A large yellow circle is positioned to the left of the title, and a thick yellow vertical bar with a rounded top extends from the bottom of the circle down the left side of the page.

Wonderware SIDirect DAServer User's Guide

All rights reserved. No part of this documentation shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Invensys Systems, Inc. No copyright or patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this documentation, the publisher and the author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.

The information in this documentation is subject to change without notice and does not represent a commitment on the part of Invensys Systems, Inc. The software described in this documentation is furnished under a license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of these agreements.

© 2003-2004, 2010-2012, 2014 by Invensys Systems, Inc. All rights reserved.

Invensys Systems, Inc.
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

<http://www.wonderware.com>

For comments or suggestions about the product documentation, send an e-mail message to ProductDocumentationComments@invensys.com.

All terms mentioned in this documentation that are known to be trademarks or service marks have been appropriately capitalized. Invensys Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

Alarm Logger, ActiveFactory, ArchestrA, Avantis, DBDump, DBLoad, DT Analyst, Factelligence, FactoryFocus, FactoryOffice, FactorySuite, FactorySuite A², InBatch, InControl, IndustrialRAD, IndustrialSQL Server, InTouch, MaintenanceSuite, MuniSuite, QI Analyst, SCADAAlarm, SCADASuite, SuiteLink, SuiteVoyager, WindowMaker, WindowViewer, Wonderware, Wonderware Factelligence, and Wonderware Logger are trademarks of Invensys plc, its subsidiaries and affiliates. All other brands may be trademarks of their respective owners.

Contents

	Welcome	7
	Documentation Conventions	8
	Technical Support	8
Chapter 1	Getting Started.....	11
	Overview	11
	Before You Begin	12
	Supported Client Protocols	13
	OPC	13
	SuiteLink	13
	DDE/FastDDE	14
	Supported Device Protocols	14
	Special Pre-Configuration for S7-1200 and S7-1500 PLCs	15
Chapter 2	Setting Up Your DAServer	17
	Checklist for Setting up the SIDirect DAServer	18
	Finding Your DAServer in the SMC	19
Chapter 3	Configuration.....	21
	Configuring the SIDirect DAServer	21
	SIDirect Hierarchy in the DAServer Manager	24

	Device Redundancy	31
	Run-time Behavior	31
	Configuring Device Redundancy	32
	Using the S7 Tag Creator	35
Chapter 4	Device Groups and Device Items	37
	Device Group Definitions	38
	Device Item Definitions	43
	Scan-Based Message Handling	46
	Unsolicited Message Handling	47
Chapter 5	Managing Your DAServer.....	49
	Configuring the DAServer as Service	50
	Configuring the DAServer as Not a Service	50
	Archiving Configuration Sets	50
	Activating/Deactivating the DAServer	53
	In-Proc/Out-of-Proc	53
	Hot Configuration	54
	Demo Mode	54
Chapter 6	Accessing the Data in Your DAServer	57
	Accessing Items Using the OPC Communications Protocol	57
	Accessing Items Using the DDE/SuiteLink Communications Protocol	59
Chapter 7	Item Naming	61
	Data Blocks and Instance Blocks	61
	Flag Bytes	65
	Input Bytes	69
	Output Bytes	73
	Peripheral Input Bytes	77
	Peripheral Output Bytes	82
	Counters	86
	Timers	88
	Block Items	89
	Read-Only Block Items	89
	Write-Only Block Items	92
	Alarms and Events	94
	Alarms and Events Terms	100
	Specific S7-200 Item Syntax	101

V Memory	101
Analog Input	104
Analog Output	105
Stage Memory	106
Special Memory	107
High-Speed Counter	109
S7-200 Timers	110
Specific S7-1200 and S7-1500 Item Syntax	111
S7-1200 and S7-1500 Data Blocks and Instance Blocks	111
LREAL Data Type and Syntax	114
DTL Data Type and Syntax	115
High-Speed Counters	120
Conversions and Suffixes of Items	120
Endian Conversion	120
Suffix BCD	120
Suffix DT	121
Suffix KT	121
Suffix S5T	121
Suffix TR	122
Suffix D	122
Suffix T	123
Suffix TOD	123
DAServer Standard System Items	123
DAServer Global System Item	125
DAServer Device-Specific System Items	126
DAServer Device-Group-Specific System Items	128
DAServer Redundant Device Specific System Items	132
DAServer Global System Item	134
Generic OPC Syntax	134
VT_Array Syntax in Application Server	135
 Chapter 8 Troubleshooting	 137
Finding the DAServer Version Number	137
Monitoring Connectivity Status with the PLC	138
Monitoring the Status of DAS Conversations	139
Using DDEStatus and IOStatus in Excel	139
Reading Values from the DAServer into Excel	139
Writing Values to the DAServer from Excel	140
Debugging Communications Between the SIDirect DAServer and the PLC	141
Client Groups	142
Structure	143

Transactions	144
Statistics	144
Messages	144
Device Groups	146
Diagnostics and Error Tracing	147
Diagnostics Facility	147
DASTrace Diagnostic Messages	149
Error Tracing with the Wonderware Logger	151
Error Messages, Trace Messages, Error Codes, and Warnings	152
Data Conversion	176
Quality Settings	176
 Chapter 9 Reference	 179
DAServer Architecture	179
DAServers	180
Component Environments	181
 Appendix A Tested DAServer Hardware and Firmware ...	 183
 Index.....	 187

Welcome

This guide describes using and customizing the SIDirect DAServer application.

You can view this document online or you can print it, in part or whole, by using the print feature in Adobe Acrobat Reader.

This guide assumes you know how to use Microsoft Windows, including navigating menus, moving from application to application, and moving objects on the screen. If you need help with these tasks, see the Microsoft online Help.

In some areas of the SIDirect DAServer, you can also right-click to open a menu. The items listed on this menu change, depending on where you are in the product. All items listed on this menu are available as items on the main menus.

The SIDirect DAServer is a Microsoft® Windows® application program that acts as a communications protocol server. It provides other Windows application programs with access to data within the Siemens S7-200/300/400/1200/1500 family of PLCs. The Industrial Application Server DAServer provides access to a Siemens PLC through an off-the-shelf standard Ethernet network interface card in the computer.

Note: The DAServer can support multiple network interface cards in a system.

The SIDirect DAServer supports the Siemens S7-200/300/400/1200/1500 family of hardware and firmware listed in "Tested DAServer Hardware and Firmware" on page 183. It does **not** require the following items:

- Siemens PC adapter cards
- Siemens CP (Communications Processor) cards that reside in the PC
- Siemens SIMATIC NET library

While the SIDirect DAServer is primarily intended for use with the Wonderware InTouch® software (Version 9.0, 9.5, 10.0, or later), it can be used by any Microsoft Windows program capable of acting as a DDE, FastDDE, SuiteLink™, or OPC client.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and file names.
Bold	Menus, commands, dialog box names, and dialog box options.
Monospace	Code samples and display text.

Technical Support

Wonderware Technical Support offers a variety of support options to answer any questions on Wonderware products and their implementation.

Before you contact Technical Support, refer to the relevant section(s) in this documentation for a possible solution to the problem. If you need to contact technical support for help, have the following information ready:

- The type and version of the operating system you are using.
- Details of how to recreate the problem.
- The exact wording of the error messages you saw.
- Any relevant output listing from the Log Viewer or any other diagnostic applications.

- Details of what you did to try to solve the problem(s) and your results.
- If known, the Wonderware Technical Support case number assigned to your problem, if this is an ongoing problem.

Chapter 1

Getting Started

This section is an overview of the Wonderware® SIDirect DAServer (Data Access Server), including the application-level and bus-level communications protocols, item naming conventions, and server features.

Overview

The DAServer is one component of a software system that connects your software application with information on the factory floor.

This DAServer documentation covers only the information you need to configure and run the DAServer component. See the documentation that comes with the related components for details on their operation. You can find installation instructions in a help file on the distribution CD.

You use the DAServer Manager to configure, activate, and troubleshoot the DAServer. The DAServer Manager is located in the System Management Console (SMC).

This documentation describes some of the features of the DAServer Manager. See the *DAServer Manager User's Guide* to find more information on:

- Global parameters
- Configuration sets
- Time zone features
- Icon definitions
- Activation/deactivation

- Configuring as a service
- Importing/exporting device items
- Standard diagnostics

You can troubleshoot problems with the DAServer using the ArchestrA Log Viewer, a snap-in to the SMC. See the Log Viewer help file to find information on:

- Viewing error messages.
- Determining which messages are shown.
- Bookmarking error messages.

You may also be able to troubleshoot problems using your client application, such as the Wonderware InTouch HMI software. The client application can use system device items to determine the status of nodes and the values of some parameters.

Before You Begin

Before configuring the DAServer, verify the following items:

- A PC is set up with the necessary network cards, and is connected to the necessary networks.
- The Windows administration account is created or identified.
- The DAServer and any other Wonderware software such as the DAServer Manager is installed with the proper licenses. For more information, see the License Utility documentation on the distribution CD.
- The client software is installed.
- The device(s) is/are connected (networked) and, if necessary, programmed.

Before configuring the DAServer, you should know:

- The device network configuration and addresses.
- Which data items are needed for the client application.
- The device name/topic name/group name.
- The desired update intervals.

Supported Client Protocols

The SIDirect DAServer communicates with clients and PLCs using following different communications protocols:

- OPC
- SuiteLink
- DDE/FastDDE

For more information about the DAServer architecture, see "Reference" on page 179.

OPC

OPC (OLE for Process Control) is a non-proprietary set of standard interfaces based upon the OLE/COM technology developed by Microsoft. This standard enables interoperability between automation/control applications, field systems/devices, and business/office applications. Avoiding the traditional requirement of software/application developers to write custom drivers to exchange data with field devices, OPC defines a common, high-performance interface that permits this writing custom drivers to be done one time, and then to be easily reused by HMI, SCADA, control, and custom applications.

Over the network, OPC uses DCOM (Distributed COM) for remote communications.

SuiteLink

SuiteLink uses a TCP/IP-based protocol to meet industrial needs such as data integrity, high throughput, and easier diagnostics. This TCP/IP standard is supported on Windows operating systems.

SuiteLink is not a replacement for DDE orFastDDE. The protocol used between a client and a server depends on your network connections and configurations.

SuiteLink provides the following features:

- Value Time Quality (VTQ) places a timestamp and quality indicator on all data values delivered to VTQ-aware clients.
- Extensive diagnostics of the data throughput, server loading, computer resource consumption, and network transport are made accessible through the operating system's performance monitor. This feature is critical for the operation and maintenance of distributed industrial networks.

- Consistent high data volumes can be maintained between applications if the applications are on a single node or distributed over a large node count.
- The network transport protocol is TCP/IP using the Microsoft standard WinSock interface.

DDE/FastDDE

DDE/FastDDE communication protocols allow communication between a client and a server. DDE protocol is developed by Microsoft whereas FastDDE protocol is proprietary to Wonderware.

DDE

DDE is a communications protocol that allows applications in the Windows environment to send and receive data and instructions to and from each other. It implements a Client/Server relationship between two concurrently running applications. The server application provides the data and accepts requests from any other application interested in its data. Requesting applications are called clients. Some applications, such as the InTouch HMI or Microsoft Excel, can simultaneously be both a client and a server.

Important: On Windows Vista and later operating systems, Local DDE is supported only when the DAServer is configured as "Not a Service" and activated from its executable file or started from the InTouch HMI. Local DDE is not supported when the DAServer is activated from the System Management Console (SMC).

FastDDE

FastDDE packs many proprietary Wonderware Dynamic Data Exchange messages into a single Microsoft DDE message. This improves efficiency and performance by reducing the total number of DDE transactions required between a client and a server. Although the Wonderware FastDDE extends the usefulness of DDE for our industry, this extension is being pushed to its performance constraints in distributed environments.

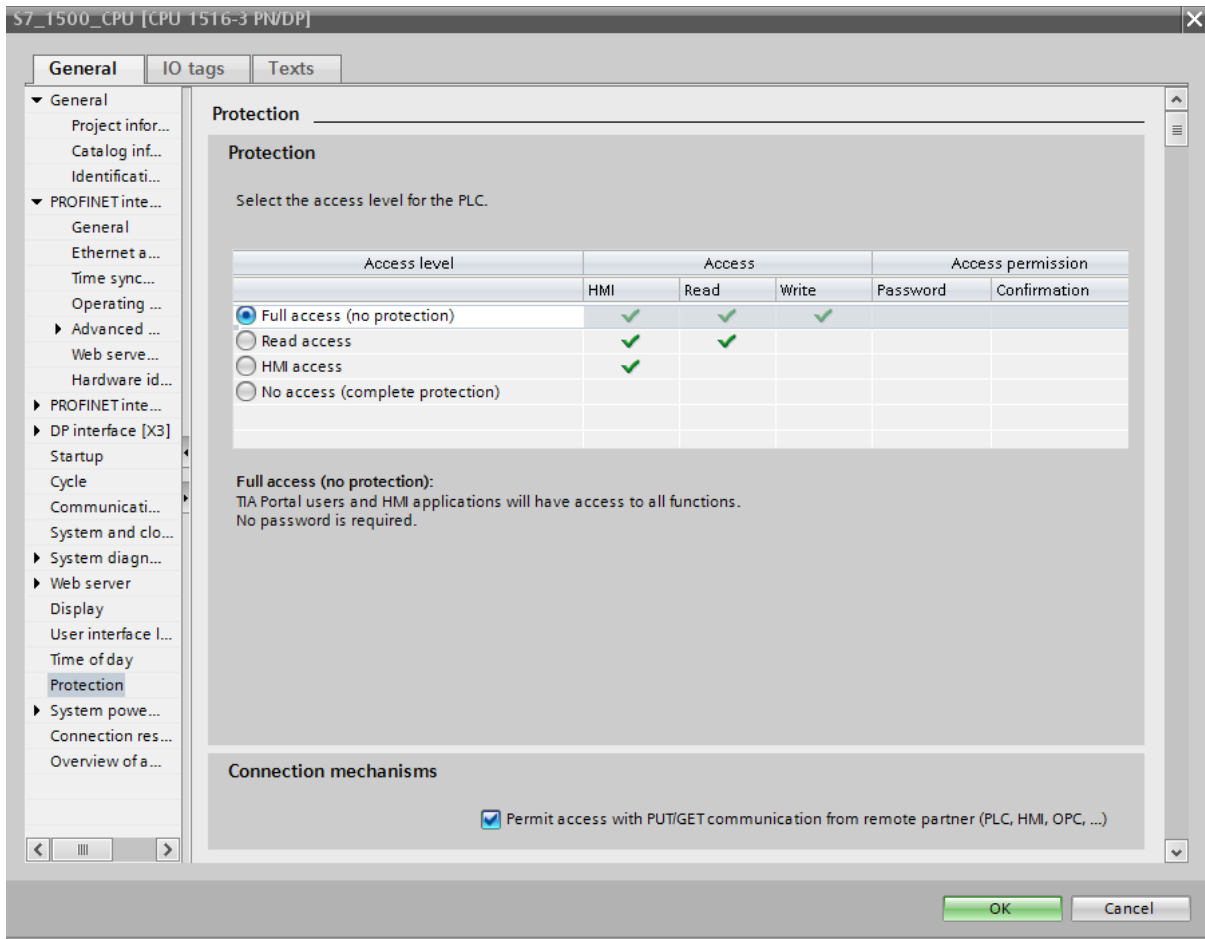
Supported Device Protocols

The SIDirect DAServer uses only the TCP bus communications protocols over the Ethernet to communicate with the Siemens S7-200/300/400/1200/1500 family of controllers. The SIDirect DAServer does **not** support MPI, Profibus, and other non-Ethernet protocols.

Special Pre-Configuration for S7-1200 and S7-1500 PLCs

To enable DASSIDirect to access data inside the S7-1500, do the following in the Siemens TIA software:

- 1 In the TIA software, locate the S7-1500 CPU in the left tree.
- 2 Right-click the S7-1500 CPU and then click **Properties**. The properties dialog box appears.

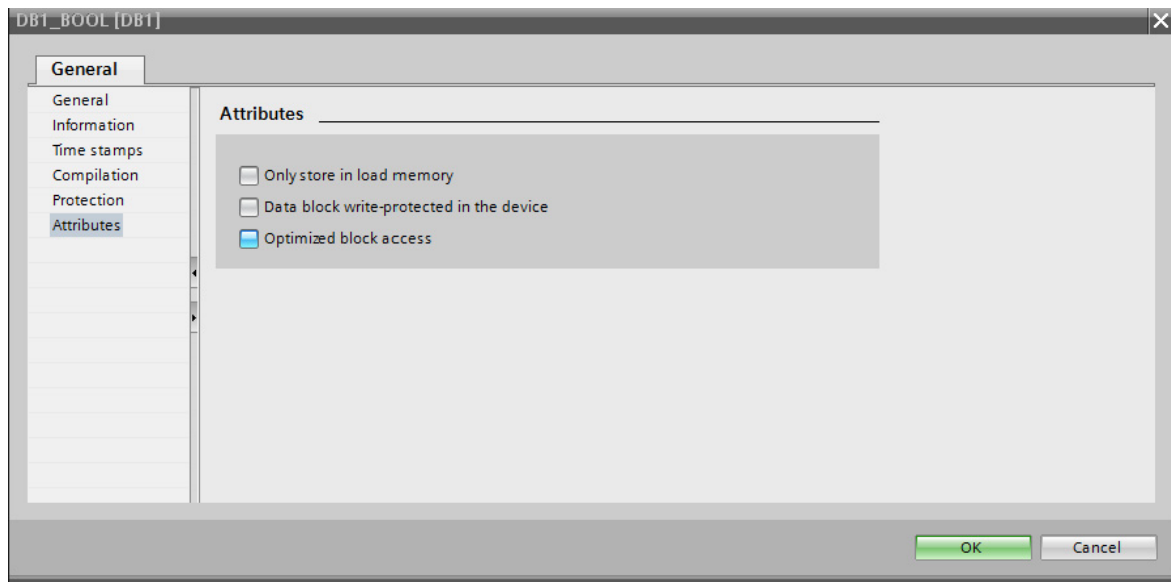


- 3 In the left pane, click **Protection**.
- 4 In the **Connection mechanisms** area, select the **Permit access with PUT/GET communication from remote partner** check box.
- 5 Click **OK**.

To enable DASSIDirect to access data inside the S7-1200 and S7-1500, do the following in the Siemens TIA software:

- 1 In the TIA software, locate the data block in the left tree.

- 2 Right-click the data block, and then click **Properties**. The properties dialog box appears.



- 3 In the left pane, click **Attributes**.
- 4 Click to clear the **Optimized block access** check box.
- 5 Click **OK**.

Chapter 2

Setting Up Your DAServer

This section describes the procedures required to set up the SIDirect DAServer for use with the supported devices and communication-interfaces.

Many high-level functions and user-interface elements of the DAServer Manager are universal to all DAServers. These universal functions are described in detail in the *DAServer Manager User's Guide*.

See the *DAServer Manager User's Guide* to find more information on:

- Global parameters
- Configuration sets
- Time zone features
- Icon definitions
- Activation/deactivation
- Configuring as a service
- Importing/exporting device items
- Standard diagnostics

The SIDirect DAServer uses the the TCP bus communications protocols over the Ethernet to communicate with the Siemens S7-200/300/400/1200/1500 family of controllers.

The SIDirect Hierarchy in the DAServer starts with the PORT_CP_S7 Object, followed by selected supported SIDirect controllers.

Note: Before attempting to configure your DAServer, you should determine the hierarchical structure of your network/controller environment.

Checklist for Setting up the SIDirect DAServer

If you are setting up a DAServer for the first time, perform the following tasks in the order listed:

- 1** Review the items described in "Before You Begin" on page 12.
- 2** Locate the DAServer in the System Management Console (SMC). See "Finding Your DAServer in the SMC" on page 19.
- 3** Configure the global parameters. See the *DAServer Manager User's Guide*.
- 4** Add a port. See "Configuring the SIDirect DAServer" on page 21.
- 5** Add and configure ports and devices. See applicable configuration set-up in "Configuration" on page 21.
- 6** Add one or more device groups. See "Device Group Definitions" on page 38.
- 7** Add device items. See "Device Item Definitions" on page 43.
- 8** Activate the DAServer. See "Configuring the SIDirect DAServer" on page 21.
- 9** Access data from the client, see "Accessing Items Using the OPC Communications Protocol" on page 57.
- 10** Troubleshoot any problems. See "Troubleshooting" on page 137.

Finding Your DAServer in the SMC

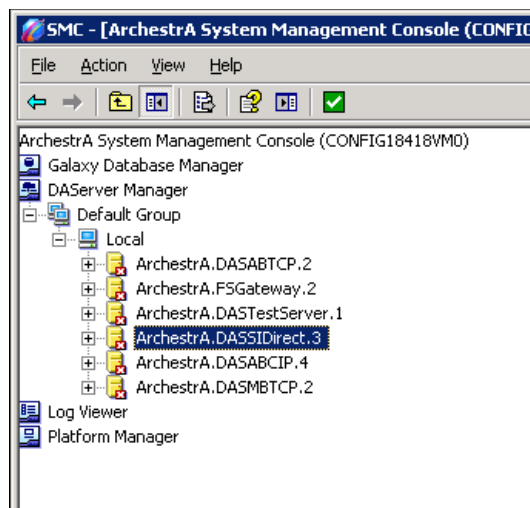
Each DAServer is identified by a unique program name (ProgID) under the SMC. The ProgID for this SIDirect DAServer is: **ArchestrA.DASSIDirect.3**.

On the computer where the DAServer is installed, it can be found in the local node of the default group of the DAServer Manager.

You do not need to install the DAServer Manager on the same computer as the DAServer. When you access the DAServer remotely, you will not find the DAServer node under the local node. You must locate and identify the DAServer on a computer in one of the node groups.

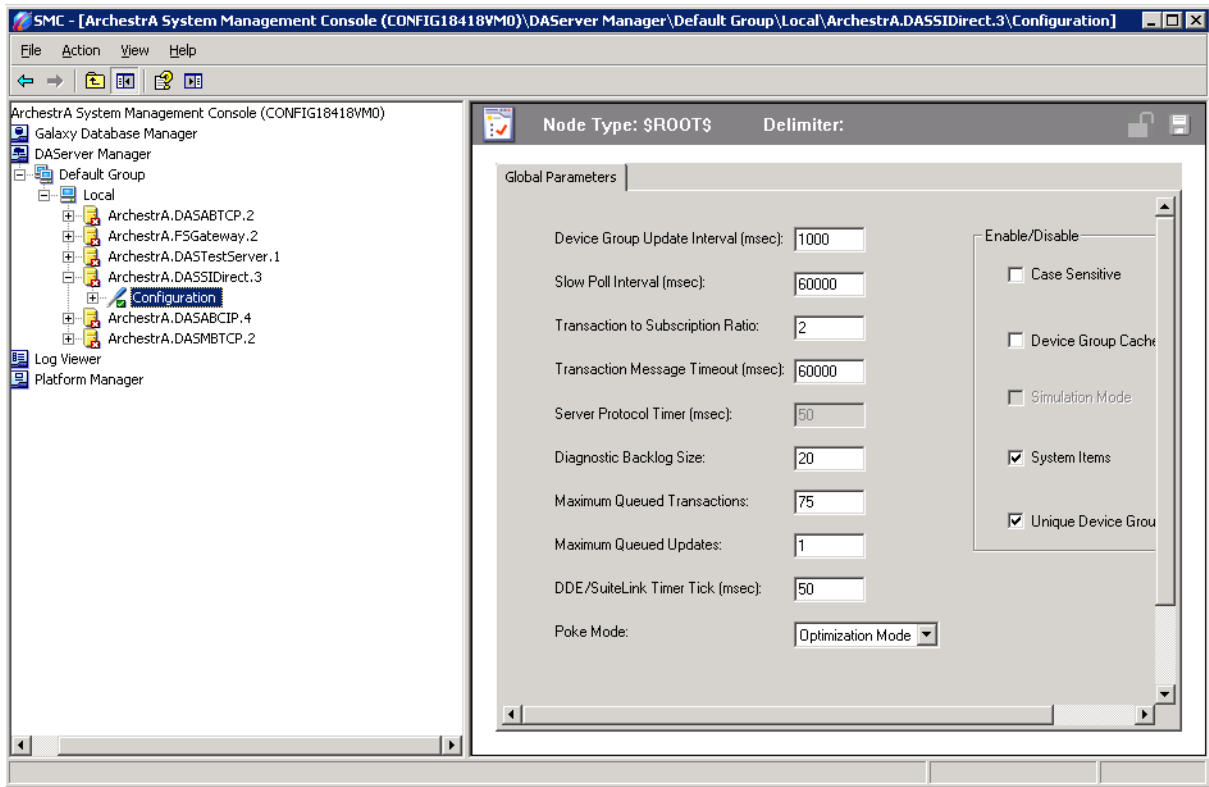
To find the DAServer

- 1 On the system **Start** menu, click **Programs**. Navigate to the Wonderware folder that contains the System Management Console and then click **System Management Console**.
- 2 In the **System Management Console**, expand **DAServer Manager**.
- 3 Locate the group with the node: **ArchestrA.DASSIDirect.3**



- 4 Expand the **ArchestrA.DASSIDirect.3** node to display the global parameters.
- 5 Select the **configuration** node to display the global parameters.

To view global parameter configuration and settings in detail, see the *DAServer Manager User's Guide*.



Chapter 3

Configuration

After you install the Wonderware SIDirect DAServer, you need to configure it. Perform this configuration using the DAServer Manager hosted in the **ArchestrA System Management Console**.

Before you activate the DAServer, you must first build the device hierarchy, simulating the physical hardware layout, to establish communications to each of the controllers. After you build the S7 hierarchy, you can configure the respective devices for communications.

Finally, you can create the desired Device Groups for each controller by clicking on the **Device Groups** tab.

Configuring the SIDirect DAServer

The SIDirect DAServer is hosted by the DAServer Manager, a Microsoft Management Console (MMC) snap-in environment. Many high-level functions and user-interface elements of the DAServer Manager are universal to all DAServers. **Only** the documentation for the DAServer Manager contains descriptions of those universal functions/UI elements.

Reading the documentation for both the MMC and the DAServer Manager is critical to understanding this user's guide. To read the documentation about the MMC and DAServer Manager, click the Help topics on the MMC **Help** menu. Both the MMC and DAServer Manager Help are opened. An Adobe Acrobat version of the DAServer Manager documentation (DAServerManager.pdf) is provided.

Note: Most items in each shortcut menu are standard Windows commands. See the **Help** menu of the MMC for more information about those commands.

Before you can configure the SIDirect DAServer, you need to perform the following steps.

- 1 Install the Wonderware SIDirect DAServer by running the **Setup.exe** program and accept all the default settings during installation. If you need more information, the DAServer installation instructions are included in a separate Help file (Install-SIDirect.chm).

Important: Because there are no default values for security settings, note the User Name and password selected during the install.

- 2 Run the DAServer Manager on at least one computer.

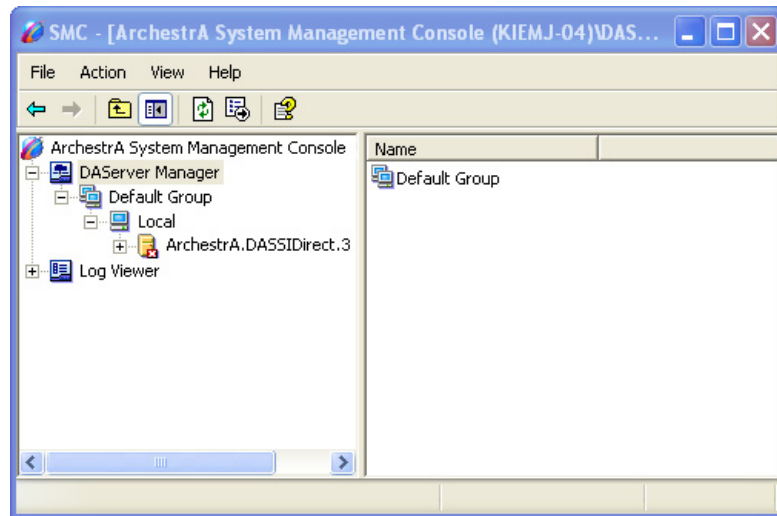
After you install the SIDirect DAServer, you need to configure it.

Note: All configuration files and Tag Database files will be moved to "C:\Documents and Settings\All Users\Application Data\Wonderware\DAServer\DASSIDirect" on Windows XP and Windows Server 2003 operating systems and to "C:\ProgramData\Wonderware\DAServer\DASSIDirect" on Windows Vista and above operating systems. The rule file will provide the version information.

To prepare the SIDirect DAServer

- 1 Start the System Management Console by clicking the **Start** button on the Windows task bar and pointing to **Programs**. Point to the **Wonderware** folder that contains the System Management Console, click **Archestra System Management Console**, and click on **DAServer Manager**.
- 2 Click on **Default Group**, then click on **Local**.
Under the Local node, the name of the DAServer is **Archestra.DASSIDirect.3**.

