Wago Ethernet Driver Help

Table of Contents

Wago Ethernet Driver Help

Help version 1.012

CONTENTS

Overview

The Wago Ethernet Driver provides an easy and reliable way to connect Wago Ethernet devices to OPC Client applications, including HMI, SCADA, Historian, MES, ERP and countless custom applications. It is intended for Wago I/O System 750 series of devices. Communication is through the 750-342 Buscoupler for Ethernet TCP/IP, or the 750-842 Programmable Fieldbus Controller (PFC) for Ethernet TCP/IP. Up to 64 I/O modules per device can be addressed. The server's automated OPC tag database generation feature frees the user from tedious manual tag definition, which would include slot configuration dependent address calculations.

Note 1: The driver will post messages when a failure occurs during operation. For more information, refer to **Error Descriptions**.

Note 2: TCP/IP must be properly installed in order to use this driver. For more information on setting up TCP/IP, refer to Windows documentation.

Device Setup

Supported Devices

Wago I/O System 750-342 Buscoupler for Ethernet TCP/IP.
Wago I/O System 750-842 Programmable Fieldbus Controller (PFC) for Ethernet TCP/IP.

Communication Protocol

Modbus Open Protocol over Ethernet using Winsock V1.1 or higher.

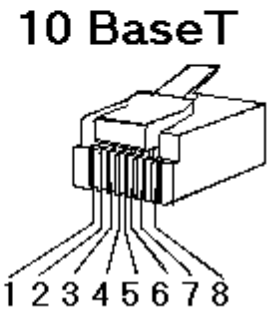
Device ID

Wago I/O System devices are networked using standard IP addressing. The Device ID has the following format YYY.YYY.YYY.YYY designating the device IP address. Each YYY byte should be in the range of 0 to 255.

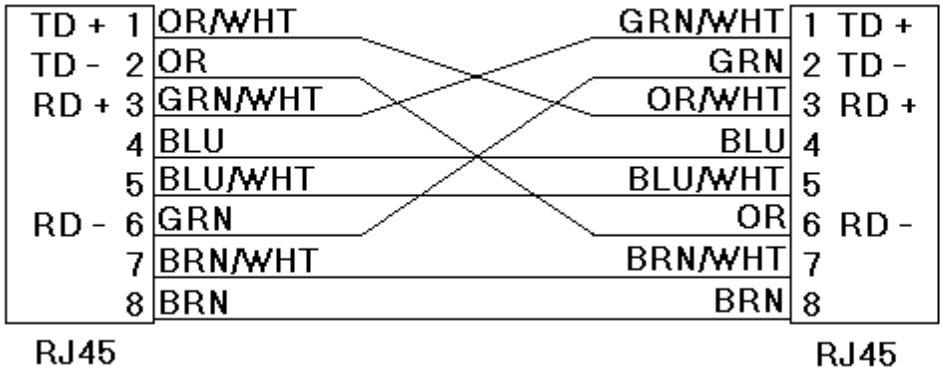
See Also: **TCP/IP**, **Block Sizes** and **Slot Configuration**.

Cable Diagrams

Patch Cable (Straight Through)



Crossover Cable



8-pin RJ45

TCP/IP

Port Number

This parameter specifies the TCP/IP port number that 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

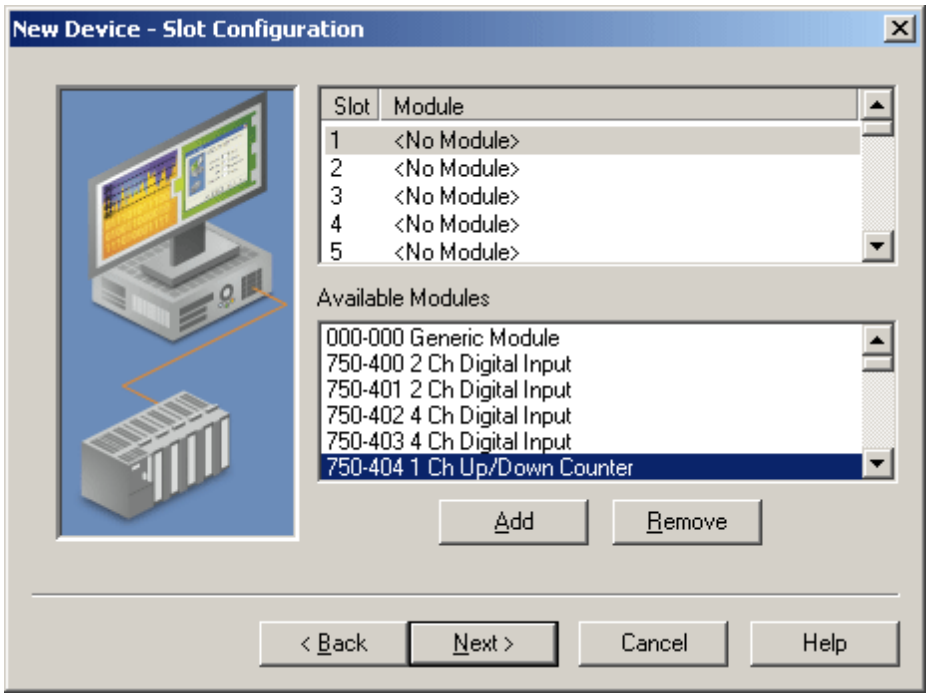
Registers can be read from 1 to 120 locations (words) at a time.

