GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Instruction Manual

Manual P/N: 1601-0235-A1
Manual Order Code: GEK-113195

GE Multilin

215 Anderson Avenue, Markham, Ontario

Canada L6E 1B3

Tel: (905) 294-6222 Fax: (905) 201-2098

Internet: http://www.GEmultilin.com

REGISTERED ISO9001:2000 GE MULTILIN

GE Multilin's Quality Management System is registered to ISO9001:2000

QMI # 005094
UL # A3775

GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Table of contents

INTRODUCTION

Getting started 1–1
- Inspecting the package and product 1–1
- Contact information 1–1

Document conventions 1–2
- Description 1–2
- Glossary 1–2

About the D485 Modbus to DeviceNet Converter 1–3
- Application 1–3

Features 1–3
- General features 1–3
- Modbus network 1–3
- DeviceNet interface features 1–3

Ordering 1–4
- Order codes 1–4

Specifications 1–4
- Mechanical 1–4
- Electrical characteristics 1–4
- Communications 1–4
- Environmental 1–4
- EMC compliance 1–4

INSTALLATION

Quick install 2–1
- Procedure 2–1

Electrical installation 2–1
- Overview 2–1
- DeviceNet connector 2–2
- Configuration cable 2–3
- Modbus connector 2–4
- Power connector 2–4

Mechanical installation 2–5
- DIN-rail mounting 2–5

Indicators and switches 2–5
- Status indicators 2–5
- Configuration switches 2–6

DeviceNet installation procedure 2–7
DeviceNet configuration tool 2–7
DeviceNet network termination 2–7
Links 2–7
Troubleshooting 2–8
Description 2–8

DATA EXCHANGE

Overview 3–1
Description 3–1
Internal memory buffer structure 3–2
I/O data vs. explicit data 3–3
Memory Map 3–4
Memory locations 3–4
Protocol configuration 3–5
Description 3–5
Communication mode 3–5
Protocol building blocks 3–5
DeviceNet object implementation 3–6
Overview 3–6
Identity object (class 01h) 3–6
Message router (class 02h) 3–7
DeviceNet object (class 03h) 3–7
Assembly object (class 04h) 3–8
Connection object (class 05h) 3–8
Acknowledge handler object (class 2Bh) 3–10
I/O data input mapping object (class A0h) 3–11
I/O data output mapping object (class A1h) 3–11
Diagnostic object (class AAh) 3–12
Parameter data input mapping object (class B0h) 3–12
Parameter data output mapping object (class B1h) 3–13

SOFTWARE OVERVIEW

Introduction 4–1
Description 4–1
System requirements 4–1
Installation procedure 4–1
Description 4–1
Installing from EnerVista CD 4–1
Installing from the GE Multilin website 4–1
Using the EnerVista P485/D485 Setup software 4–2
Description 4–2
Configuration wizard 4–2
Select fieldbus type 4–3
Sub-network properties 4–4
Device types 4–5
Connecting devices 4–7
Selecting parameters for each node 4–8
Configuration report 4–8
Configuration main window 4–9
Description 4–9
Navigation window 4–10
Parameter window 4–10
Information window 4–10
Configuration line indicator 4–10

Options window 4–11
Fieldbus configuration 4–11
Description 4–11
P485/D485 configuration 4–12
Parameter window 4–12
Modbus network configuration 4–13
Overview 4–13
Serial interface settings 4–13

COMMUNICATION MODEL

Introduction 5–1
Description 5–1
Scan list 5–2
Basic settings 5–2
Network settings 5–2
Communication 5–2
Message delimiter 5–2
Nodes 5–3
Description 5–3
Node parameters 5–3
Modbus network menu 5–3
Node menu 5–3
Query parameters 5–4
Response parameters 5–5

FRAME AND COMMAND EDITORS

Frame editor 6–1
Description 6–1
Example 6–1
Command editor 6–2
General 6–2
Specifying a new command 6–3

MODBUS NETWORK AND NODE MONITORS

Modbus network monitor 7–1
General 7–1
Operation 7–2
Node monitor 7–3
General 7–3
Operation 7–3

ADVANCED FUNCTIONS

Control and status registers 8–1
Description 8–1
Control register (DeviceNet control system to D485) 8–1
Control codes 8–2
Status register (D485 to fieldbus control system) 8–2
Status codes 8–3
Handshaking procedure 8–3
Input/output data during startup 8–4
Description 8–4
Advanced fieldbus configuration 8–5
Mailbox command 8–5
Parameter data input area mapping 8–5

Parameter data output area mapping 8–7
I/O data input area mapping 8–8
I/O data output area mapping 8–10

APPLICATION EXAMPLE

Introduction 9–1
Overview 9–1
Equipment and documents 9–1
System setup 9–2
Modbus user map setup 9–3
Description 9–3
PQMII user map 9–3
MM2 user map 9–3
System configuration 9–5
Overview 9–5
Installing the EnerVista P485/D485 Setup software 9–6
Starting the configuration wizard 9–6
Step 1: Selecting the fieldbus type 9–6
Step 2: Selecting the sub-network properties 9–7
Step 3: Include device types 9–7
Step 4: Connect devices to the sub-network 9–9
Step 5: Select parameters for each node 9–10
Step 6: Configuration report 9–11
Saving device data 9–12
Configuring the queries 9–12
Grouping I/O data 9–15
I/O data Input mapping 9–15
Parameter data input area mapping 9–17
Downloading the configuration file 9–18
DeviceNet network setup 9–19
Description 9–19
Selecting the input attribute for polling and COS 9–19

MISCELLANEOUS

Revision history 10–1
Release dates 10–1
Changes to the manual 10–1
Warranty 10–1
GE Multilin warranty statement 10–1

GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Chapter 1: Introduction

Getting started

INSPECTING THE PACKAGE AND PRODUCT

Examine the shipping container for obvious damage prior to installing this product; notify the carrier of any damage that you believe occurred during shipment or delivery. Inspect the contents of this package for any signs of damage and ensure that the items listed below are included.

Remove the items from the shipping container. Be sure to keep the shipping container should you need to re-ship the unit at a later date.

In the event there are items missing or damaged, contact the party from whom you purchased the product. If the unit needs to be returned, please use the original shipping container, if possible.

CONTACT INFORMATION

GE Multilin contact information and call center for product support is shown below:

GE Multilin
215 Anderson Avenue
Markham, Ontario
Canada L6E 1B3

- Telephone: 905-294-6222 or 1-800-547-8629 (North America), +34 94 485 88 00 (Europe)

Fax: 905-201-2098 (North America),
+34 94 485 88 45 (Europe)

- E-mail: multilin.tech@ge.com
- Website: http://www.GEmultilin.com

Document conventions

DESCRIPTION

The following conventions are used throughout this document:

- Numbered lists provide sequential steps.
- Bulleted lists provide information, not procedural steps.
- The term 'user' refers to the person or persons responsible for installing the D485 Modbus to DeviceNet Converter in a network.
- Hexadecimal values are written in the format 0xNNNN, where NNNN is the hexadecimal value.
- Decimal values are represented as NNNN, where NNNN is the decimal value.
- As in all communication systems, the terms "input" and "output" can be ambiguous, since their meaning depends on which end of the link is being referenced. The convention in this document is that "input" and "output" are always being referenced to the master/scanner end of the link (see illustration below).
- The term "sub-network" is interchangeably used for "Modbus network".

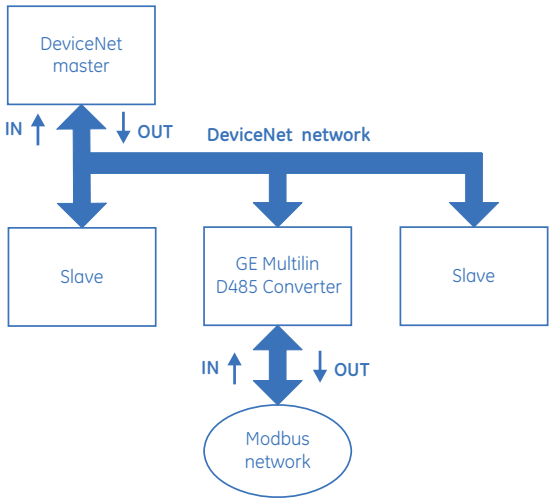


Figure 1-1: Input and output definition

GLOSSARY

The following terminology is used in the D485 manual:

- Broadcaster: A protocol specific node in the sub-network scan that holds transactions destined for all nodes.
- Command: A protocol specific transaction.
- Fieldbus: The network to which the converter is connected (DeviceNet for D485).
- Frame: A higher level series of bytes forming a complete telegram on the sub-network (Modbus).
- Monitor: A tool for debugging the D485 and network connections.
- Node: A device in the scan list that defines the communication with a slave (GE relay) on the Modbus sub-network.
- Scan list: List of configured slaves with transactions on the sub-network.
- Sub-network: Modbus network that logically is located on a subsidiary level with respect to the fieldbus and to which the D485 acts as a gateway.
- Transaction: A generic building block that is used in the sub-network scan list and defines the data that is sent out the sub-network.

About the D485 Modbus to DeviceNet Converter

APPLICATION

The D485 Modbus to DeviceNet Converter (or D485) acts as a gateway between the Modbus protocol and a DeviceNet network. Integration of industrial devices is enabled without loss of functionality, control, and reliability, both when retrofitting to existing equipment as well as when setting up new installations.

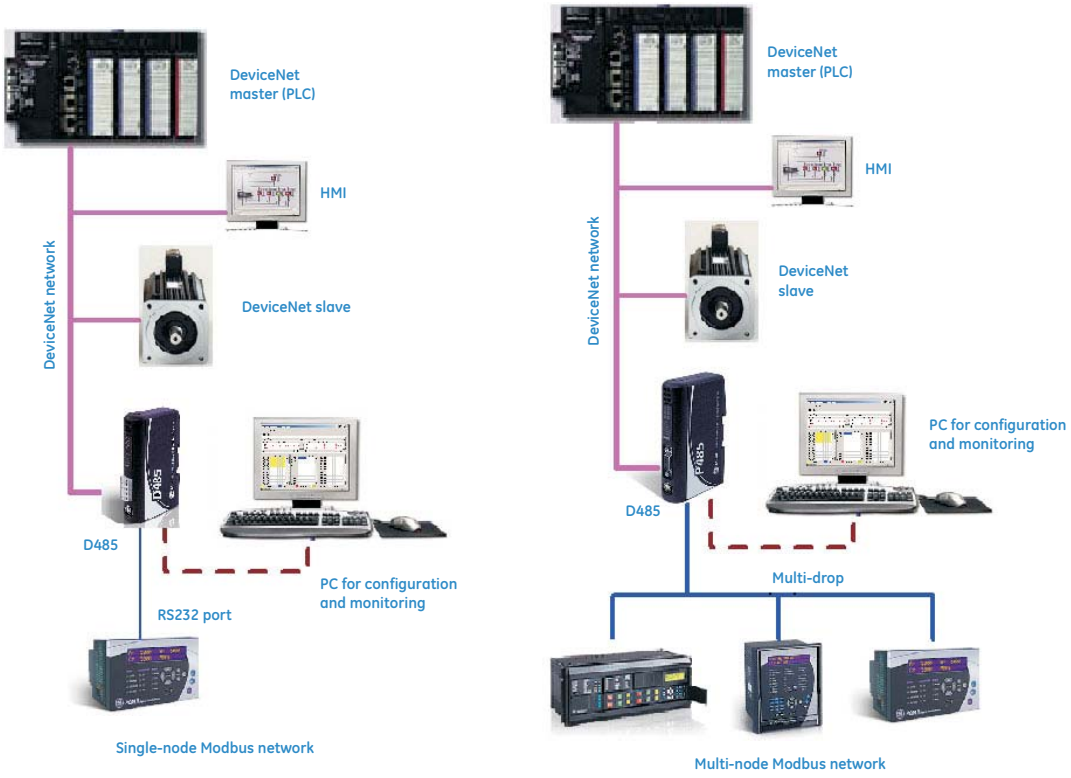


Figure 1-2: Typical applications

Features

GENERAL FEATURES

- DIN-rail mountable
- Save/load configuration in flash
- CE, UL, and cUL marked

MODBUS NETWORK

- RS232/RS422/RS485 communications
- Multi-drop or single-node configurations possible
- Modbus RTU Master mode
- Supports up to 50 commands
- Configuration via EnerVista P485/D485 Setup software

DEVICENET INTERFACE FEATURES

- Complete DeviceNet adapter functionality (profile no. 12 group 2 only server)
- Group two only server
- MAC ID and baud rate configuration via on-board switches
- Polled, Change of State (COS), and Bit Strobed I/O
- Explicit messaging

Ordering

ORDER CODES

The following table illustrates the order codes for the D485 Modbus to DeviceNet Converter.

Table 1–1: D485 order codes

	D485	–	*	
Base unit	D485		**|**	D485 Modbus to DeviceNet Converter
			C	With configuration cable
			X	No configuration cable

Specifications

MECHANICAL

HOUSING

Plastic housing with snap-on connection to DIN-rail, protection class IP20

DIMENSIONS

L x W x H:	120 mm × 75 mm × 27 mm (4.72-in × 2.95-in × 1.06-in)

PROTECTION CLASS

Protection class:	IP20

ELECTRICAL CHARACTERISTICS

POWER SUPPLY

Power:	24 V ±10% (for use in class 2 circuits)

POWER CONSUMPTION

Maximum power consumption: 280 mA on 24 V

Typically power consumption: approximately 100 mA

COMMUNICATIONS

BAUD RATES

Baud rate (DeviceNet)	125, 250, and 500 kbps
Baud rate (Modbus)	1200, 2400, 4800, 9600, 19200, 38400, and 57600 bps

I/O DATA

I/O input size:	512 bytes
I/O output size:	512 bytes

FEATURES AND INTERFACE

Supported features:	bit strobe, polling, cyclic and COS I/O messaging, explicit messaging
Modbus interface:	RS232, RS422, RS485
DeviceNet interface:	5-pin linear DeviceNet plug (Phoenix type)

ENVIRONMENTAL

RELATIVE HUMIDITY

The product is designed for a relative humidity of 0 to 95% non-condensing

TEMPERATURE

Operating:	0 to 55°C
Non Operating:	–5 to 85°C

EMC COMPLIANCE

CE-MARK

Certified according to European standards unless otherwise is stated

Emission:	according to EN 50081-2:1993
Immunity:	according to EN 61000-6-2:1999

UL/C-UL COMPLIANCE

This unit is an open type listed by the Underwriters Laboratories.

The certification has been documented by UL in file E214107.

D485 Modbus to DeviceNet Converter

Chapter 2: Installation

Quick install

PROCEDURE

1. Snap the D485 on to the DIN-rail (see *DIN-rail mounting* on page 2–5).
2. Connect the DeviceNet cable (see *DeviceNet connector* on page 2–2).
3. Connect the serial Modbus network cable (see *Modbus connector* on page 2–4 for details).
4. Connect a PC using the configuration cable (see *Configuration cable* on page 2–3).
5. Connect the power cable and apply power to the unit (see *Power connector* on page 2–4 for details).
6. Start the EnerVista P485/D485 Setup software.
7. Normally, the EnerVista P485/D485 Setup detects the correct serial port. If this does not occur, select the correct port through the **Port** menu item.
8. Configure the D485 using EnerVista P485/D485 Setup and download the configuration to the unit.
9. Configure and power-up the Modbus network device for communication.

Electrical installation

OVERVIEW

The location of the various electrical connectors is shown below.

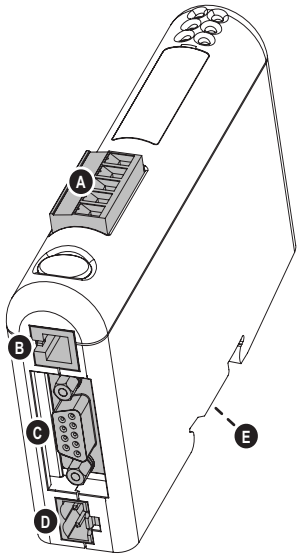


Figure 2-1: D485 electrical connections

DEVICENET CONNECTOR

The DeviceNet connector is indicated as **A** in the figure above. This connector is used to connect the D485 to the DeviceNet network.

The pin assignments for the DeviceNet connector are shown below.

Table 2–1: DeviceNet connector pin assignments

Pin	Signal	Description
1	V–	DeviceNet bus power, negative supply voltage
2	CAN L	CAN L bus line
3	Shield	Cable shield
4	CAN H	CAN H bus line
5	V+	DeviceNet bus power, positive supply voltage

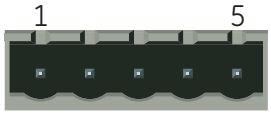


Figure 2-2: DeviceNet connector

CONFIGURATION CABLE

The PC connector is indicated as **B** in Figure 2-1: D485 electrical connections on page 2–2. This connector is used to connect the D485 to a PC using the configuration cable for configuration and monitoring purposes.

A configuration cable can be purchased from GE Multilin. The wiring for the configuration cable is shown below.

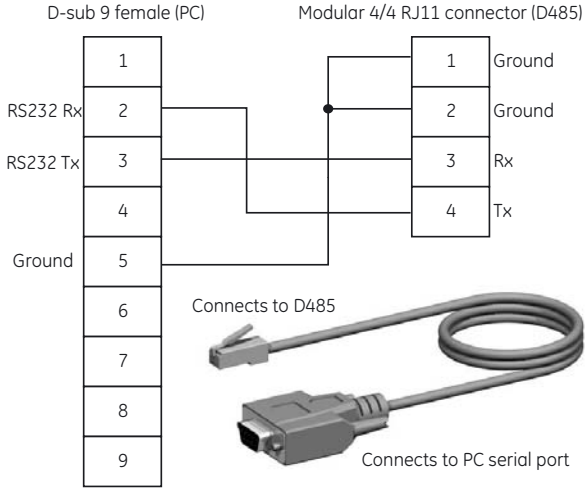


Figure 2-3: Configuration cable

The pinout for the modular 4/4 RJ11 connector (connects to the D485) is shown below.

Table 2–2: Configuration cable pin assignments (D485 end)

Pin	Description
1	Signal ground
2	Signal ground
3	RS232 Rx, data input to D485
4	RS232 Tx, data output from D485

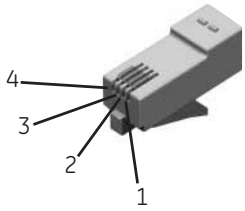


Figure 2-4: Configuration cable (D485 end)

The pinout for the DSUB 9-pin serial plug (connects to the PC) is shown below.

Table 2–3: Configuration cable pin assignments (PC end)

Pin	Description
1	Not connected
2	RS232 Rx, data input to PC
3	RS232 Tx, data output from PC
4	Not connected
5	Ground
6 to 9	Not connected

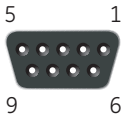


Figure 2-5: Configuration cable (PC end)

MODBUS CONNECTOR

The Modbus connector is indicated as **C** in Figure 2-1: D485 electrical connections on page 2–2. This connector is used to connect the D485 to the serial network. Based on the configuration selected in the EnerVista P485/D485 Setup software, the corresponding signals are activated.

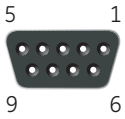


Figure 2-6: Modbus connector

Table 2–4: Modbus connector pin assignments

Pin	Description	RS232	RS422	RS485
1	+5 V output (50 mA max)			
2	RS232 Rx	✓		
3	RS232 Tx	✓		
4	Not connected			
5	Ground	✓	✓	✓
6	RS422 Rx +		✓	
7	RS422 Rx –		✓	
8	RS485 + / RS422 Tx+		✓	✓
9	RS485 – / RS422 Tx–		✓	✓

POWER CONNECTOR

The power connector is indicated as **D** in Figure 2-1: D485 electrical connections on page 2–2. Use this connector to apply power to the D485.

Pin 1: +24 V DC;
Pin 2: ground

NOTE

Use 60/75 or 75°C copper (CU) wire only. The terminal tightening torque must be between 5 to 7 lbs-in (0.5 to 0.8 nm).

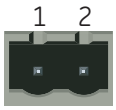


Figure 2-7: Power connector

Mechanical installation

DIN-RAIL MOUNTING

The DIN-rail connector is internally connected to the D485.

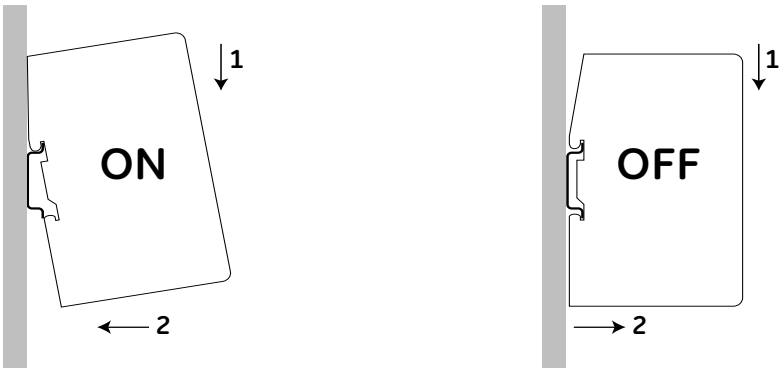


Figure 2-8: Mounting the D485 to the DIN-rail

To snap the D485 *on*, first press the D485 downwards (1) to compress the spring on the DIN-rail connector, then push the D485 against the DIN-rail as to make it snap on (2)

To snap the D485 *off*, push the D485 downwards (1) and pull it out from the DIN-rail (2), as to make it snap off from the DIN-rail.

Indicators and switches

STATUS INDICATORS

The status indicators for the D485 Modbus to DeviceNet Converter are indicated below.

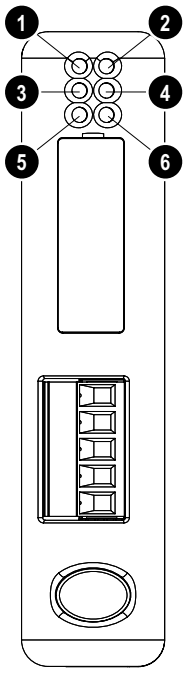


Figure 2-9: D485 status indicators

Table 2–5: D485 status indicators

Number	Description	State	Status
1	Network status	Off	Not powered / not online
		Green	Link OK, online, connected
		Green flashing	Online, not connected
		Red	Critical link failure
		Red flashing	Connection timeout
2	Module status	Off	No power to device
		Green	Device operational
		Green flashing	Data size larger than configured
		Red	Unrecoverable fault
		Red flashing	Minor fault
3	Not used	-	-
4	Not used	-	-
5	Modbus status *	Off	Power off
		Green, flashing	Initializing and not running
		Green	Running
		Red	Stopped or subnet error, or timeout
6	Device status	Off	Power off
		Red/green alternating	Invalid or missing configuration
		Green	Initializing
		Green flashing	Running
		Red flashing	If the device status LED is flashing in a sequence starting with one or more red flashes, note the sequence pattern and contact GE Multilin

* This LED turns green when all transactions have been active at least once. This includes any transactions using “change of state” or “change of state on trigger”. If a timeout occurs on a transaction, this LED will turn red.

CONFIGURATION SWITCHES

The configuration switches are used to set the DeviceNet MAC ID and baud rate settings. Normally, these switches are covered by a plastic hatch. Note that the node address can not be changed during runtime, i.e. the D485 requires a reset for any changes to have effect. Recycle the power supply to reset the module.

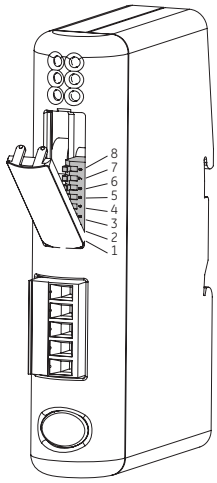


Figure 2-10: D485 configuration switches

The DeviceNet MAC ID and baud rate switches are set according to the following tables.

Table 2–6: Baud rate settings

Baud rate	Switch 1	Switch 2
125 kbps	OFF	OFF
250 kbps	OFF	ON
500 kbps	ON	OFF
Reserved	ON	ON

Table 2–7: MAC ID settings

MAC ID	Switch 3	Switch 4	Switch 5	Switch 6	Switch 7	Switch 8
0	OFF	OFF	OFF	OFF	OFF	OFF
1	OFF	OFF	OFF	OFF	OFF	ON
2	OFF	OFF	OFF	OFF	ON	OFF
3	OFF	OFF	OFF	OFF	ON	ON
↓	↓	↓	↓	↓	↓	↓
63	ON	ON	ON	ON	ON	ON

NOTE

When removing the hatch, avoid touching the circuit boards and components. Exercise caution when using tools to open the hatch.

