Simatic / TI 505 Serial Driver

© 2018 PTC Inc. All Rights Reserved.

Table of Contents

Simatic / TI 505 Serial Driver

Help version 1.025

CONTENTS

Overview

The Simatic / TI 505 Serial Driver provides a reliable way to connect Simatic/TI 505 serial devices to OPC dlient applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications.

The driver is a serial driver intended for use with TI 500/505 PLCs using the programming port of the processor. The driver supports two protocols, Non-Intelligent Terminal Protocol (NITP) and Transparent Byte (TB). Both protocols are point-to-point only, meaning only one processor can be connected at a time.

All TI 500/505 processors support the NITP protocol. The NITP protocol is an ASCII protocol. Most processors also support the TB protocol, which is a binary protocol and faster. Processors do not have to be configured to use one protocol or another. Processors that do not support the TB protocol will ignore TB requests.

The protocol selection is made when configuring a device. If NITP is selected, the parity setting for the COM port must be odd and the number of data bits 7. If TB is selected the parity must be none and the number of data bits 8. The baud rate should match the setting in the PLC. RTS_ALWAYS flow control must be selected for either protocol.

An RS232 cable with a null modem is used to connect the PC to the processor. This is the same cable that is used with the TISOFT programming software.

Setup

Supported Devices

TI Series 500/505 processors: 520, 525, 535, 545, 555, 565 and 575.

Communication Protocol

Non-Intelligent Terminal Protocol (NITP)
Transparent Byte protocol (TB)

Supported Communication Properties*

Baud Rate: 300, 600, 1200, 2400, 9600, 19200, or 38400
Stop Bits: 1
Parity: Odd for NITP, None for TB
Data Bits: 7 for NITP, 8 for TB

*Not all devices support the listed configurations.

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 or device server. Ethernet Encapsulation mode is invoked by selecting it from the COM ID property group on the Channel Properties. More help on Ethernet Encapsulation can be found in the main OPC Server help file. When used directly with a serial port, this driver supports only a single connection to a single controller per serial port. When operating in the Ethernet Encapsulation mode, the driver will support up to 31 controllers per channel. In this mode a single controller can be paired with a terminal server/device server to form a single node.

Flow Control

When using an RS232/RS485 converter, the type of flow control that is required will depend upon the needs of the converter. Some converters do not require any flow control and others will require RTS flow. Consult the converter's documentation in order to determine its flow requirements. We recommend using an RS485 converted that provides automatic flow control.

Note: When using the manufacturer's supplied communications cable, it is sometimes necessary to choose a flow control property of **RTS** or **RTS Always** under Channel Properties. For the Simatic/TI 505 Serial select RTS_ALWAYS for either protocol.

Channel Properties

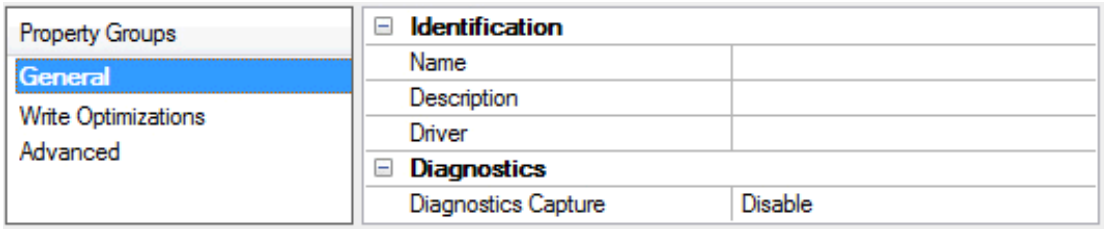
This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link.

The properties associated with a channel are broken in to logical groupings. While some groups are specific to a given driver or protocol, the following are the common groups:

General
Ethernet or Serial Communications
Write Optimization
Advanced

Channel Properties — General

This server supports the use of simultaneous multiple 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: 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.
For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

Description: User-defined information about this channel.
Many of these properties, including Description, have an associated system tag.

Driver: Selected 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.
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 the User Manager to prevent operators from changing properties and restrict access rights to server features.

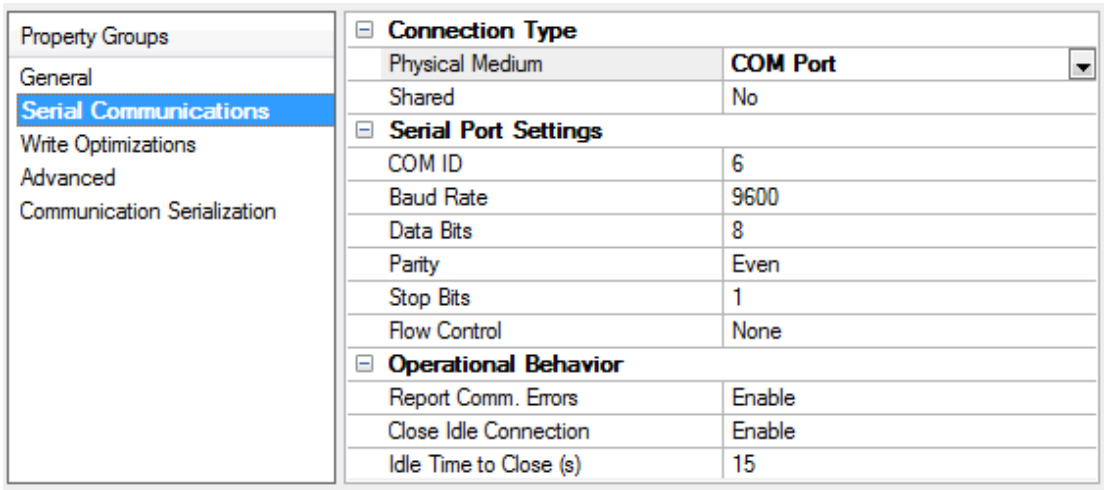
Diagnostics

Diagnostics Capture: When enabled, this option makes the channel's diagnostic information available to OPC applications. 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 disabled if the driver does not support diagnostics.
For more information, refer to "Communication Diagnostics" 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 the User Manager to restrict access rights to server features, as changes made to these properties can temporarily disrupt communications.



