Anybus® Communicator™

IIoT

USER MANUAL

SCM-1202-113 1.1 en-US ENGLISH

Connecting Devices™

Important User Information

Disclaimer

The information in this document is for informational purposes only. Please inform HMS Industrial Networks of any inaccuracies or omissions found in this document. HMS Industrial Networks disclaims any responsibility or liability for any errors that may appear in this document.

HMS Industrial Networks reserves the right to modify its products in line with its policy of continuous product development. The information in this document shall therefore not be construed as a commitment on the part of HMS Industrial Networks and is subject to change without notice. HMS Industrial Networks makes no commitment to update or keep current the information in this document.

The data, examples and illustrations found in this document are included for illustrative purposes and are only intended to help improve understanding of the functionality and handling of the product. In view of the wide range of possible applications of the product, and because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks cannot assume responsibility or liability for actual use based on the data, examples or illustrations included in this document nor for any damages incurred during installation of the product. Those responsible for the use of the product must acquire sufficient knowledge in order to ensure that the product is used correctly in their specific application and that the application meets all performance and safety requirements including any applicable laws, regulations, codes and standards. Further, HMS Industrial Networks will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features or functional side effects found outside the documented scope of the product. The effects caused by any direct or indirect use of such aspects of the product are undefined and may include e.g. compatibility issues and stability issues.

Table of Contents

Page

1 Preface 3

1.1 About This Document 3

1.2 Document Conventions 3

1.3 Trademarks 3

2 Description 4

2.1 Introduction 4

2.2 Data Exchange Model 5

2.3 Subnetwork Protocol 6

3 Installation 8

3.1 Installation Overview 8

3.2 External Parts 9

3.3 DIN Rail Mounting 9

3.4 Connectors and Indicators 10

3.5 LED Indicators 13

4 Configuration 14

4.1 Configuration Overview 14

4.2 Anybus Configuration Manager - Communicator 15

4.3 Basic TCP/IP Concepts 49

4.4 TCP/IP Configuration 50

4.5 Anybus Configuration Manager - IIoT 53

4.6 Web Interface 55

A Technical Data 57

A.1 General Specifications 57

A.2 Serial Interface 57

A.3 IIoT Interface 57

B License Information 58

This page intentionally left blank

1 Preface

1.1 About This Document

This document describes how to configure and use the Anybus Communicator IIoT.

The use of this product requires a basic knowledge of industrial network technology and of the MQTT and OPC UA protocols.

For additional documentation and software downloads, FAQs, troubleshooting guides and technical support, please visit www.anybus.com/support.

1.2 Document Conventions

Numbered lists indicate tasks that should be carried out in sequence:

1. First do this
2. Then do this

Bulleted lists are used for:

- Tasks that can be carried out in any order
- Itemized information

► An action

→ and a result

User interaction elements (buttons etc.) are indicated with bold text.

```
Program code and script examples
```

Cross-reference within this document: *Document Conventions, p. 3*

External link (URL): www.hms-networks.com

WARNING
Instruction that must be followed to avoid a risk of death or serious injury.

Caution
Instruction that must be followed to avoid a risk of personal injury.

Instruction that must be followed to avoid a risk of reduced functionality and/or damage to the equipment, or to avoid a network security risk.

Additional information which may facilitate installation and/or operation.

1.3 Trademarks

Anybus® is a registered trademark of HMS Industrial Networks. All other trademarks mentioned in this document are the property of their respective holders.

2 Description

2.1 Introduction

Anybus Communicator IIoT is intended for connecting non-networked industrial devices and equipment to industrial Ethernet networks using the MQTT and OPC UA protocols for IIoT services. The gateway can supply data to the IIoT service but cannot be controlled by it, which means that data flows only from the serial (OT) to the Ethernet (IT) side.

The Anybus Communicator gateways are configured using *Anybus Configuration Manager*, a family of configuration tools that have an easy to use graphical interface and that do not require programming skills.

Anybus Configuration Manager and additional related software and documentation are available at www.anybus.com/support.

