Cutler-Hammer ELC Ethernet Driver Help

© 2011 Kepware Technologies

Table of Contents

Cutler-Hammer ELC Ethernet Driver Help

Help version 1.021

CONTENTS

Overview

The Cutler-Hammer ELC Ethernet driver provides an easy and reliable way to connect Cutler-Hammer Eaton Logic Controllers (ELC) to OPC Client applications, including HMI, SCADA, Historian, MES, ERP and countless custom applications.

Device Setup

Supported Devices
Cutler-Hammer ELC model - PV28 Series.

Communication Protocol
Modbus TCP.

Maximum Number of Channels and Devices
The maximum number of channels supported by this driver is 256; the maximum number of devices is 8192.

Port Number
This parameter specifies the port number that the remote device is configured to use. The ELC Ethernet driver will use this port number when making solicited requests to a device. Valid port numbers are 0 to 65535. The default port number is 502.

Device ID
Device IDs are used to specify the device IP address, along with a Modbus Bridge Index on the Ethernet network. They are specified as <HOST>.XXX, where HOST is a standard UNC/DNS name or and IP address. The XXX designates the Modbus Bridge Index of the device and can be in the range of 0 to 255. If no bridge is used, set the index to 0.

Examples
1. Request data from a device with IP address 205.167.7.19. The Device ID should be entered as <205.167.7.19>.0

2. Request data from a device connected to bridge index 5 of a Modbus Ethernet bridge with an IP address of 205.167.7.5. The Device ID should be entered as <205.167.7.5>.5.

Data Encoding
First Word Low: Two consecutive 16 bit registers' addresses in an ELC device are used for 32 bit data types. Users can specify whether the driver should assume the first word is the low or the high word of the 32 bit value. Check this box to use the same word order as the ELCSoft programming software.

Block Size
Discrete Output
The output block size for ELC memory types S, Y, To, M and Co may be between 8 and 1024 in multiples of 8. The default setting is 1024.

Discrete Input
The input block size for ELC memory type X may be between 8 and 1024 in multiples of 8. The default setting is 1024.

Holding Registers
The holding register size for ELC memory types C, T and D may be between 1 and 120. The default setting is 120. The number of 32 bit counter values that can be blocked together will be one half the number of the holding registers setting, thus yielding an equivalent number of bytes per block.

Cable Diagrams

Cable Connections

Patch Cable (Straight Through)

TD +	1	OR/WHT	OR/WHT	1	TD +
TD −	2	OR	OR	2	TD −
RD +	3	GRN/WHT	GRN/WHT	3	RD +
	4	BLU	BLU	4	
	5	BLU/WHT	BLU/WHT	5	
RD −	6	GRN	GRN	6	RD −
	7	BRN/WHT	BRN/WHT	7	
	8	BRN	BRN	8	

RJ45   RJ45

10 BaseT

1 2 3 4 5 6 7 8
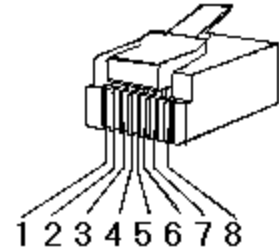
8-pin RJ45

Crossover Cable

TD +	1	OR/WHT	GRN/WHT	1	TD +
TD −	2	OR	GRN	2	TD −
RD +	3	GRN/WHT	OR/WHT	3	RD +
	4	BLU	BLU	4	
	5	BLU/WHT	BLU/WHT	5	
RD −	6	GRN	OR	6	RD −
	7	BRN/WHT	BRN/WHT	7	
	8	BRN	BRN	8	

RJ45   RJ45

Optimizing Cutler-Hammer ELC Ethernet Communications

The Cutler-Hammer ELC Ethernet driver has been designed to provide the best performance with the least amount of impact on the system's overall performance. While the Cutler-Hammer ELC 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 Cutler-Hammer ELC 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 Modbus controller from which data will be collected. While 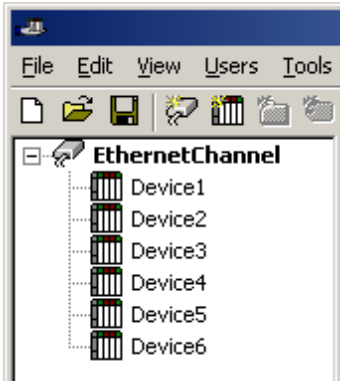this approach to defining the application will provide a high level of performance, it won't take full advantage of the Cutler-Hammer ELC Ethernet driver or the network. An example of how the application may appear when configured using a single channel is shown below.