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 Comm. 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 "How To... Use 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

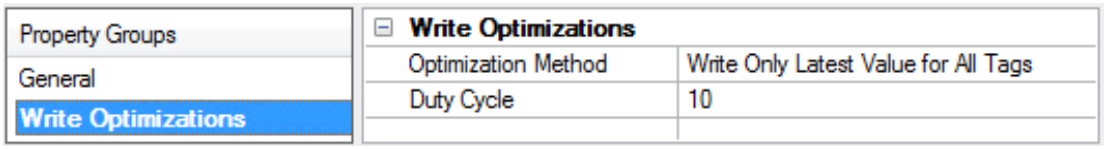
- **Modem**: Specify the installed modem to be used for communications.
- **Connect Timeout**: Specify the amount of time to wait for connections to be established before failing a read or write. The default is 60 seconds.
- **Modem Properties**: Configure the modem hardware. When clicked, it opens vendor-specific 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 Comm. 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 OPC 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.

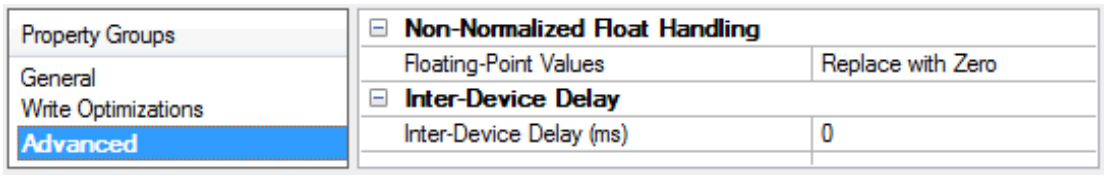
- **Write Only Latest Value for All Tags**: This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write 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 disabled 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.

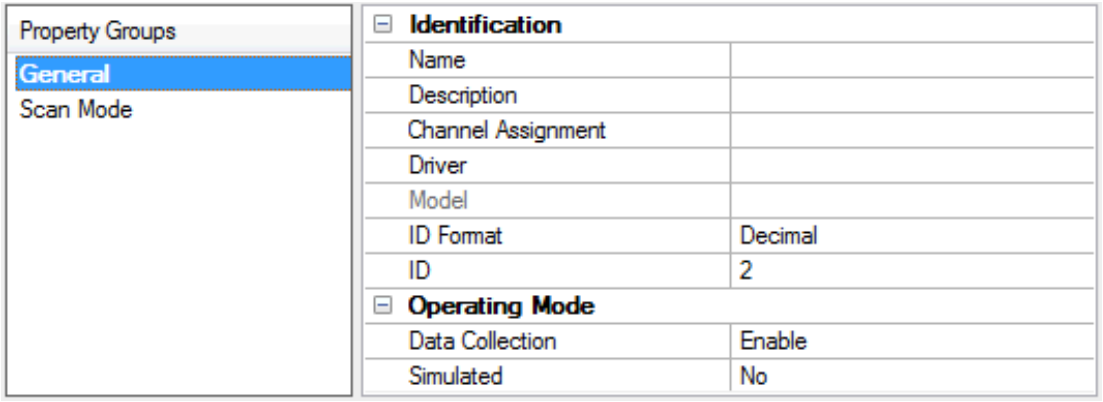
Device Properties

Device properties are organized into the following groups. Click on a link below for details about the properties in that group.

Identification
Operating Mode
Scan Mode
Communication Timeouts
Auto-Demotion
Redundancy

Device Properties — General

A device represents a single target on a communications channel. If the driver supports multiple controllers, users must enter a device ID for each controller.



Identification

Name: This property specifies the name of the device. It is a logical user-defined name that can be up to 256 characters long, and may be used on multiple channels.
Note: Although descriptive names are generally a good idea, some OPC client applications may have a limited display window when browsing the OPC server's tag space. The device name and channel name become part of the browse tree information as well. Within an OPC client, the combination of channel name and device name would appear as "ChannelName.DeviceName".
For more information, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in server help.

Description: User-defined information about this device.
Many of these properties, including Description, have an associated system tag.

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

Driver: Selected protocol driver for this device. This property specifies the driver selected during channel creation. It is disabled in the channel properties.

Model: This property specifies the specific type of device that is associated with this ID. The contents of the drop-down menu depends on the type of communications driver being used. Models that are not supported by a driver are disabled. If the communications driver supports multiple device models, the model selection can only be changed when there are no client applications connected to the device.

Note: If the communication driver supports multiple models, users should try to match the model selection to the physical device. If the device is not represented in the drop-down menu, select a model that conforms closest to the target device. Some drivers support a model selection called "Open," which allows users to communicate without knowing the specific details of the target device. For more information, refer to the driver help documentation.