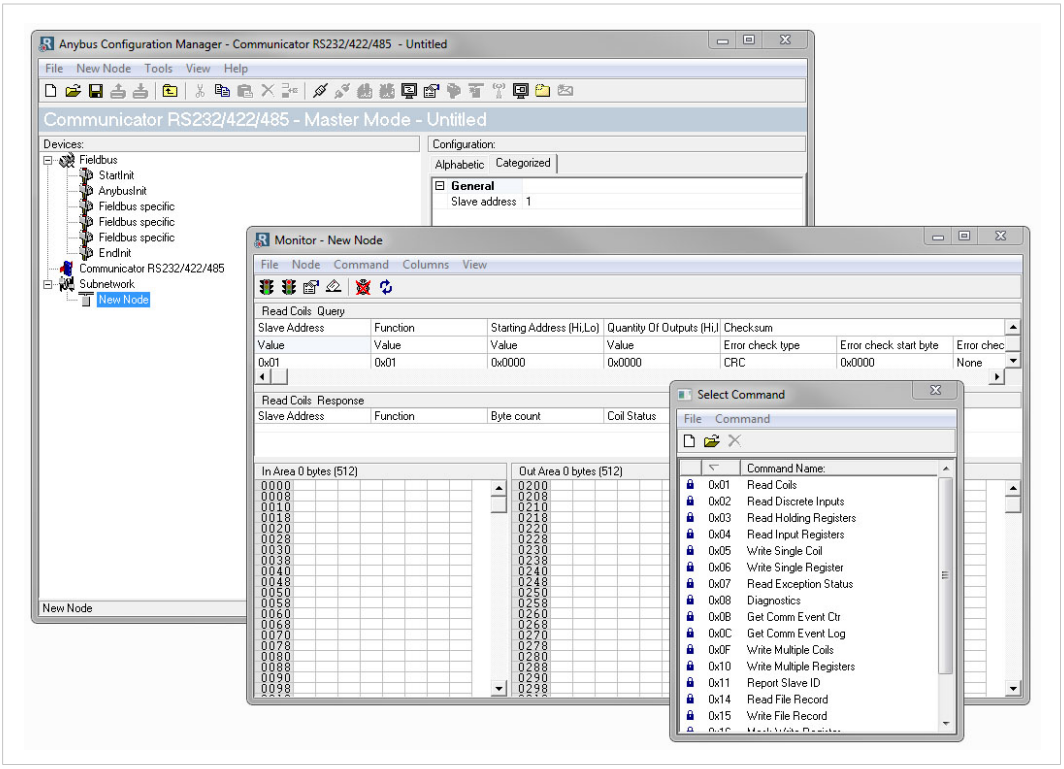


Fig. 1 Anybus Configuration Manager - Communicator

2.2 Data Exchange Model

2.2.1 Overview

The data exchanged on the serial subnetwork and on the higher level (IT) network reside in a shared memory buffer in the Anybus Communicator. The IT network can read data from the memory locations specified in Anybus Configuration Manager - Communicator.

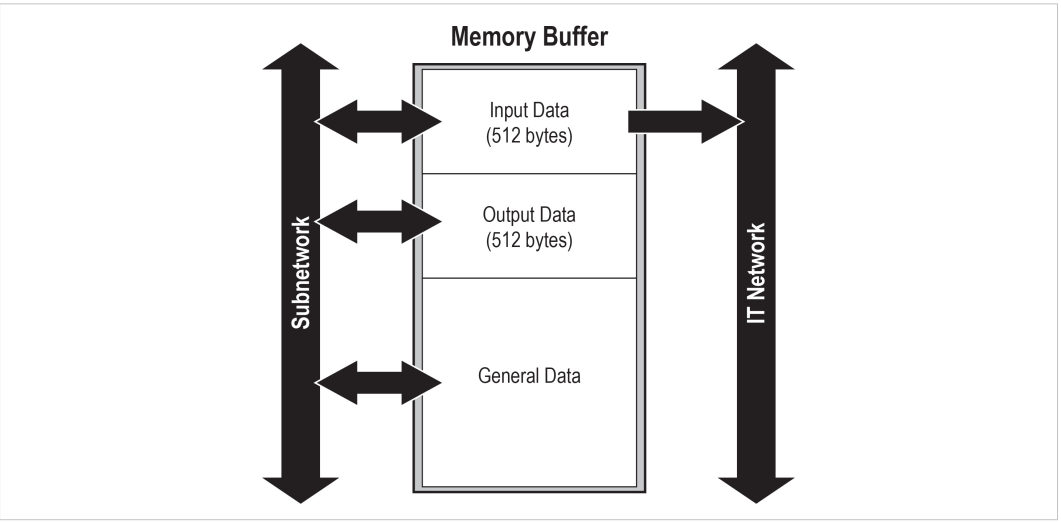


Fig. 2 Memory buffer structure

The internal memory buffer is divided into three areas based on their function:

Input Data (512 bytes)	This area can be read by the higher level (IT) network. A maximum of 256 data point tags can be defined for MQTT. A maximum of 80 data point tags can be defined for OPC UA.
Output Data (512 bytes)	This area cannot be accessed by the IT network but can be used for setting static values on the subnetwork.
General Data (max. 1024 bytes)	This area cannot be accessed by the IT network but can be used for transfers between individual nodes on the subnetwork or as a general “scratch pad” for data. The size of the General Data area depends on the amount of data exchanged on the subnetwork but can be up to 1024 bytes.

2.2.2 Memory Map

When building the subnetwork configuration in Anybus Configuration Manager - Communicator the areas in the memory buffer will be mapped to the following memory locations:

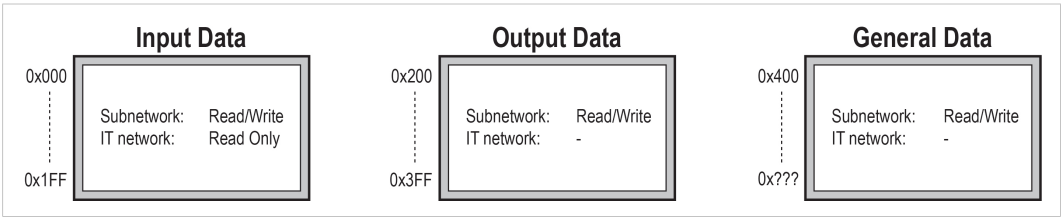


Fig. 3 Memory Map

2.3 Subnetwork Protocol

2.3.1 Protocol Modes

This section is a general description for Anybus Communicator gateways. Some models may not support all features or allow data flow in both directions.

The Anybus Communicator features three distinct modes of operation regarding subnetwork communication: *Master Mode*, *Generic Data Mode* and *DF1 Master Mode*.

Master Mode

In this mode, the Anybus Communicator acts as a master on the subnetwork and serial communication takes place in query-response fashion. The nodes on the network are not permitted to issue messages until they are addressed by the gateway.

Broadcasts are an exception: Most protocols offer some way of sending messages simultaneously to all nodes on the network without expecting a response. This is also implemented in the Anybus Communicator, which features a dedicated broadcaster node.

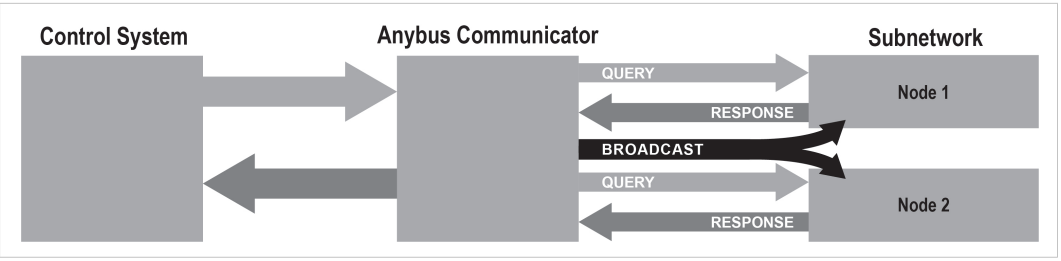


Fig. 4 Master Mode

Anybus Configuration Manager - Communicator comes preloaded with the most commonly used Modbus RTU commands in Master Mode. Note however that this does not prevent other protocols based on the same query-response message-scheme to be implemented.

Generic Data Mode

In this mode there is no master-slave relationship between the subnetwork nodes and the Anybus Communicator. Any node on the subnetwork, including the Anybus Communicator, may spontaneously *produce* or *consume* messages. Nodes do not have to respond to messages or wait for a query in order to send one.

