Lufkin Modbus Driver Help

Table of Contents

Lufkin Modbus Driver Help

Help version 1.018

CONTENTS

Overview

The Lufkin Modbus Driver provides an easy and reliable way to connect Lufkin Modbus devices to OPC Client applications, including HMI, SCADA, Historian, MES, ERP and countless custom applications. It intended for use with serial devices that support the Extended Lufkin Automation Modbus (ELAM) protocol and the Standard MODBUS protocol.

Channel Setup

This driver supports multiple channel settings. For more information, refer to "What is a Channel?" in the server help file.

Communication Serialization

The Lufkin Modbus Driver supports Communication Serialization, which specifies whether data transmissions should be limited to one channel at a time. For more information, refer to "Channel Properties - Advanced" in the server help file.

Device Setup

Supported Devices

Injection Well Controller (IWC)
Progressive Cavity Pump (PCP)
Rod Pump Controller (RPC)
Variable Speed Drive (VSD)

Communication Protocol

Extended Lufkin Automation Modbus (ELAM)
Standard MODBUS

Supported Communication Parameters

Baud Rate: All major Baud rates.
Parity: Odd, Even, and None.
Data Bits: 8.
Stop Bits: 1 and 2.

Note: Not all of the listed configurations may be supported in every device.

Maximum Number of Channels and Devices

The maximum number of channels supported by this driver is 256. The maximum number of devices supported is 2296.

Ethernet Encapsulation

This driver supports Ethernet Encapsulation, which allows the driver to communicate with serial devices attached to an Ethernet network using a terminal server. It may be enabled for the channel through the Communications dialog in Channel Properties. For more information, refer to the OPC server's help file.

Device ID (PLC Network Address)

Lufkin Modbus devices are assigned Device IDs in the range 0 to 2295. When using Modbus Device ID 0, the driver will send only broadcast Write messages to remote stations. When configuring a device under the channel, setting the Device ID to 0 will place that device in broadcast mode. Only Writes will occur from this device. Reads from the broadcast device will always return zero. All other Device IDs will read and write data to and from the remote Lufkin Modbus device.

Flow Control

When using an RS232/RS485 converter, the type of flow control that is required depends on the converter's needs. Some do not require any flow control whereas others require RTS flow. Consult the converter's documentation to determine its flow requirements. An RS485 converter that provides automatic flow control is recommended.

Note: When using the manufacturer's supplied communications cable, it is sometimes necessary to choose a flow control setting of **RTS** or **RTS Always** in Channel Properties.

Manual Flow Control

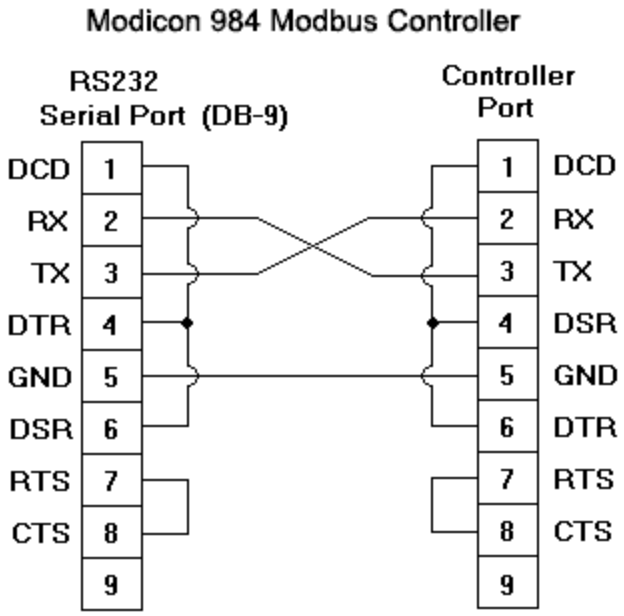
The Lufkin Modbus Driver supports RTS Manual flow control, which is used to configure the driver for operation with radio modems that require special RTS timing characteristics. For more information, refer to the OPC server's help documentation.

Automatic Tag Database Generation

Automatic Tag Database Generation is supported for the RPC and VSD device models. For more information on the Card Tags that will be created, refer to **Card Settings**.

Cable Diagram

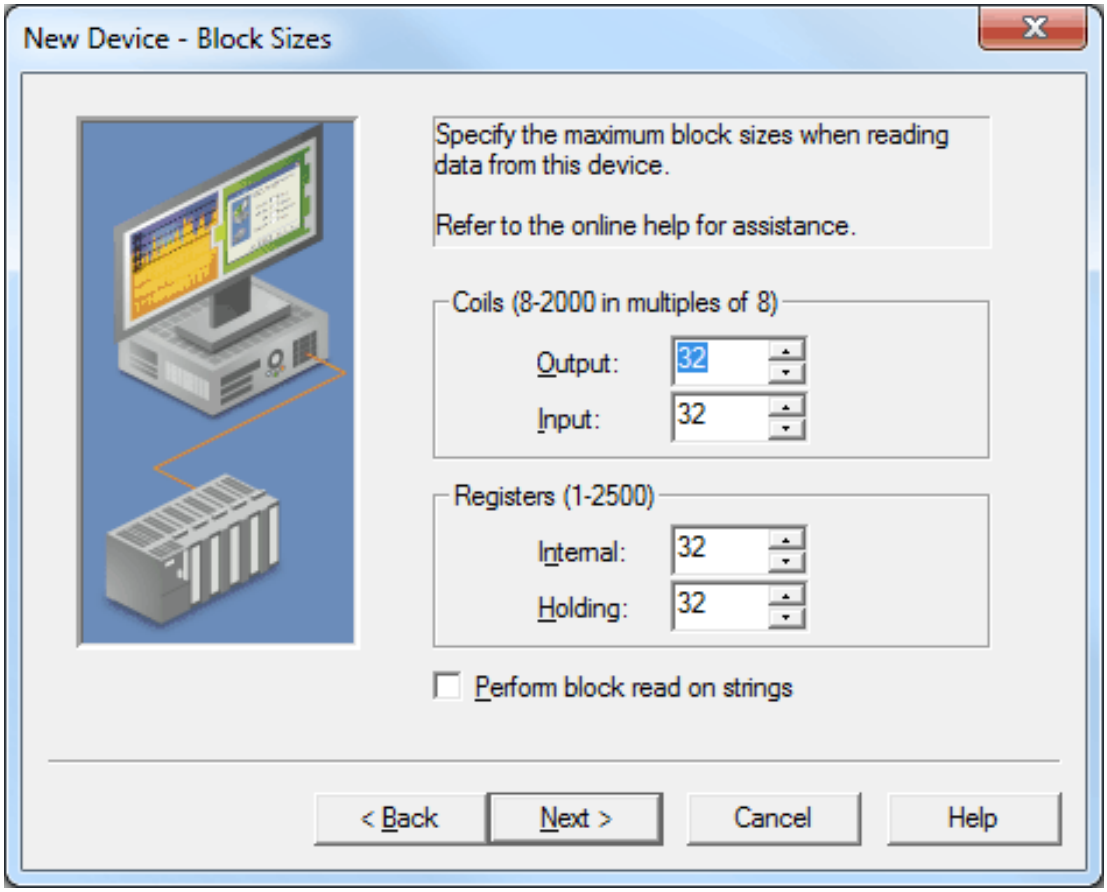
For recommended wiring and cable diagrams, refer to the Modbus device manufacturer's documentation. The Modicon 984 Modbus Controller cable diagram is shown below.



