Allen-Bradley Ethernet Driver

Table of Contents

Welcome to the Allen-Bradley Ethernet Driver Help Center

This help center is the user documentation for Kepware Allen-Bradley Ethernet Driver. This help center is updated regularly to reflect the latest functionality and information.

Version 1.062

Overview

The Allen-Bradley Ethernet Driver provides a reliable way to connect Allen-Bradley Ethernet devices to client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. This driver supports the Allen Bradley SLC 5/05 series, PLC-5 series, and SoftPLC PLCs. Address ranges are open to support future models for this series of PLCs.

Setup

Communication Protocol

Allen-Bradley Ethernet

Supported Devices

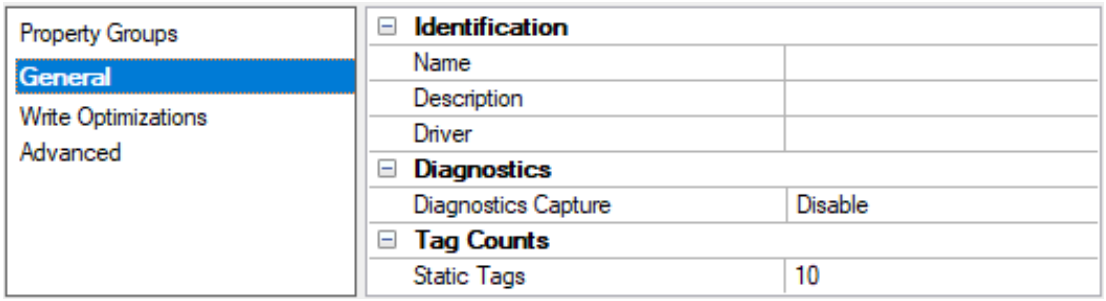
SLC 5/05 processor*
PLC-5 series (excluding the PLC-5/250 series)
SoftPLC
*Address ranges are open in the driver to allow for new devices. The driver may support a device even if it is not listed above.

Channel and Device Limits

The maximum number of channels supported by this driver is 256. The maximum number of devices supported by this driver is 1024 per channel.

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. Specify 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. 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

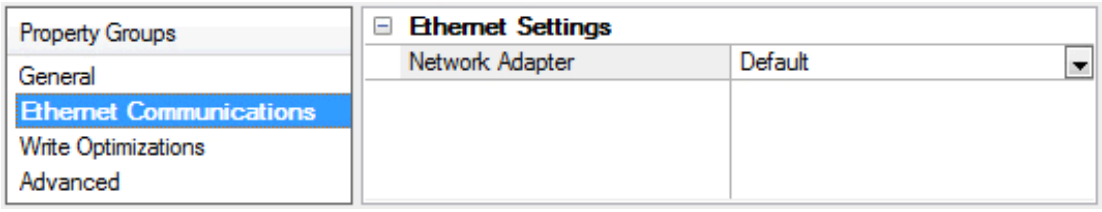
Diagnostics Capture: When enabled, this option makes the channel's diagnostic information available to OPC applications. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.
Note: This property is not available if the driver or operating system does not support diagnostics.
For more information, refer to Communication Diagnostics and Statistics Tags in server help.

Tag Counts

Static Tags: Provides the total number of defined static tags at this level (device or channel). This information can be helpful in troubleshooting and load balancing.

Channel Properties – Ethernet Communications

Ethernet Communication can be used to communicate with devices.

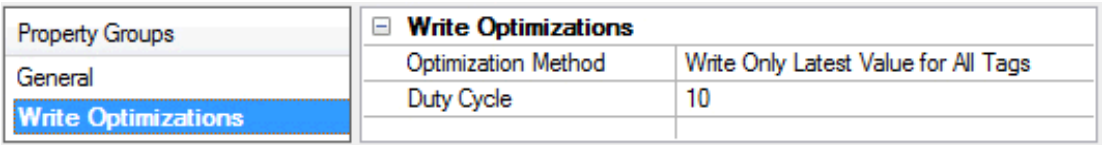


Ethernet Settings

Network Adapter: Specify the network adapter to bind. When left blank or Default is selected, the operating system selects the default adapter.

Channel Properties – Write Optimizations

The server must ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties to meet specific needs or improve application responsiveness.



Write Optimizations

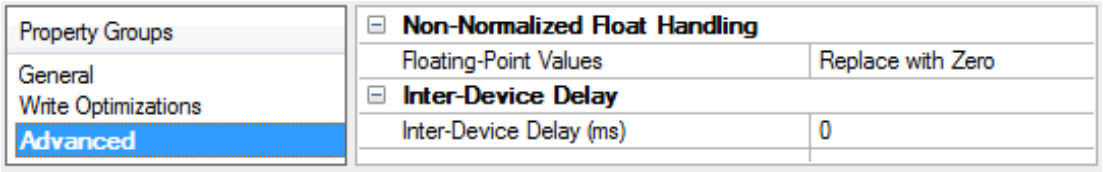
Optimization Method: Controls how write data is passed to the underlying communications driver. The options are:

- **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.

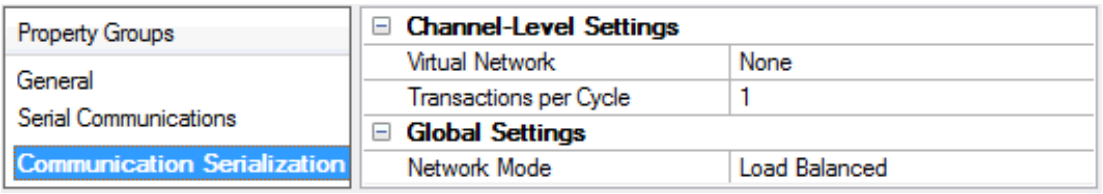
Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

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

Channel Properties – Communication Serialization

The server's multi-threading architecture allows channels to communicate with devices in parallel. Although this is efficient, communication can be serialized in cases with physical network restrictions (such as Ethernet radios). Communication serialization limits communication to one channel at a time within a virtual network.

The term "virtual network" describes a collection of channels and associated devices that use the same pipeline for communications. For example, the pipeline of an Ethernet radio is the client radio. All channels using the same client radio are associated with the same virtual network. Channels are allowed to communicate each in turn, in a "round-robin" manner. By default, a channel can process one transaction before handing communications off to another channel. A transaction can include one or more tags. If the controlling channel contains a device that is not responding to a request, the channel cannot release control until the transaction times out. This results in data update delays for the other channels in the virtual network.



Channel-Level Settings

Virtual Network: Specify the channel's mode of communication serialization. Options include None and Network 1 - Network 500. The default is None. Descriptions of the options are as follows:

- **None**: This option disables communication serialization for the channel.
- **Network 1 - Network 500**: This option specifies the virtual network to which the channel is assigned.

Transactions per Cycle: Specify the number of single blocked/non-blocked read/write transactions that can occur on the channel. When a channel is given the opportunity to communicate, this is the number of transactions attempted. The valid range is 1 to 99. The default is 1.

