Siemens S7 MPI Driver

Table of Contents

Siemens S7 MPI Driver

Help version 1.032

CONTENTS

Overview

The Siemens S7 MPI Driver provides a reliable way to connect Siemens S7 MPI devices to client applications, including HMI, SCADA, Historian, MES, ERP, and countless custom applications. It is intended for use with Siemens S7 300 and 400 PLCs communicating via the MPI interface.

The Siemens S7 MPI serial port adapter handles MPI connectivity on the PC. This adapter may be obtained from Siemens using the following part numbers:

Siemens Part: 6ES7-972-OCA23-OXAO Version 5.1
Siemens Part: 6ES7-972-OCA22-OXAO Version 5.0

This is just a partial list of the part numbers for more recent versions of this adapter. This driver has been developed to work with versions of the adapter that predate version 5.0 or 5.1.

Setup

Supported Devices

Siemens S7-300 Devices
Siemens S7-400 Devices

Channel and Device Limits

The maximum number of channels supported by this driver is 100. The maximum number of devices supported by this driver is 127 per channel.
The Siemens S7 MPI PC Adapter and the S7 MPI driver have been designed to allow more than one Siemens client device to exist on the network. This allows a programming package to operate while this driver is running or while additional PCs are accessing the network.

Communication Protocol

Multi Point Interface (MPI) S7-300/400 Communications Protocol

PC to S7 PC Adapter*
* S7 PC Adapter to S7-300/400 PLC

Ethernet Encapsulation

This driver supports Ethernet Encapsulation, which allows the driver to communicate with serial devices attached to an Ethernet network using a terminal server (such as the NetSLX). It may be invoked through the Communications dialog in Channel Properties. For more information, refer to the OPC server's help documentation.

Cable Connections - PC to MPI Adapter

The PC to MPI adapter connection is accomplished using a null modem adapter with standard RS232 cable or a null modem cable. The cable provided by Siemens (P/N 6ES7 901-1BF00-OXA0) may also be used. It is a null modem cable, as well.

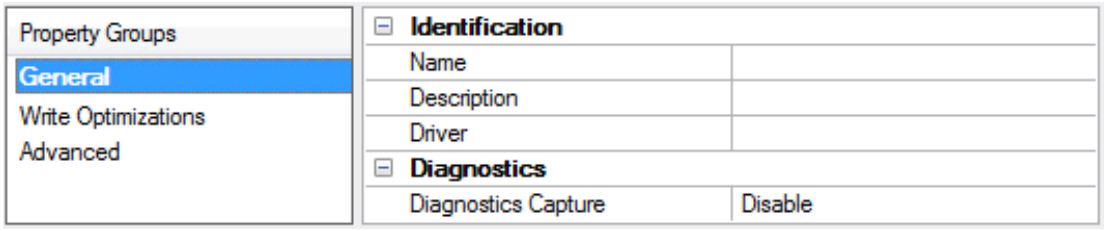
Cable Connections - MPI Adapter to PLC

The Siemens S7 PC Adapter allows quick and easy connection to a single S7-300/400 PLC. Although it is capable of multi-drop operation, additional network wiring is required for proper operation. For more information on establishing a multi-drop MPI network, refer to the Siemens S7 Hardware manual.

Note: Special consideration must be given to the PC Adapter wiring between the S7-300/400 PLCs and the Siemens S7 PC Adapter in a multi-drop configuration (because the adapter is powered from the PLC MPI port).

Channel Properties — General

This server supports the use of multiple simultaneous communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link. This group is used to specify general channel properties, such as the identification attributes and operating mode.



Identification

Name: Specify the user-defined identity of this channel. In each server project, each channel name must be unique. Although names can be up to 256 characters, some client applications have a limited display window when browsing the OPC server's tag space. The channel name is part of the OPC browser information. The property is required for creating a channel.

For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

Description: Specify user-defined information about this channel.

Many of these properties, including Description, have an associated system tag.

Driver: Specify the protocol / driver for this channel. This property specifies the device driver that was selected during channel creation. It is a disabled setting in the channel properties. The property is required for creating a channel.

Note: With the server's online full-time operation, these properties can be changed at any time. This includes changing the channel name to prevent clients from registering data with the server. If a client has already acquired an item from the server before the channel name is changed, the items are unaffected. If, after the channel name has been changed, the client application releases the item and attempts to re-acquire using the old channel name, the item is not accepted. With this in mind, changes to the properties should not be made once a large client application has been developed. Utilize proper user role and privilege management to prevent operators from changing properties or accessing server features.

Diagnostics

Diagnostics Capture: When enabled, this option makes the channel's diagnostic information available to OPC applications allows the usage of statistics tags that provide feedback to client applications regarding the operation of the channel. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.

