Triconex Ethernet Driver Help

Table of Contents

Triconex Ethernet Driver Help

Help version 1.012

CONTENTS

Overview
What is the Triconex Ethernet Driver?

Channel Setup
How do I configure channels for use with this driver?

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

Network Redundancy
How can I use a redundant network with this driver?

Optimizing Ethernet Communications
How can I configure this driver for optimum performance?

Automatic Tag Database Generation
How can I import tags from a Triconex TriStation project?

Data Types Description
What data types does this driver support?

Address Descriptions
How do I address a data location on a Triconex Ethernet device?

Error Descriptions
What error messages does the Triconex Ethernet Driver produce?

Overview

The Triconex Ethernet Driver provides an easy and reliable way to connect Triconex Ethernet devices to OPC Client applications, including HMI, SCADA, Historian, MES, ERP and countless custom applications. It is intended for use with Triconex Tricon and Trident devices equipped with Ethernet communications modules. Network redundancy is supported.

Channel Setup

Network Interface
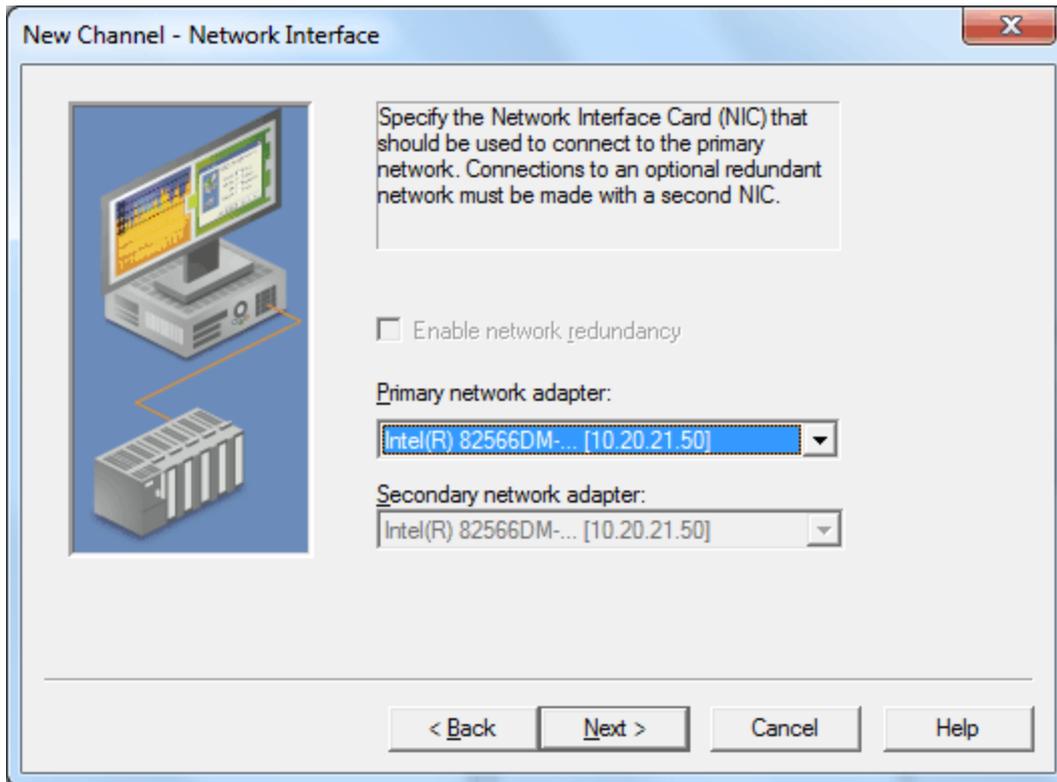
Descriptions of the parameters are as follows:

- **Enable Network Redundancy:** This parameter enables network redundancy. It will be enabled by default if the PC has more than one network adapter card (NIC) installed.
- **Primary Network Adapter:** This parameter specifies the NIC to which the primary network is connected.
- **Secondary Network Adapter:** This parameter specifies the NIC to which the optional secondary network is connected. This secondary NIC cannot be the same as the primary NIC, and should have different subnet addresses.

See Also: Network Redundancy

Note: These settings are specific to the host computer. If a Triconex Ethernet Driver project is copied to another machine, these settings will most likely need to be changed. The driver will post a message to the server's Event Log if changes are required. In order for Windows to correctly report the configuration to the driver, the network adapters must be connected to a device or hub.

See Also:
Unable to use specified primary network adapter '<adapter>' on channel '<channel>'
Unable to use specified secondary network adapter '<adapter>' on channel '<channel>'
Unable to use network redundancy on channel '<channel>'

Device Setup

Supported Devices
Tricon
Trident

Communication Protocol
Triconex System Access Application (TSAA)

Device ID
The Device ID is the node number of the device. The valid range is 1 to 31.

Request Timeout
This parameter specifies the number of milliseconds that the driver will wait for a response before retrying. Users may specify a request timeout between 100 to 30000 milliseconds. For this driver, request timeouts only apply to Write operations. The device may take up to four scan cycles to complete a Write request. The controller may be programmed to use a scan period as large as 400 milliseconds. Setting the request timeout too low may cause the driver to assume a write has failed when it hasn't been processed by the controller yet.

Note: Data is read from cache, which is updated at a regular interval. For more information, refer to **Communications**.

Fail After *x* Successive Timeouts
This parameter specifies the number of times that the driver will retry a request before failing it. This only pertains to write requests. The valid range is 1 to 10 retries.

Cable Diagrams
To view a specific model's cable diagram, select a link from the list below.

Cable Diagrams for Tricon
Cable Diagram for Trident

CM Configuration

Descriptions of the parameters are as follows:

- **IP of CM on Primary Network:** This parameter specifies the IP address of the communications module that is connected to the primary network.
- **IP of CM on Secondary Network:** This parameter specifies the IP address of the communications module that is connected to the secondary network. It will be available if network redundancy is enabled.

Note: The primary and secondary IP addresses must be different. It is recommended that the primary and secondary networks have different subnet addresses.

See Also: Channel Setup and **Network Redundancy**.

Communications
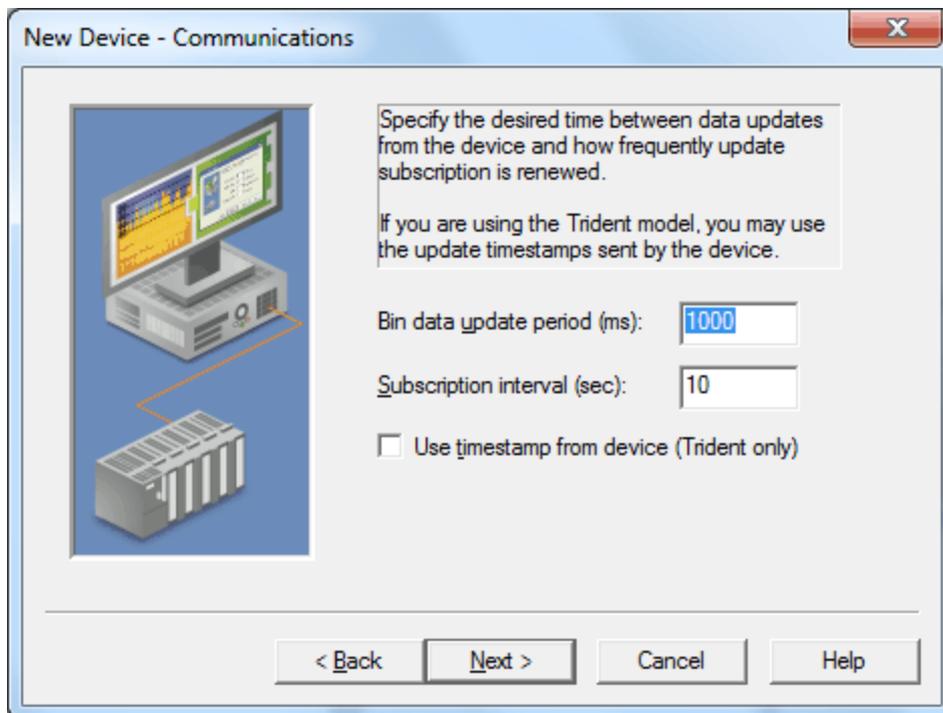
Descriptions of the parameters are as follows:

- **Bin Data Update Period (ms):** This parameter specifies the amount of time between updates. Instead of polling the device for data, the driver requests that the device send data updates at a given interval. Values for entire bins will be sent on each update. This method has proven to be more efficient with the Triconex controllers. The valid range is 50 to 10000 milliseconds. The default setting is 1000. For more information, refer to "Bin Data Update Period Notes" below.

- **Subscription Interval (sec):** This parameter specifies the frequency at which the driver will renew its subscription for data updates. Subscriptions must be renewed at least every 120 seconds, or data updates will stop. The valid range is 10 to 120 seconds. The default setting is 10.

  Note: The default subscription interval setting should be used when a TR_PORT_STATUS function block is being utilized in the device configuration. This will ensure that the driver sends something to the device every 10 seconds, and will maintain the function block's internal receive status flag.

- **Use Timestamp From Device (Trident Only):** When checked, this option uses the timestamp sent from the device with data updates (instead of the timestamp associated with data that is normally set by the server as the time of receipt). This option is supported by Trident devices only. Timestamps will always be set by the server when using the Tricon model.

Bin Data Update Period Notes

1. When the driver services a Read request from the client application, it will return the last value sent by the device. The data quality will be "Good" if the last update was received within 2 times the update

interval, "Uncertain" if the cached data is stale, or "Bad" until the first update is received.

2. If the device cannot send updates at this rate (or if it is configured with a slower scan rate), the data will most likely be flagged as stale between updates. For more information on how the driver processes data updates when network redundancy is enabled, refer to **Network Redundancy**.

3. The Bin Data Update Period is a requested value. If the device is unable to deliver updates at the requested rate, the OPC data quality may fluctuate between updates and network errors may be reported. It is recommended that this value be at least twice the device scan period. If data from many bins is requested (or if a large amount of memory is allocated for some bins), a larger margin may be required due to message segmentation. Each update contains all of the values for each bin referenced in the client. The device will break update messages into 500 byte segments and may take up to 20 milliseconds to send each segment. For more information, refer to **Driver System Tags**.

4. If reducing this setting, keep in mind that the device continues to send updates for two minutes after the last request was made. As such, the update rate may not change for up to two minutes even if new subscriptions are sent.

Tag Import
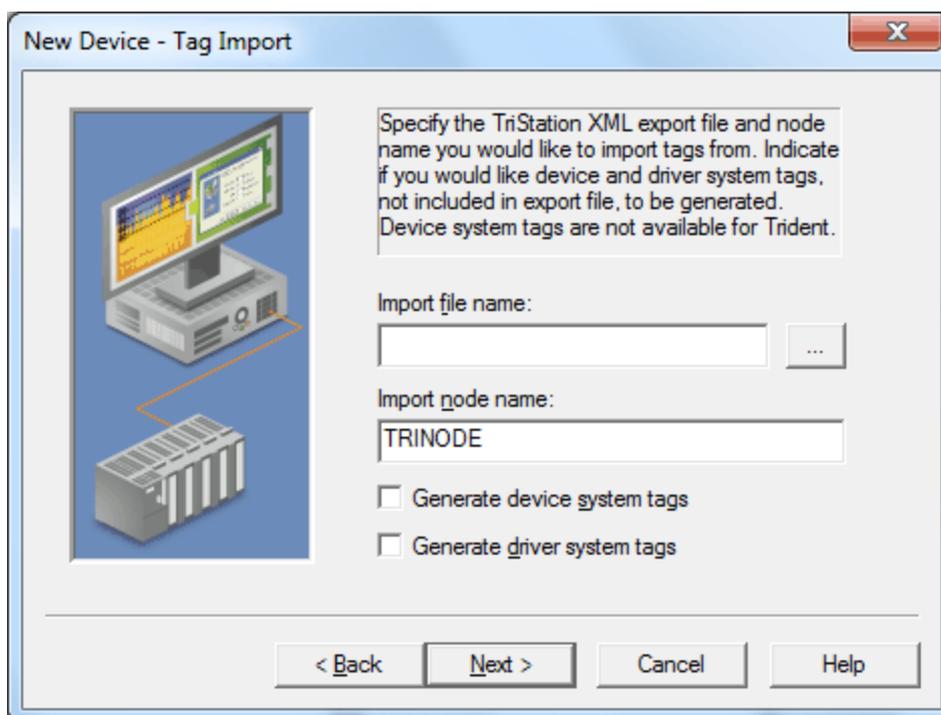
Descriptions of the parameters are as follows:

- **Import File Name:** This parameter specifies the name and location of the TriStation export file from which data will be imported.

  Note: Before the driver can automatically create tags for all of the variables defined in the TriStation project, users must export the project from TriStation in XML format. The XML export file may contain data for multiple devices, each with a unique node name.

- **Import Node Name:** This parameter specifies the node name associated with the tags that will be imported.

- **Generate Device System Tags:** When checked, this option will have the driver generate a standard set of the most useful System Tags. The TriStation export file does not contain device System Tags.

  Note: Users do not need to specify an import file in order to generate device System Tags. Data for these device System Tags is contained in bins 8 and 9. Since the Trident controllers do not support these bins, this option will not be enabled if the Trident model is selected.

- **Generate Driver System Tags:** When checked, the driver will automatically generate a number of System Tags that can be used to monitor driver operation and network health. Users do not need to specify an import file in order to generate driver System Tags. For a list of tags, refer to **Driver System Tags**.