Global Settings

Network Mode: This property is used to control how channel communication is delegated. In **Load Balanced** mode, each channel is given the opportunity to communicate in turn, one at a time. In **Priority** mode, channels are given the opportunity to communicate according to the following rules (highest to lowest priority):

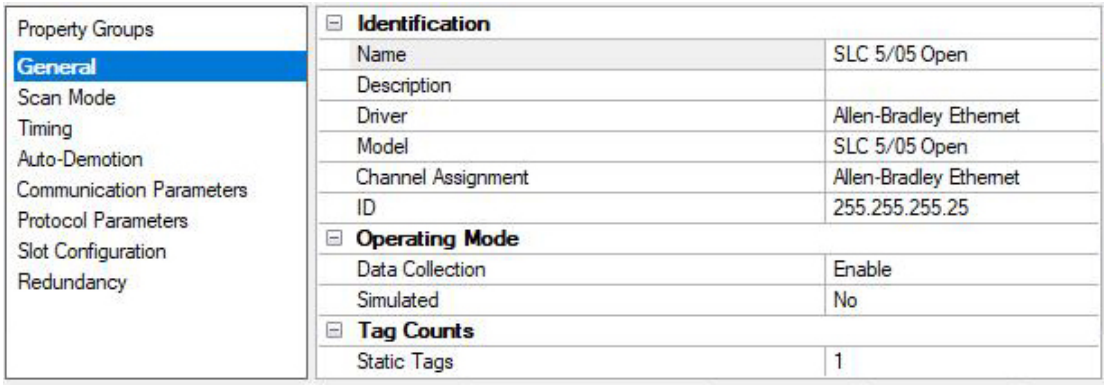
1. Channels with pending writes have the highest priority.
2. Channels with pending explicit reads (through internal plug-ins or external client interfaces) are prioritized based on the read's priority.
3. Scanned reads and other periodic events (driver specific).

The default is Load Balanced and affects *all* virtual networks and channels.
Devices that rely on unsolicited responses should not be placed in a virtual network. In situations where communications must be serialized, it is recommended that Auto-Demotion be enabled.

Due to differences in the way that drivers read and write data (such as in single, blocked, or non-blocked transactions); the application's Transactions per cycle property may need to be adjusted. When doing so, consider the following factors:

- How many tags must be read from each channel?
- How often is data written to each channel?
- Is the channel using a serial or Ethernet driver?
- Does the driver read tags in separate requests, or are multiple tags read in a block?
- Have the device's Timing properties (such as Request timeout and Fail after *x* successive timeouts) been optimized for the virtual network's communication medium?

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.

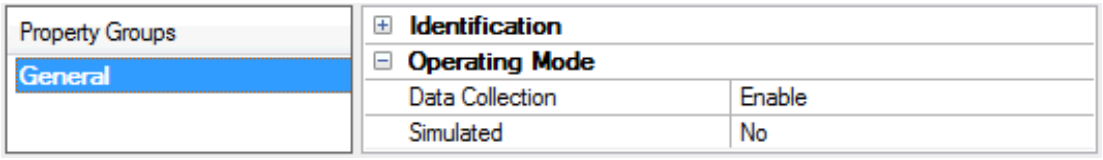
Driver: Selected protocol driver for this device.

Model: The specific version of the device.

ID: The device ID is the network address of the PLC.

See Also: Operating Mode

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: Place the device into or out of 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.

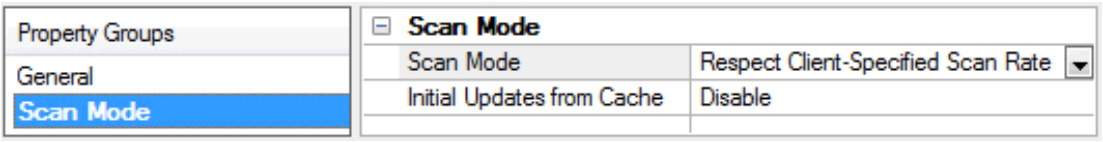
Notes:

1. Updates are not applied until clients disconnect and reconnect.
2. The 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.
3. 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.
4. When a device is simulated, updates may not appear faster than one (1) second in the client.

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:

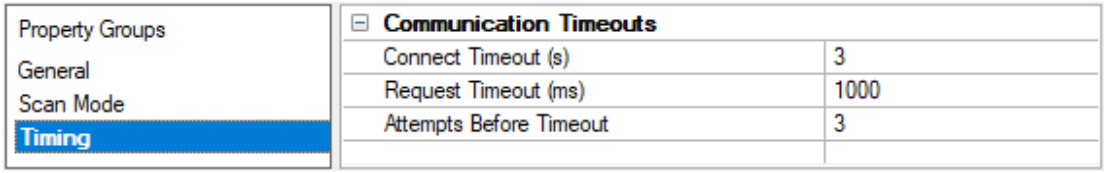
- **Respect Client-Specified Scan Rate**: This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate**: This mode specifies the value set as the maximum scan rate. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  Note: When the server 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 OPC 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.

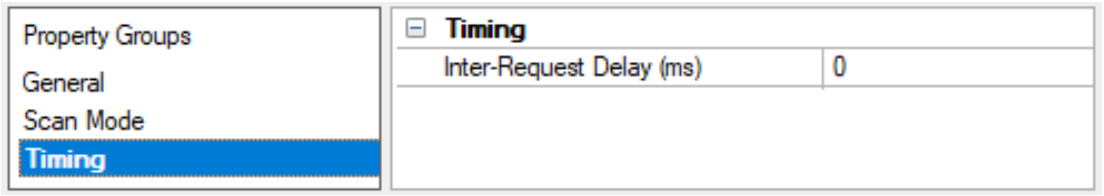
Request Timeout: Specify an interval used by all drivers to determine how long the driver waits for a response from the target device to complete. The valid range is 50 to 9999999 milliseconds (167 minutes). The default is usually 1000 milliseconds, but can vary depending on the driver. The default timeout 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

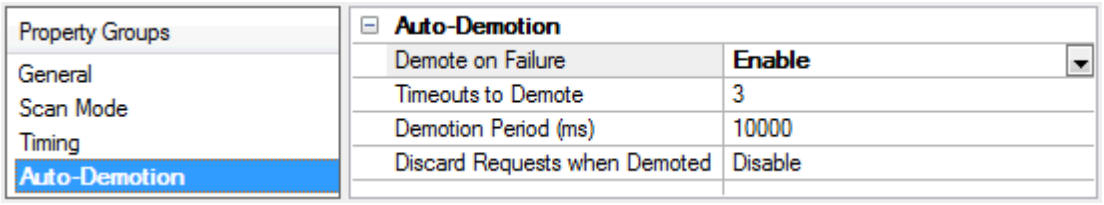
Inter-Request Delay: Specify how long the driver waits before sending the next request to the target device after receiving the response to the previous request. It overrides the normal polling frequency of tags associated with the device, as well as one-time reads and writes. This delay can be useful 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.



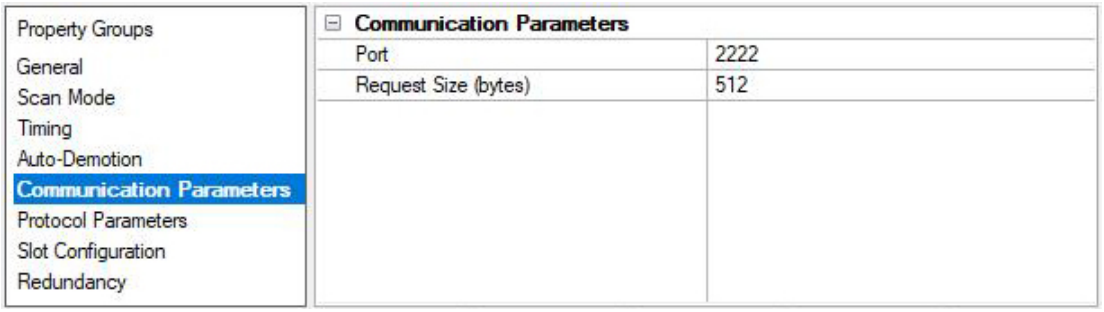
Demote on Failure: When enabled, the device is automatically taken off-scan until it is responding again.
Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

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 – Communications Parameters