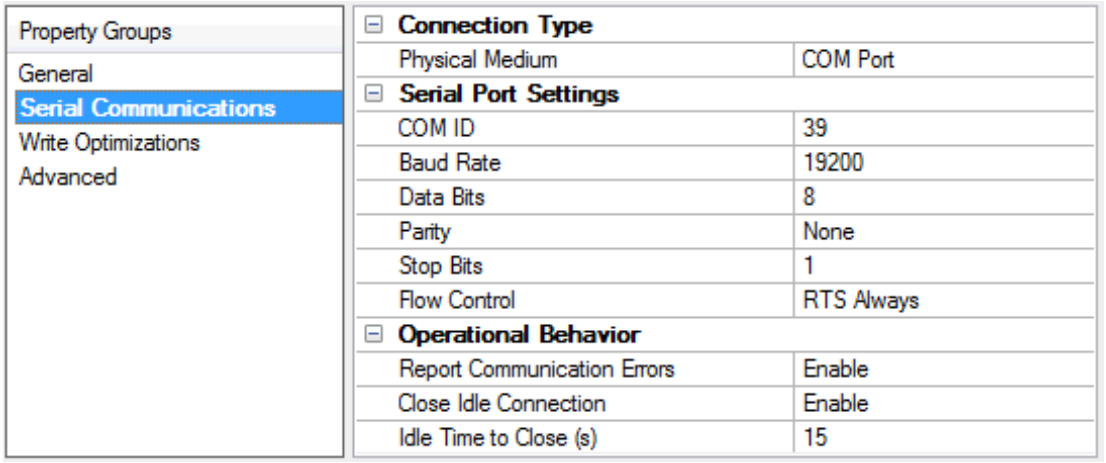
Note: This property is not available if the driver does not support diagnostics.

For more information, refer to "Communication Diagnostics" and "Statistics Tags" in the server help.

Channel Properties — Serial Communications

Serial communication properties are available to serial drivers and vary depending on the driver, connection type, and options selected. Below is a superset of the possible properties.
Click to jump to one of the sections: Connection Type, Serial Port Settings or Ethernet Settings, and Operational Behavior.

Note: With the server's online full-time operation, these properties can be changed at any time. Utilize proper user role and privilege management to prevent operators from changing properties or accessing server features.



Connection Type

Physical Medium: Choose the type of hardware device for data communications. Options include COM Port, None, Modem, and Ethernet Encapsulation. The default is COM Port.

- **None**: Select None to indicate there is no physical connection, which displays the Operation with no Communications section.
- **COM Port**: Select Com Port to display and configure the Serial Port Settings section.
- **Modem**: Select Modem if phone lines are used for communications, which are configured in the Modem Settings section.
- **Ethernet Encap.**: Select if Ethernet Encapsulation is used for communications, which displays the Ethernet Settings section.
- **Shared**: Verify the connection is correctly identified as sharing the current configuration with another channel. This is a read-only property.

Serial Port Settings

COM ID: Specify the Communications ID to be used when communicating with devices assigned to the channel. The valid range is 1 to 9991 to 16. The default is 1.

Baud Rate: Specify the baud rate to be used to configure the selected communications port.

Data Bits: Specify the number of data bits per data word. Options include 5, 6, 7, or 8.

Parity: Specify the type of parity for the data. Options include Odd, Even, or None.

Stop Bits: Specify the number of stop bits per data word. Options include 1 or 2.

Flow Control: Select how the RTS and DTR control lines are utilized. Flow control is required to communicate with some serial devices. Options are:

- **None**: This option does not toggle or assert control lines.
- **DTR**: This option asserts the DTR line when the communications port is opened and remains on.
- **RTS**: This option specifies that the RTS line is high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line is low. This is normally used with RS232/RS485 converter hardware.
- **RTS, DTR**: This option is a combination of DTR and RTS.
- **RTS Always**: This option asserts the RTS line when the communication port is opened and remains on.
- **RTS Manual**: This option asserts the RTS line based on the timing properties entered for RTS Line Control. It is only available when the driver supports manual RTS line control (or when the properties are shared and at least one of the channels belongs to a driver that provides this support).
  RTS Manual adds an **RTS Line Control** property with options as follows:
  - **Raise**: This property specifies the amount of time that the RTS line is raised prior to data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Drop**: This property specifies the amount of time that the RTS line remains high after data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Poll Delay**: This property specifies the amount of time that polling for communications is delayed. The valid range is 0 to 9999. The default is 10 milliseconds.

Tip: When using two-wire RS-485, "echoes" may occur on the communication lines. Since this communication does not support echo suppression, it is recommended that echoes be disabled or a RS-485 converter be used.

Operational Behavior

- **Report Communication Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the COM port. The default is 15 seconds.