ID: This property specifies the device's station / node / identity / address. The type of ID entered depends on the communications driver being used. For many drivers, the ID is a numeric value. Drivers that support a Numeric ID provide users with the option to enter a numeric value whose format can be changed to suit the needs of the application or the characteristics of the selected communications driver. The ID format can be Decimal, Octal, and Hexadecimal. If the driver is Ethernet-based or supports an unconventional station or node name, the device's TCP/IP address may be used as the device ID. TCP/IP addresses consist of four values that are separated by periods, with each value in the range of 0 to 255. Some device IDs are string based. There may be additional properties to configure within the ID field, depending on the driver.

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.

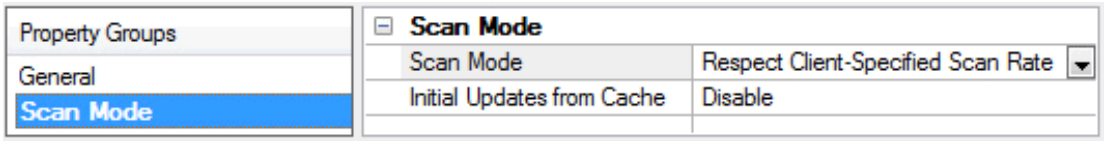
Notes:

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.



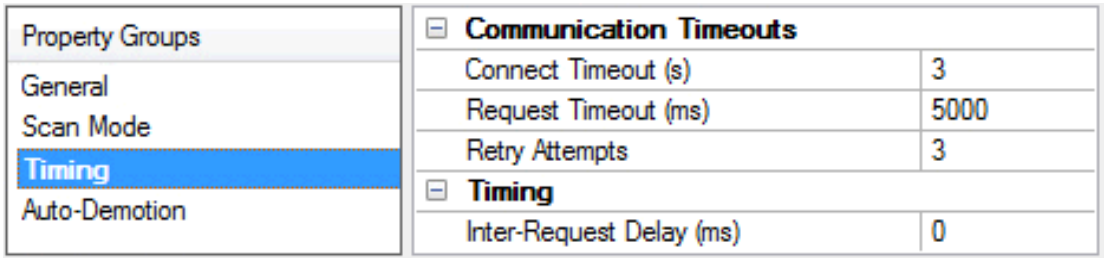
Scan Mode: specifies how tags in the device are scanned for updates sent to subscribed clients. Descriptions of the options are:

- **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 maximum scan rate to be used. 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: This property specifies 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: This property specifies 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.

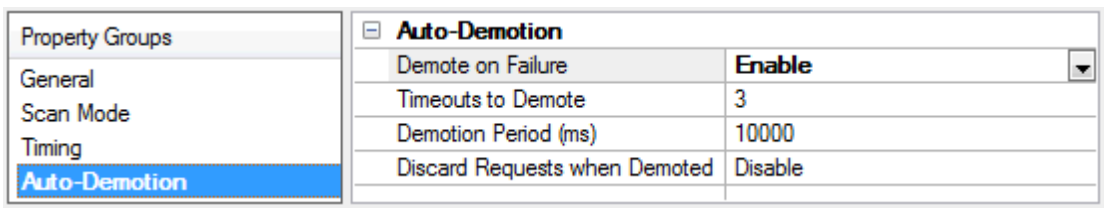
Timing

Inter-Request Delay: This property specifies 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.

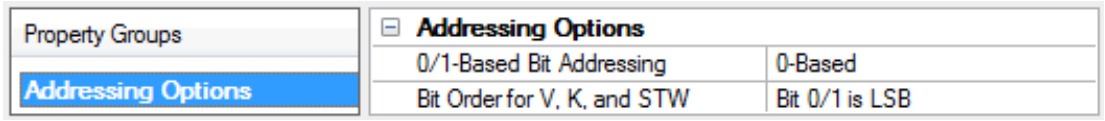
Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

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 — Addressing Options



0/1-Based Bit Addressing

Memory types that allow bit within Word (for example, V) can be referenced as a Boolean. The addressing notation for doing this is as follows:

<memory type><address>.<bit>

where <bit> represents the bit number within the Word or DWord, depending on the memory type. 0/1-Based Bit Addressing provides two ways of addressing a bit within the given Word or DWord: 0-Based and 1-Based. 0-Based addressing simply means the "first bit" begins at 0. With 1-Based, the first bit begins at 1. The bit order for the Word or DWord is irrelevant with this option. In other words, it doesn't matter whether the first bit is the Most Significant Bit or the Least Significant Bit.

Note: In this driver, the first bit will either be bit 0 or bit 1 depending on this 0/1-Based Bit Addressing property.

0-Based (Default Setting)

Data Type	Bit Range
Word	Bits 0 – 15
DWord	Bits 0 – 31

1-Based

Data Type	Bit Range
Word	Bits 1 – 16

Data Type	Bit Range
DWord	Bits 1 – 32

Note: 0/1-Based Bit Addressing does not apply to non-bit addresses such as Word addresses in V memory. These addresses are always 1-Based and are not configurable.

Bit Order for V, K, STW

This option is used to select the order in which bits will be presented to V, K, and STW memory types when bit-accessed.

Note: For the following example, the 1st through 16th bit signifies either 0-15 bits or 1-16 bits depending on if the driver is set at 0-Based Bit Addressing or 1-Based.

