Thermo Westronics Ethernet Driver Help

Table of Contents

Thermo Westronics Ethernet Driver Help

Help version 1.010

CONTENTS

Overview

The Thermo Westronics Ethernet Driver provides an easy and reliable way to connect Thermo Westronics Ethernet devices to OPC Client applications, including HMI, SCADA, Historian, MES, ERP and countless custom applications. It is intended for use with Thermo Westronics Ethernet devices.

Device Setup

Supported Devices

SV 100
SV 180
SV 180 (2.0) - version 2.0A or later

Communication Protocol

Modbus Ethernet

Connection Timeout

This parameter specifies the time that the driver will wait for a connection to be made with a device. Depending on network load, the connect time may vary with each connection attempt. The default setting is 3 seconds. The valid range is 1 to 60 seconds.

Request Timeout

This parameter specifies the time that the driver will wait on a response from the device before giving up and going on to the next request. Longer timeouts only affect performance if a device is not responding. The default setting is 1000 milliseconds. The valid range is 100 to 9999 milliseconds.

Retry Attempts

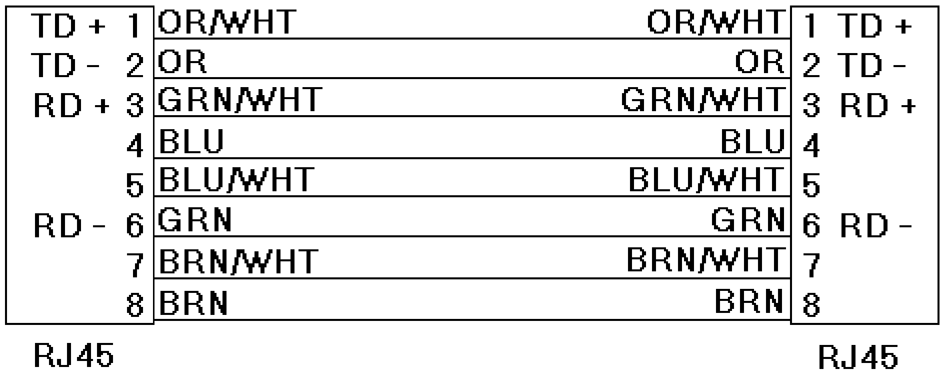
This parameter specifies the number of times that the driver will retry a message before giving up and going on to the next message. The default setting is 3 retries. The valid range is 1 to 10.

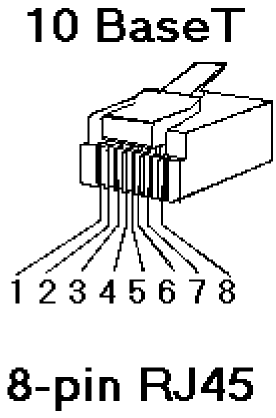
Device IDs

Device IDs are specified as YYY.YYY.YYY.YYY.XXX where YYY designates the device IP address and XXX designates the destination device unit number. Each YYY byte should be in the range of 0 to 255.

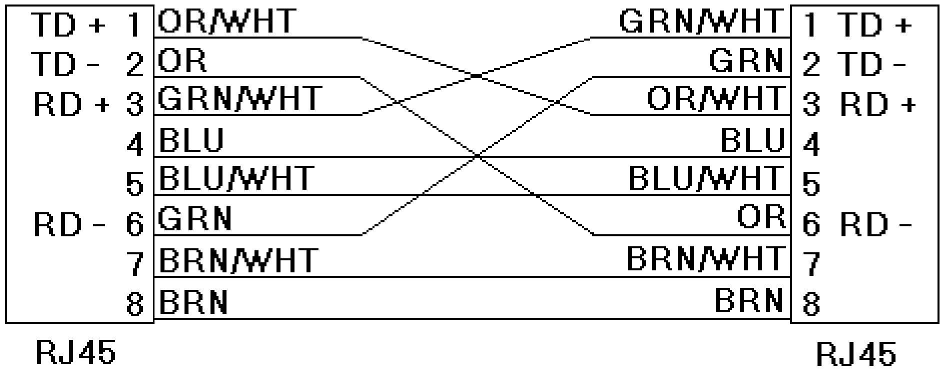
Cable Connections

Patch Cable (Straight Through)





Crossover Cable



TCP/IP

Port Number

This parameter specifies the TCP/IP port number the remote device is configured to use. The default port number is 502.

Block Sizes

Coil Block Sizes

Coils can be read from 8 to 800 points (bits) at a time.