Ethernet Settings

Note: Not all serial drivers support Ethernet Encapsulation. If this group does not appear, the functionality is not supported.

Ethernet Encapsulation provides communication with serial devices connected to terminal servers on the Ethernet network. A terminal server is essentially a virtual serial port that converts TCP/IP messages on the Ethernet network to serial data. Once the message has been converted, users can connect standard devices that support serial communications to the terminal server. The terminal server's serial port must be properly configured to match the requirements of the serial device to which it is attached. *For more information, refer to "Using Ethernet Encapsulation" in the server help.*

- **Network Adapter**: Indicate a network adapter to bind for Ethernet devices in this channel. Choose a network adapter to bind to or allow the OS to select the default.

Specific drivers may display additional Ethernet Encapsulation properties. For more information, refer to Channel Properties — Ethernet Encapsulation.

Modem Settings

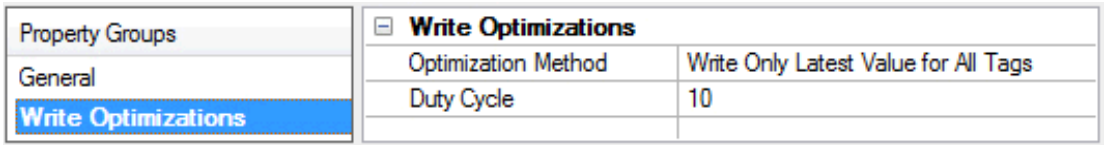
- **Modem**: Specify the installed modem to be used for communications.
- **Connect Timeout**: Specify the amount of time to wait for connections to be established before failing a read or write. The default is 60 seconds.
- **Modem Properties**: Configure the modem hardware. When clicked, it opens vendor-specific modem properties.
- **Auto-Dial**: Enables the automatic dialing of entries in the Phonebook. The default is Disable. *For more information, refer to "Modem Auto-Dial" in the server help.*
- **Report Communication Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the modem connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the modem connection. The default is 15 seconds.

Operation with no Communications

- **Read Processing**: Select the action to be taken when an explicit device read is requested. Options include Ignore and Fail. Ignore does nothing; Fail provides the client with an update that indicates failure. The default setting is Ignore.

Channel Properties — Write Optimizations

As with any server, writing data to the device may be the application's most important aspect. The server intends to ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties that can be used to meet specific needs or improve application responsiveness.



Write Optimizations

Optimization Method: Controls how write data is passed to the underlying communications driver. The options are:

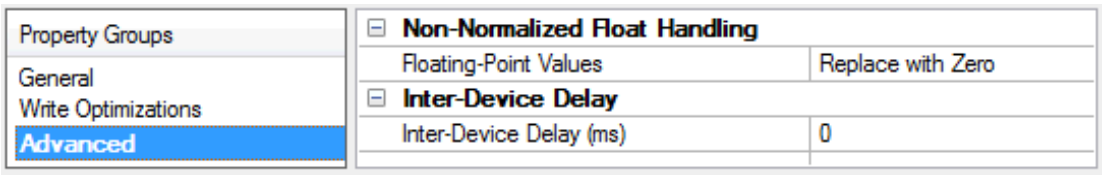
- **Write All Values for All Tags**: This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's internal write queue. The server processes the write queue and attempts to empty it by writing data to the device as quickly as possible. This mode ensures that everything written from the client applications is sent to the target device. This mode should be selected if the write operation order or the write item's content must uniquely be seen at the target device.

- **Write Only Latest Value for Non-Boolean Tags**: Many consecutive writes to the same value can accumulate in the write queue due to the time required to actually send the data to the device. If the server updates a write value that has already been placed in the write queue, far fewer writes are needed to reach the same final output value. In this way, no extra writes accumulate in the server's queue. When the user stops moving the slide switch, the value in the device is at the correct value at virtually the same time. As the mode states, any value that is not a Boolean value is updated in the server's internal write queue and sent to the device at the next possible opportunity. This can greatly improve the application performance.
  Note: This option does not attempt to optimize writes to Boolean values. It allows users to optimize the operation of HMI data without causing problems with Boolean operations, such as a momentary push button.
- **Write Only Latest Value for All Tags**: This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write queue before they are sent. This is the default mode.

Duty Cycle: is used to control the ratio of write to read operations. The ratio is always based on one read for every one to ten writes. The duty cycle is set to ten by default, meaning that ten writes occur for each read operation. Although the application is performing a large number of continuous writes, it must be ensured that read data is still given time to process. A setting of one results in one read operation for every write operation. If there are no write operations to perform, reads are processed continuously. This allows optimization for applications with continuous writes versus a more balanced back and forth data flow.
Note: It is recommended that the application be characterized for compatibility with the write optimization enhancements before being used in a production environment.