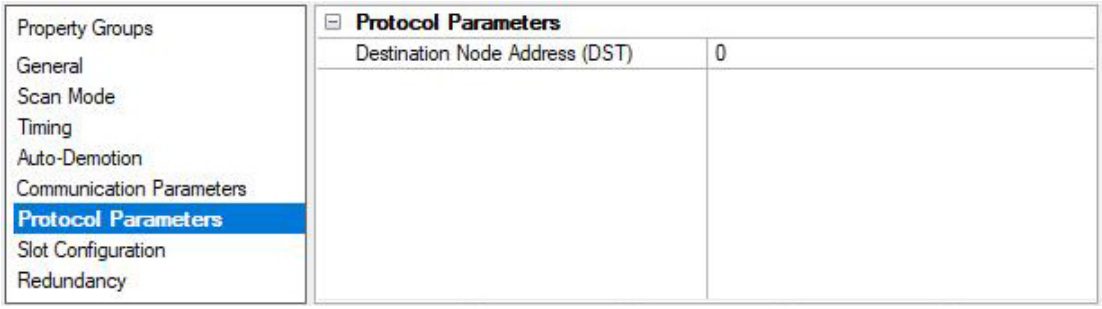


Port: Specify the port number that the remote device is configured to use. The default setting is 2222.

Request Size: Specify the maximum number of bytes that may be requested from a device at one time. To refine the driver's performance, configure the request size to one of the following settings: 32, 64, 128, 256, 512, 1024, or 2000 bytes. The default is 512 bytes.

Tip: For Boolean arrays, the block size is the bit equivalent (or, block size multiplied by 8). For example, a block size of 512 bytes is equal to 512 * 8 = 4096 bits.

Device Properties – Protocol Parameters

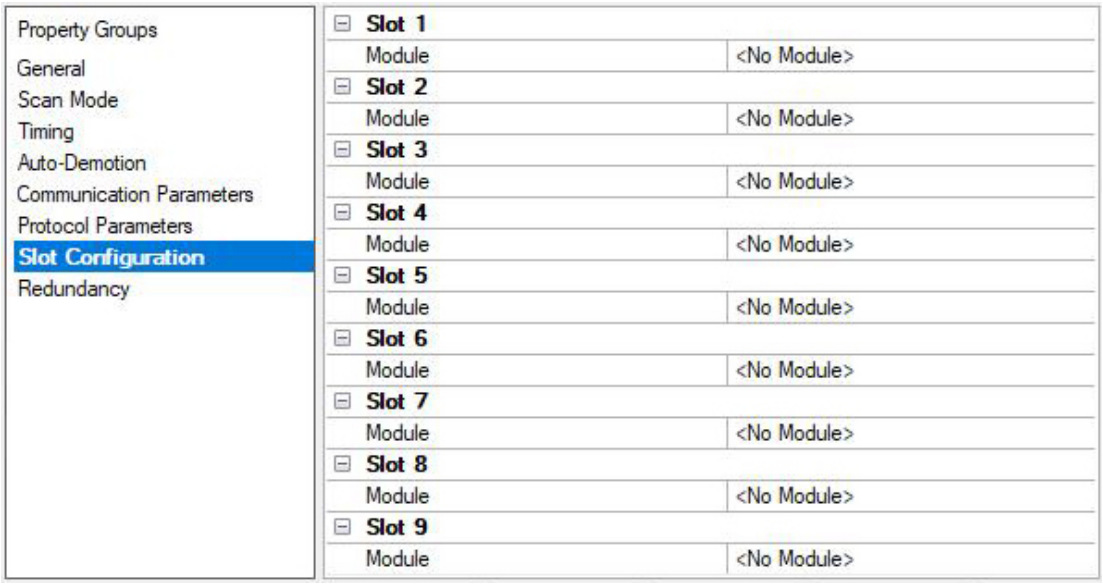


Destination Node Address (DST): Specify the destination node address. For DF1 gateway applications, select the node address of the destination device. For non-DF1 gateway applications, leave the node address at the default setting of 0.

Note: The destination device is the DH+ or DH-485 device.

Device Properties – Slot Configuration

SLC500 models (with modular I/O racks) must be configured for use with this driver if the I/O is to be accessed by the driver. Up to 30 slots can be configured per device.



To use the slot configuration:

1. Select the slot to be configured by clicking on the row in the module list box.
2. To select a module, click on it from the available modules drop-down list.
3. Configure the Input Words and Output Words if necessary.
4. To remove a slot / module, select **No Module** from the available modules drop-down list.
5. When complete, click **OK**.

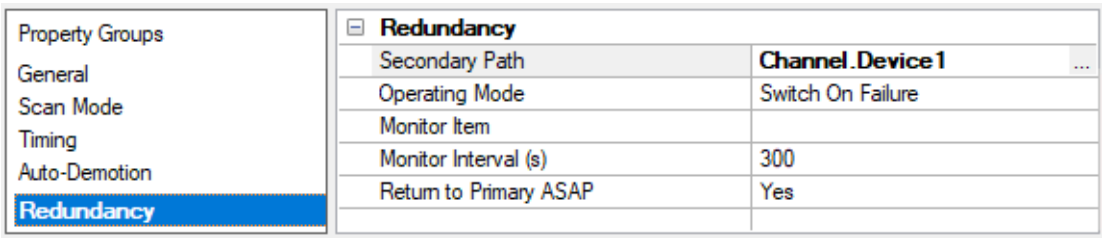
Tip: Use the 0000-Generic Module to configure I/O that is not contained in the list of Available Modules.

Note: It is common to have open slots in the rack that do not contain a physical module. To correctly access data for the various slots that do contain a module, the preceding module(s) must have the correct number of words mapped. For example, if only interested in the I/O in slot 3, but slots 1 and 2 contain I/O modules, the correct modules must be selected for slots 1, 2, and 3 from this slot configuration group.

0000-Generic Module
Use the Generic Module to map Input and Output words for modules that are not represented in the list of available modules. To correctly use the Generic Module, users must know the number of Input and Output words required for each module.

- *Consult Allen-Bradley I/O user manual documentation to confirm Input and Output requirements and be aware that requirements may be different based on Class 1 or Class 3 operation.*
- *For information on the number of input and output words available for each I/O module, refer to Modular I/O Selection Guide.*

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.*

Modular I/O Selection Guide

The following table lists the number of input and output words available for each I/O module in the Slot Configuration list.

Tip: Use the Generic Module to map input and output words for modules that are not represented in the list of available modules. The range of accepted values is shown in the table below.

Consult the Allen-Bradley user manual for the specific I/O module to configure to confirm input and output requirements. Requirements may be different based on Class 1 or Class 3 operation.