DeviceNet installation procedure

DEVICENET CONFIGURATION TOOL

Each device on a DeviceNet network is associated with a ESD file, which contains all necessary information about the device. This file is used by the DeviceNet configuration tool during configuration of the network. The file is available for download at the GE Multilin website at http://www.GEmultilin.com (the ESD file is named 'D485.ESD').

It is necessary to import the ESD file in the DeviceNet configuration tool in order to incorporate the D485 as a slave in the DeviceNet network. The properties for the D485 must then be configured from the DeviceNet configuration tool. This includes setting up the node address, input/output data areas and DeviceNet baud rate.

- **Node address**: The node address in the DeviceNet configuration tool should be set to match the one selected using the on board configuration switches of the D485 (see *Configuration switches* on page 2–6).
- **Setting up input/output data areas**: To establish the connection with the master, the D485 mus be configured for correct I/O sizes.
- **Baud rate**: The DeviceNet network baud rate should match the D485 baud rate setting.

DEVICENET NETWORK TERMINATION

If the D485 is the last node on a DeviceNet network, it is necessary to use a DeviceNet network termination resistor of 120 ohms between the CAN L and CAN H terminals.

LINKS

Additional information about the DeviceNet fieldbus system can be found at http://www.odva.org.

Troubleshooting

DESCRIPTION

Problem during configuration upload/download. The Config Line LED turns red.

- Serial communication failed – try again.

The serial port seems to be available, but it is not possible to connect to the D485.

- The serial port may be in use by another application. Exit EnerVista P485/D485 Setup and close all other applications including the ones in the system tray and try again.
- Select another serial port and try again.

Poor performance.

- Right click 'Modbus Network' in the Navigation window and select 'Modbus Network Status' to see status/diagnostic information about the sub network. If the D485 reports very many retransmissions, check your cabling and/or try a lower baud rate setting for the sub network (if possible).
- Is the Modbus Network Monitor in EnerVista P485/D485 Setup active? The Modbus network monitor has a negative influence on the overall performance of the D485, and should only be used when necessary.
- Is the Node Monitor in EnerVista P485/D485 Setup active? The node monitor has a negative influence on the overall performance of the D485, and should only be used when necessary.

GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Chapter 3: Data Exchange

Overview

DESCRIPTION

Data from the fieldbus (DeviceNet) and the sub network (Modbus) is stored in an internal memory buffer. This is a easy method for data exchange where the fieldbus control system simply reads and writes data to pre-defined memory locations, and the serial sub network also use the same internal memory buffer to read and write data.

Refer to Figure 3-1: Data exchange overview on page 3–2 for additional details.

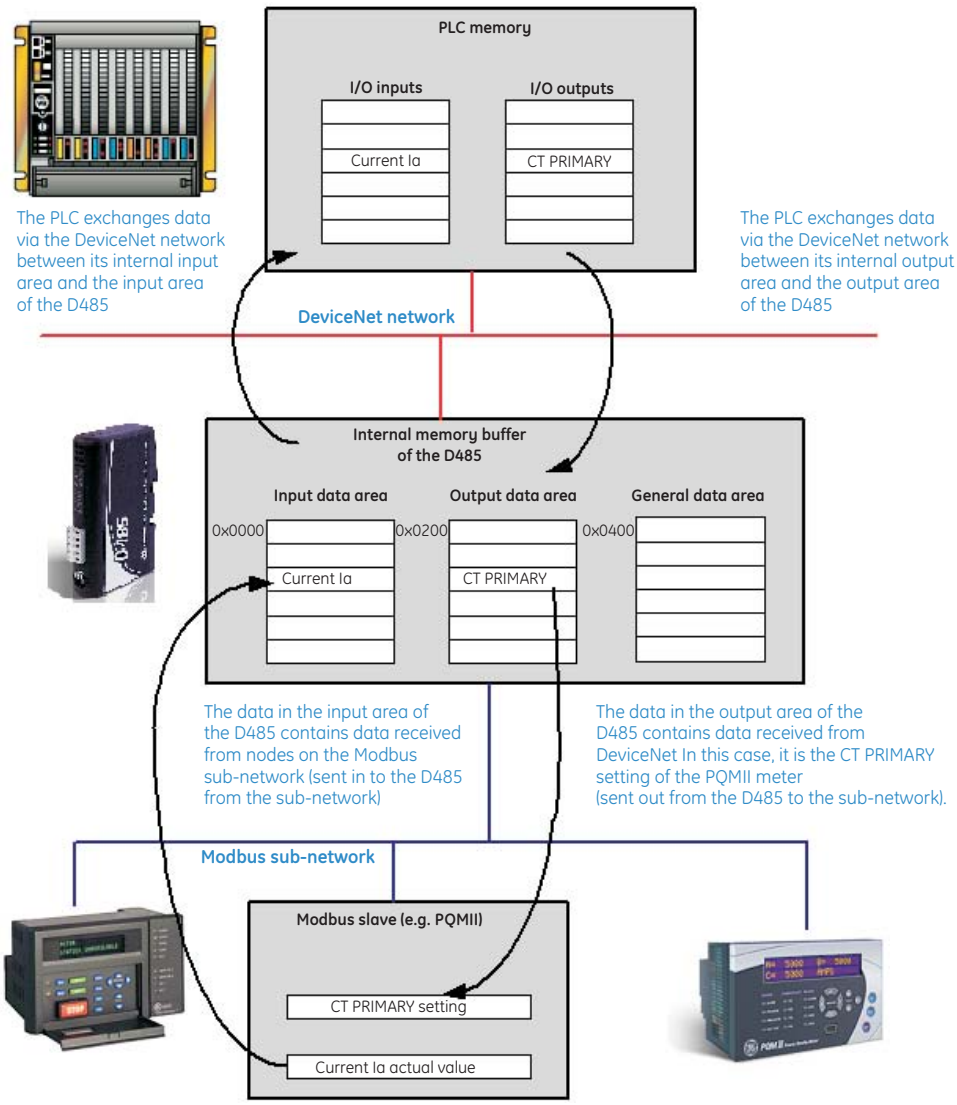


Figure 3-1: Data exchange overview

INTERNAL MEMORY BUFFER STRUCTURE

The internal memory buffer can be seen as a memory space with three different types of data; input data, output data and general data.

- **Input data**: This is data that should be sent to the fieldbus. The D485 can handle up to 512 bytes of input data.
- **Output data**: this is data recieved *from* the fieldbus. The D485 can handle up to 512 bytes of output data.
- **General data**: This data *cannot* be accessed from the fieldbus, and is used for transfers between nodes on the sub-network, or as a general "scratch pad" for data. The D485 can handle up to 1024 bytes of general data.

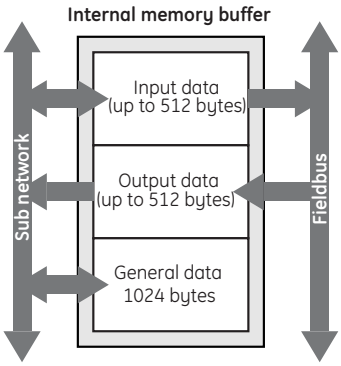


Figure 3-2: Internal memory buffer

Data exchange on the fieldbus interface is based on the standard DeviceNet objects, and five vendor specific objects. The vendor specific objects can be used to access the different memory areas. However, most applications requires only the standard DeviceNet objects.

For a complete description of the available objects, please refer to *DeviceNet object implementation* on page 3–6.

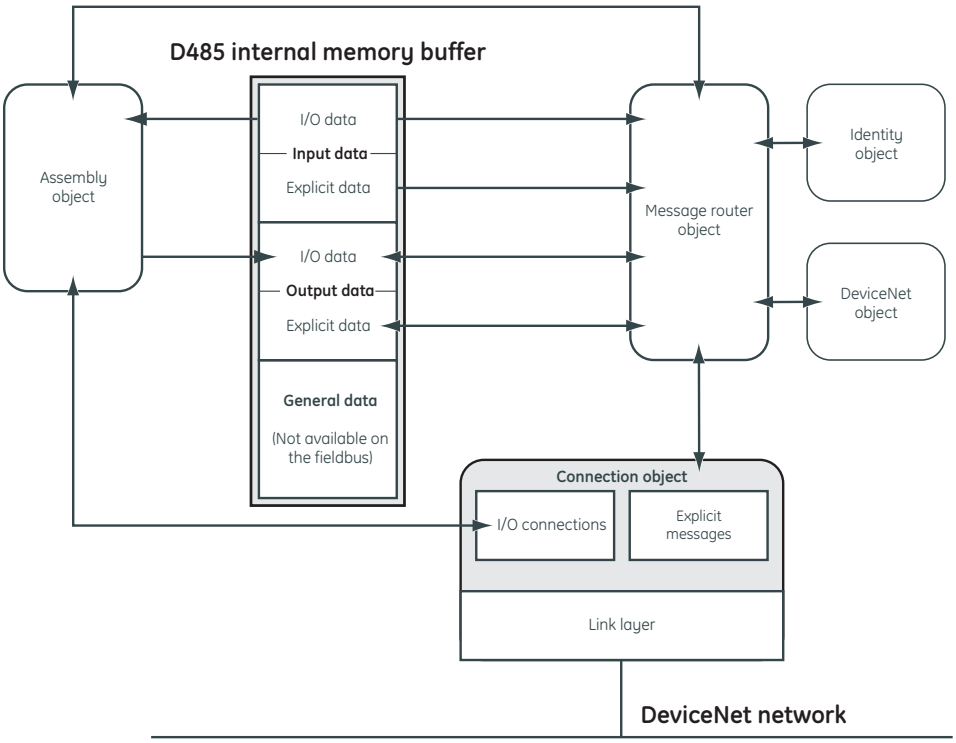


Figure 3-3: D485 memory structure

I/O DATA VS. EXPLICIT DATA

The input and output areas can hold two types of data:

- **I/O data**: This data is sent/recevied upon change of value, polling or cyclically.
- **Explicit data**: This data is updated on request from another node on the fieldbus.

The amount of data that should be treated as I/O data is determined by the 'I/O Sizes' parameter in the EnerVista P485/D485 Setup software. The remainder will be treated as explicit data. By default (that is, when using 'automatic' I/O sizes), all data is treated as I/O data.

For example, when using an input I/O size of 50 and an output I/O size of 60, the input and output data areas will be allocated as follows:

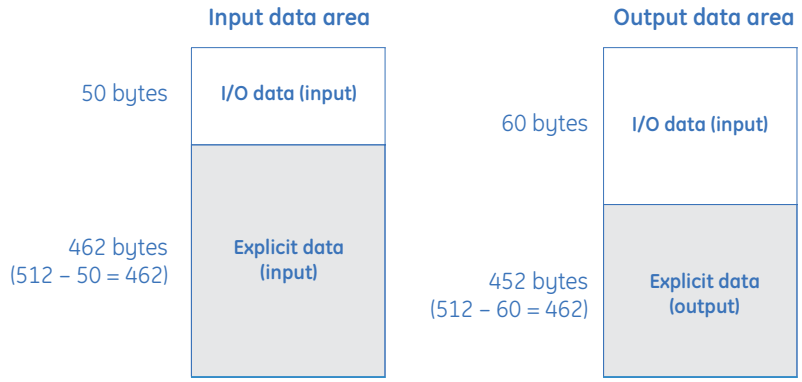


Figure 3-4: I/O data area example

Memory Map

MEMORY LOCATIONS

When configuring the sub-network, use the memory locations shown below:

Address	Contents	Access
0x0000 to 0x0001	Status register	read/write
0x0002 to 0x01FF	Input data area	read/write
0x0200 to 0x0201	Control register	read only
0x0202 to 0x03FF	Output data area	read only
0x0400 to 0x07FB	General data area	read/write

- **Status register** (0x0000 to 0x0001): If enabled, this register occupies the first two bytes in the input data area. For more information, see *Control and status registers* on page 8–1.
- **Input data area** (0x002 to 0x01FF): This area holds data that should be sent to the fieldbus (see the status and control registers).
- **Control register** (0x0200 to 0x0201): If enabled, these register occupies the first two bytes in the output data area. For more information, see *Control and status registers* on page 8–1.
- **Output data area** (0x0202 to 0x03FF): This area holds data received *from* the fieldbus. Data cannot be written to this area.
- **General data Area** (0x0400 to 0x07FB): This data *cannot* be accessed from the fieldbus, and should be used for transfers between nodes on the Modbus sub-network, or as a general “scratch pad” for data.

Protocol configuration

DESCRIPTION

In order to be able to communicate on the Modbus sub-network, the D485 must be supplied with a description of the required sub-net protocol. To accomplish this, the EnerVista P485/D485 Setup software features a flexible protocol-programming system, allowing the D485 to interpret and exchange data with almost any serial device on the Modbus sub-network.

COMMUNICATION MODE

The D485 supports the Modbus Master communication mode. In this mode, the D485 is setup to use the Modbus RTU protocol and implements a Modbus master for data exchange between the fieldbus and one or more devices on the sub-network. Refer to Chapter 5 for additional details.

PROTOCOL BUILDING BLOCKS

A description of the building blocks used to describe the sub-net protocol is shown below.

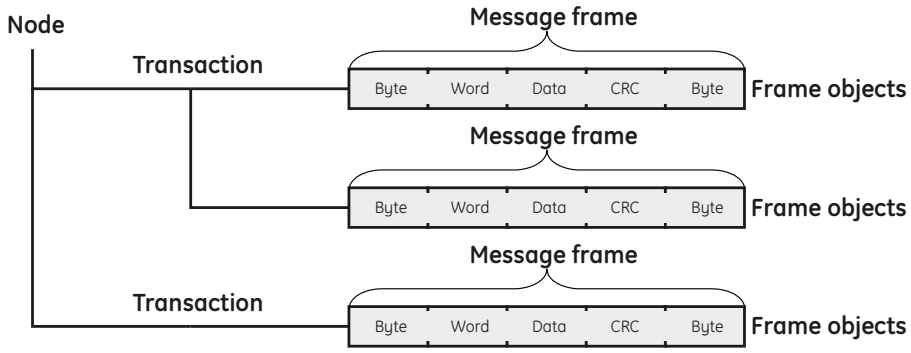


Figure 3-5: Modbus protocol blocks

- **Node**: In the D485, a node holds all transactions and parameters for a particular device on the sub network.
- **Transaction**: Transactions contains messages to be transmitted on the sub-network. A transaction consists of one or more message frames (see figure above), and has a few general parameters to specify how and when the transaction should be used on the sub-network.
- **Commands**: A command is a pre-defined transaction that has been stored in a list in the EnerVista P485/D485 Setup software. This improves readability as well as simplifying common operations by allowing transactions to be stored and reused.
- **Message frame**: The message frame contains a description of what is actually transmitted on the sub-network and consists of frame objects (see figure above).
- **Frame object**: Frame objects are used to compose a message frame. Frame objects include fixed values, dynamic values retrieved from a specified memory location in the D485, or strings.

DeviceNet object implementation

OVERVIEW

DeviceNet requires some mandatory objects; these are implemented as well as some vendor specific objects. The following objects are implemented:

Table 3–1: Mandatory objects

Object name	Class	Page
Identity object	01h	3-6
Message router object	02h	3-7
DeviceNet object	03h	3-7
Assembly object	04h	3-8
Connection object	05h	3-8
Acknowledge handler object	2Bh	3-10

Table 3–2: Vendor specific object

Object name	Class	Page
I/O data input mapping object	A0h	3-11
I/O data output mapping object	A0h	3-11
Diagnostic object	AAh	3-12
Parameter data input mapping object	B0h	3-12
Parameter data output mapping object	B1h	3-13

IDENTITY OBJECT (CLASS 01H)

Services:

Class services: Get Attribute Single
Instance services: Get Attribute Single

Table 3–3: Class attributes for identity object 01h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–4: Instance attributes for identity object 01h

Attr.	Access	Name	Type	Value	Description
1	Get	Vendor ID	UINT	03A0h (default)	GE Multilin
2	Get	Device type	UINT	000Ch (default)	Communications adapter
3	Get	Product code	UINT	0051h (default)	Anybus-C DeviceNet
4	Get	Revision	USINT	01h	Major fieldbus version
			USINT	20h	Minor fieldbus version
5	Get	Status	WORD	---	Device status (see table below)
6	Get	Serial number	UDINT	Serial number	Serial number
7	Get	Product name	SHORT_ STRING	D485 Modbus to DeviceNet Conv	Name of product
8	Get	Config consist value	UINT	N/A	---

The values for the status attribute (attribute 5) are shown below.

Bit(s)	Name
0	Module owned. A master has allocated the module.
1	Reserved
2	Configured
3 to 7	Reserved
8	Minor recoverable fault
9	Minor recoverable fault
10	Major recoverable fault
11	Major unrecoverable fault
12 to 15	Reserved

MESSAGE ROUTER (CLASS 02H)

Services:

Class services: Get Attribute Single

Table 3–5: Class attributes for message router object 02h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

DEVICENET OBJECT (CLASS 03H)

Services:

Class services: Get Attribute Single
Instance services: Get Attribute Single

Table 3–6: Class attributes for DeviceNet object 03h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0002h	Revision 2

Table 3–7: Instance attributes for DeviceNet object 03h

Attr.	Access	Name	Type	Value	Description
1	Get	MAC ID	USINT	N/A	Currently used MAC ID
2	Get	Baud rate	USINT	N/A	Currently used baud rate: 0 = 125 kbps, 1 = 250 kbps, and 2 = 500 kbps
3	Get	Allocation information	BYTE	N/A	Allocation choice byte
			USINT	N/A	Master MAC ID

ASSEMBLY OBJECT (CLASS 04H)

The assembly object binds all mapped I/O data. This data is used for I/O connections.

Services:

Class services: Get Attribute Single
Instance services: Get Attribute Single, Set Attribute Single

Table 3–8: Class attributes for assembly object 04h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0002h	Revision 2

Table 3–9: Instance attributes for assembly object 04h, instance 64h

Attr.	Access	Name	Type	Value	Description
3	Get	Data	Array of USINT	---	Data produced by the D485 to the master

Table 3–10: Instance attributes for assembly object 04h, instance 96h

Attr.	Access	Name	Type	Value	Description
3	Get	Data	Array of USINT	---	Data consumed by the D485 to the master

If the I/O input data size is set to 0, the above instances will not be initialized.

CONNECTION OBJECT (CLASS 05H)

Services:

Class services: Get Attribute Single
Instance services: Get Attribute Single, Set Attribute Single

Table 3–11: Class attributes for connection object 05h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

The instances for this object are defined as follows:

Instance 1 = Explicit messaging connection (Predefined in DeviceNet object)
Instance 2 = Polled connection / COS / Cyclic consuming connection
Instance 3 = Bit strobe connection
Instance 4 = COS / Cyclic producing connection
Instances 10 to 14 = Explicit connection (UCMM allocated)

Table 3–12: Attributes for instances 1 and 10 to 14 for connection object 05h

Attr.	Access	Name	Type	Value	Description
1	Get	State	USINT	N/A	0 = Non existent 1 = Configuring 3 = Established 4 = Timeout 5 = Deferred delete
2	Get	Instance type	USINT	0	Explicit messaging connection

Table 3–13: Attributes for instance 2 for connection object 05h

Attr.	Access	Name	Type	Value	Description
1	Get	State	USINT	N/A	0 = Non existent 1 = Configuring 3 = Established 4 = Timeout
2	Get	Instance type	USINT	0	I/O connection

Table 3–14: Attributes for instance 3 for connection object 05h

Attr.	Access	Name	Type	Value	Description
1	Get	State	USINT	N/A	0 = Non existent 1 = Configuring 3 = Established 4 = Timeout
2	Get	Instance type	USINT	0	I/O connection

Table 3–15: Attributes for instance 4 for connection object 05h

Attr.	Access	Name	Type	Value	Description
1	Get	State	USINT	N/A	0 = Non existent 1 = Configuring 3 = Established 4 = Timeout
2	Get	Instance type	USINT	0	I/O connection
3	Get	Transport class trigger	BYTE	N/A	Defines the behavior of the connection
4	Get	Produced connection ID	UINT	N/A	CAN ID for transmission
5	Get	Consumed connection ID	UINT	N/A	CAN ID for reception
6	Get	Initial communication characteristics	BYTE	0Fh (no ACK)	Produces over message group 1. Does not consume.
				01h (ACK)	Produces over message group 1. Consumes over message group 2.
7	Get	Produced connection size	UINT	N/A	Number of bytes transmitted across this connection
8	Get	Consumed connection size	UINT	0	Number of bytes received across this connection
9	Get/Set	Expected packet rate	UINT	0	Timing associated with this connection
12	Get	Watchdog timeout action	USINT	N/A	0 = Transition to the timed out state 1 = Auto delete 2 = Auto reset 3 = Deferred delete
13	Get	Produced connection path length	UINT	0006h	Number of bytes in the produced connection path attribute
14	Get	Produced connection path	EPATH	20 04 24 66 30 03h	Application object producing data on this connection
15	Get	Consumed connection path length	UINT	0004h	Number of bytes in the consumed connection path length attribute
16	Get	Consumed connection path	EPATH	20 2B 24 01h	Specifies the application object(s) that are to receive the data consumed by this connection object

ACKNOWLEDGE HANDLER OBJECT (CLASS 2BH)

Services:

Class services: Get Attribute Single
Instance services: Get Attribute Single, Set Attribute Single

Table 3–16: Class attributes for acknowledge handler object 2Bh

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1
2	Get	Max. instance	UINT	0001h	Maximum instance number

Table 3–17: Instance attributes for acknowledge handler object 2Bh

Attr.	Access	Name	Type	Value	Description
1	Get/Set	Acknowledge timer	UINT	20	Time (in ms) to wait for acknowledge before re-sending.
2	Get/Set	Retry limit	USINT	1	Number of ACK time-outs before retry limit reached event
3	Get/Set	Producing connection Instance	UINT	4	Connection instance – contains the path of the producing I/O application object which will be notified of acknowledge handler events
4	Get	Acknowledge list size	Byte	---	Maximum number of members in acknowledge list (0 = dynamic)
5	Get	Acknowledge list	Array of USINT	N/A	List of active connection instance which are receiving acknowledgements.
6	Get	Data with acknowledge path list size	Byte	---	Maximum number of members in data with acknowledge path list (0 = dynamic)
7	Get	Data with acknowledge path list	Array of USINT	N/A	List of connection instance/ consuming application object pairs.

I/O DATA INPUT MAPPING OBJECT (CLASS A0H)

This vendor-specific object provides I/O input data mapping information. The number of existing attributes depends on which attributes are initialized in the module through the EnerVista P485/D485 Setup software.

Services:

Class services: Get Attribute All
Instance services: Get Attribute Single, Set Attribute Single

Table 3–18: Class attributes for I/O data input mapping object A0h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–19: Instance attributes for I/O data input mapping object A0h, instance 01h

Attr.	Access	Name	Type	Value	Description
1	Get	Input 1	Array of USINT	---	Data that is read
2	Get	Input 2	Array of USINT	---	Data that is read
3	Get	Input 3	Array of USINT	---	Data that is read
4	Get	Input 4	Array of USINT	---	Data that is read
5	Get	Input 5	Array of USINT	---	Data that is read
6	Get	Input 6	Array of USINT	---	Data that is read

I/O DATA OUTPUT MAPPING OBJECT (CLASS A1H)

This vendor-specific object provides I/O output data mapping information. The number of existing attributes depends on which attributes are initialized in the module through the EnerVista P485/D485 Setup software.

Services:

Class services: Get Attribute All
Instance services: Get Attribute Single, Set Attribute Single

Table 3–20: Class attributes for I/O data output mapping object A1h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–21: Instance attributes for I/O data output mapping object A1h, instance 01h

Attr.	Access	Name	Type	Value	Description
1	Get/Set	Output 1	Array of USINT	---	Data that is read or written
2	Get/Set	Output 2	Array of USINT	---	Data that is read or written
3	Get/Set	Output 3	Array of USINT	---	Data that is read or written
4	Get/Set	Output 4	Array of USINT	---	Data that is read or written
5	Get/Set	Output 5	Array of USINT	---	Data that is read or written
6	Get/Set	Output 6	Array of USINT	---	Data that is read or written

DIAGNOSTIC OBJECT (CLASS AAH)

This vendor specific object provides diagnostic information from the module.