Register Block Sizes

Registers can be read from 1 to 56 locations (16 or 32-bit registers) at a time. The default block size should be changed if the device does not support block Read/Write operations. It should also be changed if the device contains non-contiguous addresses. If this is the case and the driver attempts to read a block of data that encompasses undefined memory, the device will most likely reject the request.

Settings

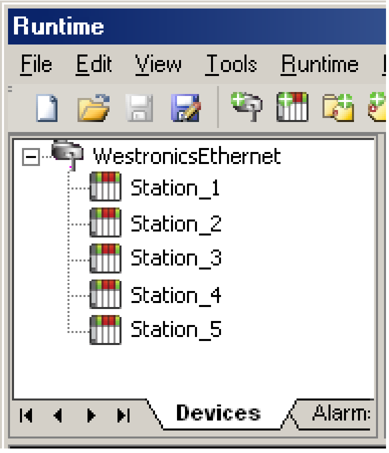
Use 1x32 Floating Point Format

Some Thermo Westronics devices, such as the SV100 and SV180, give users the option of representing floating point data in either 1x32 or 2x16 format in Modbus communications. This box should be checked when the device has been configured to use the 1x32 format.

Optimizing Your Thermo Westronics Ethernet Communications

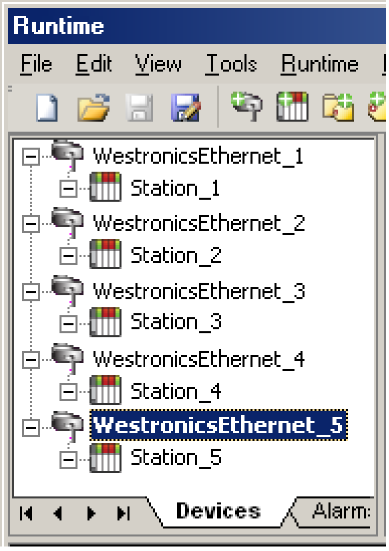
The Thermo Westronics Ethernet Driver has been designed to provide the best performance with the least amount of impact on the system's overall performance. While the Thermo Westronics Ethernet Driver is fast, there are a couple of guidelines that can be used in order to control and optimize the application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

This server refers to communications protocols like Thermo Westronics 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 Thermo Westronics Ethernet controller from which data will be collected. While this approach to defining the application will provide a high level of performance, it won't take full advantage of the Thermo Westronics 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 Thermo Westronics 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 Thermo Westronics Ethernet Driver could only define one single channel, then the example shown above would be the only option available; however, the Thermo Westronics Ethernet Driver can define up to 100 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 100 or fewer channels, it can be optimized exactly how it is shown here.

The performance will improve even if the application has more than 100 channels. While 100 or fewer channels may be ideal, the application will still benefit from additional channels. Although by 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.

Block Size, which is available on each defined device, can also affect the Thermo Westronics Ethernet Driver's performance. Block Size refers to the number of bytes that may be requested from a device at one time. To refine the performance of this driver, block size can be configured to 1-56 registers and 8-800 bits.

Data Types Description

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

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.

SV100
SV180
SV180 (2.0)

SV100 Addressing

The Thermo Westronics Ethernet Driver automatically generates tags for many of the most important device parameters. Tags can be manually created to access all device parameters mapped to Modbus address locations within the device. The following tables describe how to specify a Modbus address when creating a tag. The default data types for dynamically defined tags are shown in **bold**.

Note: The full range of addresses accepted by this driver may not be valid for each particular device.

Modbus Addressing Decimal Format

Address	Range	Data Type	Access
Output Coils	000001-065536	**Boolean**	Read/Write
Input Coils	100001-165536	**Boolean**	Read Only
Internal Registers	300001-305000 300001.0-300001.15...305000.0-305000.15 305001-306000 306001-307000 306001.0-306001.15...307000.0-307000.15 307001-308000 308001-365536 308001.0-308001.15...365536.0-365536.15	**Word**, Short **Boolean** **DWord**, Long **Word**, Short **Boolean** **Float** **Word**, Short **Boolean**	Read Only
Holding Registers	400001-406000 400001.0-400001.15...406000.0-406000.15 406001-407000 407001-408000 407001.0-407001.15...408000.0-408000.15 408001-409000 409001-465536 409001.0-409001.15...465536.0-465536.15	**Word**, Short **Boolean** **DWord**, Long **Word**, Short **Boolean** **Float** **Word**, Short **Boolean**	Read/Write