Module Type	Input Words	Output Words
0000-Generic Module	0-255	0-255
1203-SM1 SCANport Comm Module - Basic	8	8
1203-SM1 SCANport Comm Module - Enhanced	32	32
1394-SJT GMC Turbo System	32	32
1746-BAS Basic Module 500 5/01 Configuration	8	8
1746-BAS Basic Module 5/02 Configuration	8	8
1746-HS Single Axis Motion Controller	4	4
1746-HSCE High-Speed Counter/Encoder	8	1
1746-HSRV Motion Control Module	12	8
1746-HSTP1 Stepper Controller Module	8	8
1746-I*16 Any 16 pt Discrete Input Module	1	0
1746-I*32 Any 32 pt Discrete Input Module	2	0
1746-I*8 Any 8 pt Discrete Input Module	1	0
1746-IA16 16 Input 100/120 VAC	1	0
1746-IA4 4 Input 100/120 VAC	1	0
1746-IA8 8 Input 100/120 VAC	1	0
1746-IB16 16 Input (Sink) 24 VDC	1	0
1746-IB32 32 Input (Sink) 24 VDC	2	0
1746-IB8 8 Input (Sink) 24 VDC	1	0
1746-IC16 16 Input (Sink) 48 VDC	1	0
1746-IG16 16 Input [TTL] (Source) 5 VDC	1	0
1746-IH16 16 Input [Trans] (Sink) 125 VDC	1	0
1746-IM16 16 Input 200/240 VAC	1	0
1746-IM4 4 Input 200/240 VAC	1	0
1746-IM8 8 Input 200/240 VAC	1	0
1746-IN16 16 Input 24 VAC/VDC	1	0
1746-INI4I Analog 4 Ch. Isol. Current Input	8	8
1746-INI4VI Analog 4 Ch. Isol. Volt./Current Input	8	8
1746-INO4I Analog 4 Ch. Isol. Current Input	8	8
1746-INO4VI Analog 4 Ch. Isol. Volt./Current Input	8	8
1746-INT4 4 Ch. Isolated Thermocouple Input	8	8
1746-IO12 6 In 100/120 VAC 6 Out [Rly] VAC/VDC	1	1
1746-IO12DC 6 Input 12 VDC, 6 Output [Rly	1	1
1746-IO4 2 In 100/120 VAC 2 Out [Rly] VAC/VDC3	1	1
1746-IO8 4 In 100/120 VAC 4 Out [Rly] VAC/VDC4	1	1
1746-ITB16 16 Input [Fast] (Sink) 24 VDC	1	0
1746-ITV16 16 Input [Fast] (Source) 24 VDC	1	0

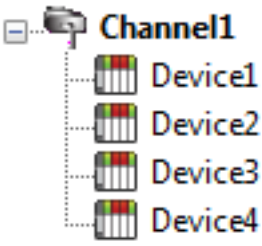
Module Type	Input Words	Output Words
1746-IV16 16 Input (Source) 24 VDC	1	0
1746-IV32 32 Input (Source) 24 VDC	2	0
1746-IV8 8 Input (Source) 24 VDC	1	0
1746-NI4 4 Ch Analog Input	4	0
1746-NI8 8 Ch Analog Input, Class 1	8	8
1746-NI8 8 Ch Analog Input, Class 3	16	12
1746-NIO4I Analog Comb 2 in & 2 Current Out	2	2
1746-NIO4V Analog Comb 2 in & 2 Voltage Out	2	2
1746-NO4I 4 Ch Analog Current Output	0	4
1746-NO4V 4 Ch Analog Voltage Output	0	4
1746-NR4 4 Ch Rtd/Resistance Input Module	8	8
1746-NT4 4 Ch Thermocouple Input Module	8	8
1746-NT8 Analog 8 Ch Thermocouple Input	8	8
1746-O*16 Any 16 pt Discrete Output Module	0	1
1746-O*32 Any 32 pt Discrete Output Module	0	2
1746-O*8 Any 8 pt Discrete Output Module	0	1
1746-OA16 16 Output (Triac) 100/240 VAC	0	1
1746-OA8 8 Output (Triac) 100/240 VAC	0	1
1746-OAP12 12 Output [Triac] 120/240 VDC	0	1
1746-OB16 16 Output [Trans] (Source) 10/50 VDC	0	1
1746-OB16E 16 Output [Trans] (Source) Protected	0	1
1746-OB32 32 Output [Trans] (Source) 10/50 VDC	0	2
1746-OB32E 32 Output [Trans] (Source) 10/50 VDC	0	2
1746-OB6EI 6 Output [Trans] (Source) 24 VDC	0	1
1746-OB8 8 Output [Trans] (Source) 10/50 VDC	0	1
1746-OBP16 16 Output [Trans 1 amp] (SRC) 24 VDC	0	1
1746-OBP8 8 Output [Trans 2 amp] (Source) 24 VDC	0	1
1746-OG16 16 Output [TLL] (SINK) 5 VDC	0	1
1746-OV16 16 Output [Trans] (Sink) 10/50 VDC	0	1
1746-OV32 32 Output [Trans] (Sink) 10/50 VDC	0	2
1746-OV8 8 Output [Trans] (Sink) 10/50 VDC	0	1
1746-OVP16 16 Output [Trans 1 amp] (Sink) 24VDC3	0	1
1746-OW16 16 Output [Relay] VAC/VDC	0	1
1746-OW4 4 Output [Relay] VAC/VDC	0	1
1746-OW8 8 Output [Relay] VAC/VDC	0	1
1746-OX8 8 Output [Isolated Relay] VAC/VDC	0	1
1747-DCM Direct Communication Module (1/2 Rack)	4	4
1747-DCM Direct Communication Module (1/4 Rack)	2	2
1747-DCM Direct Communication Module (3/4 Rack)	6	6
1747-DCM Direct Communication Module (Full Rack)	8	8
1747-DSN Distributed I/O Scanner 30 Blocks	32	32
1747-DSN Distributed I/O Scanner 7 Blocks	8	8
1747-KE Interface Module, Series A	1	0
1747-KE Interface Module, Series B	8	8
1747-MNET MNET Network Comm Module	0	0
1746-QS Synchronized Axes Module	32	32

Module Type	Input Words	Output Words
1747-QV Open Loop Velocity Control	8	8
1747-RCIF Robot Control Interface Module	32	32
1747-SCNR ControlNet SLC Scanner	32	32
1747-SDN DeviceNet Scanner Module	32	32
1747-SN Remote I/O Scanner	32	32
AMCI-1561 AMCI Series 1561 Resolver Module	8	8

Optimizing Communications

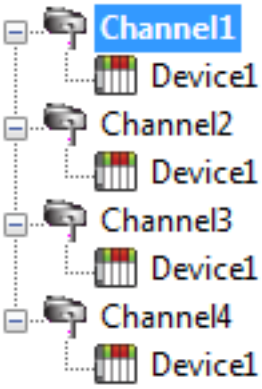
The Allen-Bradley Ethernet Driver is designed to provide the best performance with the least amount of impact on the system's overall performance. While the driver is fast, there are a couple of guidelines that can be used to control and optimize the application and gain maximum performance.

The server refers to a communications protocol like Allen-Bradley Ethernet as a channel. Each channel defined in the application represents a separate path of execution in the server. Once a channel has been defined, a series of devices can be defined under that channel. Each of these devices represents a single Allen-Bradley PLC from which data is collected. While this approach to defining the application provides a high level of performance, it doesn't take full advantage of the Allen-Bradley Ethernet Driver or the network. An example of how the application may appear when configured using a single channel is shown below.



Each device appears under a single Allen-Bradley Ethernet channel. In this configuration, the driver must move from one device to the next as quickly as possible to gather information at an effective rate. As more devices are added or more information is requested from a single device, the overall update rate begins to suffer.

If the Allen-Bradley Ethernet Driver could only define one single channel, the example above would be the only option available; however, the driver can define up to 256 channels. Using multiple channels distributes the data collection workload by simultaneously issuing multiple requests to the network. An example of how the same application may appear when configured using multiple channels to improve performance is shown below.



Each device has is defined under its own channel. In this configuration, a single path of execution is dedicated to the task of gathering data from each device. If the application has 256 or fewer devices, it can be optimized as shown here.

The performance can improve even if the application has more than 256 devices. While 256 or fewer devices may be ideal, the application still benefits from additional channels. Although spreading the device load across all channels causes the server to move from device to device again, it can do so with far fewer devices to process on a single channel.