Note: The application may benefit by a change in block size. For example, future versions of the device may not support block Read/Write operations of the default size. Futhermore, if the device contains non-contiguous addresses (such as when a binary space module is used) the device will most likely reject a request to read a block of data that encompasses undefined memory.

Slot Configuration

Automated OPC tag database generation is a powerful labor saving feature of the server. However, before the server can create a tag database, the user must specify which modules are installed in the device. The slots may be configured during the last step of the Device Wizard. They may also be accessed by double-clicking on the device and then selecting the **Slot Configuration** tab.

Note: No modifications can be made to the slot configuration while client applications are connected to the device.



To configure the device, select a slot in the top window and then select the appropriate module type from the list below. To add a module to the slot, either double-click on it or select it and then click **Add**. If a slot was previously configured, the existing module type will be replaced by the new selection. To remove the module from the selected slot, click **Remove**.

Up to 64 slots can be configured per device. Slots may be configured as having <No Module> when the physical module occupying that slot does not contribute to the process image. Examples of such modules are the power supply module 750-600 and separation module 750-616.

Some choices, such as the binary spacer module (750-622), some **special modules** (750-650, 750-651, 750-653, and 750-654), and the generic module require configuration. A configuration dialog box will be presented when one of these modules is added. Once added to the slot configuration, users may go back and edit the module's properties by double-clicking on its slot.

Tags will be created automatically as soon as the Device Wizard is finished (or after **Apply** or **OK** is selected in Device Properties).

See Also:

Digital Input Modules
Digital Output Modules
Analog Input Modules
Analog Output Modules
Power Supply and End Modules
Encoder and Resolver Modules
Binary Spacer Module
Special Modules
Generic Module

Digital Input Modules

Module	Description	IC	OC	IR	HR	CB	SB
750-400	2 Ch Digital Input (DC 24 V, 3 ms)	2	0	0	0	0	0
750-401	2 Ch Digital Input (DC 24 V, 0.2 ms)	2	0	0	0	0	0
750-402	4 Ch Digital Input (DC 24 V, 3 ms)	4	0	0	0	0	0
750-403	4 Ch Digital Input (DC 24 V, 0.2 ms)	4	0	0	0	0	0
750-404	Up/Down Counter (32-bit)	0	0	2	2	1*	1*
750-404/001	Counter w/ Enable Input (32-bit)	0	0	2	2	1*	1*
750-404/002	Peak Time Counter (32-bit)	0	0	2	2	1*	1*
750-404/003	Frequency Counter (0.1-10 kHz)	0	0	2	2	1*	1*
750-404/004	Up/Down Counter w/ Switching	0	0	2	2	1*	1*
750-404/005	2 Ch Up Counter (16-bit)	0	0	2	2	1*	1*
750-405	2 Ch Digital Input (AC 230 V)	2	0	0	0	0	0
750-406	2 Ch Digital Input (AC 120 V)	2	0	0	0	0	0
750-408	4 Ch Digital Input (DC 24 V, 3 ms)	4	0	0	0	0	0
750-409	4 Ch Digital Input (DC 24 V, 0.2 ms)	4	0	0	0	0	0
750-410	2 Ch Digital Input (DC 24, 3 ms)	2	0	0	0	0	0
750-411	2 Ch Digital Input (DC 24 V, 02. ms)	2	0	0	0	0	0
750-412	2 Ch Digital Input (DC 48 V, 3 ms)	2	0	0	0	0	0
750-412/001	2 Ch Digital Input (DC 48 V, 3 ms)	2	0	0	0	0	0
750-414	4 Ch Digital Input (DC 5V, 0.2 ms)	4	0	0	0	0	0
750-415	4 Ch Digital Input (AC/DC 5V, 0.2 ms)	4	0	0	0	0	0

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.
CB = Number of control bytes (holding register).
SB = Number of status bytes (internal register).