Channel Properties — Advanced

This group is used to specify advanced channel properties. Not all drivers support all properties; so the Advanced group does not appear for those devices.



Non-Normalized Float Handling: A non-normalized value is defined as Infinity, Not-a-Number (NaN), or as a Denormalized Number. The default is Replace with Zero. Drivers that have native float handling may default to Unmodified. Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. Descriptions of the options are as follows:

- **Replace with Zero**: This option allows a driver to replace non-normalized IEEE-754 floating point values with zero before being transferred to clients.
- **Unmodified**: This option allows a driver to transfer IEEE-754 denormalized, normalized, non-number, and infinity values to clients without any conversion or changes.

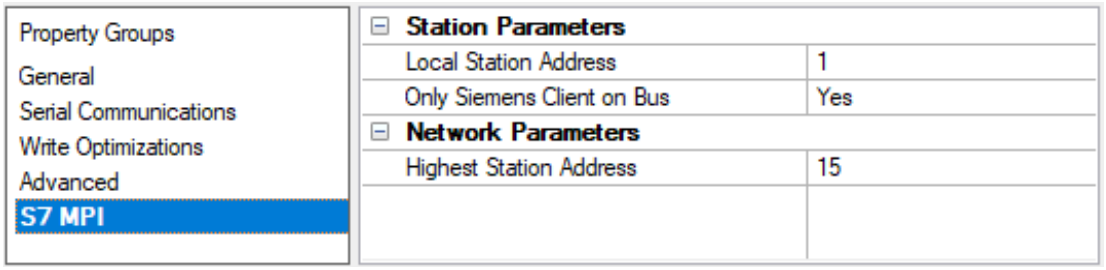
Note: This property is not available if the driver does not support floating point values or if it only supports the option that is displayed. According to the channel's float normalization setting, only real-time driver tags (such as values and arrays) are subject to float normalization. For example, EFM data is not affected by this setting.

For more information on the floating point values, refer to "How To ... Work with Non-Normalized Floating Point Values" in the server help.

Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

Note: This property is not available for all drivers, models, and dependent settings.

Channel Properties — S7 MPI Settings



Station Parameters

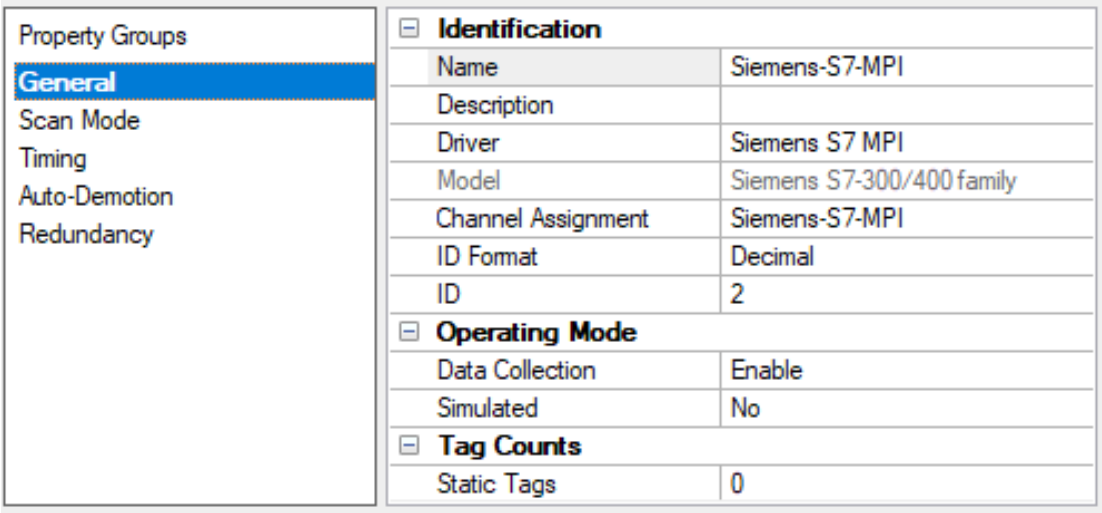
Local Station Address: Specify the node number used by the Siemens S7 PC Adapter to determine its address on the MPI network. It must not conflict with any other node number used on the network. The valid local station addresses are 0 to 126. The default setting is 1.

Only Siemens Client on Bus: Specify if there is more than one Siemens client present on the MPI network. This information is required for proper operation of the driver. For an application that requires direct peer-to-peer connection to a single PLC, this selection should be Yes. For a multi-drop network in which multiple Siemens clients are present, this selection should be No. The default setting is Yes.