DWord follows the same bit order logic as Words except instead of 16 bits, there are 32 bits.

Bit 0 Is MSB of Word

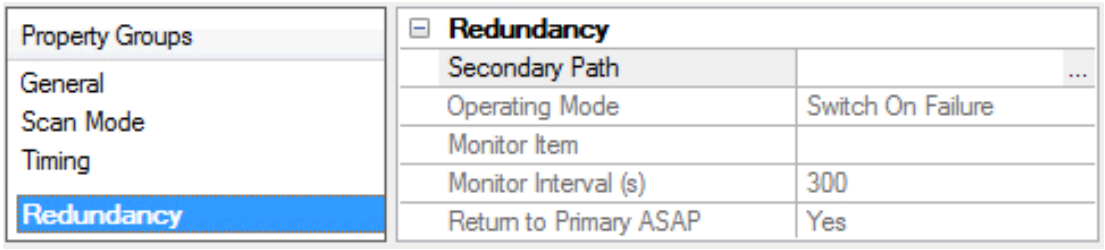
MSB															LSB
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Bit 1 Is LSB (Default Setting) of Word

MSB															LSB
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

See Also: *Device Setup*

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
Boolean	Single bit
Word	Unsigned 16 bit value bit 0 is the low bit bit 15 is the high bit
Short	Signed 16 bit value bit 0 is the low bit bit 14 is the high bit bit 15 is the sign bit
DWord	Unsigned 32 bit value bit 0 is the low bit bit 31 is the high bit
Long*	Signed 32 bit value bit 0 is the low bit bit 30 is the high bit bit 31 is the sign bit
Float*	32 bit floating point value. The driver interprets two consecutive registers as a floating-point value by making the first register the high word and the second register the low word.

*Long is the same as Double in the TISOFT programming software.

*Float is the same as Real in the TISOFT programming software.

Address Descriptions

Address specifications vary depending on the model in use. Select a link from the following list to obtain specific address information for the model of interest.

NITP Addressing
Transparent Byte Addressing
Status Words

NITP Addressing

The Simatic / TI 505 Serial Driver supports the following addresses when using the NITP protocol. The default data type for each address type is indicated in **bold**.

Address Type	Format	Range	Data Types	Access
Discrete Inputs	X<address>	1-65536	**Boolean**	Read/Write
Discrete Outputs	Y<address>	1-65536	**Boolean**	Read/Write
Word Inputs	WX<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Word Outputs	WY<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Discrete Control	C<address>	1-65536	**Boolean**	Read/Write
Word Memory	V<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Word Memory Bit Access	V<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read/Write
Constant Memory	K<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read Only
Constant Memory Bit Access	K<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read Only
Status Words	STW<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Status Words Bit Access	STW<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read/Write
Timer/Counter Preset	TCP<address>	1-65535	Short, **Word**	Read/Write
Timer/Counter Current	TCC<address>	1-65535	Short, **Word**	Read/Write
Drum Step Preset	DSP<address>	1-32767	Short, **Word**	Read/Write
Drum Step Current	DSC<address>	1-32767	Short, **Word**	Read/Write
Drum Step Current Count	DCC<address>	1-32767	Short, **Word**	Read Only
Drum Time Base	DTB<address>	1-32767	Short, **Word**	Read/Write
Drum Count Preset	DCP<drum] .<step>	drum 1-32767 step 1-16	Short, **Word**	Read/Write

*For more information, refer to **0/1-Based Bit Addressing Option**.

The actual number of addresses available for of each type is dependent on the configuration of the PLC. If at runtime the driver finds that an address is not present in the device, the driver will post an error message and remove the tag from its scan list.

V Memory Access as Arrays

The Simatic / TI 505 Serial Driver supports access to V memory in an array. The size of the array for NITP mode is limited to 100 V memory registers per array. When accessing large arrays, multiple read/write commands are used to access consecutive V memory addresses and may require additional time to process. To access V memory as an array, array notation must be used when entering an address. Array notation is shown in the following examples:

V100 [5]	This denotes an array starting at V100 with a length of 5. This means that the array contains V100, V101, V102, V103, and V104.
V100 [2][3]	This denotes a two dimensional array starting at V100 and containing V100, V101, V102, V103, V104, and V105 in a 2 by 3 array.

Note: Arrays can be either the Word or SHORT data type, with a default of Word.

Transparent Byte Addressing

The Simatic / TI 505 Serial Driver supports the following addresses when using the Tranparent Byte protocol. The default data type for each address type is indicated in **bold**.

Address Type	Format	Range	Data Types	Access
Discrete Inputs	X<address>	1-65536	**Boolean**	Read/Write
Discrete Outputs	Y<address>	1-65536	**Boolean**	Read/Write
Word Inputs	WX<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Word Outputs	WY<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Discrete Control	C<address>	1-65536	**Boolean**	Read/Write
Word Memory	V<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read/Write
Word Memory Bit Access	V<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read/Write
Constant Memory	K<address>	1-65536 1-65535	Short, **Word** Long, DWord, Float	Read Only
Constant Memory Bit Access	K<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read Only
Status Words	STW<address>	1-65536	Short, **Word**	Read/Write

Address Type	Format	Range	Data Types	Access
		1-65535	Long, DWord, Float	
Status Words Bit Access	STW<address> .bit	address: 1-65536* bit: 0-15	**Boolean**	Read/Write
Timer/Counter Preset	TCP<address>	1-65535	Short, **Word**	Read/Write
Timer/Counter Current	TCC<address>	1-65535	Short, **Word**	Read/Write
Drum Step Preset	DSP<address>	1-32767	Short, **Word**	Read/Write
Drum Step Current	DSC<address>	1-32767	Short, **Word**	Read/Write
Drum Step Current Count	DCC<address>	1-32767	Short, **Word**	Read Only
Drum Time Base	DTB<address>	1-32767	Short, **Word**	Read/Write
Drum Count Preset	DCP<drum] .<step>	drum 1-32767 step 1-16	Short, **Word**	Read/Write

*For more information, refer to **0/1-Based Bit Addressing Option**.