See Also: Automatic Tag Database Generation

Cable Diagrams For Tricon

Tricon Controller with NCM
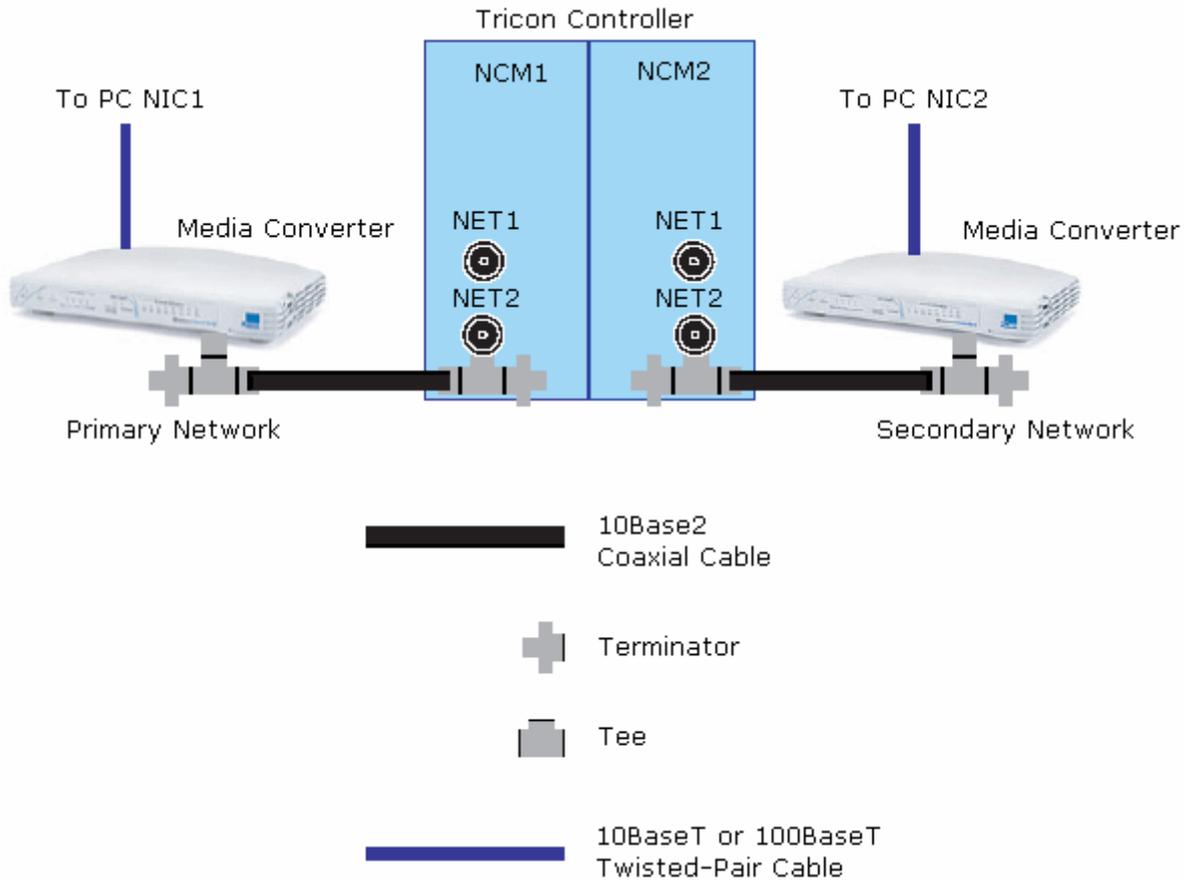
Note: Users must connect to NET2.

Tricon Controller with TCM (Tricon Communications Module)

Tricon Controller — TCM

To PC NIC1 — Hub — Primary Network
To PC NIC2 — Hub — Secondary Network

NET1 Left
NET2 Left
NET1 Right
NET2 Right

10BaseT or 100BaseT Twisted-Pair Cable
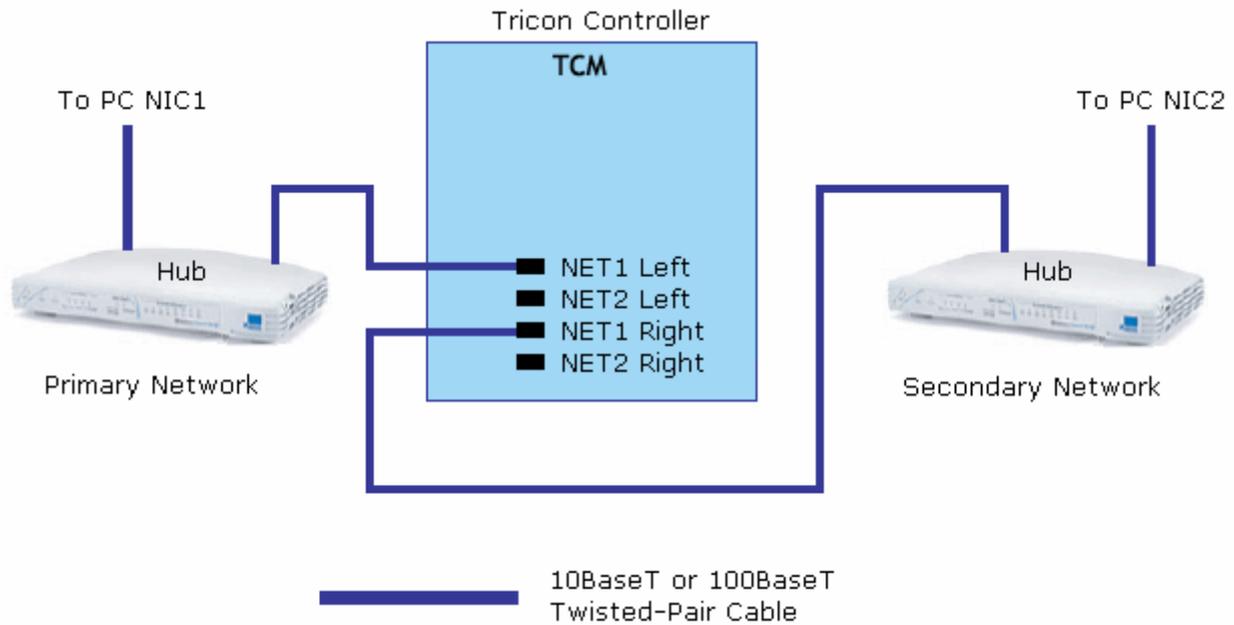
Note: For TCM models 4351A/4351B and 4352A/4352B, connect to NET1 or NET2. For TCM models 4353 and 4354, connect to NET1.

See Also: **Network Redundancy**

Cable Diagram For Trident

Trident Controller — CM

To PC NIC1 — Hub — Primary Network
To PC NIC2 — Hub — Secondary Network

NET1 Left
NET2 Left
NET1 Right
NET2 Right

10BaseT or 100BaseT Twisted-Pair Cable
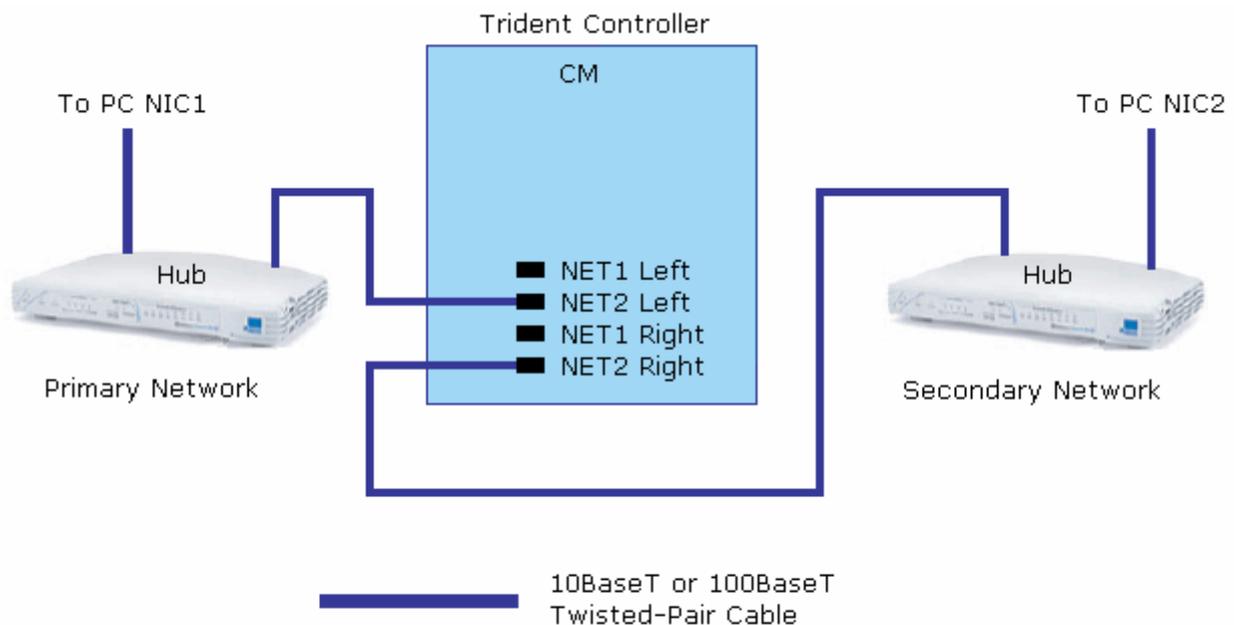
Note: Users can connect to NET1 or NET2. NET1 supports 10BaseT only. NET2 supports 10BaseT and 100BaseT.