Network Parameters

Highest Station Address: Specify the highest MPI node that can exist on the network. It is required for proper network operation and is determined by the highest PLC node on the MPI network. The four possible selection are 15, 31, 63, and 126. This selection can impact operation of the S7 MPI driver in two ways. If the value is too low, users may not be able to access PLCs with higher address settings. If the value is too high, the network performance may decrease as the S7 PC adapter attempts to locate PLCs within a larger range than necessary. For best network performance, S7 PLCs should start at address 3 and ascend consecutively. The default setting is 15.

Device Properties — General



Identification

Name: User-defined identity of this device.

Description: User-defined information about this device.

Channel Assignment: User-defined name of the channel to which this device currently belongs.

Driver: Selected protocol driver for this device.

Model: Select the specific version of the device.

ID Format: Select how the device identity is formatted. Options include Decimal, Octal, and Hex.

ID: Specify the unique identity of the device for communication with the driver. The valid range is 0 to 126. Any devices defined under this channel should not use an ID that conflicts with the channel's Local Station Address.

Operating Mode

Data Collection: This property controls the device's active state. Although device communications are enabled by default, this property can be used to disable a physical device. Communications are not attempted when a device is disabled. From a client standpoint, the data is marked as invalid and write operations are not accepted. This property can be changed at any time through this property or the device system tags.

Simulated: This option places the device into Simulation Mode. In this mode, the driver does not attempt to communicate with the physical device, but the server continues to return valid OPC data. Simulated stops physical communications with the device, but allows OPC data to be returned to the OPC client as valid data. While in Simulation Mode, the server treats all device data as reflective: whatever is written to the simulated device is read back and each OPC item is treated individually. The item's memory map is based on the group Update Rate. The data is not saved if the server removes the item (such as when the server is reinitialized). The default is No.

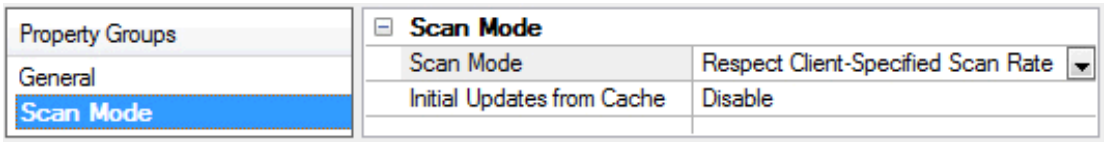
Notes:

1. This System tag (_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.
2. In Simulation mode, the item's memory map is based on client update rate(s) (Group Update Rate for OPC clients or Scan Rate for native and DDE interfaces). This means that two clients that reference the same item with different update rates return different data.

Simulation Mode is for test and simulation purposes only. It should never be used in a production environment.

Device Properties — Scan Mode

The Scan Mode specifies the subscribed-client requested scan rate for tags that require device communications. Synchronous and asynchronous device reads and writes are processed as soon as possible; unaffected by the Scan Mode properties.



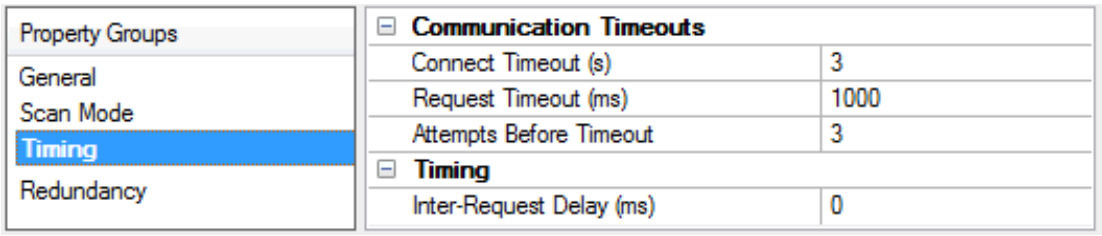
Scan Mode: Specify how tags in the device are scanned for updates sent to subscribing clients. Descriptions of the options are:

- **Respect Client-Specified Scan Rate**: This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate**: This mode specifies the value set as the maximum scan rate. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  Note: When the server has an active client and items for the device and the scan rate value is increased, the changes take effect immediately. When the scan rate value is decreased, the changes do not take effect until all client applications have been disconnected.
- **Request All Data at Scan Rate**: This mode forces tags to be scanned at the specified rate for subscribed clients. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
- **Do Not Scan, Demand Poll Only**: This mode does not periodically poll tags that belong to the device nor perform a read to get an item's initial value once it becomes active. It is the client's responsibility to poll for updates, either by writing to the _DemandPoll tag or by issuing explicit device reads for individual items. *For more information, refer to "Device Demand Poll" in server help*.
- **Respect Tag-Specified Scan Rate**: This mode forces static tags to be scanned at the rate specified in their static configuration tag properties. Dynamic tags are scanned at the client-specified scan rate.

Initial Updates from Cache: When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. A device read is used for the initial update for the first client reference only. The default is disabled; any time a client activates a tag reference the server attempts to read the initial value from the device.