The actual number of addresses available for of each type is dependent on the configuration of the PLC. If at runtime the driver finds that an address is not present in the device, the driver will post an error message and remove the tag from its scan list.

V Memory Access as Arrays

The Simatic / TI 505 Serial Driver supports access to V memory in an array. The size of the array for Transparent Byte mode is limited to 100 V memory registers per array. When accessing large arrays, multiple read/write commands are used to access consecutive V memory addresses and may require additional time to process. To access V memory as an array, array notation must be used when entering an address. Array notation is shown in the following examples:

V100 [10]	This denotes an array starting at V100 with a length of 10. This means that the array contains V100, V101, V102, V103, V104, V105, V106, V107, V108, and V109.
V100 [3] [3]	This denotes a two dimensional array starting at V100 and containing V100, V101, V102, V103, V104, V105, V106, V107, and V108 in a 3 by 3 array.

Note: Arrays can be either the Word or SHORT data type, with a default of Word.

Status Words

For all Status Words, Bit 1 is the Most Significant Bit (MSB) and Bit 16 is the Least Significant Bit (LSB) in order from left to right. Users can configure bit addressing to be 0-15 or 1-16 addressing, and can also set the driver to use 1 or 15 as the MSB (thus changing the bit referencing). For more information, refer to **Addressing Options**.

Note: Only addresses that are not used by the controller can be written to.

Register	Description	CP525	CP545	CP565	CTI 2500
STW00001	Non-Fatal Errors. Bit 4, 1 = Password has been entered. Bit 5, 1 = Password has been entered and disabled. Bit 6, 1 = User program error.* Bit 7, 1 = Subroutine stack overflow. Bit 8, 1 = Time of day clock failure. Bit 10 = Special function module communication error. Bit 11, 1 = Previous RLL instruction failed. Bit 12, 1 = I/O module failure or I/O module configuration mismatch. Bit 13, 1 = Communication port failure. Bit 14, 1 = Scan overrun. Bit 15, 1 = Battery low.	x	x	x	x
STW00002	Base Control Status. Each bit reflects the status of a single base. 0 = Status is good. 1 = Base is not present or has a problem. Bit 1, Base 15 Bit 2, Base 14 Bit 3, Base 13 Bit 4, Base 12 Bit 5, Base 11 Bit 6, Base 10 Bit 7, Base 9 Bit 8, Base 8 Bit 9, Base 7 Bit 10, Base 6 Bit 11, Base 5 Bit 12, Base 4 Bit 13, Base 3 Bit 14, Base 2 Bit 15, Base 1 Bit 16, Base 0	x	x	x	x
STW00003-00009	Profibus DP Slave Status. Each bit is 0 if a slave is present, or 1 if the slave is missing or failed. STW03 Bit 1-16 slave addresses 16-1. STW04 Bit 1-16 slave addresses 32-17. STW05 Bit 1-16 slave addresses 48-33. STW06 Bit 1-16 slave addresses 64-49. STW07 Bit 1-16 slave addresses 80-65. STW08 Bit 1-16 slave addresses 96-81. STW09 Bit 1-16 slave addresses 112-97.	x	x	x	x

Register	Description	CP525	CP545	CP565	CTI 2500
STW00010	Contains the value of the previous dynamic scan time.	x	x	x	x
STW00011	Indicates the status of the modules in the Local Base (Base 0). Each bit indicates a module in a slot. 0 = Good. 1 = Module not present or failed. Bit 1 - 16 = Module 16 - 1.	x	x	x	x
STW00012-00026	Status bits for modules in Bases 1 - 15, STW012 = Base 1... Bit range is the same as for STW011.	x	x	x	x
STW00027-00138	This range of Status Words apply to the Profibus DP Slave channels if present. STW027 is slave 1...STW138 is slave 112. Bit range is the same as for STW011.	x	x	x	x
STW00139	This Status Word provides a count of the discrete points (X/Y or C) that are currently forced.	x	x	x	x
STW00140	This Status Word provides a count of the word points (WX/WY) that are currently forced.	x	x	x	x
STW00141-00144	Date, Time, and Day of Week.	N/A	N/A	N/A	N/A
STW141	Bit 1-4, Year tens digit. Bit 5-8, Year units digit. Bit 9-12, Month tens digit. Bits 13-16, Month units.	x	x	x	x
STW142	Bit 1-4, Day - Tens digit. Bit 5-8, Day - Units digit. Bit 9-12, Hour - Tens digit. Bit 13-16, Hour - Units digit.	x	x	x	x
STW143	Bit 1-4, Minute - Tens digit. Bit 5-8, Minute - Units digit. Bit 9-12, Seconds - Tens digit. Bit 13-16, Seconds - Units digit.	x	x	x	x
STW144	Bit 1-4, Seconds - Tenths digit. Bit 5-8, Seconds - Hundredths digit. Bit 9-12, Not used - Always 0. Bit 13-16, Day of the week.	x	x	x	x
STW00145	Receive Error Counts.	x	x	x	x
STW00146	Timeout Counts.	x	x	x	x
STW00147	This Status Word records the number of times that the Profibus-DP Slaves have failed to respond to a request from the Series 505 or CTI 2500 CPU since the most recent restart.	x	x	x	x