The consumed data can be accessed from the higher level network, and/or vice versa.

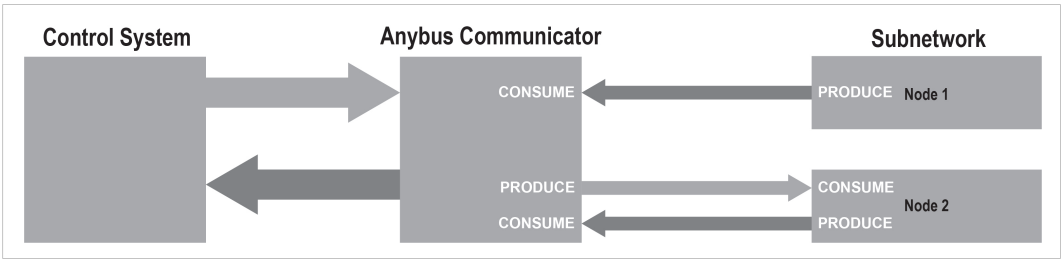


Fig. 5 Generic Data Mode

DF1 Master Mode

In this mode the Anybus Communicator act as a DF1 protocol master on the subnetwork. Serial communication takes place in query-response fashion. The nodes on the network are not permitted to issue messages until they are addressed by the gateway.

Communication in DF1 Master mode is based on *services*. A service represents a set of commands and operations on the subnetwork that have been predefined in the Anybus Communicator. Each service is associated with a set of parameters controlling how and when to use it on the subnetwork.

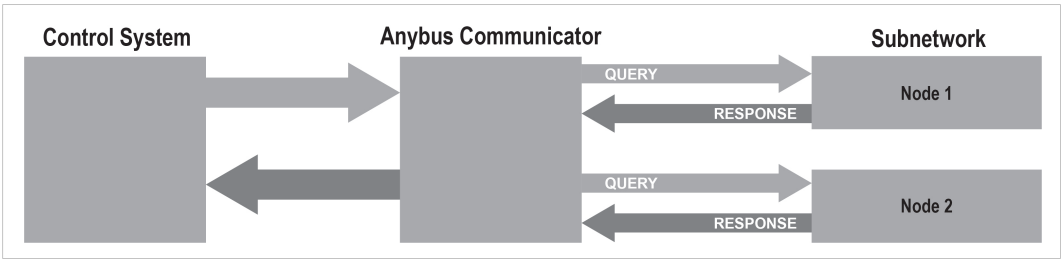


Fig. 6 DF1 Master Mode

Anybus Configuration Manager - Communicator comes preloaded with a number of services which can be selected by the user. The actual DF1 commands that perform the services during runtime have been predefined in the Anybus Communicator.

2.3.2 Protocol Building Blocks

The following building blocks are used in Anybus Configuration Manager - Communicator to describe the subnetwork communication.

Node	A *node* represents a single device on the subnetwork. Each node can be associated with a number of *transactions*.
Transaction	A *transaction* represents a complete serial telegram and consists of a number of *frame objects*. Each transaction is associated with a set of parameters controlling how and when to use it on the subnetwork.
Commands	A *command* is a predefined transaction which is stored in a list in Anybus Configuration Manager - Communicator. This simplifies common operations by allowing transactions to be stored and reused.
Frame Object	A *frame object* is a low level entity that is used to compose a *transaction*. A frame object can represent a fixed value (a constant), a range of values (limit objects), a block of data, or a calculated checksum.

See the documentation for Anybus Configuration Manager - Communicator on how to use protocol building blocks.

3 Installation

> ! This product contains parts that can be damaged by electrostatic discharge (ESD). Use ESD prevention measures to avoid damage.

3.1 Installation Overview

Prerequisites

The following items are required for installation and basic configuration:

- Configuration cable
- Subnetwork cable
- Ethernet cable
- Anybus Configuration Manager - Communicator RS-232/422/485 (4.5.0.0 or later)
- Anybus Configuration Manager - IIoT
- IPconfig (3.2.1.1 or later)

The Anybus Configuration Manager and IPconfig applications can be downloaded from www.anybus.com/support.