Note: See the DAServer Manager documentation for general information about working in this snap-in environment.



- 3 Before starting the DAServer, build the device hierarchy to establish communications to each of the controllers.

For step-by-step procedures on how to build the device hierarchy, see "SIDirect Hierarchy in the DAServer Manager" on page 24.

Important: Selecting the **Configuration** object of the hierarchy tree opens the **Global Parameters** configuration view for this SIDirect DAServer. Configure all other global parameters as required for this SIDirect DAServer.

For more information about the **Global Parameters** dialog box, including descriptions of the different Poke Modes, see the DAServer Manager documentation.

Any Global Parameters that appear dimmed are not available.

- 4 After you build the SIDirect hierarchy, you can start configuring the respective devices for communications.
- 5 Finally, create the Device Groups for each controller. Navigate to the object in the **DAServer Manager** tree view and click on the **Device Groups** tab.

For step-by-step procedures on configuring "Device Group Definitions" on page 38.

Note: If a configuration view is in an open state and you open the same server the second time, the DAServer locks the second instance of this same-server access for any update or configuration activities. Access to this second server resumes after the first one is closed.

The DAServer is ready to use after you activate it.

- If you use an OPC Client, the DAServer auto-starts.
- If you use DDE/SuiteLink, start the DAServer either as a manual or automatic service.
- To activate the DAServer, right-click on **ArchestraA.DASSIDirect.3** and select **Activate Server**.

To run the SIDirect DAServer as a service, right-click on the SIDirect **DAServer name** and select **Configure As Service**. You can configure it as an auto service or manual service.

For more information about configuring your SIDirect DAServer as a service, see the Activation/Deactivation/Service Component of the DAServer Manager documentation.

SIDirect Hierarchy in the DAServer Manager

The SIDirect DAServer uses a two-tier hierarchy for modeling the S7 objects and a custom device group configuration.

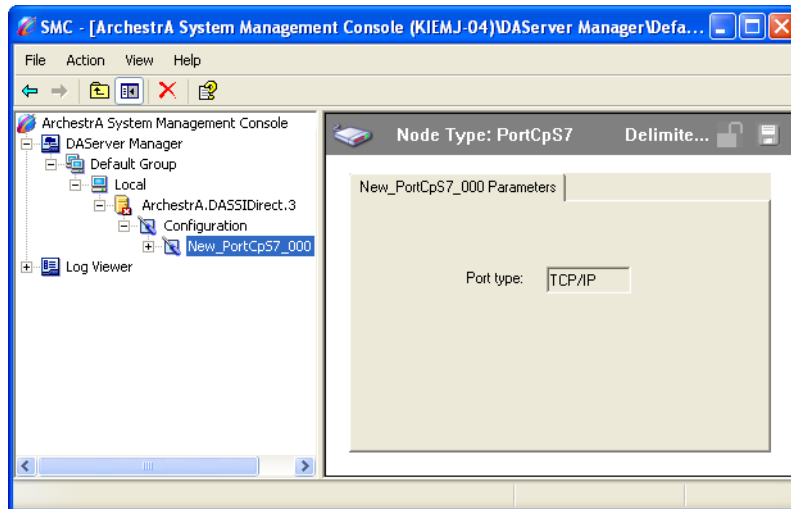
Before configuring your SIDirect DAServer, determine the hierarchical structure of your network/PLC environment.

PortCpS7 Object

The server-specific configuration portion of the SIDirect DAServer hierarchy tree under the DAServer Manager starts at the PortCpS7 (Communications Processor) object. It represents the network board in the computer that communicates with the PLC. Usually it is an ordinary network card identified by the local IP address.

To add PortCpS7 object to your SIDirect hierarchy

- 1 Select and right-click on the **Configuration** branch of the hierarchy, then select **Add PortCpS7 Object**. The **New_PortCpS7_000 Parameters** object appears.



The hierarchy entry is added in edit mode. You can appropriately describe components of your specific hardware environment. If you do not rename the object at this time, a numeric sequencing system is applied. Any hierarchy entry can be renamed at a later time

- Only 1 (one) PortCpS7 object can be created from the **Configuration** branch.
- 2 Rename this object as needed.
- The New_PortCpS7_000 Parameters configuration view shows one element already configured:

- **Port type**
The type of the port is TCP/IP.

Important: If you subsequently clear your configuration hierarchy, you must create this PortCpS7 object by right-clicking on **Configuration** and selecting **Add PortCpS7 Object**. An object called New_PortCpS7_000 Parameters is created. Rename as appropriate. From this point, all of the following instructions apply.

S7Cp Object

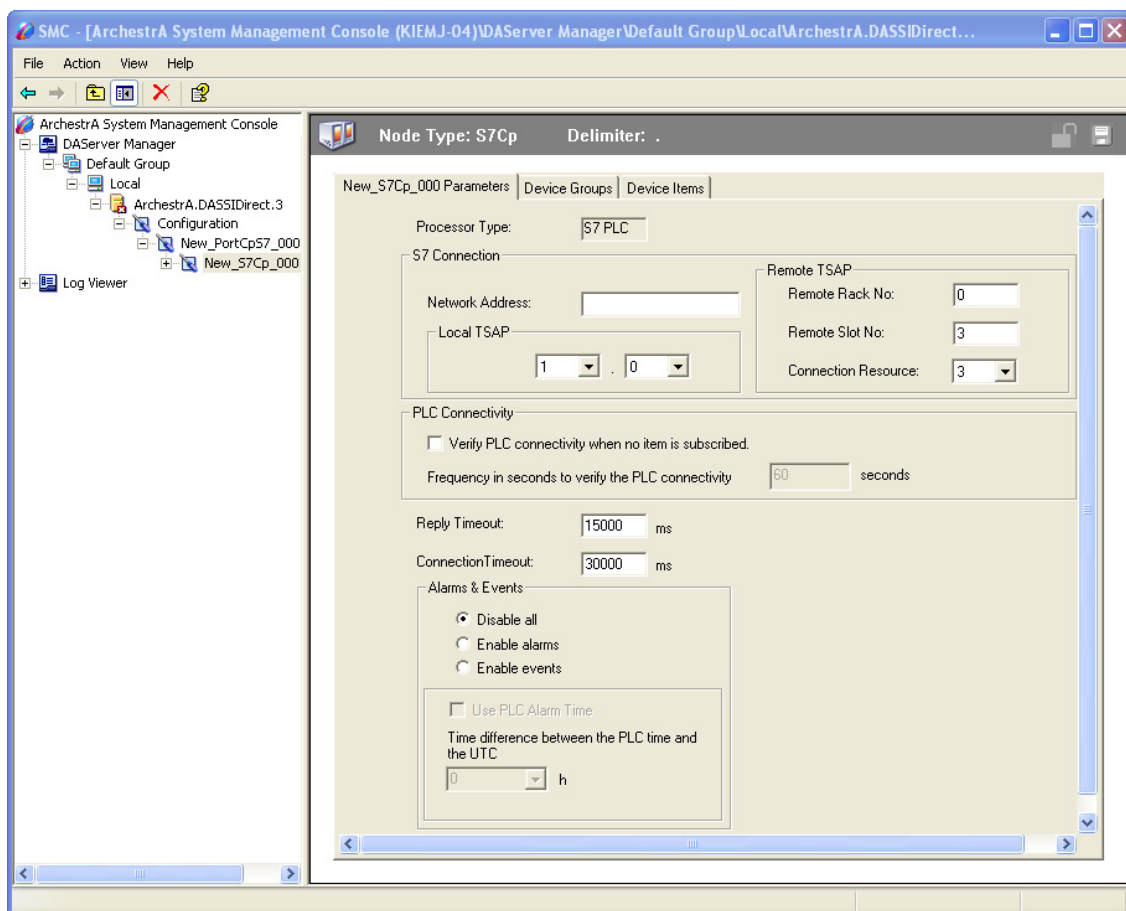
The SIDirect DAServer can connect to different CPs in an S7 PLC. Each of the S7Cp node models the end-point of the communications path. The S7Cp object can access only some of the memory type of the S7-200 PLC because the S7-200 PLC has its own specific syntax.

For the S7Cp object, you must pre-configure the connection information on the S7 200 PLC. The S7Cp_200 object can connect to the S7 200 PLC using the default settings.

From the New_PortCpS7_000 branch of the DAServer hierarchy, create the S7Cp object.

To add S7Cp object to your SIDirect hierarchy

- 1 Select and right-click on your
- 2 Select and right-click your **New_PortCpS7_000** object, then select **Add S7Cp Object**. The New_S7Cp_000 Parameters configuration view appears.



A maximum of 1024 S7Cp objects can be created from the **New_PortCpS7** branch.

- 3 Rename as needed.

The S7Cp configuration view has six elements, five of which you can configure:

- **Processor Type**

The processor type is S7 PLC.

- **S7 Connection**

The S7 Connection has the following three configurable parameters.

- **Network Address**

The IP address or host name of the remote S7 CP.

Type the network address where the PLC is located (for example, "10.11.12.13"), or a host name if one is defined in the local hostlist. The address field cannot be blank and the IP address or host name cannot be more than 255 characters.

- **Local TSAP**

The local TSAP of the computer. Select the Hex numbers for the connection resources from the menu. The Local TSAP consists of two (2) Hex numbers. The first number corresponds to the connection resource. Each number ranges from 00 to FF. The default values are 01 and 00, respectively.

- **Remote TSAP**

This Remote TSAP corresponds to what you configured in the TSAP for the S7 CP. Configure the Remote TSAP by typing in the decimal numbers for the **Remote Rack** and **Remote Slot**, and by selecting the Hex number for the **Connection Resource** from the menu. The values for the **Remote Rack No.** and **Remote Slot No.** range from 0 to 255, with the default values of 0 and 3, respectively. The value for the **Connection Resource** ranges from 00 to FF. The default value is 03.

- **PLC Connectivity**

The watchdog scheme for detecting the connectivity status to the PLC when there are no activities (no items are subscribed to).

- Select the **Verify connectivity when no item is subscribed** check box to turn on the watchdog.
 - Specify the watchdog time interval, in seconds, in the **Frequency in seconds to verify the PLC connectivity** box.

- **Reply Timeout**

Enter a value, in milliseconds, beyond which messages time out.

- Allowable range is 0 to 100,000 milliseconds.
- The default value is 15,000 milliseconds.

If you decrease this value, the SIDirect DAServer reacts faster to a communications failure.

- **Connection Timeout**

Enter a value, in milliseconds, beyond which a pending request to initiate a connection times out.

- Allowable range is 0 to 100,000 milliseconds.
- The default value is 30,000 milliseconds.

- **Alarms and Events**

Enable Alarms or Events or disable both for this connection by selecting:

- Disable all
- Enable alarms
- Enable events

On one connection, either Alarms, Events, or none can be configured. If you need to access both Alarms and Events, create two different connections.

Use PLC Alarm Time

The alarm time in the PLC used to timestamp alarm and event-related data.

You can enable this feature so that data in the alarm and event blocks will be time stamped using the timestamps provided in the alarm and event blocks. The DAServer, in this case, will not generate its timestamp for the received data. This feature is only available for data in the alarm and event blocks.

Time Difference Between the PLC Time and the UTC

The time difference, in hours, between the PLC time and the UTC time.

- The value range is from +12 hours to -12 hours.
- The default value is 0.

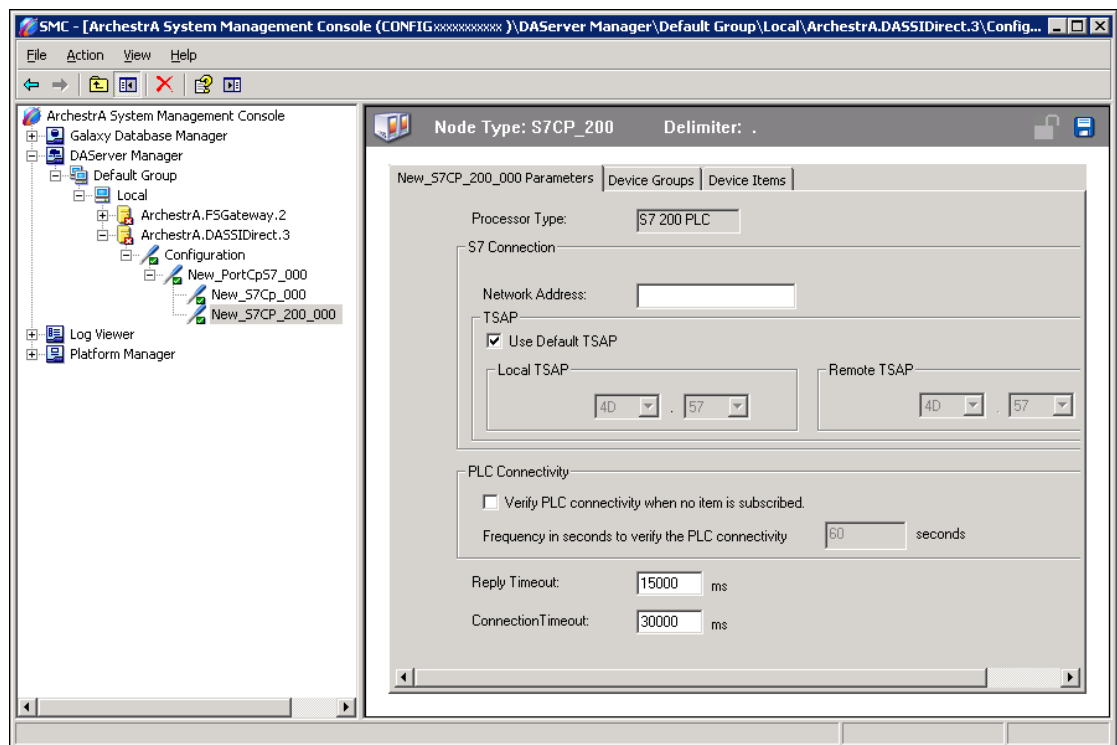
S7Cp_200 Object

The SIDirect DAServer can connect to different CPs in an S7-200 PLC. Each of the S7Cp_200 nodes models the end-point of the communications path.

From the New_PortCpS7_000 branch of the DAServer hierarchy, create the S7Cp_200 object.

To add S7Cp-200 object to your SIDirect hierarchy

- 1 Select and right-click your **New_PortCpS7_000** object, then select **Add S7Cp_200 Object**. The New_S7Cp_200_000 Parameters configuration view appears.



- A maximum of 1024 S7Cp_200 objects can be created from the **New_PortCpS7** branch.

- 2 Rename as appropriate.

The S7Cp_200 object configuration view has five elements, four of which are configurable:

- **Processor Type**
The processor type is S7-200 PLC.
- **S7 Connection**
The S7 Connection has the following three configurable parameters.

- **Network Address**

The IP address or host name of the remote S7-200 CP.

Type the network address where the PLC is located (for example, "10.11.12.13"), or a host name if one is defined in the local hostlist. The address field cannot be blank and the IP address or host name cannot be more than 255 characters.

- **Local TSAP**

The local TSAP of the computer. The Local TSAP consists of two (2) Hex numbers. The first number corresponds to the connection resource. Each number ranges from 00 to FF. The default values are 4D and 57, respectively.

- **Remote TSAP**

This Remote TSAP corresponds to what you configured in the TSAP for the S7-200 CP. The Remote TSAP consists of two (2) Hex numbers. Each number ranges from 00 to FF. The default values are 4D and 57, respectively.

- **PLC Connectivity**

The watchdog scheme for detecting the connectivity status to the PLC when there are no activities (no items subscribed to).

- Select the **Verify PLC connectivity when no item is subscribed** check box to turn on the watchdog.
- Specify the watchdog time interval, in seconds, in the **Frequency in seconds to verify the PLC connectivity** box. The default is 15 seconds, with a maximum of 30 seconds.

- **Reply Timeout**

Enter a value, in milliseconds, beyond which messages time out.

- Allowable range is 0 to 100,000 milliseconds.
- The default value is 15,000 milliseconds.

If you decrease this value, the SIDirect DAServer reacts faster to a communications failure.

- **Connection Timeout**

Enter a value, in milliseconds, beyond which a pending request to initiate a connection times out.

- Allowable range is 0 to 100,000 milliseconds.
- The default value is 30,000 milliseconds.

The logical endpoint for each branch of the SIDirect hierarchy tree is a hardware device (PLC).

Note: When adding a hierarchy object, the default name is in the format of **New_ObjectName_###**. ObjectName is the name of the object type and ### is a numeric value starting from "000" sequentially per hierarchy object. The link name for the OPC items is constructed by assembling the respective object names of the nodes along the hierarchy tree in the logical order, starting from the PortCpS7 root of this DAServer down to the leaf. This creates a link name that is always unique for the DAServer.

To use the DAServer, you have to activate it. For more information, see the DAServer Manager documentation.

Device Redundancy

The DAServer Manager provides the ability to assign a redundant device for fail-over protection in the event of a device failure. Two identical devices are expected to be configured in the DAServer Manager having identical item syntax, connected to the same DAServer.

Note: Items can not be added for updates through the Redundant Device Object if the items do not exist in both controllers.

Run-time Behavior

The DAServer will start with the active device. The DAS Engine will switch to the standby device when the active device fails to communicate. The value of the \$SYS\$Status will determine the communication failure.

Note: The value of the \$SYS\$Status of the standby device must be TRUE in order to switch over to the standby device. Otherwise, there will not be any failover.

When \$SYS\$Status shows a FALSE value at both active and standby devices, the DAS Engine will consider a complete communication failure and mark all the items subscribed to the redundancy device hierarchy with the current time and the appropriate OPC quality. The DAS Engine will activate the slow-poll mechanism to retry the communication to both devices until either one of the Ping Items returns to a good quality and update its \$SYS\$Status item to TRUE.

When the DAS Engine switches to the standby device, the standby device becomes active and the originally active device becomes the standby.

When the active device becomes the standby device the Ping Item will not be deleted from that the standby device. This will ensure the standby will be able to recover the communication again.

Note: The Ping Item must be a valid item from the controller that has not been rejected by the server for the failover to function properly.

The DASServer will log any failover activities. All other functionality such as diagnostics, enable/disable, and reset will be performed exactly same as it is performed for any other hierarchy node.

Note: Unsolicited message configuration is not supported in the Redundant Device Object itself. You can still receive unsolicited messages directly from device groups defined in the regular server hierarchy.

This feature allows the DASServer to provide fail over support by providing one node which switches between two other nodes. The Redundant device is configured with a redundancy node which directs itself to one of the two nodes and switches to the other based on lack of communications to a common user-configured controller item. In this manner the Redundant Device Object can be used to direct client requests to the redundant node, which switches between device or communication pathway failure without intervention.

You can use the S7-400H PLC in a stand-alone or redundant configuration.

- In a stand-alone S7-400H configuration (no redundant controllers configured), DASSIDirect supports subscription, poking, alarm and event communications, block services, and cyclic services.
- When two S7-400H PLCs are configured in redundant mode through the RDO, subscription and poking are supported. Alarm and event communications, cyclic services, and block services are not supported.

The \$SYS\$MasterController item indicates whether a particular S7-400H PLC is a master or slave. This information also is used internally to trigger a failover in the DAS redundant pair. You do not need to subscribe to the \$SYS\$MasterController item for failovers to take place.

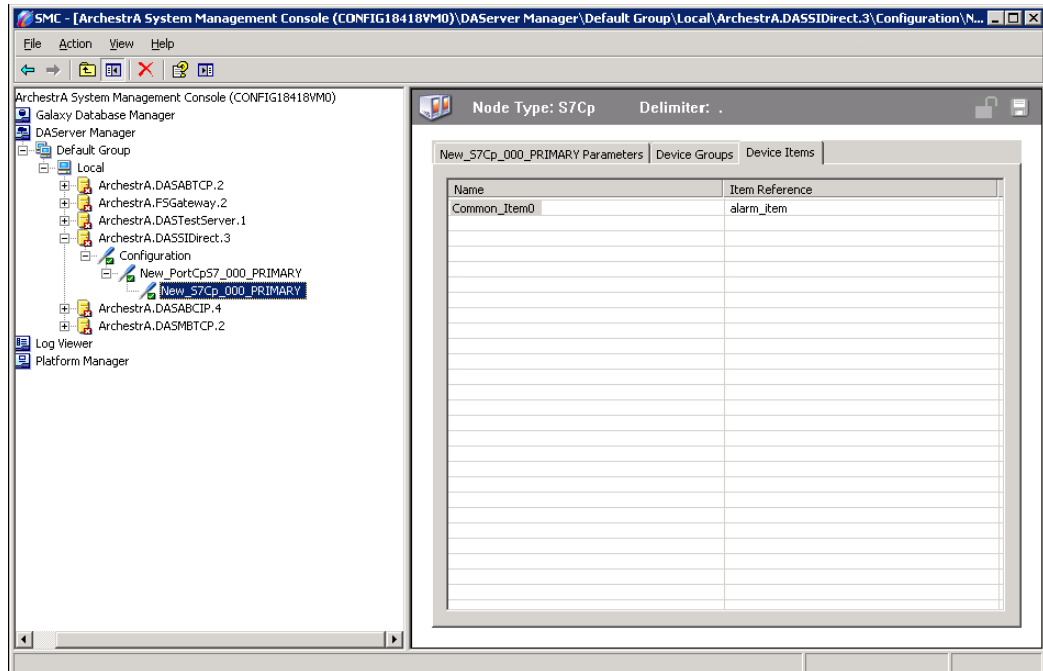
Configuring Device Redundancy

The DASServer Manager provides the ability to assign redundant device for fail-over protection in the event of device failure. Two devices must be configured in the same DASServer having identical item syntax.

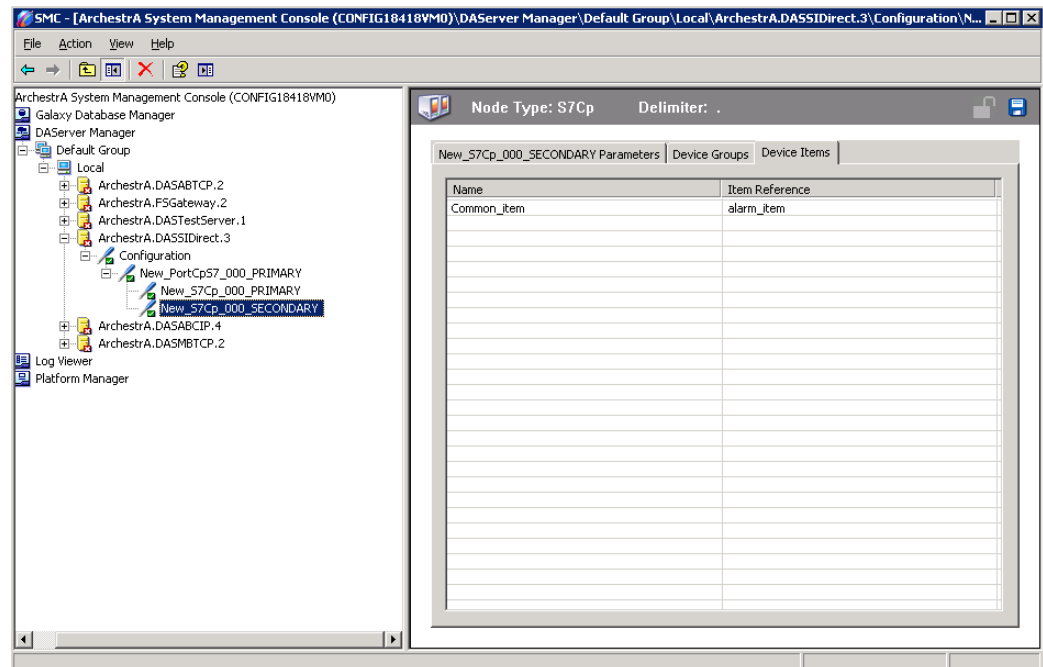
Primary and secondary devices will be setup in the REDUNDANT_DEVICE object in the SMC, along with a common item name (ping item) shared by each device to determine device status.

To setup up a REDUNDANT_DEVICE from the configuration branch

- 1 Set-up a primary device and hierarchy in the DAServer Manager in the SMC.

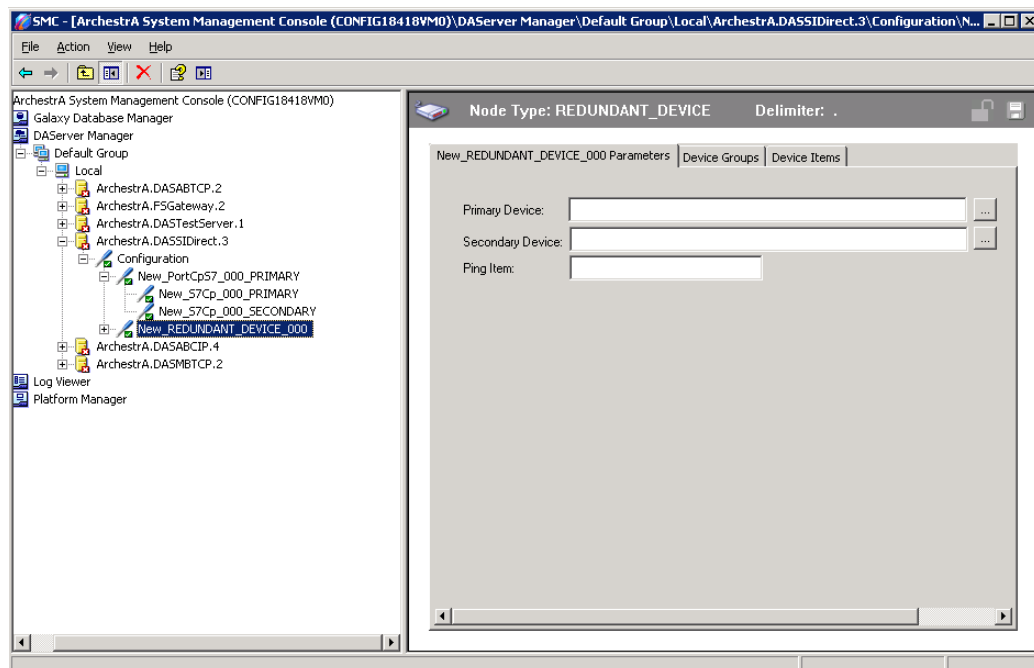


- 2 Set up a secondary device on the same DAServer.

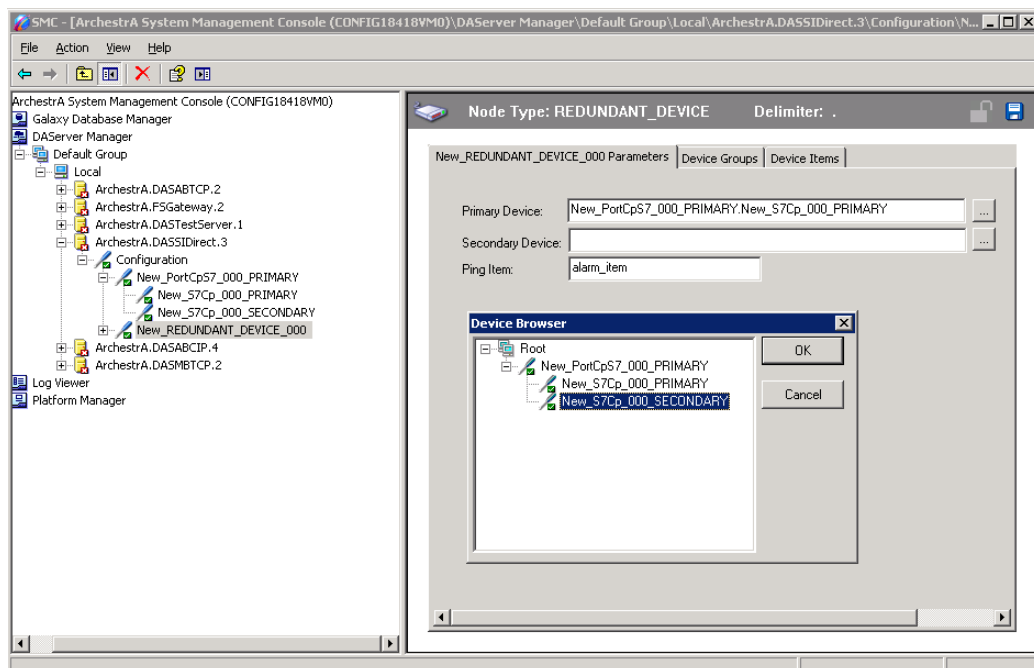


- 3 Select and right-click on the configuration node under the TCP/IP Network Device object.

- 4 Select **Add REDUNDANT_DEVICE Object**. An object called **New_REDUNDANT_DEVICE_000** is created.



- 5 Rename the newly created object as appropriate. The **New_REDUNDANT_DEVICE_000** configuration view is displayed in the Configuration branch of the hierarchy.
- 6 Enter or use the device browser to select the primary and secondary devices.