Register	Description	CP525	CP545	CP565	CTI 2500
STW00148	This Status Word records the number of times that the Profibus-DP I/O channel has experienced a loss of token since the most recent restart.	x	x	x	x
STW00149-00160	Reserved.	N/A	N/A	N/A	N/A
STW00161	Special Function Processor Fatal Error. Bit 1, 1 = ROM error. Bit 2, 1 = RAM error. Bit 3, 1 = Operating System error. Bit 4, 1 = Invalid control block encountered. Bit 5, 1 = Diagnostic failure. Bit 7, 1 = S Memory is inconsistent. Bit 8 = Special function program received from RLL is invalid.	x	x	x	x
STW00162	Special Function Processor Non-Fatal Errors. Bit 1, 1 = Port 1 communication error.** Bit 3, 1 = Port overrun error. Bit 4, 1 = Analog alarm overrun error. Bit 5, 1 = Cyclic special function programs overrun error. Bit 6, 1 = Normal special function program queue is full. Bit 7, 1 = Priority special function program queue is full. Bit 8, 1 = Cyclic special function program queue is full. Bit 9, 1 = Loop calculation error. Bit 10, 1 = Analog alarm calculation error. Bit 11, 1 = Control block disabled. Bit 12, 1 = Attempt to execute undefined special function program or subroutine. Bit 13, 1 = Attempt to invoke restricted special function program or subroutine.	x	x	x	x
STW00163	Contains the number of the ladder subroutine that caused the stack overflow.	x	x	x	x
STW00164-00165	Contains the source RLL checksum (32 Bit integer).	x	x	x	x
STW00166-00167	Contains the compiled RLL checksum (32 Bit integer).	x	x	x	x
STW00168	Dual RBC Status. Bit 1-16 are bases 15-0. For each Bit: 0 = Dual RBC present and good. 1 = Error or single RBC.	x	x	x	x
STW00169-00175	Not used.	x	x	x	x
'STW00176	Dual Power Supply Status.	x	x	x	x

Register	Description	CP525	CP545	CP565	CTI 2500
	Bit 1-16 are bases 15-0. For each Bit: 0 = Dual power supply present and good. 1 = Error or single power supply.				
STW00177-00183	Not used.	x	x	x	x
STW00184	Module Mismatch Indicator. Bit 1, 1 = Module mismatch error. Bit 5-8 = Indicates the number of the base with the error.	x	x	x	x
STW00185-00190	Not used.	x	x	x	x
STW00191	Serial Port Print Status.	N/A,	N/A	N/A	x
STW00192	Discrete Execution Scan Time - The time spent on the last scan.	x	x	x	x
STW00193-199	Not used.	x	x	x	x
STW00200	User Program Error Cause (associated with Bit 6 of STW001). Codes are as follows: 0 = No error. 1 = Reference to an application that is not installed.*** 2 = Attempted to unlock a flag that is not held by an application.*** 3 = Mismatched lock/unlock instructions.*** 4 = Subroutine nesting level exceeded. 5 = Table overflow. 6 = Attempted to call a non-existent subroutine. 7 = VMEbus access failed due to a bus error.*** 8 = Special function program has not been compiled or does not exist. 9 = Special function program has been disabled. 10= Special function program type is restricted or cyclic. 11 = Special function program or subroutine is being edited. 12 = Special function program or subroutine is being executed by an interrupt task. 13 = User-scheduled fast loop is not configured. 14 = User-scheduled fast loop is disabled.	x	x	x	x
STW00201	First Scan Flags. Bit 1, 1 = First Run Mode scan or single scan after compile. Bit 2, 1 = First Run Mode scan or single scan after Program Mode.	x	x	x	x

Register	Description	CP525	CP545	CP565	CTI 2500
	Bit 3, 1 = First Run Mode scan after transition from Hold Mode. Bit 9, 1 = First scan after battery bad power-up restart. Bit 10, 1 = First scan after battery good power-up restart. Bit 11, 1 = First scan after compile restart. Bit 12, 1 = First scan after partial restart.				
STW00202-00205	Not used.	x	x	x	x
STW00206-00207	U-Memory Checksum C0 (32 bit integer).	x	x	x	N/A
STW00208-00209	U-Memory Checksum C1 (32 bit integer).	x	x	x	N/A
STW00210	Base Poll Enabled Flags. Bit 1-16 are bases 15-0. For each Bit: 0 = Base cannot be polled. 1 = Base can be polled.	x	x	x	x
STW00211-00217	Profibus Poll Enable Flags. Each bit is 1 if the slave is defined and enabled. STW211 Bit 1-16 slave addresses 16-1. STW212 Bit 1-16 slave addresses 32-17. STW213 Bit 1-16 slave addresses 48-33. STW214 Bit 1-16 slave addresses 64-49. STW215 Bit 1-16 slave addresses 80-65. STW216 Bit 1-16 slave addresses 96-81. STW217 Bit 1-16 slave addresses 112-97.	x	x	x	x
STW00218	Not used.	x	x	x	x
STW00219	RLL Task Overrun. Bit 1, Task 1: 0 = Good, 1 = Task scan cycle overrun. Bit 2, Task 2: 0 = Good, 1 = Task scan cycle overrun.	x	x	x	N/A
STW00220	Interrupting Slots in Local Base. Bit 1-16 are slots 16-1. For each Bit: 1 = Interrupt request active at module located in this slot.	x	x	N/A	N/A
STW00221	Module Interrupt Request Count.	x	x	N/A	N/A
STW00222	Spurious Interrupt Count.	N/A	N/A	x	N/A
STW00223-00224	Binary Time of Day (32 bit integer).	x	x	x	x
STW00225	Binary Relative Day (with 1/1/1984 being day 0).	x	x	x	x
STW00226	Time of Day Status.	x	x	x	x

