and port	Connectionless protocol: with protocol-specific preassignments

The use of these blocks enables the controller to speak to any communication partner via any protocol via Ethernet. The protocol can be either TCP or UDP.

Note

WinAC RTX does not support the communication mechanism ISO over TCP for Open User Communication via Industrial Ethernet.

Quantity framework

For information on connection parameters and data volumes, refer to the Technical data of WinAC RTX 2008 (Page 193), order no. 6ES7 671-0RC06-0YA0. These are available at SIMATIC Customer Support.

6.4.2 Use standard FBs and UDTs

Introduction

The blocks and UDTs of the STEP 7 standard library must be used for open communication over Industrial Ethernet.

Procedure

1. Load the standard FBs and UDTs from the STEP 7 standard library in order to be able to use Open User Communication via Industrial Ethernet.
2. Integrate the standard FBs and UDTs from the STEP 7 standard library in your user program.

   The communication partner needs an appropriate user program in order to be able to communicate with WinAC RTX.
3. Name the standard FBs with valid parameters.

   The FBs call the blocks SFC 131 to 136.

Result

The blocks SFC 131 to 136 carry out the communication mechanism which makes the sockets interface of the PLC-internal TCP/IP stack accessible. The sockets are in PNIO stacks of the PLC and run in Ardence RTX.

Reference

The standard FBs and UDTs are made available in the STEP 7 standard library.

The individual blocks are described in the STEP 7 help.

6.4.3 Peculiarities with WinAC RTX

Introduction

To be able to assign parameters for TCP and ISO on TSP for communications connections, create a DB that contains the data structure from the UDT 65 "TCON_PAR". This data structure contains the parameters necessary for configuring the connection.

UDT 65

UDT 65 has the name TCON_PAR and is called by FB 65. It is a data block for assigning connection parameters.

For establishing a connection it is important to know to which IF slot your device is assigned. The values for local_device_id for WinAC RTX depend on the interface slot of the network adapter.

The values of local_device_id for the individual slots are:

- IF1: B#16#01
- IF2: B#16#06
- IF3: B#16#0B
- IF4: B#16#0F

Tuning the Performance of the Controller 7

7.1 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.
- 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.

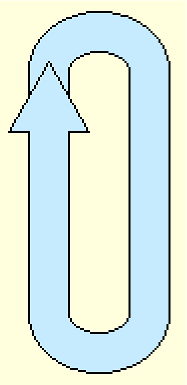
The Priority for the controller application also affects the scan cycle by determining when the controller runs or is interrupted by other Windows applications. You must ensure that the sleep time occurs every 50 milliseconds or less in order for other Windows applications, such as moving the mouse,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

The tuning panel (Page 62) 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 (Page 159), you use STEP 7 to configure the minimum 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.

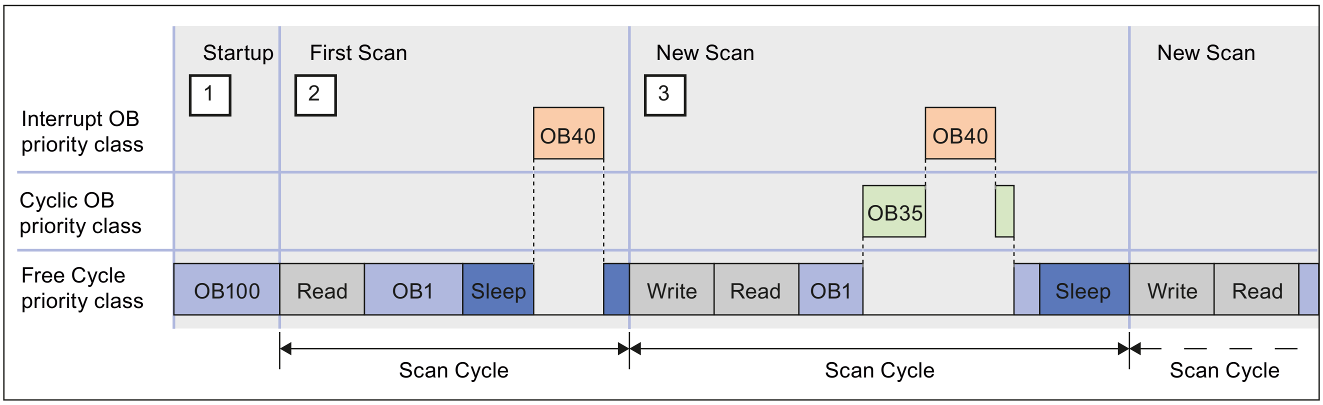
2 The controller reads the states of the I/O module inputs into the OB 1-assigned process-image input table (the I memory area).

3 The controller executes the STEP 7 user program in OB 1.

4 OB 1 waits until the minimum sleep time and minimum cycle time requirements are 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 cycle time that you configured with STEP 7, the controller suspends the free cycle (OB 1) until the minimum 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 illustration 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 (Page 147)"). You can use the following methods to manage the performance of WinLC RTX:

- Adjusting the priority for the controller (Page 152): 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 (Page 159): 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") (Page 163): 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 (Page 164): 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 (Page 62) for monitoring the performance and for modifying the parameters that affect the scan cycle.

7.2 Causes of 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

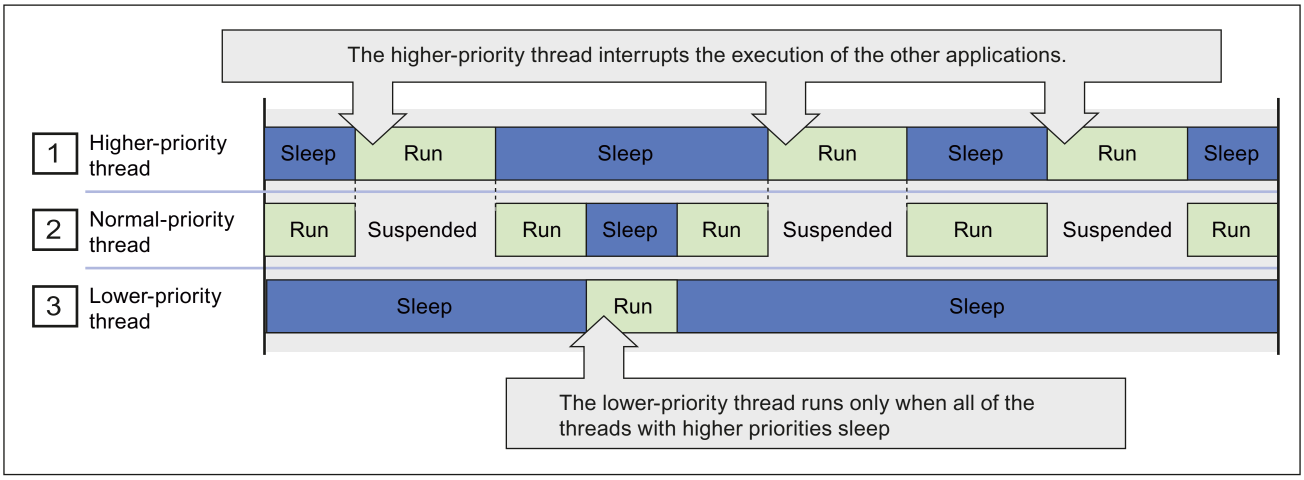
The tuning panel (Page 62) of WinLC RTX provides several tools for reducing jitter in the control program.

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. 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.

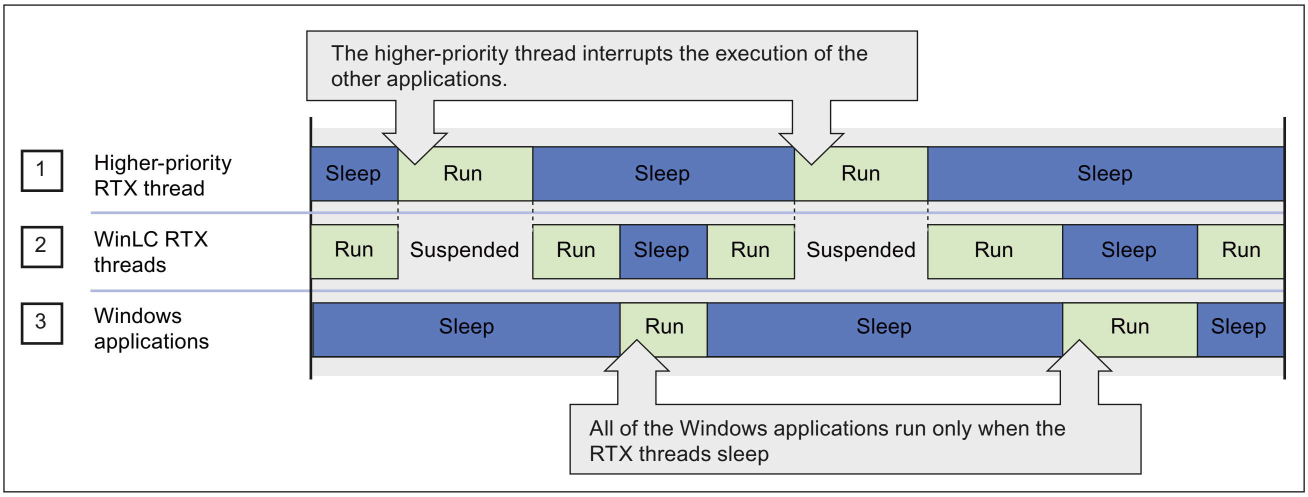
Priority Settings for Competing RTX Applications Can Cause Jitter

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.

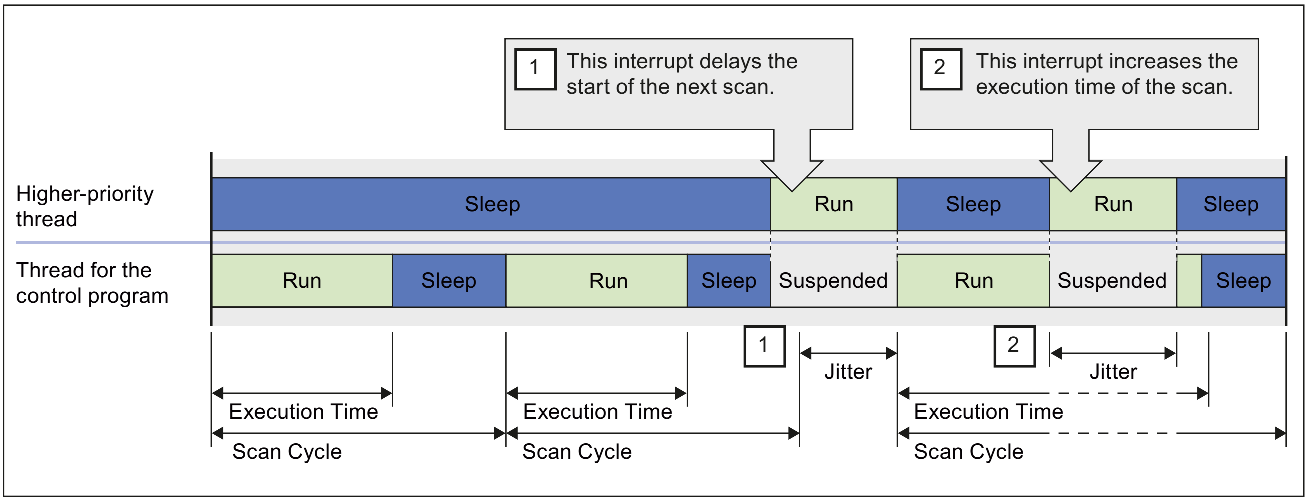

The higher-priority thread interrupts the execution of the other applications.
1 Higher-priority thread
2 Normal-priority thread
3 Lower-priority thread
Sleep
Run
Sleep
Run
Sleep
Run
Sleep
Run
Suspended
Run
Sleep
Run
Suspended
Run
Suspended
Run
Sleep
Run
Sleep
The lower-priority thread runs only when all of the threads with higher priorities sleep

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.


The higher-priority thread interrupts the execution of the other applications.
1 Higher-priority RTX thread
2 WinLC RTX threads
3 Windows applications
Sleep
Run
Sleep
Run
Sleep
Run
Suspended
Run
Sleep
Run
Suspended
Run
Sleep
Run
Sleep
Run
Sleep
Run
Sleep
All of the Windows applications run only when the RTX threads sleep

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 (Page 62) 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.

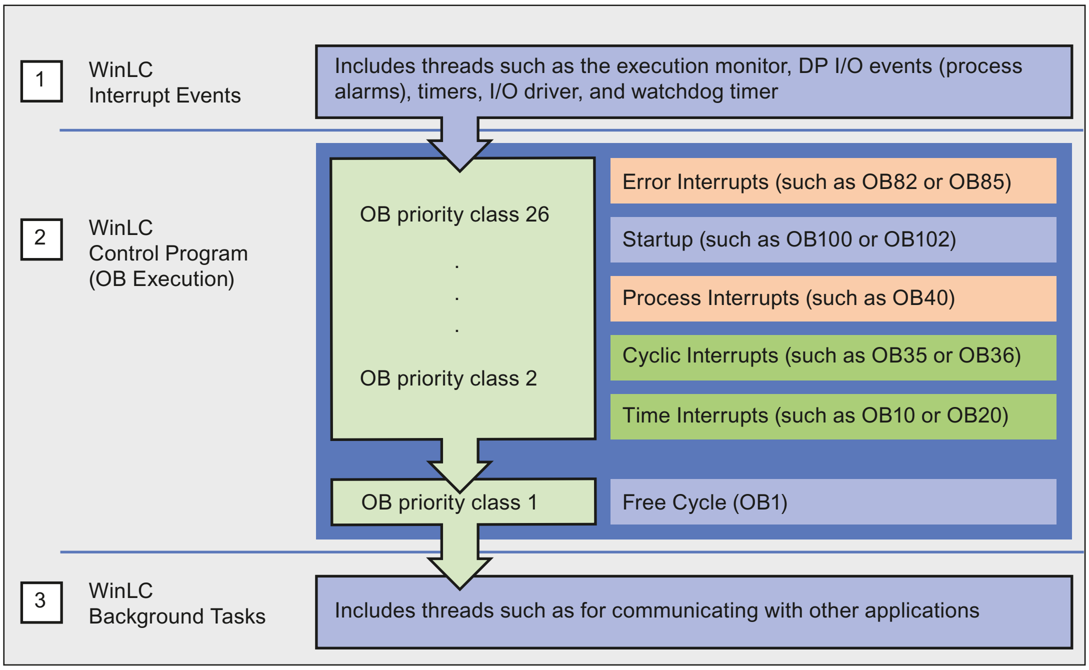
For more information about priorities, refer to the following topics:

- Adjusting the Priority (Page 152)
- Real-Time Subsystem Priorities (Page 153)

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 I/O events. 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
WinLC
Interrupt Events
Includes threads such as the execution monitor, DP I/O events (process alarms), timers, I/O driver, and watchdog timer
2
WinLC
Control Program
(OB Execution)
OB priority class 26
.
.
.
OB priority class 2
Error Interrupts (such as OB82 or OB85)
Startup (such as OB100 or OB102)
Process Interrupts (such as OB40)
Cyclic Interrupts (such as OB35 or OB36)
Time Interrupts (such as OB10 or OB20)
OB priority class 1
Free Cycle (OB1)
3
WinLC
Background Tasks
Includes threads such as for communicating with other applications

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.