- 7 Enter one device item that can be shared between the primary and secondary devices to determine device status.

8 Save the hierarchy node configuration by clicking on the save icon.

Note: Unsolicited message configuration is not supported from the device redundant hierarchy.

Important: A Ping item must be specified and be a valid tag in both the primary and secondary controllers to determine the connection status for \$SYS\$Status. The Ping item can be a static item in the device such as a firmware version or processor type. If the Ping item is invalid or does not exist in the controller, the failover operation may not work correctly as the value of \$SYS\$Status may continue to stay as FALSE in the standby device.

Using the S7 Tag Creator

The SIDirect DAServer supports the S7 Tag Creator to create the tags. You can also use the S7 Tag Creator editor for categorizing tags into different views, importing tags into different sections and areas, and generating tags automatically.

The SIDirect S7 Tag Creator is similar to the current S7 I/O Server S7 Tag Creator, except that the infrastructure for the S7 Tag Creator allows you to create S7 tags for the SIDirect DIObjects, while the I/O Server S7 Tag Creator works only with the S7 I/O Server. Additionally, the SIDirect S7 Tag Creator has numerous other features that you do not find in the I/O Server S7 Tag Creator.

Note: The S7 Tag Creator does not read the current memory/register layout from the S7 PLC online.

Using the SIDirect DIObject S7 Tag Creator, you can:

- Generate alias names and tag names from the Siemens Step 7 .awl, .asc, .dif, .sdf, and .seq output files.
- Categorize tags into three different views for your selection: S7 tag hierarchy, alias name, and tag data type.
- Continuously apply user-defined filter criteria for alias names.
- Automatically generate tags.
- Import tags into the Scan Group, Block Read, and Block Write DA Groups on the S7C_S7 DIObject editor.

The S7 Tag Creator module generates tags for ArchestrA and SIDirect DIObjects. You can either directly specify the tags or import tag files generated by the Siemens S7 Step 7 programming software.

Chapter 4

Device Groups and Device Items

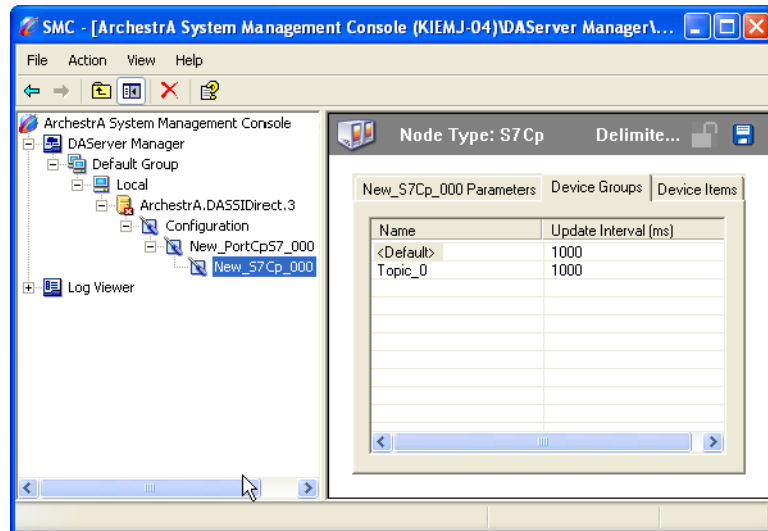
Select the **Device Groups** tab or the **Device Items** tab in the DAServer Manager user interface to create new, modify, or delete device-group and device-item definitions for an object.

- For DDE/SuiteLink communications, one or more device-group definitions must exist for each PLC that the SIDirect DAServer communicates with.
- Each device-group (topic) definition must contain a unique name for the PLC associated with it.

Each configuration view associated with objects in the SIDirect DAServer hierarchy tree has a **Save** button. When you modify the **Parameters**, **Device Groups** dialog box, or the **Device Items** dialog box, click **Save** to implement the new modifications. If you try to open another configuration dialog box you are prompted to save the new data to the configuration set.

Device Group Definitions

The **Device Groups** dialog box, which appears by clicking the **Device Groups** tab in the **New_S7Cp_000 Parameters** or the **New_S7Cp_200_000 Parameters** configuration view, is the place where you create, add, delete, and define device groups. You can also configure default update intervals for the objects and edit update intervals in this dialog box.



Note: When you select another part of the DAServer tree hierarchy, you are prompted to save the modifications to the configuration set.

The device group for the S7CP_200 object is not editable.

To create or add device groups

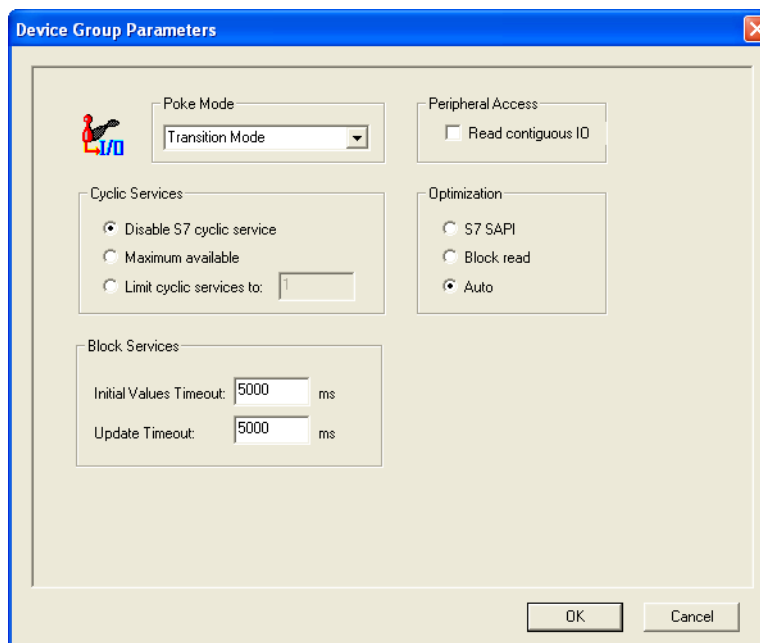
- 1 Right-click in the **Device Groups** box and click **Add**.
- 2 Enter a unique name for the device group. Device group names are case-insensitive.

To delete device groups

- ◆ Right-click on the device group to be deleted from the list and select **Delete**.

To make edits on device groups

- 1 Make edits on a device group by right-clicking on the device group to be edited. Select **Edit** from the menu to open the **Device Group Parameters** dialog box.



- 2 Make the necessary edits.

The **Edit** dialog box contains five configurable elements:

- **Poke Mode**
Select one of three settings to tune the poking behavior to the PLC.
- **Control mode**
This mode preserves the poke order and does not fold write values. Select this mode when using this device group with control clients such as the InBatch and InControl applications. If selected, the server processes all poked values in the order they are received from a client and does not discard any poke values even when several values are poked to the same item.
- **Transition mode**
This mode preserves the poke order but allows folding of poke values in the following way: if the server receives more than one value per item it can discard poke values except for the first, second, and last value for this item.
Transition mode prevents InTouch sliders from stuttering.
- **Full optimization**
This mode allows changing the poke order and folding of poke values by poking the last value of a series of pokes to one item only. This minimizes bus traffic and poke duration. Use this mode for high volume pokes such as recipe downloads where the sequence of poke does not matter.

Note: The highest performance can be achieved with the Full optimization mode. Consider your data consistency requirements with respect to poke order and folding.

The default Poke Mode settings for the SIDirect DAServer is Transition mode.

- **Peripheral Access**

Select the **Read contiguous IO** check box to force the SIDirect DAServer to read input and output blocks (also peripherals) if their address spaces are not contiguous.

Select this check box if you have some holes in the input-address or output-address space in your PLC.

- **Cyclic Services**

If you use Cyclic Services (the **Disable S7 cyclic service** option is **not** selected), configure two additional settings:

- Maximum available
- Limit cyclic services to

If you know how many services the remote PLC can handle, you can limit the use of cyclic services in this device group and distribute the available cyclic services among the device groups associated with this connection. Otherwise, you can use the maximum available services.

- Allowable range for **Limit cyclic services to** is 1 to 150.
- The default is 1.

Click on **Disable S7 cyclic service** to disable the S7 cyclic services for the device group.

- Cyclic services have a reliable update frequency and need less bus access. They are a limited resource in the PLC and/or Communications Processor.

If you select the **Disable S7 cyclic service** option, the SIDirect DAServer polls all items in this device group. If you do not select this option, the SIDirect DAServer tries to register as many items in the cyclic services as possible.

- If there are insufficient credits for cyclic services left, the SIDirect DAServer creates poll messages instead of cyclic messages for the remaining items.

Reasons for disabling Cyclic Services include the following:

- There is a device group with a long update interval. It should not occupy the cyclic services resource.
- You need to force the SIDirect DAServer to attempt to collect data faster than 100 milliseconds.
- You activate and deactivate items very often.

Note: The S7-1200 and S7-1500 PLCs do not support cyclic services.

- **Optimization**

Select one of the following settings to configure the optimization mode the SIDirect DASServer needs to use to acquire data from the PLC:

- S7 SAPI
- Block read
- Auto

The default is Auto. For more detailed information, see "Optimization Considerations" on page 42.

Note: The S7-1200 and S7-1500 PLCs do not support SAPI.

- **Block Services**

If the Block Services function is required, you have to configure two settings in the **Block Services** box:

- **Initial Values Timeout**

Allowable range for Initial Values Timeout is 0 to 65,535 milliseconds; the default value is 5,000.

- **Update Timeout**

Allowable range for Update Timeout is 0 to 65,535 milliseconds. The default value is 5,000.

The Block Services function needs time-outs to supervise reading initial values and updating the block items to this connection. A time-out value of 0 (zero) disables the time supervision of block messages.

Block services are unconfirmed services. If the remote station does not send data within this time range, the Block Services is reinitialized and an error message appears in the Logger.

To configure default update intervals

- ◆ To configure a default update interval for the object, right-click in the **Device Groups** box and click **Config Default Update Interval** on the menu.

To edit update intervals

- ◆ To edit the update interval for an object, double-click its value in the **Update Interval** column and make the edits.
 - Update Interval is the frequency in milliseconds that the SIDirect DASServer acquires data from the topics associated with that device group.

- Different topics can be polled at different rates in a PLC by defining multiple device-group names for the same PLC and setting a different Update Interval for each device group.

Note: When you select another part of the SIDirect DAServer tree hierarchy, you are prompted to save the modifications to the configuration set.

Optimization Considerations

The SIDirect DAServer uses the following same optimization considerations as the S7 I/O Server:

- Use different Poke modes.
- Use different reading optimization modes.
- Use cyclic services to minimize traffic.
- Use block services to minimize traffic.

The DAServer can also optimize its performance in getting data from the PLC by using the optimization mode. The following options are available:

- **S7 SAPI**

The S7 SAPI mode is the same optimization mode used in the pre-release of the former Wonderware Siemens SIMATIC NET S7 I/O Server. This mode is implemented to keep the server compatible to the former server. This mode is the less-preferred optimization mode.

- **Block Read**

The Block Read mode always registers a whole byte array containing some items. If you frequently switch items (activating and deactivating) that have similar addresses, this mode is the best selection. In this mode, there are less activations and deactivations on the protocol.

- **Auto**

By default, the Auto mode is best to use when exploiting the whole PDU (Protocol Data Unit). The Auto mode has the best performance in cases where you do not make many activations and deactivations. The S7-300 family of controllers can process PDU lengths of 480 to 960 bytes, depending on the controller. An S7-400 controller can process PDU lengths of 960 bytes.

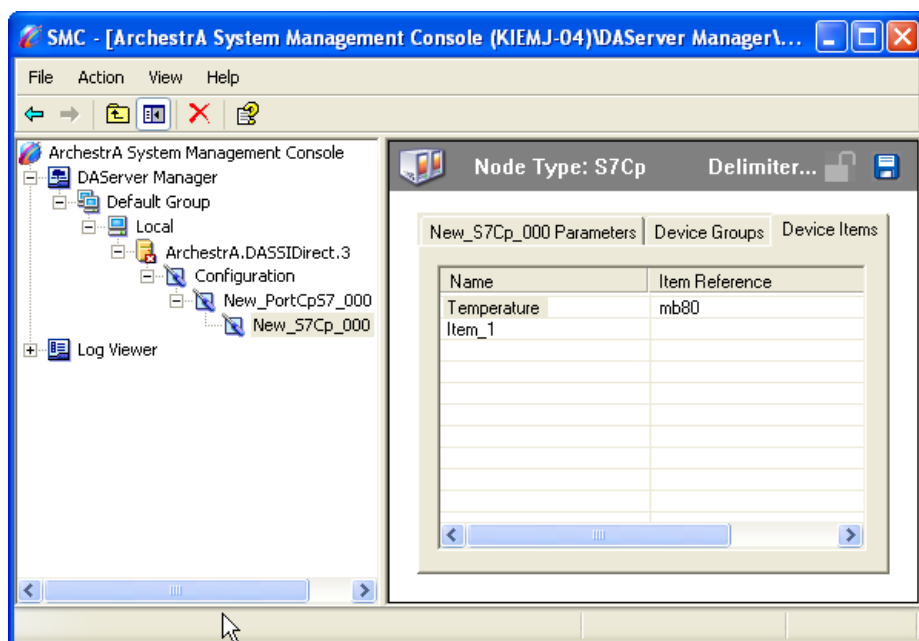
Note: The S7-1200 and 1500 PLCs do not support cyclic services or SAPI.

Device Item Definitions

The predefined item syntax/name for the S7 PLC cannot be changed. However, to make it easier to remember item names, you can create aliases for these item names.

For example, it may be easier for you to remember the item syntax "mb80" as "Temperature."

Select the **Device Items** tab in the DAServer Manager user interface to create new, modify, delete, export, or import device-item definitions for an object. The configuration is performed in the **Device Items** dialog box, which appears when you click the **Device Items** tab in the **New_S7Cp_000 Parameters** configuration view.



After you configure item names, the DAServer can perform OPC Item browsing. When the DAServer is running and an OPC client requests item information, the configured items appear under the PLC hierarchy node. User-defined data types appears at the lowest level in the hierarchy when browsed via the OPC client.

To create or add device items

- 1 Right-click in the **Device Items** box and click **Add**.
- 2 Type the item name in the **Name** column.
For example, "Temperature."
- 3 When you add a new device item, enter a unique name.
- 4 Double-click the line in the **Item Reference** column and enter the correlated item reference for the name you selected.
For example, "mb80."

Note: If the name and the item reference are the same, it is only necessary to enter a name. The DAServer assumes that the item reference is the same. This is necessary if you want to add some items for browsing via the OPC, even if they do not have a symbolic name.

To rename device items

- ◆ Right-click on the device item to be renamed and click **Rename**. Make the changes.

To delete device items

- ◆ Right-click on the device item to be deleted from the list and click **Delete**.

To clear all device items

- ◆ Right-click in the **Device Items** box and click **Clear All**. All the device items listed are cleared after you confirm their deletion.

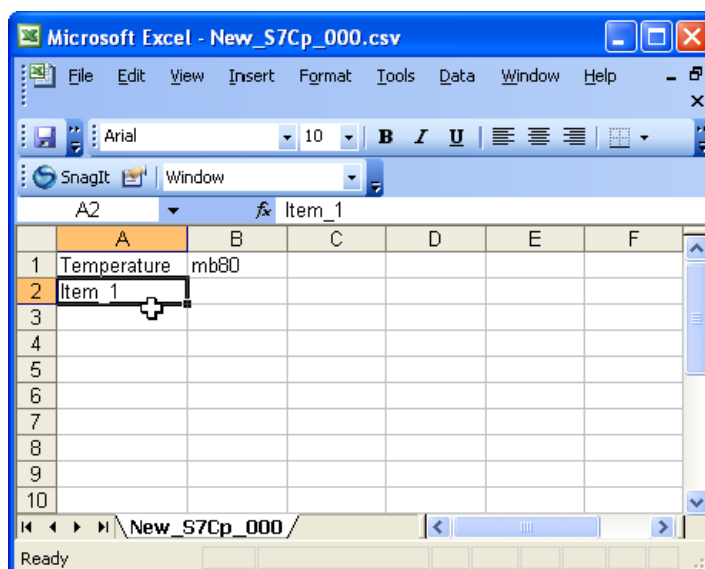
Exporting and Importing DAServer Item Data

You can export and import the DAServer item data to and from a CSV file after you configure the Device Items. This lets you perform an off-line, large-scale edit on the item data configured for a PLC and import that data back into the PLC configuration.

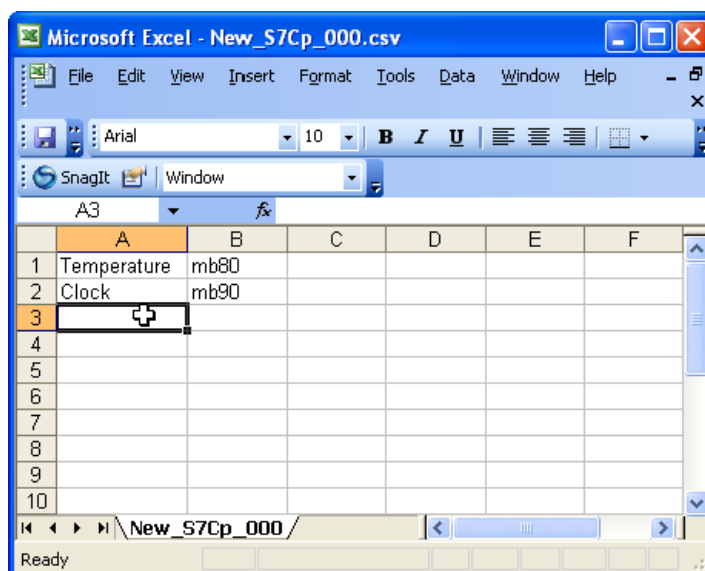
To export DAServer item data to a CSV file

- 1 Right-click in the **Device Items** box and click **Export**. The **Save As** dialog box appears and the file name defaults to "PLC Hierarchynome.csv," within the current-system-configured default directory.

- 2 Accept the defaults to save the file. The file is saved as New_S7Cp_000.csv. Now you can edit it in Microsoft Excel.



The file contains one row for each item configured with two columns: Name and Item Reference.



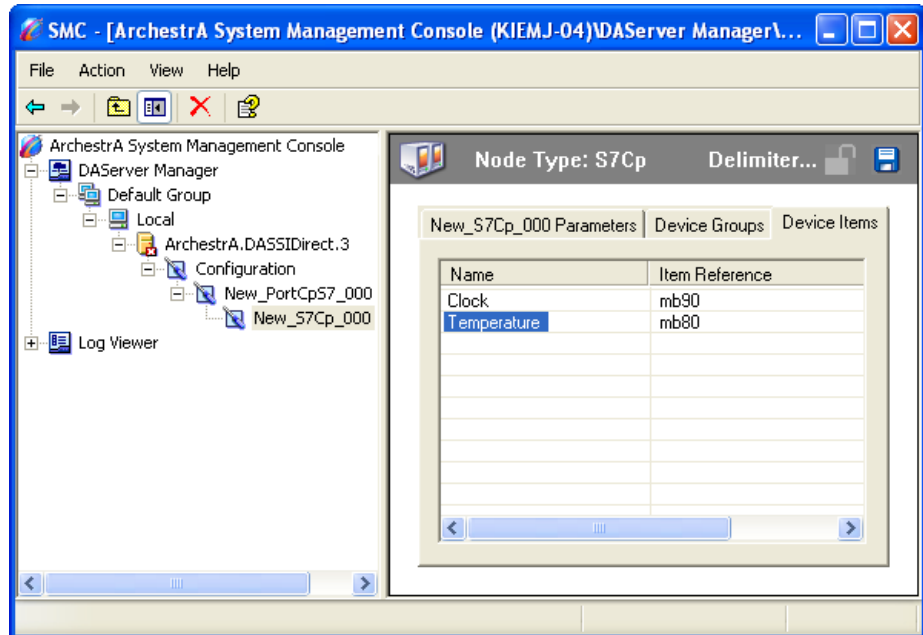
After you are done editing, you are ready to import the edited file into the DAServer Manager.

To import DAServer item data from a CSV file

- 1 Right-click in the **Device Items** box.
- 2 Click **Clear All** to clear all the item data you want to replace with the edited .csv file. The data is cleared after you click **Yes** to confirm the deletion.

- 3 Click **Import** on the menu. The **Open** dialog box appears.
- 4 Browse for the .csv file you want to import, select it, then click **OK**. The DAServer Manager imports the file. You see the data in the **Device Items** box.

When the file is imported, new item references are added based on unique names. If there are duplicate names, you can replace the existing entry with a new entry or ignore the new entry.



When the DAServer is running and an OPC Client requests item information, the imported configured items appear under the PLC hierarchy node.

Note: The SIDirect DAServer does not support importing preconfigured items (alias names of items that work with OPC) in an output file generated by the Siemens Step7 software.

Scan-Based Message Handling

Wonderware DAServers poll hardware devices for information. This polling is requested by one or more clients.

After a client requests a particular piece of information, the SIDirect DAServer creates its own request and sends that request to the hardware device. The SIDirect DAServer then waits for a response to its request. After the SIDirect DAServer receives the information, it passes that information back to the client and repeats the process until all clients stop requesting information.

You define the rate at which the SIDirect DAServer polls a particular device for a specific piece of information in the device group (topic definition) inside the SIDirect DAServer. You use a parameter called the Update Interval. When setting this parameter, there is always a trade-off between the update speed of the device group and the resulting decrease in system responsiveness.

Because you more than likely want very fast response, the temptation is to set the Update Interval to a value close to 0 (zero) seconds. However, if every item is polled at this rate, the entire system suffers due to slow response time. Therefore, compromise and set the Update Interval to a more reasonable value.

You can also create multiple device groups for each device, setting the Update Interval to different values, then assigning different items to different device groups, depending on how quickly the values change and how quickly you want to see an update of those changes.

Unsolicited Message Handling

The SIDirect DAServer processes the following three types of unsolicited messages sent by the S7 PLCs:

- Alarms
- Events
- Block Services

Alarms and Events

In the world of PLCs and DAServers, it is obvious that a PLC knows when a critical event occurs before the DAServer polls for that data. Therefore, it seems natural that if a critical event occurs, the PLC can inform the DAServer immediately, without waiting for the DAServer to poll it.

This is the role of an unsolicited message. After a PLC determines that a critical condition exists, it can generate a message immediately sent to the DAServer without a previous request from the DAServer. The unsolicited message implementation requires both the messaging instructions properly programmed in the PLC logic and the device group appropriately configured in the DAServer.

Block Services

In addition to unsolicited messages based on critical conditions or events, S7 PLCs can also handle another type of unsolicited messages called Block Services.

You can use Block Services to send blocks of data up to 64KBytes within one send job and trigger it by using a timer, an event, an I/O activity, or initiate it via a program code.

Chapter 5

Managing Your DAServer

After you configure the DAServer, there are two steps to take before you can access data with your client application.

The first step is to determine what kind of client applications are to be used with this DAServer. If any of your client applications use DDE/SuiteLink, you must configure the DAServer as a service. If only OPC client applications will be used, you can configure the DAServer as a service or as not a service.

The last step is to activate the DAServer. Some client applications can programatically activate the DAServer. If you configure the DAServer as an automatic service, the DAServer is started and activated when the computer on which the DAServer is installed starts up. If you configure the DAServer as a manual service, the DAServer is not started when the computer starts up. Instead, it is started upon the first connection from an OPC client or when activated from the DAServer Manager.

After a DAServer is running as an auto or manual service, it stays running until explicitly stopped in the DAServer Manager or the computer shuts down.

Configuring the DAServer as Service

To support DDE/SuiteLink clients, the DAServer must be configured as a service.

To configure the DAServer as a service

- 1 In the DAServer Manager, navigate to the DAServer.
 - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
- 2 Right-click **ArchestrA.DASSIDirect.3** and then click **Configure As Service**.
- 3 Click either **Auto Service** or **Manual Service**.
- 4 Read the warning message and click **Yes**.

Configuring the DAServer as Not a Service

The DAServer can only be set to run as not a service when the DAServer is in the deactivated state.

To configure the DAServer as not a service

- 1 In the DAServer Manager, navigate to the DAServer.
 - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
- 2 Right-click **ArchestrA.DASSIDirect.3** and then click **Configure As Service**.
- 3 Click **Not a Service**.
- 4 Read the warning message and click **Yes**.

Archiving Configuration Sets

A configuration set includes the DAServer's global parameters; each channel and its parameters; and each device and its parameters, device groups, and device items. It lets you manage the settings of different DAServer configurations.

The DAServer contains a default configuration set named DASSIDirect. You cannot delete the default configuration set.

Caution: Care should also be taken not to accidentally delete the configuration set file outside of the DAServer Manager using Windows Explorer.

You can create multiple configuration sets and switch between them. Archiving, clearing, and switching configuration sets can only be done when the DAServer is deactivated.

Before you create a configuration set, verify that you have saved any changes you made to the global parameters. If you change a parameter and then immediately create a configuration set, the original parameter value is saved as part of the configuration set, not the changed value.

To save a global parameter, click the **Save** icon.

To archive a configuration set

- 1** In the DAServer Manager, navigate to the configuration node.
 - a** Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
 - b** Expand the DAServer.
- 2** Click **Configuration**.
- 3** Right-click and click **Archive Configuration Set**.
- 4** In the dialog box, type the configuration set name, and click **Archive**. All the current configuration values are saved to the set.

After you archive at least one configuration set, you can select it for use.

To select a configuration set

- 1** In the DAServer Manager, navigate to the configuration node.
 - a** Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
 - b** Expand the DAServer.
- 2** Click **Configuration**.
- 3** Right-click, point to **Use Another Configuration Set**, then click the desired name.

To change the parameter values saved in a configuration set, make sure the desired configuration set is shown, then follow this procedure.

To change the parameter values in a configuration set

- 1** In the DAServer Manager, navigate to the configuration node.
 - a** Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
 - b** Expand the DAServer.
- 2** Click **Configuration**.
- 3** Change the parameters that you want to change.
- 4** Click the **Save** icon.

Clearing a configuration set returns the parameters to their default values.

To clear a configuration set

- 1** In the DAServer Manager, navigate to the configuration node.
 - a** Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
 - b** Expand the DAServer.
- 2** Click **Configuration**.
- 3** Right-click, move the mouse over **Clear Configuration Set**, then left click.
- 4** Read the warning message, then click **Yes**. The parameters are set to the default values.

To delete a configuration set

- 1** In the DAServer Manager, navigate to the configuration node.
 - a** Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
 - b** Expand the DAServer.
- 2** Click **Configuration**.
- 3** Right-click **Configuration**, point to **Delete Configuration Set** and select the configuration set to delete.
- 4** Read the warning message, then click **Yes**.

Activating/Deactivating the DAServer

When you activate the DAServer, it starts communicating and accepting requests from client applications. If a DAServer is configured as an automatic service, the DAServer is started and activated when the computer starts up. Also, a DAServer can be activated by the an OPC client connection request, but only out-of-proc.

To activate the DAServer

- 1 In the DAServer Manager, navigate to the DAServer.
 - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
- 2 Right-click **ArchestrA.DASSIDirect.3** and then click **Activate Server**.

Deactivating your DAServer stops it from communicating with client applications.

A DAServer with active OPC clients does not stop until the last OPC client shuts down.

To deactivate the DAServer

- 1 In the DAServer Manager, navigate to the DAServer.
 - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
- 2 Right-click **ArchestrA.DASSIDirect.3** and then click **Deactivate Server**.
- 3 Read the warning message and click **Yes**.

In-Proc/Out-of-Proc

The DAServer can run only as a stand-alone process (out-of-proc). If the CLXCTX_ALL option is the chosen, out-of-proc activation for the DAServer is triggered. Explicitly starting as part of the client process (in-proc) is not currently supported. Activation using the CLSCTX_ACTIVATE _64_BIT_SERVER flag is also not supported.

When the DAServer is running out-of-proc, it supports requests from both DDE/SuiteLink and OPC client applications, as well as VB clients.

If the DAServer is running as a service, the icon on the DAServer node in the SMC is yellow. If the DAServer is not running as a service, the icon is white. For more information, see the *DAServer Manager User's Guide*.

Hot Configuration

If a parameter value change takes effect right away while the DAServer is running, the parameter is a hot-configurable parameter. Certain parameters in the SIDirect DAServer are hot-configurable. Incorporated in the DAServer are the following hot-configuration functionalities:

- Modifying Global Configuration parameters.
- Adding, deleting, or modifying device nodes (without affecting any other device nodes, excluding the children of the modified device nodes).
- Adding, deleting, or modifying device groups, the **Update Interval** column in the **Device Groups** tab, and device items.

Limited support is provided for the hot configuration for the server-specific configuration parameters in this release. You can modify server-specific parameters while the server is active. However, to have those changes take effect, you have to restart the DAServer.

The following parameters are hot configurable. They can be modified online and changes will take affect without restarting the DAServer.