Data Types Description

Data Type	Description
Boolean	Single bit
Byte	Unsigned 8-bit value
Char	Signed 8-bit value
Word	Unsigned 16-bit value
Short	Signed 16-bit value
DWord	Unsigned 32-bit value
Long	Signed 32-bit value
BCD	Two-byte packed BCD, four decimal digits
LBCD	Four-byte packed BCD, eight decimal digits
Float	32-bit IEEE floating-point
String	Null-terminated character array

Note: The DWord, Long, and LBCD data types are not native to any of the PLC models. When referencing a 16-bit location as a 32-bit value, the location referenced is the low word and the next successive location is the high word. For example, if N7:10 is selected as a DWord data type, N7:10 is the low word and N7:11 the high word.

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.

Model	Output	Input	Status	Binary	Timer	Counter	Control	Integer	Float	ASCII	String	BCD	Long	PID	Message	Block Transfer	Function
SLC5/05	X	X	X	X	X	X	X	X	X	X	X						
PLC5	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	

General Addressing
SLC 5/05 Open Addressing
PLC-5 Family and Soft PLC Addressing

General Addressing

The general addresses below pertain to SLC 5/05, PLC-5, and SoftPLC.

Output Files
Input Files
Status Files
Binary Files
Timer Files
Counter Files
Control Files
Integer Files
Float Files
ASCII Files
String Files

See Also:
SLC 5/05 Open Addressing
PLC-5 Family and SoftPLC Addressing

Output Files

The syntax for accessing data in the output file differs depending on the PLC model. Data locations are read / write for PLC-5 and SoftPLC models and read only for all other models. The default data type for all syntax is shown in **bold**.

PLC-5 and SoftPLC Model Syntax

Syntax	Data Type
O:<word>	Short, **Word**, BCD
O:<word>/<bit>	**Boolean**
O:<word>/<bit>[rows][cols]	**Boolean***
O:<word>/<bit>[cols]	**Boolean***
O/bit	**Boolean**
O/bit[rows][cols]	**Boolean***
O/bit[cols]	**Boolean***

*Array types

Note: Word and bit address information is in octal for PLC-5 and SoftPLC models. This follows the convention of the programming software.

SLC 5/05 Open Models (Modular I/O) Syntax

Syntax	Data Type
O:<slot>	Short, **Word**, BCD
O:<slot>.<word>	Short, **Word**, BCD
O:<slot>/<bit>	**Boolean**
O:<slot>/<bit>[rows][cols]	**Boolean***
O:<slot>/<bit>[cols]	**Boolean***
O:<slot>.<word>/<bit>	**Boolean**
O:<slot>.<word>/<bit>[rows][cols]	**Boolean***
O:<slot>.<word>/<bit>[cols]	**Boolean***

*Array types

Slot and Word Configurations

The following slot and word locations are allowed for each model. For information, refer to Device Setup.

PLC Model	Min Slot	Max. Slot	Max. Word
SLC 5/05 Open	1	30	*
PLC-5 Family	NA	NA	277 (octal)
SoftPLC	NA	NA	777 (octal)

*The number of input or output words available for each I/O module can be found in the Modular I/O Selection Guide.

Examples

All addresses are in octal.

PLC-5 / SoftPLC	Addresses
O:0	Word 0
O:37	Word 31 (37 octal=31 decimal)
O/42	Bit 34 (42 octal=34 decimal)
O:2/2	Bit 2 word 2 (same as O/42)
O/20[9]	9 element Boolean array starting at bit 16 (20 octal=16 decimal)
O/37[8][11]	8 by 11 element Boolean array starting at bit 31 (37 octal=31 decimal)
O:47/5[3]	3 element Boolean array starting at bit 5 word 39 (47 octal=39 decimal)
O:11/13[3][7]	3 by 7 element Boolean array starting at bit 11 (13 octal-l=11 decimal) word 9 (11 octal=9 decimal)

SLC 5/05	Addresses
O:1	Word 0 slot 1
O:1.0	Word 0 slot 1 (same as O:1)
O:12	Word 0 slot 12
O:12.2	Word 2 slot 12
O:4.0/0	Bit 0 word 0 slot 4

SLC 5/05	Addresses
O:4/0	Bit 0 slot 4 (same as O:4.0/0)
O:4.2/0	Bit 0 word 2 slot 4
O:4/32	Bit 32 slot 4 (same as O:4.2/0)
O:2.12/3[17]	17 element Boolean array starting at bit 3 word 12 slot 2
O:2.2/0[12][12]	12 by 12 element Boolean array starting at bit 0 word 2 slot 2
O:2/43[5]	5 element Boolean array starting at bit 43 slot 2
O:2/11[6][12]	6 by 12 element Boolean array starting at bit 11 slot 2

Input Files

The syntax for accessing data in the input file differs depending on the PLC model. Data locations are read / write for PLC-5 models and read only for all other models. The default data type for all syntax is shown in **bold**.

PLC-5 and SoftPLC Model Syntax

Syntax	Data Type
I:<word>	Short, **Word**, BCD
I:<word>/<bit>	**Boolean**
I:<word>/<bit>[rows][cols]	**Boolean***
I:<word>/<bit>[cols]	**Boolean***
I/bit	**Boolean**
I/bit[rows][cols]	**Boolean***
I/bit[cols]	**Boolean***

*Array types

Note: Word and bit address information is in octal for PLC-5 and SoftPLC models. This follows the convention of the programming software.

SLC 5/05 Open Models (Modular I/O) Syntax

Syntax	Data Type
I:<slot>	Short, **Word**, BCD
I:<slot>.<word>	Short, **Word**, BCD
I:<slot>/<bit>	**Boolean**
I:<slot>/<bit>[rows][cols]	**Boolean***
I:<slot>/<bit>[cols]	**Boolean***
I:<slot>.<word>/<bit>	**Boolean**
I:<slot>.<word>/<bit>[rows][cols]	**Boolean***
I:<slot>.<word>/<bit>[cols]	**Boolean***

*Array types

Slot and Word Locations

The following slot and word locations are allowed for each model. For more information, refer to Device Setup.

PLC Model	Min Slot	Max. Slot	Max. Word
SLC 5/05 Open	1	30	*
PLC-5 Family	NA	NA	277 (octal)
SoftPLC Family	NA	NA	777 (octal)

*The number of input or output words available for each I/O module can be found in the Modular I/O Selection Guide.

Examples

All addresses are in octal.

PLC-5 / SoftPLC	Addresses
I:0	Word 0
I:10	Word 8 (10 octal = 8 decimal)
I/20	Bit 16 (20 octal = 16 decimal)
I:1/0	Bit 0 word 1 (same as I/20)
I/20[9]	9 element Boolean array starting at bit 16 (20 octal = 16 decimal)
I/37[8][11]	8 by 11 element Boolean array starting at bit 31 (37 octal = 31 decimal)
I:47/5[3]	3 element Boolean array starting at bit 5 word 39 (47 octal = 39 decimal)
I:11/13[3][7]	3 by 7 element Boolean array starting at bit 11 (13 octal = 11 decimal) word 9 (11 octal = 9 decimal)

SLC 5/05	Addresses
I:1	Word 0 slot 1
I:1.0	Word 0 slot 1 (same as I:1)
I:12	Word 0 slot 12
I:12.2	Word 2 slot 12
I:4.0/0	Bit 0 word 0 slot 4
I:4/0	Bit 0 slot 4 (same as I:4.0/0)
I:4.2/0	Bit 0 word 2 slot 4
I:4/32	Bit 32 slot 4 (same as I:4.2/0)
I:2.12/3[17]	17 element Boolean array starting at bit 3 word 12 slot 2
I:2.2/0[12][12]	12 by 12 element Boolean array starting at bit 0 word 2 slot 2
I:2/43[5]	5 element Boolean array starting at bit 43 slot 2
I:2/11[6][12]	6 by 12 element Boolean array starting at bit 11 slot 2

Status Files

To access Status files, specify a word (and optionally, a bit in the word). The default data type for all syntax is shown in **bold**.