Basic installation steps

1. Mount the Anybus Communicator on the DIN rail.
2. Connect the serial and IIoT network interfaces.
3. Connect the configuration cable between the gateway and a PC.
4. Connect the power cable and apply power.
5. Check the LED indicators to verify that the gateway is running and that the serial and IT networks are connected. See *LED Indicators, p. 13*.
6. Use Anybus Configuration Manager - Communicator to set up the data exchange configuration in the Anybus Communicator.
7. Use IPconfig to configure the TCP/IP settings for the Anybus Communicator.
8. Use Anybus Configuration Manager - IIoT to create a tag configuration and transfer it to the Anybus Communicator.
9. Connect to the web interface of the Anybus Communicator and configure MQTT/OPC UA communication.

3.2 External Parts

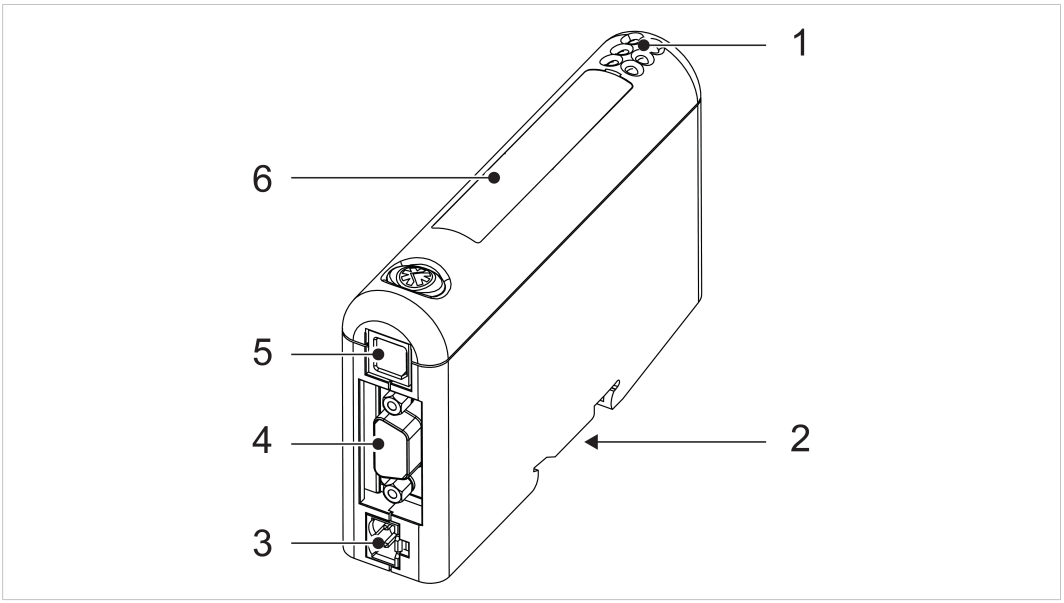


Fig. 7 Overview

1	LED indicators
2	DIN rail mount
3	Power connector
4	Serial subnetwork interface
5	PC connector
6	IIoT network interface

3.3 DIN Rail Mounting

> **!** The unit must be electrically grounded through the DIN rail for EMC compliance.

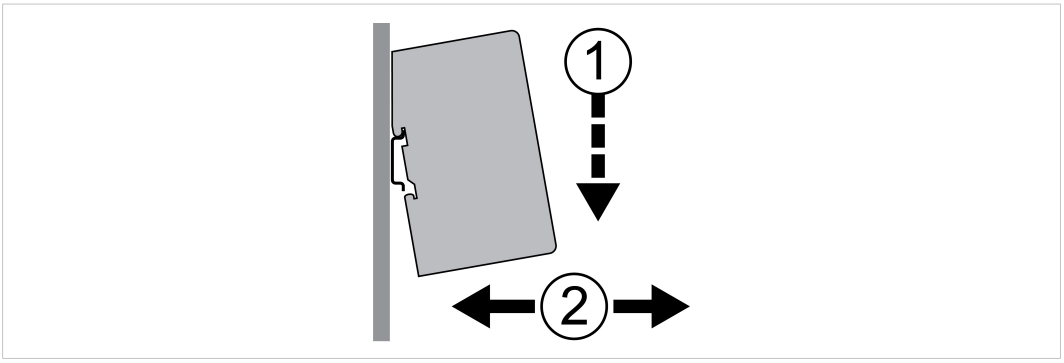


Fig. 8 DIN rail mounting

Push the unit gently downwards on the DIN rail to attach or relase it from the rail.