Modem Setup

This driver supports modem functionality. For more information, please refer to the topic "Modem Support" in the OPC Server Help documentation.

Block Sizes



Descriptions of the parameters are as follows:

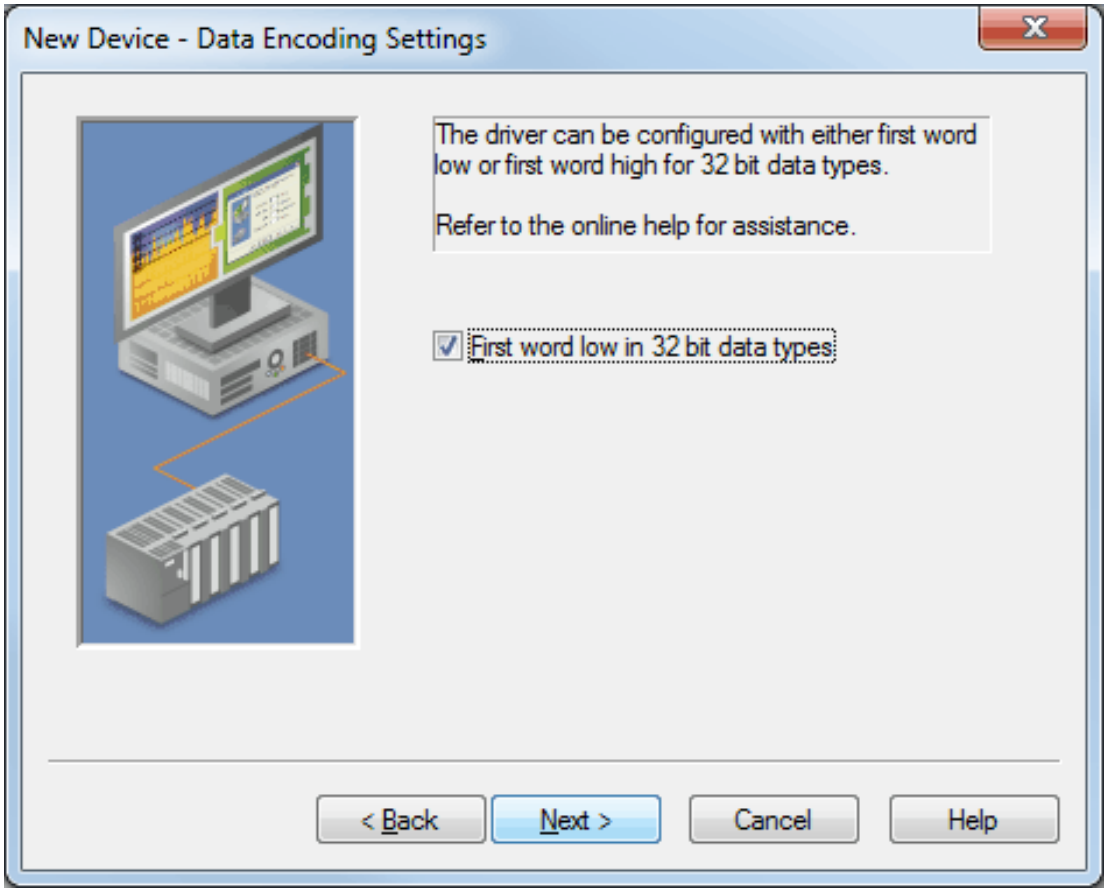
- **Coils (8-2000 in multiples of 8):** This parameter specifies the output and input coils. Coils can be read from 8 to 2000 points (bits) at a time. A higher block size means more points will be read from the device in a single request. The block size can be reduced to read data from non-contiguous locations within the device. The default setting is 32.

  Important: Certain Firmware versions for Lufkin devices may not support requests for blocks of coils greater than 1992. It is recommended that users with projects requiring the coil block size to be set above this value contact the vendor
- **Registers (1-2500):** This parameter specifies the internal and holding registers. The minimum block size is 1. The maximum block size depends on the Device ID. If the Device ID is 0 to 247, the configurable block size range is 1 to 125. If the Device ID is 248 to 2295, the configurable block size range is 1 to 2500. A higher block size means more register values will be read from the device in a single request. The block size can be reduced to read data from non-contiguous locations within the device. The default setting is 32.

  Note: Array data is read from a device in pieces no greater than the block size that was configured for the associated function code. For example, with a configured holding register block size of 100, the tag "40001[250]" would be read from a device in three transactions of 100, 100, and 50 respectively.
- **Perform block read on strings:** When checked, this option will block read string tags (which are normally read individually). String tags will also be grouped together depending on the selected block size. Block reads can only be performed for Modbus model string tags. The default setting is unchecked.

Data Encoding Settings

Two consecutive registers' addresses are used for 32-bit data types. The Data Encoding Settings are used to specify whether the driver should assume the first word is the low or high word of the 32-bit value.



Description of the option is as follows:

- **First word low in 32-bit data types:** When checked, the driver will use first word low in 32-bit data types and data type arrays (such as Float, DWord, Long, and LBCD). The default setting is checked.

  Note: This setting does not apply to the Word, Short, BCD, or Double data types.