Syntax	Data Type
S:<word>	Short, **Word**, BCD, DWord, Long, LBCD
S:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*
S:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*
S:<word>/<bit>	**Boolean**
S:<word>/<bit> [rows] [cols]	**Boolean***
S:<word>/<bit> [cols]	**Boolean***
S/bit	**Boolean**
S/bit [rows][cols]	**Boolean***
S/bit [cols]	**Boolean***

*Array types

Note: The number of array elements (in bytes) cannot exceed the block request size specified. This means that array size cannot exceed 16 words given a block request size of 32 bytes. For more information, refer to Block Request Size.

Word Locations

The following Word locations are allowed for each model. The maximum word location is one less when accessing as a 32-bit data type (Long, DWord, or Long BCD).

PLC Model	Max. Word
SLC 5/05 Open	999
PLC-5 Family	999
SoftPLC	31

Example	Description
S:0	Word 0
S/26	Bit 26
S:4/15	Bit 15 word 4
S:10 [16]	16 element array starting at word 10
S:0 [4][8]	4 by 8 element array starting at word 0
S/9 [5]	5 element Boolean array starting at bit 9
S/11 [3][7]	3 by 7 element Boolean array starting at bit 11
S:6/1 [6]	6 element Boolean array starting at bit 1 word 6
S:13/5 [2][3]	2 by 3 element Boolean array starting at bit 5 word 13

Binary Files

To access Binary files, specify a file number and a word (and optionally, a bit in the word). The default data type for all syntax is shown in **bold**.

Syntax	Data Type
B<file>:<word>	Short, **Word**, BCD, DWord, Long, LBCD
B<file>:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*
B<file>:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*
B<file>:<word>/<bit>	**Boolean**
B<file>:<word>/<bit> [rows][cols]	**Boolean***
B<file>:<word>/<bit> [cols]	**Boolean***
B<file>/bit	**Boolean**
B<file>/bit [rows][cols]	**Boolean***
B<file>/bit [cols]	**Boolean***

*Array types

Note: The number of array elements (in bytes) cannot exceed the block request size specified. This means that array size cannot exceed 16 words given a block request size of 32 bytes. For more information, refer to Block Request Size.

File Numbers and Word Locations

The following file numbers and word locations are allowed for each model. The maximum word location is one less when accessing as a 32-bit data type (Long, DWord, or Long BCD).

PLC Model	File Number	Max. Word
SLC 5/05 Open	3, 9-999	999

PLC Model	File Number	Max. Word
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

Example	Description
B3:0	Word 0
B3/26	Bit 26
B12:4/15	Bit 15 word 4
B3:10 [20]	20 element array starting at word 10
B15:0 [6][6]	6 by 6 element array starting at word 0
B3/7 [8]	8 element Boolean array starting at bit 7
B3/32 [6][9]	6 by 9 element Boolean array starting at bit 32
B3:11/2 [12]	12 element Boolean array starting at bit 2 word 11
B3:23/4 [5][8]	5 by 8 element Boolean array starting at bit 4 word 23

Timer Files

Timer files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
T<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	4, 9-999	999
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
ACC	Short, **Word**	Read/Write
PRE	Short, **Word**	Read/Write
DN	**Boolean**	Read Only
TT	**Boolean**	Read Only
EN	**Boolean**	Read Only

Example	Description
T4:0.ACC	Accumulator of timer 0 file 4
T4:10.DN	Done bit of timer 10 file 4
T15:0.PRE	Preset of timer 0 file 15

Counter Files

Counter files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
C<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	5, 9-999	999
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
ACC	Short, **Word**	Read/Write
PRE	Short, **Word**	Read/Write
UA	**Boolean**	Read Only
UN	**Boolean**	Read Only
OV	**Boolean**	Read Only
DN	**Boolean**	Read Only
CD	**Boolean**	Read Only
CU	**Boolean**	Read Only

Example	Description
C5:0.ACC	Accumulator of counter 0 file 5
C5:10.DN	Done bit of counter 10 file 5
C15:0.PRE	Preset of counter 0 file 15

Control Files

Control files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
R<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	6, 9-999	999
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
LEN	Short, **Word**	Read/Write
POS	Short, **Word**	Read/Write
FD	**Boolean**	Read Only

Element Field	Data Type	Access
IN	**Boolean**	Read Only
UL	**Boolean**	Read Only
ER	**Boolean**	Read Only
EM	**Boolean**	Read Only
DN	**Boolean**	Read Only
EU	**Boolean**	Read Only
EN	**Boolean**	Read Only

Examples	Description
R6:0.LEN	Length field of control 0 file 6
R6:10.DN	Done bit of control 10 file 6
R15:18.POS	Position field of control 18 file 15

Integer Files

To access Integer files, specify a file number and a word (and optionally, a bit in the word). The default data type for all syntax is shown in **bold**.

Syntax	Data Type
N<file>:<word>	Short, **Word**, BCD, DWord, Long, LBCD
N<file>:<word> [rows][cols]	Short, **Word**, BCD, DWord, Long, LBCD*
N<file>:<word> [cols]	Short, **Word**, BCD, DWord, Long, LBCD*
N<file>:<word>/<bit>	**Boolean**
N<file>:<word>/<bit> [rows][cols]	**Boolean***
N<file>:<word>/<bit> [cols]	**Boolean***
N<file>/bit	**Boolean**
N<file>/bit [rows][cols]	**Boolean***
N<file>/bit [cols]	**Boolean***

*Array types

Note: The number of array elements (in bytes) cannot exceed the block request size specified. This means that array size cannot exceed 16 words given a block request size of 32 bytes. For more information, refer to Block Request Size.

File Numbers and Word Locations

The following file numbers and maximum word locations are allowed for each model. The maximum word location is one less when accessing as a 32-bit data type (Long, DWord or Long BCD).

PLC Model	File Number	Max. Word
SLC 5/05 Open	7, 9-999	999
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

Example	Description
N7:0	Word 0
N7/26	Bit 26
N12:4/15	Bit 15 word 4

Example	Description
N7:10 [8]	8 element array starting at word 10
N15:0 [4][5]	4 by 5 element array starting at word 0
N7/12 [9]	9 element Boolean array starting at bit 12
N7/19 [3][11]	3 by 11 element Boolean array starting at bit 19
N7:7/0 [10]	10 element Boolean array starting at bit 0 word 7
N7:29/13 [2][15]	2 by 15 element Boolean array starting at bit 13 word 29

Float Files

To access Float files, specify a file number and an element. The only data type allowed is Float.

Syntax	Data Type
F<file>:<element>	Float
F<file>:<element> [rows][cols]	Float array
F<file>:<element> [cols]	Float array

Note: The number of array elements (in bytes) cannot exceed the block request size specified. This means array size cannot exceed 8 floats given a block request size of 32 bytes. For more information, refer to Block Request Size.

File Numbers and Word Locations

The following file numbers and maximum word locations are allowed for each model.

PLC Model	File Number	Max. Word
SLC 5/05 Open	8-999	999
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

Example	Description
F8:0	Float 0
F8:10 [16]	16-element array starting at word 10
F15:0 [4][4]	16-element array starting at word 0

ASCII Files

To access ASCII file data, specify a file number and character location. The default data type for all syntax is shown in **bold**.

Syntax	Data Type
A<file>:<char>	**Char**, Byte*
A<file>:<char> [rows][cols]	**Char**, Byte*
A<file>:<char> [cols]	**Char**, Byte*
A<file>:<word offset>/length	**String****

Note: The number of array elements cannot exceed the block request size specified. For more information, refer to Block Request Size.