See Also: Network Redundancy

Network Redundancy

Network Configuration

Network Redundancy requires that each controller be equipped with two communications modules and that the computer running this driver be equipped with two network interface cards. It is recommended that the primary and secondary networks have different subnet addresses. For more information, refer to **Channel Setup** and **CM Configuration**.

Media converters are usually required in order to connect Tricon controllers (which need a 2BaseT coaxial cable) to the NICs. NICs usually need 10BaseT or 100BaseT twisted pair cables. Trident controllers need 10BaseT and 100BaseT twisted pair cables, which can be connected to the NIC via a hub. Triconex recommends combination media converter/hubs from 3Com. A redundant network mixing Tricon and Trident controllers would appear similar to the one shown below.
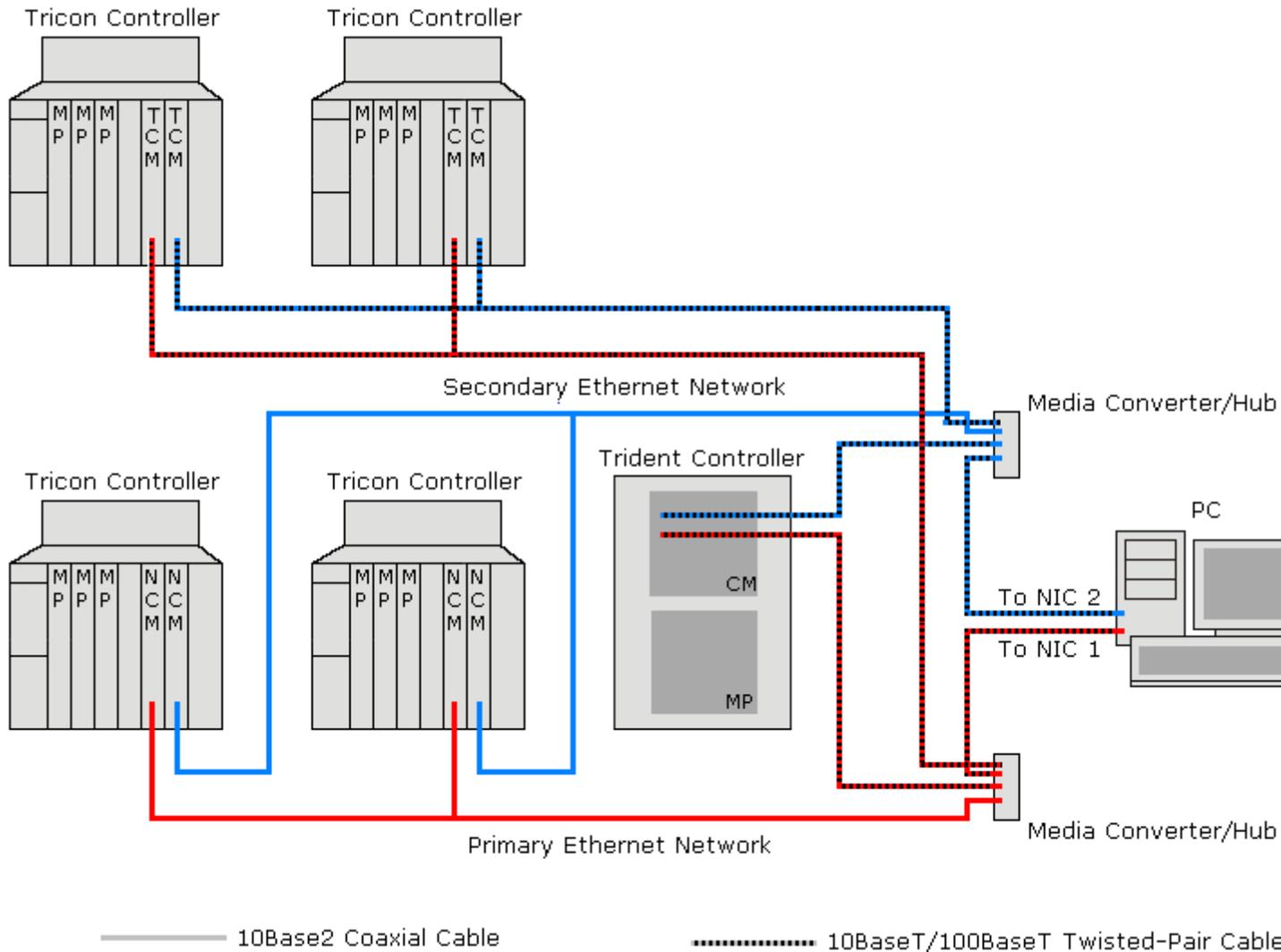
See Also: Cable Diagrams for Tricon and **Cable Diagram for Trident**.

Reads with Network Redundancy Enabled

This driver does not poll the device for data. Instead, it instructs the controller to send data updates at a given interval. Values will be sent for entire bins on each update. This method has proven to be more efficient with the Triconex controllers. For more information, refer to **Communications**.

When network redundancy is enabled, the driver will request that the device send data updates on both networks. The driver will process updates from a single network however. If an update fails to arrive on that network within two update intervals, the driver will begin processing updates from the other network. Therefore, fresh data may not be available for up to three update periods following a network problem. The UpdatesFromPrimary

driver System Tags can be used to monitor the network from which the driver is processing updates.

Though the driver processes updates from a single network, it does monitor incoming traffic from both networks. If nothing has been received on a network for over two update periods, the driver will flag that network as being in error. The PrimaryNetworkError and SecondaryNetworkError driver System Tags can be used to monitor the network error state.

See Also: Driver System Tags

Writes with Network Redundancy Enabled
The driver will send a Write request over a single network, which network will be the primary unless the primary is known to be in error and the secondary is not. In that case, the request will go out the secondary. If a Write acknowledgement is not received within the request timeout period, the request will be retried on the other network. If a Write acknowledgement is still not received after another request timeout period, the driver will send the request again to both networks. The driver will continue to retry the request on both networks until an acknowledgement is received or the maximum number of retry attempts have been performed.