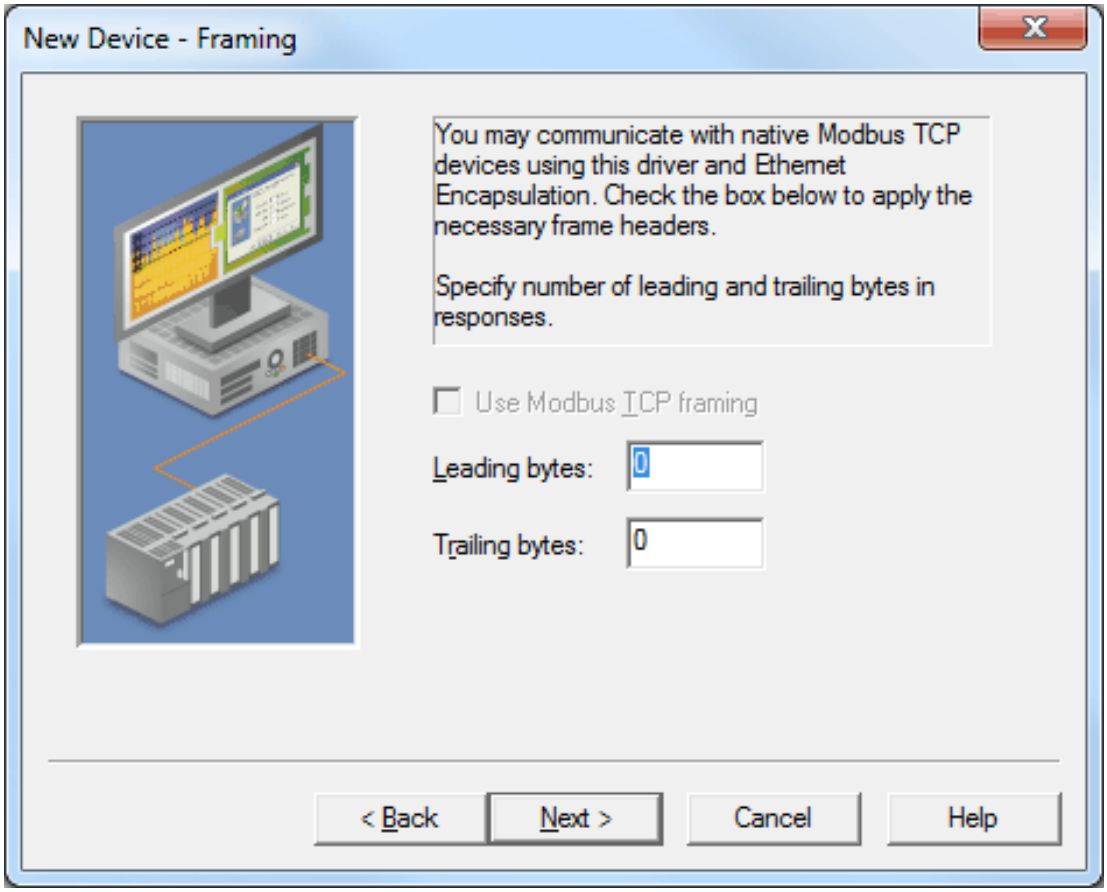
Data Encoding

To determine the correct Data Encoding setting, refer to both the table below and the device's documentation. For most devices, the default setting is acceptable.

Data Encoding Setting	High Word	Low Word
Checked	(31..16)	(15..0)
Unchecked	(15..0)	(31..16)

Framing

Some terminal server devices add additional data to Modbus frames; as such, the Framing parameters can be used to configure the driver to ignore the additional bytes in response messages.



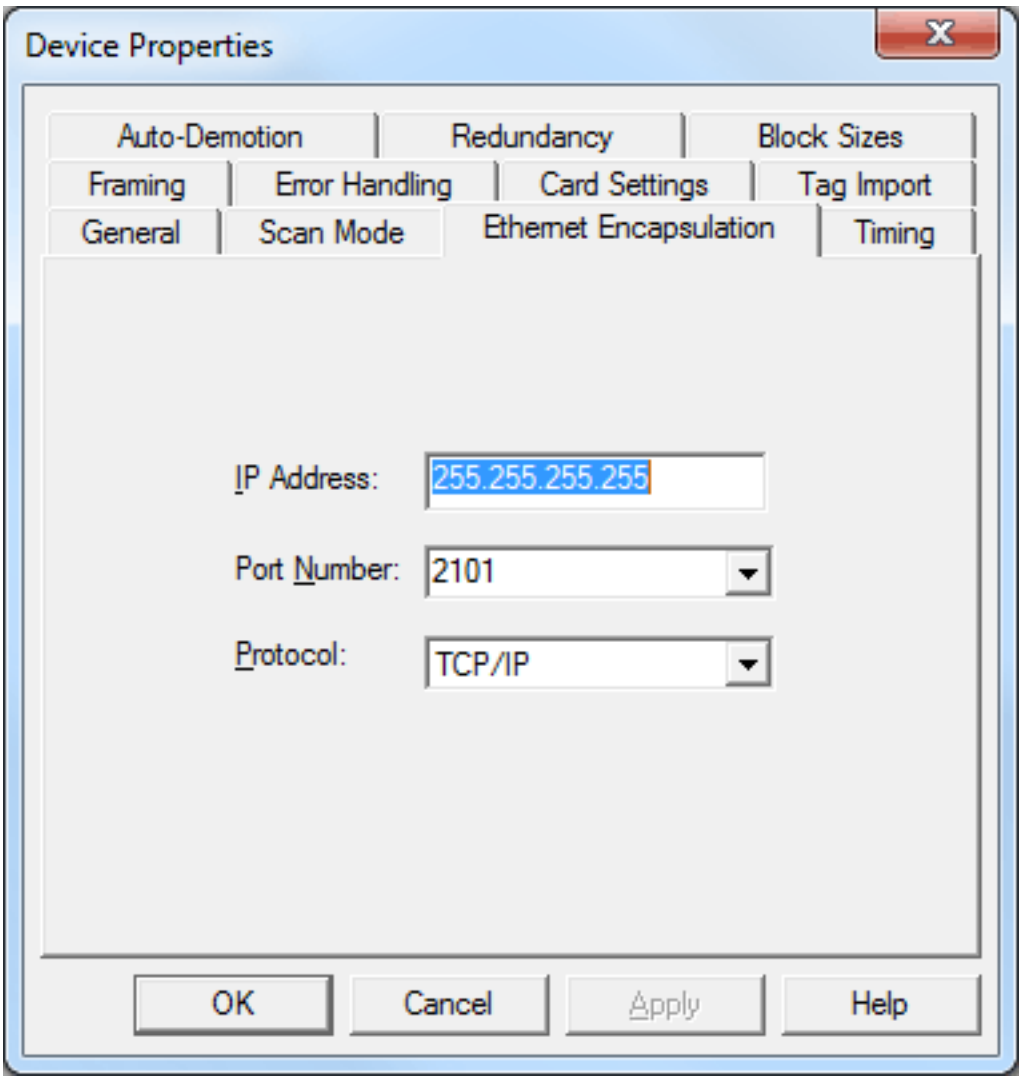
Descriptions of the parameters are as follows:

- **Use Modbus TCP Framing:** When checked, this parameter is used to communicate with native Modbus TCP devices using Ethernet Encapsulation. This option will be disabled when the device's ID is in the extended range (ID > 247).
- **Leading bytes:** This parameter is used to specify the number of bytes to be attached to the beginning of Modbus responses. Values may range from 0 to 8.
- **Trailing bytes:** This parameter is used to specify the number of bytes to be attached to the end of Modbus responses. Values may range from 0 to 8.

Using Ethernet Encapsulation

Ethernet Encapsulation must be enabled for Framing to be available; otherwise, the selection **Use Modbus TCP Framing** will be disabled. For information on enabling Ethernet Encapsulation, refer to the instructions below.

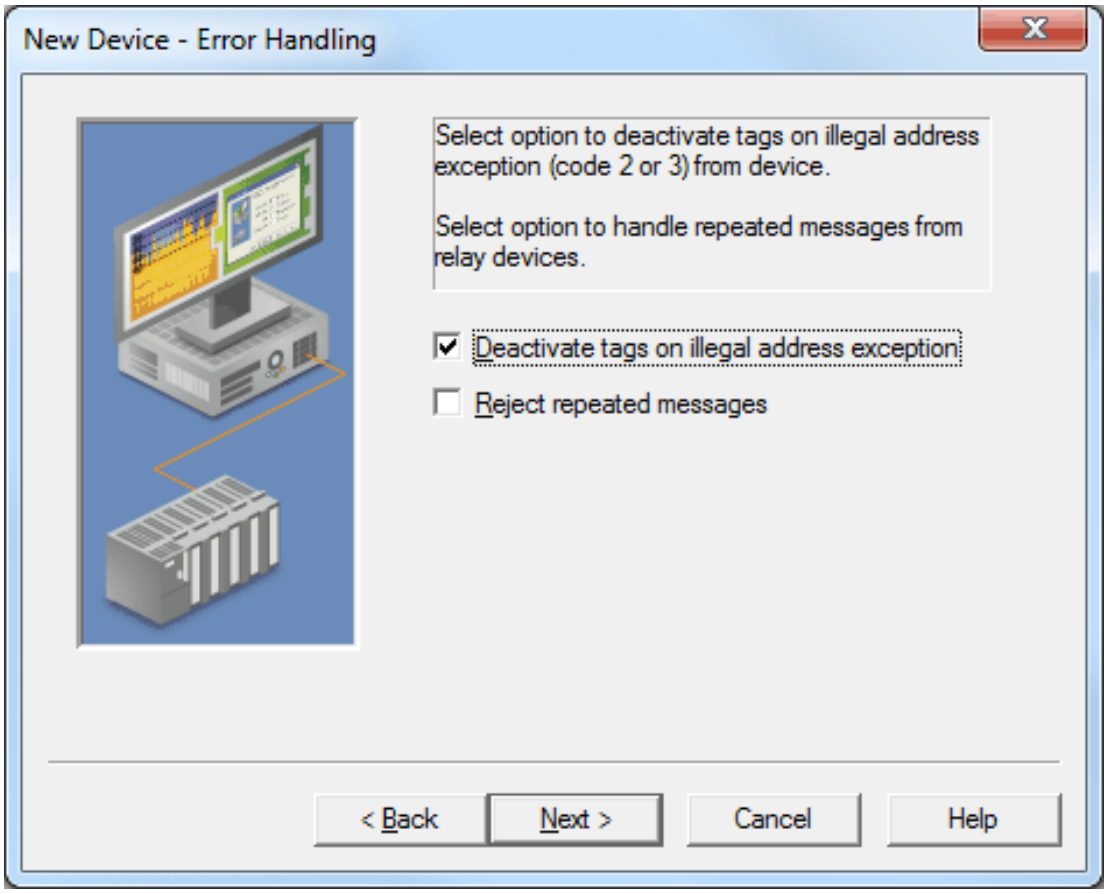
1. To start, open the device's **Channel Properties**.
2. In the **Communications** tab, select **Use Ethernet Encapsulation**. This will enable Ethernet Encapsulation for the channel.
3. Next, open the device's **Device Properties**. Descriptions of the parameters are as follows:
    - **IP Address:** This parameter specifies the device's IP address. The default setting is 255.255.255.255.
    - **Port Number:** This parameter specifies the port number. The default setting is 2101.
    - **Protocol:** This parameter specifies the protocol. The default setting is TCP/IP.



See Also: Device Setup

Error Handling

The Error Handling parameters determine how to deal with errors from the device.



Descriptions of the parameters are as follows:

- **Deactivate Tags on Illegal Address Exception:** When checked, the driver will stop polling for a block of data if the device returns Modbus exception code 2 (illegal address) or 3 (illegal data, such as number of points) in response to a read of that block. To read addresses that are accessible dynamically in the device, uncheck this option. The default setting is checked.
- **Reject Repeated Messages:** When checked, the driver will expect repeated messages. When unchecked, the driver will interpret a repeated message as an invalid response and will retry the request. The default setting is unchecked.