Services:

Class services: Get Attribute All
Instance services: Get Attribute Single

Table 3–22: Class attributes for diagnostic object AAh

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–23: Instance attributes for diagnostic object AAh

Attr.	Access	Name	Type	Value	Description
1	Get	Module serial number	UDINT	---	Serial number
4	Get	Module software version	UINT	---	DeviceNet interface software version
15	Get	Input I/O size [1]	UINT	---	Size of I/O input area (in bytes)
17	Get	Input total size	UINT	---	Total number of input bytes (I/O + explicit)
18	Get	Output I/O size [2]	UINT	---	Size of I/O output area (in bytes)
20	Get	Output total size	UINT	---	Total number of output bytes (I/O + explicit)

1. Corresponds to the "IO Size In" parameter in EnerVista P485/D485 Setup.
2. Corresponds to the "IO Size Out" parameter in EnerVista P485/D485 Setup.

PARAMETER DATA INPUT MAPPING OBJECT (CLASS B0H)

This vendor-specific object provides parameter input data mapping information. The number of existing attributes depends on which attributes are initialized in the module through the EnerVista P485/D485 Setup software.

Services:

Class services: Get Attribute All
Instance services: Get Attribute Single, Set Attribute Single

Table 3–24: Class attributes for parameter data input mapping object B0h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–25: Instance attributes for parameter data input mapping object, instance 01h

Attr.	Access	Name	Type	Value	Description
1	Get	Parameter input 1	Array of USINT	---	Data that is read
2	Get	Parameter input 2	Array of USINT	---	Data that is read
3	Get	Parameter input 3	Array of USINT	---	Data that is read
↓	↓	↓	↓	↓	↓
50	Get	Parameter input 50	Array of USINT	---	Data that is read

PARAMETER DATA OUTPUT MAPPING OBJECT (CLASS B1H)

This vendor-specific object provides parameter output data mapping information. The number of existing attributes depends on which attributes are initialized in the module through the EnerVista P485/D485 Setup software.

Services:

Class services: Get Attribute All
Instance services: Get Attribute Single, Set Attribute Single

Table 3–26: Class attributes for parameter data output mapping object B1h

Attr.	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Table 3–27: Instance attributes, parameter data output mapping object, instance 01h

Attr.	Access	Name	Type	Value	Description
1	Get/Set	Parameter output 1	Array of USINT	---	Data that is read or written
2	Get/Set	Parameter output 2	Array of USINT	---	Data that is read or written
3	Get/Set	Parameter output 3	Array of USINT	---	Data that is read or written
↓	↓	↓	↓	↓	↓
50	Get/Set	Parameter output 50	Array of USINT	---	Data that is read or written

D485 Modbus to DeviceNet Converter

Chapter 4: Software overview

Introduction

DESCRIPTION

EnerVista P485/D485 Setup is a PC-based configuration software used to describe the protocol and communication properties for a serial network. When the configuration is finished and the functionality is tested, it is possible to send memory allocation information to a printer using EnerVista P485/D485 Setup.

EnerVista P485/D485 Setup can also be used for troubleshooting and diagnostic of the D485 and the serial network during runtime.

SYSTEM REQUIREMENTS

The following hardware and software is required to use the EnerVista P485/D485 Setup software.

- Pentium 133 MHz or higher
- 10 MB of free space on the hard drive
- 8 MB RAM
- Windows 95/98/NT/2000/XP
- Internet Explorer 4.01 SP1 or higher

Installation procedure

DESCRIPTION

There are two different ways of installing EnerVista P485/D485 Setup; either via the GE EnerVista CD or from the GE Multilin website at http://www.GEmultilin.com.

INSTALLING FROM ENERVISTA CD

Run 'setup.exe' and follow the on screen instructions

INSTALLING FROM THE GE MULTILIN WEBSITE

Download the self-extracting EXE file from the GE Multilin website at http://www.GEmultilin.com.

Using the EnerVista P485/D485 Setup software

DESCRIPTION

When creating a new sub network configuration, EnerVista P485/D485 Setup provides a choice between starting out with a blank configuration, or using a predefined template (configuration wizard).

- **Configuration Wizard**: The wizard option automatically creates a configuration based on information about the sub-network (Modbus) devices; that is, the user simply has to "fill in the blanks".
- **Blank Configuration**: This option should be used when creating a new configuration when the configuration wizard does not fit the application or to modify an existing configuration for a new application. The following chapters will describe the configuration process in detail.

NOTE

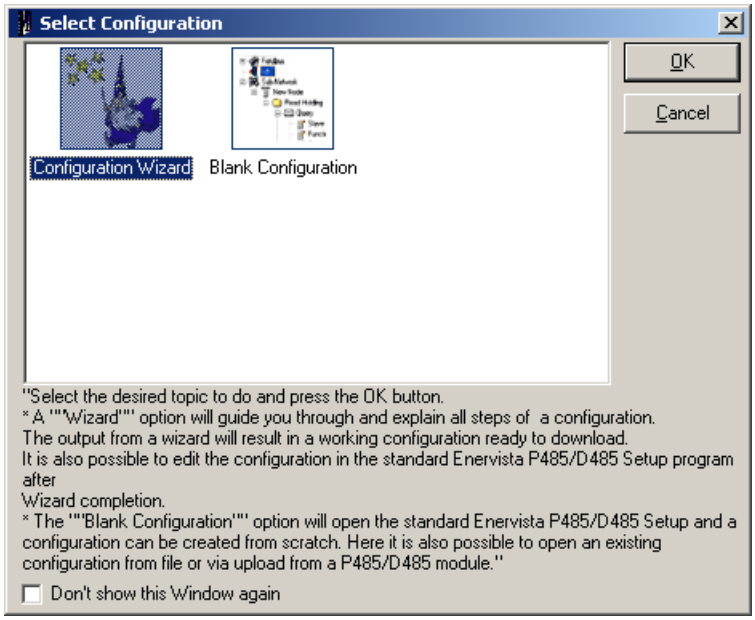
It is recommended to use the configuration wizard for its simplicity of use with GE relays and meters.

The online help system explains each configuration step in detail.

CONFIGURATION WIZARD

The purpose of the configuration wizard is to help you through the process of creating a project with a Modbus RTU sub-network. When the wizard is finished, it is possible to continue editing the project in the configuration tool.

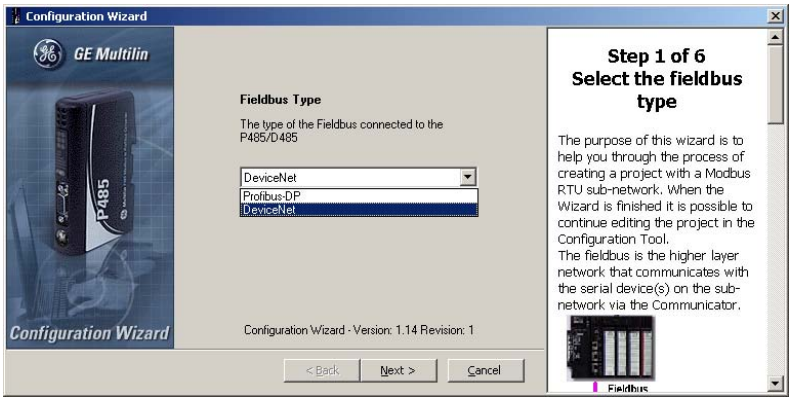
The EnerVista P485/D485 Setup software will open with following screen to select the configuration.



Select **Configuration Wizard** and click on **OK**.

SELECT FIELDBUS TYPE

The first step in the configuration wizard selects the fieldbus type. The fieldbus is the higher layer network that communicates with the serial device(s) on the Modbus sub-network via the D485 converter.



Select "DeviceNet" then click **Next** to continue. A typical DeviceNet network arrangement is shown below.

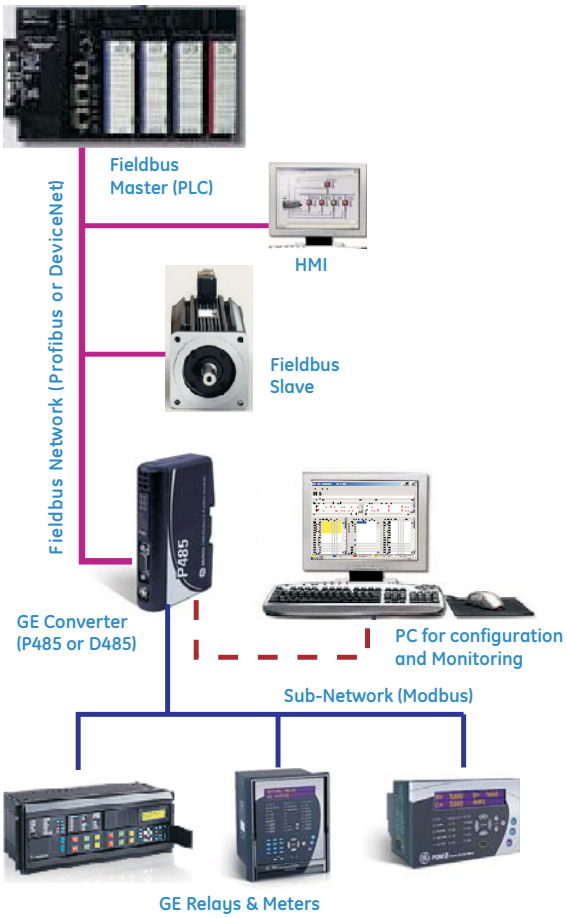


Figure 4-1: Typical network arrangement

NOTE

In the event the wizard cannot handle the specific Modbus command(s) required by the device, use the regular configuration tool or modify the commands produced by the wizard using the regular configuration tool.

SUB-NETWORK PROPERTIES

The second step in the configuration wizard selects the properties for the Modbus sub-network. The data flow for the sub-network is shown below.

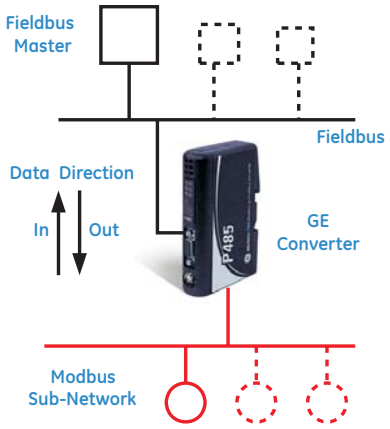


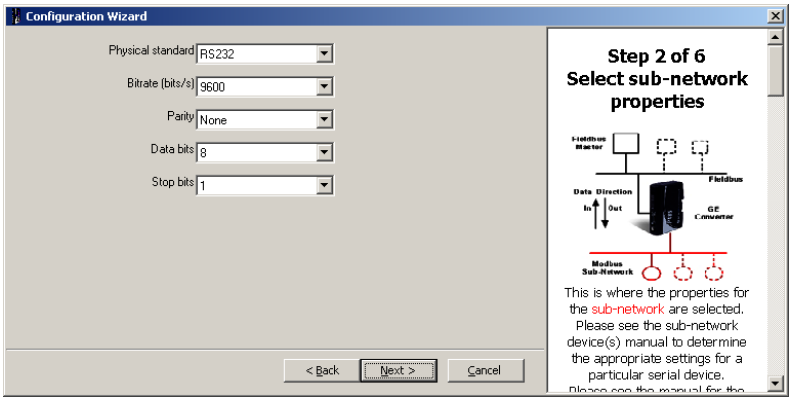
Figure 4-2: Sub-network data flow

Refer to the particular sub-network device manual(s) to determine the appropriate settings and communication options. If multiple devices are being installed on the same sub-network, they must be configurable for a common set of communication parameters.

NOTE

All numerical values are entered and shown in decimal unless otherwise specified.

The sub-network properties window is shown below.



- **Physical standard**: The physical standard can be either RS232, RS485, or RS422.

  RS232 is a point-to-point communication standard; that is, it is only possible to have one sub-network node (Modbus device) connected to the D485 converter when using RS232. RS232 supports a maximum cable length of 15 meters and is full duplex. It uses two signal lines (Rx and Tx) and the signal is measured relative to ground.

  RS485 is a common multi-drop communication standard. It is used with larger cable distances with one or several sub-network nodes (Modbus devices) connected. RS485 supports a maximum of 31 nodes, with half duplex and a total cable length up to 1200 meters. It uses two signal lines (A-line and B-line, twisted pair) with the signal being measured between the two lines.

  RS422 is a common multi-drop communication standard. It is used with larger cable distances with one or several sub-network nodes connected. RS422 supports a maximum of 31 nodes, with full duplex and a total cable length up to 1200 meters. It uses four signal lines (receive A1-B1 and transmit A2-B2, twisted pair) with the signals being measured between the two signal lines A and B.

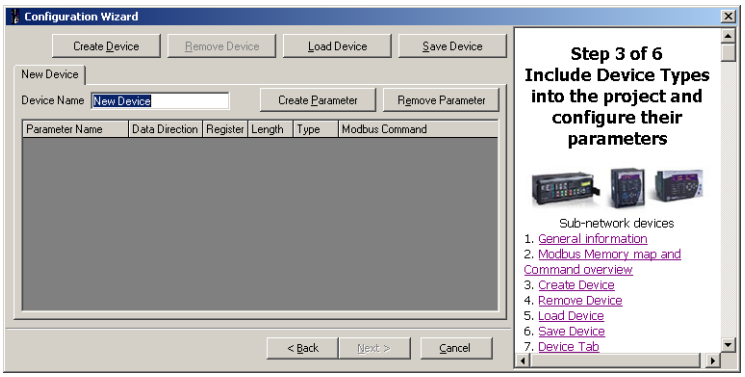
- **Bitrate (bits/s)**: This parameters refers to the speed of the sub-network. Speeds are 1200 to 57600 bps in predefined steps. The bitrate is also referred to as baud rate.

- **Parity**: The parity can be selected as "Odd", "Even", or "None". This is a simple error check method capable of detecting single bit communication errors on a serial network (i.e. the sub-network).
- **Data bits**: There can be "7" or "8" data bits. Generally, 8 data bits are used. This parameter determines how many bits per byte of user data that is transmitted on the sub-network, excluding start, stop and parity bits.
- **Stop bits**: There can be "1" or "2" stop bits. Determines the number of stop bits at the end of each byte sent on the sub-network.

DEVICE TYPES

The third step in the configuration wizard introduces device types into the project and configures their parameters. Every device must be unique. Predefined devices can be loaded from a file, and it is possible to connect devices to the sub-network at a later step in the wizard. Additional devices can be created by editing previously saved devices.

The device types window is shown below.



The Modbus address range, including bit areas and register areas (words), is shown below. The Modbus commands are also shown for the corresponding memory areas. Note that many device manuals ignore the leading digit of the address (i.e. 0, 1, 3 or 4); as such, the address 40001 is often referred to as 0001. The leading digit can be determined from the Modbus command specified.

Address			Command
0x	Output coil (bits)	00001 to 09999	#1 (decimal): Read coil status #15 (decimal): Force multiple cells
1x	Inputs status (bits)	10001 to 19999	#2 (decimal): Read input status
3x	Input status (word)	30001 to 39999	#4 (decimal): Read input registers
4x	Output (word)	40001 to 49999	#3 (decimal): Read holding registers #16 (decimal) Preset multiple registers

Users should consult the instruction manuals of the various network devices to determine the actual Modbus command code(s) implemented or required. This will determine the implied leading digit of the data address (i.e. 0, 1, 3 or 4).

In most GE Multilin relay and meter documentation, Modbus addresses are indicated in hexadecimal form. For the Modicon format used for the D485, convert the hex address to decimal, add 1, then append a prefix of 1, 2, 3, or 4, depending on the type of register. For example, to convert the input register hexadecimal address 0x0300, we have:

1. 0300h = 0768 decimal
2. 0768 decimal + 1 = 0769 decimal
3. change 0769 → 30769 (prefix "3" for input registers)

Therefore, a Modbus hex address of 0x0300 is 30769 in Modicon format.

NOTE

The wizard can accept memory addresses from 0 to 9999 (0x270F). For higher memory addresses, please use the protocol building blocks (refer to *Protocol building blocks* on page 3–5 for additional details).

The **Create Device** button creates a new empty device. A new Device tab will be created. The new device can be named in the **Device Name** text box. The **Remove Device** button removes the selected device.

The **Load Device** button opens the "open file" dialog box. Select a previously stored device to include it into the project. Device files (extension D01) for the most commonly used GE relays and meters are supplied with the EnerVista P485/D485 Setup software.

The **Save Device** button opens the "save device" dialog box. To create similar devices, click on **Save Device** to save a particular device parameter list, then click **Load Device** to recover a duplicate of the device. The duplicate device should be renamed and then modified as required. Devices can also be saved for a use at a later stage. All parameters and address settings are stored in the device file.

The Device tab shows the name of the device and the active node. The tab "in front" of the other tabs is the active one. The active device's parameters are shown in the parameter list below the tab list.

The **Device Name** is typically the technical name or designation of a device found on the devices name plate. Examples are "MM2", "469" and "PQMII". Do not confuse the device name with the node name, which is entered at a later stage. The node name is typically a name that is used to identify the device in your application. Examples are "Lube Pump 1", "Production feeder" and "Main transformer".

The **Create Parameter** button adds a new parameter to the parameter list. The loaded device from previously saved devices can be modified for a new parameter or change in the settings of the parameters. The **Remove Parameter** button removes the selected parameter from the parameter list. To select a parameter simply click the desired parameter in the list. Use the scroll bar to move the list up and down. Click the desired parameter and enter the desired **Parameter Name**. It is recommended that you enter a unique name here. Examples are "Phase A Current Ia", "Voltage Vab", and "VT ratio".

The **Data Direction** column shows if data is read from or written to the device. The D485 converter is the one who reads or writes. It is only possible to read input data; output data can be both read or written. Refer to Figure 4-2: Sub-network data flow on page 4–4 for details.

The **Register** column is where the Modbus register number for the for the parameter in the device is entered. Only register addresses can be entered here (the register address is the absolute address +1). Most device manuals contain the register address but some may provide an absolute address in hexadecimal format. In case absolute addresses are given, the address must be incremented by 1. If the address range covers multiple coils, inputs, or registers, only the start address is entered.

The **Length** column is where the total length of the parameter data is entered. The length is given in bits for the 0x and 1xxxx areas and in words for the 3xxxx and 4xxxx areas. If the parameter data for a device on the sub-network are linearly addressable, then consecutive parameters may be addressed using a single Modbus command from the D485. For example, five parameters each 2 words long can be addressed using a single Modbus command (#16 Preset Multiple Registers) with a total length of 10 (5×2). Reducing the number of transactions initiated by the D485 will optimize communications on the Modbus sub-network.

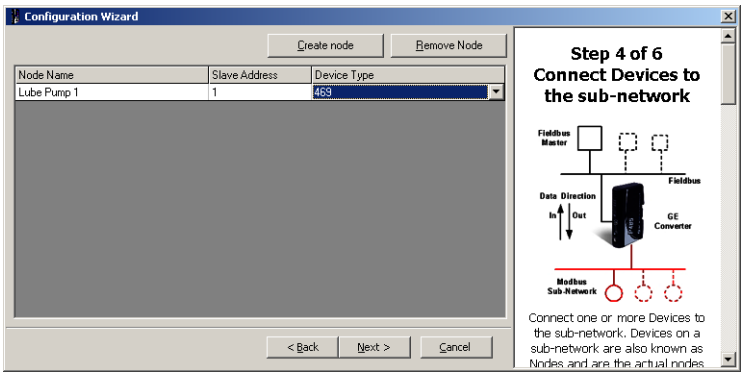
The **Type** column shows the type of data that is referenced for the respective parameter. Possible entries are bit(s) and word(s). This is automatically entered by the wizard based on the selected address and selected direction.

The **Modbus Command** column shows the Modbus command assigned by the wizard. The Modbus command is automatically selected by the wizard based on the selected address and direction.

CONNECTING DEVICES

The fourth step in the configuration wizard connects one or more devices to the Modbus sub-network. Devices on a sub-network are also known as nodes (Modbus slave devices) and are the actual nodes that will be physically connected to the Modbus sub-network. It is possible to connect devices of the same device type more than once. The created nodes will be listed to the left.

The Node window is shown below:



The **Create Node** button adds a new node (Modbus slave) to the sub-network. A new row will be added to the node list to the left. The **Remove Node** button to remove the selected node. Select a node in the node list by clicking on the desired node.

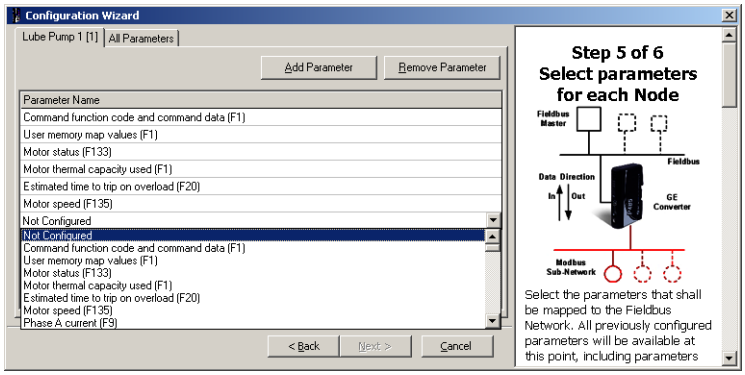
When a new node is created, the wizard assigns it a default name. Enter the desired node name in the Node Name column. The node name is typically a name that identifies the device in your application. Examples are “Lube pump1”, “Production Feeder” and “Main transformer “. Do not confuse the node name with the device name assigned at an earlier stage – the device name is typically the technical name or designation of a device found on the devices name plate (for example, “MM2”, “SR469” and “PQMII”).

Enter the Modbus slave address of the sub-network node in the **Slave Address** column. The wizard automatically assigns a default address which can be changed as needed. The node address must match the slave address setting of the device you are connecting. If you only connect one node, this address setting might be irrelevant, depending on the operation of the device.

The **Device Type** column is where previously configured devices are connected to the sub-network. If you click a row in the device column, a list will appear containing all previously configured device(s). Select the desired device from this list.

SELECTING PARAMETERS FOR EACH NODE

The fifth step in the configuration wizard selects the parameters that shall be mapped to the Fieldbus Network. All previously configured parameters will appear at this point, including parameters saved to a file. All previously configured nodes will appear in the horizontal Node tab list in the upper left of the configuration wizard. Select the All Parameters tab to view the complete list of parameters.



The Node tab in the foreground displays the active node. The number within brackets at the end of the node name is the node Modbus slave address (1 to 255) on the sub-network. Clicking a specific tab will display the parameters currently mapped to this node address.

For example, for “Lube Pump 1[1]”, the name of the node is “Lube Pump 1” and its slave address on the sub-network is 1.

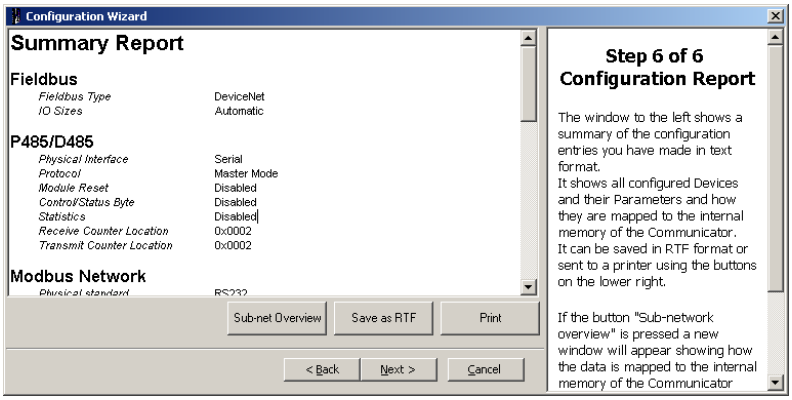
The **All Parameters** tab displays all parameters in the same list. This provides an overview of data transmitted on the sub-network. It is also possible to add or remove parameters on configured nodes in this list by using the **Add Parameter** or **Remove Parameter** buttons.