Note: The device may take up to four scan cycles to complete a write request. The controller may be programmed to use a scan period as large as 400 milliseconds. Setting the request timeout too low may cause the driver to assume a Write has failed when it hasn't been processed by the controller yet.

Optimizing Your Triconex Ethernet Communications

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

The server refers to communications protocols like Triconex 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 must then be defined under that channel. Each of these devices represents a single Triconex controller from which data will be collected. While this approach to defining the application will provide a high level of performance, it will not take advantage of the Triconex Ethernet Driver or the network.

It's important to understand how the Triconex Ethernet Driver reads and writes data. This driver is unique in the two following ways:

1. Instead of sending Read requests to the devices, the driver tells each device to send periodic data updates whose data is then stored in cache. When the server asks the driver to read a piece of data, the driver will return the last value stored in cache. This method is more efficient than polling for Triconex controllers.

2. The Triconex Ethernet Driver can service Write requests to multiple devices on a channel simultaneously. The driver must, however, wait for the last Write request pending on that channel to be completed before allowing the next set of writes to be issued (or any Reads to be performed). Since a Triconex device can take up to 4 scan cycles to complete a Write, and the scan period can be as large as 400 milliseconds, Writes can have a tremendous effect on overall performance.
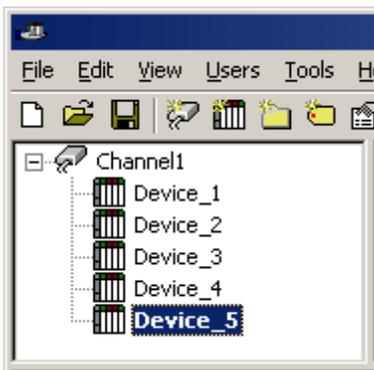
Each device appears under a single Triconex Ethernet channel. In this configuration, the driver must move from one device to the next as quickly as possible in order to gather information at an effective rate. As more devices are added or more information is requested from a single device, the overall update rate begins to suffer.

If the Triconex Ethernet Driver could only define one single channel, then the example shown above would be the only option available; however, the Triconex Ethernet Driver can define up to 32 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.
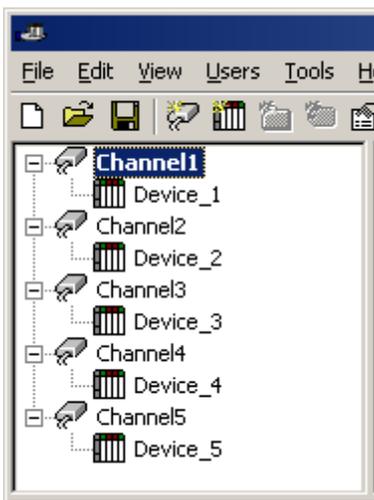
Each device has now been defined under its own channel. In this new configuration, a single path of execution is dedicated to the task of gathering data from each device. If the application has 32 or fewer devices, it can be optimized exactly how it is shown here.

The performance will improve even if the application has more than X32 devices. While 32 or fewer devices may be ideal, the application will still benefit from additional channels. Although spreading the device load across all channels will cause the server to move from device to device again, it can now do so with far less devices to process on a single channel.

Because Writes control the overall performance of the Triconex Ethernet Driver, it is important to consider how the client application writes to the server. If many writes must be performed, it may be better to send them at the

same time so that the driver can block them together. As previously mentioned, writes can be sent to multiple devices even if they are on the same channel. Writes to multiple points in a given device can also be blocked in a single request, if the client issues them at nearly the same time. Users should be familiar with the server's Write Optimization features, which can be accessed from the Channel Properties.

Note: For an application where writes never occur, there is little advantage to distributing the devices over multiple channels because the devices are not polled.

Automatic Tag Database Generation

The Triconex Ethernet Driver can automatically create tags for all of the variables in the TriStation project. For information on how to automatically generate tags, refer to the instructions below.

1. To start, export the project from TriStation using XML Format. Then, save the file on the computer that will be running the driver.

2. Next, create a new Triconex OPC server project or load an existing project.

3. Then, right-click on the device for which tags will be generated and select **Properties** | **Tag Import**. Enter the file's name and path, and specify the node name associated with the tags that will be imported.

4. Click **OK**.

5. Next, click **Device Properties** | **Database Creation** and then select **Auto Create**. The driver will read the XML file and then generate the required tags.

6. Once finished, save the OPC server project.

See Also: Tag Import

Data Types Description

Data Type	Description
Boolean	Single bit
Long	Signed 32 bit value bit 0 is the low bit bit 30 is the high bit bit 31 is the sign bit
DWord	Unsigned 32 bit value bit 0 is the low bit bit 31 is the high bit
Float	32 bit real

Address Descriptions

Address specifications vary depending on the model in use. To obtain address information for the model of interest, select a link from the list below.

Tricon Addressing
Trident Addressing

Note: The driver has several System Tags that can be used to monitor the driver's operation. For more information, refer to **Driver System Tags**.

Tricon Addressing

The Tricon model supports the following addresses. Memory may be addressed using either *Bin:Offset* syntax or Modbus aliases. The default data types for dynamically defined tags are shown in **bold** where appropriate.

Bin:Offset Addressing

Variable Type	Address Range	Data Type	Access
Output (BOOL)	00:0000-00:1999	Boolean	Read/Write
Memory (BOOL)	01:0000-01:1999	Boolean	Read/Write
Input (BOOL)	02:0000-02:1999	Boolean	Read Only
Memory (BOOL)	03:0000-03:1999	Boolean	Read Only
Input (REAL)	04:0000-04:0999 04:xxxx.00-04:xxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (DINT)	05:0000-05:0999 05:xxxx.00-05:xxxx.31	**DWord**, Long, Float Boolean	Read Only
Input (REAL)	06:0000-06:0119 06:xxxx.00-06:xxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (REAL)	07:0000-07:0999 07:xxxx.00-07:xxxx.31	DWord, Long, **Float** Boolean	Read Only
System Status (BOOL)	08:0000-08:5998	Boolean	Read Only
System Status (DINT)	09:0000-09:0368 09:xxxx.00-09:xxxx.31	**DWord**, Long, Float Boolean	Read Only
Output (REAL)	10:0000-10:0249 10:xxxx.00-10:xxxx.31	DWord, Long, **Float** Boolean	Read/Write
Memory (DINT)	11:0000-11:0749 11:xxxx.00-11:xxxx.31	**DWord**, Long, Float Boolean	Read/Write
Memory (REAL)	12:0000-12:0999 12:xxxx.00-12:xxxx.31	DWord, Long, **Float** Boolean	Read/Write

Modbus Alias Addressing

Variable Type	Address Range	Data Type	Access
Output (BOOL)	00001-02000	Boolean	Read/Write
Memory (BOOL)	02001-04000	Boolean	Read/Write
Input (BOOL)	10001-12000	Boolean	Read Only
Memory (BOOL)	12001-14000	Boolean	Read Only
Input (REAL)	30001-32000 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (DINT)	31001-32000 xxxxx.00-xxxxx.31	**DWord**, Long, Float Boolean	Read Only
Input (REAL)	32001-32120 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (REAL)	33001-34000 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
System Status (BOOL)	14001-19999	Boolean	Read Only
System Status (DINT)	39631-39999 xxxxx.00-xxxxx.15	**Word**, Short Boolean	Read Only
Output (REAL)	40001-40250 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read/Write

| Memory (DINT) | 40251-41000
xxxxx.00-xxxxx.31 | **DWord**, Long, Float
Boolean | Read/Write |
| Memory (REAL) | 41001-42000
xxxxx.00-xxxxx.31 | DWord, Long, **Float**
Boolean | Read/Write |

Note: DINT system status variables, bin 9 data, are addressed as 16-bit registers when using Modbus Aliases.
All other DINT and REAL variable types are addressed as 32 bit registers when using Modbus Aliases.