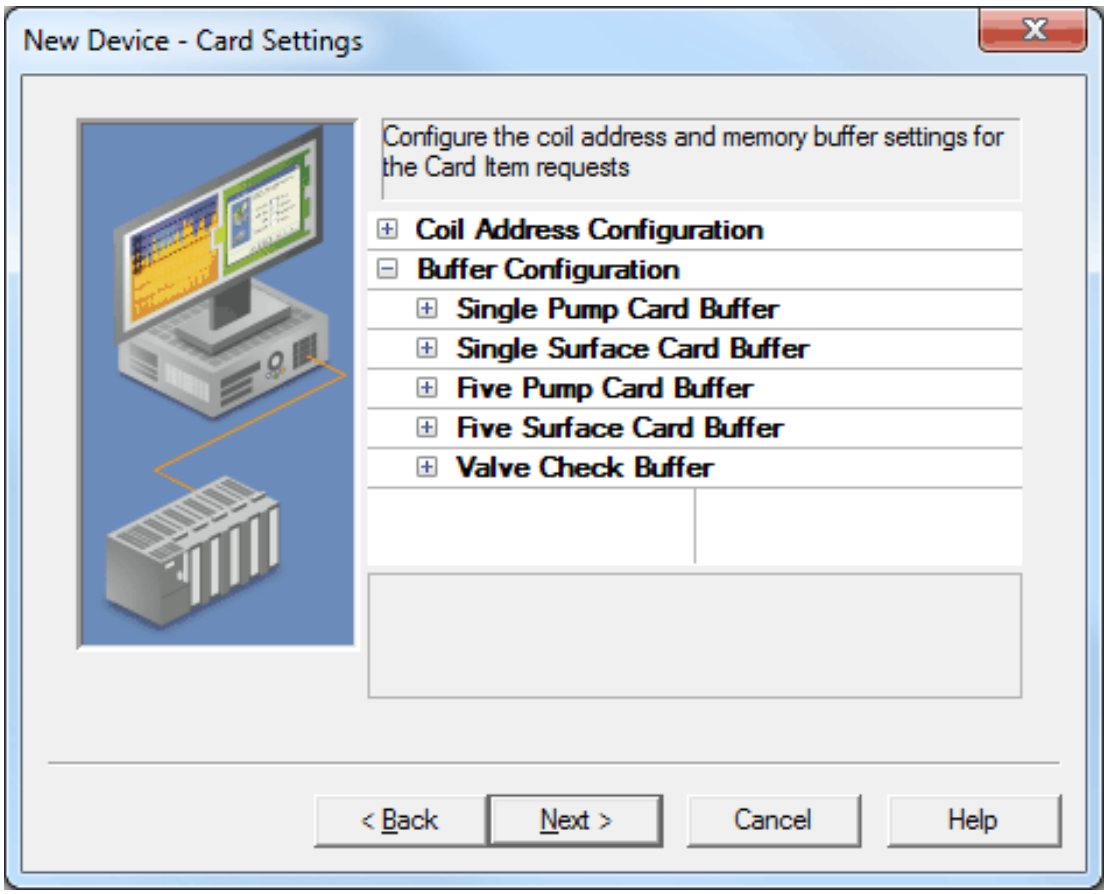
Note: Some message-relay equipment will echo Modbus requests back to the driver.

Card Settings

Card items are tags that facilitate the reading of dynagraph card data from a Lufkin device. When a client reads a card item tag, the server writes to an address on the device that initiates the loading of dynagraph card data into a memory buffer located on that same device. The server then reads the data from the device's buffer, which completes the card item read.

The Card Settings dialog specifies the address that the server will write to, as well as the buffer address and length that the server will read for a particular card item.

Note: The Card Settings parameters are only available for the RPC and VSD device models.



Descriptions of the parameters are as follows:

- **Coil Address Configuration:** This group contains the coil addresses to which the server will write to initiate a buffer load on a device. The addresses' default settings are device-specific, and will be configured when the device is first created.
- **Buffer Configuration:** This group contains the memory buffers. Each memory buffer has a starting address and length parameter that the server will use when reading the dynagraph card data. The memory buffers' default starting addresses and lengths are device-specific, and will be configured when the device is first created.

Card Item Tag to Coil Address/Memory Buffer Mapping

The table below provides a mapping between card items and their corresponding coil address, memory buffer address, and length. To change these parameters, click in the second column.

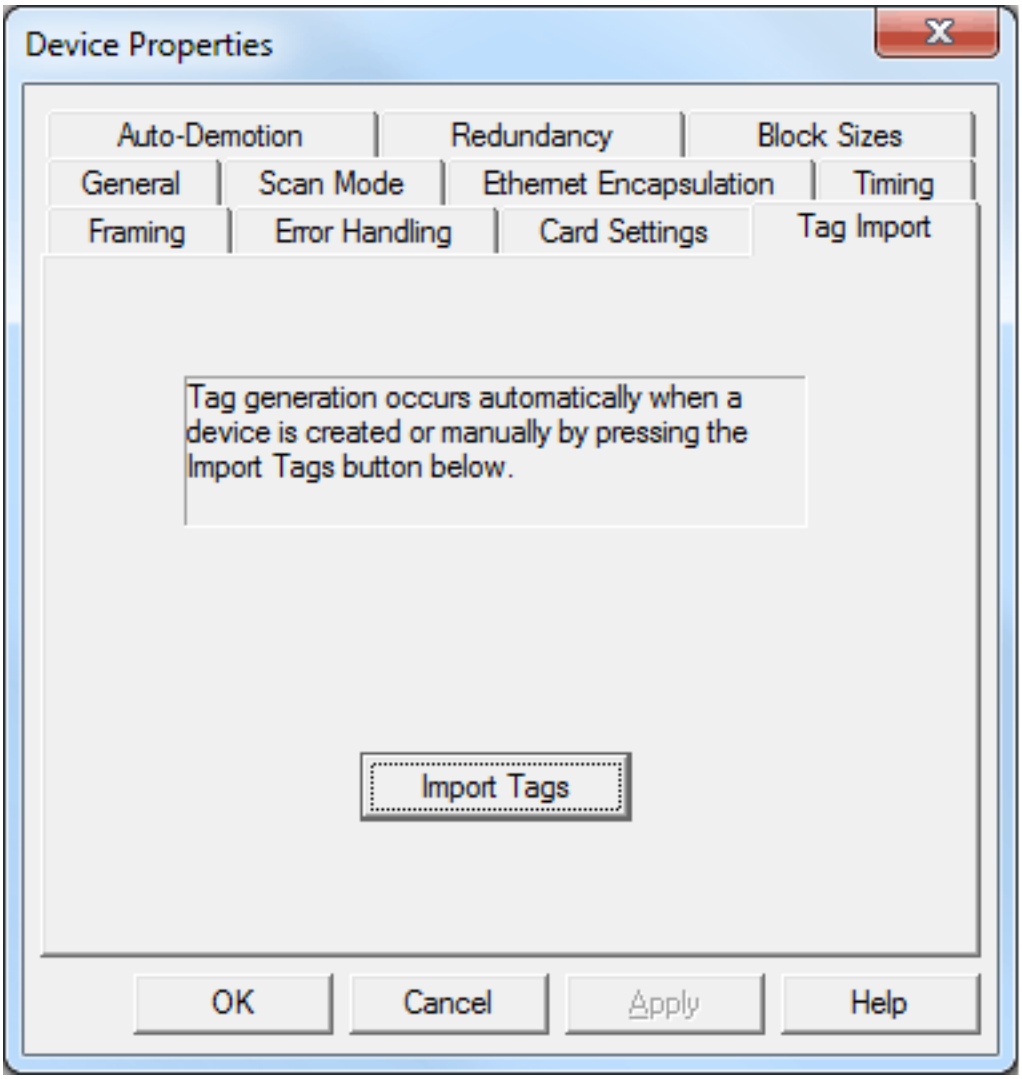
Card Item	Coil Address	Memory Buffer
Stored_SurfaceCard	Load Stored Card Coil Address	Five Surface Card Buffer
Stored_PumpCard	Load Stored Card Coil Address	Five Pump Card Buffer
Shutdown1_SurfaceCard	Load Shutdown1 Card Coil Address	Five Surface Card Buffer
Shutdown1_PumpCard	Load Shutdown1 Card Coil Address	Five Pump Card Buffer