The **Add Parameter** button adds a new parameter to the selected node. The **Remove Parameter** button remove the selected parameter from the selected node.

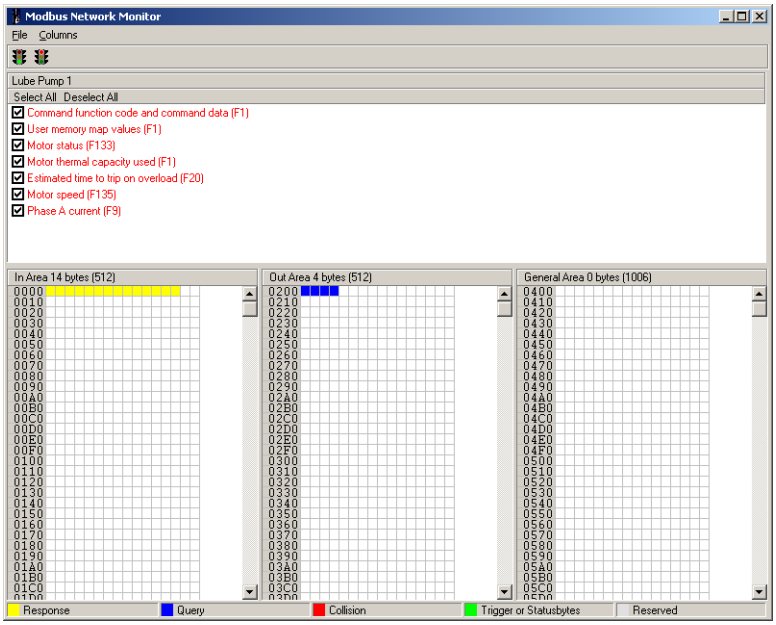
The **Parameter Name** column displays the user-assigned parameter name. When a new parameter is inserted, it is named by the software as “Not Configured”. A list of available parameters will appear when the you click the row. Select the desired parameter by clicking on it in the list.

CONFIGURATION REPORT

The sixth and final step in the configuration wizard displays a summary of the configuration entries. This includes all configured devices, their parameters, and how they are mapped to the internal memory of the D485. This report can be saved in rich text (RTF) format or sent to a printer.



If the **Sub-net Overview** button is pressed, a new window will appears that graphically displays how the data is mapped to the internal memory of the D485.



Configuration main window

DESCRIPTION

The main window is shown below. It is composed of the navigation window, parameter window, information window, and configuration line indicator.

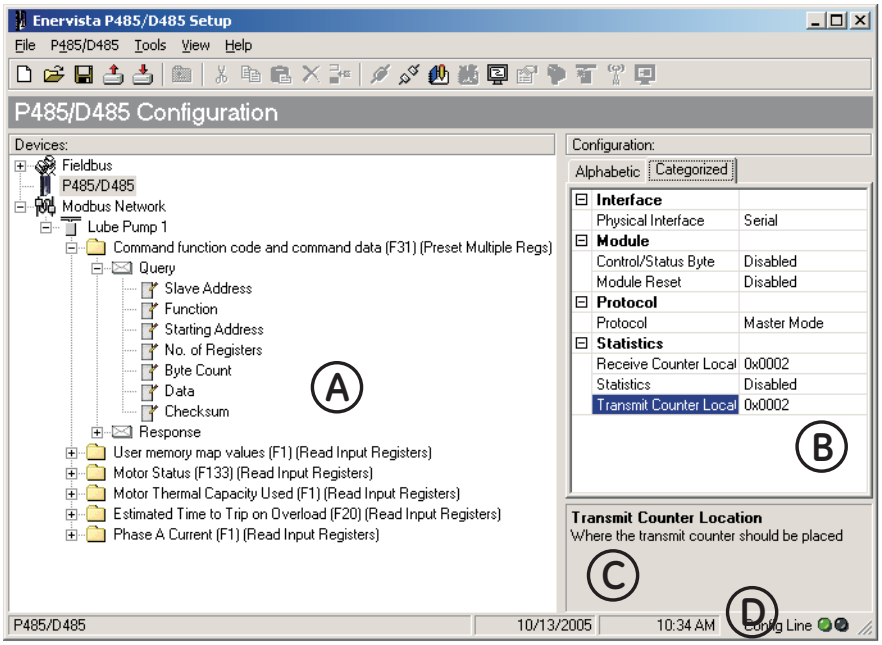
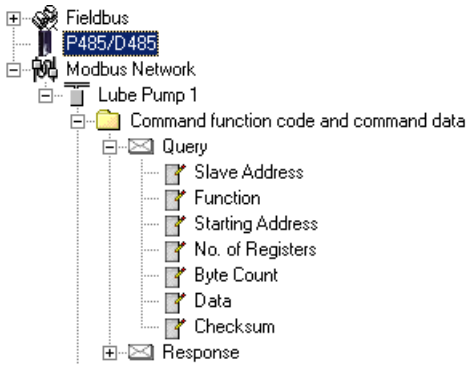


Figure 4-3: Configurator main window

NAVIGATION WINDOW

The navigation window in EnerVista P485/D485 Setup is the main tool for selecting the different levels of the configuration. There are three main levels in the navigation window, namely fieldbus, D485, and Modbus network.

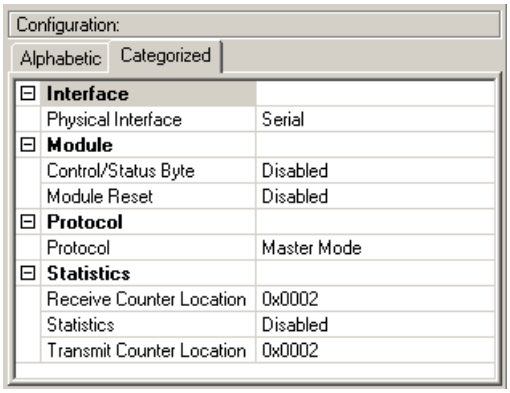


Menu entries preceded by a plus symbol (+) contain more configuration parameters or sub-menus. To gain access to these parameters, the entry must be expanded by clicking the '+' symbol.

By right-clicking entries in this window, a popup menu with functions related to this entry will appear. The options in this popup menu is often also available in the menu bar.

PARAMETER WINDOW

The parameters available in this window is different depending on what is selected in the navigation window. It consists of a grid with parameter names and, on the same row, a field for editing.



The parameters can be displayed in two modes: alphabetic and categorized. Parameter values are entered either using selection box or by entering a value. Values can be entered either in decimal form (for example, 35) or in hexadecimal form (for example, 0x1A).

If a value is entered in decimal format, it will be converted automatically to the equivalent hexadecimal value.

INFORMATION WINDOW

In the right bottom corner of EnerVista P485/D485 Setup, below the parameter window, lies the information window. It contains descriptions of currently marked parameter instances.

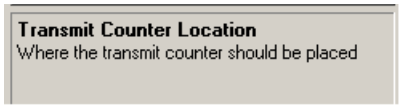


Figure 4-4: Information window

CONFIGURATION LINE INDICATOR

In the lower-right corner of the main window, two lights indicate if a connection has been established between the PC running EnerVista P485/D485 Setup and D485. A green light indicates that the connection is OK, and a red light indicates no connection.

OPTIONS WINDOW

In the main window under tools, select options.

Table 4–1: Options window functions

Function	Description
Warning on delete	When something is to be deleted, a warning window will appear.
Warning on unsaved data	When closing EnerVista P485/D485 Setup with unsaved data, a warning window will appear.
Show Wizard when “New” menu is selected	Each time a new configuration is to be made, the Wizard window will appear.
Language next time the program is launched	Select which language the program should use the next time the program is launched. Presently, only English is supported.
Size of log buffer	Set the size of the log buffer (0 to 512 bytes).
Firmware download	Download the firmware to the D485. Use with caution.
Factory restore	Restores the software on the D485 carrier board, to it´s original state.
Block configuration	Use with caution. When this button is pressed, the configuration will not be accessible and a new configuration has to be downloaded to the module.
Create error log	Creates an error log file.

Fieldbus configuration

DESCRIPTION

During start-up, the fieldbus interface is initialized to fit the configuration created in the EnerVista P485/D485 Setup software. Since EnerVista P485/D485 Setup supports both the P485 and D485 converters, the user must verify that the 'Fieldbus' parameter indicates the correct converter. Additionally, it is possible for advanced users to customize the network interface inside the converter to meet specific application demands (see *Advanced fieldbus configuration* on page 8–5 for details).

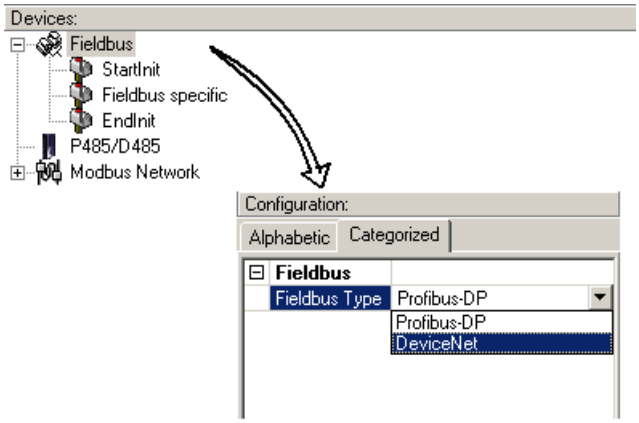


Figure 4-5: Fieldbus configuration

P485/D485 configuration

PARAMETER WINDOW

By selecting 'P485/D485 ' in the Navigation window, basic configuration options for the sub-net will appear in the Parameter window.

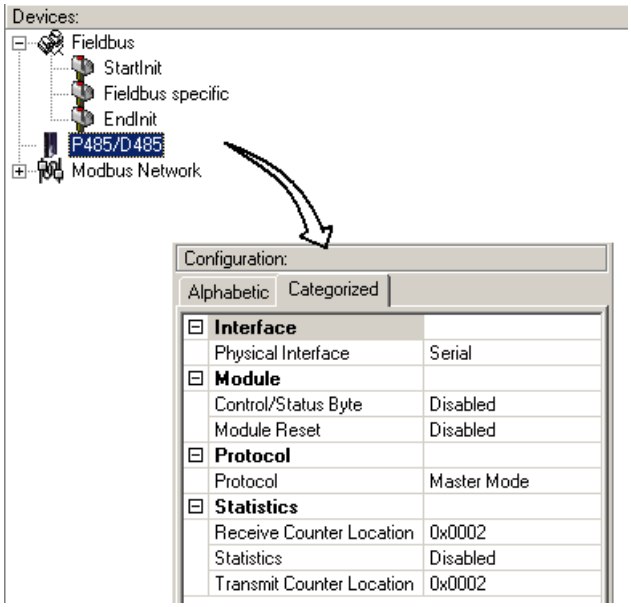


Figure 4-6: P485/D485 configuration

- **Physical interface**: Currently, the D485 supports only a serial interface. The communication settings for the selected interface are available under 'Modbus Network (see *Serial interface settings* on page 4–13 for details).
- **Control/status byte**: This parameter is used to enable/disable the control/status registers (see *Control and status registers* on page 8–1 for details).
  - Enable: enable control/status registers. The “Data Valid” parameter (bit 13 in the control register) must be set by the fieldbus control system to start the sub network communication.
  - Disable: Disables control/status registers.
  - Enable but no start up lock: The control/status registers are enabled, but the fieldbus control system is not required to set the “Data Valid” parameter (bit 13 in the control register).
- **Module Reset**: This parameter defines how the module should behave in the event of a fatal error. If Enabled, the module will reset and restart on a fatal error event, and no error will be indicated to the user. If Disabled, the module will halt and indicate an error.
- **Protocol**: The D485 supports Modbus RTU master mode.
- **Statistics**: If enabled, the receive counter location indicates the number of valid messages received from the subnet. If enabled, the transmit counter location indicates the number of messages sent to the sub network. This function is used primarily for debugging purposes.

Modbus network configuration

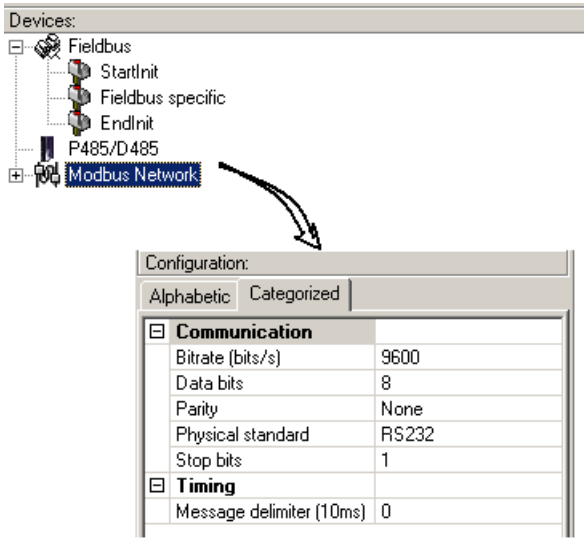
OVERVIEW

When controlling a Modbus sub-network with the D485 it is important to understand functions during starting up. If the D485 starts scanning nodes on the sub-network, before data is received from the fieldbus control system (fieldbus master), values of '00' may be transmitted to the nodes before data is updated the first time from the fieldbus.

See *Input/output data during startup* on page 8–4 for information on how to block transactions until valid data is received.

SERIAL INTERFACE SETTINGS

To be able to communicate on the Modbus network, various communication settings needs to be configured. To gain access to these settings, select 'Modbus Network' in the Navigation window.



Parameter	Description	Range
Bit rate	Selects the bit rate.	1200 to 57600
Data bits	Selects the number of data bits.	7, 8
Parity	Selects the parity.	None, Odd, Even
Physical standard	Selects the physical standard. This setting activates the corresponding signals on the subnet connector.	RS232, RS422, RS485
Start bits	Only one start bit is supported.	1
Stop bits	Either one or two stop bits can be selected.	1, 2

D485 Modbus to DeviceNet Converter

Chapter 5: Communication model

Introduction

DESCRIPTION

In master mode, the D485 is configured to run as a master on the Modbus sub-network, using a scan-list for communication with the Modbus slave devices. The scan-list is created using EnerVista P485/D485 Setup and can consist of multiple nodes with multiple transactions.

Communications between the D485 and the sub-net nodes (Modbus slaves) is based on transactions with a query/response architecture. The D485 sends out a query on the Modbus sub-network, and the addressed node is expected to send a response to this query. Slave nodes are not allowed to respond without first receiving a query.

An exception to this is broadcaster functionality. Most protocols offer some way of accessing all network nodes. In the D485, this is called a 'broadcaster'. The broadcaster can transmit messages to all nodes on the sub-network without expecting a response.

In Modbus, it is possible to broadcast a message to all nodes by sending a message to node address 0. The Modbus slaves will receive the message, but not Respond to it.

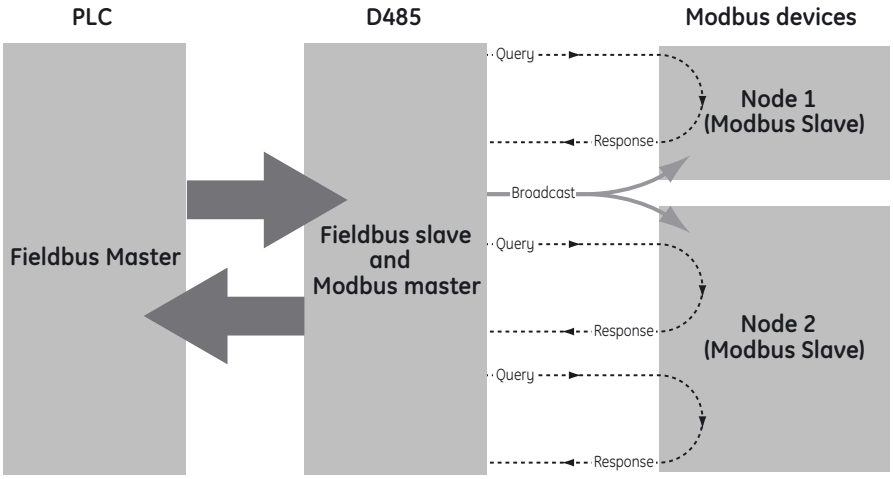


Figure 5-1: Master mode overview

The D485 uses pre-configured Modbus RTU commands, acting as a Modbus RTU master. With Modbus RTU, each transaction is substituted with a pre-defined command that can be selected from a list of available commands.

It is still possible, though, to define custom message frames by creating a transaction instead of selecting a pre-defined command. A command is actually a transaction that has been defined in advance and stored in a list.

SCAN LIST

Once the configuration has been downloaded to the D485, the D485 firmware searches the scan-list, using the defined transactions for communication with the slave-devices.

Each node in the scan-list represents a slave device on the Modbus network. In EnerVista P485/D485 Setup, each node is given a specific name and assigned an address in standard Modbus RTU commands. The address must match the internal setting on the slave device.

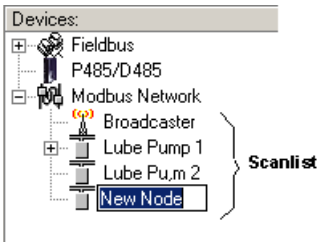


Figure 5-2: D485 scan list

Basic settings

NETWORK SETTINGS

Select 'Modbus Network' in the Navigation window to gain access to basic settings in the Parameter window.

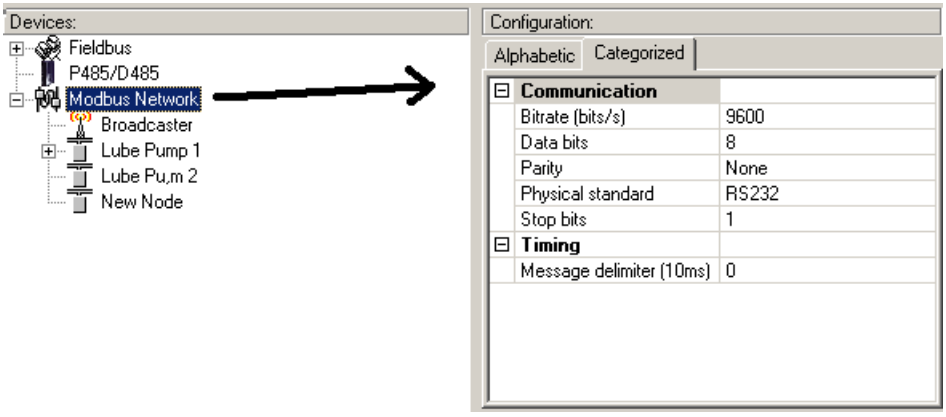


Figure 5-3: Parameter window

COMMUNICATION

Refer to *Serial interface settings* on page 4–13 for details.

MESSAGE DELIMITER

The message delimiter value is the minimum time in steps of 10 ms separating the messages. According to the Modbus specification, the message delimiter has a default setting of 3.5 characters.If this value is set to "0", the D485 will use the Modbus standard 3.5 character message delimiter. The time in milliseconds is then dependent on the selected baud rate, but this is all handled by the D485.

NOTE

Due to its impact on subnet functionality, use caution when changing this parameter.

Nodes

DESCRIPTION

A node in the EnerVista P485/D485 Setup software represents a device on the Modbus sub-network. In it's simplest form, a Node contains of a single transaction, that consists of a Query and a Response.

NODE PARAMETERS

To gain access to these parameters, select the desired node in the navigation window.

- **Slave address**: This setting shall be set to match the Modbus address setting of the destination device.
- **Name**: Node Name. This name will appear in the navigation window.

MODBUS NETWORK MENU

Right-click "Modbus Network" in the Navigation window to gain access to these functions.

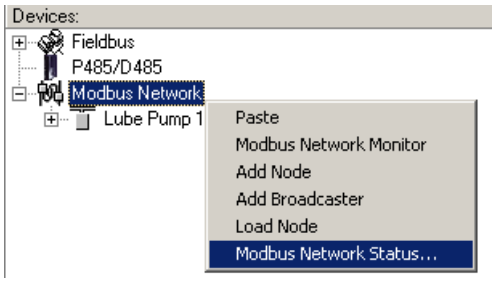


Figure 5-4: Modbus network menu

- **Paste**: Paste a node from the clipboard.
- **Modbus Network Monitor**: Launches the Modbus network monitor. Refer to *Modbus network monitor* on page 7–1 for details.
- **Add Node**: Adds a node to the scanlist.
- **Add Broadcaster**: Adds a broadcaster node to the scanlist.
- **Load Node**: Loads a node previously saved using "Save Node" from the Node menu (see details below).
- **Modbus Network Status**: Displays status/diagnostic information about the Modbus network.

NODE MENU

Right-click on a node in the Navigation window to gain access to these functions.

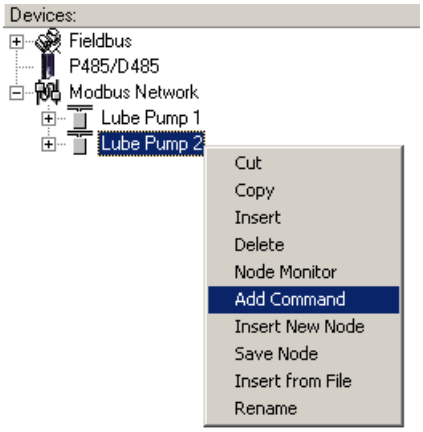


Figure 5-5: Node menu

- **Cut**: Cuts a node to the clipboard.
- **Copy**: Copies a node to the clipboard.

- **Insert**: Insert a node from the clipboard.
- **Delete**: Deletes a node and its configuration from the scan-list.
- **Monitor**: Activates the node monitor.
- **Add command**: Adds a pre-defined protocol specific command to the scan-list. The list of commands are supplied with the D485 and cannot be changed.
- **Insert new node**: Inserts a new node above the currently selected node.
- **Save node**: Saves the selected node.
- **Insert from file**: Inserts a previously saved node above the currently selected node.

QUERY PARAMETERS

To gain access to these parameters, select a Query in the Navigation window.

- **Minimum time between broadcasts (10 ms)**: The value entered here is only valid if a broadcast command is specified in the scan-list and the value specifies how long the D485 should wait after the broadcast was sent until the next command in the scan-list will be sent. This time should be selected such that all slave-devices connected to the D485 have time to finish the handling of the broadcast. The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms.
- **Offline options for fieldbus**: This parameter defines the behavior of the D485 in case the DeviceNet network goes off-line and the selection affects the data that is sent out the Modbus network
  - Clear: All data destined for the slave devices is cleared (set to 0).
  - Freeze: All data destined for the slave device is frozen.
  - NoScanning: The updating of the Modbus network is stopped.
- **Offline options for Modbus network**: This parameter defines the behavior of the D485 in case the Modbus network goes offline and the selection affects the data that is reported to the fieldbus master.
  - Clear: All data destined for the fieldbus master is cleared (set to 0).
  - Freeze: All data destined to fieldbus is frozen.

Offline options for Modbus networks are configured separately for each command.

- **Reconnect time (10 ms)**: This parameter specifies how long the D485 should wait before trying to re-connect a disconnected node. A node gets disconnected if the max number of retries is reached. The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms.
- **Retries**: This parameter specifies how many times a time-out can occur in sequence before the slave is disconnected.
- **Timeout time (10 ms)**: This parameter specifies the time the D485 waits for a response from the slave-device. If this time is exceeded the D485 re-sends the command until the “retries” parameter value is reached.

  The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms.
- **Trigger byte address**: This parameter specifies location in the internal memory buffer where the trigger byte is located. In D485 a trigger byte is implemented to support non-cyclic data that means that the DeviceNet master has the ability to notify the D485 when it should send a specific command to a slave.

  To use this functionality correctly the DeviceNet master should update the data area associated with the trigger byte, and then update the trigger byte. The trigger byte should be incremented by one for activation.

  This parameter has no affect unless the “Update mode” parameter is set to “Change of state on trigger”.

- **Update mode**: This parameter is used to specify when the command should be sent to the slave. The following modes are possible:
  - Cyclically: The command is sent to the slave at the time interval specified in the "Update time" parameter.
  - On data change: The command is sent to the slave when the data area connected to this command changes.
  - Single shot: The command is sent to the slave once at start-up.
  - Change of state on trigger: The command is sent to the slave when the trigger byte value is changed.
- **Update time (10 ms)**: This parameter specifies with what frequency this command will be sent. The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms.