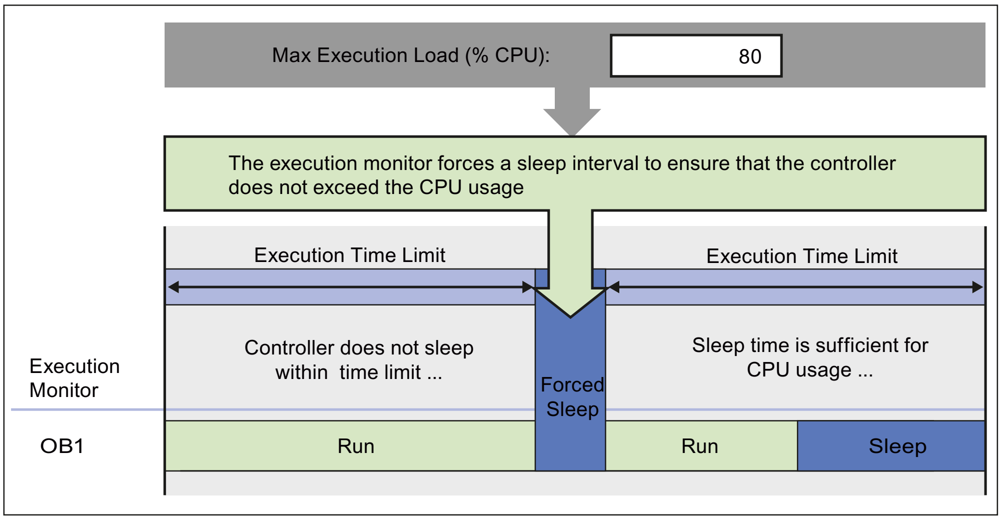
2 The OB Execution thread includes the different priority classes for the OBs of the control program. The interrupt OBs can cause jitter not only by interrupting the free cycle (OB 1), but also by interrupting another interrupt OB with a lower priority class.

3 The background tasks for WinLC RTX include the threads that are 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

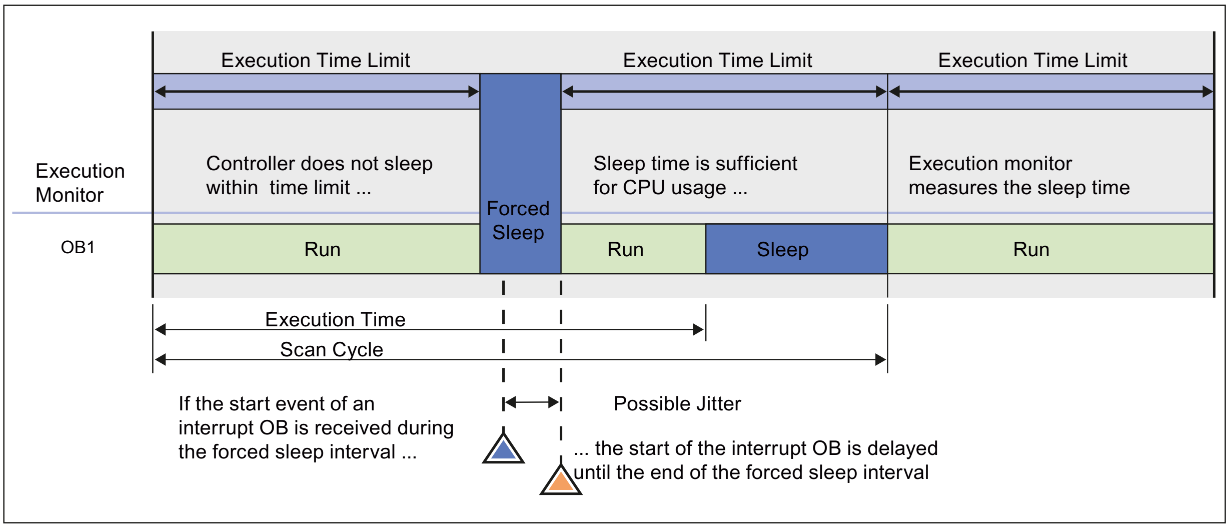
WinLC RTX must sleep (release the CPU) periodically in order for the other applications to run. The free cycle includes a sleep interval that follows the execution of OB 1. However, this sleep interval can be interrupted by higher-priority OBs. Also, a scan cycle with a relatively long execution time 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 (Page 164) forces a sleep interval.


Max Execution Load (% CPU):
80
The execution monitor forces a sleep interval to ensure that the controller does not exceed the CPU usage
Execution Time Limit
Execution Time Limit
Controller does not sleep within time limit ...
Forced Sleep
Sleep time is sufficient for CPU usage ...
Execution Monitor
OB1
Run
Run
Sleep

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.


Execution Time Limit
Execution Time Limit
Execution Time Limit
Execution Monitor
Controller does not sleep within time limit ...
Forced Sleep
Sleep time is sufficient for CPU usage ...
Execution monitor measures the sleep time
OB1
Run
Run
Sleep
Run
Execution Time
Scan Cycle
If the start event of an interrupt OB is received during the forced sleep interval ...
Possible Jitter
... the start of the interrupt OB is delayed until the end of the forced sleep interval

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

- You can increase the minimum sleep time (Page 159) parameter for managing the sleep time for the free cycle (priority class 1, or OB 1).
- You can call SFC 47 ("WAIT") (Page 163) 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 (Page 164) for managing sleep time at a higher priority class than any OB.

7.3 Adjusting the Priority of the Controller

7.3.1 Adjusting the Priority of the Controller

If other RTSS applications are executing on your computer 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 the 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.

A PC-based controller must maintain the essential features of a SIMATIC S7 PLC; however, the PC-based controller must also allow the other applications to run on the computer.

Procedure

To change the priority, follow these steps:

1. From the tuning panel, use the Priority slider to choose a priority based on the priority levels (Page 153) 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.

7.3.2 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 (Page 154) 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.

7.3.3 Threads and Priorities

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.

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.

7.4 Managing the Sleep Time

7.4.1 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 (Page 159). 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 (Page 163). 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 (Page 164). 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.

Managing the Sleep Time of the Controller

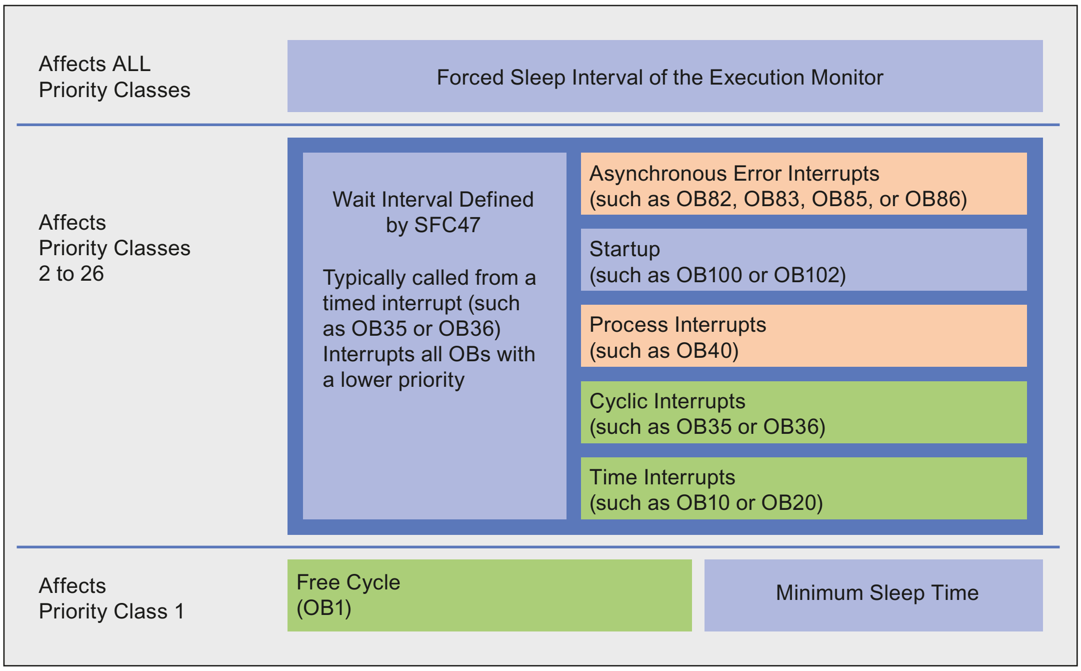
Because the controller shares the resources of your computer with other applications, you must ensure that the controller sleeps for a sufficient interval to allow the other applications to run.

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.


Affects ALL Priority Classes
Forced Sleep Interval of the Execution Monitor
Affects Priority Classes 2 to 26
Wait Interval Defined by SFC47
Typically called from a timed interrupt (such as OB35 or OB36) Interrupts all OBs with a lower priority
Asynchronous Error Interrupts (such as OB82, OB83, OB85, or OB86)
Startup (such as OB100 or OB102)
Process Interrupts (such as OB40)
Cyclic Interrupts (such as OB35 or OB36)
Time Interrupts (such as OB10 or OB20)
Affects Priority Class 1
Free Cycle (OB1)
Minimum Sleep Time

7.4.2 Tuning the Scan Cycle

Procedure

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 (Page 159) 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 (Page 62) 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 (Page 163) to add sleep time.
4. To further reduce any jitter, increase the execution time limit (Page 164) to the maximum possible execution time for your control program.

Result

The tuning techniques in most cases achieve an acceptable or imperceptible jitter time.

If the sleep management techniques do not provide adequate improvement in reducing jitter, consider increasing the priority for the controller (Page 152). (The priority of the controller is not the same as the priority class of an OB.)

7.4.3 Example: Using the Execution Monitor Alone

To help explain the tools for managing the sleep time of the controller, the following example shows how the execution monitor (Page 164) alone generates sleep time. You do not specify a minimum sleep time (Page 159) in this example.

A second example, shows how adding a minimum sleep time to the free cycle affects the execution of the free 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 tuning panel of controller has been configured with the following parameters:

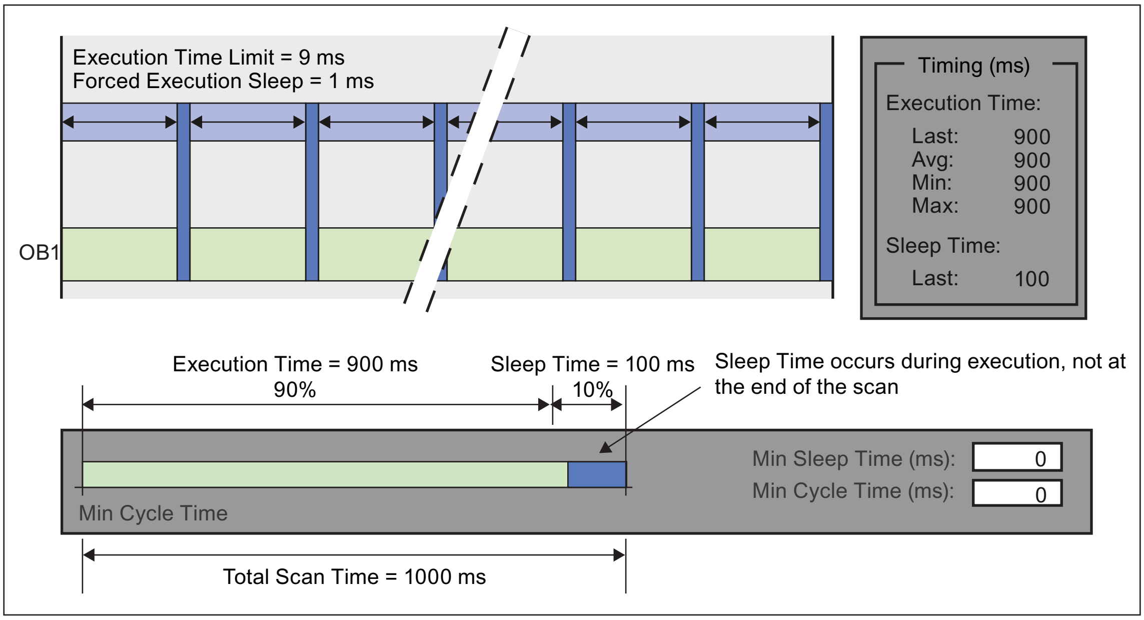
Parameter	Value
Execution Time	OB 1 takes 900 ms to execute.
Minimum Sleep Time	0 ms
Minimum Cycle Time	0 ms
Maximum Execution Load	90% (uses the default wake/sleep algorithm)
Execution Time Limit	9 ms (uses the default value)
Forced Execution Sleep	1 ms (uses the default value)

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.

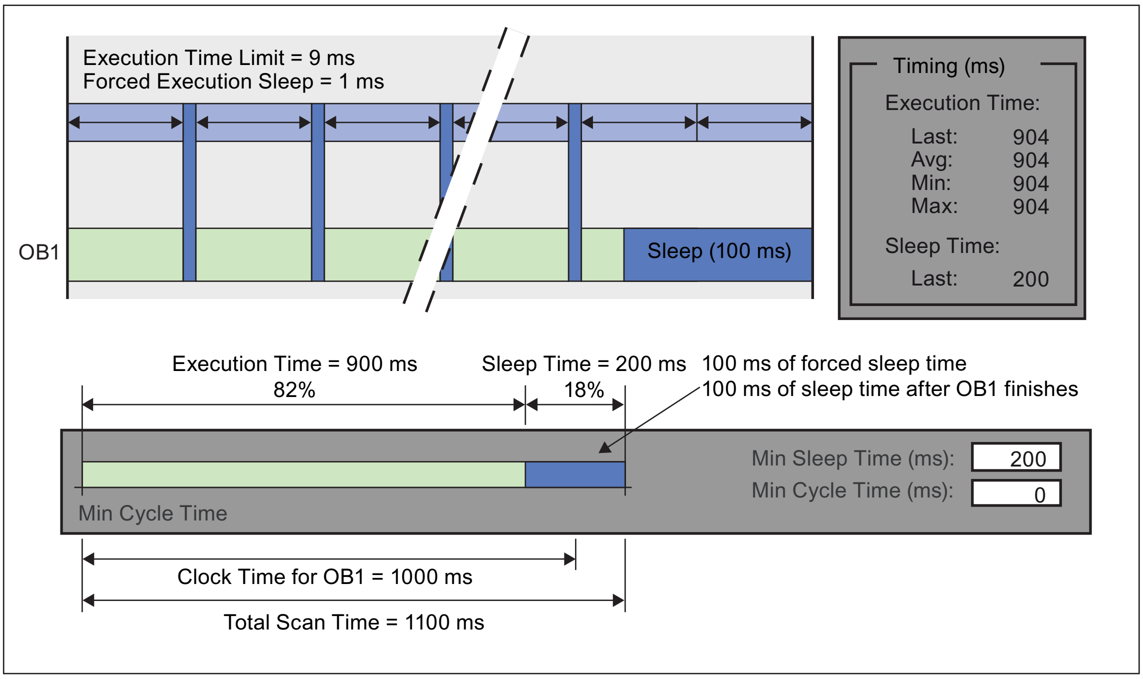
Notice that the sleep time occursat intervals within the execution of OB 1.



