

Ping Driver

© 2018 PTC Inc. All Rights Reserved.

Table of Contents

Ping Driver	1
Table of Contents	2
Ping Driver	3
Overview	4
Channel Properties — General	4
Channel Properties — Ethernet Communications	5
Channel Properties — Write Optimizations	5
Channel Properties — Advanced	6
Device Properties — General	7
Device ID	8
Device Properties — Scan Mode	8
Device Properties — Timing	9
Device Properties — Auto-Demotion	10
Device Properties — Tag Generation	11
Device Properties — Redundancy	13
Device Discovery	13
Device Discovery Procedure	14
Data Types Description	15
Address Descriptions	16
Error Descriptions	17
Device <address> is not responding	17
Device Discovery has exceeded <max devices> maximum allowed devices	17
The Device ID for <address> could not be resolved	17
Required functionality of icmp.dll is not found	18
Unable to bind to adapter: <adapter>. Connect failed	18
Unable to generate a tag database for device <device name>. Reason: Could not generate tags due to low system resources	18
Unable to load required file <file>. This driver will not be operational	18
Winsock initialization failed (OS Error = <error>)	19
Winsock shut down failed (OS Error = <error>)	19
Winsock V1.1 or higher must be installed to use the Ping Driver	19
Index	20

Ping Driver

Help version 1.025

CONTENTS

Overview

What is the Ping Driver?

Channel Setup

How do I configure the driver to search for devices on the network?

Device Setup

How do I configure a device for use with this driver?

Data Types Description

What data types can be used with the Ping Driver?

Address Descriptions

How do I specify an address with the Ping Driver?

Error Descriptions

What error messages does the Ping Driver produce?

Overview

The Ping Driver plugs into the industrial based communications OPC server to monitor any unmanaged devices. It also provides data to OPC Client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. It enables users to monitor the status of a network device as well as the time that it takes for the ICMP message to reach its destination and return a response (the RoundTripTime).

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.

Property Groups	[-] Identification	
General	Name	
Write Optimizations	Description	
Advanced	Driver	
	[-] Diagnostics	
	Diagnostics Capture	Disable

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 not available if the driver does not support diagnostics.

• For more information, refer to "Communication Diagnostics" in the server help.

Channel Properties — Ethernet Communications

Ethernet Communication can be used to communicate with devices.

Property Groups	[-] Ethernet Settings	
General	Network Adapter	Default
Ethernet Communications		
Write Optimizations		
Advanced		

Ethernet Settings

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

Channel Properties — Write Optimizations

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

Property Groups	[-] Write Optimizations	
General	Optimization Method	Write Only Latest Value for All Tags
Write Optimizations	Duty Cycle	10

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.

Property Groups	[-] Non-Normalized Float Handling	
General	Floating-Point Values	Replace with Zero
Write Optimizations	[-] Inter-Device Delay	
Advanced	Inter-Device Delay (ms)	0

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

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

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

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

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

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

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.

Property Groups	Identification	
General	Name	
Scan Mode	Description	
	Channel Assignment	
	Driver	
	Model	
	ID Format	Decimal
	ID	2
	Operating Mode	
	Data Collection	Enable
	Simulated	No

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 ID

A device represents a single computer or device that can communicate via TCP/IP.

The Ping Driver supports any IPv4 IP address as its Device ID. Domain names can also be used as a Device ID, since these will be converted to an IP Address by the Ping Driver automatically. When using a domain name, do not include the http:// prefix.

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.

Property Groups	<input checked="" type="checkbox"/> Scan Mode	
General	Scan Mode	Respect Client-Specified Scan Rate ▾
Scan Mode	Initial Updates from Cache	Disable

Scan Mode: specifies 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 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.

Property Groups	<input type="checkbox"/> Communication Timeouts	
General	Connect Timeout (s)	3
Scan Mode	Request Timeout (ms)	5000
Timing	Retry Attempts	3
Auto-Demotion	<input type="checkbox"/> Timing	
	Inter-Request Delay (ms)	0

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.

Property Groups	Auto-Demotion	
General	Demote on Failure	Enable
Scan Mode	Timeouts to Demote	3
Timing	Demotion Period (ms)	10000
Auto-Demotion	Discard Requests when Demoted	Disable

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 — Tag Generation

The automatic tag database generation features make setting up an application a plug-and-play operation. Select communications drivers can be configured to automatically build a list of tags that correspond to device-specific data. These automatically generated tags (which depend on the nature of the supporting driver) can be browsed from the clients.

● *Not all devices and drivers support full automatic tag database generation and not all support the same data types. Consult the data types descriptions or the supported data type lists for each driver for specifics.*

If the target device supports its own local tag database, the driver reads the device's tag information and uses the data to generate tags within the server. If the device does not natively support named tags, the driver creates a list of tags based on driver-specific information. An example of these two conditions is as follows:

1. If a data acquisition system supports its own local tag database, the communications driver uses the tag names found in the device to build the server's tags.
2. If an Ethernet I/O system supports detection of its own available I/O module types, the communications driver automatically generates tags in the server that are based on the types of I/O modules plugged into the Ethernet I/O rack.