RESPONSE PARAMETERS

To gain access to these parameters, select a Response in the Navigation window.

- **Trigger byte**: This parameter disables and enables the trigger functionality for the response. If the "trigger byte" is enabled then the D485 will increase the byte at the "trigger byte address" by one when the D485 receives new data from the Modbus network. This will notify the DeviceNet master of updated data.
- **Trigger byte address**: This parameter is used to specify the address in the internal memory buffer where the trigger byte is located. Valid settings range from 0x0002 to 0x00F3.

D485 Modbus to DeviceNet Converter

Chapter 6: Frame and command editors

Frame editor

DESCRIPTION

The frame editor makes it easier to add specific custom commands. The same parameters are available in both the frame editor and the parameter window, but in the frame editor presents the message frames in a more visual manner than the navigation / parameter window.

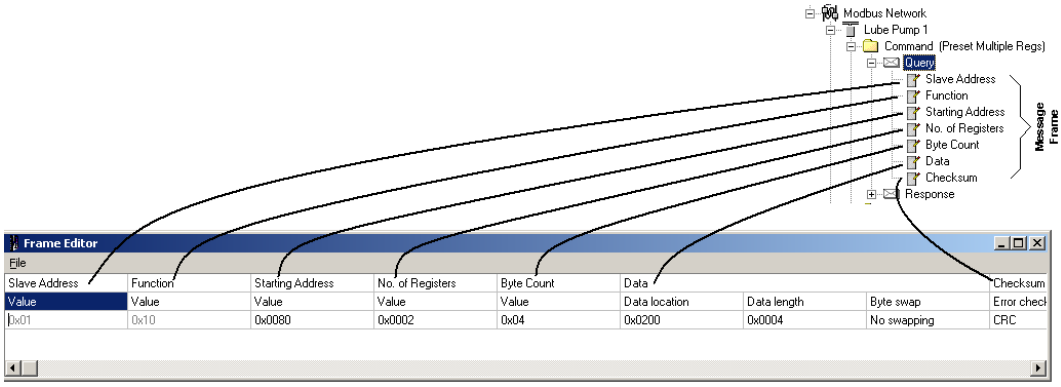
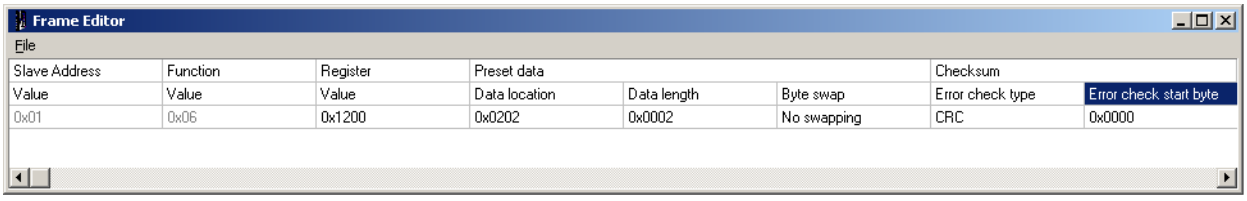


Figure 6-1: Frame editor window

EXAMPLE

Consider the following frame. The first byte holds the slave address (0x01) followed by the function code (0x06). The next word is the register address of the device where data is to be written (0x1200). This is a query command – the data is to be sent to the slave device and therefore is to be fetched from the OUT area starting at 0x0202. The next word indicates the data size (in bytes) to be written (in this case, 0x0002).



This command will allocate two bytes of output data in the OUT area and no swapping will occur. The data is followed by a two-byte CRC error check field and the CRC calculation starts with the first byte in the frame (0x0000).

The same steps are required for the response frame. If the response holds data, it should be allocated in the input area that starts at address 0x002. To apply the changes, select **File > Apply Changes**. To exit without saving, select **File > Exit**.

Command editor

GENERAL

The command editor makes it possible to add custom commands to the D485.

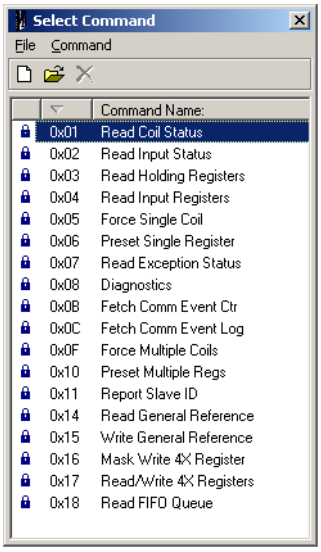


Figure 6-2: Select command window

To open the command editor, right click a node and select 'Add Command'. A list of predefined commands will appear.

To add a new command to the command list, select 'Add Command' in the 'Command' menu. To edit a previously defined command, highlight the command in the command list, and select 'Edit Command' in the 'Command' menu.The following window pops up upon selecting 'Edit Command' or 'Add Command'.

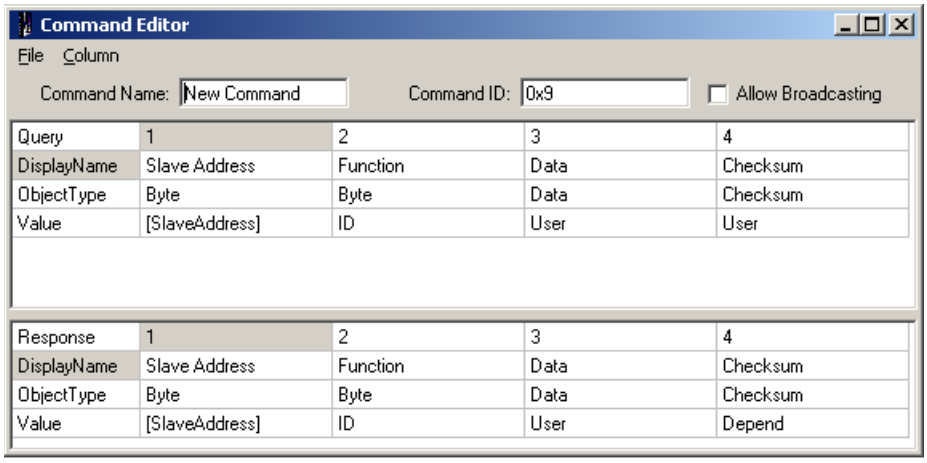


Figure 6-3: Command editor

SPECIFYING A NEW COMMAND

Select 'Add Command' as described earlier.

This example is taken from a Modbus RTU implementation, which means that the frame will always consist of one byte for slave address, one byte for function code and two bytes for CRC. Furthermore, each command always consists of a query and a response.

The Modbus RTU specific frame objects are already in place and a data object is inserted between the function code and the CRC. These objects cannot be moved or deleted, however it is possible to add objects between the function code and the CRC.

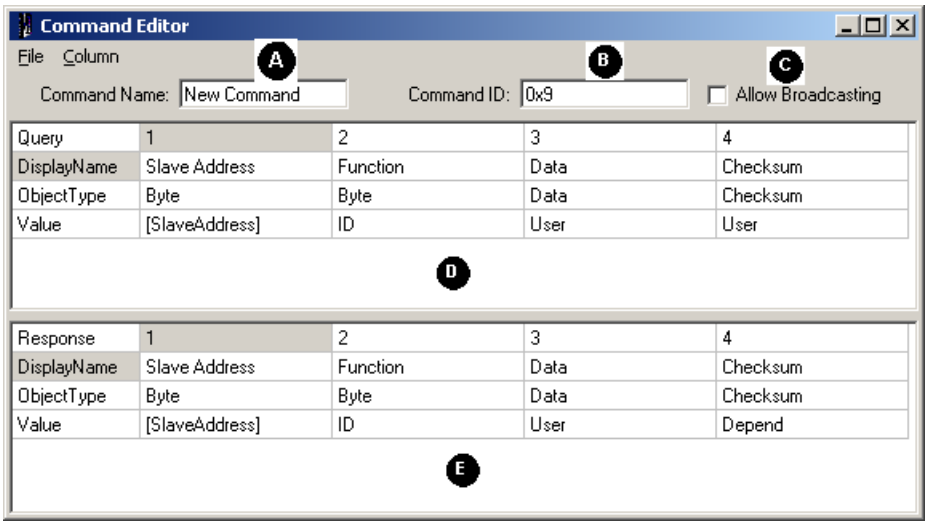


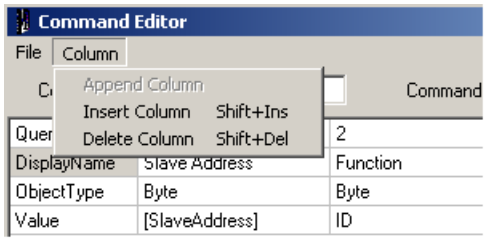
Figure 6-4: Specifying a new command

First, enter a name for the command in the Command Name field (A) and an identifier in the Command ID field (B). If the command is allowed to be broadcast on the sub-network, check the **Allow Broadcasting** check box (C).

The Query (D) field has the following characteristics:

Query	**Column**			
	1: Slave Address.	**2: Modbus Function Code**	**3: See below**	**4: Error Check field.**
DisplayName	Slave Address	Function	Data	Checksum
	Protocol specific; cannot be altered.		(See below)	
Object Type	Byte	Byte	Data	Checksum
	Modbus defines this object as a byte.		(See below)	
Value	[SlaveAddress]	ID	User	User
	Linked to the actual 'Slave Address' parameter in the parameter window.	This value is linked to the Command ID field.	(See below)	Linked to "User". Determined by user at configuration by selecting the Error Check object in the parameter window.

It is not possible to alter the contents of columns 1, 2 and 4, as these are pre-defined commands. However, on column 3 there are two possible actions: Insert Column and Delete Column. These actions are available in the Columns menu.



Column 3 in the Command Editor is where objects can be added for custom commands. Supported objects are Byte, Word, DWord, Data and Error Check. In this Modbus example it makes no sense to add an Error Check object since it is already incorporated in the standard frame but all other objects could be added in any way.

The “response” field (E) is defined much in the same way as the “query”, with the difference that a “response” can depend on what is entered in the “query”

Query	**Column**			
	1: Slave Address.	**2: Modbus Function Code**	**3: See below**	**4: Error Check field.**
DisplayName	Slave Address	Function	Data	Checksum
	See Query	See Query	See Query	See Query
Object Type	Byte	Byte	Data	Checksum
	See Query	See Query	See Query	See Query
Value	[SlaveAddress]	ID	User	Depend
	See Query	See Query	See Query	Object has same setting as the corresponding Query object. It also will appear as non-editable in the parameter window (see below)

If ‘Depend’ is selected then this object in the “response” will get the same setting as the corresponding object in the “query”, furthermore the object will appear as non-editable in the parameter window (see below).

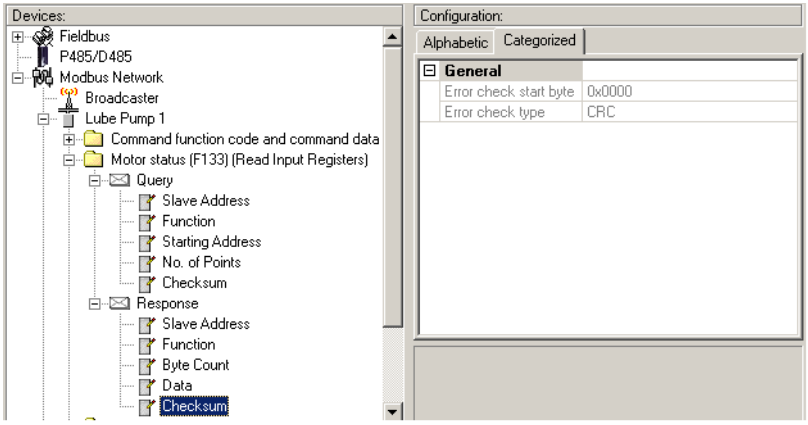


Figure 6-5: Main window

D485 Modbus to DeviceNet Converter

Chapter 7: Modbus network and node monitors

Modbus network monitor

GENERAL

The Modbus network monitor is intended to simplify configuration and troubleshooting of the Modbus network. It's main function is to display the data allocated for Modbus network communication and detect if any area has been allocated twice; that is, if a collision has occurred.

All configured nodes, together with the commands, are listed in the middle of the screen (B). Selecting and deselecting commands makes it possible to view any combination of allocated data.

NOTE

The Modbus network monitor has a negative influence on the overall performance of the D485. Therefore the monitor functionality should be used with care.

OPERATION

The Modbus network monitor window is shown below.

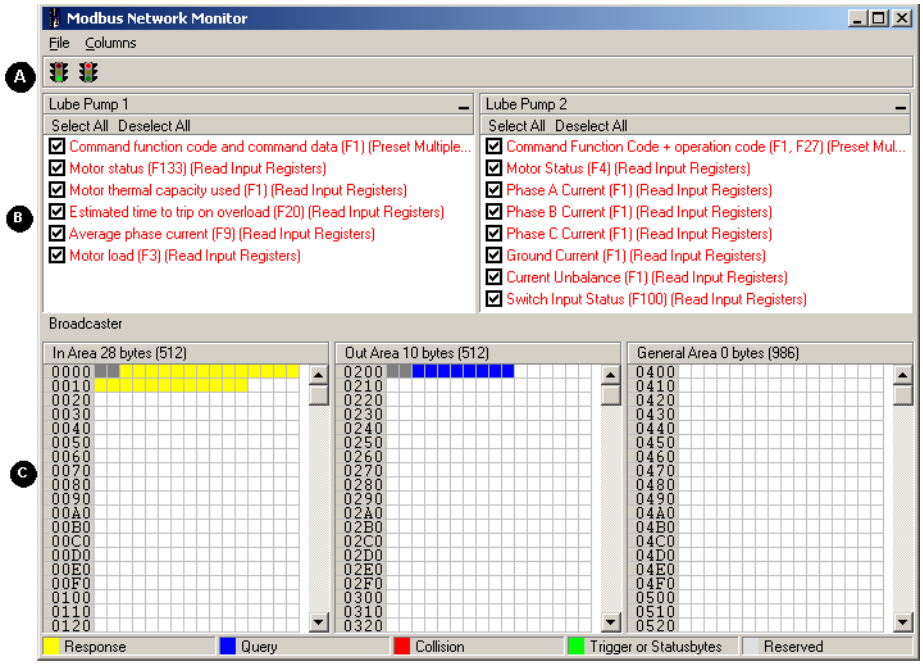


Figure 7-1: Modbus network monitor

A: Start/stop sub-network scanning

These icons are used to start / stop the scanning of the Modbus network. To stop the scanning, click on the red light. To start scanning again, simply click on the green light.

B: Nodes/transactions

To view data blocks linked to a single command, select the command and the data will appear in the monitor area, see below. (C)

C: Monitor Area: input / output / general data areas

These areas display the data allocated in the input, output and general data areas. This information is color coded as follows:

- **White**: No data allocated.
- **Yellow**: Data allocated by a response/consume transaction.
- **Blue**: Data allocated by a query/produce transaction.
- **Red**: Collision. This area has been allocated more than once.
- **Grey**: Data allocated by the control/status registers.

Node monitor

GENERAL

The node monitor functionality provides an aid when setting up the communication with the slave-devices on the Modbus network.

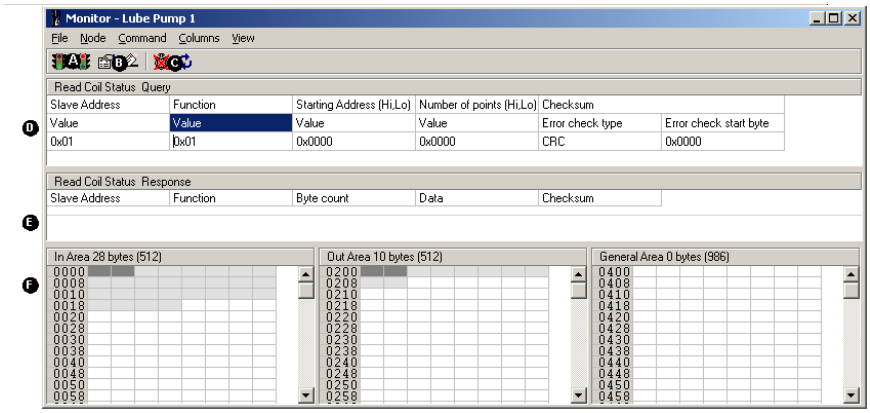
It offers an easy way of testing a specific command on a node, and monitor the result. It also provides an overview of the memory used by the node.

NOTE

Using the node monitor has a negative influence on the overall performance of the D485. Therefore the monitor functionality should be used with care.

OPERATION

The node monitor window is shown below.



A: Start/stop node communication

These icons are used to start or stop a node. Stopping is done by clicking the red light and could be seen as a temporary removal of the node, i.e no data will be sent to the node but it is still available. To start the node again, simply click on the green light.

This is a powerful feature when there is a problem with a particular node; the other nodes can be disconnected, helping to isolate the problem.

NOTE

If the control and status registers are enabled, the subnet cannot be started or stopped without being activated from the fieldbus.

B: Select/send command

Select the command to monitor using the 'Select' icon, and click 'Send' to send the command.

C: Data update ON/OFF

These icons are used to turn the monitor functionality ON or OFF (see 'Monitor Area' below).

D: Command area

This area displays the currently selected command.

E: Response area

This area displays the response of a previously sent command.

F: Monitor area

This area provides an overview of the data sent/received from the node. Areas in dark grey are reserved for the status/control registers.

Areas displayed in light grey are data objects used by the node. If data updating is enabled (see sub-section C above) the contents of these areas are also displayed in hex.

D485 Modbus to DeviceNet Converter

Chapter 8: Advanced functions

Control and status registers

DESCRIPTION

The control/status registers forms an interface for exchanging information between the fieldbus control system and the D485.

The main purpose of these registers is to report Modbus network related problems to the fieldbus control system. This interface is also used to ensure that only valid data is going out on the sub-network and that valid data is reported back to the fieldbus control system. See *Input/output data during startup* on page 8–4 for details.

Using these registers, it is also possible for the fieldbus control system to instruct the D485 to enable/disable specified nodes.

By default, these registers are located in the internal memory buffer at 0x000 to 0x001 (status register) and 0x200 to 0x201 (control register). However, they can be disabled using EnerVista P485/D485 Setup – refer to *Modbus network configuration* on page 4–13 for details. Disabling these registers will release the two reserved bytes in the internal memory buffer, however, the status and control functionality will not be available.

The handshaking procedure described on page 8–3 must be followed for all changes to these registers

CONTROL REGISTER (DEVICENET CONTROL SYSTEM TO D485)

Bytes 0 and 1 of the control register are shown below.

Byte 0 (Offset 0x200)								Byte 1 (Offset 0x201)							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	Control Code					Data							

Bits	Name	Description
15	Handshake Confirmation Bit (CR_HS_CONFIRM)	When the DeviceNet control system has read the new information from the status register, it should set this bit to the same value as bit 15 in the status register
14	Handshake Toggle Bit (CR_HS_SEND)	The fieldbus (DeviceNet) control system should toggle this bit when new information has been written in the control register.
13	Data Valid (CR_DV)	This bit is used to indicate to the D485 if the data in the output data area is valid or not. The bit shall be set by the fieldbus control system when new data has been written (1 indicates data is valid; 0 indicates that data is NOT valid)
12 to 8	Control Code (CR_EC)	See table below.
7 to 0	Data (CR_ED)	See table below.

CONTROL CODES

The following control codes are recognized by the D485 and can be used by the fieldbus control system.

Code	Name	Description
0x10	DISABLE_NODE	Slave address of the node to disable. This instructs the D485 to disable a specific node from the sub network communication
0x11	ENABLE_NODE	Slave address of the node number to enable. This instructs the D485 to enable a specific node to be active in the sub network communication
0x12	ENABLE_NODES	Number of nodes to enable. This instructs the D485 to enable a number of nodes from a complete configuration

STATUS REGISTER (D485 TO FIELDBUS CONTROL SYSTEM)

The status codes below are handled by the D485 and reported to the fieldbus control system using the status code and data bits in the status register. The meaning of these bits are different depending on the used communication model.

Byte 0 (Offset 0x200)								Byte 1 (Offset 0x201)							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	Status Code					Data							

Bits	Name	Description
15	Handshake Toggle Bit (SR_HS_SEND)	The D485 toggles this bit when new information is available in the status register.
14	Handshake Confirmation Bit (SR_HS_CONFIRM)	When the D485 has read the new information from the control register, it sets this bit to the same value as bit 14 in the control register
13	Data Valid (SR_DV)	Indicates to the fieldbus control system if the data in the input data area is valid or not. The bit is set by the D485 when new data has been written (1 indicates data is valid; 0 indicates that data is NOT valid).
12 to 8	Status Code (SR_EC)	Status code, see table below.
7 to 0	Data (SR_ED)	Status user data, see table below.

STATUS CODES

The status codes are described in the following table.

Code	Error	Description
0x00	Re-transmission	Number of re-transmissions. Reports the total number of re-transmissions on the subnetwork
0x01	Single node missing	Slave address of the missing node. Reports if a node is missing
0x02	Multiple nodes missing	Number of missing nodes. Reports if multiple nodes are missing
0x03	Overrun	Slave address of the node that sent too much data. Reports if more data than expected was received from a node
0x04	Other error	Slave address. Reports unidentified node
0x1F	No error	Normal Condition

HANDSHAKING PROCEDURE

The handshake bits are used to indicate any changes in the status and control registers. The procedure below must be followed for all changes to these registers with the exception of the handshake bits themselves (bits 14 and 15).

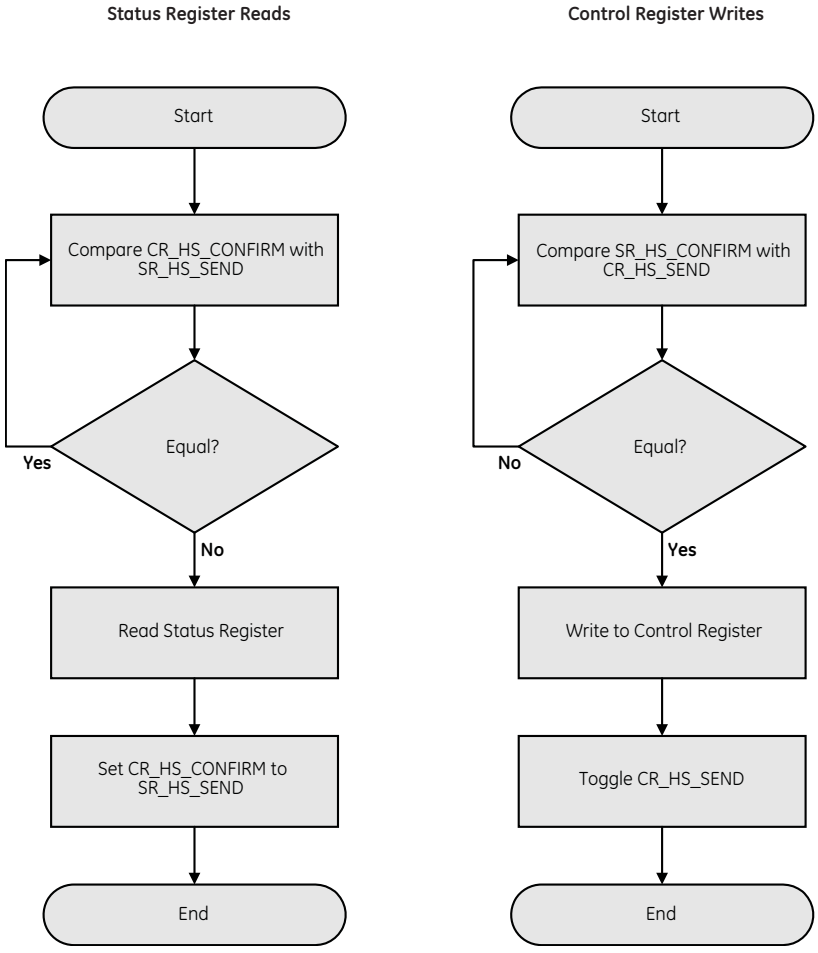


Figure 8-1: Handshaking flowchart

Input/output data during startup

DESCRIPTION

This section is only relevant when the control/handshake registers are enabled.