Register	Description	CP525	CP545	CP565	CTI 2500
	Bit 1, 1 = Current time is prior to the time reported in the last task 1 RLL scan. Bit 2-9, Reserved. Bit 10, 1 = Time is valid. Bit 11, 1 = Time synchronization is over a network. Bit 12-13, Time Resolution. 00 = .001 second. 01 = .01 second. 02 = .1 second. 03 = 1 second. Bit 14, 1 = Time synchronization error. Bit 15, 1 = No time synchronization input for the time transmitter.				
STW00227-00228	Bus Error Access Address.	N/A	N/A	x	N/A
STW00229-00230	Bus Error Program Offset.	N/A	N/A	x	N/A
STW0231	Profibus DP I/O Status. Bit 1, 1 = DP in operate state. Bit 2, 1 = DP in clear state. Bit 3, 1 = Error: Unable to download configuration to the Profibus interface. Bit 4, 1 = Error: Unable to retrieve slave diagnostics from the interface. Bit 5, 1 = DP bus error. Bit 16, 1 = DP I/O bus system in not configured.	x	x	x	x
STW00232-00238	Profibus I/O Diagnostics Status. Each bit is 1 if the slave signals a diagnostic that has not been read by a RSD RLL instruction. STW232 Bit 1-16 slave addresses 16-1. STW233 Bit 1-16 slave addresses 32-17. STW234 Bit 1-16 slave addresses 48-33. STW235 Bit 1-16 slave addresses 64-49. STW236 Bit 1-16 slave addresses 80-65. STW237 Bit 1-16 slave addresses 96-81. STW238 Bit 1-16 slave addresses 112-97.	x	x	x	x
STW00239-00240	Source Special Function Program/Subroutine Checksum.	x	x	x	x
STW00241-00242	Compiled Special Function Program/Subroutine Checksum.	x	x	x	x
STW00243	Reserved.	N/A	N/A	N/A	x
STW00244	Additional Control Status Flags.	N/A	N/A	N/A	x

Register	Description	CP525	CP545	CP565	CTI 2500
	Bit 1, Controller Mode 0 = Program Mode, 1 = Run Mode. Bit 2, Scan Mode 0 = Variable, 1 = Fixed. Bit 3, User Program Source 0 = Ram, 1 = Flash. Bit 4, Ethernet Port Link Status 1 = Connected. Bit 5, TCP/IP Network Status 1 = Operational. Bit 6, Duplicate IP Address Status 1 = Duplicate Detected.				
STW00245	Additional Controller Error Status. Bit 1, 1 = Fatal error present. Bit 2, Reserved. Bit 3, 1 = One or more remote bases are not communicating.	N/A	N/A	N/A	x
STW00246	Fatal Error Code. This contains the fatal error code when a fatal error is present.	N/A	N/A	N/A	x
STW00247-00257	CTI Support Diagnostics.	N/A	N/A	N/A	x
STW00259	Product Serial Number.	N/A	N/A	N/A	x
STW00260	Firmware Major Release Number.	N/A	N/A	N/A	x
STW00261	Firmware Minor Release Number.	N/A	N/A	N/A	x
STW00262-00298	CTI Support Diagnostics.	N/A	N/A	N/A	x
STW00299	Peak Scan Time.	N/A	N/A	N/A	x
STW00300-454	CTI Support Statistics.	N/A	N/A	N/A	x
STW00455-00469	Remote Base Receive Errors. This contains the number of times that the controller encountered an error reading the response message from the remote base. STW 455 corresponds to remote base 1. STW 456 – STW 469 correspond to remote bases 2 – 15.	N/A	N/A	N/A	x
STW00470	Not used.	N/A	N/A	N/A	x
STW00471-00485	Abnormal Logoff Count – Remote Base 1 - 15. This contains the number of times that the controller stopped communicating with the remote base due to communications errors or response timeouts. STW 471 corresponds to remote base 1. STW 472 – STW 485 correspond to remote bases 2 – 15.	N/A	N/A	N/A	x
STW00486	Not used.	N/A	N/A	N/A	x
STW00487-00501	Timeout Count – Remote Base 1 – 15. This contains the number of times that the base failed to respond to a request from the controller within the specified time. STW 487 corresponds to remote base 1. STW 488 – STW 501 correspond to remote bases 2 – 15.	N/A	N/A	N/A	x

*For more information, refer to the register "STW200".
**Not used by the CTI 2500.
***This is only for CP575.