Device Properties — Timing

The device Timing properties allow the driver's response to error conditions to be tailored to fit the application's needs. In many cases, the environment requires changes to these properties for optimum performance. Factors such as electrically generated noise, modem delays, and poor physical connections can influence how many errors or timeouts a communications driver encounters. Timing properties are specific to each configured device.



Communications Timeouts

Connect Timeout: This property (which is used primarily by Ethernet based drivers) controls the amount of time required to establish a socket connection to a remote device. The device's connection time often takes longer than normal communications requests to that same device. The valid range is 1 to 30 seconds. The default is typically 3 seconds, but can vary depending on the driver's specific nature. If this setting is not supported by the driver, it is disabled.

Note: Due to the nature of UDP connections, the connection timeout setting is not applicable when communicating via UDP.

Request Timeout: Specify an interval used by all drivers to determine how long the driver waits for a response from the target device to complete. The valid range is 50 to 9,999,999 milliseconds (167.6667 minutes). The default is usually 1000 milliseconds, but can vary depending on the driver. The default timeout for most serial drivers is based on a baud rate of 9600 baud or better. When using a driver at lower baud rates, increase the timeout to compensate for the increased time required to acquire data.

Attempts Before Timeout: Specify how many times the driver issues a communications request before considering the request to have failed and the device to be in error. The valid range is 1 to 10. The default is typically 3, but can vary depending on the driver's specific nature. The number of attempts configured for an application depends largely on the communications environment. This property applies to both connection attempts and request attempts.

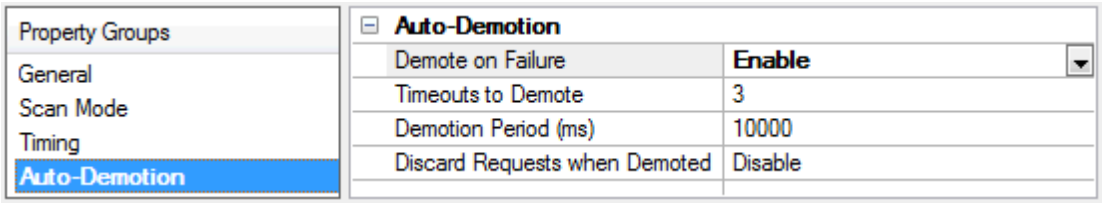
Timing

Inter-Request Delay: Specify how long the driver waits before sending the next request to the target device. It overrides the normal polling frequency of tags associated with the device, as well as one-time reads and writes. This delay can be useful when dealing with devices with slow turnaround times and in cases where network load is a concern. Configuring a delay for a device affects communications with all other devices on the channel. It is recommended that users separate any device that requires an inter-request delay to a separate channel if possible. Other communications properties (such as communication serialization) can extend this delay. The valid range is 0 to 300,000 milliseconds; however, some drivers may limit the maximum value due to a function of their particular design. The default is 0, which indicates no delay between requests with the target device.

Note: Not all drivers support Inter-Request Delay. This setting does not appear if it is not available.

Device Properties — Auto-Demotion

The Auto-Demotion properties can temporarily place a device off-scan in the event that a device is not responding. By placing a non-responsive device offline for a specific time period, the driver can continue to optimize its communications with other devices on the same channel. After the time period has been reached, the driver re-attempts to communicate with the non-responsive device. If the device is responsive, the device is placed on-scan; otherwise, it restarts its off-scan time period.



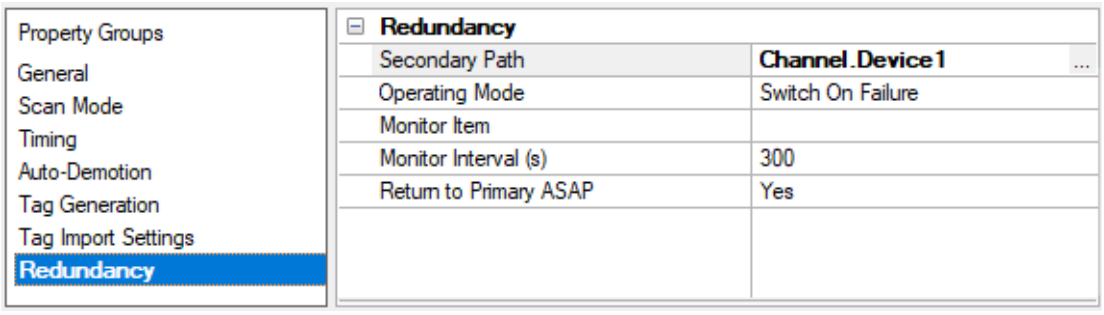
Demote on Failure: When enabled, the device is automatically taken off-scan until it is responding again.
Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

Timeouts to Demote: Specify how many successive cycles of request timeouts and retries occur before the device is placed off-scan. The valid range is 1 to 30 successive failures. The default is 3.