Bit 13 in the control register is used to ensure data consistency during start-up and at fieldbus off-line/on-line transitions. The bit should be treated as follows:

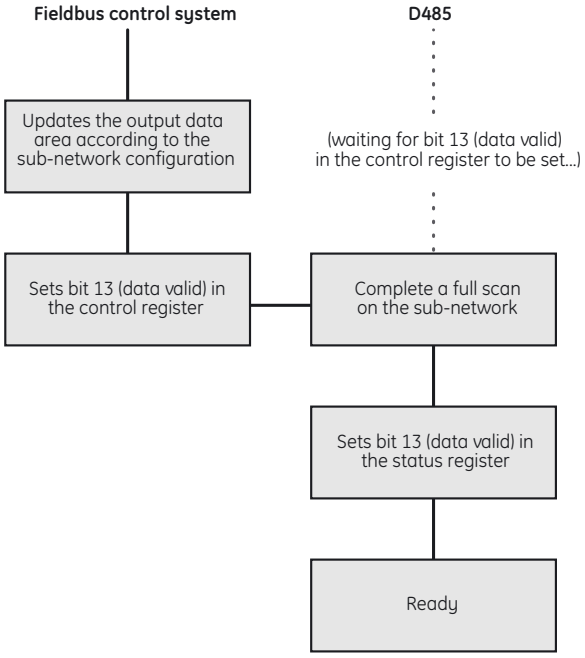


Figure 8-2: Input/output data during startup

When the fieldbus changes from off-line to on-line state, the fieldbus control system should clear (0) the 'data valid' bit in the control register. The D485 will then clear the 'data valid' bit in the status register.

During startup, the D485 waits for the fieldbus control system to set the 'data valid' bit in the control register. Before this is done, it will not communicate with the devices on the Modbus network.

The 'data valid' bit in the status register may in some cases be delayed. This latency can be caused by a missing node or a bad connection to a node with a long timeout value assigned to it.

Therefore, the fieldbus control system should not wait for this bit to be set before communicating with the sub-network devices. It should be considered as an aid for the fieldbus control system to know when all data has been updated.

NOTE

As with all changes to these registers, the handshaking procedure (refer to *Handshaking procedure* on page 8–3) must be followed.

Advanced fieldbus configuration

MAILBOX COMMAND

The mailbox commands are for advanced usage of the D485. Right clicking on the Fieldbus sub-menu items provides an option of inserting a mailbox.

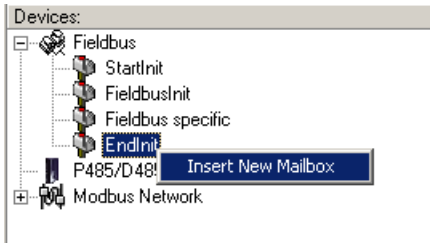


Figure 8-3: Mailbox command

By default, all mapped I/O input data is grouped in one attribute and is used for polled production (input) data, and all mapped I/O output data is used as polled consumed (output) data. The mailbox commands are used to split the I/O data and explicit data into different attributes. These attributes can be configured for polling, COS, and strobe I/O messaging through the DeviceNet master.

PARAMETER DATA INPUT AREA MAPPING

This command makes it possible to map attributes in the parameter data input mapping object (class B0h). By mapping data, a Get_Attribute_Single command from the DeviceNet master to the parameter data input mapping object can return a specified block of data. Up to 50 attributes can be mapped in this object. Attributes are mapped with start of attribute 1. Offset is set from the start of the parameter data input area and the length specifies the amount of bytes to map. Attribute 1 in the mailbox is followed by attribute 2, and continues to attribute 50. If length is set to zero, the attribute will not exist. As such, it is possible to map only object 1 and 10 by letting the length of attributes 2 through 9 be zero. It is only necessary to include mailbox information up to the last attribute number.

If any offset or length is invalid, the length and offset will be set to zero in the mailbox answer and the attribute will not be mapped.

Parameter	Description
Command initiator	Application
Command name	PARAMETER_INPUT_MAP
Message type	02h
Command number	0004h
Fragmented	No
Extended header data	---
Command data	Offset and length of the attributes to map
Response data	Indicates if the message was accepted

A command and response layout example for setting 1 to 5 attributes is shown below.

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	
Command	0004h	0004h	*PARAMETER_INPUT_MAP*
Data size	0014h	0014h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Offset for Attribute 1	Offset for Attribute 1	*Offset from parameter data input area start*
Message data word 2	Length for Attribute 1	Length for Attribute 1	
Message data word 3	Offset for Attribute 2	Offset for Attribute 2	
Message data word 4	Length for Attribute 2	Length for Attribute 2	
Message data word 5	Offset for Attribute 3	Offset for Attribute 3	
Message data word 6	Length for Attribute 3	Length for Attribute 3	
Message data word 7	Offset for Attribute 4	Offset for Attribute 4	
Message data word 8	Length for Attribute 4	Length for Attribute 4	
Message data word 9	Offset for Attribute 5	Offset for Attribute 5	
Message data word 10	Length for Attribute 5	Length for Attribute 5	

Figure 8-4: Command and response layout for parameter input area

The following figure shows the mailbox command to set 1 to 5 attributes. Note that the message field opens only after entering the data size.

Attribute 1 length = 4 bytes
Attribute 2 length = 2 bytes
Attribute 3 length = 2 bytes
Attribute 4 length = 2 bytes
Attribute 5 length = 6 bytes

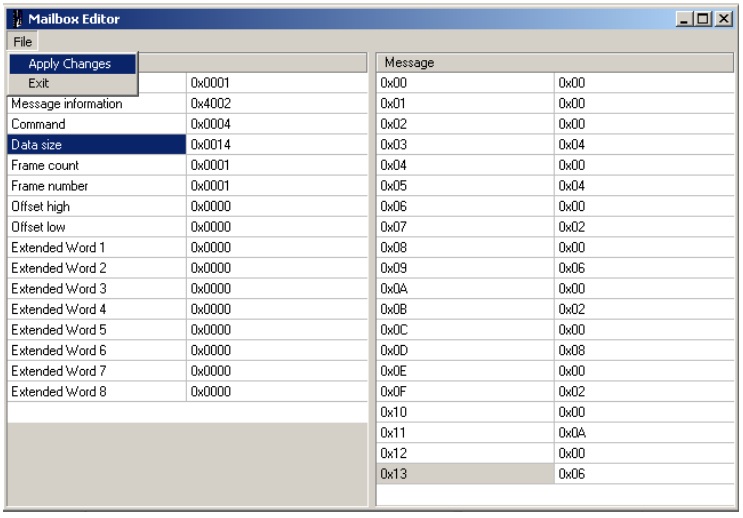


Figure 8-5: Mailbox example for parameter data input mapping

The mailbox command should be saved before closing the mailbox command window.

PARAMETER DATA OUTPUT AREA MAPPING

This command makes it possible to map attributes in the parameter data output mapping object (class B1h). By mapping data, a Get_Attribute_Single command or a Set_Attribute_Single command from the DeviceNet master to the parameter data output mapping object can return a specified block of data or write a block of data. Up to 50 attributes can be mapped. Attributes are mapped with start of attribute 1. The offset is set from the start of the parameter data output area and length specifies the amount of bytes to map. Attribute 1 in the mailbox is followed by attribute 2, and continues to attribute 50. If length is set to zero, the attribute will not exist. As such, it is possible to map only object 1 and 10 by letting the length of attributes 2 through 9 be zero. It is only necessary to include mailbox information up to the last attribute number.

If any offset or length is invalid, the length and offset will be set to zero in the mailbox answer and the attribute will not be mapped.

Parameter	Description
Command initiator	Application
Command name	PARAMETER_OUTPUT_MAP
Message type	02h
Command number	0005h
Fragmented	No
Extended header data	---
Command data	Offset and length of the attributes to map
Response data	Indicates if the message was accepted

A command and response layout example for setting 1 to 5 attributes is shown below.

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	
Command	0005h	0005h	*PARAMETER_OUTPUT_MAP*
Data size	0014h	0014h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Offset for Attribute 1	Offset for Attribute 1	*Offset from parameter data output area start*
Message data word 2	Length for Attribute 1	Length for Attribute 1	*Number of bytes to map*
Message data word 3	Offset for Attribute 2	Offset for Attribute 2	
Message data word 4	Length for Attribute 2	Length for Attribute 2	
Message data word 5	Offset for Attribute 3	Offset for Attribute 3	
Message data word 6	Length for Attribute 3	Length for Attribute 3	
Message data word 7	Offset for Attribute 4	Offset for Attribute 4	
Message data word 8	Length for Attribute 4	Length for Attribute 4	
Message data word 9	Offset for Attribute 5	Offset for Attribute 5	
Message data word 10	Length for Attribute 5	Length for Attribute 5	

Figure 8-6: Command and response layout for parameter output area

The following figure shows mailbox commands to set 1 to 4 attributes.

Attribute 1 length = 6 bytes
Attribute 2 length = 0 bytes
Attribute 3 length = 4 bytes
Attribute 4 length = 2 bytes

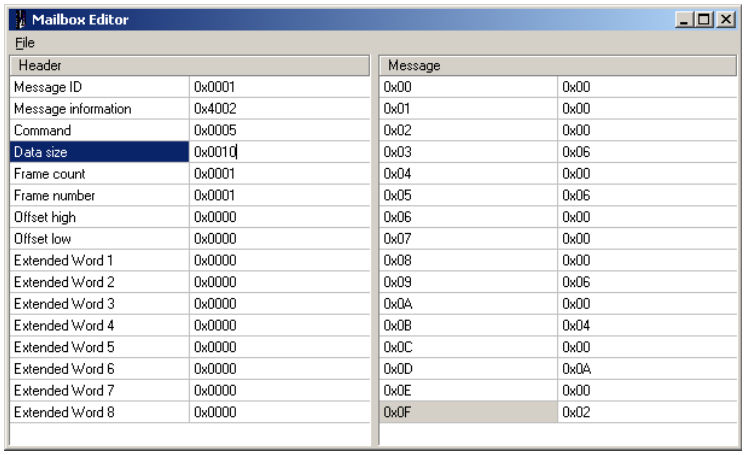


Figure 8-7: Mailbox example for parameter data output mapping

I/O DATA INPUT AREA MAPPING

This command makes it possible to map attributes in the I/O data input mapping object (class A0h). By mapping data, a Get_Attribute_Single command from the DeviceNet master to the I/O data input mapping object can return a specified block of data. Up to six attributes can be mapped in this object. Attributes are mapped with start of attribute 1. The offset is set from the start of the I/O data input area and length specifies the amount of bytes to map. Attribute 1 in the mailbox is followed by attribute 2, and continues to attribute 6. If length is set to zero, the attribute will not exist. As such, it is possible to map only object 1 and 5 by letting the length of attributes 2 through 4 be zero. It is only necessary to include mailbox information up to the last attribute number.

If any offset or length is invalid, the length and offset will be set to zero in the mailbox answer and the attribute will not be mapped.

The attributes of I/O data input mapping object class A0h are directly mapped to assembly object 04h instances as given below.

Class A0h, instance 01h, attribute	Corresponding instance in assembly object class 04h (attribute 03h)
1	64h
2	65h
3	66h
4	67h
5	67h
6	68h

Parameter	Description
Command initiator	Application
Command name	IO_INPUT_MAP
Message type	02h
Command number	0006h
Fragmented	No
Extended header data	---
Command data	Offset and length of the attributes to map
Response data	Indicates if the message was accepted

A command and response layout example for setting 1 to 5 attributes is shown below.

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	
Command	0006h	0006h	*IO_INPUT_MAP*
Data size	0014h	0014h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Offset for Attribute 1	Offset for Attribute 1	*Offset from I/O input area start*
Message data word 2	Length for Attribute 1	Length for Attribute 1	*Number of bytes to map*
Message data word 3	Offset for Attribute 2	Offset for Attribute 2	
Message data word 4	Length for Attribute 2	Length for Attribute 2	
Message data word 5	Offset for Attribute 3	Offset for Attribute 3	
Message data word 6	Length for Attribute 3	Length for Attribute 3	
Message data word 7	Offset for Attribute 4	Offset for Attribute 4	
Message data word 8	Length for Attribute 4	Length for Attribute 4	
Message data word 9	Offset for Attribute 5	Offset for Attribute 5	
Message data word 10	Length for Attribute 5	Length for Attribute 5	

Figure 8-8: Command and response layout for IO input area

The following figure shows mailbox commands to set 1 to 4 attributes.

Attribute 1 length = 2 bytes
Attribute 2 length = 4 bytes
Attribute 3 length = 2 bytes
Attribute 4 length = 2 bytes

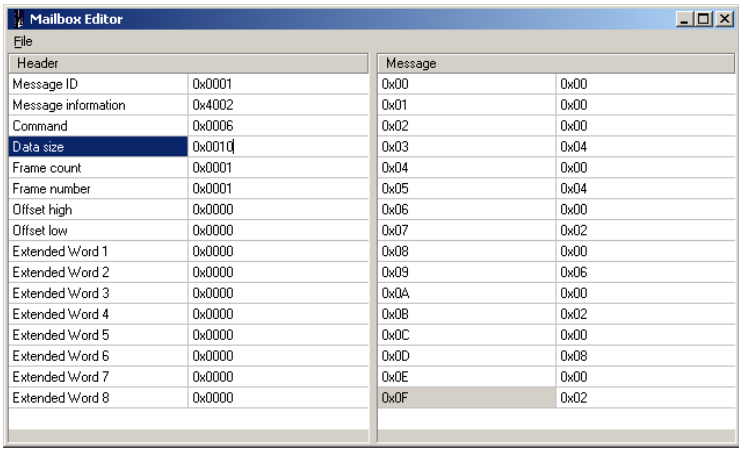


Figure 8-9: Mailbox example for IO data input mapping

I/O DATA OUTPUT AREA MAPPING

This command makes it possible to map attributes in the I/O data output mapping object (class A1h). By mapping data, a Get_Attribute_Single command or a Set_Attribute_Single command from the DeviceNet master to the I/O data output mapping object can return a specified block of data or write a block of data. Up to 6 attributes can be mapped in this object. Attributes are mapped with start of attribute 1. The offset is set from the start of the I/O data output area and length specifies the amount of bytes to map. Attribute 1 in the mailbox is followed by attribute 2, and continues to attribute 6. If length is set to zero, the attribute will not exist. As such, it is possible to map only object 1 and 5 by letting the length of attributes 2 through 4 be zero. It is only necessary to include mailbox information up to the last attribute number.

If any offset or length is invalid, the length and offset will be set to zero in the mailbox answer and the attribute will not be mapped.

The attributes of I/O data output mapping object class A1h are directly mapped to assembly object 04h instances as given below.

Class A1h, instance 01h, attribute	Corresponding instance in assembly object class 04h (attribute 03h)
1	96h
2	97h
3	98h
4	99h
5	9Ah
6	9Bh

Parameter	Description
Command initiator	Application
Command name	IO_OUTPUT_MAP
Message type	02h
Command number	0007h
Fragmented	No
Extended header data	---
Command data	Offset and length of the attributes to map
Response data	Indicates if the message was accepted

A command and response layout example for setting 1 to 5 attributes is shown below.

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	
Command	0007h	0007h	*IO_OUTPUT_MAP*
Data size	0014h	0014h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Offset for Attribute 1	Offset for Attribute 1	*Offset from I/O output area start*
Message data word 2	Length for Attribute 1	Length for Attribute 1	*Number of bytes to map*
Message data word 3	Offset for Attribute 2	Offset for Attribute 2	
Message data word 4	Length for Attribute 2	Length for Attribute 2	
Message data word 5	Offset for Attribute 3	Offset for Attribute 3	
Message data word 6	Length for Attribute 3	Length for Attribute 3	
Message data word 7	Offset for Attribute 4	Offset for Attribute 4	
Message data word 8	Length for Attribute 4	Length for Attribute 4	
Message data word 9	Offset for Attribute 5	Offset for Attribute 5	
Message data word 10	Length for Attribute 5	Length for Attribute 5	

Figure 8-10: Command and response layout for IO output area

The following figure shows mailbox commands to set 1 to 4 attributes.

Attribute 1 length = 2 bytes
Attribute 2 length = 4 bytes
Attribute 3 length = 2 bytes
Attribute 4 length = 2 bytes

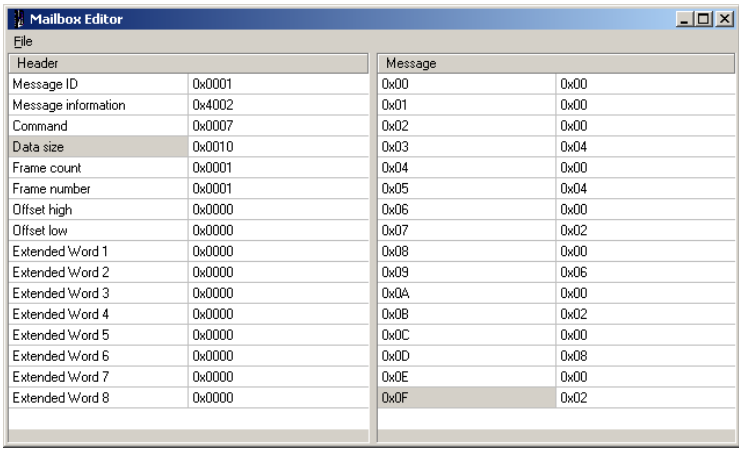


Figure 8-11: Mailbox example for IO data output mapping

Refer to chapter 9 for an application example using the mailbox commands.

CAUTION

Incorrect usage of mailbox commands may permanently damage the converter. For additional information, consult the product support team at GE Multilin.

D485 Modbus to DeviceNet Converter

Chapter 9: Application example

Introduction

OVERVIEW

The chapter describes how to configure the D485 Modbus to DeviceNet Converter to allow GE Multilin relays and meters to communicate on a DeviceNet network. The GE Multilin MM2 Motor Manager 2 and PQMII Power Quality Meter are used as examples.

GE Multilin relays and meters support a very useful feature called the Modbus User Map in their software. This feature can be used in configuring the D485 to reduce the number of Modbus transactions and improve communication speed.

It is assumed that the reader is familiar with serial communication, DeviceNet networks, and PLC architecture.

EQUIPMENT AND DOCUMENTS

The examples in this chapter make use of the following equipment and documentation

- D485 Modbus to DeviceNet Converter
- RS485 cable to connect D485 to the relays/meters (MM2 and PQMII)
- EnerVista P485/D485 Setup software with configuration cable
- EDS file for the D485
- standard DeviceNet cable with connectors
- 24 V DC power supply for the D485
- PLC with DeviceNet master card
- PQMII Power Quality meter and instruction manual (publication code GEK-106435D)
- Enervista PQMII Setup software
- MM2 relay and instruction manual (publication code GEK-106294B)
- MM2PC software

SYSTEM SETUP

This chapter describes how to set up the D485 with MM2 relay and PQMII meter to read and write parameters. It can also be used as a guideline to setup the D485 Modbus to DeviceNet Converter for communication with any GE Multilin relays.

The PQMII and MM2 devices are serially connected (daisy-chained) through RS485. The following data is set up for the PQMII:

- Read phase current Ia, Ib, and Ic actual values from memory locations 0240h to 0242h
- Read average current from memory location 0244h
- Read neutral current from memory location 0245h
- Read average phase voltage from memory location 0286h to 0287h

The following data is set up for the MM2:

- Read motor status from memory location 0023h
- Read switch input status from memory location 0010h
- Read motor load from memory location 0035h
- Read thermal capacity from memory location 0036h
- Read metered voltage from memory location 0040h
- The START A (command code 0x0005) and STOP (command code 0x0004) commands using Modbus function 10h: *Preset multiple registers*.

The memory addresses below are taken from the PQMII and MM2 instruction manuals available at http://www.GEmultilin.com. For the PQMII, we have:

Type	Parameter	Address	Format	Read/write
Actual values	Phase current Ia	0x0240	F1	Read only
Actual values	Phase current Ib	0x0241	F1	Read only
Actual values	Phase current Ic	0x0242	F1	Read only
Actual values	Average phase current	0x0244	F1	Read only
Actual values	Neutral current	0x0245	F1	Read only
Actual values	Average phase voltage (2 words)	0x0286	F3	Read only

For the MM2, we have:

Type	Parameter	Address	Format	Read/write
Actual values	Motor status	0x0023	F7	Read only
Actual values	Switch input status	0x0010	F100	Read only
Actual values	Motor Load	0x0035	F1	Read only
Actual values	Thermal Capacity	0x0036	F1	Read only
Actual values	Voltage	0x0040	F1	Read only
Commands	Command function code	0x1160	F1	Read/write
Commands	Command operation code	0x1161	F22	Read/write

The write command returns data in the response that should not be visible from the Fieldbus. Also, the write commands should only be sent if the data from the DeviceNet master has changed.

If the D485 detects a timeout while talking to the devices (PQMII and/or MM2), it should try to re-establish communications before it considers the device in subnet is missing (**Number of retries**), and then try again after some time (**Reconnect time**).

The serial communication parameters are set to 19200 bps, with no parity, 1 stop bit, and 8 data bits. The physical interface is set to RS485.

Modbus user map setup

DESCRIPTION

GE Multilin Relays and Meters support the Modbus User Map feature in their software. This feature can be used with the D485 to reduce the number of Modbus transactions and improve communication speed.

PQMII USER MAP

There are six parameters to be read from PQMII as indicated in the previous section. Normally, six read input register command transactions are required to read these parameters. However, these parameters can be grouped together with the Modbus User Map feature and read using only one read input register command transaction.

Set the Modbus User Map for the PQMII as follows.

1. Start the Enervista PQMII Setup software.
2. Establish communication between the device and PC.
3. Select the **Setpoint > User Map** menu item.
4. Set the user map registers as follows and save to the meter.

User map address	Parameter	User map data address
0000	0x0240	0x0100
0001	0x0241	0x0101
0002	0x0242	0x0102
0003	0x0244	0x0103
0004	0x0245	0x0104
0005	0x0286	0x0105

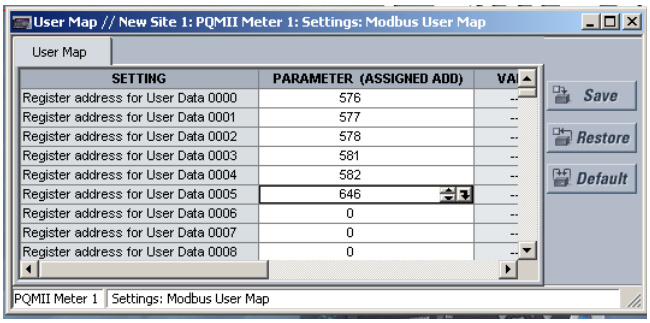


Figure 9-1: PQMII meter user map setting

The six parameters are now mapped to user memory map data at 0x0100 to 0x0106. These parameters can now be read by one *Read input data register* command at 0x0100 with register length = 7 words (note that average phase voltage value is in 32-bit).

MM2 USER MAP

As indicated earlier, there are five parameters to be read from MM2. Normally, five read input register command transactions are required to read these five parameters. However, all the parameters can be grouped together using the Modbus User Map feature and read using only one read input register command transaction.

Set the Modbus User Map for the MM2 as follows.

1. Start the MM2PC software.
2. Establish communications between the device and PC.
3. Select the **Setpoint > User Map** menu item.

4. Set the user map registers as follows and save to the relay.

User map address	Parameter	User map data address
1280	0x0023	0x0100
1281	0x0010	0x0101
1282	0x0035	0x0102
1283	0x0036	0x0103
1284	0x0040	0x0104

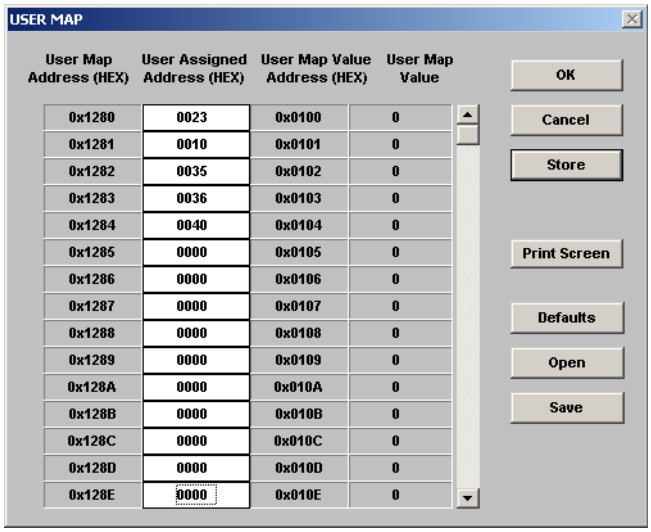


Figure 9-2: MM2 meter user map setting

The five parameters are now mapped to user memory map data at 0x0100 to 0x0104. These parameters can now be read by one *Read input data register* command at 0x0100 with register length = 5 words.