Examples
The following examples show the relationship between some Modbus Aliases and Bin:Offset addresses.

1. 39633 and 9:1 both address the system variable Year.

2. 39635 and 9:2 both address the system variable Month.

3. 39663 addresses the low word of the DINT value addressed by 9:16. This is the current control program version number system variable. The low word is the minor version number.

4. 39664 addresses the high word of the DINT value addressed by 9:16. This is the current control program version number system variable. The high work is the major version number.

5. 00001 and 0:0 both address the same BOOL output.

6. 00002 and 0:1 both address the same BOOL output.

7. 30001 and 4:0 both address the same REAL input.

8. 30002 and 4:1 both address the same REAL input.

Write Requests
For the device to process Write requests, the following statements must be true.

1. The variables must be defined as Read/Write.

2. For Tricon, the keyswitch must be in the Program or Remote position. For Trident, the Remote mode must be enabled by setting a system attribute.

3. The TriStation configuration setting (Disable Remote Changes to Outputs) which was downloaded to the controller must allow remote changes.

   Note: This setting only affects changes to discrete outputs and analog outputs.

4. The TriStation configuration for the ACM or NCM (Tricon) and CM (Trident) must be configured as Read/Write.

If the controller cannot write the data, it sends as reject response coded. For more information, refer to **Write failed for tag '<address>' on device '<device>'. TSAA Response Code: <code>**.

Basic Address Example
Users may address values using the *Bin:Offset* syntax or with a Modbus Alias.

10:0012 addresses the bin 10 float value with offset 12.
The same value can be addressed using the Modbus Alias 40013.

Bit Address Example
Individual bits in DINT types may be addressed by appending the "dot bit number" to a basic address.

05:0010.2 addresses bit 2 of the bin 5 DINT value with offset 10.
The same bit can be addressed using the Modbus Alias 31011.2.

Array Address Examples
Multiple data points may be accessed as array data.

1. 05:0010 [4] addresses bin 5 DINT values with offsets 10, 11, 12, and 13.
The same range of values can be addressed using the Modbus Alias 31011 [4].

2. 05:0010 [4][5] addresses bin 5 DINT values with offsets from 10, to 29 as a 2-D array.
The same range of values can be addressed using the Modbus Alias 31011 [4][5].

Trident Addressing

The Trident model supports the following addresses. Memory can be addressed using either *Bin:Offset* syntax or Modbus aliases. The default data types for dynamically defined tags are shown in **bold** where appropriate.

Bin:Offset Addressing

Variable Type	Address Range	Data Type	Access
Output (BOOL)	00:0000-00:1999	Boolean	Read/Write
Memory (BOOL)	01:0000-01:1999	Boolean	Read/Write
Input (BOOL)	02:0000-02:1999	Boolean	Read Only
Memory (BOOL)	03:0000-03:1999	Boolean	Read Only
Input (REAL)	04:0000-04:0999 04:xxxx.00-04:xxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (DINT)	05:0000-05:0999 05:xxxx.00-05:xxxx.31	**DWord**, Long, Float Boolean	Read Only
Input (REAL)	06:0000-06:0119 06:xxxx.00-06:xxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (REAL)	07:0000-07:0999 07:xxxx.00-07:xxxx.31	DWord, Long, **Float** Boolean	Read Only
Output (REAL)	10:0000-10:0249 10:xxxx.00-10:xxxx.31	DWord, Long, **Float** Boolean	Read/Write
Memory (DINT)	11:0000-11:0749 11:xxxx.00-11:xxxx.31	**DWord**, Long, Float Boolean	Read/Write
Memory (REAL)	12:0000-12:0999 12:xxxx.00-12:xxxx.31	DWord, Long, **Float** Boolean	Read/Write

Modbus Alias Addressing

Variable Type	Address Range	Data Type	Access
Output (BOOL)	00001-02000	Boolean	Read/Write
Memory (BOOL)	02001-04000	Boolean	Read/Write
Input (BOOL)	10001-12000	Boolean	Read Only
Memory (BOOL)	12001-14000	Boolean	Read Only
Input (REAL)	30001-32000 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (DINT)	31001-32000 xxxxx.00-xxxxx.31	**DWord**, Long, Float Boolean	Read Only
Input (REAL)	32001-32120 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
Memory (REAL)	33001-34000 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read Only
Output (REAL)	40001-40250 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read/Write
Memory (DINT)	40251-41000 xxxxx.00-xxxxx.31	**DWord**, Long, Float Boolean	Read/Write
Memory (REAL)	41001-42000 xxxxx.00-xxxxx.31	DWord, Long, **Float** Boolean	Read/Write

Note: All DINT and REAL variable types are addressed as 32 bit registers when using Modbus Aliases.

Examples

The following examples show the relationship between some Modbus Aliases and Bin:Offset addresses.

1. 00001 and 0:0 both address the same BOOL output.

2. 00002 and 0:1 both address the same BOOL output.

3. 30001 and 4:0 both address the same REAL input.

4. 30002 and 4:1 both address the same REAL input.

Write Requests

For the device to process Write requests, the following statements must be true.

1. The variables must be defined as Read/Write.

2. For Tricon, the keyswitch must be in the Program or Remote position. For Trident, the Remote mode must be enabled by setting a system attribute.

3. The TriStation configuration setting (Disable Remote Changes to Outputs) which was downloaded to the controller must allow remote changes.

   Note: This setting only affects changes to discrete outputs and analog outputs.

4. The TriStation configuration for the ACM or NCM (Tricon) and CM (Trident) must be configured as Read/Write.

If the controller cannot write the data, it sends as reject response coded. For more information, refer to **Write failed for tag '<address>' on device '<device>'. TSAA Response Code: <code>**.

Basic Address Example
Users may address values using the *Bin:Offset* syntax or with a Modbus Alias.

10:0012 addresses the bin 10 float value with offset 12.
The same value can be addressed using the Modbus Alias 40013.

Bit Address Example
Individual bits in DINT types may be addressed by appending the "dot bit number" to a basic address.

05:0010.2 addresses bit 2 of the bin 5 DINT value with offset 10.
The same bit can be addressed using the Modbus Alias 31011.2.

Array Address Examples
Multiple data points may be accessed as array data.

1. 05:0010 [4] addresses bin 5 DINT values with offsets 10, 11, 12, and 13.
The same range of values can be addressed using the Modbus Alias 31011 [4].

2. 05:0010 [4][5] addresses bin 5 DINT values with offsets from 10, to 29 as a 2-D array.
The same range of values can be addressed using the Modbus Alias 31011 [4][5].

Driver System Tags
Tags may be created with the following addresses to monitor driver operation.