● **Note:** Automatic tag database generation's mode of operation is completely configurable. For more information, refer to the property descriptions below.

Property Groups	<input type="checkbox"/> Tag Generation	
General	On Property Change	Yes
Scan Mode	On Device Startup	Do Not Generate on Startup
Timing	On Duplicate Tag	Delete on Create
Auto-Demotion	Parent Group	
Tag Generation	Allow Automatically Generated Subgroups	Enable
Tag Import	Create	Create tags
Redundancy		

On Property Change: If the device supports automatic tag generation when certain properties change, the **On Property Change** option is shown. It is set to **Yes** by default, but it can be set to **No** to control over when tag generation is performed. In this case, the **Create tags** action must be manually invoked to perform tag generation.

On Device Startup: This property specifies when OPC tags are automatically generated. Descriptions of the options are as follows:

- **Do Not Generate on Startup:** This option prevents the driver from adding any OPC tags to the tag space of the server. This is the default setting.
- **Always Generate on Startup:** This option causes the driver to evaluate the device for tag information. It also adds tags to the tag space of the server every time the server is launched.

- **Generate on First Startup:** This option causes the driver to evaluate the target device for tag information the first time the project is run. It also adds any OPC tags to the server tag space as needed.

● **Note:** When the option to automatically generate OPC tags is selected, any tags that are added to the server's tag space must be saved with the project. Users can configure the project to automatically save from the **Tools | Options** menu.

On Duplicate Tag: When automatic tag database generation is enabled, the server needs to know what to do with the tags that it may have previously added or with tags that have been added or modified after the communications driver since their original creation. This setting controls how the server handles OPC tags that were automatically generated and currently exist in the project. It also prevents automatically generated tags from accumulating in the server.

For example, if a user changes the I/O modules in the rack with the server configured to **Always Generate on Startup**, new tags would be added to the server every time the communications driver detected a new I/O module. If the old tags were not removed, many unused tags could accumulate in the server's tag space. The options are:

- **Delete on Create:** This option deletes any tags that were previously added to the tag space before any new tags are added. This is the default setting.
- **Overwrite as Necessary:** This option instructs the server to only remove the tags that the communications driver is replacing with new tags. Any tags that are not being overwritten remain in the server's tag space.
- **Do not Overwrite:** This option prevents the server from removing any tags that were previously generated or already existed in the server. The communications driver can only add tags that are completely new.
- **Do not Overwrite, Log Error:** This option has the same effect as the prior option, and also posts an error message to the server's Event Log when a tag overwrite would have occurred.

● **Note:** Removing OPC tags affects tags that have been automatically generated by the communications driver as well as any tags that have been added using names that match generated tags. Users should avoid adding tags to the server using names that may match tags that are automatically generated by the driver.

Parent Group: This property keeps automatically generated tags from mixing with tags that have been entered manually by specifying a group to be used for automatically generated tags. The name of the group can be up to 256 characters. This parent group provides a root branch to which all automatically generated tags are added.

Allow Automatically Generated Subgroups: This property controls whether the server automatically creates subgroups for the automatically generated tags. This is the default setting. If disabled, the server generates the device's tags in a flat list without any grouping. In the server project, the resulting tags are named with the address value. For example, the tag names are not retained during the generation process.

● **Note:** If, as the server is generating tags, a tag is assigned the same name as an existing tag, the system automatically increments to the next highest number so that the tag name is not duplicated. For example, if the generation process creates a tag named "AI22" that already exists, it creates the tag as "AI23" instead.

Create: Initiates the creation of automatically generated OPC tags. If the device's configuration has been modified, **Create tags** forces the driver to reevaluate the device for possible tag changes. Its ability to be accessed from the System tags allows a client application to initiate tag database creation.

- **Note:** **Create tags** is disabled if the Configuration edits a project offline.

Device Properties — Redundancy

Property Groups	<input type="checkbox"/> Redundancy	
General	Secondary Path	...
Scan Mode	Operating Mode	Switch On Failure
Timing	Monitor Item	
Redundancy	Monitor Interval (s)	300
	Return to Primary ASAP	Yes

Redundancy is available with the Media-Level Redundancy Plug-In.

- Consult the website, a sales representative, or the user manual for more information.

Device Discovery

This channel-level dialog is used to specify properties for locating devices on the network. Once devices are found, they may be added to the channel. The maximum number of devices that can be discovered at once is 65535. For more information on device discovery, refer to the server help file.

Discovery Settings

This dialog is used to specify the discovery properties.