Each device appears under a single Cutler-Hammer ELC 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 Cutler-Hammer ELC Ethernet Driver driver could only define one single channel, then the example shown above would be the only option available; however, the Cutler-Hammer ELC Ethernet Driver driver can define up to 256 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.
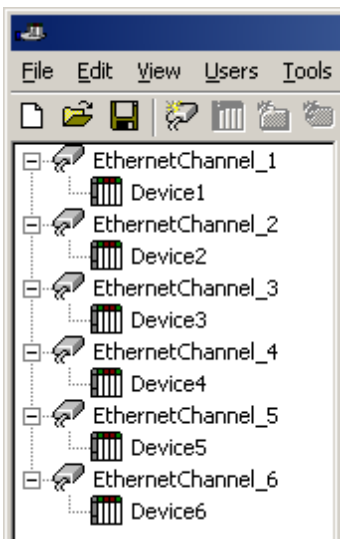
Each device has now been defined under its own channel. In this new configuration, a single path of execution is dedicated to the task of gathering data from each device. If the application has 256 or fewer devices, it can be optimized exactly how it is shown here.

The performance will improve even if the application has more than 256 devices. While 256 or fewer devices may be ideal, the application will still benefit from additional channels. Although by spreading the device load across all 256 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.

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 Supported on all models and includes HiLo LoHi byte order selection.
Double*	64 bit Floating point value The driver interprets four consecutive registers as a double precision value by making the last two registers the high DWord and the first two registers the low DWord.
Double Example	If register 40001 is specified as a double, bit 0 of register 40001 would be bit 0 of the 64 bit data type and bit 15 of register 40004 would be bit 63 of the 64 bit data type.
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.

*The descriptions assume the default first DWord low data handling of 64 bit data types, and first word low data handling of 32 bit data types.

Address Descriptions

PV28 Series

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

Memory Type	Range	Data Type	Access*	Remarks
Discrete Input (external I/O)	X0-X377 (octal)	**Boolean**	Read Only	
Discrete Output (external I/O)	Y0-Y377 (octal)	**Boolean**	Read/Write	
Main Relay	M0-M4095	**Boolean**	Read/Write	
Timer Status	To0-To255	**Boolean**	Read/Write	Contact = ON when timer reaches preset value
Counter Status (16 bit counters)	Co0-Co199	**Boolean**	Read/Write	Contact = ON when counter reaches preset value
Counter Status (32 bit counters)	Co200-Co255	**Boolean**	Read/Write	Contact = ON when counter reaches preset value
Step Point	S0-S1023	**Boolean**	Read/Write	Sequential Function Chart usage
Timer Current Value	T0-T255 Txxx.0-Txxx.15	**Word**, Short, BCD Float, DWord, Long LBCD, Double Boolean	Read/Write	Represents the current value of the 1mS, 10mS, or 100mS timers
Counter Current Value (16 bit counters)	C0-C199 Cxxx.0-Cxxx.15	**Word**, Short, BCD Float, DWord, Long LBCD, Double Boolean	Read/Write	Represents the current value of the counter
Counter Current Value (32 bit counters)	C200-C255 Cxxx.0-Cxxx.31	**DWord**, Float Long, LBCD, Double Boolean	Read/Write Read Only	Represents the current value of the counter
Data Register	D0-D9999 Dxxx.0-Dxxx.15	**Word**, Short, BCD Float, DWord, Long LBCD, Double Boolean	Read/Write	General data storage
Data Register as String with HiLo Byte Order	D0.2H-D9999.32H	**String****	Read/Write	.Bit is string length, range 2 to 32 bytes
Data Register as String with LoHi Byte Order	D0.2L-D9999.32L	**String****	Read/Write	.Bit is string length, range 2 to 32 bytes

*Write Only Access

All Read/Write addresses may be set as Write by prefixing a "W" to the address such as "WD0", which 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, then 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.