7.4.4 Example: Effect of 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).



7.4.5 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 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

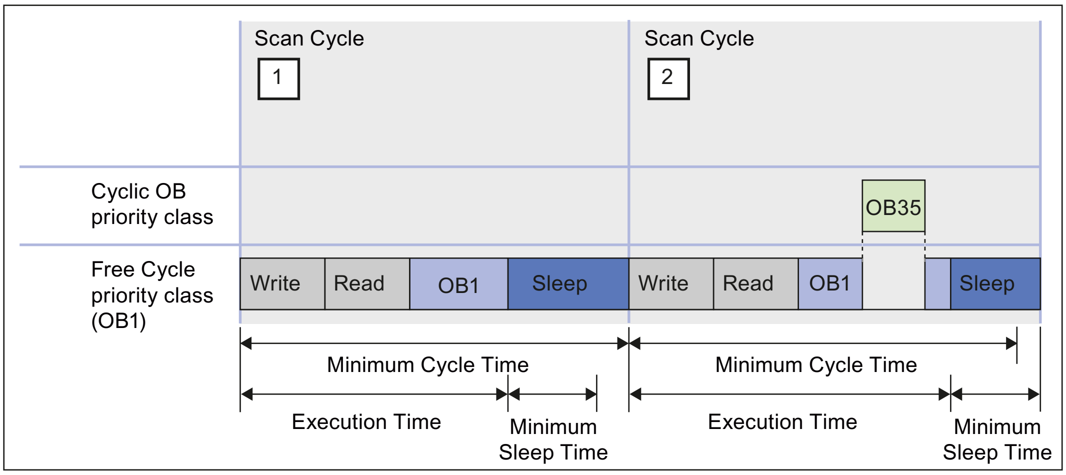
If you set the minimum scan time to a value larger than the watchdog time, WinLC goes to STOP mode during the first scan at the end of the watchdog time interval.

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

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

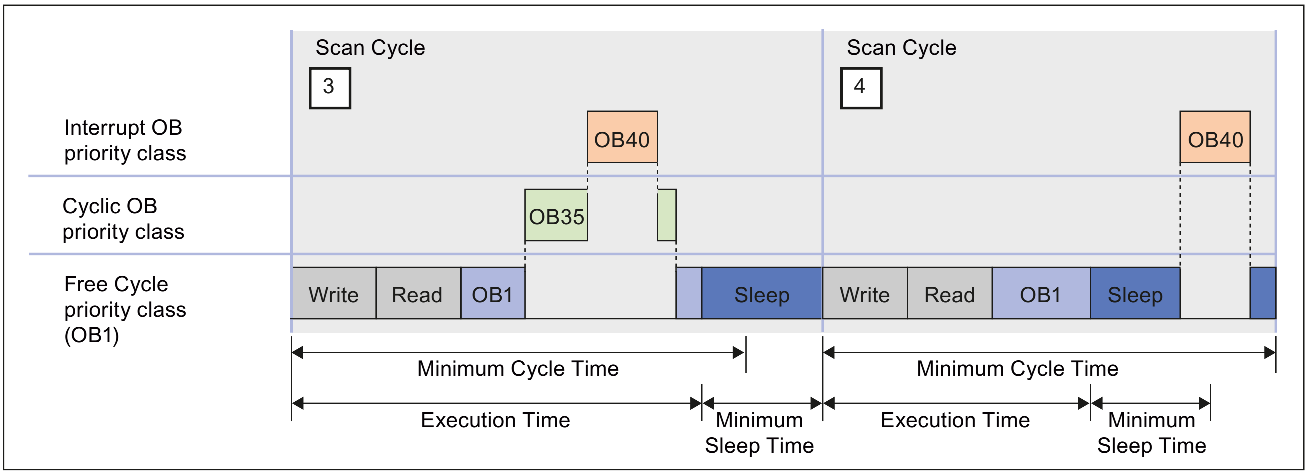
Parameters That Affect the Sleep Time for the Free Cycle

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.

  Note

  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 (Page 163)
- Adjusting the sleep-monitoring algorithm of the execution monitor (Page 164)

7.4.6 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.

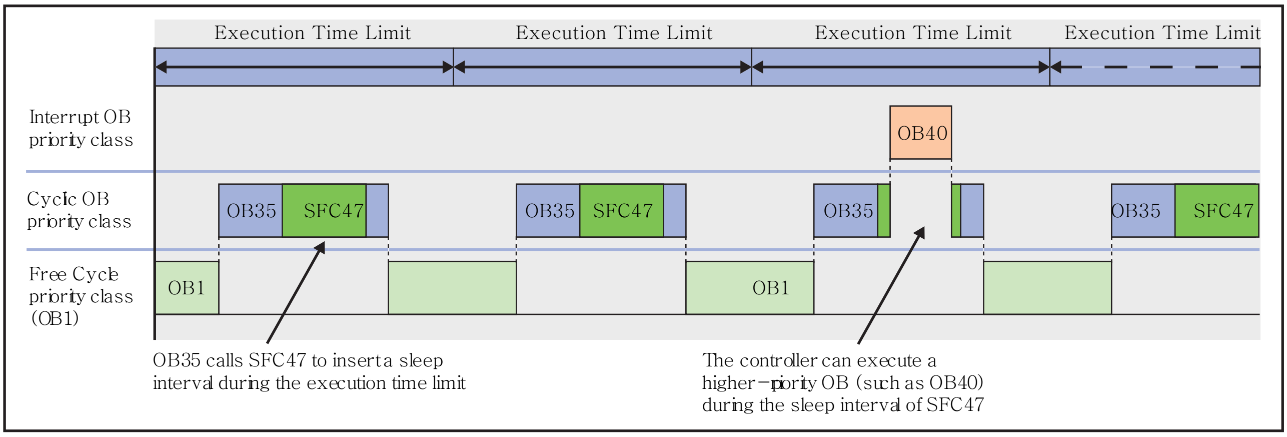
Procedure

To use SFC 47 to add sleep time, follow these steps:

1. Call SFC 47 from an OB in your STEP 7 user program. Typically, you call SFC 47 from a cyclic OB (such as OB 35) that starts within the execution time limit of the execution monitor.
2. To provide greater control over when the sleep time occurs, you can define which OBs are affected by setting the priority class of the OB that calls SFC 47.
   For more information, refer to the example: Avoiding Jitter in the Start Time of an OB (Page 169)

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 (Page 194) 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.


Execution Time Limit
Execution Time Limit
Execution Time Limit
Execution Time Limit
Interrupt OB priority class
Cyclic OB priority class
Free Cycle priority class (OB1)
OB40
OB35
SFC47
OB35
SFC47
OB35
OB35
SFC47
OB1
OB1
OB35 calls SFC47 to insert a sleep interval during the execution time limit
The controller can execute a higher-priority OB (such as OB40) during the sleep interval of SFC47

Additional Methods for Managing the Sleep Time

- Adjusting the Minimum Sleep Time and Cycle Time (Page 159)
- Adjusting the Sleep-Monitoring Algorithm of the Execution Monitor (Page 164)

7.4.7 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 (Page 169)

This topic contains the following information:

- 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

Operation of the Execution Monitor

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.

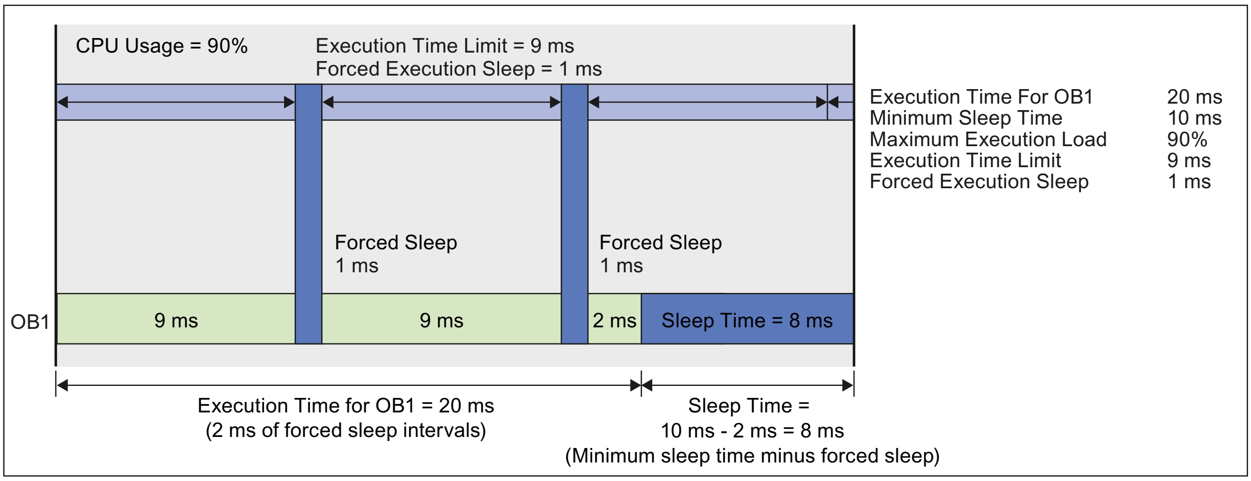
By measuring the sleep time, the execution monitor ensures that the controller allows the other applications to access the computer resources while the controller sleeps. The execution monitor also provides the safety net in cases where there are programming errors (for example, an infinite loop in OB 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.

The following illustration 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.


CPU Usage = 90%
Execution Time Limit = 9 ms
Forced Execution Sleep = 1 ms
Forced Sleep
1 ms
Forced Sleep
1 ms
OB1
9 ms
9 ms
2 ms
Sleep Time = 8 ms
Execution Time for OB1 = 20 ms
(2 ms of forced sleep intervals)
Sleep Time =
10 ms - 2 ms = 8 ms
(Minimum sleep time minus forced sleep)

Execution Time For OB1	20 ms
Minimum Sleep Time	10 ms
Maximum Execution Load	90%
Execution Time Limit	9 ms
Forced Execution Sleep	1 ms

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

Table 7-1 The sleep-monitoring algorithm of the execution monitor uses the following parameters:

Parameter	Description
Execution Time Limit	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. 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. The default value is 9000 microseconds (9 ms). **Note:** 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.
Maximum Execution Load	This value defines the maximum percentage of CPU usage that is allowed for the controller to execute OBs during each monitor interval. The default value is 90%.
Forced Execution Sleep	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. 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 (Page 194) 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. The default value is 1000 microseconds (or 1 ms).

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), and performs the following actions:

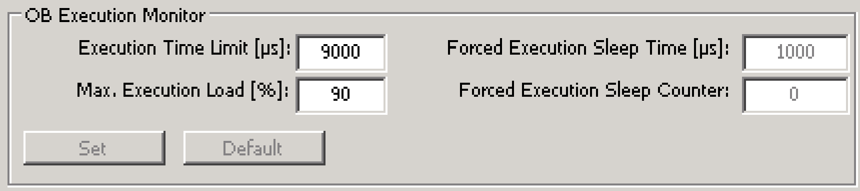
- If the controller sleeps **longer** than the sleep interval (forced execution sleep time), then the execution monitor restarts another monitor interval and does not affect the control program.
- If the controller sleeps **less** than the sleep interval (forced execution sleep time), then the execution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



OB Execution Monitor

Execution Time Limit [µs]: 9000

Max. Execution Load [%]: 90

Forced Execution Sleep Time [µs]: 1000

Forced Execution Sleep Counter: 0

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

Enter values in the Execution Time Limit and the Max. Execution Load fields. You can change one of the fields or both.

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

Table 7-2 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	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. By forcing a sleep interval, the execution monitor ensures that the other Windows application can be processed.

Situations that Prevent the Execution Monitor from Providing Sufficient Sleep Time

Table 7-3 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	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. 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.
Execution time for the control program and the configured execution time limit exceed approximately 50 ms	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. 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.

Additional Methods for Managing Sleep Time

- Adjusting the minimum sleep time and minimum cycle time parameters (Page 159)
- Inserting sleep time into the control program (SFC 47 "WAIT") (Page 163)

7.4.8 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):

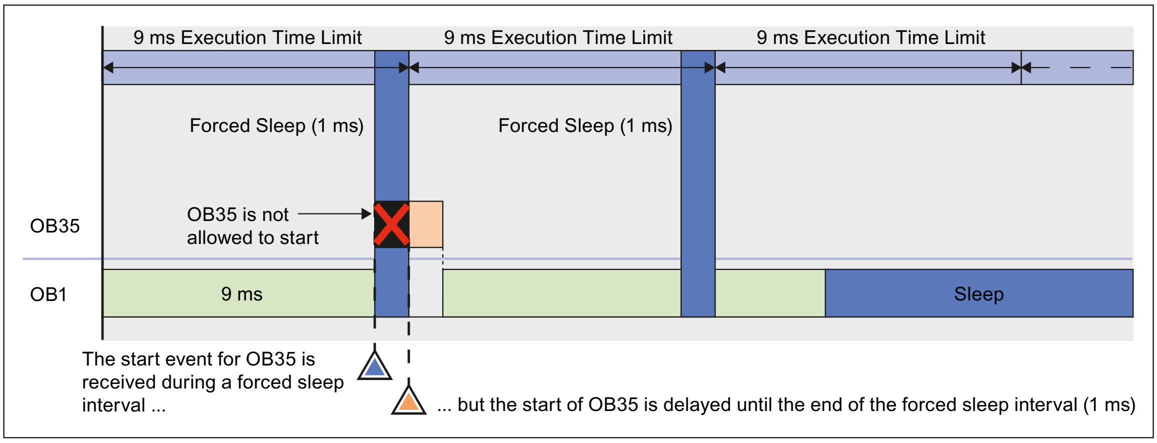
- Inserting a sleep interval into the execution of your STEP 7 user program. For this solution, you call SFC 47 ("WAIT") and specify the length of time to sleep. The controller can interrupt this sleep interval to process other OBs.
- Changing the sleep-monitoring algorithm of the execution monitor.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

Scenario

Table 7-4 In the example, a STEP 7 user program 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 of up to 1 ms between 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.