*For the 740-404 counter modules, the control byte is the low byte of a holding register, and the address of this register will precede the 2 data output registers. Likewise, the status byte is the low byte of an internal register and the address of this register will precede the 2 data input registers.

Digital Output Modules

Module	Description	IC	OC	IR	HR	CB	SB
750-501	2 Ch Digital Output (DC 24 V, 0.5 A)	0	2	0	0	0	0
750-502	2 Ch Digital Output (DC 24 V, 2 A)	0	2	0	0	0	0
750-504	4 Ch Digital Output (DC 24, 0.5 A)	0	4	0	0	0	0
750-506	2 Ch Digital Output (DC 24 V, 0.5 A)	4	4	0	0	0	0
750-507	2 Ch Digital Output (DC 24 V, 2 A)	2	2	0	0	0	0
750-509	2 Ch Digital Output (Solid State Relay)	0	2	0	0	0	0
750-511	2 Ch Pulsewidth (250 Hz-20 kHz)	0	0	2	2	2*	2*
750-511/002	2 Ch Pulsewidth (2 Hz-250 Hz)	0	0	2	2	2*	2*
750-512	2 Ch Output Output Relay (AC 250 V)	0	2	0	0	0	0
750-513	2 Ch Output Output Relay (AC 250 V)	0	2	0	0	0	0
750-514	2 Ch Digital Output Relay (AC 125 V)	0	2	0	0	0	0
750-516	4 Ch Digital Output (DC 24 V)	0	4	0	0	0	0

750-517	2 Ch Digital Output Relay (AC 250)	0	2	0	0	0	0
750-519	4 Ch Digital Output (DC 5 V, 20 mA)	0	4	0	0	0	0

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.
CB = Number of control bytes (holding register).
SB = Number of status bytes (internal register).

*For the 740-511 pulse width modules, the control byte is the low byte of a holding register, and the address of this register will precede the 2 data output registers. Likewise, the status byte is the low byte of an internal register and the address of this register will precede the 2 data input registers.

Analog Input Modules

Module	Description	IC	OC	IR	HR	Scaling	Format
750-452	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Unit Numerical
750-452/200	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Siemens
750-454	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Unit Numerical
750-454/200	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Siemens
750-456	2 Ch Analog Input (+/-10 V)	0	0	2	0	None	Unit Numerical
750-456/200	2 Ch Analog Input (+/- 10 V)	0	0	2	0	None	Siemens
750-461	2 Ch Analog Input (PT 100)	0	0	2	0	TC	Unit Numerical
750-461/002	2 Ch Analog Input (Resistance Test)	0	0	2	0	None	Unit Numerical
750-461/003	2 Ch Analog Input (PT 1000)	0	0	2	0	TC	Unit Numerical
750-461/004	2 Ch Analog Input (NI 100)	0	0	2	0	TC	Unit Numerical
750-461/005	2 Ch Analog Input (NI 1000)	0	0	2	0	TC	Unit Numerical
750-461/007	2 Ch Analog Input (Resistance Test)	0	0	2	0	None	Unit Numerical
750-461/200	2 Ch Analog Input (PT 100)	0	0	2	0	None	Siemens
750-462	2 Ch Analog Input (TC)	0	0	2	0	TC	Unit Numerical
750-462/001	2 Ch Analog Input (TC Type S)	0	0	2	0	TC	Unit Numerical
750-462/002	2 Ch Analog Input (TC Type T)	0	0	2	0	TC	Unit Numerical
750-462/003	2 Ch Analog Input (+/1120 mV))	0	0	2	0	None	Unit Numerical
750-462/006	2 Ch Analog Input (TC Type J)	0	0	2	0	TC	Unit Numerical
750-462/007	2 Ch Analog Input (TC Type B)	0	0	2	0	TC	Unit Numerical
750-462/008	2 Ch Analog Input (TC Type E)	0	0	2	0	TC	Unit Numerical
750-462/009	2 Ch Analog Input (TC Type N)	0	0	2	0	TC	Unit Numerical
750-462/010	2 Ch Analog Input (TC Type R)	0	0	2	0	TC	Unit Numerical
750-462/011	2 Ch Analog Input (TC Type U)	0	0	2	0	TC	Unit Numerical
750-462/050	2 Ch Analog Input (TC Type K)	0	0	2	0	TC	Unit Numerical
750-462/061	2 Ch Analog Input (TC Type U)	0	0	2	0	TC	Unit Numerical
750-465	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Unit Numerical