System configuration

OVERVIEW

An overview of the system configuration described in this document is shown below.

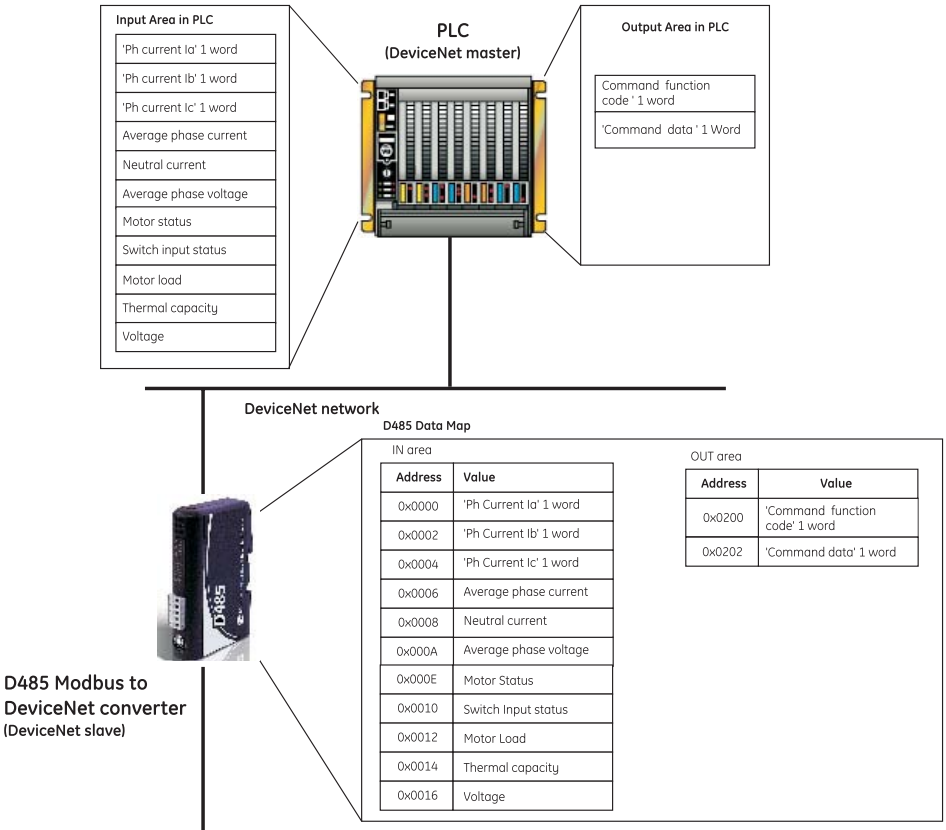


Figure 9-3: System configuration

The following procedures describe how to configure for the D485 with the PQMII and MM2. It is assumed that the reader has some basic knowledge of the Modbus RTU protocol and DeviceNet communication protocol.

INSTALLING THE ENERVISTA P485/D485 SETUP SOFTWARE

The following procedure describes how to configure the EnerVista P485/D485 Setup software.

1. Install the EnerVista P485/D485 Setup software.
2. Connect the configuration port of D485 to the PC via the configuration cable.
3. Connect the devices (PQMII and MM2) to D485 through the DB-9 Modbus network connector using the proper RS485 connections shown below.

DB9 pin	Description
5	Ground
8	RS485 +
9	RS485 –

STARTING THE CONFIGURATION WIZARD

Start the D485 configuration wizard as follows.

1. Launch the EnerVista P485/D485 Setup software.
2. A window for selecting a configuration will be displayed. Select the **Configuration Wizard** icon.

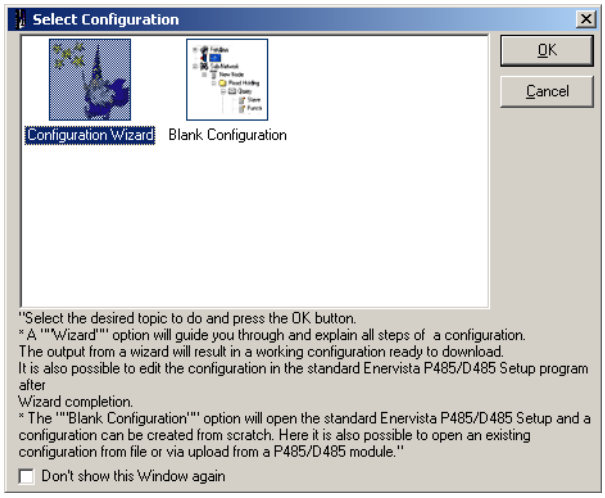


Figure 9-4: Select configuration wizard

3. Click **OK** to proceed to step 1 of the configuration wizard.

STEP 1: SELECTING THE FIELDBUS TYPE

The first step in the configuration wizard is setting the fieldbus type.

1. Set the **Fieldbus type** to "DeviceNet".

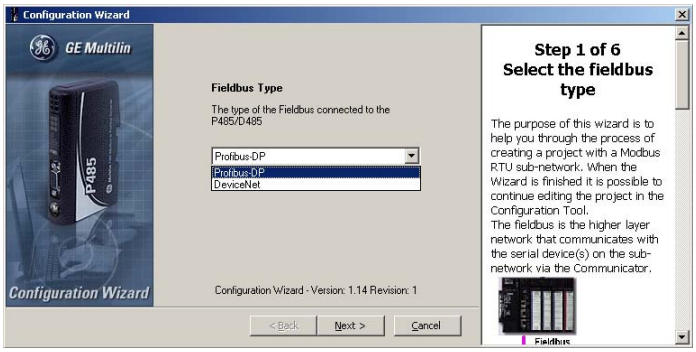


Figure 9-5: Select fieldbus type

2. Click **Next** to proceed to step 2 of the configuration wizard.

STEP 2: SELECTING THE SUB-NETWORK PROPERTIES

The second step in the configuration wizard is selecting the sub-network properties.

1. Set the Modbus network properties as follows: baud rate to 19200, with 8 data bits, no parity, RS485 physical standard, and 1 stop bit.

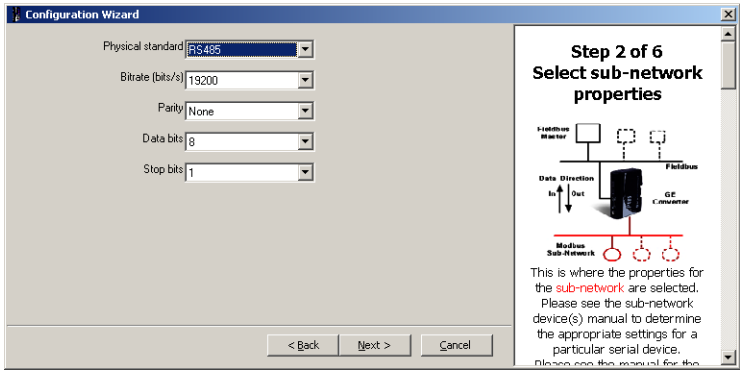


Figure 9-6: Modbus network properties

2. Click **Next** to proceed to step 3 of the configuration wizard.

STEP 3: INCLUDE DEVICE TYPES

The third step in the configuration wizard is to include device types. The PQMII and MM2 devices are added in this step.

1. The configuration wizard gives the option to create a configuration for a new device or to load a configuration of saved device.

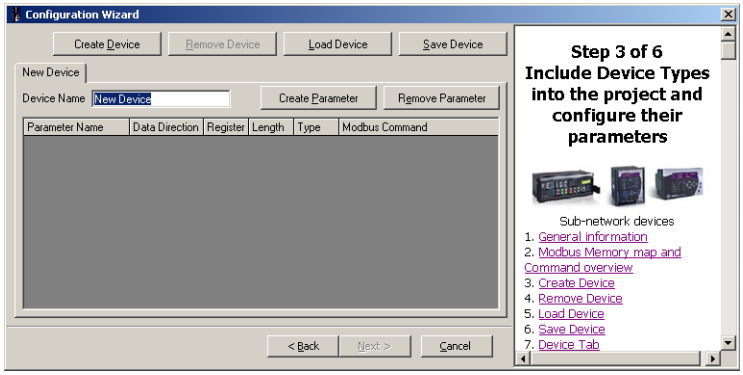


Figure 9-7: Device types

2. Configuration files for MM2 and PQMII are supplied with the EnerVista P485/D485 Setup software. Click the **Load Device** button to see the available configuration files.

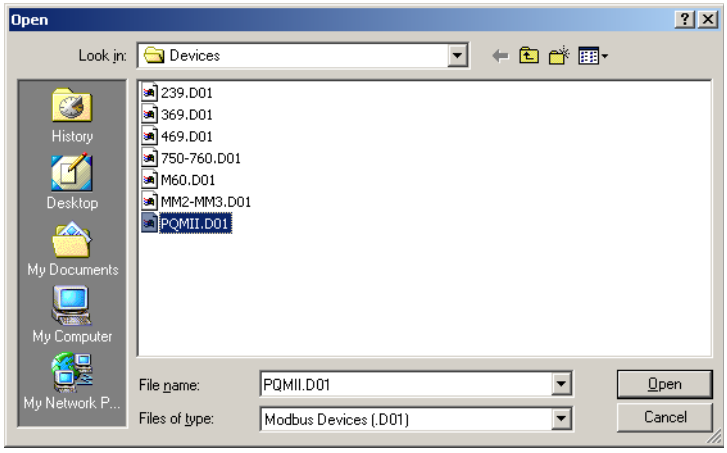


Figure 9-8: Available Modbus devices

3. Select the **PQMII.D01** file from the list and click on **Open**.
4. The software will display a list of the most commonly used parameters configured for the PQMII.

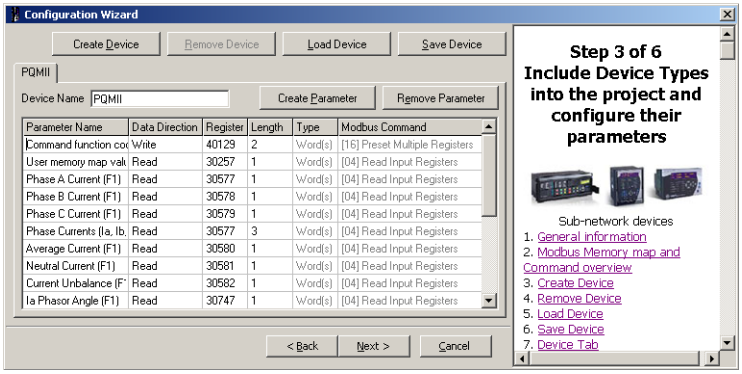


Figure 9-9: Parameters configured for PQMII

5. The Modbus User Map values are already present in the configuration file; as such, it is not required to create the new parameters. However, it is necessary to save number of registers to be read. Change the data length to 7 words as required.
6. Similarly, load the configuration file for MM2 by clicking the **Load Device** button and selecting the **MM2-MM3.D01** file.

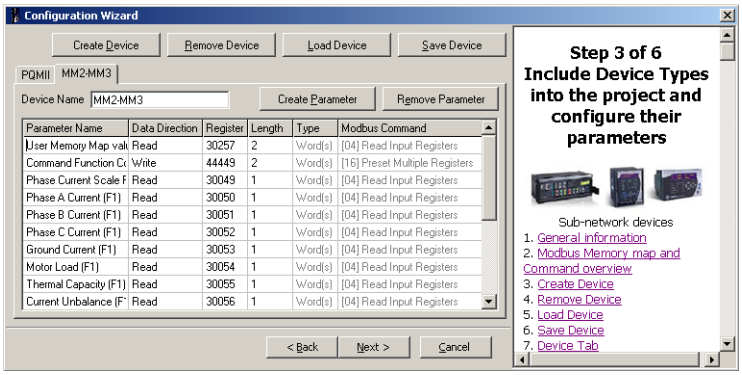


Figure 9-10: Parameters configured for the MM2

7. The software will display a list of the most commonly used parameters configured for the MM2. The Modbus User Map values are already present in the configuration file; as such, it is not required to create the new parameters. However, it is necessary to save number of registers to be read. Change the data length to 5 words as required.
8. The Motor Start A command and Motor Stop command can be executed using command function + operation code.
9. Save this device configuration by clicking the **Save Device** button.
10. Click **Next** to proceed to step 4 of the configuration wizard.

STEP 4: CONNECT DEVICES TO THE SUB-NETWORK

The fourth step in the configuration wizard is to connect the configured device types to the sub-network. The PQMII and MM2 devices are connected in this step.

1. Click **Next** to proceed to create nodes on Modbus network as shown below.

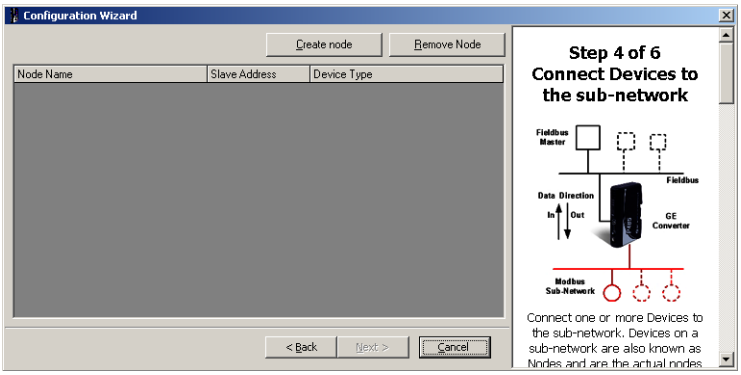


Figure 9-11: Connect devices to sub-network

2. There are two nodes on the Modbus network: the PQMII meter and the MM2 relay. To insert a node, click the **Create Node** button.
3. Set the **Node Name** as "PQMII Meter", the **Slave Address** to "20", and the **Device Type** to "PQMII".

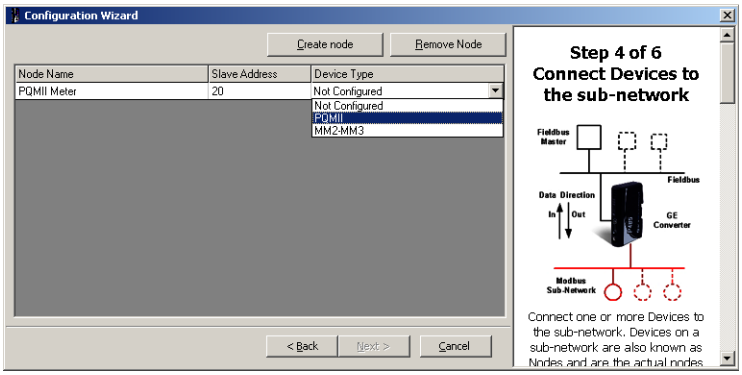


Figure 9-12: Node for PQMII meter

4. Similarly create another node for the MM2. Set the **Node Name** as "MMII Relay", the **Slave Address** to "4", and the **Device Type** as "MM2-MM3".

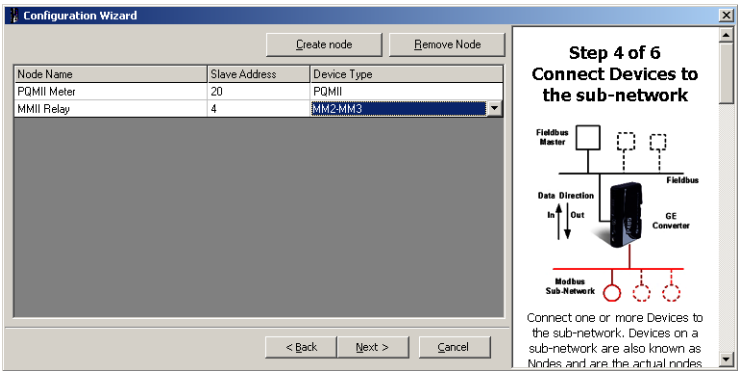


Figure 9-13: Node for MM2 relay

5. Click **Next** to proceed to step 5 of the configuration wizard.

STEP 5: SELECT PARAMETERS FOR EACH NODE

The fifth step in the configuration wizard is to select parameters for each node. The parameters for the PQMII and MM2 devices are selected in this step.

1. Parameters can now be added to each node. The tabs indicate the node name and slave address

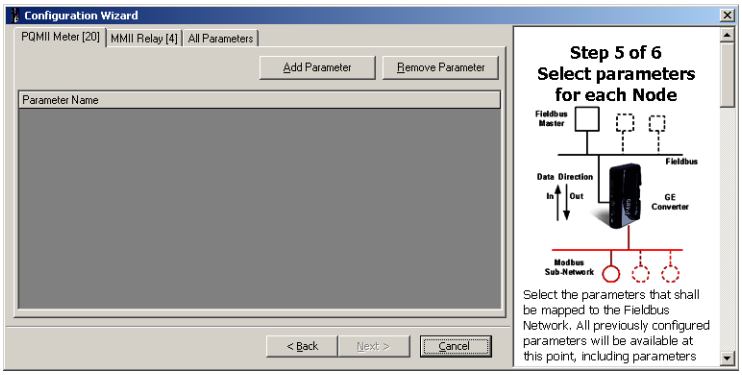


Figure 9-14: Select parameter window

2. To add parameter for the PQMII meter. select the “User memory map value (F1)” item from the drop-down list.

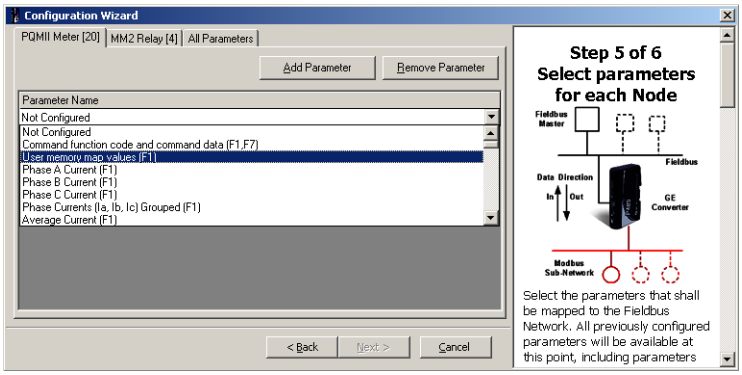


Figure 9-15: Choosing parameters for the PQMII meter node

3. Select parameters for MM2 relay by click on the **MM2 Relay (4)** tab followed by the **Add Parameters** button. Choose the following parameters from the drop-down list:

   User memory map values (F1)
   Command function code + operation code (F22)

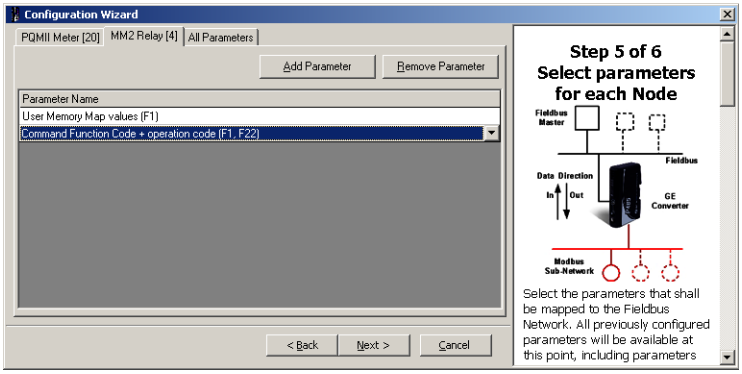


Figure 9-16: Choosing parameters for the MM2 relay node

4. The values in the bracket indicate Modbus data format codes – refer to the instruction manuals for details.

5. Click **Next** to proceed to the final step of the configuration wizard

STEP 6: CONFIGURATION REPORT

The final step in the wizard provides a configuration report for the device.

1. If desired, click on **Print** to print the configuration report.

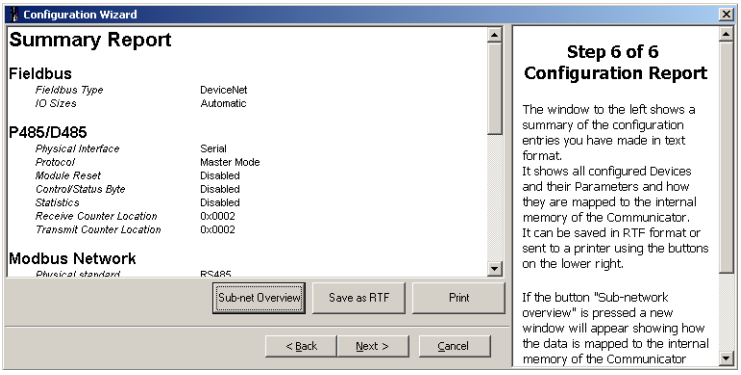


Figure 9-17: Configuration report

2. For future reference, saved the file in RTF format by clicking the **Save as RTF** button and selecting an appropriate directory.
3. Click the **Sub-net Oveview** button to view the data mapping.

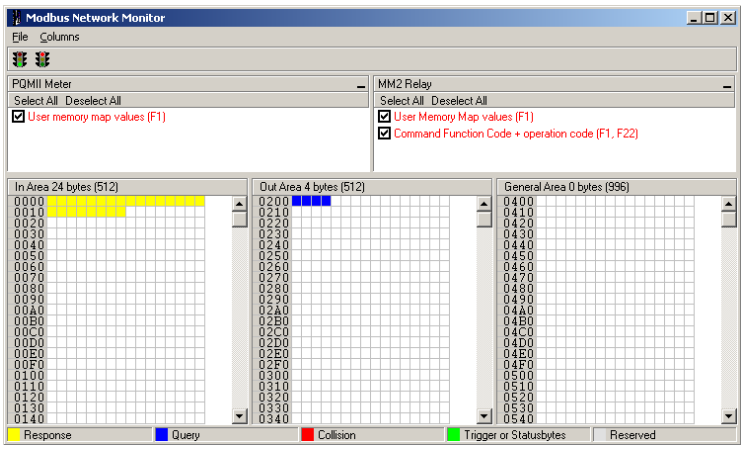


Figure 9-18: Modbus network monitor

4. Close the Modbus network monitor window.
5. Click on **Next** to complete the configuration wizard.

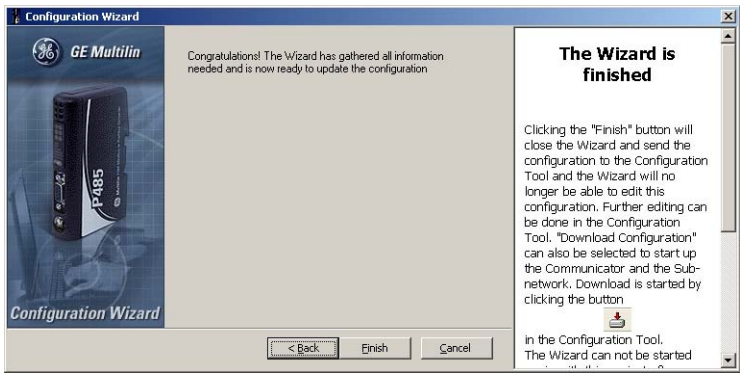
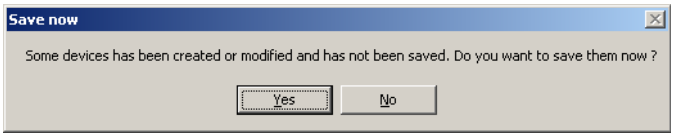


Figure 9-19: Wizard finished

SAVING DEVICE DATA

After the configuration wizard is complete, the software will prompt to save the device data (if necessary).



Click on **Yes** to save the device. The data can be saved in the same or different device files.