Card Item	Coil Address	Memory Buffer
Shutdown2_SurfaceCard	Load Shutdown2 Card Coil Address	Five Surface Card Buffer
Shutdown2_PumpCard	Load Shutdown2 Card Coil Address	Five Pump Card Buffer
Standard_SurfaceCard	Load Standard Card Coil Address	Single Surface Card Buffer
Standard_PumpCard	Load Standard Card Coil Address	Single Pump Card Buffer
Start_SurfaceCard	Load Start Card Coil Address	Single Surface Card Buffer
Start_PumpCard	Load Start Card Coil Address	Single Pump Card Buffer
PumpUp_SurfaceCard	Load Pump Up Card Coil Address	Single Surface Card Buffer
PumpUp_PumpCard	Load Pump Up Card Coil Address	Single Pump Card Buffer
LastStroke_SurfaceCard	Load Last Stroke Card Coil Address	Single Surface Card Buffer
LastStroke_PumpCard	Load Last Stroke Card Coil Address	Single Pump Card Buffer
LastShutdown_SurfaceCard	Load Shutdown1 Card Coil Address	Five Surface Card Buffer
LastShutdown_PumpCard	Load Shutdown1 Card Coil Address	Five Surface Card Buffer
Reference_ValveCheck	Load Reference Valve Check Coil Address	Valve Check Buffer
Working_ValveCheck	Load Working Valve Check Coil Address	Valve Check Buffer

Tag Import

The Tag Import dialog is used to import tags from the Lufkin Modbus device.

Note: Tag Import is only available for the RPC and VSD device models.



Description of the button is as follows:

- **Import Tags:** When clicked, this button will import card item tags from the Lufkin Modbus device. If the server configuration utility does not have a connection to the Runtime (or if the device does not support Card Items), this button will be disabled.

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.
String	Null terminated ASCII string Includes HiLo and LoHi byte order selection.
Float	32-bit floating point value The driver interprets two consecutive registers as a single precision value by making the last register the high word and the first register the low word.
Float Example	If register 40001 is specified as a float, bit 0 of register 40001 would be bit 0 of the 32-bit data type and bit 15 of register 40002 would be bit 31 of the 32-bit data type.

Address Descriptions

The default data types for dynamically defined tags are shown in **bold** where appropriate.

ELAM Addressing Decimal Format

Address	Range	Data Type	Access*
Output Coils [Function Codes (decimal): 01, 05, 15]	00001-09999	**Boolean**	Read/Write
Input Coils [Function Code (decimal): 02]	10001-19999	**Boolean**	Read Only
Internal Registers [Function Code (decimal): 04]	30001-39999 30001-39998 3xxxx.0-3xxxx.15	**Word**, Short, BCD, Float, DWord, Long, LBCD, Boolean	Read Only
Internal Registers As String with HiLo Byte Order [Function Codes (decimal): 04]	30001.2H-39999.240H .Bit is string length, range 2 to 240 bytes.	**String****	Read Only
Internal Registers As String with LoHi Byte Order [Function Codes (decimal): 04]	30001.2L-39999.240L .Bit is string length, range 2 to 240 bytes.	**String****	Read Only
Holding Registers [Function Codes (decimal): 03, 06, 16] [Function Codes (decimal): 03, 06, 16]	40001-49999 40001-49998 4xxxx.0-4xxxx.15	**Word**, Short, BCD, Float, DWord, Long, LBCD, Boolean	Read/Write
Holding Registers As String with HiLo Byte Order [Function Codes (decimal): 03, 16]	40001.2H-49999.240H .Bit is string length, range 2 to 240 bytes.	**String****	Read/Write
Holding Registers As String with LoHi Byte Order [Function Codes (decimal): 03, 16]	40001.2L-49999.240L .Bit is string length, range 2 to 240 bytes.	**String****	Read/Write

*All Read/Write addresses may be set as Write Only by prefixing a "W" to the address such as "W40001." This will prevent the driver from reading the register at the specified address. Any attempts by the client to read a Write Only tag will result in obtaining the last successful write value to the specified address. If no successful writes have occurred, the client will receive 0/NULL for numeric/string values for an initial value.

Caution: Setting the Client Access privileges of Write Only tags to Read Only will cause writes to these tags to fail and the client to always receive 0/NULL for numeric/string values.

For more information, refer to **String Support.