9 ms Execution Time Limit
9 ms Execution Time Limit
9 ms Execution Time Limit
Forced Sleep (1 ms)
Forced Sleep (1 ms)
OB35
OB35 is not allowed to start
OB1
9 ms
Sleep
The start event for OB35 is received during a forced sleep interval ...
... but the start of OB35 is delayed until the end of the forced sleep interval (1 ms)

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, follow these steps:

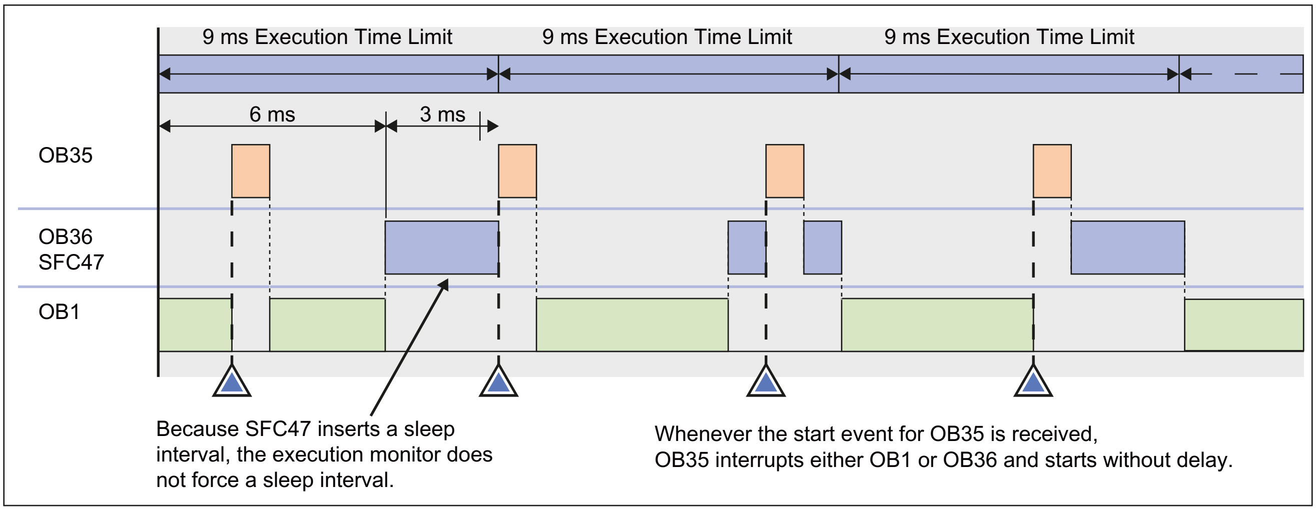
3. From the STEP 7 Program Editor, create an OB 36 for your STEP 7 user program.
4. Enter the following program:

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

To configure the priority level time and execution time for OB 36, follow these steps:

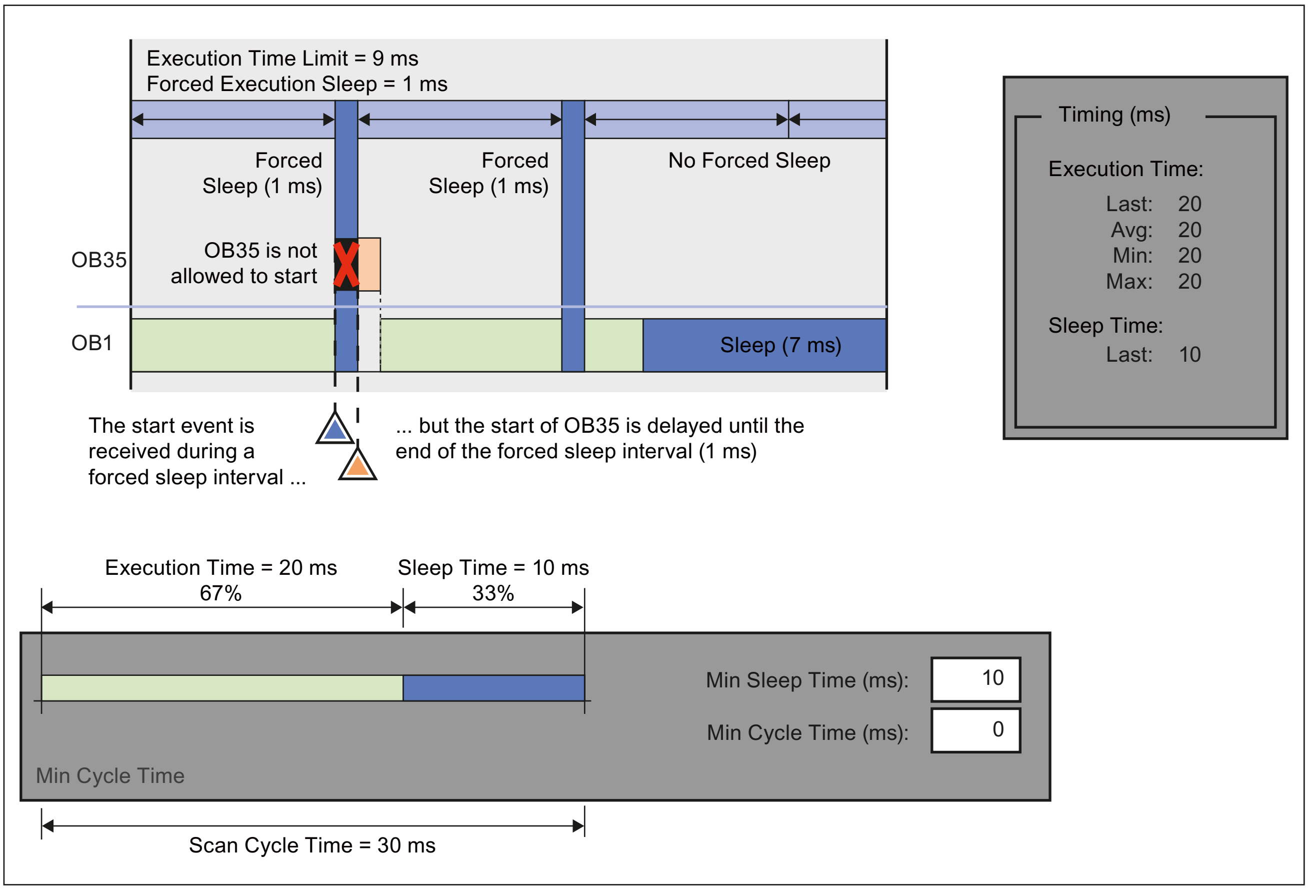
1. Using the STEP 7 Hardware Configuration tool, open the WinLC Properties dialog and select the Cyclic Interrupt tab.
2. Set the priority for OB 36 to 2 (or any other priority lower than the priority for OB 35).
3. Configure OB 36 to execute every 6 ms (by entering 6 in the Execution field).

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.


9 ms Execution Time Limit
9 ms Execution Time Limit
9 ms Execution Time Limit
6 ms
3 ms
OB35
OB36
SFC47
OB1
Because SFC47 inserts a sleep interval, the execution monitor does not force a sleep interval.
Whenever the start event for OB35 is received, OB35 interrupts either OB1 or OB36 and starts without delay.

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

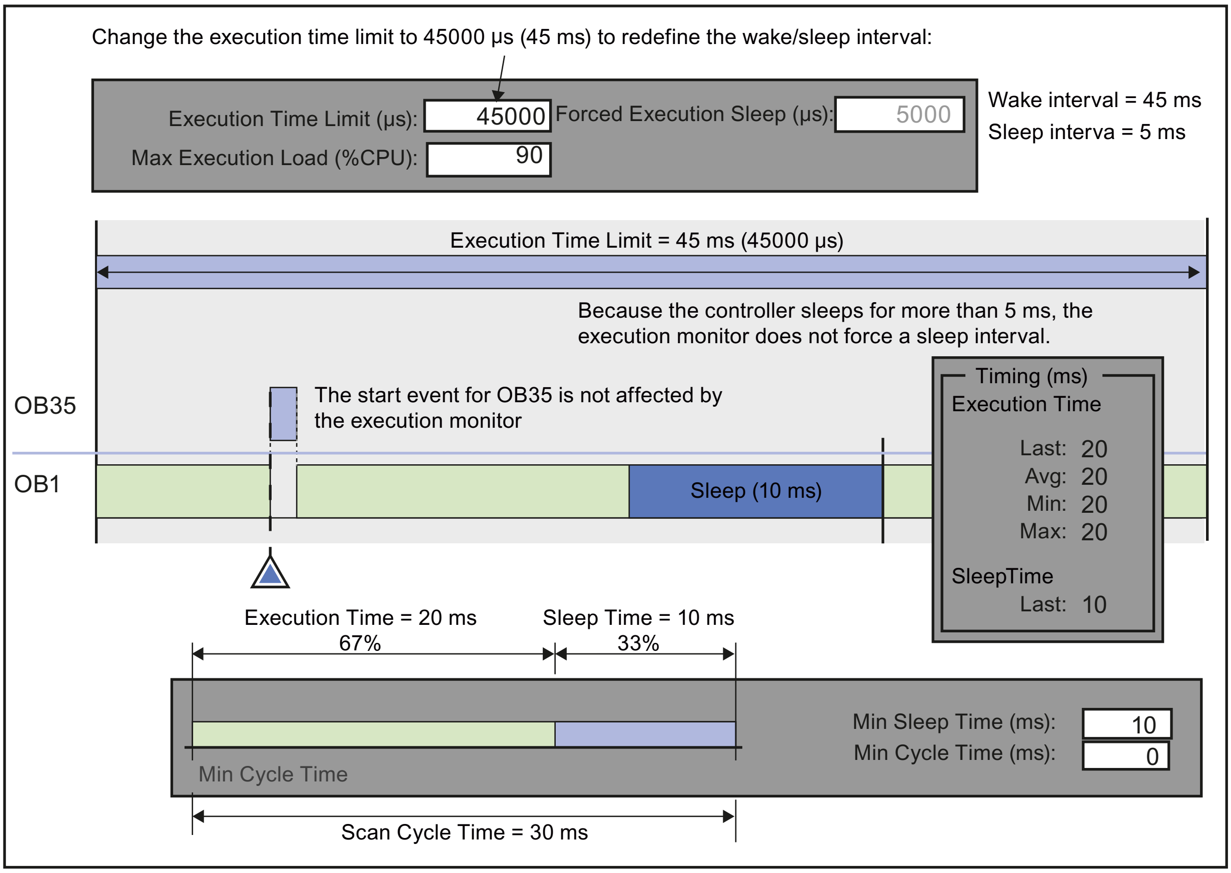
The following figure shows the jitter in the start time of OB 35 and also shows the values displayed by the tuning panel. Notice that the tuning panel shows only the information about OB 1. The tuning panel does not display information about OB 35. For this example, the execution time for OB 1 is 20 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 illustration shows the effect of the changed execution time limit.



Change the execution time limit to 45000 µs (45 ms) to redefine the wake/sleep interval:

Execution Time Limit (µs): 45000
Forced Execution Sleep (µs): 5000
Max Execution Load (%CPU): 90

Wake interval = 45 ms
Sleep interva = 5 ms

Because the controller sleeps for more than 5 ms, the execution monitor does not force a sleep interval.

The start event for OB35 is not affected by the execution monitor

Timing (ms)
Execution Time
Last: 20
Avg: 20
Min: 20
Max: 20
SleepTime
Last: 10

Min Sleep Time (ms): 10
Min Cycle Time (ms): 0

7.5 Isochronous Mode for a Constant Bus Cycle

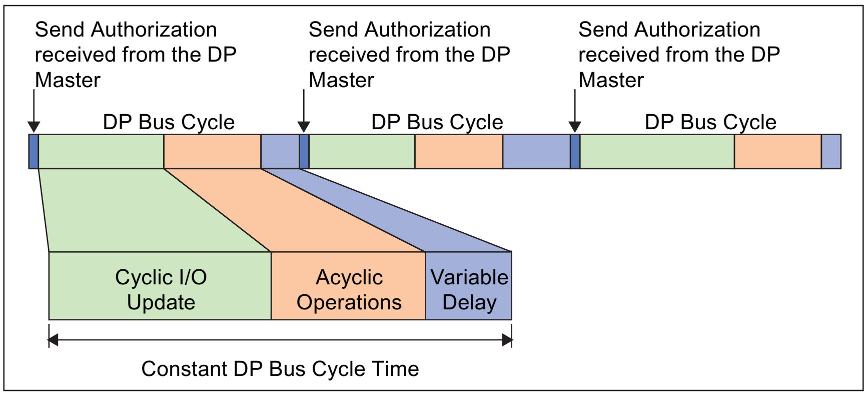
With WinLC RTX, you can operate the DP Master in an isochronous mode to maintain a constant bus cycle time.

Note

WinLC RTX allows you to use isochronous mode on more than one PROFIBUS-DP subnet. For CP 5611/21 and Intel PRO/1000 communication interfaces, your computer must not share the interrupt (IRQ) of the PCI slots (Page 197) 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. Check the technical documentation for your specific PC regarding interrupts and PCI slots.

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.


Send Authorization received from the DP Master
Send Authorization received from the DP Master
Send Authorization received from the DP Master
DP Bus Cycle
DP Bus Cycle
DP Bus Cycle
Cyclic I/O Update
Acyclic Operations
Variable Delay
Constant DP Bus 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 an Intel PRO/1000, CP 5611/21 or a CP 5613 card, hardware revision 6 or higher operating in interrupt mode (Page 197). If you use an Intel PRO/1000 or CP 5611/21 communication interface, the interrupt cannot be shared.

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 (Page 41).

Connecting the Controller to the SIMATIC NET OPC Server 8

8.1 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 (Page 182) 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

Step 1 (Page 178): Station Configuration Editor (SIMATIC NET)
Add the OPC server to the PC station.

Step 2 (Page 179): HW-Config (STEP 7)
Add the OPC server to the hardware configuration in STEP 7.

Step 3 (Page 182): NetPro (STEP 7)
Add an S7 connection for the OPC server to the configuration of WinLC RTX.

Step 4 (Page 185): SIMATIC Manager (STEP 7)
Download the configuration to the controller.

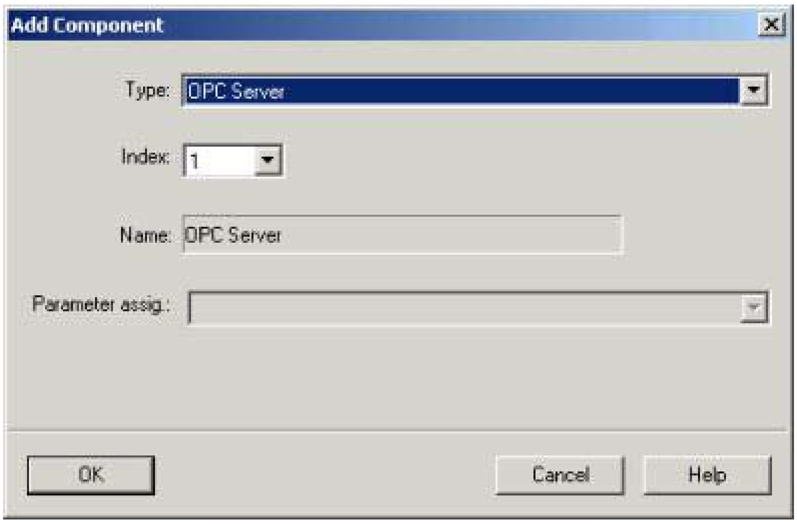
Step 5 (Page 186): OPC Scout (SIMATIC NET)
Connect the controller to the OPC server.