Error Descriptions

The following error/warning messages may be generated. Click on the link for a description of the message.

Address Validation

Missing address
Device address '<address>' contains a syntax error
Address '<address>' is out of range for the specified device or register
Data Type '<type>' is not valid for device address '<address>'
Device address '<address>' is Read Only
Array size is out of range for address '<address>'
Array support is not available for the specified address: '<address>'

Serial Communications

COMn does not exist
Error opening COMn
COMn is in use by another application
Unable to set comm properties on COMn
Communications error on '<channel name>' [<error mask>]

Device Status Messages

Device '<device name>' is not responding
Unable to write to '<address>' on device '<device name>'

Device Specific Messages

Unable to write tag '<address>' for device '<device name> : Error code <code>
Unable to read block starting at '<address>' for device '<device name>' : Error code <code>
Bad address in block [<start address> to <end address>] on device '<device name>'

See Also *Error Codes*

Missing address

Error Type:

Warning

Possible Cause:

A tag address that has been specified dynamically has no length.

Solution:

Re-enter the address in the client application.

Device address '<address>' contains a syntax error

Error Type:

Warning

Possible Cause:
A tag address that has been specified dynamically contains one or more invalid characters.

Solution:
Re-enter the address in the client application.

Address '<address>' is out of range for the specified device or register

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically references a location that is beyond the range of supported locations for the device.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Data Type '<type>' is not valid for device address '<address>'

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically has been assigned an invalid data type.

Solution:
Modify the requested data type in the client application.

Device address '<address>' is Read Only

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically has a requested access mode that is not compatible with what the device supports for that address.

Solution:
Change the access mode in the client application.

Array size is out of range for address '<address>'

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically is requesting an array size that is too large for the address type or block size of the driver.

Solution:

Re-enter the address in the client application to specify a smaller value for the array or a different starting point.

Array support is not available for the specified address: '<address>'

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically contains an array reference for an address type that doesn't support arrays.

Solution:
Re-enter the address in the client application to remove the array reference or correct the address type.

COMn does not exist

Error Type:
Fatal

Possible Cause:
The specified COM port is not present on the target computer.

Solution:
Verify that the proper COM port has been selected.

Error opening COMn

Error Type:
Fatal

Possible Cause:
The specified COM port could not be opened due an internal hardware or software problem on the target computer.

Solution:
Verify that the COM port is functional and may be accessed by other Windows applications.

COMn is in use by another application

Error Type:
Fatal

Possible Cause:
The serial port assigned to a device is being used by another application.

Solution:
Verify that the correct port has been assigned to the channel.

Unable to set comm properties on COMn

Error Type:
Fatal

Possible Cause:
The serial properties for the specified COM port are not valid.

Solution:
Verify the serial properties and make any necessary changes.

Communications error on '<channel name>' [<error mask>]

Error Type:
Serious

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.

Possible Cause:

1. The serial connection between the device and the host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the device.
2. Verify that the specified communications properties match those of the device.

Device '<device name>' is not responding

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the host PC is broken.
2. The response from the device took longer to receive than the amount of time specified in the "Request Timeout" device property.
3. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the device.
2. Increase the Request Timeout property so that the entire response can be handled.
3. Verify the baud rate selected matches that of the device.
4. If NITP protocol is selected, ensure that odd parity and 7 data bits are selected.
5. If TB protocol is selected, ensure that no parity and 8 data bits are selected.
6. Make sure that RTS_ALWAYS flow control is selected.

Unable to write to '<address>' on device '<device name>'

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the host PC is broken.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the device.
2. Verify that the specified communications properties match those of the device.

Unable to write tag <address> for device <device name>: Error code <code>

Error Type:
Serious

Possible Cause:

1. The address does not exist in the device.
2. The location is read only in the device.
3. The device could not perform the write operation.

Solution:
Refer to the list of error codes.

***See Also**: Error Codes*

Unable to read block starting at <address> for device <device name>: Error code <code>

Error Type:
Serious

Possible Cause:

1. The address does not exist in the device.
2. The device could not perform the read operation.

Solution:
Refer to the list of error codes.

***See Also**: Error Codes*

Bad address in block [<start address> to <end address>] on device '<device name>'

Error Type:
Serious

Possible Cause:
An attempt has been made to reference a nonexistent location in the specified device.

Solution:
Verify the tags assigned to addresses in the specified range on the device and eliminate ones that reference invalid locations.

Error Codes

Error Code	Description
2	Address out of range
3	Requested data not found
4	Illegal task code requested
7	Fatal error detected
9	Incorrect amount of data sent with request
10	Illegal request in current operational mode
12	Attempted write operation did not verify
13	Illegal number of ASCII characters received (NITP)
15	Data not inserted
16	Data not written
17	Invalid data sent with command
19	The store and forward buffer is busy
22	Attempted to write to a protected variable
28	Processor busy - Cannot complete the requested operation

Note: If the error code returned is 2, the driver will remove the tags in the block from its scan list.

Resources

In addition to this user manual, there are a variety of resources available to assist customers, answer questions, provide more detail about specific implementations, or help with troubleshooting specific issues.

Knowledge Base
Whitepapers
Connectivity Guides
Technical Notes
Training Programs
Training Videos
Kepware Technical Support
PTC Technical Support

Index

A

B

C

D

E

F

I

L

M

N

O

P

R

S

T

U

W