ELAM Addressing Hexadecimal Format

Address	Range	Data Type	Access
Output Coils [Function Codes (decimal): 01, 05, 15]	H00001-H0270F	**Boolean**	Read/Write
Input Coils [Function Code (decimal): 02]	H10001-H1270F	**Boolean**	Read Only
Internal Registers [Function Code (decimal): 04]	H30001-H3270F H30001-H3270E H3xxxx.0-H3xxxx.15	**Word**, Short, BCD, Float, DWord, Long, LBCD, Boolean	Read Only
Internal Registers As String with HiLo	H30001.2H-H3270F.240H	**String***	Read Only

Address	Range	Data Type	Access
Byte Order [Function Codes (decimal): 04]	 .Bit is string length, range 2 to 240 bytes.		
Internal Registers As String with LoHi Byte Order [Function Codes (decimal): 04]	H30001.2L-H3270F.240L .Bit is string length, range 2 to 240 bytes.	**String***	Read Only
Holding Registers [Function Codes (decimal): 03, 06, 16] [Function Codes (decimal): 03, 06, 16]	H40001-H4270F H40001-H4270E H4xxxx.0-H4xxxx.15	**Word**, Short, BCD, Float, DWord, Long, LBCD, Boolean	Read/Write
Holding Registers As String with HiLo Byte Order [Function Codes (decimal): 03, 16]	H40001.2H-H4270F.240H .Bit is string length, range 2 to 240 bytes.	**String***	Read/Write
Holding Registers As String with LoHi Byte Order [Function Codes (decimal): 03, 16]	H40001.2L-H4270F.240L .Bit is string length, range 2 to 240 bytes.	**String***	Read/Write

*For more information, refer to **String Support**.

String Support

The Lufkin Modbus Driver supports reading and writing holding register memory as an ASCII string. When using holding registers for string data, each register will contain two bytes of ASCII data. The order of the ASCII data within a given register can be selected when the string is defined. The length of the string can be from 2 to 240 bytes and is entered in place of a bit number. The length must be entered as an even number. The byte order is specified by appending either a "H" or "L" to the address.

String Examples

1. To address a string starting at 40200 with a length of 100 bytes and HiLo byte order, enter 40200.100H.
2. To address a string starting at 40500 with a length of 78 bytes and LoHi byte order, enter 40500.78L.

Note: Write requests to String Tags are limited to a maximum of 60 registers. Requests that surpass 60 will be truncated.

Normal Address Examples

1. The 255'th output coil would be addressed as '0255' using decimal addressing or 'H0FF' using hexadecimal addressing.
2. Some documentation refers to Modbus addresses by function code and location. For instance, function code 3, location 2000 would be addressed as '42000' or 'H47D0'. The leading '4' represents holding registers or function code 3.
3. Some documentation refers to Modbus addresses by function code and location. For instance, setting function code 5, location 100 would be addressed as '0100' or 'H064'. The leading '0' represents output coils or function code 5. Writing 1 or 0 to this address would set or reset the coil.

Array Support

Arrays are supported for internal and holding register locations for all data types except for Boolean and strings. Arrays are also supported for input and output coils (Boolean data types). There are two methods of addressing an array. Examples are given using holding register locations.

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

For arrays, rows multiplied by cols cannot exceed the maximum request size allowed by the protocol. For devices in standard Modbus mode, rows multiplied by cols multiplied by the data type length in registers cannot exceed 125. For devices in ELAM mode (ID > 247), rows multiplied by cols multiplied by the data type length in registers cannot exceed 2500.

Note: Write requests to Array Tags are limited to a maximum of 60 registers. Requests that surpass 60 will result in only the first 60 registers of the array being written to the device.

Packed Coil Address Type

The Packed Coil address type allows access to multiple consecutive coils as an analog value. This feature is available for both input coils and output coils, polled mode only. The only valid data type is Word. The syntax is as follows:

Output coils: 0*xxxx#nn* Word Read/Write
Input coils: 1*xxxx#nn* Word Read Only

where *xxxx* is the address of the first coil (decimal and hex values allowed), and *nn* is the number of coils to be packed into an analog value (1-16, decimal only).

The bit order will be such that the start address will be the least significant bit (LSB) of analog value.

Statistics Items

Statistical items use data collected through additional diagnostics information, which is not collected by default. To use statistical items, Communication Diagnostics must be enabled. To enable Communication Diagnostics, right-click on the channel in the Project View and click **Properties | Enable Diagnostics**. Alternatively, double-click on the channel and select **Enable Diagnostics**.

Channel-Level Statistics Items

The syntax for channel-level statistics items is *<channel>._Statistics*.

Note: Statistics at the channel level are the sum of those same items at the device level.

Item	Data Type	Access	Description
_CommFailures	DWord	Read/Write	The total number of times communication has failed (or has run out of retries).
_ErrorResponses	DWord	Read/Write	The total number of valid error responses received.
_ExpectedResponses	DWord	Read/Write	The total number of expected responses received.
_LastResponseTime	String	Read Only	The time at which the last valid response was received.
_LateData	DWord	Read/Write	The total number of times that a driver tag's data update occurred later than expected (based on the specified scan rate).
_MsgResent	DWord	Read/Write	The total number of messages sent as a retry.
_MsgSent	DWord	Read/Write	The total number of messages sent initially.
_MsgTotal	DWord	Read Only	The total number of messages sent (both _MsgSent + _MsgResent).
_PercentReturn	Float	Read Only	The proportion of expected responses (Received) to initial sends (Sent) as a percentage.
_PercentValid	Float	Read Only	The proportion of total valid responses received (_ TotalResponses) to total requests sent (_MsgTotal) as a percentage.
_Reset	Bool	Read/Write	Resets all diagnostic counters. Writing to the _Reset Tag causes all diagnostic counters to be reset at this level.
_RespBadChecksum	DWord	Read/Write	The total number of responses with checksum errors.
_RespTimeouts	DWord	Read/Write	The total number of messages that failed to receive any kind of response.
_RespTruncated	DWord	Read/Write	The total number of messages that received only a partial response.
_TotalResponses	DWord	Read Only	The total number of valid responses received (_ ErrorResponses + _ExpectedResponses).

Statistical items are not updated in simulation mode *(see device general properties)*.

Device-Level Statistics Items

The syntax for device-level statistics items is *<channel>.<device>._Statistics*.

Item	Data Type	Access	Description
_CommFailures	DWord	Read/Write	The total number of times communication has failed (or has run out of retries).
_ErrorResponses	DWord	Read/Write	The total number of valid error responses received.
_ExpectedResponses	DWord	Read/Write	The total number of expected responses received.
_LastResponseTime	String	Read Only	The time at which the last valid response was received.
_LateData	DWord	Read/Write	The total number of times that a driver tag's data update occurred later than expected (based on the specified scan rate).
_MsgResent	DWord	Read/Write	The total number of messages sent as a retry.
_MsgSent	DWord	Read/Write	The total number of messages sent initially.
_MsgTotal	DWord	Read Only	The total number of messages sent (both _MsgSent + _MsgResent).
_PercentReturn	Float	Read Only	The proportion of expected responses (Received) to initial sends (Sent) as a percentage.
_PercentValid	Float	Read Only	The proportion of total valid responses received (_ TotalResponses) to total requests sent (_MsgTotal) as a percentage.
_Reset	Bool	Read/Write	Resets all diagnostic counters. Writing to the _Reset Tag causes all diagnostic counters to be reset at this level.
_RespBadChecksum	DWord	Read/Write	The total number of responses with checksum errors.
_RespTimeouts	DWord	Read/Write	The total number of messages that failed to receive any kind of response.
_RespTruncated	DWord	Read/Write	The total number of messages that received only a partial response.
_TotalResponses	DWord	Read Only	The total number of valid responses received (_ ErrorResponses + _ExpectedResponses).