- Replay timeout
- String variable style
- Register type

Note: If changes are made to server-specific parameters while the server is active, the DAServer will issue a warning message to the logger.

Demo Mode

You can install a fully functioning version of this SIDirect DAServer for demonstration purposes without a license. Demo mode allows you to test the functionality of the server for 120 minutes. After that time, you must install a license to continue using the DAServer.

When you first start this SIDirect DAServer, it checks for a license. If the DAServer cannot find a valid license installed on the local computer, it logs a warning message indicating a valid license cannot be retrieved and enters Demo mode. Thereafter, the SIDirect DAServer repeats its request for the license every 30 seconds. If no licenses are found, the DAServer again logs a warning message on the issue.

This process is repeated for 120 minutes, after which the DAServer stops updating read/write on all device items. Read from cache is allowed, but all non-system data receive Bad quality status. The SIDirect DAServer continues to request a license and clients continue to function normally. For example, you can still add or remove an item, but its quality is set to Bad until a license is obtained.

Note: Use the \$SYS\$Licensed system item, a read-only Boolean item, to check the status of your license: True for Licensed and False for Not Licensed.

If you subsequently add a license to the License Manager, the DAServer logs a message acknowledging the license, switches out of Demo mode, and runs normally.

Note: After a DAServer obtains a valid license, it no longer checks for a license. If your license expires while the DAServer is running, the DAServer continues to function normally until it is stopped.

Chapter 6

Accessing the Data in Your DAServer

The method for accessing items through the DAServer depends on the communications protocol used.

Accessing Items Using the OPC Communications Protocol

In the case of OPC communications, the protocol addresses an element of data in a conversation with six characteristics:

- node name
- program name
- group name
- device group
- link name
- item name

The node name and device group are optional. A fully qualified OPC item name (ItemID) is composed of the link name and item name. All other characteristics are specified through separate DAServer means.

To access an OPC item, the OPC client connects to the DAServer, either in-process or out-of-process, and creates an OPC group defining the data-acquisition properties for the collection of items to be added. OPC groups can be either public or private. Public OPC groups are shared across multiple clients. Private OPC groups are local to a single client.

Note: DAServers only support private OPC groups.

You can optionally specify a device group, which indicates the access path to the items for Read/Write.

The following briefly describes each characteristic of the OPC protocol.

- **node name**
Computer (host) name identifying a specific node on the network. This node name is required for remote access and is optional for local access.
- **program name**
The registered OPC server name uniquely identifying a specific server (ProgID).
For this DAServer, the program name is **ArchestrA.DASSIDirect.3**.
- **group name**
The OPC group created from the client for logically organizing a collection of items with the same data-acquisition properties between the client and the server, such as update rate.
- **device group**
Meaningful names configured in the DAServer under a specific controller for the common custom attributes between the DAServer and the device, such as update interval.
If not specified from the client, the default device group using the global-configuration attribute values from the DAServer is assumed.
Functionally, a device group is equivalent to an access path (optional).

- **link name**
The set of hierarchy node names separated by delimiters. Link names represent the specific devices on a communications path link from the hierarchy root to a specific controller as configured for this DAServer under the DAServer Manager.
- **item name**
A specific data element, the leaf of the hierarchy tree of this DAServer, within the specified group.
For example, when using this DAServer, an item can be a relay, timer, counter, register, and so on, in the controller.

Accessing Items Using the DDE/SuiteLink Communications Protocol

In the case of DDE/SuiteLink communications, the protocol addresses an element of data in a conversation that uses a four-part naming convention. The naming convention includes the node name, application name, topic name, and item name. The fully qualified DDE/SuiteLink naming convention includes all four parts, although the optional node-name is only required for remote access.

Important: On Windows Vista and later operating systems, Local DDE is supported only when the DAServer is configured as "Not a Service" and activated from its executable file or started from the InTouch HMI. Local DDE is not supported when the DAServer is activated in the System Management Console (SMC).

The following briefly describes each portion of this naming convention.

- **node name**
Computer name or host name identifying a specific node on the network. This node name is required for remote access and is optional for local access.
- **application name**
The name of the Windows program (this DAServer) that accesses the data element. In the case of data coming from or going to the Siemens devices via the DDE/SuiteLink Plugin of this DAServer, the application name portion of the address is **DASSIDirect**.

- **topic name**

Meaningful names configured in the DAServer to identify specific devices. These names are used as the topic names in all conversations with that device.

For example, **FASTPOLL**.

Topic names map to a device group defined in the DAServer.

Note: You can define multiple device group (topic) names for the same device (PLC) to poll different points or items at different rates.

- **item name**

A specific data element within the specified topic.

For example, when using this DAServer, an item can be a relay, timer, counter, register, and so on, in the PLC.

For more information on item names, see "Item Naming" on page 61.

Chapter 7

Item Naming

The Wonderware SIDirect DAServer uses an item-naming convention based on the two-letter data-type identifiers used in programming the Siemens PLCs. The server accepts both the English and German standard identifiers.

The tables in this section describe the item naming convention for the Siemens S7 PLCs. The ranges specified in those tables vary according to the type of the controller used.

Note: PDU (Protocol Data Unit) size is factored in determining Read/Write size limits. The S7-300 family of controllers can process PDU lengths of 480 to 960 bytes, depending on the controller. An S7-400 controller can process PDU lengths of 960 bytes.

Data Blocks and Instance Blocks

The following table summarizes the data format, item or point, suffix, data type, and range for Data Blocks and Instance Blocks.

See the S7-200, S7-1200, and S7-1500 sections in this documentation for specific item naming conventions and/or limitations for those controllers.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	D<B,I>d,Xx.y		VT_BOOL	0 or 1
String	D<B,I>d,Sx.v		VT_BSTR	String
	D<B,I>d,STRINGx.v		VT_BSTR	String

Data Format	Item/Point	Suffix	Data Type	Range
S7String	D<B,I>d,S7Sx,w		VT_BSTR	String
	D<B,I>d,S7STRINGx,w		VT_BSTR	String
Byte	D<B,I>d,Bx		VT_UI1	0 to 255
	D<B,I>d,BYTEx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	D<B,I>d,Bx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	D<B,I>d,BYTEx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I>d,USINTx	DT	VT_UI1	0 to 255
			VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I>d,USINTx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	D<B,I>d,CHARx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	D<B,I>d,CHARx,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
SINT	D<B,I>d,SINTx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I>dSINTx,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	D<B,I>d,Wn		VT_UI2	0 to 65535
	D<B,I>d,WORDn		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	D<B,I>d,Wn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	D<B,I>d,WORDn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
UINT	D<B,I>d,UINTn		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	D<B,I>d,UINTn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	D<B,I>d,INTn		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	D<B,I>d,INTn,v		VT_ARRAY VT_UI1	-32768 to 32767 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Double Word	D<B,I>d,Dm		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	D<B,I>d,Dm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	D<B,I>d,DWORDm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
UDINT	D<B,I>d,UDINTm		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	D<B,I>d,UDINTm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	D<B,I>d,DINTm		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	D<B,I>d,DINTm,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
Real	D<B,I>d,REALm		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	D<B,I>d,REALm,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big endian format when data is returned to the DDE/SuiteLink layer.

***: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- d is the data block number, with a range from 1 to 65535.
- x is the start address. For S7-300/400/1200/1500 PLCs, the range is from 0 to 65535. For S7-200 PLCs, the value is 1.

Where:

- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays.
- m is the start address of 4-byte data/4-byte data arrays.
- v is the length of data in elements (an item in an array), with a range from 0 to 65534.
- w is the length of the net S7 string data in characters (size in S7 message is w+1, size of string representation in S7 PLC is w+2).

***: The SIDIrect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All data blocks are **Read/Write**. The longest string or array that can be read in a cyclic service has the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The DAServer processes a write (**POKE**) to a Data Block.

Examples for S7-300/400/1200/1500 PLCs:

DB123,W24

DB23,DINT10BCD

DI5,X2.0

DI6,BYTE4,10

Flag Bytes

The following table summarizes data format, item or point, suffix, data type, and range for Flag Bytes.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	FX x,y		VT_BOOL	0 or 1
	MX x,y		VT_BOOL	0 or 1
String	FS x,v		VT_BSTR	String
	MS x,v		VT_BSTR	String
	FSTRING x,v		VT_BSTR	String
	MSTRING x,v		VT_BSTR	String
Byte	FB x		VT_UI1	0 to 255
	MB x		VT_UI1	0 to 255
	FBYTE x		VT_UI1	0 to 255
	MBYTE x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	FB x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	MB x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	FBYTE x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	MBYTE x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I> d ,USINT x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I> d ,USINT x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	FCHAR x		VT_I1	-128 to 127
	MCHAR x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	FCHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
	MCHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I> d ,SINT x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I> d SINT x,v		VT_ARRAY VT_UI1	-128 to 127 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Word	FW_n		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30ms
		TR	VT_R4	0.0 to 9990.0 (s)
Word Array	$FW_{n,v}$		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
UINT	$D<B,I>d,UINT_n$		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
UINT Array	$D<B,I>d,UINT_{n,v}$		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
Integer	$FINT_n$		VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30ms
		TR	VT_R4	0.0 to 9990.0 (s)
Integer Array	$FINT_{n,v}$		VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30ms
		TR	VT_R4	0.0 to 9990.0 (s)
Integer Array	$MINT_{n,v}$		VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30ms
		TR	VT_R4	0.0 to 9990.0 (s)

Data Format	Item/Point	Suffix	Data Type	Range
Double Word	FD m		VT_UI4	0 to 4294967295**
	MD m		VT_UI4	0 to 4294967295**
	FDWORD m		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
	MDWORD m	BCD	VT_UI4	0 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	FD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	MD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	FDWORD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	MDWORD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
UDINT	D<B,I> d , UDINT m		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	D<B,I> d , UDINT m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	FDINT m		VT_14	-2147483648 to 2147483647
	MDINT m		VT_14	-2147483648 to 2147483647
		BCD	VT_14	-9999999 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	FDINT m,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
	MDINT m,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Real	FREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
	MREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	FREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	MREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 65535.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All flags are **Read/Write** ($x=0$ to 65535, $y=0$ to 7, $n=0$ to 65534, $m=0$ to 65532, $v=1$ to net PDU data size/type size - header information, this size may vary). The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The DAServer processes a write (**POKE**) to a Flag Byte.

Input Bytes

The following table summarizes the data format, item or point, suffix, data type, and range for Input Bytes.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	Ix,y		VT_BOOL	0 or 1
	Ex,y		VT_BOOL	0 or 1
	IXx,y		VT_BOOL	0 or 1
	EXx,y		VT_BOOL	0 or 1
String	ISx,v		VT_BSTR	String
	ESx,v		VT_BSTR	String
	ISTRINGx,v		VT_BSTR	String
	ESTRINGx,v		VT_BSTR	String
Byte	IBx		VT_UI1	0 to 255
	EBx		VT_UI1	0 to 255
	IBYTEx		VT_UI1	0 to 255
	EBYTEx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	IBx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	EBx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	IBYTEx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	EBYTEx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I>d,USINTx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I>d,USINTx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	ICHARx		VT_I1	-128 to 127
	ECHARx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	ICHARx,v		VT_ARRAY VT_I1	-128 to 127 for each element*
	ECHARx,v		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I>d,SINTx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***

Data Format	Item/Point	Suffix	Data Type	Range
SINT Array	D<B,I>dSINT _{x,v}		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	IW _n		VT_UI2	0 to 65535
	EW _n		VT_UI2	0 to 65535
	IWORD _n		VT_UI2	0 to 65535
	EWORD _n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	IW _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	EW _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	IWORD _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	EWORD _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
UINT	D<B,I>d,UINT _n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	D<B,I>d,UINT _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	IINT _n		VT_I2	-32768 to 32767
	EINT _n		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	IINT _{n,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	EINT _{n,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Double Word	ID_m		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
Double Word Array	$ID_{m,v}$	T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
			VT_ARRAY VT_UI4	0 to 4294967295 for each element*
			VT_ARRAY VT_UI4	0 to 4294967295 for each element
			VT_ARRAY VT_UI4	0 to 4294967295 for each element*
			VT_ARRAY VT_UI4	0 to 4294967295 for each element
			VT_ARRAY VT_UI4	0 to 4294967295 for each element
UDINT	$D<B,I>d, UDINT_m$		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	$D<B,I>d, UDINT_{m,v}$		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	$IDINT_m$		VT_I4	-2147483648 to 2147483647
			VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
			VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
Double Integer Array	$IDINT_{m,v}$		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
			VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Real	IREAL m		VT_R4	+/-1.2e-38 to +/-3.4e+38
	EREAL m		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	IREAL m,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	EREAL m,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 65535.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All inputs are **Read-Only**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The DAServer does **not** process a write (**POKE**) to an Input Byte.

Output Bytes

The following table summarizes data format, item or point, suffix, data type, and range for Output Bytes.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	$Ox.y$		VT_BOOL	0 or 1
	$Ax.y$		VT_BOOL	0 or 1
	$Qx.y$		VT_BOOL	0 or 1
	$OXx.y$		VT_BOOL	0 or 1
	$AXx.y$		VT_BOOL	0 or 1
	$QXx.y$		VT_BOOL	0 or 1
String	OSx,v		VT_BSTR	String
	ASx,v		VT_BSTR	String
	QSx,v		VT_BSTR	String
	$OSTRINGx,v$		VT_BSTR	String
	$ASTRINGx,v$		VT_BSTR	String
	$QSTRINGx,v$		VT_BSTR	String
Byte	OBx		VT_UI1	0 to 255
	ABx		VT_UI1	0 to 255
	QBx		VT_UI1	0 to 255
	$OBYTEx$		VT_UI1	0 to 255
	$ABYTEx$		VT_UI1	0 to 255
	$QBYTEx$		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	OBx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	ABx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	QBx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	$OBYTEx,v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	$ABYTEx,v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	$QBYTEx,v$		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	$D<B,I>d,USINTx$		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	$D<B,I>d,USINTx,v$		VT_ARRAY VT_UI1	0 to 255 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Char	OCHAR x		VT_I1	-128 to 127
	ACHAR x		VT_I1	-128 to 127
	QCHAR x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	OCHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
	ACHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
	QCHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I> d ,SINT x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I> d SINT x,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	OW n		VT_UI2	0 to 65535
	AW n		VT_UI2	0 to 65535
	QW n		VT_UI2	0 to 65535
	OWORD n		VT_UI2	0 to 65535
	AWORD n		VT_UI2	0 to 65535
	QWORD n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	OW n,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	AW n,v		VT_ARRAY VT_UI2	0 to 65535 for each element
	QW n,v		VT_ARRAY VT_UI2	0 to 65535 for each element
	OWORD n,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	AWORD n,v		VT_ARRAY VT_UI2	0 to 65535 for each element
	QWORD n,v		VT_ARRAY VT_UI2	0 to 65535 for each element
UINT	D<B,I> d ,UINT n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31

Data Format	Item/Point	Suffix	Data Type	Range
UINT Array	D<B,I> <i>d</i> ,UINT <i>n</i> , <i>v</i>		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	OINT <i>n</i>		VT_I2	-32768 to 32767
	AINT <i>n</i>		VT_I2	-32768 to 32768
	QINT <i>n</i>		VT_I2	-32768 to 32768
		BCD		-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	OINT <i>n</i> , <i>v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	AINT <i>n</i> , <i>v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	QINT <i>n</i> , <i>v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	OD <i>m</i>		VT_UI4	0 to 4294967295**
	AD <i>m</i>		VT_UI4	0 to 4294967295**
	QD <i>m</i>		VT_UI4	0 to 4294967295**
	ODWORD <i>m</i>		VT_UI4	0 to 4294967295**
	ADWORD <i>m</i>		VT_UI4	0 to 4294967295**
	QDWORD <i>m</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	OD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	AD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	QD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	ODWORD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	ADWORD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	QDWORD <i>m</i> , <i>v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
UDINT	D<B,I> <i>d</i> , UDINT <i>m</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	D<B,I> <i>d</i> , UDINT <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	ODINT <i>m</i> ADINT <i>m</i> QDINT <i>m</i>		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-9999999 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	ODINT <i>m,v</i>		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
	ADINT <i>m,v</i>			
	QDINT <i>m,v</i>			
Real	OREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
	AREAL <i>m</i>			
	QREAL <i>m</i>			
Real Array	OREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	AREAL <i>m,v</i>			
	QREAL <i>m,v</i>			

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

x is the start address, with a range from 0 to 65535.

y is the bit position, with a range from 0 to 7.

● 0 is the LSB (Least Significant Bit).

● 7 is the MSB (Most Significant Bit).

Where:

- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All outputs are **Read/Write**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The DAserver processes a write (**POKE**) to an Output Byte.

Peripheral Input Bytes

The S7-200 PLCs do **not** support Peripheral Input Bytes.

The following table summarizes the data format, item or point, suffix, data type, and range for Peripheral Input Bytes.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	PIx.y		VT_BOOL	0 or 1
	PEx.y		VT_BOOL	0 or 1
	PIXx.y		VT_BOOL	0 or 1
	PEXx.y		VT_BOOL	0 or 1
String	PISx,v		VT_BSTR	String
	PESx,v		VT_BSTR	String
	PISTRINGx,v		VT_BSTR	String
	PESTRINGx,v		VT_BSTR	String

Data Format	Item/Point	Suffix	Data Type	Range
Byte	PIB x		VT_UI1	0 to 255
	PEB x		VT_UI1	0 to 255
	PIBYTE x		VT_UI1	0 to 255
	PEBYTE x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	PIB x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	PEB x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	PIBYTE x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	PEBYTE x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I> d ,USINT x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I> d ,USINT x,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	PICHA Rx		VT_I1	-128 to 127
	PECHA Rx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	PICHA Rx,v		VT_ARRAY VT_I1	-128 to 127 for each element*
	PECHA Rx,v		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I> d ,SINT x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I> d ,SINT x,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	PIW n		VT_UI2	0 to 65535
	PEW n		VT_UI2	0 to 65535
	PIWORD n		VT_UI2	0 to 65535
	PEWORD n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31

Data Format	Item/Point	Suffix	Data Type	Range
Word Array	PIW _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PEW _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PIWORD _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PEWORD _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
UINT	D<B,I>d,UINT _n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	D<B,I>d,UINT _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	PIINT _n		VT_I2	-32768 to 32767
	PEINT _n		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	PIINT _{n,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	PEINT _{n,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	PID _m		VT_UI4	0 to 4294967295**
	PED _m		VT_UI4	0 to 4294967295**
	PIDWORD _m		VT_UI4	0 to 4294967295**
	PEDWORD _m		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	PID _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	PED _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element
	PIDWORD _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element
	PEDWORD _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element

Data Format	Item/Point	Suffix	Data Type	Range
UDINT	D<B,I> <i>d</i> , UDINT <i>m</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	D<B,I> <i>d</i> , UDINT <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	PIDINT <i>m</i>		VT_I4	-2147483648 to 2147483647
	PEDINT <i>m</i>		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	PIDINT <i>m,v</i>		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
	PEDINT <i>m,v</i>			
Real	PIREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
	PEREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	PIREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	PEREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 65535.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.

Where:

- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All peripheral inputs are **Read-Only**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The DAServer does **not** process a write (**POKE**) to a Peripheral Input Byte. Some input modules are not readable.

Peripheral Output Bytes

The S7-200 PLCs do **not** support Peripheral Output Bytes.

The following table summarizes the data format, item or point, suffix, data type, and range for Peripheral Output Bytes.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	PO $x.y$		VT_BOOL	0 or 1
	PA $x.y$		VT_BOOL	0 or 1
	PQ $x.y$		VT_BOOL	0 or 1
	POX $x.y$		VT_BOOL	0 or 1
	PAX $x.y$		VT_BOOL	0 or 1
	PQX $x.y$		VT_BOOL	0 or 1
String	POS $x.v$		VT_BSTR	String
	PAS $x.v$		VT_BSTR	String
	PQS $x.v$		VT_BSTR	String
	POSTRING $x.v$		VT_BSTR	String
	PASTRING $x.v$		VT_BSTR	String
	PQSTRING $x.v$		VT_BSTR	String
Byte	POB x		VT_UI1	0 to 255
	PAB x		VT_UI1	0 to 255
	PQB x		VT_UI1	0 to 255
	POBYTE x		VT_UI1	0 to 255
	PABYTE x		VT_UI1	0 to 255
	PQBYTE x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	POB $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	PAB $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	PQB $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	POBYTE $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	PABYTE $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
	PQBYTE $x.v$		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I> d ,USINT x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***

Data Format	Item/Point	Suffix	Data Type	Range
USINT Array	D<B,I> <i>d</i> ,USINT _{<i>x</i>} , <i>v</i>		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	PACHAR _{<i>x</i>}		VT_I1	-128 to 127
	POCHAR _{<i>x</i>}		VT_I1	-128 to 127
	PQCHAR _{<i>x</i>}		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	POCHAR _{<i>x,v</i>}		VT_ARRAY VT_I1	-128 to 127 for each element*
	PACHAR _{<i>x,v</i>}		VT_ARRAY VT_I1	-128 to 127 for each element*
	PQCHAR _{<i>x,v</i>}		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I> <i>d</i> ,SINT _{<i>x</i>}		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I> <i>d</i> SINT _{<i>x,v</i>}		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	POW _{<i>n</i>}		VT_UI2	0 to 65535
	PAW _{<i>n</i>}		VT_UI2	0 to 65535
	PQW _{<i>n</i>}		VT_UI2	0 to 65535
	POWORD _{<i>n</i>}		VT_UI2	0 to 65535
	PAWORD _{<i>n</i>}		VT_UI2	0 to 65535
	PQWORD _{<i>n</i>}		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	POW _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PAW _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PQW _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	POWORD _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PAWORD _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*
	PQWORD _{<i>n,v</i>}		VT_ARRAY VT_UI2	0 to 65535 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
UINT	D<B,I> <i>d</i> ,UINT <i>n</i>		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	D<B,I> <i>d</i> ,UINT <i>n,v</i>		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	POINT <i>n</i>		VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
			VT_I2	-32768 to 32767
	PAINT <i>n</i>	BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
	PQINT <i>n</i>			
Integer Array	POINT <i>n,v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	PAINT <i>n,v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	PQINT <i>n,v</i>		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	POD <i>m</i>		VT_UI4	0 to 4294967295**
	PAD <i>m</i>		VT_UI4	0 to 4294967295**
	PQD <i>m</i>		VT_UI4	0 to 4294967295**
	PODWORD <i>m</i>		VT_UI4	0 to 4294967295**
	PADWORD <i>m</i>		VT_UI4	0 to 4294967295**
	PQDWORD <i>m</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	POD <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	PAD <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	PQD <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
UDINT	D<B,I>d,UDINT <i>m</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
UDINT Array	D<B,I>d,UDINT <i>m,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	PODINT <i>m</i>		VT_I4	-2147483648 to 2147483647
			VT_I4	-2147483648 to 2147483647
			VT_I4	-2147483648 to 2147483647
	PADINT <i>m</i>	BCD	VT_I4	-9999999 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	PODINT <i>m,v</i>		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
	PADINT <i>m,v</i>		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
	PQDINT <i>m,v</i>		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
Real	POREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
	PAREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
	PQREAL <i>m</i>		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	POREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	PAREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*
	PQREAL <i>m,v</i>		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 65535.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All peripheral outputs are **Write-Only**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. All output modules are not readable. Only POKES are allowed.

Counters

The following table summarizes data format, item or point, suffix, data type, and range for Counters.

Data Format	Item/Point	Suffix	Data Type	Range
Word	Cx	None	VT_UI2	0...65535
	Zx	None	VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s

Where:

x is the start address, with a range from 0 to 65535.

When the suffix is used, the client is responsible for ensuring the correct value is specified or returned.

Note: S7-1200 and S7-1500 PLCs do not support "direct" counter\timer access by the client/server. That is, you cannot use the current S7-300/400 item name Cx or Tx and its associated suffixes to subscribe to counter\timer values from the S7-1200 and S7-1500 PLCs. However, you can use the database items to indirectly output the associated system counter\timer value (internal IEC counter\timer) from the PLC.

For example:

If the value of counter C0 is 42, the value read by just using the "C0" item syntax is 42. However, if the item syntax is "C0 BCD," the value returned by the DAServer is 66.

Similar principle applies to poking:

If the value for the counter is 42, using the "C0" pokes a value of 42 into the counter C0. However, with the "C0 BCD" syntax, the poke value is 66.

Note: All counters are **Read/Write**. The DAServer processes a write (**POKE**) to a counter. Although the DAServer allows poking any word value into counters, the S7 PLC can only process values in the range of 0...2457 or 0...999 (BCD).

Timers

For S7-200 Timers, see "S7-200 Timers" on page 110.

The following table summarizes data format, item or point, suffix, data type, and range only for the S7-300, S7-400, and S7-1200 Timers.

Data Format	Item/Point	Suffix	Data Type	Range
Word	Tx	None	VT_UI2	0 to 14745
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
	TREALx	None	VT_R4	0.0 to 9990.00

Where:

x is the start address, with a range from 0 to 65535.

When the suffix is used, the client is responsible for ensuring the correct value is specified or returned.

Note: S7-1200 and S7-1500 PLCs do not support "direct" counter\timer access by the client/server. That is, you cannot use the current S7-300/400 item name Cx or Tx and its associated suffixes to subscribe to counter\timer values from the S7-1200 and S7-1500 PLCs. However, you can use the database items to indirectly output the associated system counter\timer value (internal IEC counter\timer) from the PLC.

For example:

If the value of timer T0 is 42, the value read by just using the "T0" item syntax is 42. However, if the item syntax is "T0BCD," the value returned by the DAServer is 66.

Similar principle applies to poking:

If the value for the timer is 42, using the "T0" pokes a value of 42 into the timer T0. However, with the "T0BCD" syntax, the poke value is 66.

Note: All timers are **Read/Write**. The DAServer processes a write (**POKE**) to a timer. Although the DAServer allows poking any word value into timers, the S7 PLC can only process values that represent a valid time format.

Block Items

The SIdirect DAsServer supports Block Items for the S7-300, S7-400, and S7-1500 PLCs. The server does **not** support Block Items for the S7-200 and S7-1200 PLCs.

The Block Items have two sets of items:

- Read-Only Block Items
- Write-Only Block Items

Read-Only Block Items

The following table summarizes the data format, item or point, suffix, data type, and range for Read-Only Block Items.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	BLd,Xx.y		VT_BOOL	0 or 1
String	BLd,Sx,v		VT_BSTR	String
	BLd,STRINGx,v		VT_BSTR	String
Byte	BLd,Bx		VT_UI1	0 to 255
	BLd,BYTEx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	BLd,Bx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	BLd,BYTEx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I>d,USINTx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I>d,USINTx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	BLd,CHARx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	BLd,CHARx,v		VT_ARRAY VT_I1	-128 to 127 for each element*
SINT	D<B,I>d,SINTx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I>d,SINTx,v		VT_ARRAY VT_UI1	-128 to 127 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Word	<i>BLd,Wn</i>		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word	<i>BLd,Wn,v</i>		VT_ARRAY VT_UI2	0 to 65535 for each element*
Array	<i>BLd,WORDn,v</i>		VT_ARRAY VT_UI2	0 to 65535 for each element*
UINT	<i>D<B,I>d,UINTn</i>		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	<i>D<B,I>d,UINTn,v</i>		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	<i>BLd,INTn</i>		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	<i>BLd,INTn,v</i>		VT_BSTR	-32768 to 32767 for each element*
Double Word	<i>BLd,Dm</i>		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	<i>BLd,Dm,v</i>		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	<i>BLd,DWORDm,v</i>			0 to 4294967295 for each element**
UDINT	<i>D<B,I>d,UDINTm</i>		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS

Data Format	Item/Point	Suffix	Data Type	Range
UDINT Array	D<B,I>d,UDINT m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	BLd,DINT m		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-9999999 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	BLd,DINT m,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element
Real	BLd,REAL m		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	BLd,REAL m,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 65535.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

Note: All Block items are **Read-Only**. The longest string or array that can be read in a block service is the length of 65534 bytes. The longest string the InTouch software can process is 131 bytes.

Write-Only Block Items

The following table summarizes the data format, item or point, suffix, data type, and range for Write-Only Block Items.

Data Format	Item/Point	Suffix	Data Type	Range
Byte Array	BW <i>d</i>		VT_ARRAY VT_UI1	0 to 255 for each element*,** Note also that the DAServer does not cache the value of the item written to the PLC. 0 to 255 for each element*,***
	BWC <i>d-q,x</i>		VT_ARRAY VT_UI1	Note also that the DAServer caches the value of the item written to the PLC.
	BWC <i>d-q.Send</i>		VT_BOOL	TRUE (1)***,**** FALSE (0)***,**** The value of item (BWC <i>d-q.x</i>), cached by the DAServer, is sent to the PLC when this item is transitioned from FALSE to TRUE. The value of this item remains TRUE until the appropriate acknowledgment is received from the PLC. After the acknowledgment is received from the PLC, the value of this item is set to 0.