Address	Data Type	Access	Description
PrimaryNetworkError	Boolean	Read Only	Set to 0 if the state of the primary network appears to be good or is uncertain. Set to 1 if the state of the primary network appears to be bad. Set when an expected message is not received.
SecondaryNetworkError	Boolean	Read Only	Set to 0 if the state of the secondary network appears to be good or is uncertain. Set to 1 if the state of the secondary network appears to be bad. Set when an expected message is not received or network redundancy is not enabled.
UpdatesFromPrimary	Boolean	Read Only	Set to 1 when the driver is processing data updates from the primary network.*
UpdatesFromSecondary	Boolean	Read Only	Set to 1 when the driver is processing data updates from the secondary network.**

*Updates may also be coming in on the secondary network, but are ignored by the driver.
**Updates may also be coming in on the primary network, but are ignored by the driver.

Error Descriptions

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

Address Validation
Address '<address>' is out of range for the specified device or register
Data Type '<type>' is not valid for device address '<address>'
Device address '<address>' contains a syntax error
Device address '<address>' is not supported by model '<model name>'
Device address '<address>' is Read Only
Missing address

Device Status Messages
Could not detect available network adapters for <primary/secondary> network
Device '<device name>' is not responding
Failed to load Winsock or related library
Tag import error - invalid bin in item path: '<item path>'
Tag import error - invalid item path format: '<item path>'
Tag import error - invalid name: '<tag name>'
Tag import error - invalid offset in item path: '<item path>'
Tag import failed. Could not find data for node '<node>'
Tag import failed. Could not open file '<file>'
Tag import failed due to unexpected XML format in '<file>'
Tag import failed. File name given, but no node name
Tag import failed. No import file name or system tags specified
Unable to bind to local address (<IP, port>)
Unable to process data update on device '<device>'. CRC error
Unable to process data update on device '<device>'. Unexpected message format
Unable to process data update on device '<device>'. Unexpected model
Unable to process data update on device '<device>'. Unexpected node number
Unable to use network redundancy on channel '<channel>'
Unable to use specified primary network adapter '<adapter>' on channel '<channel>'. Using '<adapter>'
Unable to use specified secondary network adapter '<adapter>' on channel '<channel>'. Using '<adapter>'
Winsock initialization failed (OS Error = <error code>)
Winsock V1.1 or higher must be installed to use the Triconex Ethernet device driver
Write failed for tag '<address>' on device '<device>'. CRC error
Write failed for tag '<address>' on device '<device>'. No data for read/modify/write
Write failed for tag '<address>' on device '<device>'. Timeout
Write failed for tag '<address>' on device '<device>'. TSAA Response Code: <code>
Write failed for tag '<address>' on device '<device>'. Unexpected model
Write failed for tag '<address>' on device '<device>'. Unexpected node number
Write failed for tag '<address>' on device '<device>'. Unexpected response format

Address Validation

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

Address Validation
Address '<address>' is out of range for the specified device or register
Data Type '<type>' is not valid for device address '<address>'
Device address '<address>' contains a syntax error
Device address '<address>' is not supported by model '<model name>'
Device address '<address>' is Read Only
Missing address

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

Error Type:
Warning

Possible Cause:

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

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

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

Error Type:
Warning

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

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

Device address '<address>' contains a syntax error

Error Type:
Warning

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

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

Device address '<address>' is not supported by model '<model name>'

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically references a location that is valid for the communications protocol but not supported by the target device.

Solution:
1. Verify that the address is correct; if it is not, re-enter it in the client application.
2. Verify that the selected model name for the device is correct.

Device address '<address>' is Read Only

Error Type:
Warning

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

Solution:
Change the access mode in the client application.

Missing address

Error Type:
Warning

Possible Cause:
A tag address that has been specified dynamically has no length.

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

Device Status Messages

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

Device Status Messages
Could not detect available network adapters for <primary/secondary> network
Device '<device name>' is not responding
Failed to load Winsock or related library
Tag import error - invalid bin in item path: '<item path>'
Tag import error - invalid item path format: '<item path>'
Tag import error - invalid name: '<tag name>'
Tag import error - invalid offset in item path: '<item path>'
Tag import failed. Could not find data for node '<node>'
Tag import failed. Could not open file '<file>'
Tag import failed due to unexpected XML format in '<file>'
Tag import failed. File name given, but no node name
Tag import failed. No import file name or system tags specified
Unable to bind to local address (<IP, port>)
Unable to process data update on device '<device>'. CRC error
Unable to process data update on device '<device>'. Unexpected message format
Unable to process data update on device '<device>'. Unexpected model
Unable to process data update on device '<device>'. Unexpected node number
Unable to use network redundancy on channel '<channel>'
Unable to use specified primary network adapter '<adapter>' on channel '<channel>'. Using '<adapter>'
Unable to use specified secondary network adapter '<adapter>' on channel '<channel>'. Using '<adapter>'
Winsock initialization failed (OS Error = <error code>)
Winsock V1.1 or higher must be installed to use the Triconex Ethernet device driver
Write failed for tag '<address>' on device '<device>'. CRC error
Write failed for tag '<address>' on device '<device>'. No data for read/modify/write
Write failed for tag '<address>' on device '<device>'. Timeout
Write failed for tag '<address>' on device '<device>'. TSAA Response Code: <code>
Write failed for tag '<address>' on device '<device>'. Unexpected model
Write failed for tag '<address>' on device '<device>'. Unexpected node number
Write failed for tag '<address>' on device '<device>'. Unexpected response format

Could not detect available network adapters for <primary/secondary> network

Error Type:
Serious

Possible Cause:
The network adapters are not connected to a hub or other device.

Solution:
The network adapters must be connected to a device or hub in order for Windows to correctly report their configuration to the driver. Verify the network connection.

Device '<device name>' is not responding

Error Type:
Serious

Possible Cause:
1. The network connection between the device and the Host PC is broken.
2. The named device may have been assigned an incorrect IP, Device ID, or model.
3. An expected data update was late.
4. The response from the device took longer to receive than the amount of time specified in the Request Timeout.

Solution:
1. Verify the cabling between the PC and the PLC.

2. Verify the Device Properties.

3. Consider increasing the Bin Data Update Period if the device is not capable of sending updates at the requested rate.

4. Increase the Request Timeout setting so that the entire response can be handled.

See Also:
Communications

Failed to load Winsock or related library

Error Type:
Fatal

Possible Cause:
The driver could not load Winsock (winsock.dll) or the IP Helper API dll (iphlpapi.dll).

Solution:
These libraries are normally installed with Windows. Verify that these files exist in the system folder.

Tag import error - invalid bin in item path: '<item path>'

Error Type:
Warning

Possible Cause:
An entry in the import file had an item path with an invalid bin number.

Solution:
Verify that the item path has the format *<node name>:<bin>:<offset>* and that the bin number is between 0 and 12.

Tag import error - invalid item path format\: '<item path>'

Error Type:
Warning

Possible Cause:
An entry in the import file had an unexpected item path format.

Solution:
Verify that the item paths in the export file have the format *<node name>:<bin>:<offset>*.

Tag import error - invalid name: '<tag name>'

Error Type:
Warning

Possible Cause:
The PST alias name value does not conform to the server's tag naming rules.

Solution:
1. Verify that the name contains only alphanumeric characters and underscores.
2. Verify that the first character is not an underscore, editing the import file if necessary.

Note:
There should be two entries for each tag. The first entry gives the tag name and the second gives the Modbus alias. The driver skips the entries with Modbus alias values.

Tag import error - invalid offset in item path: '<item path>'

Error Type:
Warning

Possible Cause:
An entry in the import file had an item path with an invalid offset value.