8.2 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:



Add Component

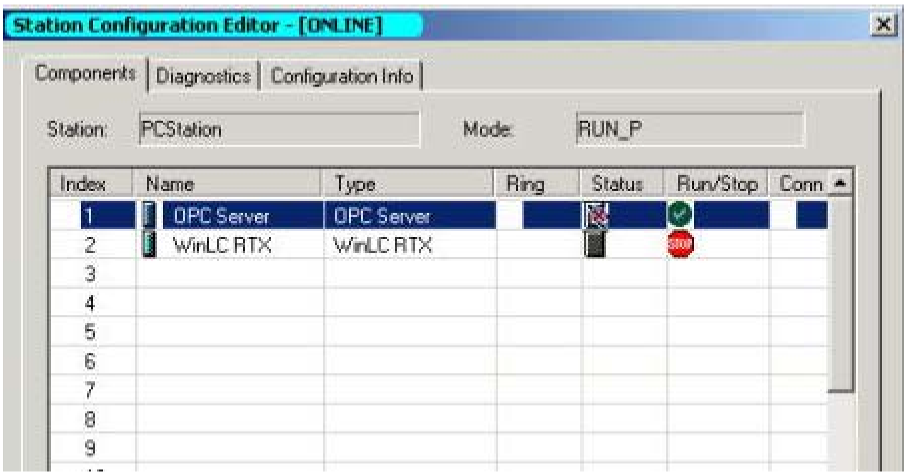
Type: OPC Server

Index: 1

Name: OPC Server

Parameter assig.:

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.



Station Configuration Editor - [ONLINE]

Station: PCStation

Mode: RUN_P

Index	Name	Type	Ring	Status	Run/Stop	Conn
1	OPC Server	OPC Server			✓	
2	WinLC RTX	WinLC RTX			STOP	
3						
4						
5						
6						
7						
8						
9						

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

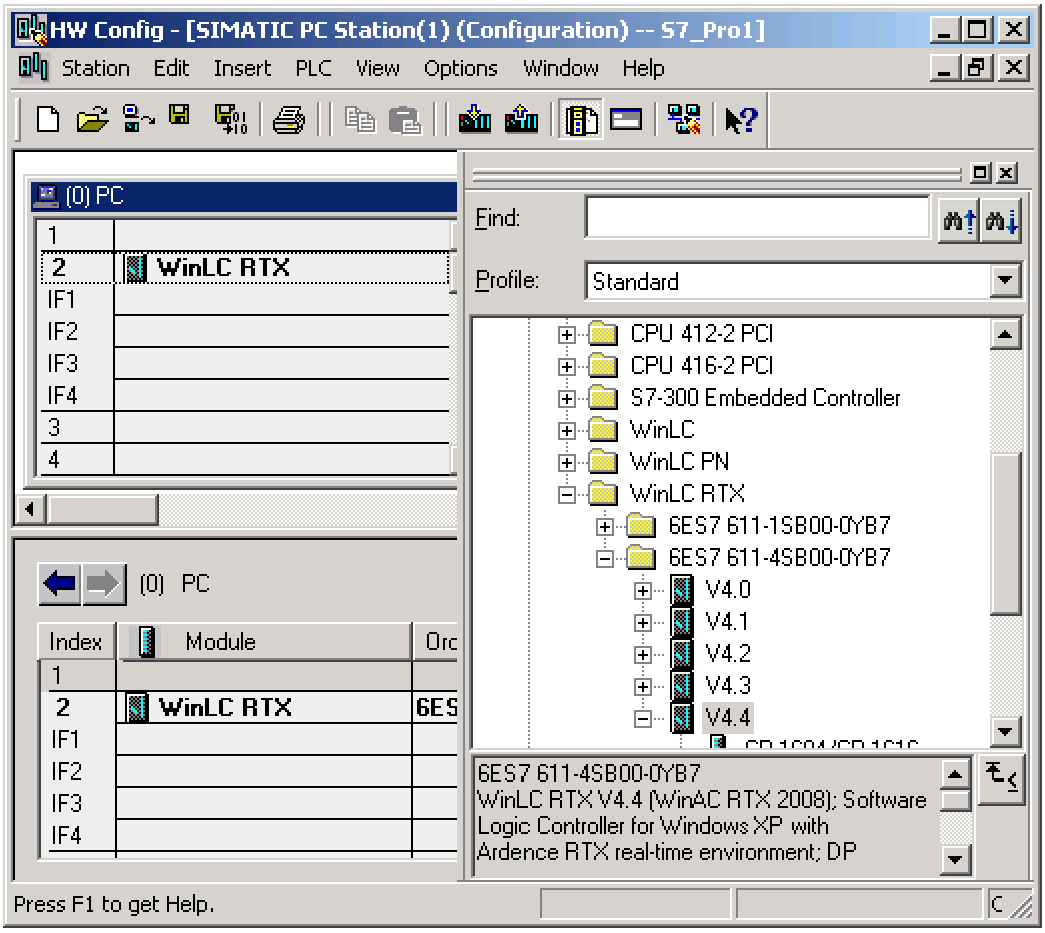
Tool: HW Config (STEP 7)

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 (Page 178).



Profile: Standard

1	
2	WinLC RTX
IF1	
IF2	
IF3	
IF4	
3	
4	

Index	Module	Orc
1		
2	WinLC RTX	6ES
IF1		
IF2		
IF3		
IF4		

6ES7 611-4SB00-0YB7
WinLC RTX V4.4 (WinAC RTX 2008): Software Logic Controller for Windows XP with Ardence RTX real-time environment; DP

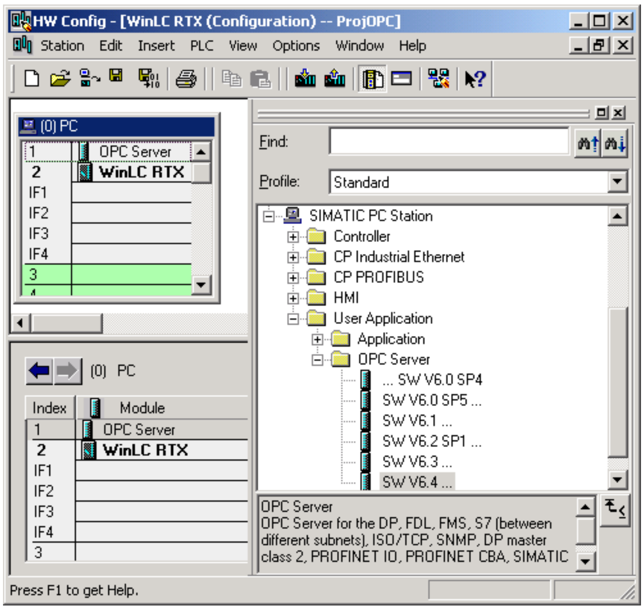
Press F1 to get Help.

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 (Page 178). (For this example, the OPC server is configured for Index 1.)



1	OPC Server
2	WinLC RTX
IF1	
IF2	
IF3	
IF4	
3	

Index	Module
1	OPC Server
2	WinLC RTX
IF1	
IF2	
IF3	
IF4	
3	

Find:

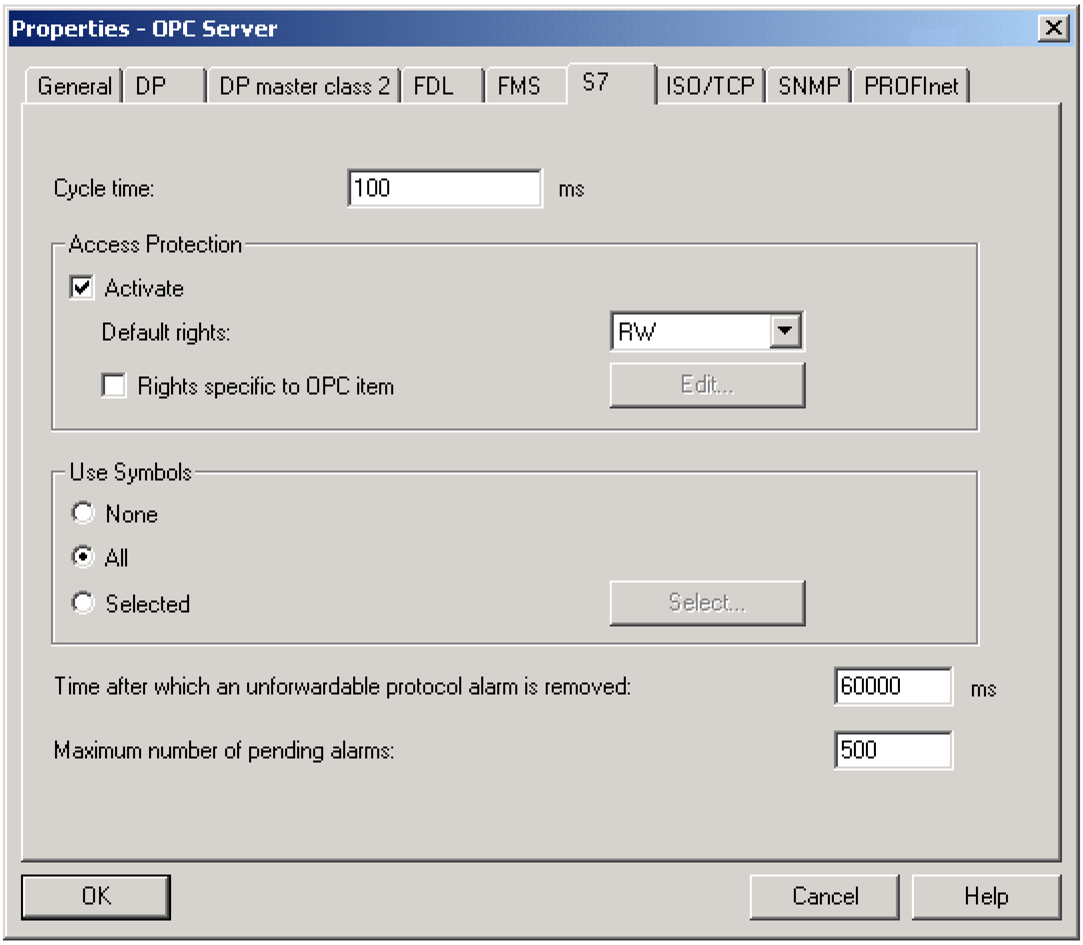
Profile: Standard

OPC Server
OPC Server for the DP, FDL, FMS, S7 (between different subnets), ISO/TCP, SNMP, DP master class 2, PROFINET IO, PROFINET CBA, SIMATIC

Press F1 to get Help.

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 (Page 186) 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.



Cycle time: 100 ms

Access Protection

☑ Activate

Default rights: RW

☐ Rights specific to OPC item

Use Symbols

○ None

● All

○ Selected

Time after which an unforwardable protocol alarm is removed: 60000 ms

Maximum number of pending alarms: 500

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

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

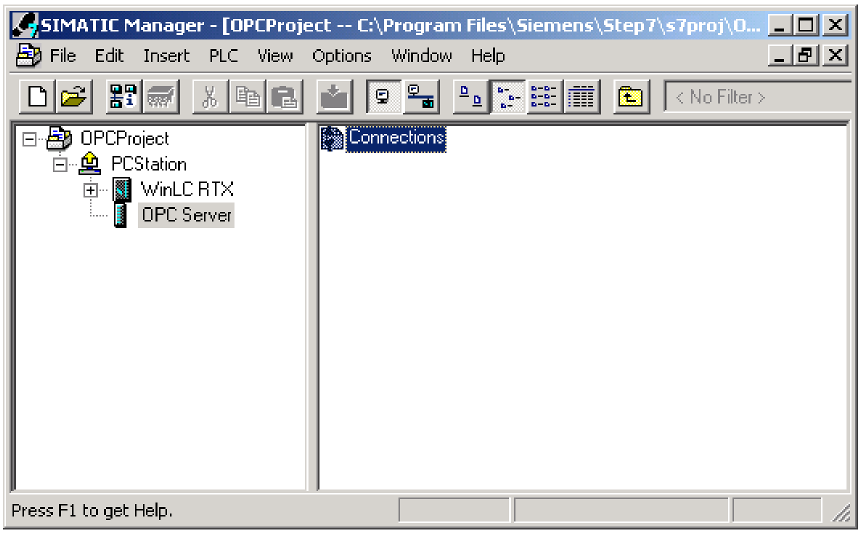
Tool: NetPro (STEP 7)

Task Summary

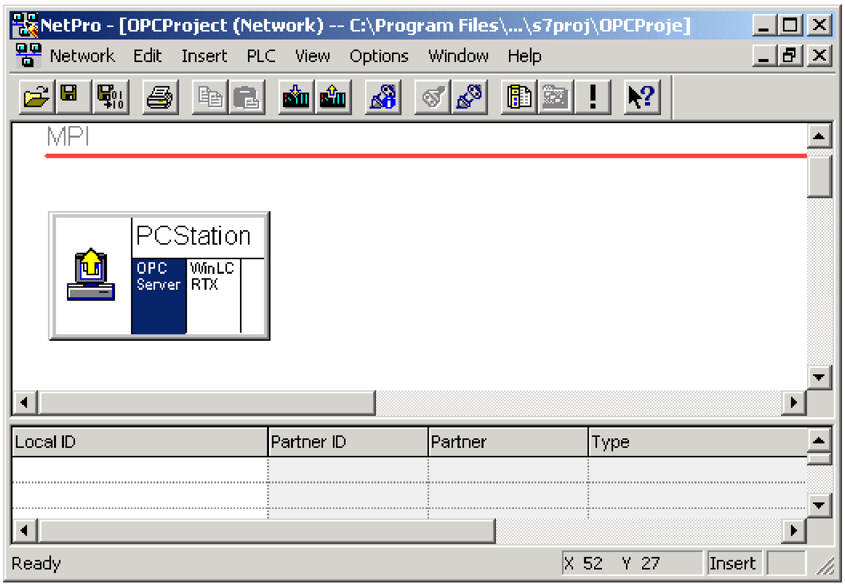
- Configure an S7 connection for the OPC server to the PC Station configuration.
- Assign a Local ID for the OPC server connection.

Configuring an OPC Server Connection in NetPro

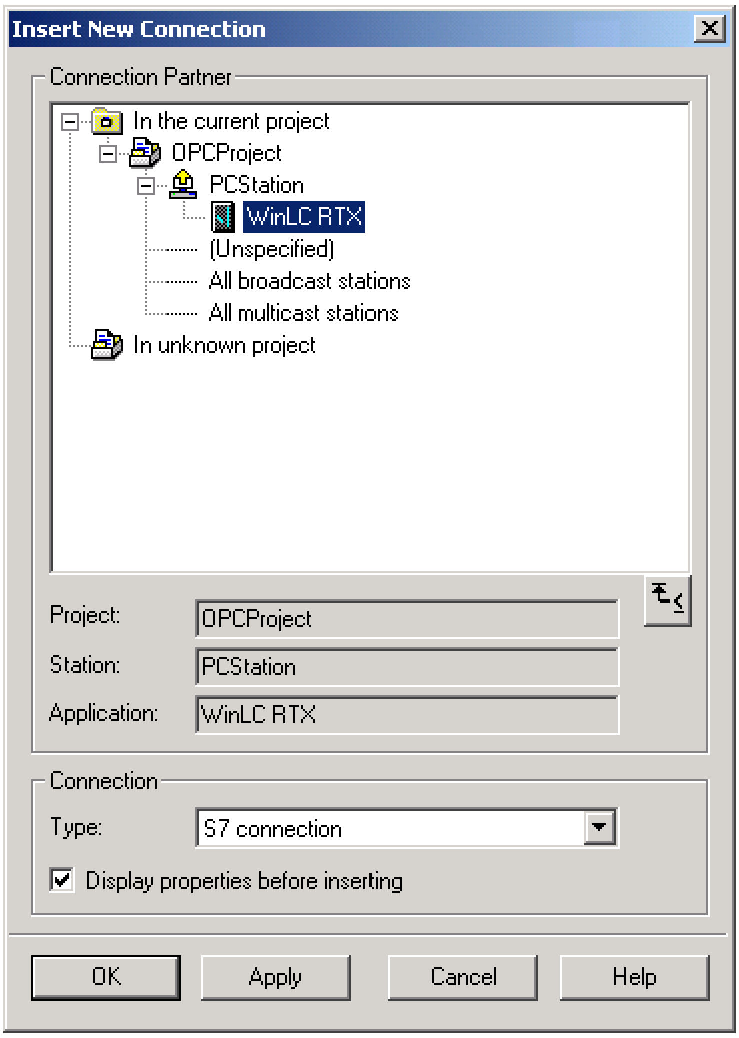
1. In SIMATIC Manager, browse to the OPC server and double-click the Connections icon to open NetPro.