750-465/200	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Siemens
750-466	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Unit Numerical
750-466/200	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Siemens
750-467	2 Ch Analog Input (0-10 V)	0	0	2	0	None	Unit Numerical
750-467/200	2 Ch Analog Input (0-10 V)	0	0	2	0	None	Siemens
750-468	4 Ch Analog Input (0-10 V)	0	0	4	0	None	Unit Numerical
750-468/200	4 Ch Analog Input (0-10 V)	0	0	4	0	None	Siemens
750-469	2 Ch Analog Input (TC Type K)	0	0	2	0	TC	Unit Numerical
750-469/001	2 Ch Analog Input (TC Type S)	0	0	2	0	TC	Unit Numerical
750-469/002	2 Ch Analog Input (TC Type T)	0	0	2	0	TC	Unit Numerical
750-469/003	2 Ch Analog Input (+/- 120 mV)	0	0	2	0	None	Unit Numerical
750-469/006	2 Ch Analog Input (TC Type J)	0	0	2	0	TC	Unit Numerical
750-469/008	2 Ch Analog Input (TC Type E)	0	0	2	0	TC	Unit Numerical
750-469/012	2 Ch Analog Input (TC Type L)	0	0	2	0	TC	Unit Numerical
750-469/200	2 Ch Analog Input (TC)	0	0	2	0	None	Siemens
750-472	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Unit Numerical
750-472/200	2 Ch Analog Input (0-20 mA)	0	0	2	0	None	Siemens
750-474	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Unit Numerical
750-474/200	2 Ch Analog Input (4-20 mA)	0	0	2	0	None	Siemens
750-476	2 Ch Analog Input (+/-10 V)	0	0	2	0	None	Unit Numerical
750-476/200	2 Ch Analog Input (+/-10 V)	0	0	2	0	None	Siemens
750-478	2 Ch Analog Input (0-10 V)	0	0	2	0	None	Unit Numerical
750-478/200	2 Ch Analog Input (0-10 V)	0	0	2	0	None	Siemens

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.

Note: TC Scaling is used for thermocouple modules. Raw data is multiplied by 10 and returned as a float value.

Analog Output Modules

Module	Description	IC	OC	IR	HR	Scaling	Format
750-550	2 Ch Analog Output (0-10 V)	0	0	0	2	None	Unit Numerical
750-550/200	2 Ch Analog Output (0-10 V)	0	0	0	2	None	Siemens
750-552	2 Ch Analog Output (0-20 mA)	0	0	0	2	None	Unit Numerical
750-552/200	2 Ch Analog Output (0-20 mA)	0	0	0	2	None	Siemens
750-554	2 Ch Analog Output (4-20 mA)	0	0	0	2	None	Unit Numerical
750-554/200	2 Ch Analog Output (4-20 mA)	0	0	0	2	None	Siemens
750-556	2 Ch Analog Output (+/-10 V)	0	0	0	2	None	Unit Numerical
750-556/200	2 Ch Analog Output (+/-10 V)	0	0	0	2	None	Siemens

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.

Power Supply and End Modules

Module	Description	IC	OC	IR	HR
750-610	Power Supply (DC 24 V)	2*	0	0	0
750-611	Power Supply (AC 230 V)	2*	0	0	0
750-622	Binary Spacer	X**	X	0	0

*For the 750-610 and 750-611 power supply modules, digital inputs DI1 and DI2 report over/under voltage and blown fuse.

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.

DI1	DI2	Description
0	0	Under voltage
1	0	Blown fuse
0	1	Over voltage, fuse OK

**The binary spacer module 750-622 can be configured to used 2, 4, 6, or 8 digital inputs, or 2, 4, 6, or 8 digital outputs. Selecting this module in the slot configuration dialog will invoke a "Binary Space Module configuration" dialog, where the setting of the physical module can be matched.

See Also: Slot Configuration

Encoder and Resolver Modules

Module	Description	IC	OC	IR	HR	CB	SB
750-630	SSI Transmitter Interface (Gray code)	0	0	2	0	0	0
750-630/001	SSI Transmitter Interface (Binary code)	0	0	2	0	0	0
750-630/006	SSI Transmitter Interface (Gray code)	0	0	2	0	0	0
750-631	Incremental Encoder Interface (4X)	0	0	3	3	1*	1*
750-631/001	Incremental Encoder Interface (1X)	0	0	3	3	1*	1*

*For the 740-631 family of incremental encoder interface modules, the control byte is the low byte of a holding register, and the address of this register will precede the 3 data output registers. Likewise, the status byte is the low byte of an internal register, and the address of this register will precede the 3 data input registers. The first input or output register contains counter data, the second contains period data, and the third contains latch data.

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.
CB = Number of control bytes (holding register).
SB = Number of status bytes (internal register).

Binary Spacer Module

When adding a Binary Spacer Module to the slot configuration, specify how many input and output coils will be used by the module. There can be 2, 4, 6 or 8 of each type of coil.