Starting IP Address: This property specifies the starting IP address. The default setting is 0.0.0.0.

Ending IP Address: This property specifies the ending IP address. The default setting is 0.0.0.0.

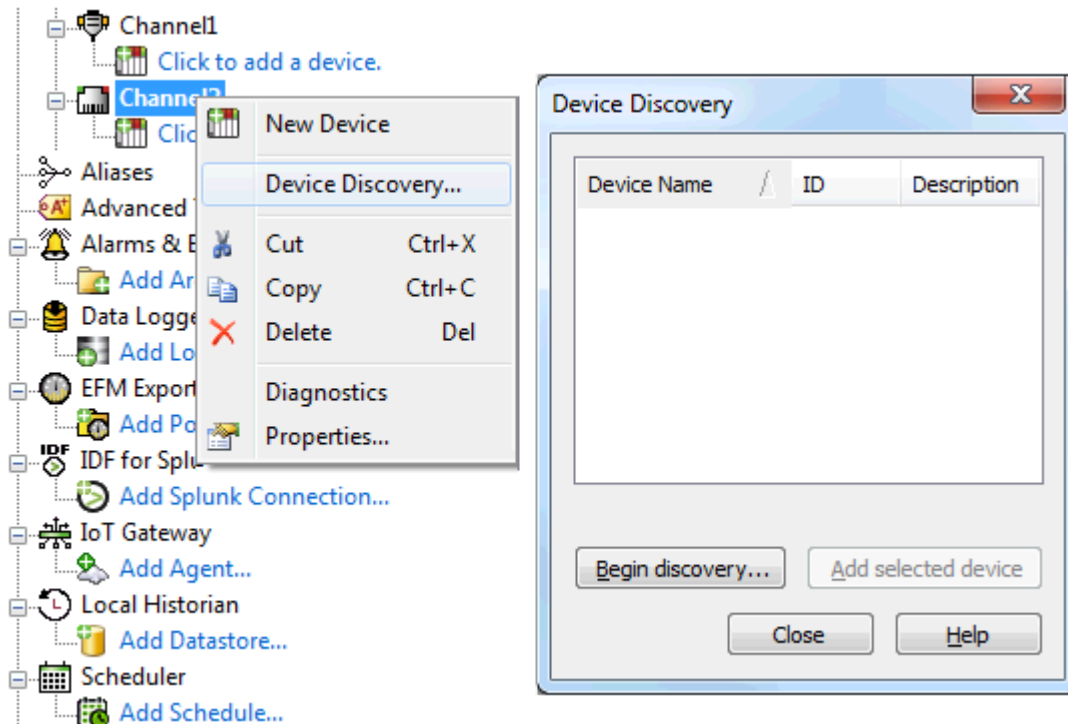
Timeout (msec): This property specifies the time that the driver will wait for a connection to be made with a device, as well as the time that the driver will wait on a response from the device before giving up and going on to the next request. The default setting is 500 msec.

Number of Retries: This property specifies the number of times the driver will retry a message before giving up and going on to the next message. The default setting is 1.

Resolve IP Addresses to Host Names: When checked, the driver will attempt to find host names for the discovered devices. Once found, the host names will be displayed in the Description field of the Device Discovery tab located in Channel Properties. The default setting is unchecked.

Device Discovery Procedure

Device Discovery is available for drivers that support locating devices on the network. Once devices are found, they may be added to a channel. The maximum number of devices that can be discovered at once is 65535.



1. Select the channel in which devices should be discovered and added.
2. Right click on the channel node and select **Device Discovery...**
3. Click the **Begin discovery...** button to start the discovery process.
4. Specify the discovery properties, which are driver-specific, such as address range, timeout, discovery scope.
5. Click **OK**.
6. Devices discovered populate the dialog with the following information / headings **Name, ID, Description**.
7. If any discovered device is of interest, select that device and click **Add selected device...**
8. Click **Close**.

Data Types Description

Each address that can be accessed must be assigned a data type. The Ping Driver supports two data types: DWord and Long.

Data Type	Description
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 but bit 31 is the sign bit

Address Descriptions

The Ping Driver supports two types of addresses: **Status** and **RoundTripTime**. Both addresses are Read Only and supply different information about the device. The Status address indicates 1 if the device can be reached or 0 if the Ping Driver could not reach the device. The RoundTripTime indicates the amount of time in milliseconds that it took the request to reach its destination and to reply back to the Ping Driver.

Address	Data Type	Access
Status	DWord	Read Only
comStatus	DWord	Read Only
RoundTripTime	Long	Read Only
comTime	Long	Read Only

● **Note:** The 'comStatus' and 'comTime' addresses behave exactly as the Status and RoundTripTime addresses with the exception that they will not be auto generated.

Error Descriptions

The following error / warning messages may be generated. The messages are listed here in alphabetical order.