SIMATIC Manager - [OPCProject -- C:\Program Files\Siemens\Step7\s7proj\O...
File Edit Insert PLC View Options Window Help
< No Filter >
OPCProject
PCStation
WinLC RTX
OPC Server
Connections
Press F1 to get Help.

2. Select the OPC Server in the PC station.


NetPro - [OPCProject (Network) -- C:\Program Files\...\s7proj\OPCProje]
Network Edit Insert PLC View Options Window Help
MPI
PCStation
OPC Server
WinLC RTX
Local ID
Partner ID
Partner
Type
Ready
X 52 Y 27
Insert

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.



Project: OPCProject

Station: PCStation

Application: WinLC RTX

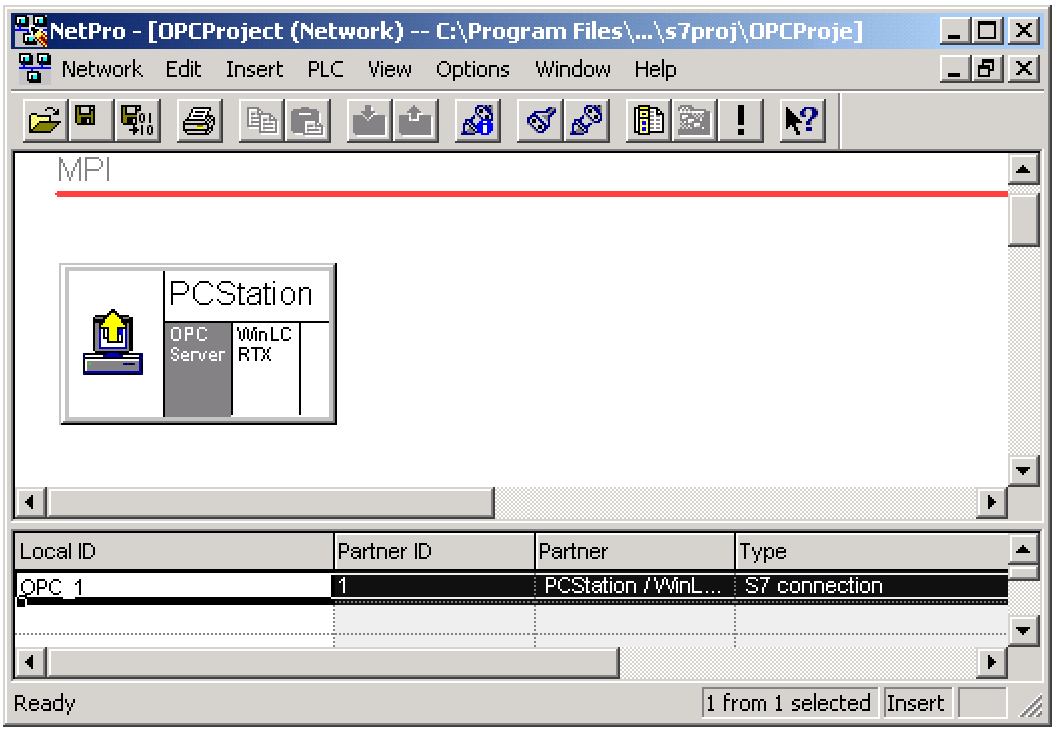
Type: S7 connection

☑ Display properties before inserting

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.


NetPro - [OPCProject (Network) -- C:\Program Files\...\s7proj\OPCProje]
Network Edit Insert PLC View Options Window Help
MPI
PCStation
OPC Server
WinLC RTX

Local ID	Partner ID	Partner	Type
OPC_1	1	PCStation / WinL...	S7 connection

Ready
1 from 1 selected
Insert

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.

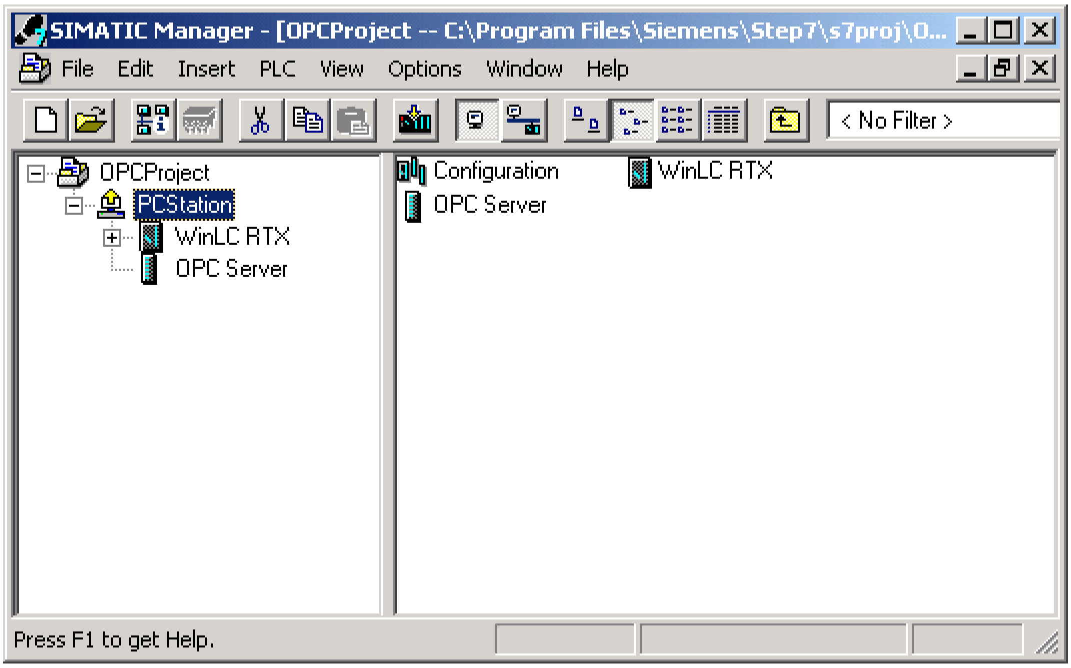
8.5 Step 4: Download the Configuration to the Controller

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 the controller.
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.



SIMATIC Manager - [OPCProject -- C:\Program Files\Siemens\Step7\s7proj\O...
File Edit Insert PLC View Options Window Help
< No Filter >
OPCProject
PCStation
WinLC RTX
OPC Server
Configuration
WinLC RTX
OPC Server
Press F1 to get Help.

8.6 Step 5: Connect the Controller to the OPC server

Tool: OPC Scout

Task Summary

- Add a connection in an OPC project for the SIMATIC NET OPC server.
- Define the items to be accessed through the OPC server.

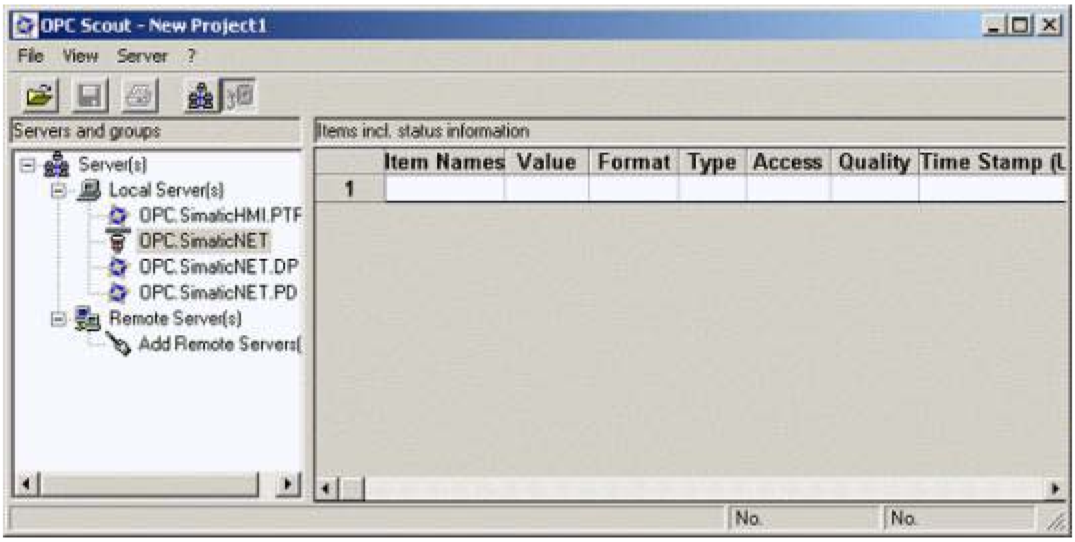
Creating an OPC Project

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

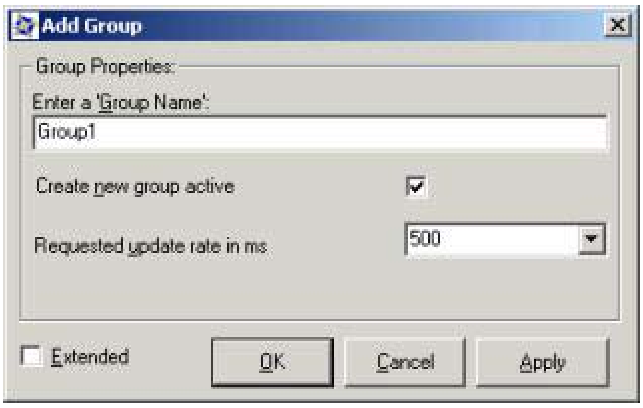
Adding a Connection (Group) in an OPC Project for the OPC Server

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).



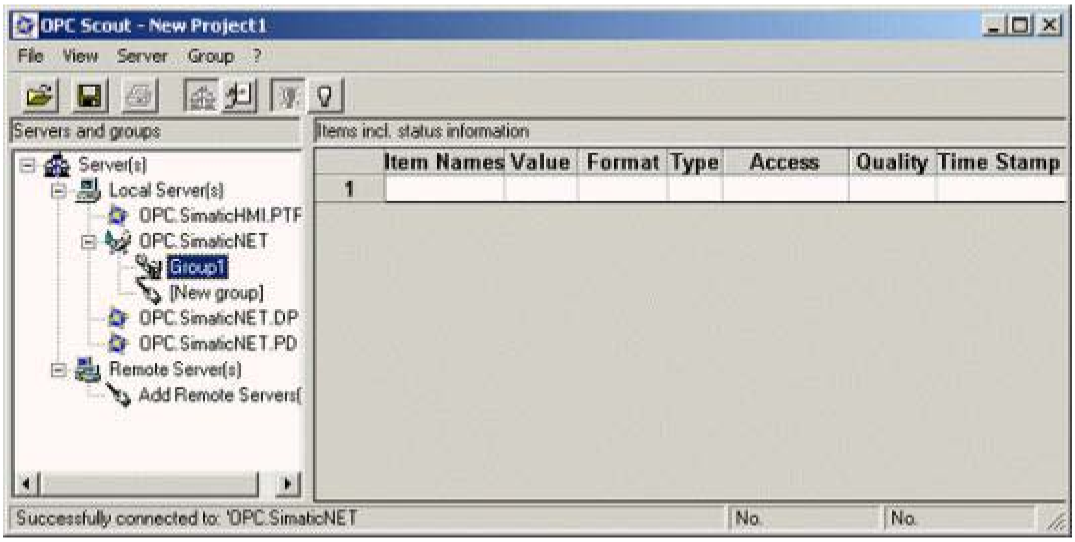
Enter a 'Group Name': Group1

Create new group active ☑

Requested update rate in ms 500

□ Extended

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



	Item Names	Value	Format	Type	Access	Quality	Time Stamp
1							

Successfully connected to: 'OPC.SimaticNET | No. | No.

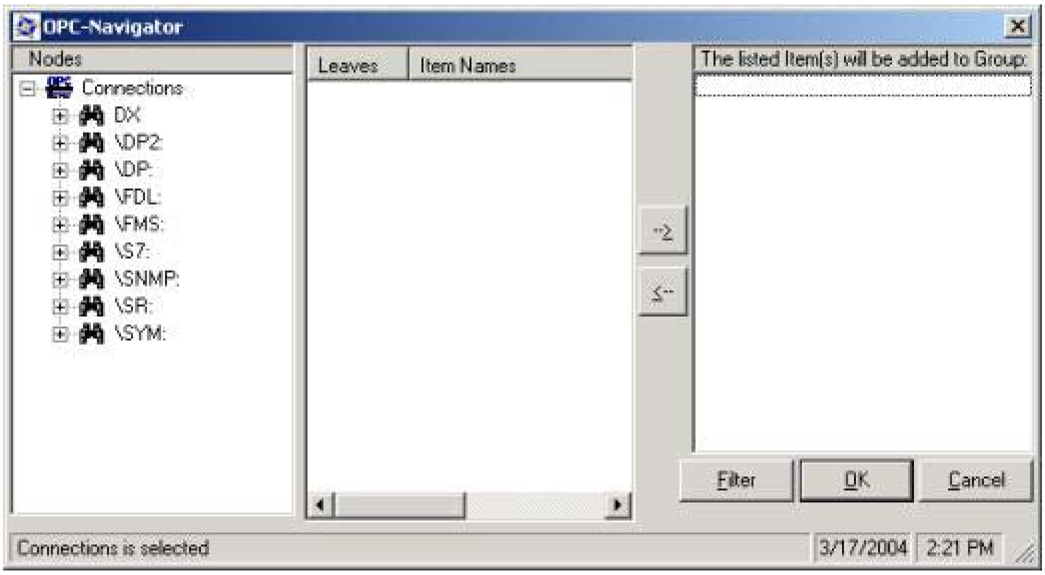
Configuring the Items to be Accessed (Using Absolute Addressing)

Note

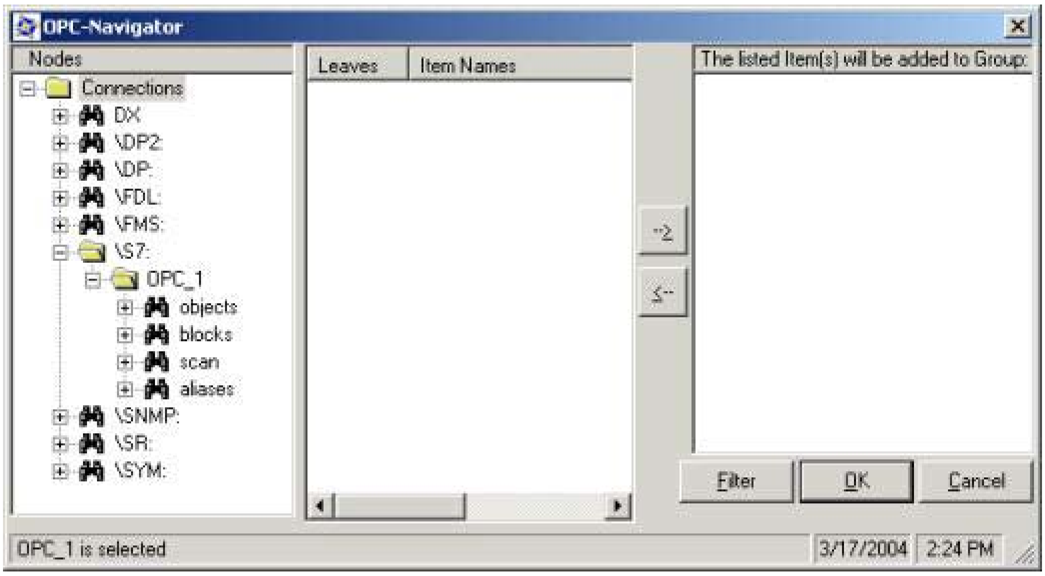
This procedure describes how to use absolute addressing when configuring the OPC server. You can also use the STEP 7 symbol table for connecting the OPC server, as described in the section "Configuring the Items to be Accessed (Using the STEP 7 Symbol Table)".