Special Modules

Module	Description	IC	OC	IR	HR
750-650	RS232C Interface (3 byte)	0	0	2	2
750-650/001	RS232C Interface (5 byte)	0	0	3	3
750-651	TTY Interface (3 byte)	0	0	2	2
750-651/001	TTY Interface (5 byte)	0	0	3	3
750-653	RS485C Interface (3 byte)	0	0	2	2
750-653/001	RS485C Interface (5 byte)	0	0	3	3
750-654	Data Exchange Module (3 byte)	0	0	2	2
750-654/001	Data Exchange Module (5 byte)	0	0	3	3

IC = Number of Input Coils used.
OC = Number of Output Coils used.
IR = Number of Internal (input) Registers used.
HR = Number of Holding (output) Registers used.

Data is structured as (3 byte versions):
Analog Input 1 = [Data byte 0] [Status byte]
Analog Input 2 = [Data byte 2] [Data byte 1]

Analog Output 1 = [Data byte 0] [Control byte]
Analog Output 2 = [Data byte 2] [Data byte 1]

Data is structured as (5 byte versions):
Analog Input 1 = [Data byte 0] [Status byte]
Analog Input 2 = [Data byte 2] [Data byte 1]
Analog Input 3 = [Data byte 4] [Data byte 3]

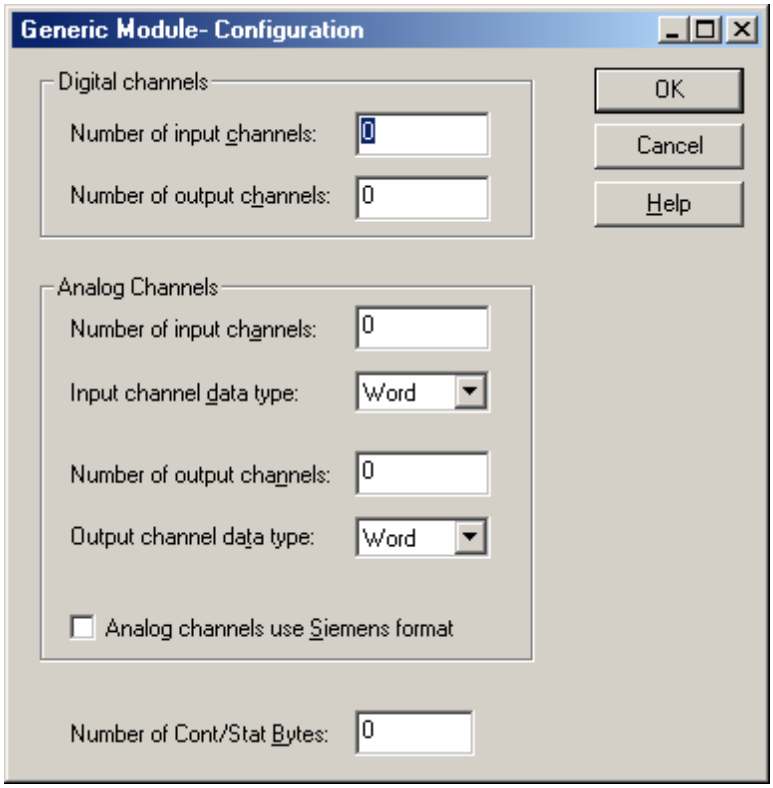
Analog Ouput 1 = [Data byte 0] [Control byte]
Analog Output 2 = [Data byte 2] [Data byte 1]
Analog Output 3 = [Data byte 4] [Data byte 3]

Note: When a special module is added to the slot configuration, tags specific to that module will be automatically generated. Before that occurs, however, users will have the following options:

- To have a Byte tag created for each data byte and a Bool tag created for each status/control bit.
- To have Word tags created with the packed data.

Generic Module

A module that is not listed in the driver's Slot Configuration dialog can use the driver's Generic Module to communicate. To invoke the Generic Module Configuration dialog, select **Generic Module Type**.



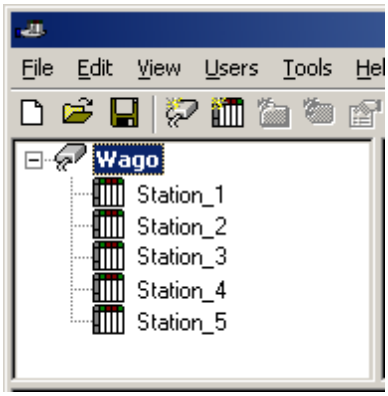
The generic module is extremely flexible. Up to 255 module channels may be configured: each of digital input, digital output, analog input and analog output. For analog channels, specify the data type of the input and output channel tags. The number of registers addressed by the generic module is determined from the number of channels and the data type. The number of internal registers used is equal to the number of input channels if the input channel data type is Short or Word, and two times the number of channels for Long, Float and DWord types. The number of holding registers used is calculated in a similar fashion from the number of output channels and output channel data type.