Demotion Period: Indicate how long the device should be placed off-scan when the timeouts value is reached. During this period, no read requests are sent to the device and all data associated with the read requests are set to bad quality. When this period expires, the driver places the device on-scan and allows for another attempt at communications. The valid range is 100 to 3600000 milliseconds. The default is 10000 milliseconds.

Discard Requests when Demoted: Select whether or not write requests should be attempted during the off-scan period. Disable to always send write requests regardless of the demotion period. Enable to discard writes; the server automatically fails any write request received from a client and does not post a message to the Event Log.

Device Properties — Redundancy



Redundancy is available with the Media-Level Redundancy Plug-In.
Consult the website, a sales representative, or the user manual for more information.

Data Types Description

Data Type	Description	IEC 1131 Data Type
Boolean	Single bit of an 8-bit value*	BOOL
Byte	Unsigned 8-bit value	Byte
Char	Signed 8-bit value	Char
Word	Unsigned 16-bit value	Word
Short	Signed 16-bit value	INT
BCD	Two byte packed BCD Value range is 0-9999. Behavior is undefined for values beyond this range.	Word
DWord	Unsigned 32-bit value	DWORD
Long	Signed 32-bit value	DINT
LBCD	Four byte packed BCD Value range is 0-99999999. Behavior is undefined for values beyond this range.	DWORD
Float	32-bit floating point value The driver interprets two consecutive registers as a floating point value by making the second register the high word and the first register the low word.	REAL
String	Null terminated ASCII string**	STRING

For more information, refer to* ***Address Descriptions**.*

** The Data Block subtype STRING is a null padded ASCII String.

Address Descriptions

The default data types for dynamically defined tags are shown in **bold**.

Address Type	Range	Data Type	Access
Discrete Inputs	I00000.b-I65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Discrete Inputs	IB00000-IB65535	**Byte**, Char, String*	Read/Write
Discrete Inputs	IW00000-IW65535	**Word**, Short, BCD	Read/Write
Discrete Inputs	ID00000-ID65535	**DWord**, Long, Float	Read/Write
Discrete Inputs	E00000.b-E65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Discrete Inputs	EB00000-EB65535	**Byte**, Char, String*	Read/Write
Discrete Inputs	EW00000-EW65535	**Word**, Short, BCD	Read/Write
Discrete Inputs	ED00000-ED65535	**DWord**, Long, Float	Read/Write
Note: I and E access the same memory area.			
Discrete Outputs	Q00000.b-Q65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Discrete Outputs	QB00000-QB65535	**Byte**, Char, String*	Read/Write
Discrete Outputs	QW00000-QW65535	**Word**, Short, BCD	Read/Write
Discrete Outputs	QD00000-QD65535	**DWord**, Long, Float	Read/Write
Discrete Outputs	A00000.b-A65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Discrete Outputs	AB00000-AB65535	**Byte**, Char, String*	Read/Write
Discrete Outputs	AW00000-AW65535	**Word**, Short, BCD	Read/Write
Discrete Outputs	AD00000-AD65535	**DWord**, Long, Float	Read/Write
Note: Q and A access the same memory area.			
Flag Memory	F00000.b-F65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Flag Memory	FB00000-FB65535	**Byte**, Char, String*	Read/Write
Flag Memory	FW00000-FW65535	**Word**, Short, BCD	Read/Write
Flag Memory	FD00000-FD65535	**DWord**, Long, Float	Read/Write
Flag Memory	M00000.b-M65535.b .b is Bit Number 0-7	**Boolean**	Read/Write
Flag Memory	MB00000-MB65535	**Byte**, Char, String*	Read/Write
Flag Memory	MW00000-MW65535	**Word**, Short, BCD	Read/Write
Flag Memory	MD00000-MD65535	**DWord**, Long, Float	Read/Write
Note: F and M access the same memory area.			
Data Block	DB1-N.DBX00000.b-DBX65535.b 1-N is DB Block Number .b is Bit Number 0-7	**Boolean**	Read/Write
Data Block	DB1-N.DBB00000-DBB65535 1-N is DB Block Number	**Byte**, Char, String*	Read/Write

Address Type	Range	Data Type	Access
Data Block	DB1-N.DBW00000-DBW65535 1-N is DB Block Number	**Word**, Short, BCD	Read/Write
Data Block	DB1-N.DBD00000-DBD65535 1-N is DB Block Number	**DWord**, Long, Float	Read/Write
Timer Current Values	T00000-T65535	**DWord**	Read Only
Counter Current Values	C00000-C65535	**BCD**, Word, Short	Read Only
Counter Current Values	Z00000-Z65535	**BCD**,Word, Short	Read Only

* Byte memory types (i.e. MB) support Strings. The syntax for strings is *<address>.<length>* where 0 < length <= 212.