Solution:
Verify that the item path has the format *<node name>:<bin>:<offset>* and that the offset is within the valid range for the bin.

See Also:
Address Descriptions

Tag import failed. Could not find data for node '<node>'

Error Type:
Warning

Possible Cause:
The driver did not find any entires in the import file with the specified node name.

Solution:
Verify that the correct node name has been specified.

Note:
The node name is given by the PSTAliasGroup name.

Tag import failed. Could not open file '<file>'

Error Type:
Warning

Possible Cause:
The driver could not find or open the import file specified on the Tag Import page in Device Properties.

Solution:
1. Verify that the file exists and that the specified path is correct.
2. Make sure that no other application has opened the file.

See Also:
Tag Import

Tag import failed due to unexpected XML format in '<file>'

Error Type:
Warning

Possible Cause:
The driver could not process the import file because it did not have the expected format.

Solution:
Verify that the data is exported from TriStation in XML format.

Tag import failed. File name given, but no node name

Error Type:
Warning

Possible Cause:
No node name was specified on the Tag Import page in Device Properties.

Solution:
Specify the node name used in the import file.

See Also:
Tag Import

Tag import failed. No import file name or system tags specified

Error Type:

Warning

Possible Cause:

No tag import options were specified in the Tag Import page in Device Properties when the driver attempted to generate a tag database.

Solution:

Verify that an import file name is specified or that one of the system tag options is selected when tag generation is started.

See Also:

Tag Import

Unable to bind to local address (<IP, port>)

Error Type:

Fatal

Possible Cause:

The driver was not able to bind a socket to receive data sent to the specified IP and port.

Solution:

Make sure no other application is using that IP and port.

Unable to process data update on device '<device>'. CRC error

Error Type:

Warning

Possible Cause:

The driver rejected a data update message because its CRC checksum was not valid.

Solution:

This error implies noise in the network. If this error is persistent, efforts should be made to improve network reliability.

Unable to process data update on device '<device>'. Unexpected message format

Error Type:

Warning

Possible Cause:

The driver rejected a data update message because it did not have the expected format.

Solution:

If the problem persists and does not appear to be network related, perform a Channel Diagnostics capture of the incoming update messages and contact Technical Support.

Unable to process data update on device '<device>'. Unexpected model

Error Type:

Warning

Possible Cause:

The driver rejected a data update message because it did not have the expected model number.

Solution:

Verify the model setting.

See Also:

Device Setup

Unable to process data update on device '<device>'. Unexpected node number

Error Type:
Warning

Possible Cause:
The driver rejected a data update message because it did not have the expected node number.

Solution:
Verify the Device ID setting.

See Also:
Device Setup

Unable to use network redundancy on channel '<channel>'

Error Type:
Warning

Possible Cause:
1. The driver loaded a project that was created on a computer that had multiple network interface cards.
2. The network adapter cards are not connected to a device or hub.

Solution:
1. Reset the Network Interface in Channel Properties to be compatible with the host computer. In order to use network redundancy, the host computer must have two network interface cards (NICs) installed. Save the adjusted setting.
2. The network adapters must be connected to a device or hub in order for Windows to correctly report their configuration to the driver. Ensure that the adapters are connected.

See Also:
Channel Setup
Network Redundancy

Unable to use specified primary network adapter '<adapter>' on channel '<channel>'

Error Type:
Warning

Possible Cause:
1. The driver loaded a project that was created on another computer with different network interface hardware or IP address.
2. The network adapter cards are not connected to a device or hub.

Solution:
1. Reset the Network Interface in Channel Properties to be compatible with the host computer. In order to use network redundancy, the host computer must have two network interface cards (NICs) installed. Save the adjusted setting.
2. The network adapters must be connected to a device or hub in order for Windows to correctly report their configuration to the driver. Ensure that the adapters are connected.

See Also:
Channel Setup
Network Redundancy

Unable to use specified secondary network adapter '<adapter>' on channel '<channel>'

Error Type:
Warning

Possible Cause:

1. The driver loaded a project that was created on another computer with different network interface hardware or IP address.
2. The network adapter cards are not connected to a device or hub.

Solution:
1. Reset the Network Interface in Channel Properties to be compatible with the host computer. In order to use network redundancy, the host computer must have two network interface cards (NICs) installed. Save the adjusted setting.
2. The network adapters must be connected to a device or hub in order for Windows to correctly report the configuration to the driver. Ensure that the adapters are connected.

See Also:
Channel Setup
Network Redundancy

Winsock initialization failed (OS Error = <error code>)

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 V1.1 or higher must be installed to use the Triconex Ethernet device 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.

Write failed for tag '<address>' on device '<device>'. CRC error

Error Type:
Warning

Possible Cause:
The driver rejected a write complete message because its CRC checksum was not valid.

Solution:
This error implies noise in the network. If this error is persistent, efforts should be made to improve network reliability.

Write failed for tag '<address>' on device '<device>'. No data for Read/Modify/Write

Error Type:
Warning

Possible Cause:
Bit writes require a read/modify/write procedure. The last known value has bad quality.

Solution:
Ensure that the driver has had time to receive a data update for the value in question before bit writes are performed. Failure can occur before an initial data update has been received or following a network failure.

Write failed for tag '<address>' on device '<device>'. Timeout

Error Type:
Warning

Possible Cause:
The driver did not receive a write complete message after the request timeout period and number of retries.

Solution:
Triconex devices can take a long time to process Writes. Thus, a timeout error does not necessarily mean the device did not receive the write request or has failed to process it. Ensure that an ample request timeout period has been specified that is consistent with the device's scan rate.

Note:
Other reasons for timeout errors could be related to hardware or network failure.

See Also:
Device Setup

Write failed for tag '<address>' on device '<device>'. TSAA Response Code: <code>

Error Type:
Warning

Possible Cause:
The device could not complete the write request for the reason given by the response code.

Solution:
Proper solution depend on failure reason. For more information, refer to the table below.

Code	Description
1	No buffer available to process the request.
2	Bin number specified in the request was out of range.
3	The Tricon NCM or Trident CM module is busy processing precious request and cannot accept another request. This can happen if more than four write request are outstanding at one time.
4	No MP is running.
5	TSX has rejected the request. the Reason contains the specific reason.
6	Request to TSX timed out.
7	Invalid response from TX.
8	Message was too big.
9	Offset or number of values in the request was invalid.
10	No control program (TriStation applications).
11	Read Only port.
236	Bad SOE number.
237	Invalid SOE type.
238	Invalid SOE state.

Write failed for tag '<address>' on device '<device>'. Unexpected model

Error Type:
Warning

Possible Cause:
The driver rejected a write complete message because it did not have the expected model number.

Solution:
Verify the model setting.

See Also:
Device Setup

Write failed for tag '<address>' on device '<device>'. Unexpected node number

Error Type:
Warning

Possible Cause:
The driver rejected a write complete message because it did not have the expected node number.

Solution:
Verify the Device ID setting.

See Also:
Device Setup

Write failed for tag '<address>' on device '<device>'. Unexpected response format

Error Type:
Warning

Possible Cause:
The driver rejected a write complete message because it did not have the expected format.

Solution:
If the problem persists and does not appear to be network related, perform a Channel Diagnostics capture of the incoming update messages and contact Technical Support.

Index

O

R

T

U