To generate circuit and overflow status tags, check the **Analog channels use Siemens format** box (if the module uses this format option). If the module uses control/status bytes, set the total number used in the bottom box. This will affect how the module (and others in higher slot positions) will be addressed. It will also signal the automated tag generation feature to create control and status byte tags. The number of control bytes is equal to the number of status bytes.

Optimizing Your Wago Ethernet Communications

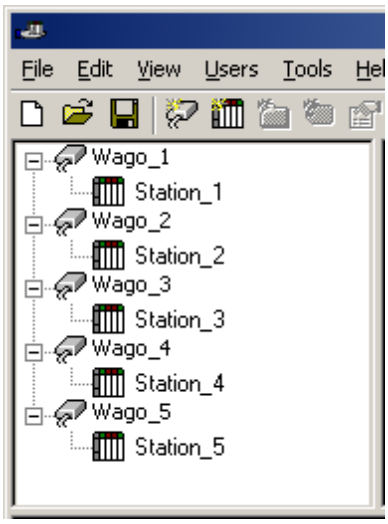
The Wago Ethernet driver has been designed to provide the best performance with the least amount of impact on the system's overall performance. While the Wago Ethernet driver is fast, there are a couple of guidelines that can be used in order to control and optimize the application and gain maximum performance.

Our server refers to communications protocols like Wago 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 Wago 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 Wago 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 Wago 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 Wago Ethernet driver could only define one single channel, then the example shown above would be the only option available; however, the Wago Ethernet driver can define up to 16 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 16 or fewer devices, it can be optimized exactly how it is shown here.

The performance will improve even if the application has more than 16 devices. While 16 or fewer devices may be ideal, the application will still benefit from additional channels. Although by spreading the device load across all 16 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 Wago 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, the block size may be configured to 1 to 120 registers and 8 to 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
BCD	Two byte packed BCD Value range is 0-9999. Behavior is undefined for values beyond this range.
LBCD	Four byte packed BCD Value range is 0-99999999. Behavior is undefined for values beyond this range.
Float	32 bit floating point value. The driver interprets two consecutive registers as a floating point value by making the second register the high word and the first register the low word. When accessing data from a thermocouple or resistance sensor input module, the driver will use a single 16-bit register as a floating point value.
Float Example	If register 40001 is specified as a float, bit 0 of register 40001 would be bit 0 of the 32 bit word, and bit 15 of register 40002 would be bit 31 of the 32 bit word.

Address Descriptions

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

750-342 Buscoupler
750-842 Programmable Fieldbus Controler (PFC)

750-342 Buscoupler

The 750-342 Buscoupler maps all of the installed modules to a process image. The rules used for this mapping have been programmed into this driver, allowing for automated tag generation. For information on how to manually create tags to access module data (or on how this driver computes addresses during automated tag generation) refer to **Process Image**.

Only part of the 750-342 available memory is used for the process image. All memory, including that used by the process image, can be accessed with manually created tags. Conventional Modbus addressing is used.

See Also:
Input Coils
Output Coils
Internal Registers
Holding Registers

750-842 Programmable Fieldbus Controller (PFC)

The 750-842 Programmable Fieldbus Controller maps all of the installed modules to a process image. The rules used for this mapping have been programmed into this driver, allowing for automated tag generation. For information on how to manually create tags to access module data (or on how this driver computes addresses during automated tag generation) refer to **Process Image**.

Only part of the 750-842 available memory is used for the process image. All memory, including that used by the process image, can be accessed with manually created tags. Conventional Modbus addressing is used.

See Also:
Input Coils
Output Coils
Internal Registers
Holding Registers

Process Image

The server's automated OPC tag database generation takes care of all addressing issues for data associated with modules and is the preferred method of creating tags. In order to manually create tags to access module data, users must understand how the data is mapped by the Buscoupler or PFC. The Wago I/O System uses the **Open Modbus protocol**, addressing digital inputs and outputs as Modbus coils and analog inputs and outputs as Modbus registers. When the buscoupler or PFC is powered up, it creates a process image of all of the data points associated with the installed modules.

Modbus addresses are assigned sequentially as the modules are arranged in the device (for all input coils, output coils, internal registers and holding registers). Addresses with a preceding "0" are interpreted by the driver as input coil addresses (digital input), "1" as output coil addresses (digital output), "3" as internal register addresses (Read Only analog), and "4" as holding register addresses (Read and Write analog).