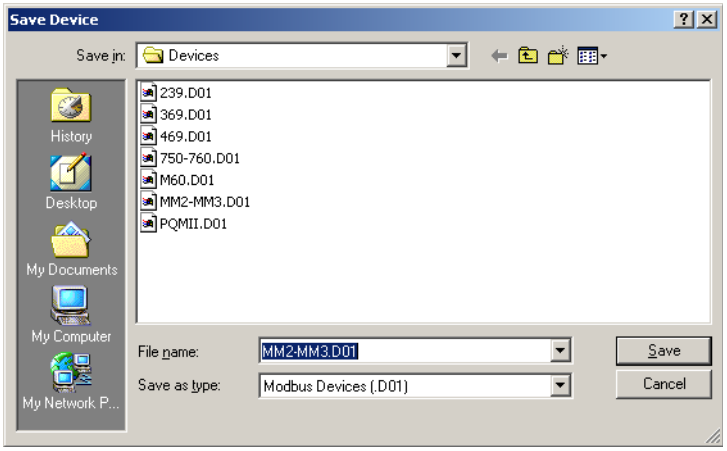
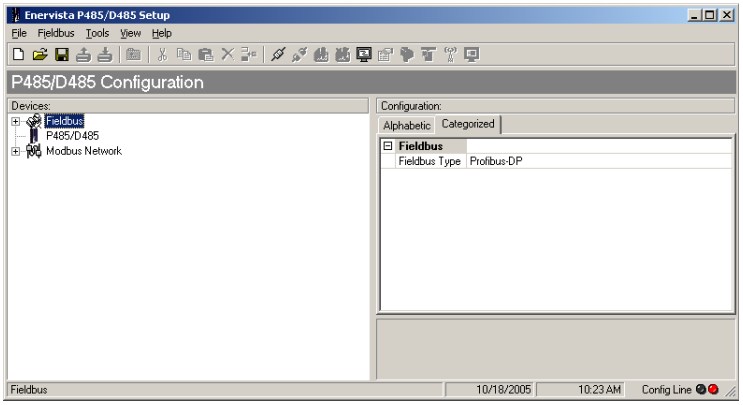


Figure 9-20: Save device file

CONFIGURING THE QUERIES

The main screen will appear after the wizard is closed.



1. Expand the **Modbus Network** item in the tree. All the configured parameters will appear as commands.

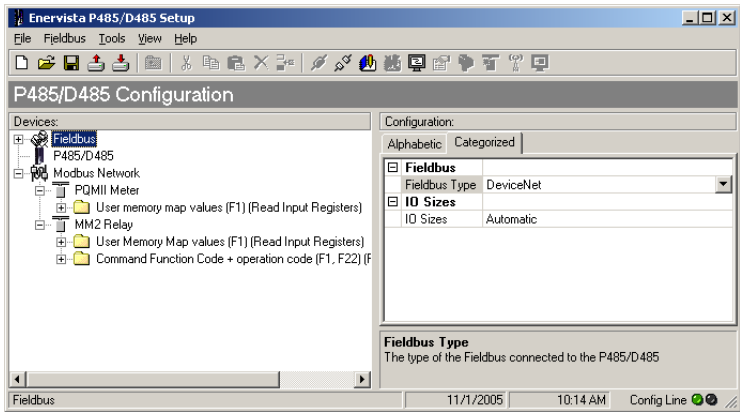


Figure 9-21: Expanding the tree

2. Expand the **User memory map values (F1)** command in the **PQMII Meter** item and click on **Query**.

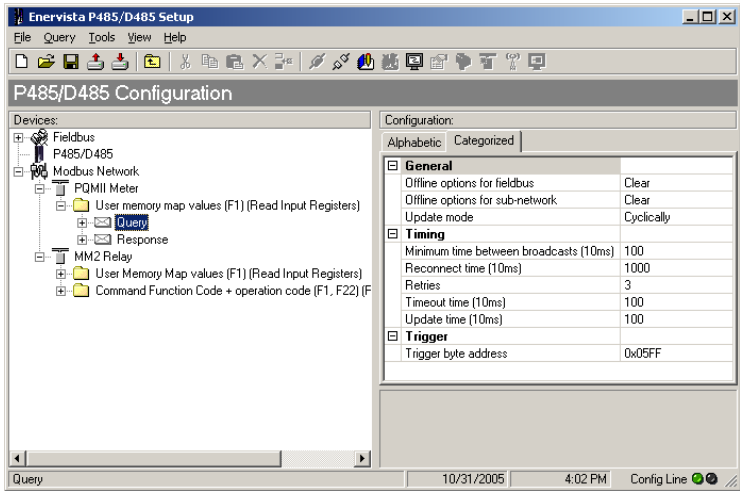


Figure 9-22: Expand query

3. Change the **Reconnect time** to 5 seconds by changing the value to 500 (500 × 10 ms = 5 seconds) and the **Retries** to 5.

4. Verify that the data **Update mode** is "Cyclically" with default **Update time** of 100 × 10 ms = 1000 ms. This can be changed to any value between 10 ms (1 × 10 ms) to 655350 ms (65535 × 10 ms).

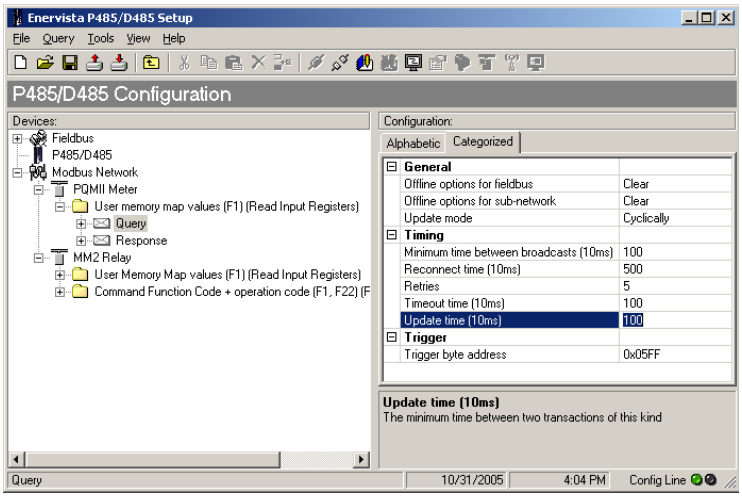


Figure 9-23: Changing configuration parameters for a query

5. Expand the **User memory map values (F1)** command in the **MM2 Meter** item and click on **Query**. Set the configuration parameters as above.

6. Expand the **Command Function code + operation code (F1, F22)** command in the **MM2 Meter** item and click on **Query**.

7. Set the following configuration parameters.

   Offline option for Fieldbus = Freeze
   Offline Options for sub-network = Freeze
   Upload mode = On data change
   Reconnect time = 500 (5 sec)
   Retries = 5

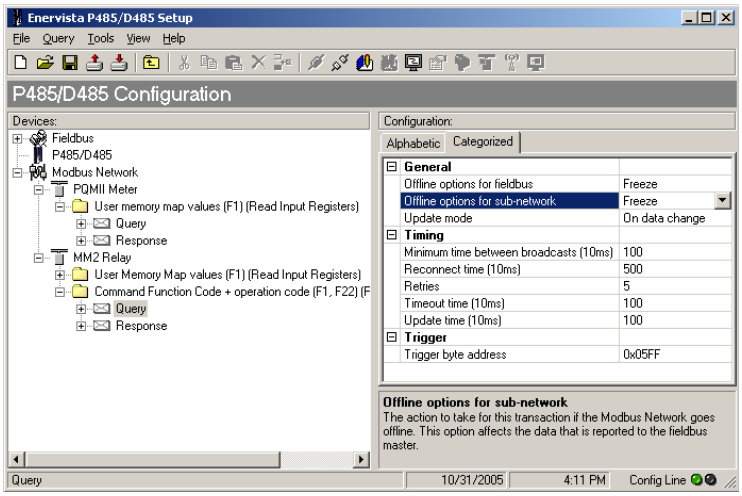


Figure 9-24: Query parameters for the command function

GROUPING I/O DATA

By default, all mapped I/O input data is grouped in one attribute and is used for polled production data, and all mapped I/O output data is used as polled consumption (output) data. The mailbox commands can be used to split this data into different attributes.

We have mapped 24 bytes of input data and 4 bytes of output data. The following procedure splits the input data in 16 bytes of I/O input data and 8 bytes of explicit input data.

1. Click on the **Fieldbus** item and change **IO sizes** option to "User defined" as shown below.

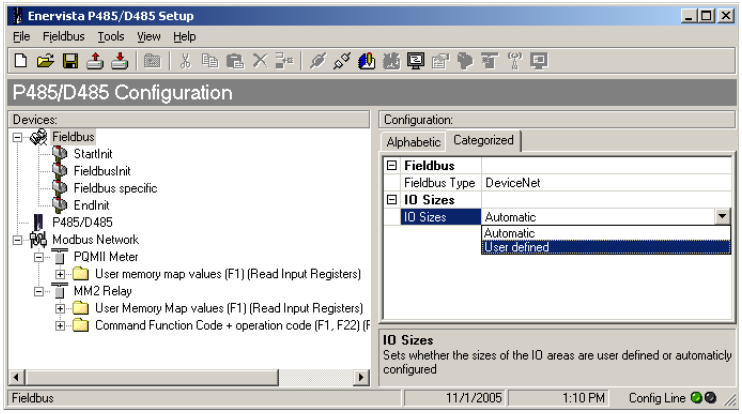


Figure 9-25: User defined IO sizes

2. Set the **IO Size In** (IO input data size) to 16 bytes (i.e. 0x000A) and the **IO Size Out** (IO output data size) to 4 bytes.

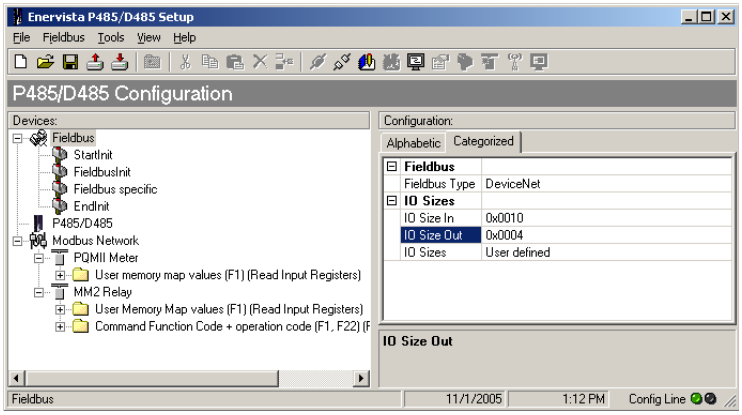


Figure 9-26: Setting the IO sizes

3. With this setting, input data is divided into two parts: the first 16 bytes of input is IO input data and remaining 8 bytes is explicit input data. All 4 bytes of output data is IO output data.

I/O DATA INPUT MAPPING

The IO data can be further assigned to different attributes of I/O data input mapping object A0h. Define the following five attributes out of 16 bytes of data.

Attribute	Bytes	Assignment
Attribute 1 (Input 1)	6 bytes	Phase current Ia, Ib, Ic
Attribute 2 (Input 2)	2 bytes	Average phase current
Attribute 3 (Input 3)	2 bytes	Neutral current
Attribute 4 (Input 4)	4 bytes	Average phase voltage
Attribute 5 (Input 5)	2 bytes	Motor status

This can be done by using I/O data input area mapping mailbox command (06h)

1. Right click on **EndInit** item and insert a new mailbox.
2. Set command code to 06 (row 3).
3. The offset and length of attribute for each attribute must be provided. Each attribute requires two words (four bytes) in message data; therefore, 5 attributes require 20 bytes. Enter 20 in the **Data size** field - it will automatically convert and display as 0x0014 (hex).

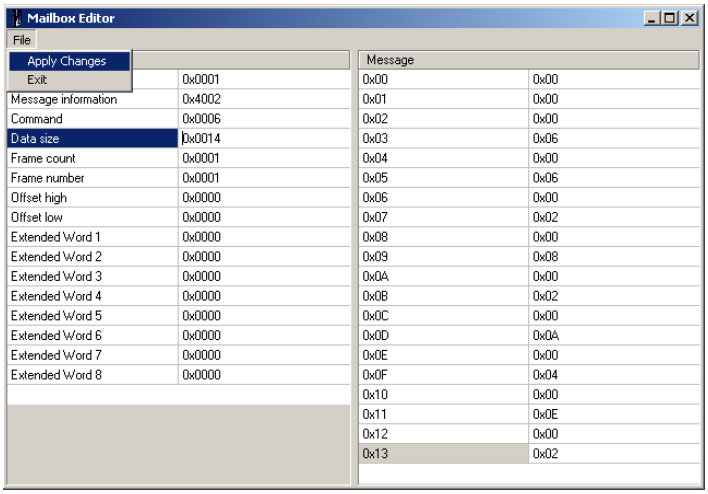


Figure 9-27: Setting offset and data length for IO data input attributes

4. The offset and data length for each attribute in bytes in given below.

Attribute	Offset	Data length
1	0x0000	0x0006
2	0x0006	0x0002
3	0x0008	0x0002
4	0x000A	0x0004
5	0x000E	0x0002

5. Enter the offset and data length values from the table above, then select the **File > Apply Changes** menu item to save the changes.
6. The above I/O data input mapping object class A0h attributes are mapped to assembly object class 04h instances with fixed attribute number 03h. The mapping is given below.

Class A1h, instance 01h, attribute	Corresponding instance in assembly object class 04h (Attribute 03h)	Description
1	96h	Input 1
2	97h	Input 2
3	98h	Input 3
4	99h	Input 4
5	9Ah	Input 5

PARAMETER DATA INPUT AREA MAPPING

The explicit data can be further assigned to different attributes of parameter data input mapping object B0h. Define the following four attributes out of eight bytes of explicit data.

Attribute	Bytes	Assignment
Attribute 1	2 bytes	Switch input status
Attribute 2	2 bytes	Motor load
Attribute 3	2 bytes	Thermal capacity
Attribute 4	2 bytes	Voltage

This can be done by using I/O data input area mapping mailbox command (04h)

1. Right click on the **EndInit** item and insert a new mailbox.
2. Set the command code to 04 (row 3).
3. The offset and length of attribute must be provided for each of the four attributes. Enter 16 in the Data size field - it will automatically convert and display as 0x0010.

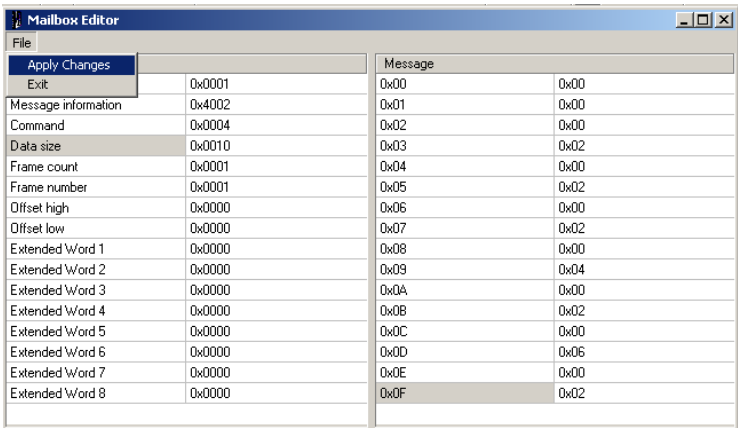


Figure 9-28: Offset and data length for parameter data input mapping attributes

4. The offset and data length for each attribute in bytes in given below.

Attribute	Offset	Data length
1	0x0000	0x0002
2	0x0002	0x0002
3	0x0004	0x0002
4	0x0006	0x0002

5. Enter the offset and data length values from the table above, then select the **File > Apply Changes** menu item to save the changes.

DOWNLOADING THE CONFIGURATION FILE

Save the configuration file for future use. The save command is available in **File** menu. The following procedure demonstrates how to save the configuration file to the D485.

1. To open the saved configuration file, select the **File > Open** menu item. The following window will appear.

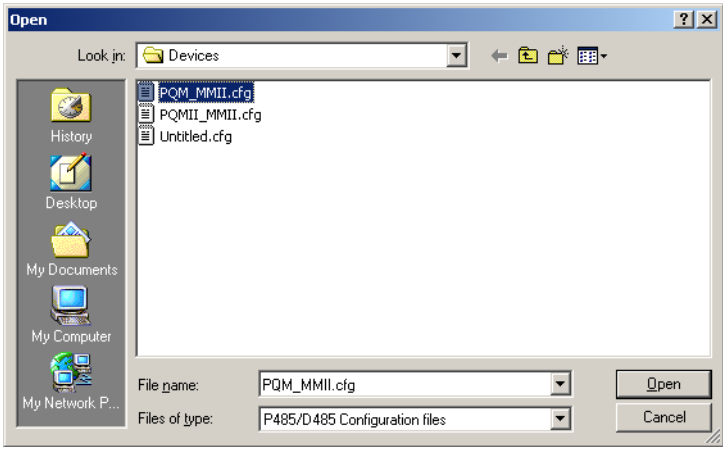


Figure 9-29: Opening a saved configuration file

2. To connect to the D485, select the **Tools > Port** menu item, then select the port connected to D485.
3. Click on the **Connect** icon.
4. Verify that the green LED is shown in the right corner of the configuration tool, then click the download icon in the toolbar. The download in progress bar will appear.

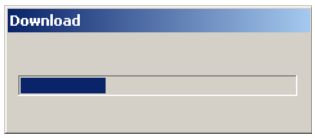


Figure 9-30: Download in progress

5. If the D485 does not respond to the download, ensure all connections are OK and that the port selection is valid. On some laptop computers, it might be worth trying the other serial ports. Also ensure that no other software (such as any PLC communication drivers) are blocking access to the serial ports.

DeviceNet network setup

DESCRIPTION

Each device on a DeviceNet network is associated with a EDS file that contains all necessary information about the device. This file is used by the DeviceNet configuration tool during configuration of the network. The file is available for download at the GE Multilin website at http://www.GEmultilin.com (the EDS file is named **D485.EDS**).

It is necessary to import the EDS file in the DeviceNet configuration tool to incorporate the D485 as a slave in the network. The properties for the D485 must then be configured from the DeviceNet configuration tool. This includes setting up the MAC ID, input/output data areas, and baud rate.

- **MAC ID**: The Mac ID in the DeviceNet configuration tool should be set to match the one selected using the on board address DIP switches of the D485.
- **Baud rate**: The baud rate on the device shall match with the DeviceNet network baud rate.
- **Setting up the input/output data areas**: Input/output data is used as IO polling data by the DeviceNet master. Select the correct IO size in the DeviceNet configuration tool.

NOTE

The last devices (end devices) on the DeviceNet network must be terminated with 120 ohm resistors.

SELECTING THE INPUT ATTRIBUTE FOR POLLING AND COS

Once the device is added into DeviceNet Network, any instance from the assembly object class can be selected for polled, COS and bit strobe data.

Select Input 1 as polled production data and Input 5 for COS production data through the drop down menu and then download the settings to the device. The data length and offset for each of the input and output can be read from the device in the parameter section of device properties.

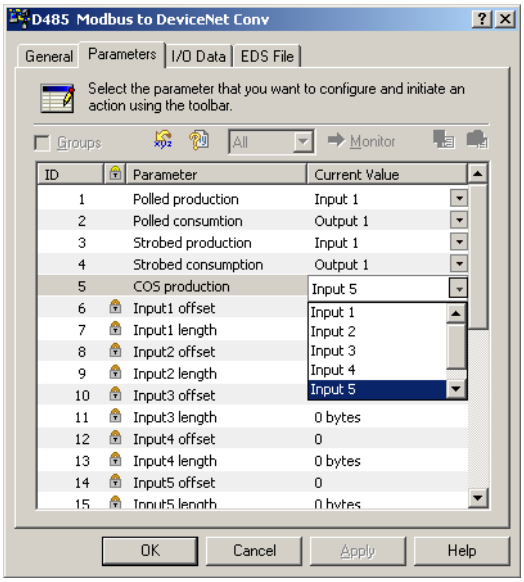


Figure 9-31: Selecting inputs for polled and COS production data

Use correct size of IO data for polled and COS connections. In this case, the input data size is 6 bytes for Polled IO and 2 bytes for COS IO. The output data size is 4 bytes for polled IO and 0 bytes for COS IO.

GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Chapter 10: Miscellaneous

Revision history

RELEASE DATES

Table 10–1: Release dates

Manual	Part No.	Revision	Release Date
GEK-113195	1601-0235-A1	1.0x	December 15, 2005

CHANGES TO THE MANUAL

As this is the first version of the D485 Modbus to DeviceNet Converter manual, there are no changes to report.

Warranty

GE MULTILIN WARRANTY STATEMENT

General Electric Multilin (GE Multilin) warrants each device it manufactures to be free from defects in material and workmanship under normal use and service for a period of 24 months from date of shipment from factory.

In the event of a failure covered by warranty, GE Multilin will undertake to repair or replace the device providing the warrantor determined that it is defective and it is returned with all transportation charges prepaid to an authorized service centre or the factory. Repairs or replacement under warranty will be made without charge.

Warranty shall not apply to any device which has been subject to misuse, negligence, accident, incorrect installation or use not in accordance with instructions nor any unit that has been altered outside a GE Multilin authorized factory outlet.

GE Multilin is not liable for special, indirect or consequential damages or for loss of profit or for expenses sustained as a result of a device malfunction, incorrect application or adjustment.

For complete text of Warranty (including limitations and disclaimers), refer to GE Multilin Standard Conditions of Sale.

GE Consumer & Industrial

D485 Modbus to DeviceNet Converter

Index

A

APPLICATIONS 1–3

B

BAUD RATE 1–4, 2–7, 4–4, 4–13
BROADCASTER 1–2

C

CHANGES TO THE MANUAL 10–1
COMMAND EDITOR 6–2, 6–3
COMPLIANCE 1–4
CONFIGURATION REPORT 4–8
CONFIGURATION SWITCHES 2–6
CONFIGURATION WIZARD 4–2
CONNECTING NODES 4–7
CONTACT INFORMATION 1–1
CONTROL CODES 8–2
CONTROL REGISTERS 8–1

D

DATA BITS 4–5
DATA DIRECTION 4–6
DATA EXCHANGE 3–2
DATA FLOW 4–4
DEVICE TYPES 4–5
DEVICENET
- configuration 2–7
- connector 2–2
- input/output data areas 2–7
- typical network arrangement 4–3

DIMENSIONS 1–4
DIN-RAIL CONNECTION 2–5
DOCUMENT CONVENTIONS 1–2

E

ENERVISTA P485/D485 SETUP
- see entry for SOFTWARE

ENVIRONMENT 1–4

F

FEATURES 1–3
FIELDBUS TYPE 4–3
FRAME EDITOR 6–1

G

GLOSSARY 1–2

H

HANDSHAKING PROCEDURE 8–3
HUMIDITY 1–4

I

INFORMATION WINDOW 4–10
INPUT/OUTPUT DATA 8–4
INSTALLATION 2–1
INTERNAL MEMORY BUFFER 3–3

L

LEDs 2–5

M

MAC ID 2–7
MAILBOX 8–5
MASTER MODE 5–1
MESSAGE DELIMITER 5–2
MODBUS
- address format 4–5
- commands 4–5
- configuration 4–13
- memory map 3–4
- menu 5–3
- network monitor 7–1
- protocol 3–5

N

NAVIGATION WINDOW 4–10
NETWORK TERMINATION 2–7
NODE MONITOR 7–3
NODE PARAMETERS 4–8
NODES 5–3

O

OPTION WINDOW 4–11
ORDERING 1–4

P

PARAMETER WINDOW 4–10, 5–2
PARITY 4–5, 4–13
POWER SUPPLY
- connections 2–4
- specifications 1–4

PROTECTION CLASS 1–4

Q

QUICK INSTALL 2–1

R

REVISION HISTORY 10–1
RS232 4–4
RS422 4–4
RS485 4–4

S

SCAN LIST 5–2
SOFTWARE
- advanced functions 8–1
- command editor 6–2
- communication model 5–1
- configuration wizard 4–2
- data exchange 3–1
- frame editor 6–1
- installation 4–1
- Modbus network monitor 7–1
- node monitor 7–3
- overview 4–1

SPECIFICATIONS 1–4
STATUS CODES 8–3
STATUS INDICATORS 2–5
STATUS REGISTER 8–1, 8–2
STOP BITS 4–5
SWITCHES 2–6

T

TECHNICAL SUPPORT 1–1
TEMPERATURE 1–4
TROUBLESHOOTING 2–8
TYPICAL APPLICATIONS 1–3

U

UNPACKING THE SWITCH 1–1

W

WARRANTY 1–1, 10–1
WEBSITE 1–1