*The PLC packs two characters per word in the file, with the high byte containing the first character and the low byte containing the second character. The PLC programming software allows access at the word level or two-character level. The AB Ethernet driver allows accessing to the character level. Examples are as follows:

- Using the programming software A10:0=AB would result in 'A' being stored in the high byte of A10:0 and 'B' being stored in the low byte.
- Using the AB Ethernet driver, two assignments, A10:0=A and A10:1=B, would result in the same data being stored in the PLC memory.

**Referencing this file as string data allows access to data at word boundaries like the programming software. The length can be up to 236 characters. If a string that is sent to the device is smaller in length than the length specified by the address, the driver null terminates the string before sending it down to the controller.

File Numbers and Character Locations

The following file numbers and maximum character locations are allowed for each model.

PLC Model	File Number	Max. Character
SLC 5/05 Open	9-999	1999
PLC-5 Family	3-999	1999
SoftPLC	N/A	N/A

Note: All SLC 500 PLCs do not support ASCII file types. For more information, refer to the PLC documentation.

Example	Description
A9:0	Character 0 (high byte of word 0)
A27:10 [80]	80-character array starting at character 10
A15:0 [4][16]	4 by 16 character array starting at character 0
A62:0/32	32-character string starting at word offset 0

String Files

To access data in a String file, specify a file number and an element. The only data type allowed is string, which are 82-character null-terminated arrays. The driver places the null terminator based on the string length returned by the PLC.

Syntax	Data Type
ST<file>:<element>	String

Note: Arrays of strings are not supported.

Tip: String length can be obtained with a COPY or MOVE function.

File Numbers and Word Locations

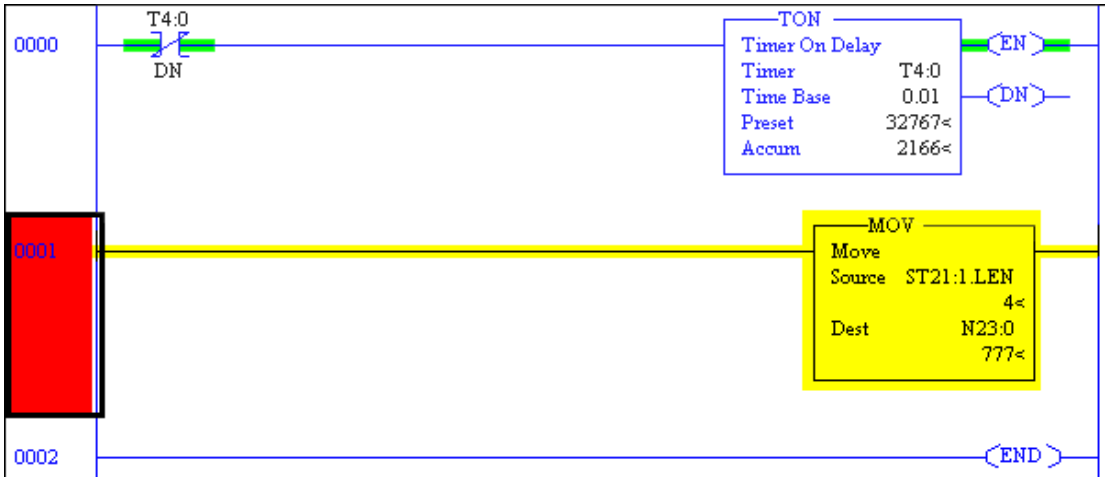
The following file numbers and maximum word locations are allowed for each model.

PLC Model	File Number	Max. Word
SLC 5/05 Open	9-999	999
PLC-5 Family	3-999	999
SoftPLC	3-9990	9999

Example	Description
ST9:0	String 0
ST18:10	String 10

String Length

While the .LEN field is not supported, string length can be obtained with a COPY or MOVE function, as shown below.



SLC 5/05 Open Addressing

The actual number of addresses available depends on the model of the PLC. The ranges have been opened up to allow for maximum flexibility with future models. If the driver finds at runtime that an address is not present in the device, it posts an error message and removes the tag from its scan list.

Note: This model has no specific addressing.

See Also: General Addressing

PLC-5 Family and SoftPLC Addressing

General Addressing

General Addressing

Model-Specific Addressing

BCD Files
PID Files
Message Files
Block Transfer Files

BCD Files

To access BCD files, specify a file number and a word. The only data types allowed are BCD and long BCD. The default data type is always BCD.

Syntax	Data Type
D<file>:<word>	**BCD**, LBCD
D<file>:<word> [rows][cols]	**BCD**, LBCD*
D<file>:<word> [cols]	**BCD**, LBCD*

*Array types

Note: The number of array elements (in bytes) cannot exceed the block request size specified. This means array size cannot exceed 16 BCDs given a block request size of 32 bytes. For more information, refer to Block Request Size.

File Numbers and Word Locations

The following file numbers and maximum word locations are allowed for each model.

PLC Model	File Number	Max. Word
SLC 5/05 Open	NA	NA
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

Example	Description
D9:0	Word 0
D27:10 [16]	16 element array starting at word 10
D15:0 [4][8]	32 element array starting at word 0

PID Files

PID files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
PD<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	NA	NA
PLC-5 Family	3-999	999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
SP	**Real**	Read/Write
KP	**Real**	Read/Write
KI	**Real**	Read/Write
KD	**Real**	Read/Write
BIAS	**Real**	Read/Write
MAXS	**Real**	Read/Write
MINS	**Real**	Read/Write
DB	**Real**	Read/Write
SO	**Real**	Read/Write
MAXO	**Real**	Read/Write
MINO	**Real**	Read/Write
UPD	**Real**	Read/Write
PV	**Real**	Read/Write
ERR	**Real**	Read/Write
OUT	**Real**	Read/Write
PVH	**Real**	Read/Write
PVL	**Real**	Read/Write
DVP	**Real**	Read/Write
DVN	**Real**	Read/Write
PVDB	**Real**	Read/Write

Element Field	Data Type	Access
DVDB	**Real**	Read/Write
MAXI	**Real**	Read/Write
MINI	**Real**	Read/Write
TIE	**Real**	Read/Write
FILE	Short, **Word**	Read/Write
ELEM	Short, **Word**	Read/Write
EN	**Boolean**	Read/Write
CT	**Boolean**	Read/Write
CL	**Boolean**	Read/Write
PVT	**Boolean**	Read/Write
DO	**Boolean**	Read/Write
SWM	**Boolean**	Read/Write
CA	**Boolean**	Read/Write
MO	**Boolean**	Read/Write
PE	**Boolean**	Read/Write
INI	**Boolean**	Read/Write
SPOR	**Boolean**	Read/Write
OLL	**Boolean**	Read/Write
OLH	**Boolean**	Read/Write
EWD	**Boolean**	Read/Write
DVNA	**Boolean**	Read/Write
DVHA	**Boolean**	Read/Write
PVLA	**Boolean**	Read/Write
PVHA	**Boolean**	Read/Write

Example	Description
PD14:0.SP	Set point field of PD 0 file 14
PD18:6.EN	Status enable bit of PD 6 file 18

Message Files

Message files are a structured type whose data is accessed by specifying a file number, an element and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
MG<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	NA	NA
PLC-5 Family	3-999	999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
ERR	Short, **Word**	Read/Write
RLEN	Short, **Word**	Read/Write
DLEN	Short, **Word**	Read/Write
EN	**Boolean**	Read/Write
ST	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
ER	**Boolean**	Read/Write
CO	**Boolean**	Read/Write
EW	**Boolean**	Read/Write
NR	**Boolean**	Read/Write
TO	**Boolean**	Read/Write

Example	Description
MG14:0.RLEN	Requested length field of MG 0 file 14
MG18:6.CO	Continue bit of MG 6 file 18

Block Transfer Files

Block transfer files are a structured type whose data is accessed by specifying a file number, an element, and a field. The default data type depends on the field being accessed. Integer fields receive a default data type of Word.