WAGO Modbus and Modbus/TCP Buscoupler and PFC Memory Map

	Conventional Modbus Addressing	Alternative Modbus Addressing	PFC Addressing	Memory Stack		
0 255	Word addressing begins at 30001 Bit addressing begins at 10001	Word addressing begins at 40001 Bit addressing begins at 00001	Begins at %IX0.0, %IB0, %IW0, etc.	Real World Inputs (Buscoupler or PFC)	Analog In's Digital In's Available Memory	Addressing begins at: 30001 + numberOfAnalogOutputWords OR 40001 + numberOfAnalogOutputWords OR 00001 OR 10001
256 511	Word addressing begins at 30257 Bit addressing not available	Word addressing begins at 40257 Bit addressing not available	Begins at %QX256.0, %QB512, %QW256, etc.	Available Memory (PFC Only)		
512 767	Word addressing begins at 40001 Bit addressing begins at 00001 (These addresses are Write Only.)	Word addressing begins at 40513 Bit addressing begins at 00513	Begins at %QX0.0, %QB0, %QW0, etc.	Real World Outputs (Buscoupler or PFC)	Analog Out's Digital Out's Available Memory	Addressing begins at: 40001 + numberOfAnalogOutputWords * OR 40513 + numberOfAnalogOutputWords OR 00001 * OR 00513 * Write Only
768 1023	Word addressing begins at 40257 Bit addressing not available (These addresses are Write Only.)	Word addressing begins at 40769 Bit addressing not available	Begins at %IX256.0, %IB512, %IW256, etc.	Available Memory (PFC Only)		

Note: Various system parameters are located beyond this range. For more information, refer to the hardware documentation.

Example of Module Mapping

Slot	Wago Module	Modbus Addresses
1	4 Channel Digital Input - Input 1 - Input 2 - Input 3	(Input Coils 0-3) - 10000 - 10001 - 10002

	- Input 4	- 10003
2	1 Channel Analog Input (32-bit) - Input 1	(Internal Registers 0-1) - 30000
3	4 Channel Digital Output - Output 1 - Output 2 - Output 3 - Output 4	(Output Coils 0-3) - 00000 - 00001 - 00002 - 00003
4	2 Channel Analog Output (16-bit) - Output 1 - Output 2	(Holding Registers 0-1) - 40000 - 40001
5	4 Channel Digital Input - Input 1 - Input 2 - Input 3 - Input 4	(Input Coils 4-7) - 10004 - 10005 - 10006 - 10007
6	2 Channel Analog Input (16-bit) - Input 1 - Input 2	(Internal Registers 2-3) - 30002 - 30003
7	4 Channel Digital Output - Output 1 - Output 2 - Output 3 - Output 4	(Output Coils 4-7) - 00004 - 00005 - 00006 - 00007
8	2 Channel Analog Output (16-bit) - Output 1 - Output 2	(Holding Registers 2-3) - 40002 - 40003

Note: The Wago I/O System Buscoupler and PFC handle reads to output channels differently. In these cases, 0x0200 must be added to the address. This is done by the driver: users do not have to be concerned with this off-set when defining tags manually.

Input Coils

	Decimal Addressing	Hexadecimal Addressing
Type	**Boolean**	**Boolean**
Format	1xxxxx	H1yyyyy
Security	Read Only	Read Only
Range	1-65536	1-10000

Example
The 127th input coil would be addressed as '1127' using decimal addressing or 'H17F' using hexadecimal addressing.

Output Coils

	Decimal Addressing	Hexadecimal Addressing
Type	**Boolean**	**Boolean**
Format	0xxxxx	H0yyyyy
Security	Read/Write	Read/Write
Range	1-65536	1-10000

Example
The 255th output coil would be addressed as '0255' using decimal addressing or 'H0FF' using hexadecimal addressing.

Internal Registers

The default data types are shown in **bold**.

	Decimal Addressing	Hexadecimal Addressing
Type	**Word**, Short, BCD	**Word**, Short, BCD
Format	3xxxxx	H3yyyyy
Security	Read Only	Read Only
Range	1-65536	1-10000
Type	Byte	Byte
Format	3xxxxxL (Low Byte in Word) 3xxxxxH (High Byte in Word)	H3yyyyyL (Low Byte in Word) H3yyyyyH (High Byte in Word)
Security	Read Only	Read Only
Range	1-65536	1-10000
Type	**Boolean**	**Boolean**
Format	3xxxxx.bb	H3yyyyy.c
Security	Read Only	Read Only
Range	3xxxxx.0-3xxxxx.15	H3yyyyy.0-H3yyyyy.F
Type	Float, DWord, Long, LBCD	Float, DWord, Long, LBCD
Format	3xxxxx	H3yyyy
Security	Read Only	Read Only
Range	1-65535	1-FFFF

Arrays

Arrays are also supported for the internal register addresses. The syntax for declaring an array (shown using decimal addressing) is:

3xxxx[rows][cols]
3xxxx[cols] with assumed row count of 1

For Word, Short and BCD arrays, the base address + (rows * cols) cannot exceed the number of internal registers in the device configuration.