**String Support

The ELC devices support reading and writing data register memory as an ASCII string. When using data 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 32 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 an "H" or "L" to the address.

Examples

1. To address a string starting at D200 with a length of 32 bytes and HiLo byte order, enter:
D200.32H

2. To address a string starting at D500 with a length of 32 bytes and LoHi byte order, enter:

D500.32L

Note: The string length of may be limited by the maximum size of the write request that the device will allow. If while utilizing a string tag you receive an error message in the server event window of "Unable to write to address <address> on device <device>: Device responded with exception code 3." This means the device did not accept the length of the string. If possible, try shortening the string.

Normal Address Example

The 177'th (octal) output coil would be addressed as 'Y177' using octal addressing.

Array Support

Arrays are supported for registers (i.e. T, C, and D addresses) except for bit-within-word and strings. Arrays are also supported for input and output coils (Boolean data types, i.e. M, S, X, Y, To, and Co addresses). There are two methods of addressing an array. Examples are given using register addresses.

Dxxx [rows] [cols]
Dxxx [cols] this method assumes rows is equal to one

For arrays, rows multiplied by cols cannot exceed the block size that has been assigned to the device for the register/coil type. For register 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>'
Unable to write to address '<address>' on device '<device>'. Device responded with exception code <code>

Device Specific Messages
Failure to initiate 'winsock.dll' (Error: <error code>)
Unable to create a socket connection for device '<device>'
Bad array spanning [<start address> to <end address>] on device'<device>'
Bad address in block [<start address> to <end address>] on device '<device>'
Device '<device>' block request [<start address> to <end address>] responded with exception <code>

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 statically 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 statically 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 statically references a location that is beyond the range of supported locations for the device.

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

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.

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>' 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.

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 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 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.

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>

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.

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>' 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:

Please refer to the Modbus Exception Codes table below.

Solution:

Solution will depend upon the Exception Code. Please refer to the table below.

Modbus Exception Codes

The following information is from the Modbus Application Protocol Specifications documentation.

Code Dec/Hex	Name	Meaning
01/0x01	ILLEGAL FUNC-TION	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 mis-configured 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. Usually means that the device is not present on the network.

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' (Error: <error code>)
Unable to create a socket connection for device '<device>'
Bad array spanning [<start address> to <end address>] on device'<device>'
Bad address in block [<start address> to <end address>] on device '<device>'
Device '<device>' block request [<start address> to <end address>] responded with exception <code>

Failure to initiate 'winsock.dll' (Error: <error code>)

Error Type:
Fatal

OS Error Code 10091:
Indicates that the underlying network subsystem is not ready for network communication. Wait a few seconds and restart the driver.

OS Error Code 10067:
Limit on the number of tasks supported by the Windows Sockets implementation has been reached. Close one or more applications that may be using Winsock and restart the driver.

Unable to create a socket connection for device '<device>'

Error Type:
Serious

Possible Cause:
Either the device is offline or there is a Network problem.

Solution:
Make sure device is online and/or check network hardware.

Bad array spanning [<start address> to <end address>] on device '<device>'

Error Type:
Warning

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

Solution:
Adjust array bounds so that all locations within range are valid.

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

Error Type:
Warning

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.

Device '<device>' block request [<start address> to <end address>] responded with exception <code>

Error Type:
Warning

Possible Cause:
Please refer to the Modbus Exception Codes table below.

Solution:
Solution will depend upon the Exception Code. Please refer to the table below.

Modbus Exception Codes
The following information is from the Modbus Application Protocol Specifications documentation.

Code Dec/Hex	Name	Meaning
01/0x01	ILLEGAL FUNC-TION	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 mis-configured 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. Usually means that the device is not present on the network.

Index