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

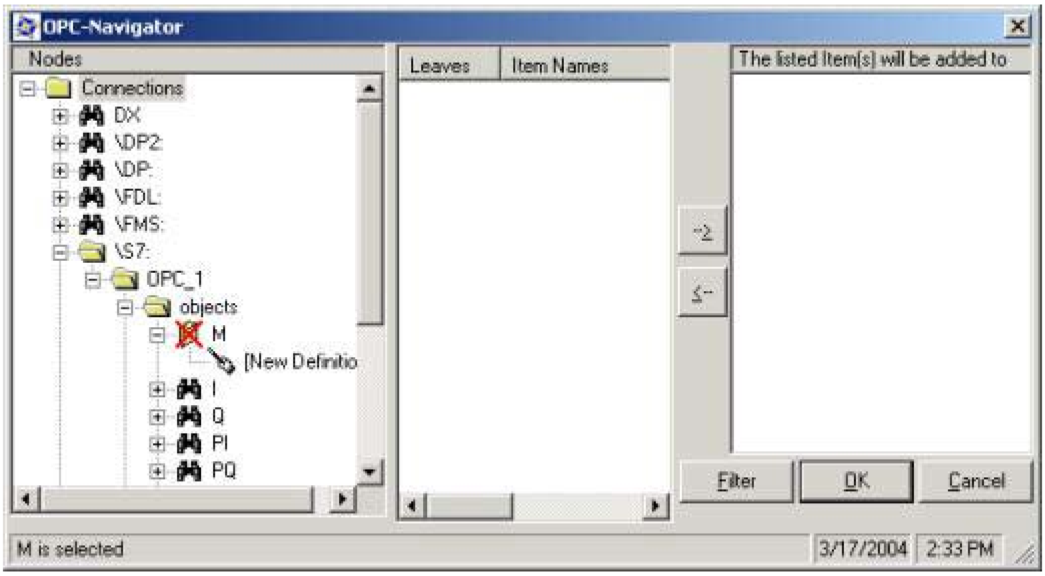
1. Open the OPC Navigator by double-clicking the connection (Group1) for the OPC server.



2. To add an item to be accessed, expand the \S7: folder and select OPC_1.

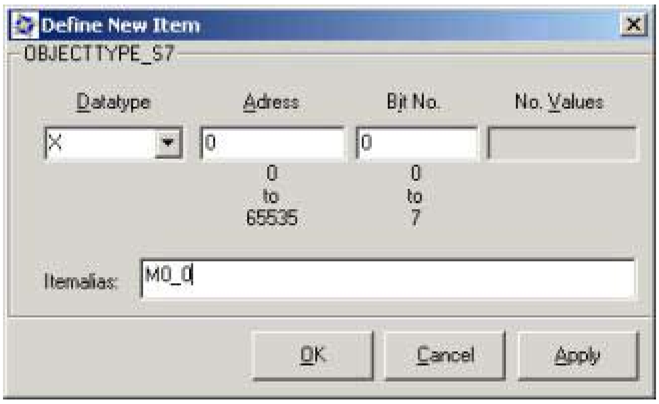


3. To configure access to M 0.0, expand the Objects folder and expand the M folder (for the bit memory area).



M is selected
3/17/2004 2:33 PM

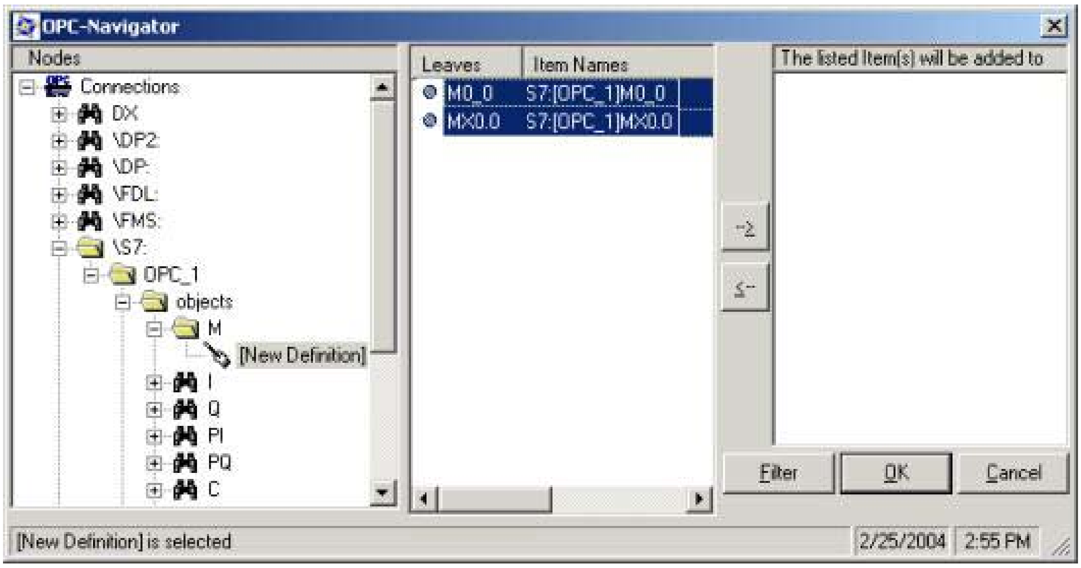
4. Double-click the New Definition icon to open the Define New Item dialog.
5. To define a connection for M0.0, select X (for bit) field from the drop-down list in the Data Type and enter the byte address (0) and bit number (0). (You can also enter an alias for the item.)



Datatype	Adress	Bit No.	No. Values
X	0	0	
	0 to 65535	0 to 7	

Itemalias: M0_0

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

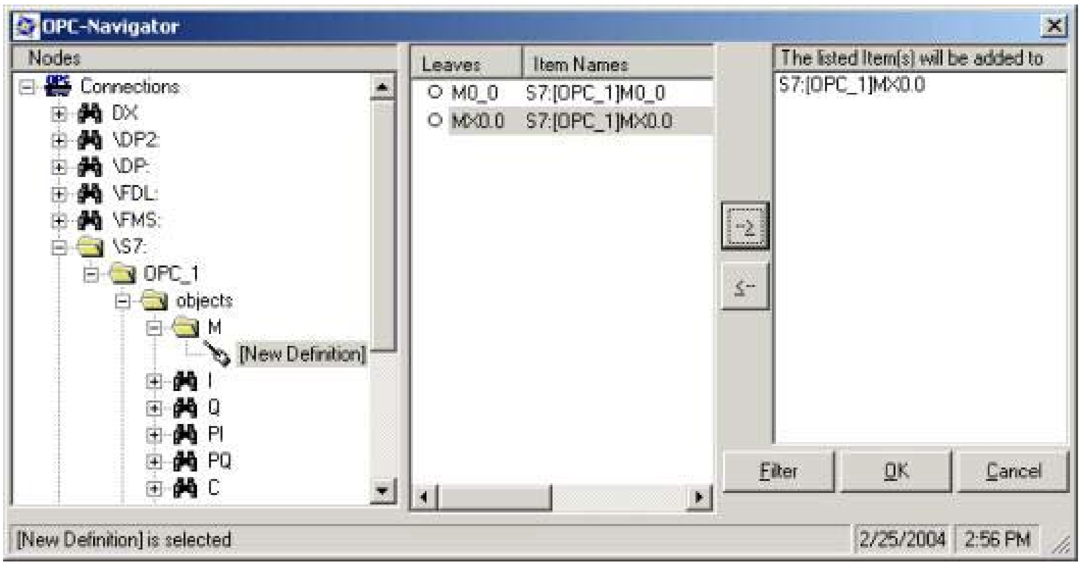


Leaves	Item Names
● M0_0	S7:[OPC_1]M0_0
● MX0.0	S7:[OPC_1]MX0.0

[New Definition] is selected 2/25/2004 2:55 PM

7. 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

8. Select the entry (S7:[OPC_1]MX0.0) and click OK to add the connection for MX0.0 to Group1.



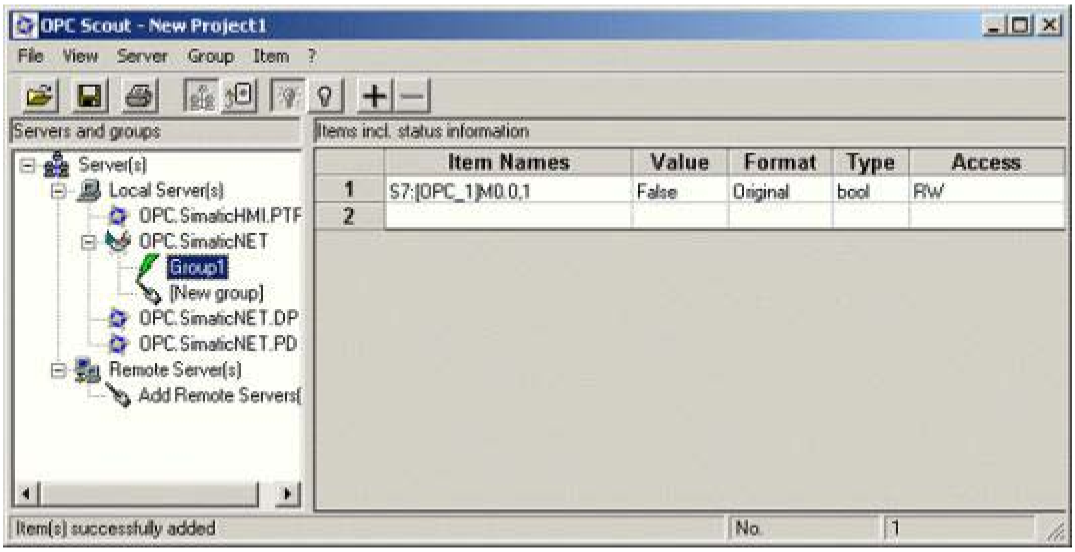
Leaves	Item Names
○ M0_0	S7:[OPC_1]M0_0
○ MX0.0	S7:[OPC_1]MX0.0

The listed Item(s) will be added to: S7:[OPC_1]MX0.0

[New Definition] is selected 2/25/2004 2:56 PM

Result

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.



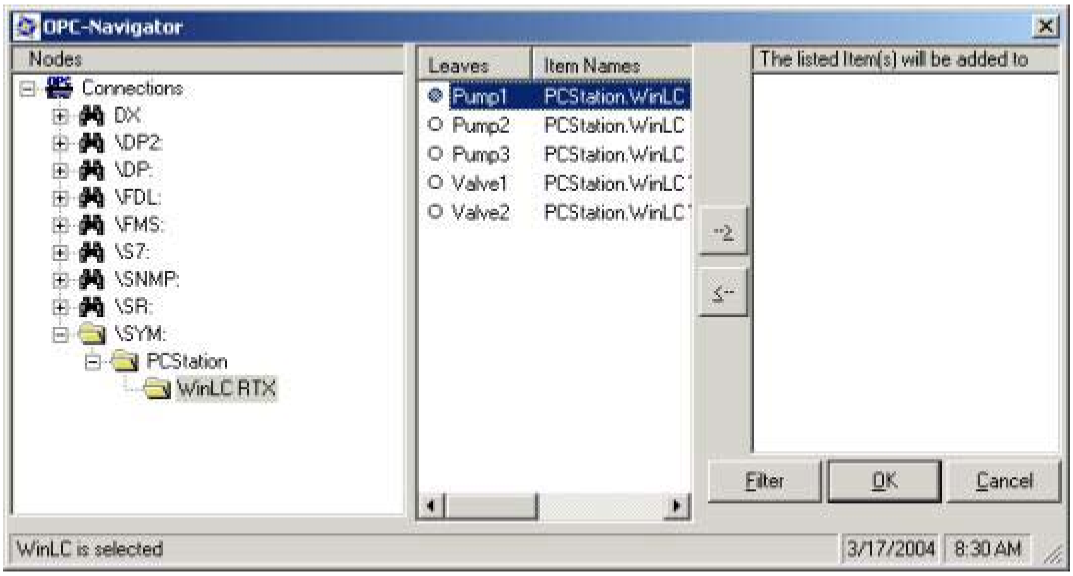
	Item Names	Value	Format	Type	Access
1	S7:[OPC_1]M0.0,1	False	Original	bool	RW
2					

Item(s) successfully added | No. | 1

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.