For Float, DWord, Long and Long BCD arrays, the base address + (rows * cols * 2) cannot exceed one minus the number of internal registers in the device configuration.

For all arrays, the total number of registers being requested cannot exceed the internal register block size that was specified for this device.

Holding Registers

The default data types are shown in **bold**.

	Decimal Addressing	Hexadecimal Addressing
Type	**Word,** Short, BCD	**Word**, Short, BCD
Format	4xxxxx	H4yyyyy
Security	Read/Write	Read/Write

Range	1-65536	1-10000
Type	Byte	Byte
Format	4xxxxxL (Low Byte in Word) 4xxxxxH (High Byte in Word)	H4yyyyyL (Low Byte in Word) H4yyyyyL (High Byte in Word)
Security	Read/Write	Read/Write
Range	1-65536	1-10000
Type	**Boolean**	**Boolean**
Format	4xxxxx.bb	H4yyyyy.c
Security	Read/Write	Read/Write
Range	4xxxxx.0-4xxxxx.15	H4yyyyy.0-H4yyyyy.F
Type	Float, DWord, Long, LBCD	Float, DWord, Long, LBCD
Format	4xxxxx	H4yyyy
Security	Read/Write	Read/Write
Range	1-65535	1-FFFF

Arrays

Arrays are also supported for the holding register addresses. The syntax for declaring an array (using decimal addressing) is as follows:

4xxxx[rows][cols]
4xxxx[cols] with assumed row count of 1

For Word, Short and BCD arrays, the base address + (rows * cols) cannot exceed the number of holding registers in the device configuration.

For Float, DWord, Long and Long BCD arrays, the base address + (rows * cols * 2) cannot exceed one minus the number of holding registers in the device configuration.

For all arrays, the total number of registers being requested cannot exceed the holding register block size that was specified for this device.

Tag Naming Convention

Tags created by the automated tag database generator will be named according to the following convention:

Slot<slot number>_<module model number>_<I/O type><tag number>[_Bit<bit number>]

where

I/O Type	Meaning
DI	Digital Input
DO	Digital Output
AI	Analog Input
AO	Analog Output
ByteIn	Data Byte Input
ByteOut	Data Byte Output
CB	Control Byte*
SB	Status Byte*
Circuit	Circuit Short/Open Status**
Over	Numerical Overflow Status**

*All control and status tags will have the "_Bit<bit number>" attached.

**Siemens format only.

Note: All slot numbers and tag numbers are 1-based (1, 2, 3, ...8). All bit numbers are 0-based (0, 1, 2, ...7) by convention.

Examples
1. Slot1_750_402_DI3 is the name given to the digital input 3 of a 750-402 module installed in slot 1.
2. Slot2_750_404_CB1_Bit5 is the name given to a tag that reads and writes the value of bit 5 of the control byte of a 750-404 module installed in slot 2.

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>'

Device Specific Messages

Failure to initiate 'winsock.dll'
Bad address in block [x to y] on device '<device name>'
Bad received length [x to y] 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:
An invalid tag address has been specified in a dynamic request.

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.

Device Specific Messages

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

Device Specific Messages

Failure to initiate 'winsock.dll'
Bad address in block [x to y] on device '<device name>'
Bad received length [x to y] 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 [x to y] on device '<device name>'

Error Type:

Fatal for addresses falling in this block.

Possible Cause:
This error is reported when the driver attempts to read a location in a PLC that does not exist. For example in a PLC that only has holding registers 40001 to 41400, requesting address 41405 would generate this error. Once this error is generated, the driver will not request the specified block of data from the PLC again. Any other addresses being requested that are in this same block will also go invalid.

Solution:
The client application should be modified to ask for addresses within the range of the device.

Bad received length [x to y] on device '<device name>'

Error Type:
Fatal for addresses falling in this block.

Possible Cause:
The driver attempted to read a block of memory in the PLC. The PLC responded with no error, but did not provide the driver with the requested block size of data.

Solution:
Ensure that the range of memory exists for the PLC.

Index

7

A

B

D

E

F

G

H

I

L

M

O

P

S

T

U

W