Note: The actual number of addresses of each type depends on the Siemens S7-300 or S7-400 device in use. Each type does not necessarily support an address of 0 to 65535. For a list of all address ranges, refer to the device's documentation. All offsets represent a byte starting location within the specified memory type.

Arrays

All memory types and subtypes support arrays, except for Boolean data types, timers and counters. The valid syntax for declaring an array is as shown below. If no rows are specified, a row count of 1 is assumed.

<address>[rows][cols]

For Word, Short, BCD, and UBCD arrays; the base address + (rows * cols * 2) cannot exceed 65536. The elements of the array are words, located on a word boundary. For example, AW0[4] would return AW0, AW2, AW4, and AW6.

For Float, DWord, Long, Long BCD, KF, and KG arrays; the base address + (rows * cols * 4) cannot exceed 65536. The elements of the array are DWords, located on a DWord boundary. For example, AD0[4] returns AD0, AD4, AD8, and AD12.

For all arrays, the total number of bytes being requested cannot exceed the internal block size of 218 bytes.

Examples

- To access bit 3 of Flag Memory F20, declare an address as follows: F20.3
- To access Data Block 5 as word memory at byte offset 30, declare an address as follows: DB5.DBW30
- To access Data Block 2 as Boolean at byte offset 20 and bit 7, declare an address as follows: DB2.DBX20.7
- To access Data Block 1 as byte memory at byte offset 10, declare an address as follows: DB1.DBB10
- To access Flag Memory F20 as a DWORD, declare an address as follows: FD20
- To access Input Memory I10 as a Word, declare an address as follows: IW10

Data Block Strings

To reference Data Block Strings, the STRING subtype is used.

STRING Subtype

The STRING subtype follows the STEP 7 STRING data type definition. The syntax for the STRING subtype is *DBx.STRINGy.n* where *x* is the Data Block, *y* is the Byte offset, and *n* is the maximum String length. If *n* is not specified, the maximum String length is 210 characters. String values read and written are stored at Byte offset y+2 in Data Block x. The first two bytes contain the "maximum string length (n)" and the "actual string length". The "actual string length" gets updated with every write based on the string length of the string being written.

y	y+1	y+2	y+3	y+4	...	y+2+n-1
max string length (n)	actual string length	' '	' '	' '	...	' '

Note: STRING Strings are NULL padded. If the maximum string length is 10 and 3 characters are written, characters 4-10 are set to NULL.

When modifying Word, Short, DWord, Long, and Float types, remember that each address starts at a byte offset within the device. Therefore, Words MW0 and MW1 overlap at byte 1. Writing to MW0 modifes the value held in MW1. Similarly, DWord, Long and Float types can also overlap. It is recommended that these memory types be used so that overlapping does not occur. For example, when using DWords, use MD0, MD4, MD8, and so on to prevent overlapping bytes.

Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the server help on filtering and sorting the Event Log detail view. Server help contains many common messages, so should also be searched. Generally, the type of message (informational, warning) and troubleshooting information is provided whenever possible.

Connection timeout on MPI node <device ID>.

Error Type:

Warning

Possible Cause:

1. The serial connection between the device and MPI adapter (or the MPI adapter and the host PC) is invalid.
2. The communications parameters for the serial connection are incorrect.
3. The named device may have an incorrect network ID.

Possible Solution:

1. Verify the cabling between the device and MPI adapter (or the MPI adapter and PC) is connected and intact.
2. Verify that the specified communications parameters match those of the device.
3. Verify that the network ID given to the named device matches that of the actual device.

Request timeout on MPI node <device ID>.

Error Type:

Warning

Possible Cause:

1. The serial connection between the device and MPI adapter (or the MPI adapter and the host PC) is invalid.
2. The communications parameters for the serial connection are incorrect.
3. The named device may have been assigned an incorrect Network ID.

Possible Solution:

1. Verify the cabling between the device and MPI adapter (or the MPI adapter and PC) is connected and intact.
2. Verify that the specified communications parameters match those of the device.
3. Verify that the network ID given to the named device matches that of the actual device.

Bad address in block. The block has been deactivated. | Block range = '<address>' to '<address>'.

Error Type:
Warning

Possible Cause:
An attempt was made to reference a block of memory that contains at least one non-existent location in the specified device.

Possible Solution:
Verify that the tags assigned to addresses are within the specified range on the device and eliminate any that reference invalid locations.

Error Mask Definitions

B = Hardware break detected
F = Framing error
E = I/O error
O = Character buffer overrun
R = RX buffer overrun
P = Received byte parity error
T = TX buffer full

Index

E

F

G

I

L

M

N

O

R

S

T

W