Modbus Addressing Hexadecimal Format

Address	Range	Data Type	Access
Output Coils	H000001-H010000	**Boolean**	Read/Write
Input Coils	H100001-H110000	**Boolean**	Read Only
Internal Registers	H300001-H301388 H300001.0-H300001.F...H301388.0-H301388.F H301389-H301770	**Word**, Short **Boolean** **DWord**, Long	Read Only

	H301771-H301B58	**Word**, Short	
	H301771.0-H301771.F...H301B58.0-H301B58.F	**Boolean**	
	H301B59-H301F40	**Float**	
	H301F41-H310000	**Word**, Short	
	H301F41.0-H301F41.F...H31000.0-H310000.F	**Boolean**	
Holding Registers	H400001-H401770	**Word**, Short	Read/Write
	H400001.0-H400001.F...H401770.0-H401770.F	**Boolean**	
	H401771-H401B58	**DWord**, Long	
	H401B59-H401F40	**Word**, Short	
	H401B59.0-H401B59.F...H401F40.0-H401F40.F	**Boolean**	
	H401F41-H402328	**Float**	
	H402329-H410000	**Word**, Short	
	H402329.0-H402329.F...H410000.0-H410000.F	**Boolean**	

Array Support

Arrays are supported for 16-bit internal and holding register locations for all data types except Boolean and strings. There are two methods of addressing an array. The examples below use holding register locations.

4xxxx [rows] [cols]
4xxxx [cols] (this method assumes rows is equal to one)

Rows multiplied by cols cannot exceed the block size that has been assigned to the device for the register type. For arrays of 32 bit data types, rows multiplied by cols multiplied by 2 cannot exceed the block size.

SV180 Addressing

The SV180 model is for SV180 devices using firmware versions prior to 2.0A. If the device uses firmware 2.0A or later, select the SV180 (2.0) model. For more information, refer to **SV180 (2.0) Addressing**.

The Thermo Westronics Ethernet Driver automatically generates tags for many of the most important device parameters. Tags can be manually created to access all device parameters mapped to Modbus address locations within the device. The following tables describe how to specify a Modbus address when creating a tag. The default data types for dynamically defined tags are shown in **bold**.

Note: The full range of addresses accepted by this driver may not be valid for each particular device.

Modbus Addressing Decimal Format

Address	Range	Data Type	Access
Output Coils	000001-065536	**Boolean**	Read/Write
Input Coils	100001-165536	**Boolean**	Read Only
Internal Registers	300001-305000	**Word**, Short	Read Only
	300001.0-300001.15...305000.0-305000.15	**Boolean**	
	305001-306000	**DWord**, Long	
	306001-307000	**Word**, Short	
	306001.0-306001.15...307000.0-307000.15	**Boolean**	
	307001-308000	**Float**	
	308001-365536	**Word**, Short	

	308001.0-308001.15...365536.0-365536.15	**Boolean**	
Holding Registers	400001-405000	**Word**, Short	Read/Write
	400001.0-400001.15...405000.0-405000.15	**Boolean**	
	405001-406000	**DWord**, Long	
	406001-407000	**Word**, Short	
	406001.0-406001.15...407000.0-407000.15	**Boolean**	
	407001-408000	**Float**	
	408001-465536	**Word**, Short	
	408001.0-408001.15...465536.0-465536.15	**Boolean**	

Modbus Addressing Hexadecimal Format

Address	Range	Data Type	Access
Output Coils	H000001-H010000	**Boolean**	Read/Write
Input Coils	H100001-H110000	**Boolean**	Read Only
Internal Registers	H300001-H301388	**Word**, Short	Read Only
	H300001.0-H300001.F ...H301388.0-H301388.F	**Boolean**	
	H301389-H301770	**DWord**, Long	
	H301771-H301B58	**Word**, Short	
	H301771.0-H301771.F ...H301B58.0-H301B58.F	**Boolean**	
	H301B59-H301F40	**Float**	
	H301F41-H310000	**Word**, Short	
	H301F41.0-H301F41.F ...H310000.0-H310000.F	**Boolean**	
Holding Registers	H400001-H401388	**Word**, Short	Read/Write
	H400001.0-H400001.F...H401388.0-H401388.F	**Boolean**	
	H401389-H401770	**DWord**, Long	
	H401771-H401B58	**Word**, Short	
	H401771.0-H401771.F... H401B58.0-H401B58.F	**Boolean**	
	H401B59-H401F40	**Float**	
	H401F41-H410000	**Word**, Short	
	H401F41.0-H401F41.F...H410000.0-H410000.F	**Boolean**	

Array Support

Arrays are supported for 16-bit internal and holding register locations for all data types except Boolean and strings. There are two methods of addressing an array. The examples below use holding register locations.