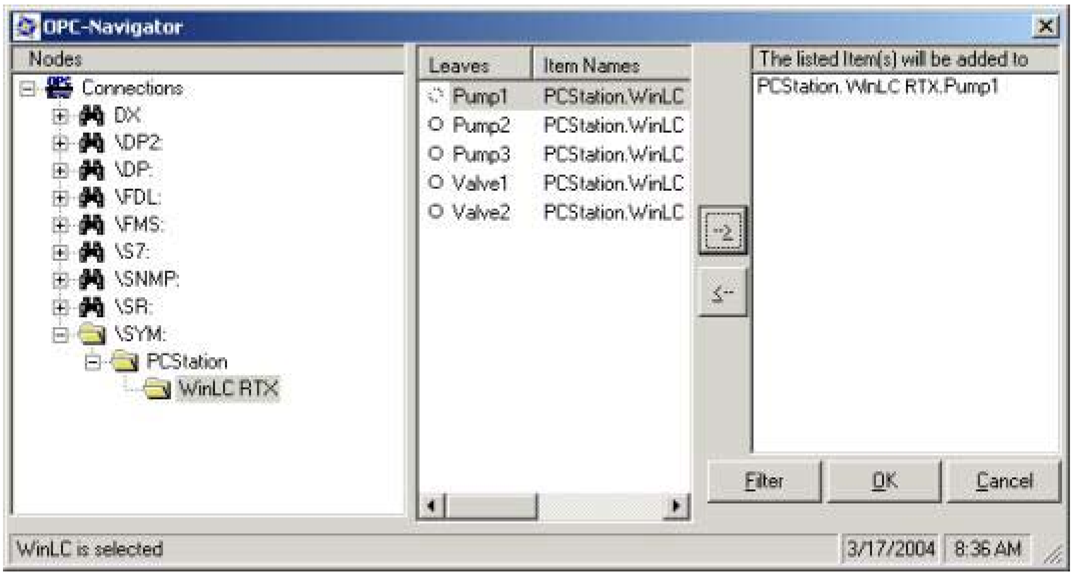


Leaves	Item Names
● Pump1	PCStation.WinLC
○ Pump2	PCStation.WinLC
○ Pump3	PCStation.WinLC
○ Valve1	PCStation.WinLC
○ Valve2	PCStation.WinLC

The listed Item(s) will be added to

WinLC is selected | 3/17/2004 | 8:30 AM

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



Leaves	Item Names
Pump1	PCStation.WinLC
Pump2	PCStation.WinLC
Pump3	PCStation.WinLC
Valve1	PCStation.WinLC
Valve2	PCStation.WinLC

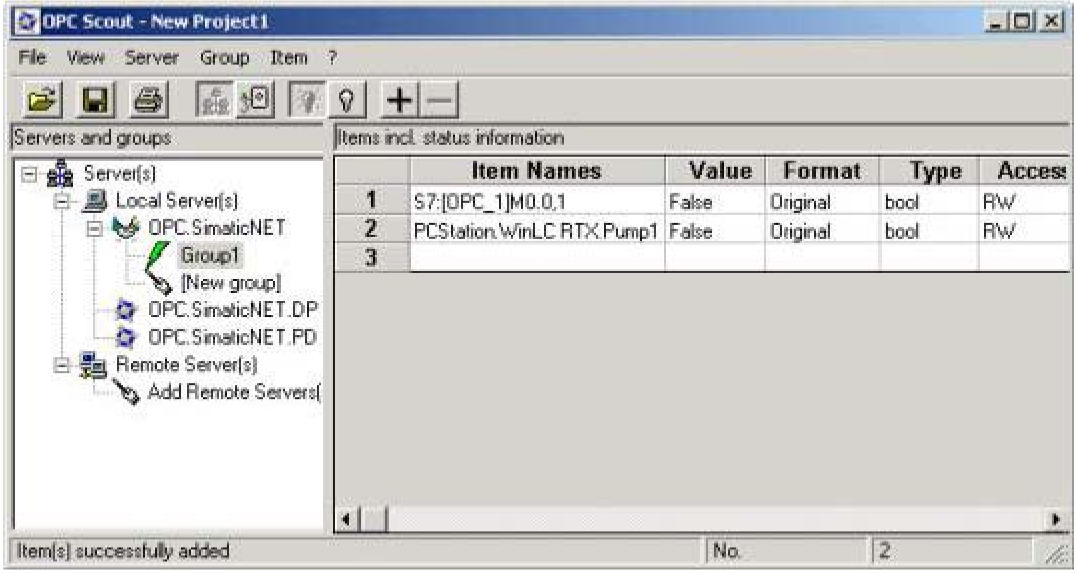
The listed Item(s) will be added to
PCStation. WinLC RTX.Pump1

WinLC is selected 3/17/2004 8:36 AM

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

Result

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



	Item Names	Value	Format	Type	Acces
1	S7:[OPC_1]M0.0,1	False	Original	bool	RW
2	PCStation.WinLC RTX.Pump1	False	Original	bool	RW
3					

Item(s) successfully added No. 2

Reference Information 9

9.1 Technical Data

Order Number

WinLC RTX V4.4 is a component of the WinAC RTX 2008 package:
6ES7 671-0RC06-0YA0.

WinLC RTX communicates with distributed I/O as a PROFIBUS-DP master device or as a PROFINET IO Controller. WinAC RTX with Embedded Controller supports local I/O.

Technical Specifications

You can find all Technical data of WinAC RTX 2008 with the order number 6ES7 671-0RC06-0YA0 at SIMATC Customer Support (http://support.automation.siemens.com/WW/view/en/6ES7671-0RC06-0YA0).

9.2 Changing the Virtual Memory Paging Configuration

Procedure

To change the virtual memory paging configuration, follow these steps:

1. Select System from the Windows Control Panel.
2. From the Advanced tab of the System Properties dialog, click the Settings button for Performance.
3. From the Advanced tab of the Performance Options dialog, click the Change button for Virtual memory.
4. Make any changes you need, and click OK on the dialogs to complete your configuration.

Result

The virtual paging memory configuration has your new settings.

9.3 Troubleshooting

9.3.1 Relevant Information for Ardence RTX

The RTX real-time extensions provide the determinism and performance of a real-time operating system within the 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 communication interface (Page 33) in interrupt mode (varies for different computer manufacturers). If a free interrupt is not available, the communication 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 a Communication Interface in Interrupt Mode

On some computers, Ardence RTX allows a free interrupt for a communication interface. (This varies for different computer manufacturers.) If a free interrupt is not available, CP cards (including integrated PROFIBUS or PROFINET communication interfaces on Siemens PCs) operate only in polled mode and not in interrupt mode, which can affect the performance of the CP card.

Refer to the topic on improving the performance of a communication interface (Page 197).

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.

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.

9.3.2 Troubleshooting Network Problems

The controller panel provides the EXTF and BUSF status indicators that can be used to diagnose problems with the I/O communication 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 submodule 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 or PROFINET IO devices 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 or PROFINET IO devices could not be accessed	Wait for completion of the power-on cycle. If the indicator continues to flash, check the DP slaves or PROFINET IO devices and evaluate the diagnostic data.
—	On	Bus fault (hardware failure)	Check the bus cable or Ethenet cable for an electrical short, a broken wire, or no connection.
On	Off	Diagnostic error	Indicates that a fault condition has not been cleared or that one of the following conditions has occurred: • An I/O module with diagnostic capability has initiated OB 82. • A submodule configuration does not match the configuration downloaded from STEP 7, for example one is CP 5613 and the other is CP 5611/21.
On	On	CP error	Indicates that a configured CP can not be found or is defective.

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.

9.3.3 Improving the Performance of a Communication Interface

To use a PROFIBUS-DP interface in isochronous mode (Page 174), the DP interface must operate in interrupt mode. For maximum performance of a PROFINET IO interface, the interface must also operate in interrupt mode.

Note

WinLC RTX allows you to use isochronous mode (Page 174) on more than one PROFIBUS-DP subnet. For CP 5611/21 and Intel PRO/1000 communication interfaces, 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.

Checking IRQ Assignments

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.

Determining whether a device is Windows-controlled or RTX-controlled

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 a CP 5611/21 or Intel PRO/1000 communication interface 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.

See also

What Is a Communication Interface? (Page 33)

9.3.4 Responding to Diagnostic Events

If an error is detected by the controller, the error condition is logged in the diagnostic buffer (Page 65) 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 of a DP Slave or PROFINET IO device, (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 distributed 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 to read the diagnostic information from a DP slave, or SFB 52 or SFB 54 to read the diagnostic information from a PROFINET IO device.

For information about using OBs, SFC 13, SFC 52, and SFB 54 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".

9.3.5 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.

9.4 System Status List (SSL)

9.4.1 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, which are accessible by SSL_ID. WinLC RTX supports a significant set of the STEP 7 SSL_IDs.

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** menu command. Select your language and then double-click "STEP 7 - System and Standard Functions for S7-300 and S7-400".

Some SSLs are available only when you have configured at least one WinLC RTX submodule.

See also

SSL_ID 0x11 (Module Identification) (Page 201)
SSL_ID 0x12 (CPU Characteristics) (Page 201)
SSL_ID 0x13 (Memory Areas) (Page 201)
SSL_ID 0x14 (System Areas) (Page 202)
SSL_ID 0x15 (Block Types) (Page 202)
SSL_ID 0x19 (Local Module LED Status) (Page 202)
SSL_ID 0x1C (Component Identification) (Page 203)
SSL_ID 0x22 (Interrupt Status) (Page 203)
SSL_ID 0x25 (Process Image Partitions) (Page 203)
SSL_ID 0x32 (Communications Status) (Page 204)
SSL_ID 0x74 (LED Status) (Page 204)
SSL_ID 0x90 (DP Master System) (Page 205)
SSL_ID 0x91 (Module Status) (Page 205)
SSL_ID 0x92 (Rack and Station Status) (Page 206)
SSL_ID 0x94 (Station status) (Page 206)
SSL_ID 0x95 (Expanded DP Master or PN IO System Status) (Page 207)
SSL_ID 0x96 (Additional PN IO or DP Status Information) (Page 207)
SSL_ID 0xA0 (Diagnostic Buffer) (Page 207)
SSL_ID 00B1, 00B2, 00B3, and 00B4 (Module Diagnostics) (Page 208)

9.4.2 SSL_ID Descriptions

9.4.2.1 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

9.4.2.2 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	

9.4.2.3 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)

9.4.2.4 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	

9.4.2.5 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

9.4.2.6 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 (Page 204) 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	

9.4.2.7 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	

9.4.2.8 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 RTX (Page 107) and Organization Blocks (OBs) (Page 108).

9.4.2.9 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	

9.4.2.10 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

9.4.2.11 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 (Page 202).

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) 0021: MAINT (maintenance is required)

9.4.2.12 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	

9.4.2.13 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 DP interface module or PROFINET interface module
0D91	Specific station, identified either by rack/station, or one of these means: • PROFIBUS DP: DP master identifier, or DP master identifier with station number • PROFINET IO station number and last two places in the PNIO subsystem ID	Station identifier: Features and parameters for all the modules of the specified station (DP or PROFINET)
0E91	Module status information of all assigned modules	Irrelevant
0F91	Header information only	

9.4.2.14 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

9.4.2.15 SSL_ID 0x94 (Station status)

0094, 0194, 0294, 0694, 0794, 0F94 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0094	Specified status of stations of a central rack, PN IO or DP subsystem	0: central module 1 to 31: distributed module on PROFIBUS-DP 100 to 115: distribute module on PROFINET IO
0194	Activation status of a station	1 to 31: distributed module on PROFIBUS-DP 100 to 115: distribute module on PROFINET IO
0294	Actual status of a station	0: central module 1 to 31: distributed module on PROFIBUS-DP 100 to 115: distribute module on PROFINET IO
0694	All faulty stations of a central rack, PN IO or DP system	0: central module 1 to 31: distributed module on PROFIBUS-DP 100 to 115: distribute module on PROFINET IO
0794	All faulty stations of a central rack, PN IO or DP system; using additional bits to determine whether a station is faulty	0: central module 1 to 31: distributed module on PROFIBUS-DP 100 to 115: distribute module on PROFINET IO
0F94	Header information only	

9.4.2.16 SSL_ID 0x95 (Expanded DP Master or PN IO System Status)

0195, 0F95 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0195	Specific DP master or PN IO system	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) PN IO system identifier: Properties for the stations of the specified IO Controller (such as rack number, slot of the IO controller, type of IO Controller, and logical start address)
0F95	Header information only	

9.4.2.17 SSL_ID 0x96 (Additional PN IO or DP Status Information)

0696, 0C96 (hexadecimal)

SSL_ID	Sublist	Index and Contents of the Record
0696	Status info of all configured submodules of a module	Logical base address of a module (PN IO modules only)
0C96	Status info of a submodule	Logical base address of a module

9.4.2.18 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	

9.4.2.19 SSL_ID 00B1, 00B2, 00B3, and 00B4 (Module Diagnostics)

00B1, 00B2, 00B3, 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
00B2	All of the diagnostic information for a specific module, identified by its rack and slot (S7-mEC modules only)	Rack and slot: Complete 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

Note: 00B2 is only available for the S7-mEC.

Glossary

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.

Cold restart

The controller executes OB102 before starting the free cycle (OB1). 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.

Communication interface

CP card, Siemens PC built-in PROFIBUS interface, or Industrial Ethernet interface that WinLC RTX uses for communication.

Control program

The control program is the application program created with STEP 7 and downloaded to the controller for execution. The control program includes all organization blocks (such as OB1 or OB35) and the other logic blocks that they call, including functions (FCs), system functions (SFCs), function blocks (FBs), and system function blocks (SFBs).

CP card

Communications Processor card: (See topic "What Is a Communication Interface?")

Cycle time

The cycle time is the time required to execute the complete scan cycle, which includes the execution of OB1 and the minimum sleep time

Deterministic behavior

Predictability of execution time and response time.

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 control program. This includes executing OB1 and updating the I/O.

Execution time limit

The execution time limit defines the maximum amount of time allowed for the controller to execute the control program. The execution monitor uses this value and the maximum execution load to calculate the forced execution sleep time.

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 OB1, 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.)

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, S7 applications, and PROFINET IO.

Isochronous mode

Configuration of DP cycle that yields a constant bus cycle time. (See topic "Isochronous Mode for a Constant Bus Cycle".)

Jitter

Jitter is the difference in the actual cycle time from the configured minimum scan time.

Load memory

Memory area (RAM) allocated for all of the blocks downloaded from STEP 7 excluding the symbol table and comments.

Maximum execution load

The maximum execution load is the 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

The minimum cycle time is the 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

The minimum sleep time is the 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

The 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.

Non-deterministic behavior

Lack of predictability of execution time and response time associated with jitter. (See topic "What Causes Jitter?")

OP

Operator panel.

Organization block (OB)

The OBs represent the interface between the operating system and the control program. Called by the operating system, they control cyclic and interrupt-driven program execution, startup behavior of the controller and error handling.

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 control program. Organization blocks (OBs) are ranked by priority class. Higher priority OBs interrupt lower priority OBs. The free cycle (OB1) 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.

PROFINET CBA

PROFINET Component Based Automation

PROFINET IO

Communications network protocol used to communicate to PROFINET IO devices.

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 control program and variables. The two restart methods are Cold Restart (OB102) and Warm Restart (OB100).

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.

S7 communication

Communication between controllers on the network, hardware or software, using the S7 communication functions. (See topic "Communication Blocks.")

S7 routing

Communication 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 OB1and all other OBs, and completing the sleep time requirement.

Sleep time

The sleep time is the 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 OB.

Station configuration editor

Tool, accessible from taskbar, for configuring the PC Station: for WinLC RTX this includes WinLC Properties, submodule assigments, and submodule diagnostics for some communication 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)

An SFC is a preprogrammed function that is integrated as a part of the operating system of the controller and is not downloaded as part of the control program. You can call the SFC in your control program. Like a function (FC), an SFC is a block "without memory."

System function block (SFB)

An SFB is a function block that is integrated as a part of the operating system of the controller and is not downloaded as part of the control 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 control program.

Time synchronization

The ability to broadcast a system standard time from a single source to all devices within the system so that they can set their own clocks to the standard time.

Time synchronization service (WinAC Time Synchronization)

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.)

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?")

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

The controller executes OB100 before starting the free cycle (OB1). 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