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

: The starting address is always at 0 for **Writes. The number of bytes written to the PLC block is determined from the length of the input byte stream. However, the length of the byte stream must be less than 65535.

***: The Siemens S7 block write protocol BSEND does not allow a starting address. All block writes to the PLC always start at address 0. To circumvent this limitation, the BWC item syntax allows for a starting address. The logical number of bytes written to the PLC block is determined from the difference in the start address and end address, (q-x+1), or the length of the input byte stream.

If the input byte stream is longer than the $(q-x+1)$, only $(q-x+1)$ bytes are written.

If the input byte stream is shorter than $(q-x+1)$, the whole input byte stream is written.

Internally, the DAServer allocates cache buffers based on the PLC data block and the end address. For more information, see Block Write Caching.

****: The *BWCd-q.Send* item is readable to allow monitoring of the status of the block send function.

Where:

d is the block ID, in decimal, with a range from 0 to 4294967295.

x is the start address, with a range from 0 to 65535.

q is the end address, with a range from 0 to 65535.
It must be equal or greater than x.

Note: All BW and BWC block items are **Write-Only**. BWCxx.Send items are **Read/Write**. The longest string or array that can be written in a block service is the length of 65534 bytes. The longest string the InTouch software can process is 131 bytes.

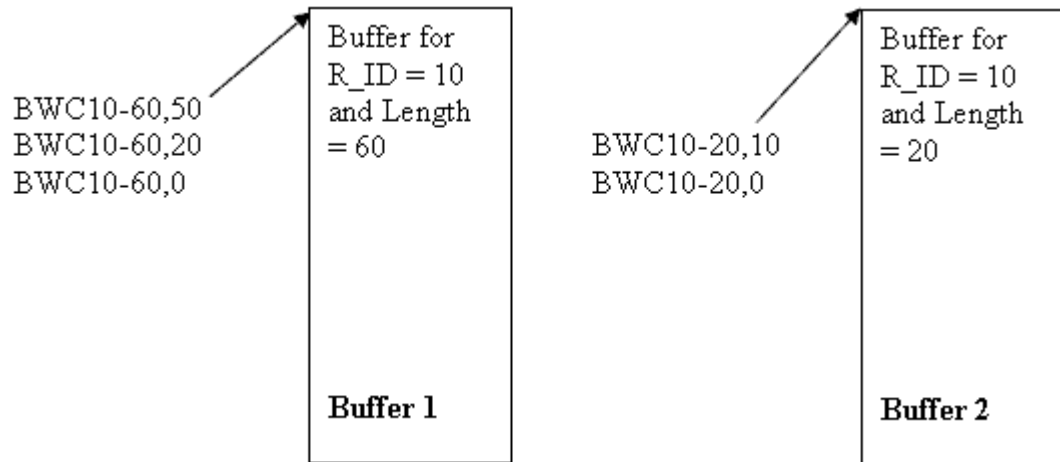
Block Write Caching

BWC is a special item that caches the data before sending it to the PLC. The item syntax for the BWC item is:

BWC<R_ID>.<length>,<start_address>

The R_ID is the ID that is configured for the Block Write in the PLC. A caching buffer is created, based on R_ID and length as an index, inside the server whenever you advise the BWC item. The buffer is not created if one already exists for the same R_ID and length.

For example, you advise the following items: BWC10-60,50, BWC10-60,20, BWC10-60,0, BWC10-20,10 and BWC10-20,0. Only two buffers are created inside the server for these items:



Consider following scenarios:

- 1 You poke 20 bytes of data in item BWC10-60,0. Buffer-1 is filled from byte 0 to 19.
- 2 You poke 15 bytes of data in item BWC10-60,50. Buffer-1 is filled from byte 50 to 59. The last 5 bytes are truncated because only 10 bytes are available in that offset.
- 3 You poke 30 bytes of data in item BWC10-60,20. Buffer-1 is filled from byte 20 to 49.
- 4 You poke 20 bytes of data in item BWS10-20,0. Buffer-2 is filled from byte 0 to 19.
- 5 You poke 1 to item BWC10-20.Send. Buffer-2 is flushed to the PLC with byte 20.
- 6 You poke 1 to item BWC10-60.Send. Buffer-1 is flushed to the PLC with byte 60.

Alarms and Events

Note: The S7-200, S7-1200 and S7-1500 PLCs do **not** provide alarms and events capability.

Alarm and event information can be received from the S7-300 and S7-400 PLCs. The item syntax for Alarms and Events is as follows:

ALARM<EV_ID>.<Extension 1>[,<Extension 2>[<Suffix>]]

EVENT<EV_ID>.<Extension 1>[,<Extension 2>[<Suffix>]]

The following table shows valid values and valid value combinations for Extension 1, Extension 2, and Suffix.

Item	Extension 1	Extension 2	Suffix	Data Type	Range
ALARM	EVENT_STATE			VT_UI2	0 to 65535
<EV_ID>	STATE			VT_UI2	0 to 65535
	ACK_STATE			VT_UI2	0 to 65535
	TIME_STAMP			VT_BSTR	String*, ****
	NO_ADD_VALUES			VT_UI2	0 to 10
	ADD_VALUE _w	DATA_TYPE		VT_BSTR	String
		LENGTH		VT_UI2	0 to 65535
		X _{x,y}		VT_BOOL	0 or 1
		S _{x,v}		VT_BSTR	String
		STRING _{x,v}		VT_BSTR	String
		B _x		VT_UI1	0 to 255
		BYTE _x		VT_UI1	0 to 255
			DT	VT_BSTR	String
		B _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element**
		BYTE _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element**
		CHAR _x		VT_I1	-128 to 127
			DT	VT_BSTR	String
		CHAR _{x,v}		VT_ARRAY VT_I1	-128 to 127 for each element**
		W _n		VT_UI2	0 to 65535
		WORD _n		VT_UI2	0 to 65535
			BCD	VT_UI2	0 to 9999
			KT	VT_BSTR	0.0 to 999.3
			S5T	VT_BSTR	0ms to 2h46m30s0ms
			D	VT_BSTR	String
		W _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element**
		WORD _{n,v}		VT_ARRAY VT_UI2	0 to 65535 for each element**
		INT _n		VT_I2	-32768 to 32767
			BCD	VT_I2	0 to 9999
			D	VT_BSTR	String

Item	Extension 1	Extension 2	Suffix	Data Type	Range
		INT _{n,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element**
		D _m		VT_UI4	0 to 4294967295***
		DWORD _m		VT_UI4	0 to 4294967295***
			BCD	VT_UI4	0 to 99999999
			T	VT_BSTR	String
			TOD	VT_BSTR	String
		D _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
		DWORD _{m,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
		DINT _m		VT_I4	-2147483648 to 2147483647
			BCD	VT_I4	0 to 99999999
			T	VT_BSTR	String
			TOD	VT_BSTR	String
		DINT _{m,v}		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element**
		REAL _m		VT_R4	+/-1.2e-38 to +/-3.4e+38
		REAL _{m,v}		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element**

Item	Extension 1	Extension 2	Suffix	Data Type	Range
EVENT <EV_ID>	EVENT_			VT_UI2	0 to 65535
	STATE				
	STATE			VT_UI2	0 to 65535
	ACK_STATE			VT_UI2	0 to 65535
	TIME_			VT_BSTR	String*, ****
	STAMP				
	NO_ADD_			VT_UI2	0 to 10
	VALUES				
	ADD_VALUE _w	DATA_TYPE		VT_BSTR	String
		LENGTH		VT_UI2	0 to 65535
		X _{x,y}		VT_BOOL	0 or 1
		S, _v		VT_BSTR	String
		STRING _{x,v}		VT_BSTR	String
		B _x		VT_UI1	0 to 255
		BYTE _x		VT_UI1	0 to 255
			DT	VT_BSTR	String
		B _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element**
		BYTE _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element**
		CHAR _x		VT_I1	-128 to 127
			DT	VT_BSTR	String
		CHAR _{x,v}		VT_ARRAY VT_I1	-128 to 127 for each element**

Item	Extension 1	Extension 2	Suffix	Data Type	Range
		W_n		VT_UI2	0 to 65535
		WORD n		VT_UI2	0 to 65535
			BCD	VT_UI2	0 to 9999
			KT	VT_BSTR	0.0 to 999.3
			S5T	VT_BSTR	0ms to 2h46m30s0ms
			D	VT_BSTR	String
		$W_{n,v}$		VT_ARRAY VT_UI2	0 to 65535 for each element**
		WORD n,v		VT_ARRAY VT_UI2	0 to 65535 for each element**
		INT n		VT_I2	-32768 to 32767
			BCD	VT_I2	0 to 9999
			D	VT_BSTR	String
		INT n,v		VT_ARRAY VT_I2	-32768 to 32767 for each element**
		D m		VT_UI4	0 to 4294967295***
		DWORD m		VT_UI4	0 to 4294967295***
			BCD	VT_UI4	0 to 99999999
			T	VT_BSTR	String
			TOD	VT_BSTR	String
		$D_{m,v}$		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
		DWORD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
		DINT m		VT_I4	-2147483648 to 2147483647
			BCD	VT_I4	0 to 99999999
			T	VT_BSTR	String
			TOD	VT_BSTR	String
		DINT m,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element**
		REAL m		VT_R4	+/-1.2e-38 to +/-3.4e+38
		REAL m,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element**

*: Starting with version 1.1, this string value can be used to timestamp other alarm items.

** : For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

***: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

****: For alarm blocks (such as SFB31, 34, 35) that provide multiple event states in one notification, the timestamp that comes with the notification reflects only the timestamp of the last changing state. This restriction is prescribed by the message that it receives from the PLC.

Where:

<EV_ID>	is the ID defined by Step7, in the integer format, filled with leading zeros up to six (6) characters.
x	is the start address, with a range from 0 to 65535.
y	is the bit position, with a range from 0 to 7.
	● 0 is the LSB (Least Significant Bit).
	● 7 is the MSB (Most Significant Bit).
n	is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 65534.
m	is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 65532.
v	is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).
w	is the length of the net S7 string-data in characters, with a range from 1 to 10.

Note: All alarms and events are **Read-Only**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The SIDirect DAServer does **not** process writes (**POKES**) to Alarms and Events.

Examples:

```
ALARM000010.TIME_STAMP
EVENT001234.ADD_VALUE2,LENGTH
ALARM000555.ADD_VALUE10,REAL0
EVENT000001.ADD_VALUE3,D0TOD
```

Alarm and event information is delivered from the S7-300/400 PLCs in the form of data items with the syntax described above. Alarm provider and alarm acknowledgment functionality is **not** supported.

The configuration of the alarms is first performed in the appropriate function blocks in the S7-300/400 PLCs as follows:

Name	SFB/SFC	S7 CPU
ALARM_SQ	SFC 17	S7-300/400
ALARM_S	SFC 18	S7-300/400
ALARM_DQ	SFC 107	S7-300/400
ALARM_D	SFC 108	S7-300/400
NOTIFY_8P	SFB 31	S7-400
ALARM	SFB 33	S7-400
ALARM_8	SFB 34	S7-400
ALARM_8P	SFB 35	S7-400
NOTIFY	SFB 36	S7-400

Events must be configured in the Symbol Editor.

Alarms and Events Terms

The following table lists the terms available in Alarms and Events Terms and their descriptions.

Term	Description
EVENT_STATE:	State of the Alarm/Event itself.
	If the Alarm/Event is TRUE, then EVENT_STATE is TRUE and vice versa.
	For more detailed information, see the Siemens Step7 documentation.
STATE:	The state in general whether the Alarm/Event is available. Maybe a data block is deleted where a bit should be monitored.
ACK_STATE:	The state of the acknowledgment of coming or going Alarms/Events.
	For more detailed information, see the Siemens Step7 documentation.
TIME_STAMP:	The timestamp of the Alarm/Event provided by the PLC.
NO_ADD_VALUES:	The number of additional values sent with this Alarm/Event message.
ADD_VALUE _w , DATA_TYPE:	The data type of a specific additional value of an Alarm/Event.

Term	Description
ADD_VALUE _w ,LENGTH:	The length of a specific additional value of an Alarm/Event.
<EV_ID>:	The event ID created automatically by the Step7 programming software. In case of Alarms (FB33 to FB36, SFC17/18/107/108), this is the EV_ID-parameter of the function block. The value of the parameter must be converted from hexadecimal to decimal, and then filled up with leading zeros to the length of 6 characters (for example: EV_ID: DW#16#4E25 => <EV_ID>: 020005). In case of Events (generated by the symbol editor) this is the "Message number." This number is in decimal format and must be filled with leading zeros up to 6 characters (for example: "Message number": 20000 => <EV_ID>: 020000).

Specific S7-200 Item Syntax

The S7-200 PLCs use different syntax than the S7-300, S7-400, and S7-1200 PLCs.

The memory address range in the S7-200-series PLCs varies based on the size of the CPU. Therefore, you can configure the SIDirect DAServer to read and write a PLC memory address that does not exist in the PLC. To prevent this from happening, ensure that the configured PLC item syntax does not exceed the PLC memory ranges of the PLC you are using.

V Memory

The S7-200 PLCs support V Memory.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	VX _{x,y}		VT_BOOL	0 or 1
String	VS _{x,v}		VT_BSTR	String
	VSTRING _{x,v}		VT_BSTR	String
Byte	VB _x		VT_UI1	0 to 255
	VBYTE _x		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Byte Array	VB _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element*
	VBYTE _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Char	VCHAR x		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999
Char Array	VCHAR x,v		VT_ARRAY VT_I1	-128 to 127 for each element*
Word	VW n		VT_UI2	0 to 65535
	VWORD n		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	VW n,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	VWORD n,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	VINT n		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	VINT n,v		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	VD m		VT_UI4	0 to 4294967295**
	VDWORD m		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	VD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	VDWORD m,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	VDINT m		VT_14	-2147483648 to 2147483647
		BCD	VT_14	-9999999 to 9999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS

Data Format	Item/Point	Suffix	Data Type	Range
Double Integer Array	VDINT m,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
Real	VREAL m		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	VREAL m,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- x is the start address, with a range from 0 to 10239.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays, with a range from 0 to 10238.
- m is the start address of 4-byte data/4-byte data arrays, with a range from 0 to 10236.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

Note: All V Memory bytes are **Read/Write**. The longest string or array that can be read in a cyclic service is the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The SIDirect DAServer processes a write (**POKE**) to V Memory.

Analog Input

The following table summarizes the data format, item or point, suffix, data type, and range for Analog Input.

Data Format	Item/Point	Suffix	Data Type	Range
Word	AIW x		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word Array	AIW x,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	AIWORD x,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	AIINT x		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	AIINT x,v		VT_ARRAY VT_I2	-32768 to 32767 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

Where:

- x is the start address, with a range from 0 to 62.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

Analog Output

The following table summarizes the data format, item or point, suffix, data type, and range for Analog Output.

Data Format	Item/Point	Suffix	Data Type	Range
Word	AOW x		VT_UI2	0 to 65535
	AQW x			
	AOWORD x		VT_UI2	0 to 65535
	AQWORD x			
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
Word Array		D	VT_BSTR	1990-1-1 to 2168-12-31
	AO x,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	AQ x,v			
	AOWORD x,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	AQWORD x,v			
Integer	AOINT x		VT_I2	-32768 to 32767
	AQINT x			
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	AOINT x,v		VT_ARRAY VT_I2	-32768 to 32767 for each element*
	AQINT x,v			

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

Where:

- x is the start address, with a range from 0 to 30.
- v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

Stage Memory

The following table summarizes the data format, item or point, suffix, data type, and range for Stage Memory.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	$SXx.k$		VT_BOOL	0 or 1
Byte	SBx		VT_UI1	0 to 255
Byte Array	SBx,v		VT_ARRAY VT_UI1	0 to 255*
Word	SWy		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
Word Array	SWy,v	D	VT_BSTR	1990-1-1 to 2168-12-31
			VT_ARRAY VT_UI2	0 to 65535 for each element*
			VT_ARRAY VT_UI2	0 to 65535 for each element*
			VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	$SINTy,v$		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	SDz		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 999999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	SDz,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
			VT_ARRAY VT_UI4	0 to 4294967295 for each element*

Data Format	Item/Point	Suffix	Data Type	Range
Double Integer	SDINT _z		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	SDINT _{z,v}		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

** : For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

k is the bit position, with a range from 0 to 7.

- 0 is the LSB (Least Significant Bit).
- 7 is the MSB (Most Significant Bit).

x is the start address, with a range from 0 to 31.

y is the start address, with a range from 0 to 30.

z is the start address, with a range from 0 to 28.

v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

Special Memory

The following table summarizes the data format, item or point, suffix, data type, and range for Special Memory.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	SM _{x,k}		VT_BOOL	0 or 1
Byte	SMB _x		VT_UI1	0 to 255
Byte Array	SMB _{x,v}		VT_ARRAY VT_UI1	0 to 255 for each element**

Data Format	Item/Point	Suffix	Data Type	Range
Word	SMW _y		VT_UI2	0 to 65535
			VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999
		KT	VT_BSTR	0.0 to 999.3
		S5T	VT_BSTR	0ms to 2h46m30s0ms
		TR	VT_R4	0.0 to 9990.0 (s)
		D	VT_BSTR	1990-1-1 to 2168-12-31
Word	SMW _{y,v}		VT_ARRAY VT_UI2	0 to 65535 for each element**
Array	SMWORD _{y,v}		VT_ARRAY VT_UI2	0 to 65535 for each element**
Integer	SMINT _y		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	SMINT _{y,v}		VT_ARRAY VT_I2	-32768 to 32767 for each element*
Double Word	SMD _z		VT_UI4	0 to 4294967295***
			VT_UI4	0 to 4294967295***
		BCD	VT_UI4	0 to 999999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Word Array	SMD _{z,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
	SMDWORD _{z,v}		VT_ARRAY VT_UI4	0 to 4294967295 for each element**
Double Integer	SMDINT _z		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS
Double Integer Array	SMDINT _{z,v}		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element

Special Memory addresses from 0 to 29 are **Read-Only**. Addresses above 29 are both **Read** and **Write**. If you perform a poke (a Write request) using addresses 0 through 29, the PLC will reject it.

**: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big-endian format when data is returned to the DDE/SuiteLink layer.

***: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

k is the bit position, with a range from 0 to 7.

● 0 is the LSB (Least Significant Bit).

● 7 is the MSB (Most Significant Bit).

x is the start address, with a range from 0 to 549.

y is the start address, with a range from 0 to 548.

z is the start address, with a range from 0 to 546.

v is the length of data in elements (an item in an array), with a range from 1 to (net PDU data size/type size - header information).

High-Speed Counter

The following table summarizes the data format, item or point, suffix, data type, and range for High-Speed Counter.

High speed counters are restricted to items HC0 through HC5. The DAServer returns Bad quality to high speed counter items with an address higher than 5.

The following table summarizes the data format, item or point, suffix, data type, and range for High-Speed Counter.

Data Format	Item/Point	Suffix	Data Type	Range
Double Word	HCx	x	VT_I4	-2147483648 to 2147483647*

*: The high speed counter item is **Read-Only**. The item type is double word. The value ranges only from -2147483648 to 2147483647.

Where:

x is the start address, with a range from 0 to 5.

S7-200 Timers

The following table summarizes the data format, item or point, suffix, data type, and range for S7-200 Timers.

Data Format	Item/Point	Suffix	Data Type	Range
Word	Tx		VT_UI2	0 to 32767*,***
Double Word	Tx	MS	VT_UI4	0 to 3276700 milliseconds**,***

*: The Tx item syntax returns or pokes the timer value in its native form as stored in the PLC. The timer resolution may be different for different timer addresses. Be aware of the timer resolution and convert it to the proper value for processing. See *** below.

** : The Tx MS item syntax translates the S7-200 counter value to its corresponding time in milliseconds. The SIDirect DAServer performs the conversion automatically. It is based on the timer address, x, specified in the item syntax using the timer resolution table stated in the *** note.

***: The timer resolution is based on its address listed in the following table.

Timer Address	Resolution in Milliseconds	Maximum Value in Milliseconds
T0, T32, T64, T96	1	32767
T1-T4, T33-T36, T65-T68, T97-T100	10	327670
T5-T31, T37-T63, T69-T95, T101-T255	100	3276700

For example, if the PLC's T33 timer value is 1000, advising item T33 returns a timer value of 1000, advising item T33 MS returns a value of 10000 milliseconds. Similarly, if T255 timer in the PLC has a value of 2000, advising item T255 MS returns a value of 200000 milliseconds. Poking a value of 5000 to T33 MS writes a value of 500 to the T33 timer in the S7-200 PLC.

Where:

x is the start address, with a range from 0 to 255.

Note: All timers are **Read** and **Write**.

Specific S7-1200 and S7-1500 Item Syntax

The S7-1200 and S7-1500 PLCs are a newer series of Siemens S7 PLCs. Its item syntax follows a similar convention as the S7 300/400 series, with some differences. The S7-1200 and S7-1500 PLCs support the additional datatypes of LREAL and DTL (Date Time Long).

S7-1200 and S7-1500 Data Blocks and Instance Blocks

The following table summarizes the data format, item or point, suffix, data type, and range for Data Blocks and Instance Blocks.

Data Format	Item/Point	Suffix	Data Type	Range
Bit	D<B,I>d,Xx,y		VT_BOOL	0 or 1
String	D<B,I>d,Sx,v		VT_BSTR	String
	D<B,I>d,STRINGx,v		VT_BSTR	String
S7String	D<B,I>d,S7Sx,w		VT_BSTR	String
	D<B,I>d,S7STRINGx,w		VT_BSTR	String
Byte	D<B,I>d,Bx		VT_UI1	0 to 255
	D<B,I>d,BYTEx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
Byte Array	D<B,I>d,Bx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
	D<B,I>d,BYTEx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
USINT	D<B,I>d,USINTx		VT_UI1	0 to 255
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
USINT Array	D<B,I>d,USINTx,v		VT_ARRAY VT_UI1	0 to 255 for each element*
Char	D<B,I>d,CHARx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
Char Array	D<B,I>d,CHARx,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
SINT	D<B,I>d,SINTx		VT_I1	-128 to 127
		DT	VT_BSTR	1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999***
SINT Array	D<B,I>dSINTx,v		VT_ARRAY VT_UI1	-128 to 127 for each element*
Word	D<B,I>d,Wn		VT_UI2	0 to 65535
	D<B,I>d,WORDn		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31

Data Format	Item/Point	Suffix	Data Type	Range
Word Array	D<B,I>d,Wn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
	D<B,I>d,WORDn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
UINT	D<B,I>d,UINTn		VT_UI2	0 to 65535
		BCD	VT_UI2	0 to 9999***
		KT	VT_BSTR	0.0 to 999.3***
		S5T	VT_BSTR	0ms to 2h46m30s0ms***
		TR	VT_R4	0.0 to 9990.0 (s)***
		D	VT_BSTR	1990-1-1 to 2168-12-31
UINT Array	D<B,I>d,UINTn,v		VT_ARRAY VT_UI2	0 to 65535 for each element*
Integer	D<B,I>d,INTn		VT_I2	-32768 to 32767
		BCD	VT_I2	-999 to 999***
		D	VT_BSTR	1990-1-1 to 2168-12-31
Integer Array	D<B,I>d,INTn,v		VT_ARRAY VT_UI1	-32768 to 32767 for each element*
Double Word	D<B,I>d,Dm		VT_UI4	0 to 4294967295**
			VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to
				24D_20H_31M_23S_647MS
Double Word Array	D<B,I>d,Dm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
	D<B,I>d,DWORDm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
UDINT	D<B,I>d,UDINTm		VT_UI4	0 to 4294967295**
		BCD	VT_UI4	0 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to
				24D_20H_31M_23S_647MS
UDINT Array	D<B,I>d,UDINTm,v		VT_ARRAY VT_UI4	0 to 4294967295 for each element*
Double Integer	D<B,I>d,DINTm		VT_I4	-2147483648 to 2147483647
		BCD	VT_I4	-99999999 to 99999999***
		TOD	VT_BSTR	0:00:00.000 to 23:59:59.999
		T	VT_BSTR	-24D_20H_31M_23S_648MS to
				24D_20H_31M_23S_647MS
Double Integer Array	D<B,I>d,DINTm,v		VT_ARRAY VT_I4	-2147483648 to 2147483647 for each element*
Real	D<B,I>d,REALm		VT_R4	+/-1.2e-38 to +/-3.4e+38
Real Array	D<B,I>d,REALm,v		VT_ARRAY VT_R4	+/-1.2e-38 to +/-3.4e+38 for each element*

*: For DDE/SuiteLink, the item value is the HexASCII representation of the complete array. The result is one string containing all the elements of the array in the HexASCII representation of the binary data in big endian format when data is returned to the DDE/SuiteLink layer.

**: For DDE/SuiteLink, this value is restricted to the range of 0 to 2147483647. Values higher than that are clamped to the maximum value of 2147483647 in a SuiteLink or DDE client. In this case, the quality of the item shows "Clamp High."

Where:

- d is the data block number, with a range from 1 to 65535.
- x is the start address. For S7-300/400/1200/1500 PLCs, the range is from 0 to 65535. For S7-200 PLCs, the value is 1.
- y is the bit position, with a range from 0 to 7.
 - 0 is the LSB (Least Significant Bit).
 - 7 is the MSB (Most Significant Bit).
- n is the start address of 2-byte data/2-byte data arrays.
- m is the start address of 4-byte data/4-byte data arrays.
- v is the length of data in elements (an item in an array), with a range from 0 to 65534.
- w is the length of the net S7 string data in characters (size in S7 message is w+1, size of string representation in S7 PLC is w+2).

***: The SIDirect DAServer provides suffixes (S5T, TR, DT, KT, and BCD) that are used to interpret raw bytes as having a special data format. Review the PLC processor documentation, as not all processors have ladder logic internally to interpret those data formats the same way (for example, the S7-1200 PLC).

Note: All data blocks are **Read/Write**. The longest string or array that can be read in a cyclic service has the length of the PDU size minus 32 bytes. The longest string the InTouch software can process is 131 bytes. The longest string that can be poked is 256 bytes or the PDU size minus 28 bytes, whichever is less. The DAServer processes a write (**POKE**) to a Data Block.

For example:

DB5,DTL5 TREAL

LREAL Data Type and Syntax

The S7-1200 and S7-1500 PLCs contain a data type named LREAL. The LREAL data type is a ANSI/IEEE 754-1985 64-Bit Floating Point value with a range illustrated in the following table:

Data Type	Bit Size	Range
LREAL	64	-1.7976931348623158e+308 to -2.2250738585072014e-308, ±0, +2.2250738585072014e-308 to +1.7976931348623158e+308

The S7-1200 and S7-1500 PLCs may not support LREAL types on all memory regions.

The item name syntax for LREAL is as follows:

Memory Type	Syntax
Data Block	D<B,I> <i>d</i> ,LREAL <i>m</i> D<B,I> <i>d</i> ,LREAL <i>m,v</i>
Flag Bytes	FLREAL <i>m</i> MLREAL <i>m</i> FLREAL <i>m,v</i> MLREAL <i>m,v</i>
Input Bytes	ILREAL <i>m</i> ELREAL <i>m</i> ILREAL <i>m,v</i> ELREAL <i>m,v</i>
Output Bytes	OLREAL <i>m</i> ALREAL <i>m</i> QLREAL <i>m</i> OLREAL <i>m,v</i> ALREAL <i>m,v</i> QLREAL <i>m,v</i>

Memory Type	Syntax
Peripheral Input Bytes	PILREAL m
	PELREAL m
	PILREAL m,v
	PELREAL m,v
Peripheral Output Bytes	POLREAL m
	PALREAL m
	PQLREAL m
	POLREAL m,v
	PALREAL m,v
	PQLREAL m,v

Where:

- d is the data block number, with a range from 1 to 65535
- m is the start offset of an LREAL tag
- v is the length of an array in elements, with a range from 1 to 65535

DTL Data Type and Syntax

The S7-1200 and S7-1500 PLCs contain a data type named DTL (Date Time Long). The DTL can be subscribed as a string (VT_BSTR), which provides the full range and resolution of 1 nanosecond, or as a date and time (VT_DATE), which provides the full range but with the highest resolution of 1 second. The DTL data type in the PLC is a 12-byte structure with the following format:

Byte	Field Name	Date Type	Value Range
0,1	Year	UINT	1970 to 2554
2	Month	USINT	1 to 12
3	Day	USINT	1 to 31
4	Weekday	USINT	1(Sunday) to 7(Saturday)
5	Hour	USINT	0 to 23

Byte	Field Name	Date Type	Value Range
6	Minute	USINT	0 to 59
7	Seconds	USINT	0 to 59
8,9,10,11	Nanoseconds	UDINT	0 to 999,999,999

The SIDirect DAServer reads the 12-byte structure from the S7-1200 PLC and converts it into one of the following canonical types based on the item name suffix specified:

DTL Suffix	Canonical Type	Range
<none>	VT_BSTR	1970-1-1-0:0:0.000000000 to 2554-12-31-23:59:59.999999999 (Highest resolution in units of 1 nanoseconds increments)
TREAL	VT_DATE	1970-1-1-0:0:0 to 2554-12-31-23:59:59 (Highest resolution is in units of 1 sec increments) The syntax for VT_DATE can be YYYY/MM/DD, MM/DD/YYYY, or YYYY-MM-DD.