4xxxx [rows] [cols]
4xxxx [cols] (this method assumes rows is equal to one)

Rows multiplied by cols cannot exceed the block size that has been assigned to the device for the register type. For arrays of 32 bit data types, rows multiplied by cols multiplied by 2 cannot exceed the block size.

SV180 (2.0) Addressing

The SV180 (2.0) model is for SV180 devices using firmware version 2.0A or later. If the device uses an earlier firmware version, select the SV180 model. For more information, refer to **SV180 Addressing**.

The Thermo Westronics Ethernet Driver automatically generates tags for many of the most important device parameters. Tags can be manually created to access all device parameters mapped to Modbus address locations within the device. The following tables describe how to specify a Modbus address when creating a tag. The default data types for dynamically defined tags are shown in **bold**.

Note: The full range of addresses accepted by this driver may not be valid for each particular device.

Modbus Addressing Decimal Format

Address	Range	Data Type	Access
Output Coils	000001-065536	**Boolean**	Read/Write
Input Coils	100001-165536	**Boolean**	Read Only
Internal Registers	NA	NA	NA
Holding Registers	400001-465535*	**Float**	Read/Write

Modbus Addressing Hexadecimal Format

Address	Range	Data Type	Access
Output Coils	H000001-H010000	**Boolean**	Read/Write
Input Coils	H100001-H110000	**Boolean**	Read Only
Internal Registers	NA	NA	NA
Holding Registers	H400001-H410000*	**Float**	Read/Write

*Each value uses two registers when the device is configured to use the 2x16 floating point format. For example, Point 1 Data uses 400001 and 400002, Point 2 Data 2 uses 400003 and 400004 and so forth. Tags should address the first register used for the value. Each value uses one register when the device is configured to use the 1x32 floating point format. For example, Point 1 Data 1 uses 400001, Point 2 Data uses 400002 and so forth. For more information, refer to **Settings**.

Array Support

Arrays are supported for 16-bit internal and holding register locations for all data types except Boolean and strings. There are two methods of addressing an array. The examples below use holding register locations.

4xxxx [rows] [cols]
4xxxx [cols] (this method assumes rows is equal to one)

Rows multiplied by cols cannot exceed the block size that has been assigned to the device for the register type. For arrays of 32 bit data types, rows multiplied by cols multiplied by 2 cannot exceed the block size.

Error Descriptions

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

Address Validation

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

Device Status Messages

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

Thermo Westronics Ethernet Device Specific Messages

Failure to initiate 'Winsock.dll'
Bad address in block [<start address> to <end address>] on device '<device name>'

Address Validation

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

Address Validation

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

Missing address

Error Type:
Warning

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

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

Device address '<address>' contains a syntax error

Error Type:
Warning

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

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

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

Error Type:
Warning

Possible Cause:

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

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

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

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

Error Type:
Warning

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

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

Device address '<address>' is Read Only

Error Type:
Warning

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

Solution:
Change the access mode in the client application.

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

Error Type:
Warning

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

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

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

Error Type:
Warning

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

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

Device Status Messages

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

Device Status Messages

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

Device '<device name>' is not responding

Error Type:
Serious

Possible Cause:
1. The connection between the device and the Host PC is broken.
2. The communication parameters for the connection are incorrect.
3. The named device may have been assigned an incorrect Network ID.
4. The response from the device took longer to receive than the amount of time specified in the "Request Timeout" device setting.

Solution:
1. Verify the cabling between the PC and the device.
2. Verify that the specified communication parameters match those of the device.
3. Verify that the Network ID given to the named device matches that of the actual device.
4. Increase the Request Timeout setting so that the entire response can be handled.

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

Error Type:
Serious

Possible Cause:
1. The named device may not be connected to the network.
2. The named device may have been assigned an incorrect Network ID.
3. The named device is not responding to write requests.
4. The address does not exist in the PLC.

Solution:
1. Check the PLC network connections.
2. Verify that the Network ID given to the named device matches that of the actual device.

Thermo Westronics Ethernet Device Specific Messages

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

Thermo Westronics Ethernet Device Specific Messages

Failure to initiate 'Winsock.dll'
Bad address in block [<start address> to <end address>] on device '<device name>'

Failure to initiate 'Winsock.dll'

Error Type:
Fatal

Possible Cause:
Could not negotiate with the operating systems Winsock 1.1 functionality.

Solution:
Verify that the Winsock.dll is properly installed on the system.

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

Error Type:
Serious

Possible Cause:

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

Solution:

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

Index