Syntax	Data Type
BT<file>:<element>.<field>	Depends on field

File Numbers and Elements

The following file numbers and maximum element are allowed for each model.

PLC Model	File Number	Max. Element
SLC 5/05 Open	NA	NA
PLC-5 Family	3-999	1999
SoftPLC	3-9999	9999

The following fields are allowed for each element. Refer to the PLC documentation for the meaning of each field.

Element Field	Data Type	Access
RLEN	Short, **Word**	Read/Write
DLEN	Short, **Word**	Read/Write
FILE	Short, **Word**	Read/Write
ELEM	Short, **Word**	Read/Write
RW	**Boolean**	Read/Write
ST	**Boolean**	Read/Write
DN	**Boolean**	Read/Write
ER	**Boolean**	Read/Write
CO	**Boolean**	Read/Write
EW	**Boolean**	Read/Write
NR	**Boolean**	Read/Write
TO	**Boolean**	Read/Write

Example	Description
BT14:0.RLEN	Requested length field of BT 0 file 14
BT18:6.CO	Continue bit of BT 6 file 18

Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the OPC 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.

Tip: Messages that originate from a data source (such as third-party software, including databases) are presented through the Event Log. Troubleshooting steps should include researching those messages online and in vendor documentation.

Unable to read data block from device. Frame received contains errors. | Block start address = '<address>'.

Error Type:
Warning

Possible Cause:

1. An incorrect frame size was received.
2. A TNS mismatch occurred.
3. An invalid response command was returned from the device.
4. Misalignment of packets occurred due to connection/disconnection between the PC and device.
5. Bad cabling connecting the devices is causing noise.

Possible Solution:
While the driver can recover from the error without intervention, there may be an issue with the cabling or the device itself that should be corrected.

Unable to read data block from the device. Tag deactivated. | Block start address = '<address>', Status code = <code>, Extended status code = <code>.

Error Type:
Warning

Possible Cause:

1. The address requested does not exist in the PLC.
2. The address requested cannot be accessed because the PLC is in error state.
3. The communications parameters for the Ethernet connection are incorrect.

Possible Solution:

1. Verify the address exists in the PLC.
2. Verify the PLC is not in an error state or restore the PLC to operation.
3. Verify the communications parameters for the Ethernet connection are correct.
4. Verify the correct port is specified for the named device.
5. Verify the IP address given to the named device matches that of the actual device.

Note:

1. Check the status and extended status codes returned by the PLC. The extended status code may not always be returned; error information is contained within the status code. The codes are displayed in hexadecimal.
2. Status code errors in the low nibble indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot find the destination PLC on the network.
3. Status code errors in the high nibble indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this error. This error can be generated if the address does not exist in the PLC.

Unable to write to address on device. Frame received contains errors. | Address = '<address>'.

Error Type:
Warning

Possible Cause:

1. Incorrect frame size was received.
2. TNS mismatch occurred.
3. Invalid response command was returned from the device.
4. Misalignment of packets was caused by connection/disconnection between the PC and the device.
5. Bad cabling connecting the devices is causing noise.

Possible Solution:
While the driver can recover from this error without intervention, there may be an issue with the cabling or the device itself that should be corrected.

Unable to read data block from device. | Block start address = '<address>', Status code = <code>, Extended status code = <code>.

Error Type:
Warning

Possible Cause:

1. The address requested does not exist in the PLC.
2. The address requested cannot be accessed because the PLC is in error state.
3. The communications parameters for the Ethernet connection are incorrect.

Possible Solution:

1. Verify the address exists in the PLC.
2. Verify the PLC is not in an error state or restore the PLC to operation.
3. Verify the communications parameters for the Ethernet connection are correct.
4. Verify the correct port is specified for the named device.
5. Verify the IP address given to the named device matches that of the actual device.

Note:

1. Check the status and extended status codes returned by the PLC. The extended status code may not always be returned; error information is contained within the status code. The codes are displayed in hexadecimal.
2. Status code errors in the low nibble indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot find the destination PLC on the network.
3. Status code errors in the high nibble indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this error. This error can be generated if the address does not exist in the PLC.

Unable to read data block from device. Tag deactivated. | Block start address = '<address>', Status code = <code>.

Error Type:
Warning

Possible Cause:
1. The address requested does not exist in the PLC.
2. The address requested cannot be accessed because the PLC is in error state.
3. The communications parameters for the Ethernet connection are incorrect.

Possible Solution:
1. Verify the address exists in the PLC.
2. Verify the PLC is not in an error state or restore the PLC to operation.
3. Verify the communications parameters for the Ethernet connection are correct.
4. Verify the correct port is specified for the named device.
5. Verify the IP address given to the named device matches that of the actual device.

Note:
1. Check the status and extended status codes returned by the PLC. The extended status code may not always be returned; error information is contained within the status code. The codes are displayed in hexadecimal.
2. Status code errors in the low nibble indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot find the destination PLC on the network.
3. Status code errors in the high nibble indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this error. This error can be generated if the address does not exist in the PLC.

Unable to write to address on device. | Address = '<address>', Status code = <code>, Extended status code = <code>.

Error Type:
Warning

Possible Cause:
The address written to does not exist in the PLC.

Possible Solution:
Verify the address exists in the PLC.

Note:

1. Check the status and extended status codes returned by the PLC. The extended status code may not always be returned; error information is contained within the status code. The codes are displayed in hexadecimal.
2. Status code errors in the low nibble indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot find the destination PLC on the network.
3. Status code errors in the high nibble indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this error. This error can be generated if the address does not exist in the PLC.

Unable to read data block from device. | Block start address = '<address>', Status code = <code>.

Error Type:
Warning

Possible Cause:

1. The address requested does not exist in the PLC.
2. The address requested cannot be accessed because the PLC is in error state.
3. The communications parameters for the Ethernet connection are incorrect.

Possible Solution:

1. Verify the address exists in the PLC.
2. Verify the PLC is not in an error state or restore the PLC to operation.
3. Verify the communications parameters for the Ethernet connection are correct.
4. Verify the correct port is specified for the named device.
5. Verify the IP address given to the named device matches that of the actual device.

Note:

1. Check the status and extended status codes returned by the PLC. The extended status code may not always be returned; error information is contained within the status code. The codes are displayed in hexadecimal.
2. Status code errors in the low nibble indicate errors found by the local node. The driver continues to retry reading these blocks of data periodically. Errors found by the local node occur when the KF module cannot find the destination PLC on the network.
3. Status code errors in the high nibble indicate errors found by the PLC. These errors are generated when the block of data the driver is asking for is not available in the PLC. The driver does not ask for these blocks again after receiving this error. This error can be generated if the address does not exist in the PLC.

Unable to write to address on device. | Address = '<address>', Status code = <code>.

Error Type:
Warning

Possible Cause:

1. The Ethernet connection between the device and the host PC is broken.
2. The communication parameters for the Ethernet connection are incorrect.
3. The named device may have been assigned an incorrect IP address.

Possible Solution:

1. Verify the cabling between the PC and the device.
2. Verify that the correct port is specified for the named device.
3. Verify that the IP address given to the named device matches that of the actual device.

Unable to write to address on device. Packet length is out of range. | Address = '<address>', Expected packet length = <low> to <high> (bytes).

Error Type:
Informational

Unable to write to address on device. TNS is out of range. | Address = '<address>', Expected TNS range = <low> to <high>.

Error Type:
Informational

Index

A

B

C

D

E

F

G

I

L

M

N

O

P

R

S

T

U

V

W