The S7-1200 PLC may not support DTL types on all memory regions.
The item name syntax for DTL is as follows:

Memory Type	Syntax	Canonical Type
Data Block	D<B,I>d,DTLm	VT_BSTR
	D<B,I>d,DTLmTREAL	VT_DATE
	D<B,I>d,DTLm,v	VT_ARRAY VT_DATE
	D<B,I>d,DTLm,vTREAL	VT_ARRAY VT_DATE
Flag Bytes	FDTLm	VT_BSTR
	MDTLm	VT_BSTR
	FDTLmTREAL	VT_DATE
	MDTLmTREAL	VT_DATE
	FDTLm,v	VT_ARRAY VT_BSTR
	MDTLm,v	VT_ARRAY VT_BSTR
	FDTLm,vTREAL	VT_ARRAY VT_DATE
	MDTLm,vTREAL	VT_ARRAY VT_DATE

Memory Type	Syntax	Canonical Type
Input Bytes	IDTL m	VT_BSTR
	EDTL m	VT_BSTR
	IDTL m TREAL	VT_DATE
	EDTL m TREAL	VT_DATE
	IDTL m,v	VT_ARRAY VT_BSTR
	EDTL m,v	VT_ARRAY VT_BSTR
	IDTL m,v TREAL	VT_ARRAY VT_DATE
	EDTL m,v TREAL	VT_ARRAY VT_DATE
	ODTL m	VT_BSTR
	ADTL m	VT_BSTR
	QDTL m	VT_BSTR
Output Bytes	ODTL m TREAL	VT_DATE
	ADTL m TREAL	VT_DATE
	QDTL m TREAL	VT_DATE
	ODTL m,v	VT_ARRAY VT_BSTR
	ADTL m,v	VT_ARRAY VT_BSTR
	QDTL m,v	VT_ARRAY VT_BSTR
	ODTL m,v TREAL	VT_ARRAY VT_DATE
	ADTL m,v TREAL	VT_ARRAY VT_DATE
	QDTL m,v TREAL	VT_ARRAY VT_DATE
	ODTL m	VT_BSTR
	ADTL m	VT_BSTR

Memory Type	Syntax	Canonical Type
Peripheral Input Bytes	PIDTL m	VT_BSTR
	PEDTL m	VT_BSTR
	PIDTL m TREAL	VT_DATE
	PEDTL m TREAL	VT_DATE
	PIDTL m,v	VT_ARRAY VT_BSTR
	PEDTL m,v	VT_ARRAY VT_BSTR
	PIDTL m,v TREAL	VT_ARRAY VT_DATE
	PEDTL m,v TREAL	VT_ARRAY VT_DATE
Peripheral Output Bytes	PODTL m	VT_BSTR
	PADTL m	VT_BSTR
	PQDTL m	VT_BSTR
	PODTL m TREAL	VT_DATE
	PADTL m TREAL	VT_DATE
	PQDTL m TREAL	VT_DATE
	PODTL m,v	VT_ARRAY VT_BSTR
	PADTL m,v	VT_ARRAY VT_BSTR
	PQDTL m,v	VT_ARRAY VT_BSTR
	PODTL m,v TREAL	VT_ARRAY VT_DATE
	PADTL m,v TREAL	VT_ARRAY VT_DATE
	PQDTL m,v TREAL	VT_ARRAY VT_DATE

Where:

- d is the data block number, with a range from 1 to 65535
- m is the start offset of an DTL tag
- v is the length of an array in elements (an item in an array), with a range from 1 to 65535

DTL Clamping

Any DTL value being read from or poked to the PLC that has a year field less than the minimum or greater than the maximum year results in the date and time being clamped as follows:

For Canonical VT_BSTR:

Year < 1970 = 1970-01-01-00:00:00.0000000000

Year > 2554 = 2554-12-31-23:59:59.9999999999

For Canonical VT_DATE:

Year < 1970 = 1970-01-01-00:00:00

Year > 2554 = 2554-12-31-23:59:59

If any other field, month, day, hour, min, second, or nanosecond is less than the minimum or greater than the maximum, it results in a rejected poke or read. Also, an uncertain quality results for subscribed tags.

This applies to the syntax of all DTL names.

Using DTL without a Suffix

When poking or subscribing a DTL without a suffix where the RequestedType=VT_BSTR and the CanonicalType=VT_BSTR and you do not enter the entire 9 digits of the nanosecond field, the field is padded to the right with zeros.

The reason for doing this is to maintain the meaning of the decimal place after the seconds field. For example, 2011-10-31-1:1:1.5 results in 500000000 being poked to the nanoseconds field of the DTL, which really means 1/2 a second. To enter 5 nanoseconds, you must poke 2011-10-31-1:1:1.000000005.

All other fields of the date and time are considered to be padded to the left.

Using DTL with a TREAL Suffix

When poking or subscribing a DTL using the TREAL suffix where the RequestedType=VT_BSTR and the CanonicalType=VT_DATE, the string passed between the SIDirect DAServer and the client is formatted using the "short date" and "long time" format that you configure in the region and language settings of the computer.

High-Speed Counters

S7-1200 and S7-1500 PLCs support High-Speed Counters. After the High-Speed Counters are enabled in the PLC program, they can be accessed through the Input Bytes in DWord format (for example, ID1000, depending on which counter is to be addressed, the configuration of the counter, and the CPU model of the S7-1200 or S7-1500 controller).

For more information about configuring and using the High-Speed Counters, see the S7-1200 and S7-1500 PLC documentation.

Conversions and Suffixes of Items

This section describes what data-format items and suffixes are converted and what they are converted into.

Endian Conversion

In endian conversions, all items with the following data formats are copied in a reverse-byte order to convert the data from the big endian of the PLC to the little endian of the computer:

- Word
- Integer
- Double Word
- Double Integer
- Real

Suffix BCD

All items with the following data formats and suffix BCD are converted from the BCD format into the integer and back:

- Word
- Integer
- Double Word
- Double Integer

Suffix DT

All items with the following data formats and suffix DT (Date and Time) are converted from DT into a message and back to store a value in the range of 1990-1-1-0:00:00.000 to 2089-12-31-23:59:59.999."

- Byte
- Char

This is an 8-byte value (although declared as "byte") that contains both the date and time. In the client, you see a string such as:

1999-12-13-07:06:05.888. The construction is a BCD interpretation.

This means that the value in the memory of the PLC (seen as a hex value) represents directly the single "parts" of the string above.

The example above looks like the following in the memory:

0x9912130706058880

The last character ("0" in this example) is not used in this string, but represents the day of the week. If a DT item is poked, the server writes the correct day of the week to the PLC.

Suffix KT

All items with the following data format and suffix KT are converted from KT to a message and back to store a value in the range of 0.0 to 999.3.

- Word

The item contains a time value in the same format as in the old Step-5 PLCs. In the client, you see a string such as: 999.3. The construction is like a BCD interpretation, but the digits are twisted.

The example above looks like the following in the memory of the PLC:

0x3999

Another example, 0x2345, in the memory of the PLC is 345.2 as the item value.

Suffix S5T

All items with the following data format and suffix S5T are converted from S5T to a message and back to store a value in the range of 0ms to 2h46m30s.

- Word

The memory in the PLC is exactly the same as for the KT items, but the presentation is different although it means the same time. This means a memory content of 0x3999 (as in the first example for KT) results in the string of 2h46m30s0ms.

999.3 (KT) means as follows:

- | | |
|-----|---|
| 999 | The first three characters are the time value in BCD. |
| 3 | The last digit is the multiplier. Possible values are:
0: 0.01s
1: 0.1s
2: 1s
3: 10s. |

This means:

A value of 123.0 represents: $123 * 0.01s = 1.23s$ (equals 1s230ms)

A value of 543.2 represents: $543 * 1s = 543s$ (equals 9m3s0ms)

A value of 999.3 represents: $999 * 10s = 9990s$ (equals 2h46m30s0ms)

Suffix TR

All items with the following data format and with suffix TR (Time as real value) are converted from TR into a real value or back to store a value in the range of 0.0 to 9990.0 (s).

- Word

The memory in the PLC is exactly the same as for the KT items, but the presentation is different, although it means the same time. The memory content of 0x3999 (as in the example for KT) results in the real value of 9990.0. The construction is the result of the multiplication as described in the examples for S5T, given to the client as a real value.

Suffix D

All items with the following data formats and with suffix D (Date) are converted from D into a message or back to store a value in the range of 1990-1-1 to 2168-12-31.

- Word

- Integer

The item contains the date. The construction is the number of days since 1/1/1990. The integer value 4010, for example, represents 2000-12-24.

Suffix T

All items with the following data formats and with suffix T (Time) are converted from T into a message or back to store a value in the range of

–24D_20H_31M_23S_648MS to 24D_20H_31M_23S_647MS.

- Double Word
- Double Integer

The item contains a time in the IEC format. The client shows a value such as: 3D_7H_32M_11S_153MS. This is the time in milliseconds, shown as a more readable string.

The range from 0 to 2147483647 (0x0 to 0x7FFFFFFF) is interpreted as a positive time value. The range from

-2147483648 to -1 (0x80000000 to 0xFFFFFFFF) is interpreted as a negative time value.

Suffix TOD

All items with the following data formats and with suffix TOD (Time Of Day) are converted from TOD into a message or back to store a value in the range of 0:00:00.000 to 23:59:59.999.

- Double Word
- Double Integer

The item contains the time of a day. The client shows a value such as: 4:58:15.654. This is the time in milliseconds (as for T), shown as a more readable string. The highest value is 23:59:59.999. There are no negative values. All values greater than 86399999 (0x05265BFF) are shown with quality 0x0056 (Clamp Hi).

Note: If you use the DAServer in the English operating system, the following applies. The string is always represented in a 24-hour format, regardless of the time representation of the operating system used. This means you see the time 1:13:5 P.M. as 13:13:5.0.

DAServer Standard System Items

System items supply you with easy access to the DAServer status and diagnostic information. They are treated just like ordinary items with respect to the client. However, in most cases these items are not directly acquired via the communications layer. System item values are usually generated through internal calculations, measurements, and tracking of the DAS Engine.

No DAServer-specific system items are provided in this SIDirect DAServer.

System items, like ordinary items, are defined by name in the following context:

- **Group** (client group/OPC group)
Arbitrary collection of items, not correlated.
- **Hierarchical location** (link name/OPC path: Hierarchical node section of the fully qualified OPC item ID)
The device the item is attached to.
- **Device group** (OPC access path/topic, or a Scan Group on a hierarchical branch)
Collection of items on the same physical location with the same protocol update rate.

To check the status of an external device, the reference can be:

```
<PortCpS7 name>.<S7Cp>.$SYS$Status
```

Example:

```
TCPIP.PLC1.$SYS$Status
```

In this example, the scope of the item is not limited to a specific access path/device group. As long as the data requested is from the same external device specified by the same hierarchical location, the value is always the same.

Note: For DDE/SuiteLink clients, \$SYS\$Status always comes from the leaf level of a DAServer hierarchy branch, which is defined by the unique device group. For OPC clients, \$SYS\$Status can be accessed at all hierarchy levels. \$SYS\$Status at the root level of the whole hierarchy tree is always good, as it represents the quality status of the local computer itself. For practical application, OPC clients should reference \$SYS\$Status at hierarchy levels other than the root.

In the ArchestrA context, the device group plays the most important role of identifying the scope of any item. The device group defines the hierarchical location implicitly when using globally unique device group names, which is required for DDE/SuiteLink compatibility.

All system items follow the same naming convention:

- All system items start with \$SYS\$.
- Parsing of the name is case-insensitive.
The DAS Engine scans and parses the name for system items.

All system items can be accessed through subscriptions to a device group. However, while some system items return data for that device group, others are server-wide.

DAServer Global System Item

The following system item refers to specific information regarding a global condition of the DAServer.

System Item Name	Type/ Access Rights	Description	Values
\$SYS\$Licensed	Boolean/Read	The Binary status indication of the existence of a valid license for the DAServer. If FALSE, this item causes the DAServer to stop updating existing tags, to refuse activation of new tags, and to reject write requests as well as setting quality for all items to BAD. If TRUE, the DAServer functions as configured. All instances have the same value.	RANGE: 0, 1 1: Valid license exists. 0: No valid licenses exist.

DAServer Device-Specific System Items

The following system items refer to specific information regarding the device(s) the DAServer is connected to.

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$Status	Boolean/Read	The Binary status indication of the connection state to the device (hierarchy level) the item is attached to. The device group (OPC access path/topic) does not affect the value. The status can be good even if individual items have errors. For DDE/SuiteLink clients, \$SYS\$Status always comes from the leaf level of a DAServer hierarchy branch, which is the destination PLC node. For OPC clients, \$SYS\$Status can be accessed at all hierarchy levels. \$SYS\$Status at the root level of the whole hierarchy tree is always good, as it represents the quality status of the local computer itself. Hence, for practical application, OPC clients should reference \$SYS\$Status at any hierarchy levels other than the root.	RANGE: 0, 1 1: DAServer connection to the device is intact. 0: Error communicating with the device.
\$SYS\$ErrorCode	Longint/Read	Detailed error code of the communications state to the device. The device group (OPC access path/topic) does not affect the value.	>=0: Good status (0 is the default state – connected. >0: is some device state such as, connecting, initializing, and so on. <0: Error status (value indicates the error).

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$errorText	String/Read	Detailed error string of the communications state of the device. The device group (OPC access path/topic) does not affect the value.	Descriptive text for the communications state corresponding to the error code.
\$SYS\$StoreSettings	Integer/ Read/Write	Makes the temporary update interval changes via the \$SYS\$updateInterval item permanent. If the client pokes a value of 1 into this system item, the currently set update interval is written to the servers configuration file. The value of this system item clears to 0 after being set if the configuration file write is successful. If the write fails, then the value is set to -1. If the update interval is changed via the \$SYS\$updateInterval item and this item is not poked to 1, the DAServer uses the original update interval for that topic the next time it is started. Reading the item always provides 0. Read/Write values are persisted only if you set this system item. The values other than this persist only for the life of the DAServer.	RANGE: -1, 0, 1 -1: Error occurred during saving the configuration file. 0: Always Read value if status is OK. 1: Persist settings (cleared immediately).

DAServer Device-Group-Specific System Items

The following system items refer to specific information regarding device groups that are configured in the DAServer.

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$updateInterval	Dword/ Read/Write	Accesses the currently set update interval. It is the current update interval of the device group in milliseconds. A client can poke new values into this item. The value of zero indicates that no non-system items on that topic are updated (data for these items are not acquired from the device).	RANGE: 0...2147483647 0: Topic inactive, no items are updated. Data acquisition is stopped. >0: Expected updated interval for the set of all items in the device group.
\$SYS\$maxInterval	Dword/Read	Accesses the currently measured maximum update interval in milliseconds of all items of the corresponding device group. This item is read-only. The value of the slowest item is shown.	RANGE: 0...2147483647 0: If update interval is 0 or if the status is false. >0: Measured update interval.

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$WriteComplete	Integer/ Read/Write	Accesses the state of pending write activities on the corresponding device group. On device group creation (adding items to an OPC group) the value of this system item is initially 1, indicating all write activities are complete – no pokes are pending. If values are poked into any items of the device group, the value of this item changes to 0, indicating write activity is currently in progress. If the server completes all write activities, the value of this item changes to 1 if all pokes were successful or to -1 if at least one poke failed. If the value of this item is not zero, the client can poke 1 or -1 to it (poke a 1 to clear errors, or a -1 to test a client reaction on write errors). If the value of this item is zero, it cannot be poked.	RANGE: -1, 0, 1 1: Write complete (no writes are pending – initial state). 0: Writes are pending. -1: Writes completed with errors.
\$SYS\$ReadComplete	Integer/ Read/Write	Accesses the state of reads on all items in the corresponding device group. The value is unequal 0 if all active items in a device group are read. If at least one item in the device group is activated, this item changes to 0. It changes to 1 if all newly activated items are read successfully or to -1 if at least one item has a non-good quality. Poking a 0 to this item resets the internal-read states of all items in this device group. This resets this item to 0. If all items are read again after this poke, this item changes back to 1 or -1.	RANGE: -1, 0, 1 1: Read complete (all values have been read). 0: Not all values have been read. -1: All values have been read but some have a non-good quality.

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$ItemCount	Dword/Read	Accesses the number of items in the corresponding device group. This item is read-only.	RANGE: 0...2147483647 ≥0: Number of active and inactive items.
\$SYS\$ActiveItemCount	Dword/Read	Accesses the number of active items in the corresponding device group. This item is read-only.	RANGE: 0...2147483647 ≥0: Number of active items.
\$SYS\$ErrorCount	Dword/Read	Accesses the number of all items (active and inactive) with errors (non-good OPC quality) in the corresponding topic. If the communications status of a device group is bad, all items have errors. This item is read-only.	RANGE: 0...2147483647 ≥0: Number of all items (active and inactive) with errors.

System Item Name (Type)	Type/ Access Rights	Description	Values
\$SYS\$PollNow	Boolean/ Read/Write	Poking a 1 to this item forces all items in the corresponding device group to be read immediately (all messages in this device group become due). This is useful if you want to force getting the newest values from the device, regardless of its update interval. This also works on device groups with a zero update interval (manual protocol triggering).	RANGE: 0, 1
\$SYS\$MasterController	Int/Read	This system item indicates whether a particular S7-400H PLC is a master or slave. Applicable only for S7-400H controllers. This information also is used internally to trigger a failover in the DAS redundant pair. You do not need to subscribe to the \$SYS\$MasterController item for failovers to take place.	1 = master 0 = slave -1 = communicating with a non-400H PLC

DAServer Redundant Device Specific System Items

These system items are specific to the Redundant Device.

System Item Name	Type/Access Rights	Description	Values
\$SYS\$ForceFailover	Boolean/ ReadWrite	This is required to achieve the failover condition to be forced by client. Note: By poking a value of "1" (True) into the Force Failover item, a client can conveniently switch to the secondary device. The ForceFailover command has no effect when the RDO is communicating with a redundant pair of S7-400H controllers. An accompanying Logger message is shown indicating the reason the ForceFailover did not trigger.	TRUE, FALSE
\$SYS\$ActiveDevice	String/Read	This system item will show the current runtime active device.	Node Hierarchy Name
\$SYS\$FailoverTime	Time/Read	This system item will show the time at which the switch occurred.	Time at which the switch occurred
\$SYS\$StandbyDevice	String/Read	This system item will show the current runtime standby device.	Node Hierarchy Name

System Item Name	Type/Access Rights	Description	Values
\$SYS\$Secondary DeviceStatus	Boolean/Read	This system item will show the status of the secondary device. This is the status of the second device defined in the configuration and is not changed with any failover. RANGE: 0, 1	RANGE: 0, 1 (Contains the value of the system item \$SYS\$Status)
\$SYS\$PrimaryDevice Status	Boolean/Read	This system item will show the status of the primary device. This is the status of the first device defined in the configuration and is not changed with any failover. RANGE: 0, 1	RANGE: 0, 1 (Contains the value of the system item \$SYS\$Status)
\$SYS\$FailoverReason	String/Read	This system item will show the reason for the failover.	Descriptive text “ForceFailover” or the value of the system item \$SYS\$ErrorText.

Important: The Redundant Hierarchy, including the Device Group, is not hot-configurable, and requires a Reset on the Redundant Hierarchy to effect a configuration change.

DAServer Global System Item

The following system item refers to specific information regarding a global condition of the DAServer.

System Item Name	Type/ Access Rights	Description	Values
\$SYS\$Licensed	Boolean/Read	The Binary status indication of the existence of a valid license for the DAServer. If FALSE, this item causes the DAServer to stop updating existing tags, to refuse activation of new tags, and to reject write requests as well as setting quality for all items to BAD. If TRUE, the DAServer functions as configured. All instances have the same value.	RANGE: 0, 1 1: Valid license exists. 0: No valid licenses exist.

Generic OPC Syntax

A DAServer is a container for OPC Groups, providing the mechanism for containing and logically organizing OPC items. Within each OPC Group, an OPC-compliant client can register OPC items, which represent connections to devices in the field device. All access to OPC items is maintained through the OPC Group.

The fully qualified name for an OPC item is the Item ID, equivalent to Item Name. The syntax for specifying a unique Item ID is DAServer-dependent. In OPC data acquisition DAServers, the syntax can be as follows:

```
TCPIP.PLC1.DB1,B20
```

Where each component (delimited by a period) represents a branch or leaf of the field device's hierarchy.

In this example:

- PLC1 is the name of the target PLC.
- DB1,B20 is the specific data point or item desired.

An item is typically a single value such as an analog, digital, or string value, where:

- Item ID describes the syntax for defining the data point.
- OPC provides another parameter, called Access Path, that defines optional specifications for obtaining that data.

In DAServers, Access Paths are equivalent to Device Groups. This parameter defines the update interval between the DAServer and the field device for accessing the values of data points in the PLC.

VT_Array Syntax in Application Server

VT_Array items from SIDirect can fail to return good VTQ data in Object Viewer without the addition of an index to the Item/Point syntax. While the syntax provided in this section is correct, VT_Arrays require a [-1] offset. Without this offset, the VT_Array will not count the 0 byte in the range.

Following is an example of the syntax for a VT_Array item to receive proper validation under a LMX client such as Object Viewer in Application Server:

```
S7C_S7_001.SG1.attribute(DB111,WORD10,4)[-1]
```

In this example, the syntax has the following structure:

```
<SIDirect_DiDevice_Instance>.<ScanGroup>.attribute(D<B>d,WORDn,v)[-1]
```

where

- d is the data block number,
- n is the start address of a 2-byte data array, and
- v is the length of data in elements (the size of the array). Each item in an array is called an element.

Chapter 8

Troubleshooting

This section describes the troubleshooting tools that you can use to deal with SIDirect DAServer problems.

The DAServer Manager provides access to diagnostics and other statistical data. The Log Viewer provides access to event messages logged during the operation of the SIDirect DAServer. Your client, for example, the InTouch software, can also monitor connectivity with the PLC through the \$SYS\$Status item. Use these tools together with the information in this chapter to troubleshoot your SIDirect DAServer.

Finding the DAServer Version Number

This section describes finding the version number of your DAServer.

To find the version number

- 1 From the **Start** menu, point to **Settings**, and click on the **Control Panel** option.
- 2 Click **Add or Remove Programs**
- 3 Find and click on **Wonderware SIDirect DAServer** from the **Currently installed programs** list and click the hyperlink, **Click here for support information**. The release version of the DAServer appears in the **Support Info** dialog box.

OR

- ◆ Click on the DAServer node in the hierarchy-tree view. In the **Details** pane on the right you see the build version numbers of the respective DAServer components.

OR

- 1** Search for **DASSIDirect.dll**.
- 2** Right-click on the **File Name** and select **Properties** on the menu.
- 3** Click the **Version** tab on the **Properties** dialog box. The version of your DASServer is listed under **File Version**.

Monitoring Connectivity Status with the PLC

You can use the built-in discrete item, `$SYS$Status`, to monitor the status of communications with the PLC. This item is set to:

- 0 (zero) when communications with the PLC fails.
- 1 (one) when communications is successful.

For DDE/SuiteLink clients, `$SYS$Status` always comes from the leaf level of a DASServer hierarchy branch, which is the destination PLC node. For OPC clients, `$SYS$Status` can be accessed at all hierarchy levels. `$SYS$Status` at the root level of the whole hierarchy tree is always good, as it represents the quality status of the local computer itself. For practical application, OPC clients should reference `$SYS$Status` at any hierarchy levels other than the root.

Enter the following DDE reference formula in the appropriate place in your client:

=DASSIDirect|S7PLC!\$SYS\$Status

where:

DASSIDirect	is the name of the DASServer application.
S7PLC	is the exact device group defined in the DASServer for the PLC.
\$SYS\$Status	is the discrete item that monitors the status of connectivity with the PLC.

Enter the following OPC item reference syntax when adding the item in your OPC client:

YourOPCAccessPath.\$SYS\$Status

where:

YourOPCAccessPath	is the assembly of hierarchy node names leading to a specific device (controller).
\$SYS\$Status	is the discrete item used to monitor the status of connectivity with the device (controller).

Monitoring the Status of DAS Conversations

The **InTouch WindowsViewer** supports built-in topic names, **DDEStatus** and **IOStatus**, that can monitor the status of specific DAS conversations.

For example, assume that **WindowViewer (VIEW)** is communicating with the **SIDirect DAServer** to a PLC. The PLC is defined in the InTouch application with the access name **S7PLC**. The discrete items, **DDEStatus** and **IOStatus**, are set to 0 when this DAS conversation failed, and to 1 when this DAS conversation is successful.

Using DDEStatus and IOStatus in Excel

The status of communications between the PLC and InTouch software can be read into Excel by entering the following DDE reference formula in a cell on a spreadsheet:

=view | DDEStatus!S7PLC

or

=view | IOStatus!S7PLC

where:

view	is the name of the InTouch application.
[DDE][IO] Status	is the built-in topic name that monitors the status of communications between the DAServer and the InTouch software.
S7PLC	is the exact access name defined in the InTouch application for the PLC.

Reading Values from the DAServer into Excel

Values can be read directly into Excel spreadsheets from the DAServer by entering a DDE formula into a cell using the following format:

=applicationname | <devicegroup>!itemname

Example formula:

=DASSIDirect | S7PLC!'DB1,B20'

where:

DASSIDirect	is the name of the DAServer application.
S7PLC	is the exact device group name defined in the DAServer for the PLC.
DB1,B20	is the actual location in the PLC that contains the data value. This is the item name.

In this example, each time the value of <DB1,B20> changes in the PLC, the DAServer automatically sends the new value to the cell containing the formula in Excel.

Note: See the Microsoft Excel documentation for complete details on entering Remote Reference formulas for cells.

Writing Values to the DAServer from Excel

You can write values from Microsoft Excel to the DAServer by creating an Excel macro that uses the **POKE** command.

The proper command is entered in Excel as follows:

```
channel=INITIATE("applicationname","topicname")
=POKE(channel,"itemname", Data_Reference)
=TERMINATE (channel)
=RETURN()
```

The following describes each of the above **POKE** macro statements:

```
channel=INITIATE("applicationname","topicname")
```

- Opens a channel to a specific topic name that is defined in the DAServer in a particular application name (the executable name without the .exe).
- Assigns the number of that opened channel to **channel**.

Note: When using the **channel=INITIATE** statement, the word **channel** must be used in the **=POKE** statement instead of the actual cell reference. The "**application name**" and "**topic name**" portions of the formula must be enclosed in quotation marks.

```
=POKE(channel,"itemname", Data_Reference)
```

- Pokes the value contained in the **Data_Reference** to the specified item name or the actual location in the PLC, via the **channel** number that is returned by the previously executed **INITIATE** function.
- **Data_Reference** is the row/column ID of the cell containing the data value.

```
=TERMINATE(channel)
```

- Closes the channel at the end of the macro.
- **Channel** is the channel number returned by the previously executed **INITIATE** function.
- Some applications have a limited number of channels, therefore they should be closed when finished.

=RETURN()

- Marks the end of the macro.

Note: See the **.xlm** sample Excel poke macro provided on the DAServer CD. See the Microsoft Excel documentation for complete details on entering Remote Reference formulas for cells.

Debugging Communications Between the SIDirect DAServer and the PLC

The DAServer Manager lets you use on-line diagnostics of the SIDirect DAServer components at run-time, locally and remotely.

To perform on-line diagnostics

- ◆ Select any active SIDirect DAServer on any node in the DAServer Manager.

The Diagnostics branch is visible only if the DAServer is active as indicated by the green icon on the server branch. It contains the following sub-branches:

- Client Groups
- Structure
- Transactions
- Statistics
- Messages
- Device Groups

Each of these sub-branches contains live information from the DAServer. They allow detailed diagnostics of objects within the SIDirect DAServer.

Note: If you have pokes that are folded, the diagnostics shows ALL items in the transaction. However, because they are folded, only items that have actually been sent have both the message ID and value. All other items that have not been sent, because of the folding, are listed in this transaction with the same timestamp but without the message and value.

Client Groups

The information provided in the Client Groups diagnostic root is organized like all DAServers. See the DAServer Manager user's guide for detailed explanation of the generic diagnostic information of DAServers.

In the Diagnostics Client Groups branch, the OPC client groups are listed as created by the OPC clients. The DDE/SL plug-in always creates only one client group called DDESLPlugIn.

The list view shows the following information:

Name	The name of the client group.
Items	The number of created items in the client group.
Active Items	The number of active items in the client group.
Errors	The number of active and inactive items with errors or the OPC quality is not good in the client group.
Update Interval	The client group update interval.
State	The state of the client group, Active or Inactive.