Device <address> is not responding

Device Discovery has exceeded <max devices> maximum allowed devices

The Device ID for <address> could not be resolved

Required functionality of icmp.dll is not found

Unable to bind to adapter: <adapter>. Connect failed

Unable to generate a tag database for device <device name>. Reason: Could not generate tags due to low system resources

Unable to load required file <file>. This driver will not be operational

Winsock initialization failed (OS Error = <error>)

Winsock shut down failed (OS Error = <error>)

Winsock V1.1 or higher must be installed to use the Ping Driver

Device <address> is not responding

Error Type:

Warning

Possible Cause:

Device may not be connected or the device cannot be reached.

Solution:

1. Ensure that the Ethernet cable is connected properly.
2. Ensure that the Device ID is correct.

Device Discovery has exceeded <max devices> maximum allowed devices

Error Type:

Warning

Possible Cause:

The Device Discovery has exceeded the maximum number of allowed devices.

Solution:

Limit the discovery range and then try again.

The Device ID for <address> could not be resolved

Error Type:

Warning

Possible Cause:

The host name entered does not exist or is not currently in use.

Solution:

Check that the host name is spelled correctly and that all network cables are properly connected. If using web based addresses, make sure that the http:// prefix is not included.

Required functionality of icmp.dll is not found

Error Type:

Fatal

Possible Cause:

icmp.dll is corrupted or out of sync with the operating system.

Solution:

Check that the proper version of icmp.dll is in the system folder of the particular operating system.

Unable to bind to adapter: <adapter>. Connect failed

Error Type:

Fatal

Possible Cause:

The reason for this error message is that some other device in the driver has already been bound to this adapter/port combination.

Solution:

Wait a few seconds and restart the driver.

**Unable to generate a tag database for device <device name>. Reason:
Could not generate tags due to low system resources**

Error Type:

Warning

Possible Cause:

Memory required for database generation could not be allocated. The process is aborted.

Solution:

Close any unused applications and/or increase the amount of virtual memory. Then, try again.

Unable to load required file <file>. This driver will not be operational

Error Type:

Fatal

Possible Cause:

The required file is not found.

Solution:

Ensure that the file is in the proper location specified in <file>.

Winsock initialization failed (OS Error = <error>)

Error Type:

Fatal

OS Error	Indication	Possible Solution
10091	Indicates that the underlying network subsystem is not ready for network communication.	Wait a few seconds and restart the driver.
10067	Limit on the number of tasks supported by the Windows Sockets implementation has been reached.	Close one or more applications that may be using Winsock and restart the driver.

Winsock shut down failed (OS Error = <error>)

Error Type:

Fatal

OS Error	Indication	Possible Solution
10091	This indicates that there is a problem in the underlying network subsystem that is preventing the Winsock Library from shutting down correctly.	Check to make sure that the version of Winsock installed is 1.1 or higher.

Winsock V1.1 or higher must be installed to use the Ping Driver

Error Type:

Fatal

Possible Cause:

The version number of the Winsock DLL found on the system is less than 1.1.

Solution:

Upgrade Winsock to version 1.1 or higher.

Index

A

Address Descriptions 16
Allow Sub Groups 12
Attempts Before Timeout 10

C

Channel Assignment 7
Communications Timeouts 9-10
Connect Timeout 9
Create 12

D

Data Collection 8
Data Types Description 15
Delete 12
Demote on Failure 10
Demotion Period 10
Description 7
Device <address> is not responding 17
Device Discovery has exceeded <max devices> maximum allowed devices 17
Device ID 8
Device Properties — Auto-Demotion 10
Device Properties — General 7
Device Properties — Tag Generation 11
Discard Requests when Demoted 11
Do Not Scan, Demand Poll Only 9
Driver 7

E

Error Descriptions 17

G

Generate 11

H

Help Contents 3

I

ID 7

Initial Updates from Cache 9

Inter-Request Delay 10

M

Model 7

N

Name 7

O

On Device Startup 11

On Duplicate Tag 12

On Property Change 11

Overview 4

Overwrite 12

P

Parent Group 12

R

Redundancy 13

Request All Data at Scan Rate 9
Request Data No Faster than Scan Rate 9
Request Timeout 9
Required functionality of icmp.dll is not found 18
Respect Client-Specified Scan Rate 9
Respect Tag-Specified Scan Rate 9

S

Scan Mode 8
Simulated 8

T

Tag Generation 11
The address <address> could not be resolved 17
Timeouts to Demote 10

U

Unable to bind to adapter: <adapter>. Connect failed 18
Unable to generate a tag database for device <device name>. Reason: Could not generate tags due to low system resources 18
Unable to load required file <file>. This driver will not be operational 18

W

Winsock initialization failed (OS Error = <error>) 19
Winsock shut down failed (OS Error = <error>) 19
Winsock V1.1 or higher must be installed to use the PingDriver 19