Statistical items are not updated in simulation mode *(see device general properties)*.

Error Descriptions

The following categories of error/warning messages may be generated. Click on the link for list of messages.

See Also:
Modbus Exception Codes

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.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically 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.

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

Error Type:
Warning

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

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 statically 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.

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

Error Type:

Warning

Possible Cause:
A tag address that has been specified statically 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 statically 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 statically 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.

Device address <address> is read only.

Error Type:
Warning

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

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

Serial Communications

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

Serial Communications
COMn does not exist.
COMn is in use by another application.
Error opening COMn [OS Error == <OS Error ID>].
Serial communications error on channel <channel name> [<error mask>].
Unable to set comm parameters on COMn [OS Error == <OS Error ID>].

COMn does not exist.

Error Type:
Fatal

Possible Cause:
The specified COM port is not present on the target computer.

Solution:
Verify that the proper COM port has been selected.

COMn is in use by another application.

Error Type:
Fatal

Possible Cause:
The serial port assigned to a device is being used by another application.

Solution:

1. Verify that the correct port has been assigned to the channel.
2. Verify that only one copy of the current project is running.

Error opening COMn [OS Error == <OS Error ID>].

Error Type:
Fatal

Possible Cause:
The specified COM port could not be opened due to an internal hardware or software problem on the target computer.

Solution:
Verify that the COM port is functional and may be accessed by other Windows applications.

Serial communications error on channel <channel name> [<error mask>].

Error Type:
Serious

Error Mask Definitions:
B = Hardware break detected.
F = Framing error.
E = I/O error.
O = Character buffer overrun.
R = RX buffer overrun.
P = Received byte parity error.
T = TX buffer full.

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications parameters for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications parameters match those of the device.

Unable to set comm parameters on COMn [OS Error == <OS Error ID>].

Error Type:
Fatal

Possible Cause:
The serial parameters for the specified COM port are not valid.

Solution:
Verify the serial parameters and make any necessary changes.

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>.
Unable to write to address <address> on device <device>: Device responded with exception code <code>.
Write failed for <tag name> on device <device name>. Maximum path length of <number> characters exceeded.

Device <device name> is not responding.

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the Host PC is broken.
2. The communications parameters for the serial 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 PLC device.
2. Verify that the specified communications 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 serial connection between the device and the host PC is broken.
2. The communications parameters for the serial connection are incorrect.
3. The named device may have been assigned an incorrect network ID.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications parameters match those of the device.

3. Verify that the Network ID given to the named device matches that of the actual device.

Unable to write to address <address> on device <device>: Device responded with exception code <code>.

Error Type:
Warning

Possible Cause:
See **Modbus Exception Codes** for a description of the exception code.

Solution:
See **Modbus Exception Codes**.

Write failed for <tag name> on device <device name>. Maximum path length of <number> exceeded.

Error Type:
Warning

Possible Cause:
Path length is limited to the indicated number of characters.

Solution:
Devise a shorter path.

Lufkin Modbus Specific Messages

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

Lufkin Modbus Specific Messages
Bad address in block [<start address> to <end address>] on device <device name>.
Bad array spanning [<address> to <address>] on device <device>.

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

Error Type:
Serious

Possible Cause:

1. An attempt has been made to reference a nonexistent location in the specified device.
2. An attempt has been made to read more registers than allowed by the protocol.

Solution:

1. Verify the tags assigned to addresses in the specified range on the device and eliminate ones that reference invalid locations.
2. Decrease the register **block size** value to 125.

See Also:
Error Handling
Block Sizes

Bad array spanning [<address> to <address>] on device <device>.

Error Type:
Serious

Possible Cause:

1. An attempt has been made to reference a nonexistent location in the specified device.
2. An attempt has been made to read more registers than allowed by the protocol.

Solution:

1. Verify that all the register addresses requested in the array exist in the device and reduce the array size such that only valid addresses (that exist in the device) are requested by the array.
2. Reduce the array size value to 125.

See Also:
Error Handling
Block Sizes

Modbus Exception Codes

The following data is from Modbus Application Protocol Specifications documentation.

Code Dec/Hex	Name	Meaning
01/0x01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example, because it is unconfigured and is being asked to return register values.
02/0x02	ILLEGAL DATA ADDRESS	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with offset 96 and length 4 would succeed. A request with offset 96 and length 5 will generate exception 02.
03/0x03	ILLEGAL DATA VALUE	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the MODBUS protocol is unaware of the significance of any particular value of any particular register.
04/0x04	SLAVE DEVICE FAILURE	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.
05/0x05	ACKNOWLEDGE	The slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the master. The master can next issue a Poll Program Complete message to determine if processing is completed.
06/0x06	SLAVE DEVICE BUSY	The slave is engaged in processing a long-duration program command. The master should retransmit the message later when the slave is free.
07/0x07	NEGATIVE ACKNOWLEDGE	The slave cannot perform the program function received in the query. This code is returned for an unsuccessful programming request using function code 13 or 14 decimal. The master should request diagnostic or error information from the slave.
08/0x08	MEMORY PARITY ERROR	The slave attempted to read extended memory, but detected a parity error in the memory. The master can retry the request, but service may be required on the slave device.
10/0x0A	GATEWAY PATH UNAVAILABLE	Specialized use in conjunction with gateways indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request. This usually means that the gateway is misconfigured or overloaded.
11/0x0B	GATEWAY TARGET DEVICE FAILED TO RESPOND	Specialized use in conjunction with gateways indicates that no response was obtained from the target device. This usually means that the device is not present on the network.

Note: For this driver, the terms Slave and Unsolicited are used interchangeably.

Index

A

B

C

D

E

F

L

M

N

O

P

S

T

U

W