Clicking an item in the list view shows the full diagnostic information for this item.

For example:

Name:	DDESLPlugIn
Items:	10234
Active Items:	10234
Items with Errors:	0
Update Interval:	0

Selecting a client group in the tree view shows all items in this client group. The item information is grouped into the following columns:

Name	The OPC leaf item name without the fully qualified OPC item ID path.
Client Value	The last updated value to the client.
Client Time	The timestamp the client was updated last.
Client Quality	The item quality of last update.
Subscr Msg	The DAS identifier for the subscription message of the item.
Location	The path item name, which is the OPC path part of the fully qualified item ID.
Device Group	The name of the device group the item is in.

Clicking an item in the list view shows the full diagnostic information of this item.

For example:

Name:	MB9
Client Value:	129
Client Time:	Monday, July 29, 2002
Quality:	GOOD: Nonspecific: ...
Location:	New_PortCpS7_000 New_ ...
Subscription Message:	1002809
Device Group:	PLC1

Structure

The structure view shows the physical hierarchical organization of the DAServer. In the Diagnostics Structure branch, the structure view shows generic DAServer information as well as SIDirect-DAServer-specific information. See the DAServer Manager user's guide for detailed explanation of the generic diagnostic information of DAServers.

The list view shows either the branches, items, or both, populating different columns. The branches populate the following columns:

Name	The branch name.
Items	The number of active items.
Errors	The number of items with errors.
R/W Status	The Read/Write status of the corresponding branch. R: Read complete All item values are acquired. W: Write complete All write operations are completed.
Messages	The number of messages and messages on this branch.
Device Groups	The number of device groups on this branch.

The SIDirect DAServer provides the following additional S7-specific columns for branches:

S7 Error:	The S7 error code.
A&E:	The Alarms and Events setting. Valid values are OFF, ALARMS ON, and EVENTS ON.

The items populate the following columns:

Name	The item leaf name.
R/W Status	R: Item was updated. W: No writes are pending on the item.
Value	The last protocol value.

Time	The last protocol time.
Quality	The last protocol quality.

No S7-specific columns are populated for items.

Transactions

The transactions view shows currently pending transactions and completed transactions, depending on the backlog setting. It also shows generic DAServer information as well as SIDirect-DAServer-specific information. Refer to the DAServer Manager user's guide for detailed explanation of the generic diagnostic information pertaining to the DAServers.

In the Diagnostics Transactions branch, the transactions populate the following columns:

Type	The Demand Read and Demand Write icons showing the state of the transaction.
ID	The unique identifier for the transaction.
Items	The number of items.
Status	The status of the transaction.
Start	The time the transaction starts.
End	The time the transaction ends. The column is empty if the transaction is not complete.

Expanding a transaction lists all items with the same columns as in Structure.

Statistics

The statistics view shows current statistics of a DAServer. This generic information is not specific to the SIDirect DAServer. See the DAServer Manager user's guide for detailed explanation of the generic diagnostic information pertaining to DAServers.

Messages

The message view shows generic and S7-specific information of the SIDirect DAServer. See the DAServer Manager user's guide for detailed explanation of the generic diagnostic information of DAServers.

In the Diagnostics Messages branch, the message view shows all messages in the DAS Engine.

MsgID	The message unique identifier.
Items	The number of items.

Errors The number of items with non-good quality.

Status The status of the message.

The following is the SIDirect-DAServer-specific diagnostic information:

S7 Msg ID The S7 message ID consisting of one of the following letters:

P: poll message

C: cyclic service

B: block message

W: write message

S: scan message

A: alarm message and a number

PDU size <req msg>/<data resp>

Where <data resp> is the size of the data in the response message and <req msg> is the size of the request message.

Message error The S7-message communications error code.

Clicking a message in the list view shows the full diagnostic information of this message.

For example:

Name: 0100280A

Active Items: 462

Error Items: 0

Status: IDLE

S7 Msg ID: P0

PDU Size (resp/req): 480/24

Msg Error: OK

Expanding a message lists all items with the same columns as in Structure.

Name The item leaf name.

R/W Status R means the item was updated. W means no writes are pending on the item.

Value The last updated value from the protocol.

Time The last updated time from the protocol.

Quality The last updated quality from the protocol.

Msg ID The DAServer internal message identifier.

Location The OPC path of fully qualified item ID.

The following is the SIDirect-DAServer-specific diagnostic information:

S7 Msg ID	The S7 message ID consisting of one of the following letters: P: poll message C: cyclic service B: block message W: write message S: scan message A: alarm message and a number
Item error	The S7-item communications error code where 255 = OK.

Clicking an item in the list view shows the full diagnostic information of this item.

For example:

Name:	New_PortCpS7_000.New_...
Read status:	complete
Write status:	complete
Value:	127
Type:	VT_U11 - unsigned c...
Time:	Monday, July 29, 2002...
Quality:	GOOD: Nonspecific:...
S7 Msg ID:	P2
Item Error:	OK

Device Groups

The device groups view shows all device groups in the DAS Engine.

Device Group	The device group name.
Update Interval	The protocol update interval in milliseconds.
Items	The number of items.
Active Items	The number of active items.
Errors	The number of items with errors.
Location	The OPC path of the fully qualified Item IDs of items in this device group.
Poll Msgs	The number of poll messages.
Cyclic Srvs	The number of cyclic services.

Clicking a device group in the list view shows the full diagnostic information of this device group.

For example:

Name:	PLC1
Updated Interval:	1000
Number of Items:	10234
Located at:	New_PortCpS7_000.New_...
Poll Messages:	23
Cyclic Services:	0

Expanding a device group in the tree view shows all items in this group. The diagnostic information on items in a device group is identical to the items in Messages.

Diagnostics and Error Tracing

The SIDirect DAServer uses the standard diagnostic information provided by the DAS Toolkit. Access to other internal diagnostic registers of the PLC is performed through reads and writes via the syntax used in Item Naming.

Diagnostics Facility

This section describes the areas in which the SIDirect DAServer diagnostics can help you.

Communications Processor

The diagnostics window of the CP (Communications Processor) shows all DAS-provided diagnostic columns plus a column for any error code on the CP level. The diagnostic information shows DAS information text plus the error text description.

S7 Communications Processor

The diagnostics window of the CP shows all DAS-provided diagnostic columns plus a column for any error code on the PLC CP level. The diagnostic information shows the DAS information text plus the error code description.

Items

To the DAS-provided diagnostics of items, the following two columns are added:

- **The message ID**
The message ID consists of two parts:
 - A letter indicating the type of message.
For example, P-poll, C-cyclic, B-block, and so on.
 - A message number.
For example, order ID, block ID, event ID, and so on.
- **The S7 error code**
The diagnostic information shows the message ID. For example, order ID, block ID, or event ID, and the error code description.

Messages

Four types of messages having different diagnostic appear.

Variable Service Message with:

- P – poll message
- W – poke message
- C – cyclic message
- B – block message

In addition to the standard DAS diagnostic messages, the following information is added:

- Message ID
- Two PDU sizes indicating block size if applicable and request block size
- The message or S7 error code

The diagnostic information shows the DAS information text plus the message ID, PDU size, and error text description.

Device Group

The device group diagnostics window shows the DAS-provided standard diagnostic columns plus the following four additional columns:

- Number of poll messages
- Number of cyclic services
- Number of block messages
- Number of alarm & event messages

The diagnostic information shows the corresponding additional values.

DASTrace Diagnostic Messages

The SIDirect DAServer generates messages that you can use for diagnostic purposes. These DASTrace diagnostic messages do not necessarily indicate that error conditions exist.

The following table is not a complete list of messages that are shown in the Logger when DASTrace is enabled. Other diagnostic messages are shown, as well.

Note: The logger messages use the following codes: %s to represent strings, %d to represent numbers (integer), %x to represent the address of the object, and %ums to represent time in milliseconds.

Logger Message	Explanation	Probable Cause	Solution
Message is in SlowPollMode (%s msg=0x%08X) for topic %s	The message state is set to Slow poll interval.	The response for this message has not been received by the server. Either the response time is very small or the PLC is very busy.	Adjust the message reply timeout in the server in accordance with the performance of the PLC to achieve the optimal behavior.
Message leaves the SlowPollMode (%s msg=0x%08X) for topic %s	The message state is coming out of the Slow poll interval.	The server recovered from the error situation.	N/A
UpdateInterval for Device Group: %s changed to %d	The update interval for the device group has been changed. The messages are now to be polled at a new update interval for this device group.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
Setting all items to bad in hierarchy: %s	The server updates the status of all the items as Bad on this connection.	The connection with the PLC is broken.	Check the connection with the PLC.
Connection with '%s' dropped while pending, forcing immediate reconnect	For the S7-200 only. The socket connection with the PLC fails due to a given error code: "The PLC initiated termination of the Socket Connection."	The PLC is not responding to the Connect request from the DASServer. The maximum number of connections on the configured RTSAP may have been exceeded.	DASSIDirect will attempt to automatically reconnect. No user action is required.
Connection with '%s' failed twice while pending, going to slow poll	For the S7-200 only. The socket connection with the PLC fails due to a given error code: "The PLC initiated termination of the Socket Connection."	The PLC is not responding to the Connect request from the DASServer. The maximum number of connections on the configured RTSAP have been exceeded.	Configure the PLC to accept connections on a different RTSAP or turn off all other clients accessing the same PLC.
Connection with '%s' dropped, restarting connection	The socket connection with the PLC fails due to a given error code: "The PLC initiated termination of the Socket Connection."	The PLC is not responding to the Connect request from the DASServer.	DASSIDirect will attempt to automatically reconnect. No user action is required.
Connection with '%s' dropped while pending, going to slow poll	The socket connection with the PLC fails due to a given error code: "The PLC initiated termination of the Socket Connection."	The PLC has been disconnected and/or has faulted.	Check the PLC connection and clear any outstanding PLC faults.
Block receive size: doesn't match msg size	The server receives an invalid block packet from the PLC.	This message from the PLC is garbled.	Check the PLC health status and the network condition.

Error Tracing with the Wonderware Logger

The SIDirect DAServer supports error messages, board-specific error messages, trace logger messages, and error codes. Use the Log Flag data to customize the type of messages logged to the Log Viewer.

Note: See the Log Viewer online documentation for more information about using log flags.

SIDirect DAServer Logger Flags

The SIDirect DAServer supports the following server-specific DAS logger flags.

- **Errors**
General errors from the server have the prefix "ERROR." All related errors, such as for poll messages, have the prefix "POLL_ERROR." They include CONNECTION_ERROR, POLL_ERROR, POKE_ERROR, CYCLIC_ERROR, BLOCK_ERROR, and ALARMS_AND_EVENTS_ERROR.
- **Trace**
General traces from the server have the prefix "TRACE." All related traces, such as for poll messages, have the prefix "POLL_TRACE." They include CONNECTION_TRACE, POLL_TRACE, POKE_TRACE, CYCLIC_TRACE, BLOCK_TRACE, and ALARMS_AND_EVENTS_TRACE.

The following table lists all the available logger flags and their meanings.

Type	Name	Description
Server Flag	ERROR	Shows general server errors.
	TRACE	Shows general server traces.

Type	Name	Description
Transaction Flag	CONNECTION_	Shows connection errors.
	ERROR	
	POLL_ERROR	Shows errors of poll messages.
	POKE_ERROR	Shows errors of poke messages.
	CYCLIC_ERROR	Shows errors of cyclic messages.
	BLOCK_ERROR	Shows errors of block messages.
	ALARMS_AND_	Shows errors of alarm and scan messages.
	EVENTS_ERROR	
	CONNECTION_	Shows connection traces.
	TRACE	
	POLL_TRACE	Shows traces of poll messages.
	POKE_TRACE	Shows traces of poke messages.
	CYCLIC_TRACE	Shows traces of cyclic messages.
	BLOCK_TRACE	Shows traces of block messages.
	ALARMS_AND_	Shows traces of alarm and scan messages.
	EVENTS_TRACE	

Error Messages, Trace Messages, Error Codes, and Warnings

In addition to the SIDirect DASServer error and warning messages, S7 Trace messages and generic DASServer error codes are supported. Use these messages together with the DASServer Manager Diagnostic root data to troubleshoot SIDirect DASServer problems.

Note: The logger messages use the following codes: %s to represent strings, %d to represent numbers (integer), %x to represent the address of the object, and %ums to represent time in milliseconds.

You can also use the Log Flag data to customize the type of messages logged to the Log Viewer. For more information about using log flags, see the Log Viewer online Help.

DAServer Error Messages

The following list shows error messages produced by the DAServer that are logged to the Log Viewer with the DASProtFail log flags.

Logger Message	Explanation	Probable Cause	Solution
DASProtFail Messages			
Internal Error: CioVariant::Update()	The internal type conversion encounters an invalid or unknown type.	This is an internal program error.	Turn on POLL_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Send fail because of wrong order id	The message cannot be sent to the PLC because of the order ID is incorrect.	This is an internal program error.	Turn on POLL_TRACE, POKE_TRACE, and CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, report it to Wonderware Technical Support.
(2):s7_multiple_read_req (orderid=%d) [(%d) %s]x	The poll message cannot be sent to the PLC.	This is an internal program error.	Turn on POLL_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, report it to Wonderware Technical support.
Internal state error: multiple read response (S7Type 0x%X)	The server receives a duplicate response for the poll message.	This is an unknown error.	Turn on POLL_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, check with the PLC vendor.

Logger Message	Explanation	Probable Cause	Solution
(%d):Invalid item name: %s (%s)	The requested item name has a bad syntax.	The item syntax is wrong.	Correct the item syntax as defined in this user's guide.
(%d):Invalid item name suffix: %s (%s)	The requested item suffix has a bad syntax.	The item suffix is wrong.	Correct the item suffix as defined in this user's guide.
Alarm event header key not: FF09: %02X%02X	The alarm received from the PLC has a bad header.	This is a PLC issue.	Turn on ALARMS_AND_EVENTS_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Send: (MSG_FAIL): message (%s msg=0x%08X) [msg_state=%d,con_state=%d]	The server fails to send the message to the PLC.	This is an internal error.	Turn on POLL_TRACE, POKE_TRACE, and CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
(%d): s7_brcv_init (r_id=%d) [0]	There is an error in s7_brcv_init (Block services).	A configuration or communications error occurred.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Timeout for initial values of block with r_id=%d	The block message has timed out for the initial updates.	The PLC does not send the block update.	Check the PLC program and see the B_SEND is configured correctly.
Timeout updating values of block with r_id=%d	Timeout occurs while updating the subsequent values for the block message.	The connection with the PLC may be dropped.	Check the PLC program and see the B_SEND is configured correctly.

Logger Message	Explanation	Probable Cause	Solution
Release blockid:%d for message (%s msg=0x%08X,con=%s) [number of blockids=%d] was not successful	Releasing the block ID from the block message operation fails.	This is an internal error.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. If the error persists, report it to Wonderware Technical Support.
S7BlockMessage::HandleResponse: premature block end received at size %d (expected: %d)	The message received from the PLC for the block service has no data.	This is an unknown error.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
S7BlockMessage::HandleResponse: unsegmented block messages did not have correct size: %d (len: %d)	The unsegmented message received from the PLC for the block service has incorrect data size.	This is an unknown error.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
S7BlockMessage::HandleResponse: segmented messages did not add up correctly in size: %d to indicated response data (len: %d)	The segmented message received from the PLC for the block service has incorrect data size.	This is an unknown error.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
S7BlockMessage::HandleResponse: cannot allocate memory for response data (len: %d)	The server cannot allocate the memory for the response received from the PLC for the block message.	The server runs out of memory.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Restart the server or the computer.

Logger Message	Explanation	Probable Cause	Solution
S7BlockMessage::HandleResponse: too much block data received: (received: %d + new: %d more than expected: %d)	The server receives too much data for the block message.	This is a PLC problem.	Turn on BLOCK_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Internal Error: Unknown type: %d updating event item: %s	The server receives the updates for the unknown type of alarm and event message.	This is an unknown error.	Turn on ALARMS_AND_EVENTS_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Could not generate data for item %s	The server cannot read the poke value for the item.	This is an internal error.	Turn on POKE_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Can't send request for msg=%.08X (no orderid available)	The server cannot send the message to the PLC because it runs out of order IDs.	This is an unknown error.	Turn on POLL_TRACE, POKE_TRACE, and CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.

Logger Message	Explanation	Probable Cause	Solution
Send fail because of wrong order id	The server cannot send the message to the PLC because the order ID is incorrect.	This is an unknown error.	Turn on POLL_TRACE, POKE_TRACE, and CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
(%d):s7_multiple_write_req (orderid=%d) [0]	The write request to the PLC fails.	The data poke may be too long or this is an internal error.	Turn on POKE_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Internal state error: multiple write response (S7Type 0x%X)	The server receives a duplicate response for the poke message.	This is an unknown error.	Turn on POKE_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, report the error to Wonderware Technical Support.
(%d):s7_cycl_read (orderid=%d) [0]	An error occurs when sending the S7 cyclic read message.	A configuration or communications error occurred.	Check the PLC configuration/connection and resource limitation, particularly the ones related to the cyclic service.
(%d):s7_cycl_read_init_req (orderid=%d) [0]	An error occurs when sending the S7 cyclic read initiation request message.	A configuration or communications error occurred.	Check the PLC configuration/connection and resource limitation, particularly the ones related to the cyclic service.

Logger Message	Explanation	Probable Cause	Solution
(%d):s7_cycl_read_start_req (orderid=%d): [0]	An error occurs when sending the S7 cyclic read start request message.	A configuration or communications error occurred.	Check the PLC configuration or connection.
(%d):s7_cycl_read_delete_req (orderid=%d): [0]	An error occurs when sending the S7 cyclic read delete message.	A configuration or communications error occurred.	Check the PLC configuration or connection.
ERROR in S7_CYCL_READ_INIT_CNF - PLC responded with error code	The PLC returns an error code in response to the cyclic read initiation request.	It is a PLC, DAServer configuration, or resource error.	Turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. Check the PLC configuration/connection and resource limitation, particularly the ones related to the cyclic service.
ERROR in S7_CYCL_READ_START_CNF - PLC responded with error code	The PLC returns an error code in response to the cyclic read start request.	It is a PLC or DAServer configuration error.	Turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, contact Wonderware Technical Support.
ERROR in S7_CYCL_READ_DELETE_CNF - PLC responded with error code	The PLC returns an error code in response to the cyclic read delete request.	It is a PLC or DAServer configuration error.	Turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, contact Wonderware Technical Support.

Logger Message	Explanation	Probable Cause	Solution
Internal state error: cyclic read response (S7Type 0x%X)	The server receives a duplicate response for the cyclic message.	This is an unknown error.	Turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional trace information. If the problem persists, contact Wonderware Technical Support.
Scan event header key not: 120A: %02X%02X	When parsing the packet returned by the Scan event packet from the PLC, the header key is incorrect.	This is a programming error.	Turn on ALARMS_AND_EVENTS_TRACE in the Wonderware Logger to obtain additional trace information. Repeat the test and/or restart the DAServer/PLC. If the error persists, contact the PLC vendor.
Item doesn't fit in a single message and won't be advised	The server cannot fit this item into a message, therefore it cannot advise this item.	The item byte range is larger than the PLC PDU size.	Split the item into smaller items so that they can be fitted into the available PDU size. The S7-300 family of controllers can process PDU lengths of 480 to 960 bytes, depending on the controller. An S7-400 controller can process PDU lengths of 960 bytes.

Logger Message	Explanation	Probable Cause	Solution
Can't create poke message for item %s, data size is too large	The server cannot fit this item into a poke message, therefore it cannot poke this item.	The item byte range is larger than the PLC PDU size.	Split the item into smaller items so that they can be fitted into the available PDU size. The S7-300 family of controllers can process PDU lengths of 480 to 960 bytes, depending on the controller. An S7-400 controller can process PDU lengths of 960 bytes.
Can't create poke message for item %s, not able to generate data	The poke data for creating a poke message cannot be generated.	The poke data value cannot be converted into the PLC datatype.	Check the value being poked and create the correct format.
%s Leaving Slow Poll Mode	This is only a piece of information about the server leaving the Slow Poll mode.	The connectivity to the PLC recovered from a failure. Normal communications is resumed.	If this message shows up consistently, verify the network connectivity to the PLC.
%s Entering Slow Poll Mode	This is only a piece of information about the server entering the Slow Poll mode.	The connectivity to the PLC failed. The DAServer tries to reconnect at the Slow Poll interval.	Verify the network connectivity to the PLC. Turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.
TIMEOUT for pending initiate request	A timeout occurs while waiting for an initiate confirmation.	A configuration or communications error occurred.	Check the communications or configuration. If the problem persists, turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.

Logger Message	Explanation	Probable Cause	Solution
(%d):s7_initiate_req [0]	There is an error in initiating a request (establishing a connection).	A configuration or communications error occurred.	Check the connection and the PLC configuration/program. If the problem persists, turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.
Can't connect	There is an error establishing a connection.	A configuration or communications error occurred.	Check the connection and the PLC configuration/program. If the problem persists, turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.
Received order id on a deleted non read/write (type: 0x%04X) message: 0x%x	The server receives a response from the PLC for a message that has already been deleted from the server.	Unexpected order ID is received from the PLC. The DAServer discards the message associated with it.	If the problem persists, contact the PLC vendor.
Internal HEXASCII conversion buffer overflow	The server cannot allocate memory for the HEXASCII conversion.	The server may be out of memory.	There could be a memory contention issue on the computer. Check the memory allocations for all processes on the computer.

Logger Message	Explanation	Probable Cause	Solution
Generate data failed for item %s because at least one element of the array is not filled	The server cannot poke the array item as the items are not filled correctly.	You poke the array items but some of the element are left unfilled. All elements in the array are rejected for poking.	Fill all elements in the array before poking the array.
TIMEOUT for connection (while %s), m_state=%d	A timeout occurs while waiting for a response message to <command>.	A communications/ configuration error occurred.	Verify the network connectivity to the PLC. Turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.
(%d):s7_msg_initiate_req (orderid=%d) for %s [(%d) %s]	There is an error in the message initiate request (initiating alarms and events).	A configuration or communications error occurred.	Verify the network connectivity to the PLC. Turn on CONNECTION_TRACE, DASSend, and DASReceive in the Wonderware Logger to obtain additional diagnostic information.
ERROR: order ID %d exists for cyclic ID: %d in cyclic reference map	The server tries to add the order ID for the cyclic service that has already been occupied by some other message.	This is an internal error.	If the problem persists, turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional diagnostic information.
ERROR: Retrieving order ID: %d from cyclic ID: %d in cyclic reference map	The server cannot find the order ID in the order ID map for this message.	This is an internal error.	If the problem persists, turn on CYCLIC_TRACE in the Wonderware Logger to obtain additional diagnostic information.

Logger Message	Explanation	Probable Cause	Solution
ERROR: UNKNOWN cyclic ID (%d) in cyclic read indication	The server receives an unknown cyclic message form the PLC.	The PLC reports a cyclic response that is not requested by the DAServer.	Check with the PLC vendor.
%s Connection aborted	The connection to the PLC is closed.	Either the PLC closed the connection or the server closed the connection.	If this is caused by the normal shutdown or items removal, no actions are required. If not, verify the PLC configuration for the Keep-Alive parameter.

S7 Trace Messages

The SIDirect DAServer provides five types of trace messages as follows:

- Connection Trace
- Poll Trace
- Cyclic Trace
- Block Trace
- Alarms and Events Trace

The following table lists the trace messages produced by the DAServer. For more information about trace messages, see "SIDirect DAserver Logger Flags" on page 151.

Logger Message	Explanation	Probable Cause	Solution
CONNECTION_TRACE Messages			
(%d):s7_initiate_req [0]	There is an error establishing a connection. The server cannot send the Connect Request to the PLC. The first parameter is the error code of the function call (-1 = Message Blocked, -2 = Message Failed) from the PLC. The parameters inside the square bracket is 0.	The credit is not available to send this request.	If after some time the problem still exists, restart the server.
Can't connect	There is an error establishing a connection. The server cannot send the Connect Request to the PLC.	The credit is not available to send this request.	If after some time the problem still exists, restart the server.
Close connection (con=%s): (cpd=%d,cref=%d)	INFO: The DAserver closes the connection with the PLC. The parameters associated with the PLC connection are listed.	N/A	N/A
Connection (con=%s) was not successful	The server connection attempts to the PLC fails.	The PLC is in a faulty condition or the connection is broken.	Check the PLC or check the cable connection.
Connection (con=%s) was successful	INFO: The server connects with the PLC successfully.	N/A	N/A
Internal Error: %s Set state of connection to %d	The connection with the PLC cannot be established.	This is an unknown error.	Check the PLC or cable connection, or restart the server.

Logger Message	Explanation	Probable Cause	Solution
Open connection (con=%s)	INFO: The socket connect call to the PLC is successful but the connection negotiation is still not completed	N/A	N/A
s7_abort = OK	INFO: The SCDirect DServer closes the connection with the PLC successfully.	The connection is closed by either the server or the PLC	N/A
s7_get_initiate_cnf cnf amq called: %d	INFO: PLC agrees on this parameter with the DServer. (cnf_amq_called)	N/A	N/A
s7_get_initiate_cnf cnf amq calling: %d	INFO: PLC agrees on this parameter with the DServer. (cnf_amq_calling)	N/A	N/A
s7_get_initiate_cnf cnf pdu size: %d	INFO: PLC agrees on this parameter with the DServer. (cnf_pdu_size). The PDU size is returned.	N/A	N/A
s7_get_initiate_cnf ind amq called: %d	INFO: This is the DServer side negotiation parameter with the PLC. (ind_amq_called)	N/A	N/A
s7_get_initiate_cnf ind amq calling: %d	INFO: This is the DServer side negotiation parameter with the PLC. (ind_amq_calling).	N/A	N/A
s7_get_initiate_cnf ind pdu size: %d	INFO: This is the DServer side negotiation parameter with the PLC. (ind_pdu_size) The PDU size of the PLC is requested.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
s7_initiate_req = OK	The server sends the Connect Request to the PLC successfully.	N/A	N/A
%s Set state of connection to CLOSED	The SIDirect DAServer closes the connection to the PLC.	The connection is closed by the server or by the PLC.	If the cable is not disconnected, check the PLC configuration and see if you have specified the Keep-Alive parameter. The Keep-Alive parameter causes the closing of connection if there are no activities for some specified amount of time.
%s Set state of connection to ERROR	The connection with the PLC cannot be established.	The server cannot connect to the PLC at all.	Check the PLC or cable connection, or restart the server.
%s Set state of connection to OPEN	INFO: The server opens a connection with the PLC.	N/A	N/A
%s Set state of connection to PENDING	INFO: The server is awaiting the response from the PLC for the Connection Request.	N/A	N/A
%s Set state of connection to SLOW POLL MODE	The server is going into the Slow Poll mode.	The connection with the PLC is bad.	Check the PLC or check the cable connection.
POLL_TRACE Messages			
%s dumping s7 objects (containing %d items)	INFO: Dumping the information about the S7Info in the logger.	N/A	N/A
(1):s7_multiple_read_req (orderid=%d) [0]	INFO: The multiple read request message cannot be sent to the PLC because the server runs out of credit.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
(2):s7_multiple_read_req (orderid=%d) [0]	The multiple read request message cannot be sent to the PLC because of unknown reasons.	N/A	If the problem persists, restart the server.
add pItem (%s) to S7Info (pS7Info=0x%08X, addr: %d range: %d)	INFO: The server adds item to the S7Info while building the Poll message.	N/A	N/A
Build message (%s msg=0x%08X) for topic %s	INFO: The server builds a POLL message for the topic.	N/A	N/A
Delete message (%s msg=0x%08X) for topic %s	INFO: Destructor for the POLL message gets called.	N/A	N/A
dumping read values	INFO: Dumping the read values received from the PLC in the logger.	N/A	N/A
Got orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: The server assigns the order ID for the POLL message.	N/A	N/A
Release orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: Release the order ID from the message.	N/A	N/A
remove pItem (%s) from S7Info (pInfo=0x%08X)	INFO: The server removes the item from the S7Info.	N/A	N/A
s7_multiple_read_req = OK (orderid=%d)	INFO: The multiple read request message is sent to the PLC successfully.	N/A	N/A
S7Info=%s:result=%d var_length=%d value (in HEXASCII):	INFO: The server logs the information about the read value received from the PLC in the logger.	N/A	N/A
POKE_TRACE Messages			

Logger Message	Explanation	Probable Cause	Solution
%s dumping s7 objects (containing %d items)	INFO: Dumping the information about the S7Infos in the logger.	N/A	N/A
%s: result=%d var_length=%d value (in HEXASCII)	INFO: Logging the Result received from the PLC for the write request.	N/A	N/A
(1):s7_multiple_write_req (orderid=%d) [(%d) %s]	INFO: The multiple write request message cannot be sent to the PLC because running out of credit.	N/A	N/A
(2):s7_multiple_write_req (orderid=%d) [(%d) %s]	The multiple write request message cannot be sent to the PLC because of unknown reasons.	N/A	If problem persists, restart the Server or check the PLC.
add pItem (%s) to S7Info (pS7Info=0x%08X, addr: %d range: %d)	INFO: Server adding item to the S7Info while building the Poke Message.	N/A	N/A
Build message (%s msg=0x%08X) for topic %s	INFO: Server building a Poke message for the topic.	N/A	N/A
Delete message (%s msg=0x%08X) for topic %s	INFO: Destructor for the Poke message gets called.	N/A	N/A
Got orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: Server assigns the order id for the Poke Message.	N/A	N/A
Release orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: Release the order ID from the poke message.	N/A	N/A
remove pItem (%s) from S7Info (pInfo=0x%08X)	INFO: Server removing the item from the S7Info.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
s7_multiple_write_req = OK (orderid=%d)	INFO: The multiple write request message sent to the PLC successfully.	N/A	N/A
S7PokeMessage::AddPokeItem (Item=%s)	INFO: Server adding item to the Poke Message.	N/A	N/A
TREAL item %s high clamped to 9990s	INFO: The value of TREAL item is clamped at high value.	N/A	N/A
TREAL item %s low clamped to 0ms	INFO: The value of TREAL item is clamped at low value.	N/A	N/A
CYCLIC_TRACE Messages			
%s dumping s7 objects (containing %d items)	INFO: Dumping the information about the S7Infos in the logger.	N/A	N/A
(-1):s7_cycl_read (orderid=%d) [(%d) %s]	The server could not send the cyclic read request to the PLC. Temporary error.	N/A	N/A
(1):s7_cycl_read_delete_req (orderid=%d): [0]	The server could not send the cyclic read delete request to the PLC because of running out of credit.	N/A	N/A
(1):s7_cycl_read_start_req (orderid=%d): [0]	The server could not send the cyclic read start request to the PLC because of running out of credit.	N/A	N/A
(2):s7_cycl_read_delete_req (orderid=%d): [0]	The server could not send the cyclic read delete request to the PLC because of Unknown reason.	N/A	N/A
add pItem (%s) to S7Info (pS7Info=0x%08X, addr: %d range: %d)	INFO: Server adding item to the S7Info while building the cyclic Message.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
Build message (%s msg=0x%08X) for topic %s	INFO: Server building a cyclic message for the topic.	N/A	N/A
Delete message (%s msg=0x%08X) for topic %s	INFO: Destructor for the cyclic message gets called.	N/A	N/A
Got orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: Server assigns the order id for the cyclic Message.	N/A	N/A
Release orderid:%d for message (%s msg=0x%08X,con=%s) [number of orderids=%d]	INFO: Release the order ID from the cyclic message.	N/A	N/A
remove pItem (%s) from S7Info (pInfo=0x%08X)	INFO: Server removing the item from the S7Info.	N/A	N/A
s7_cycl_read = OK (orderid=%d)	INFO: The server successfully sends the cyclic read request to the PLC.	N/A	N/A
s7_cycl_read_delete_req = OK (orderid=%d)	INFO: The server successfully sends the cyclic read delete request to the PLC.	N/A	N/A
s7_cycl_read_start_req = OK (orderid=%d)	INFO: The server successfully sends the cyclic read start request to the PLC.	N/A	N/A
s7_get_cycl_read_init_cnf (cpd=%d,m_cref=%d,orderid=%d) Build Poll Messages: [(%d) %s]	INFO: The server gets the confirmation from the PLC for the Cyclic read request.	N/A	N/A
S7Info=%s:result=%d var_length=%d value (in HEXASCII)	INFO: Logging the value received from the PLC for the cyclic message	N/A	N/A
BLOCK_TRACE Messages			

Logger Message	Explanation	Probable Cause	Solution
(0): s7_brcv_init (r_id=%d) [0]	INFO: The block receive initiation request is successful.	N/A	N/A
add item (%s) to S7BlockMessage (%s msg=0x%08X)	INFO: Adding items to the block message.	N/A	N/A
Build message (%s msg=0x%08X) for topic %s	INFO: Server building a Block message for the topic.	N/A	N/A
Delete message (%s msg=0x%08X) for topic %s	INFO: Destructor for the block message gets called.	N/A	N/A
Release blockid:%d for message (%s msg=0x%08X,con=%s) [number of blockids=%d]	INFO: Release block id for the block message.	N/A	N/A
remove item (%s) from S7BlockMessage (%s msg=0x%08X)	INFO: Remove item from the block message.	N/A	N/A
s7_brcv_init = OK (r_id=%d)	INFO: The block receive initiation request is successful.	N/A	N/A
s7_brcv_stop = OK (r_id=%d)	INFO: The block receive stop request is successful.	N/A	N/A
ALARMS_AND_EVENTS_TRACE Messages			
%s: ack_state=%d	INFO: Update the alarm or event with the acknowledgement state information.	N/A	N/A
%s: event_state=%d	INFO: Update the alarm or event with the event state information.	N/A	N/A
%s: no_add_value=%d	INFO: Update the alarm or event with the number of additional value information.	N/A	N/A

Logger Message	Explanation	Probable Cause	Solution
%s: state=%d	INFO: Update the alarm or event with the state information.	N/A	N/A
%s: time_stamp=%s	INFO: Update the alarm or event with the time stamp information.	N/A	N/A
(%d):s7_msg_abort_req (orderid=%d) for %s [0]	The alarm abort request to the PLC failed.	N/A	N/A
add item (%s) to S7Event (msg=0x%08X)	INFO: Add item to the event message.	N/A	N/A
Build alarm object (0x%08X) for connection %s	INFO: Server builds the alarm object that are going to receive the alarms and events.	N/A	N/A
Build scan object (0x%08X) for connection %s	INFO: Server builds the scan object.	N/A	N/A
Delete alarm object (0x%08X) for connection %s	INFO: The destructor of the alarm object gets called.	N/A	N/A
Release eventid:%d (con=%s) [number of eventids=%d]	INFO: Release order id from the event message.	N/A	N/A
remove item (%s) from S7Event (0x%08X)	INFO: Remove item from the event message.	N/A	N/A
s7_msg_abort_req = OK (orderid=%d) for %s	INFO: The server sends the alarm abort request to the PLC successfully.	N/A	N/A
s7_msg_initiate_req = OK (orderid=%d) for %s	INFO: The server sends the alarm registration request to the PLC successfully.	N/A	N/A

DAServer Error Codes

The following table lists the Wonderware DAServer error codes and the error messages that appear with the codes, and their descriptions.

Code	Error Message	Description
C004D000L	Invalid item name	The requested item name has bad syntax.
C004D001L	Item name not exist	The requested item name has good syntax, but it does not exist.
C004D002L	Device not connect	The device is not connected, so data cannot be acquired.
C004D100L	Device off scan	The device is communicating, but it cannot accept queries for data items.
C004D101L	Timeout	A message transaction with the device timed out.

DAServer Protocol Warnings

The following table lists protocol warnings generated by the DAServer. The log flag for these messages is DASProtWarn.

Logger Message	Explanation	Probable Cause	Solution
DASProtWarn Messages			
Send: exit (MSG_OK): Attempt to send POLL message while pending (%s msg=0x%08X) [msg_state=%d,con_state=%d]	The server tries to send the Poll message while it waits for the response from the PLC for the same message.	The response from the PLC is slow.	This is a flow control issue. The server is too fast to send the message but the PLC is slow in responding to those messages. Try to reduce the load from the PLC by disconnecting other clients from the PLC or reducing the scan rate of the message. If the problem exists call the PLC vendor.
Update item (%s, quality=0x%04X) on %s	The server updates the item with Bad quality. This message shows up only when the item quality is Bad.	N/A	Check the OPC quality for the appropriate error message.

Logger Message	Explanation	Probable Cause	Solution
S7Info (%s pS7Info=0x%08X/pM sg=0x%08X) returned error: %s	Logging the error code returned by the PLC for the read request.	There is an item access error in the PLC.	Check the PLC configuration and see whether the memory area you try to access exists in the PLC with proper access right.
Could not generate data for item %s	The server cannot read the poke value.	This is an internal error.	Turn on POKE_TRACE in the Wonderware Logger to obtain additional trace information. Report the error to Wonderware Technical Support.
Send: exit (MSG_OK): Attempt to send POKE message while pending (%s msg=0x%08X) [msg_state=%d,con_s tate=%d]	The server tries to send the poke message while it waits for the response from the PLC for the same message.	The response from the PLC is slow.	This is a flow control issue. The server is too fast to send the message but the PLC is slow in responding to those messages. Try to reduce the load from the PLC by disconnecting other clients from the PLC or reducing the scan rate of the message. If the problem exists call the PLC vendor.
S7 Topic's <%s> property <%s> was changed to <%s>	The server is not hot-configurable for the given property.	The server does not use the changed value.	Re-start the server to see this change in effect.
Invalid value, clamp at high limit for poking item: %s on %s	Poke data is clamped into a valid range before it is sent to the PLC.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
Clamping S5T poke data for %s on %s (client poke %ums clamped to 9990000)	S5T poke value is clamped to 9990000 before it is sent to the PLC.	Poke value exceeds the valid range.	See the DAServer user's guide for the correct range of values.
Loosing precision on converting S5T poke data for %s on %s (client poke %ums converted to 0ms)	Non-zero S5T poke value is converted to 0.	Poke value is below the S7 S5T type resolution.	See the DAServer user's guide for the correct range of values.
Loosing precision on converting S5T poke data for %s on %s (client poke %ums converted to %ums)	Precision is lost on value.	The resolution of the S7 data type does not match the poked value.	See the DAServer user's guide for the correct range of values.

Logger Message	Explanation	Probable Cause	Solution
Invalid poke value, clamp at low limit for item: %s on %s	Poke data is clamped into a valid range before it is sent to the PLC.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
Invalid poke value, cannot convert value for item: <item name> on <device group>	Poke data is set to constant.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
ERROR: Invalid value, clamp at high limit for poking item: %s on %s	Poke data is clamped into a valid range.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
ERROR: Invalid value, clamp at low limit for poking item: %s on %s	Poke data is clamped into a valid range.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
ERROR: Invalid value, cannot convert for poking item: %s on %s	Poke data is set to constant.	Poke value exceeds the S7 data type range.	See the DAServer user's guide for the correct range of values.
Write complete fails - item: %s on %s	The server cannot write the value of the item to the PLC.	Connection to the PLC is bad or Item access is denied by the PLC.	Check the PLC connection or configuration.
S7Cp's <%s> property <%s> was changed to <%s>	The server is not hot-configurable for the given property.	The server does not use the changed value.	Restart the server to see this change in effect.
(%d):s7_msg_initiate_req (orderid=%d) for %s [(0)]	There is an error in the message initiate request (initiating alarms and events).	It is a communications error or a PLC configuration error.	Check the connection and the PLC configuration/program.
S7CP_200's <%s> property <%s> was changed to <%s>	The server is not hot-configurable for the given property.	The server does not use the changed value.	Restart the server to see this change in effect.

Data Conversion

The following table describes how the SIDirect DAServer handles values that cannot be converted or do not meet the limit specifications.

Conversion	Description
NONSPECIFIC	If a value cannot be converted, the quality of the item goes to NONSPECIFIC.
Uncertain-HIGHLIMITED	If a value is greater than the upper limit, the quality of the item goes to uncertain-HIGHLIMITED.
Uncertain-LOWLIMITED	If a value is minor than the lower limit, the quality of the item goes to uncertain-LOWLIMITED.

Quality Settings

The SIDirect DAServer uses the general OPC-defined quality settings. An item can have six basic data quality states.

Quality Code	Quality State	Description
00C0	Data quality good	Data communications is good and data is good. The register is read or written to without any problems converting the data.
0055	Clamp low	Data communications is good but the data is uncertain. The data is clamped at a low limit. The register is correctly read or written to, but it is necessary to clamp its value to a limit. The value is smaller than the minimum allowed.
0056	Clamp high	Data communications is good but the data is uncertain. The data is clamped at a high limit. The register is correctly read or written to, but it is necessary to clamp its value to a limit. The value is larger than the maximum allowed. A string is truncated. For example, a floating point value is clamped to FLT_MAX.

Quality Code	Quality State	Description
0040	Quality uncertain/No convert	Data communications is good but the data is uncertain.
		The data cannot be converted.
		The server may return either a constant in place of the data or return quality information alone.
		The data is useable. However, it is not known whether the value is too large or too small.
		Incorrect data type.
		Floating point is not a number.
		For example, 0x000a in a PLC BCD register.
0004	Bad configure/No access	This is a configuration error.
		Data communications is good but the data cannot be sent and/or received. The data is bad and cannot be used.
		Item cannot be accessed.
		The item does not exist or is not available.
		The server can communicate with the PLC but cannot access the register.
		The server determined the point is not valid.
		The PLC responds that the register does not exist, cannot be read, or cannot be written to.
		The server cannot access a fenced, write-protected, or read-only item.
		The PLC is in a mode that does not permit access to this item.
		The number of data bytes is incorrect but the message is otherwise good.
		The command or op code is invalid but the message is otherwise good.
		The PLC is busy. The server has given up retrying.

Quality Code	Quality State	Description
0018	No communications	Data communications is down.
		Cannot access the PLC due to a communications error.
		Data is bad and cannot be used.
		The device group is in a slow poll or equivalent mode.
		The PLC does not exist and/or is not responding.
		There is no link validating the message.
		There is a lack of resources in the server. A TSR or driver cannot allocate memory.
		There is a lack of resources in the communications link.
		The communications link is off-line.
		All communications channels are in use.
		The network cannot route the message to the PLC.

Chapter 9

Reference

This section describes the architecture of the DAServer, its collection of components, and the environments in which the components work.

DAServer Architecture

The SIDirect DAServer is supported on the Windows operating systems that are specified in the Readme file. The NetDDE protocol is not supported by DAServers.

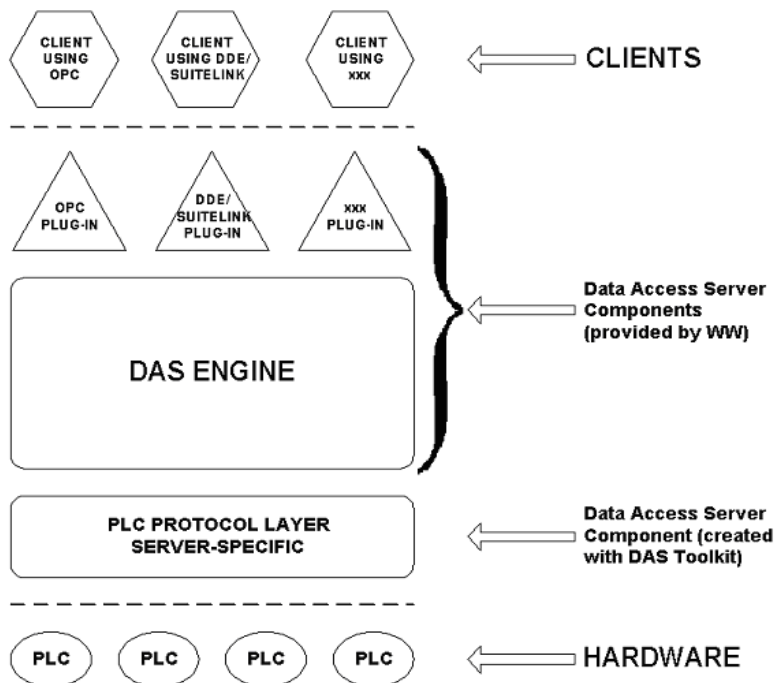
This DAServer is a collection of components that work together to provide communications access to the hardware field devices. These components include:

- **DAServer Manager**
The Microsoft Management Console (MMC) snap-in environment, supplied with the DAServer, that provides the necessary user-interface for diagnostics, configuration, and activation.
- **Client Plug-ins**
The components that are added to a DAServer to enable communications with clients.
Examples are OPC, DDE/Suitelink, and so on.
- **DAS Engine**
The library that contains all the common logic to drive data access.
- **Device Protocol**
The custom code provided by the DAServer to define the communications with a particular device.

DAServers

A DAServer consists of the following three physical parts:

- **Plug-in Component(s)**
Component responsible for communicating with clients.
- **DAS Engine**
Common component used by all DAServers.
- **PLC Protocol Layer, DAServer-specific**
Component responsible for communicating with the hardware.



DAServer Architecture

Each physical part of a DAServer consists of a set of .exe and/or .dll modules. Wonderware provides the plug-ins and the DAS Engine. Using the DAS Toolkit, you create the PLC Protocol Layer (DAServer-specific) modules. All three sets of modules are required for a fully functioning DAServer.

Plug-ins

Plug-ins provide a protocol translation function for device integration clients. Typical plug-ins communicate in DDE, SuiteLink, or OPC protocol, and serve as interfaces between their clients and the DAS Engine.

Note: Items in an array are not supported in the DDE/SL plug-in. Arrays are converted to HEXASCII strings, which provide legacy behavior for DAServers that support this.

DAS Engine

The DAS Engine is a middleware component that exposes two sets of interfaces, one for communicating with the plug-ins and the other one for communicating with the PLC Protocol Layer components.

PLC Protocol Layer

The PLC Protocol Layer provides a protocol translation function for specific hardware such as ModBus, and serves as an interface between the DAS Engine and the hardware.

Component Environments

Stand-alone DAServers have the following characteristics:

- The DAS Engine is dynamically linked to the other DAServer components. A new DAS Engine such as feature enhancements or bug fixes do not require relinking to the other components nor re-QA of those other components. When deployed to the system, the new DAS Engine attaches to all existing DAServer components.
- Newly deployed plug-ins such as feature enhancements or bug fixes do not require relinking nor re-QA of associated components. Even new plug-ins (for example, OPC Alarm & Events) do not require any development changes to the other components, and require no relinking in a customer-installed base. In fact, you can implement new functionality in a plug-in to enhance the DAServer without involving the code of the other components.
- DAServers can be configured in one stand-alone configuration utility (DAServer Manager) capable of showing specific configuration pages for all DAServers. This utility allows the browsing and editing of DAServers on different nodes.
- The DAServer Manager diagnostics tool shows generic diagnostic objects common to all DAServers, as well as DAServer-specific/DAServer-developer-defined diagnostic data.

The DAServer data configuration format is XML. Any XML-enabled program such as, XML Editor can read this format.

Appendix A

Tested DAServer Hardware and Firmware

You are encouraged to use hardware/firmware that has been tested; however, testing of every possible combination is not feasible. Often the PLC manufacturer uses different model numbers when there are functionally minor differences, such as increased memory or faster CPU. Though we may not have tested these specific models, models within the same PLC series are supported. For example, if we have tested the S7-1211C, then the S7-1200 family is supported. For Siemens hardware/firmware, here is the general guideline for support:

- 1** Tested hardware is supported.
- 2** Other hardware with the same series number as the test hardware is supported.
- 3** Supported hardware series with latest firmware version from Siemens is supported, unless noted otherwise.

The following table lists the supported hardware and firmware tested with the SIDirect DAServer.

Supported Device Series	Description	Tested Hardware and Firmware
S7-200	Controller	CPU214 214-1AD22-0XB0 Firmware Version: V1.22
S7-200	Ethernet Module	CP243-1 TCP 243-1EX00-0XE0 Hardware Version: 1 Firmware Version: V2.00
		CP243-1 TCP 243-1EX01-0XE0 Firmware Version: V1.02
S7-300	Controller	CPU314C-2 PN/DP 314-6EH04 -0AB0 Hardware Version 1 Firmware Version: V3.3.10
		CPU315-2 DP 315-2AG10-0AB0 Hardware Version: 1 Firmware Version: V2.6.11
S7-300	Controller with Built-in Ethernet	CP343-1 343-1EX30-0XE0 Hardware Version: 1 Firmware Version: V2.3.2
		CP343-1 343-1EX11-0XE0 Hardware Version: 2 Firmware Version: V2.3.5
S7-300F	Controller with Built-in Ethernet	CPU315F 315-2FJ14-0AB0 Hardware Version: 5 Firmware Version: V3.2.10
		Controller with Built-in Ethernet CPU319F 318-3FL01-0AB0 Hardware Version: 8 Firmware Version: V3.2.10

Supported Device Series	Description	Tested Hardware and Firmware
S7-300	Ethernet Module	CP343-1 343-1EX30-0XE0
		Hardware Version: 1
		Firmware Version: V2.3.2
		CP343-1 343-1EX11-0XE0
		Hardware Version: 2
		Firmware Version: V2.3.5
S7-400	Controller	CPU414-3 DP 414-3XJ00-0AB0
		Hardware Version: 7
		Firmware Version: V3.1.4
		CPU414-4H 414-4HM14-0AB0
		Hardware Version: 1
		Firmware Version: V4.5.6
S7-400	Controller with Built-in Ethernet	CPU412-2 PN 412-2EK06-0AB0
		Hardware Version: 1
		Firmware Version: V6.0.2
S7-400	Ethernet Module	CP443-1 443-1EX11-0XE0
		Hardware Version: 4
		Firmware Version: V2.7.3
		CP443-1 443-1EX20-0XE0
		Hardware Version: 1.2.0
		Firmware Version: V2.0.49

Supported Device Series	Description	Tested Hardware and Firmware
S7-1200	Controller with Built-in Ethernet	CPU1211C 211-1AD30-0XB0 Firmware Version: 2.2.0
		CPU1214C 214-1AD30-0XB0 Firmware Version: 2.2.0
S7-1500	Controller with Built-in Ethernet	CPU1513 513-1AL00-0AB0 Hardware Version: 2 Firmware Version: V1.5.0
		CPU1516 516-3AN00-0AB0 Hardware Version: 2 Firmware Version: V1.5.0

Index

A

- access path 135
- Active Device 31, 32, 132
- active items 142, 146
- adding a PLC object 26, 29
- adding a port object 25
- alarms and events 28, 94, 143
- alarms and events terms 100
- aliases 43
- analog input 104
- analog output 105
- application name 59
- ArchestrA 124
- ArchestrA System Management Console 21
- ArchestrA.DASSIDIRECT.x 19, 50, 53
- ArchestrA.DASSIDirect.x 22, 58
- ArchestrA.SIDIRECT.x 19
- archiving configuration sets 50, 54
- arrays 64, 81, 113, 180
- auto 41

B

- backlog setting 144
- BCD format 120

- BCD interpretation 121

- before you begin 12

- block

- messages 41

- read 41

- services 41

- block items 89

- read-only 89

- write-only 89, 92

- block services 41

C

- checklist

- setting up the SIDIRECT DAServer 18

- clearing

- all device items 44

- client plug-ins 179

- client quality 142

- client time 142

- client value 142

- command

- clear all 45

- import 46

- communication protocols 13

- communications protocol server 7

- configuration
 - branch 25
- configuration view 26, 29
- configuring
 - default update intervals 41
 - the DAServer 21
- counters 86
- creating or adding
 - device groups 38
 - device items 43
- CSV file 44
- cyclic services 40, 146

D

- DAS Engine 179, 180, 181
- DAS logger flags 151
- DAS Server Manager 21
- DAS Toolkit 147, 180
- DAServer
 - activating/deactivating the DAServer 53
 - configuring as service 50
 - managing your DAServer 49
 - setting up your DAServer 17
- DAServer architecture 13
- DAServer Manager 22, 59, 179
 - diagnostics tool 181
 - documentation 22, 23
 - tree view 23
- DASSIDirect 59
- DASSIDirect.dll 138
- Data Access Server 11
- data blocks 61, 111
- data points 135
- data-type identifiers 61
- DCOM 13
- DDE 8, 13, 14
- DDE/SuiteLink 49, 50, 53
- DDE/SuiteLink clients 138
- DDESLPlugIn client group 142
- DDEStatus 139
- default group 22
- deleting
 - device groups 38
 - device items 44
- demo mode 54
- device group 58, 124, 142, 146
- Device groups. 54

- device groups 23, 39, 135, 143
- device groups view 146
- device hierarchy 21, 23
- device item definitions 43
- Device items 54
- Device nodes 54
- device protocol 179
- device redundancy 31
- diagnostics 13, 141, 143, 144
- diagnostics messages 144
- DTL data type 115
- Dynamic Data Exchange 14

E

- edit mode 25
- editing
 - update intervals 41
- end 144
- endian conversion 120
- errors 142, 143, 145, 146
- Excel 139
- Excel macro 140
- exporting
 - DAServer item data to a CSV file 44

F

- FastDDE 8, 13, 14
- FASTPOLL 60
- features 13
- flag bytes 65

G

- Global Configuration parameters 54
- global parameters 23
- group 124
- group name 58

H

- HEXASCII strings 180
- hierarchical location 124
- hierarchy 24
- high-speed counter 109
- high-speed counters 120
- HMI 13
- host 58
- Hot configurable 54

I

- ID 144
- importing
 - DAServer item data from a CSV file 45
- initial values 41
- in-proc/out-of-proc 53
- input and output block 40
- input bytes 69
 - peripheral 77
- instance blocks 61, 111
- intervals
 - update 38
- IOStatus 139
- item
 - error 146
 - ID 134
 - name 60, 134
- item data 44
- item name 59
- ItemID 57
- items 142, 143, 144, 146

L

- license 54
- License Manager 55
- link name 59
- local node 22
- location 142, 145, 146
- Log Flag data 151, 152
- Log Viewer 137, 152
- Logger 54
- LREAL data type 114

M

- memory
 - special 107
 - stage 106
- message
 - view 144
- message error 145
- messages 143
 - cyclic 40
 - poll 40
- Microsoft Management Console 21
- middleware component 181
- MMC 21
- ModBus 181

- Msg ID 145

- msgID 144

N

- name 142, 143, 145
- NetDDE 13
- network address 27, 30
- network transport protocol 14
- node name 58, 59

O

- object name 31
- off-line edit on item data 44
- OLE/COM technology 13
- OPC 13
 - clients 138
 - item browsing 43
 - item name 57
 - protocol 58
- OPC client 8, 43
- optimization 41
- optimization mode 41
- output bytes 73
 - peripheral 82

P

- PDU size 145
- peripheral
 - access 40
 - input bytes 77
 - output bytes 82
- Ping item 32, 35
- PLC hierarchy node 43
- PLC Hierarchyname.csv file 44
- PLC protocol layer 180, 181
- plug-in component 180
- poke mode 39
- poke mode settings 40
- poll messages 146
- port type 25
- PortCpS7 object
 - adding 25
- predefined item syntax 43
- preparing
 - the SIDIrect DAServer 22
- Primary Device 33, 133
- processor type 27, 29

ProgID 58

program name 58

Q

quality 144, 145

quality settings 176

R

rate

 update 58

read contiguous IO 40

read/write 58

read/write status 143, 145

Redundant 31, 32, 33, 34, 35, 132

remote access 58

remote communications 13

remote reference formulas 141

renaming

 device items 44

reverse-byte order 120

runtime behavior 31

S

S7 cyclic services 40

 disabling 40

S7 error 143

S7 hierarchy 21

S7 msg ID 145, 146

S7 SAPI 41

S7 tag creator 35

SCADA 13

scan group 124

Secondary Device 33, 132, 133

service

 automatic 24

 manual 24

 unconfirmed 41

setup.exe 22

SIDIRECT DASServer

 configuring as not a service 50

SIDirect DASServer 7

SMC

 finding your DASServer 19

snap-in environment 23

special memory 107

stage memory 106

start 144

state 142

statistics view 144

status 144, 145

status of connectivity 138

structure 145

structure branch 143

structure view 143

subscription message 142

suffix

 BCD 120

 D 122

 DT 121

 KT 121

 S5T 121

 T 123

 TOD 123

 TR 122

SuiteLink 13

SuiteLink client 8

system item

 DASServer global 125

system items

 \$SYS\$ActiveDevice 132

 \$SYS\$ErrorText 133

 \$SYS\$FailoverReason 133

 \$SYS\$FailoverTime 132

 \$SYS\$ForceFailover 132

 \$SYS\$MasterController 32, 131

 \$SYS\$PrimaryDeviceStatus 133

 \$SYS\$SecondaryDeviceStatus 133

 \$SYS\$StandbyDevice 132

 \$SYS\$Status 31, 35, 133

 DASServer device-group-specific 128

 DASServer device-specific 126

 DASServer global 125, 134

System Management Console 22

T

time 144, 145

time supervision 41

timeout

 connection 28, 30

 initial values 41

 reply 28, 30

 update 41

timers 88, 110

topic name 60

transactions 144

transactions view 144

transition mode 40

TSAP

 local 27, 30

 remote 27, 30

type 144

U

update

 intervals 38

 rate 58

update interval 41, 58, 142, 146

 parameter 47

V

V Memory 101

value 143, 145

Value Time Quality (VTQ) 13

values

 converting 176

version 137

VIEW (WindowViewer) 139

W

WindowViewer 139

WinSock 14

Wonderware InTouch software 8

X

XML 181

XML Editor 181