3.4 Connectors and Indicators

3.4.1 Serial Subnetwork Interface

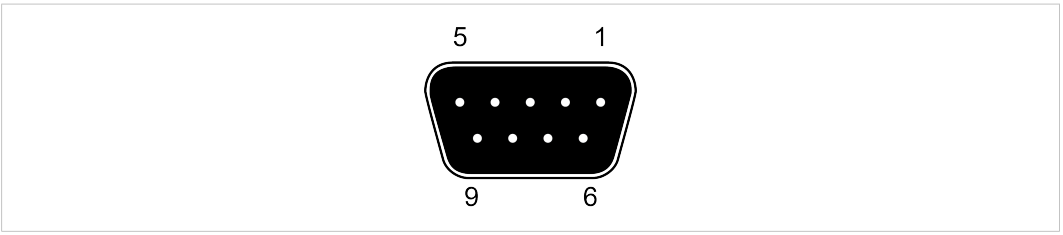


Fig. 9 D-sub connector (DE-9F)

Pin	Description	RS-232	RS-422	RS-485
1	+5 V Output (100 mA max.)	x	x	x
2	RS-232 Rx	x		
3	RS-232 Tx	x		
4	(reserved)			
5	Signal Ground	x	x	x
6	RS-422 Rx +		x	
7	RS-422 Rx -		x	
8	RS-485 + / RS-422 Tx +		x	x
9	RS-485 - / RS-422 Tx -		x	x
Housing	Shield	x	x	x

Bias and/or termination resistors may be required depending on the type of serial network.

> **!** Do not connect Signal Ground to Protective Earth (PE) of other nodes on the subnetwork as this may damage the on-board serial transceivers. Connect it only to the Signal Ground on other nodes.

3.4.2 Termination and Bias Resistors

Termination (RS-485 and RS-422)

The serial subnetwork should be terminated at each end node to prevent reflections on the serial lines. The resistor value should match the characteristic impedance of the cable, typically 100–120 Ω.

Bias Resistors (RS-485 only)

RS-485 enters an indeterminate state when idle, which may cause serial receivers to pick up noise from the serial lines and interpret it as data. To prevent this, the serial lines should be forced into a known state using *bias resistors*.

The bias resistors form a voltage divider that forces the voltage between the differential pair to be higher than the threshold for the serial receivers, typically >200 mV.

Bias resistors should only be installed on one node. Installing bias resistors on several nodes may compromise signal quality and cause transmission problems.

3.4.3 Serial Subnetwork Connection Examples

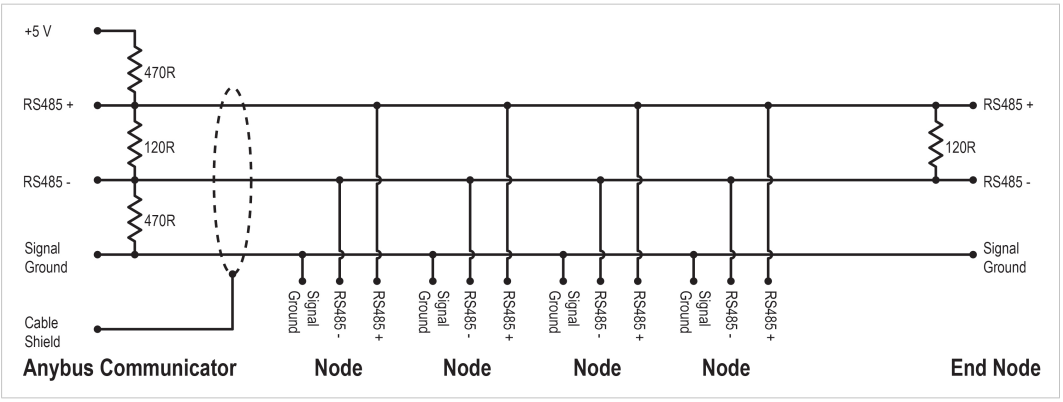


Fig. 10 Typical connection – RS485

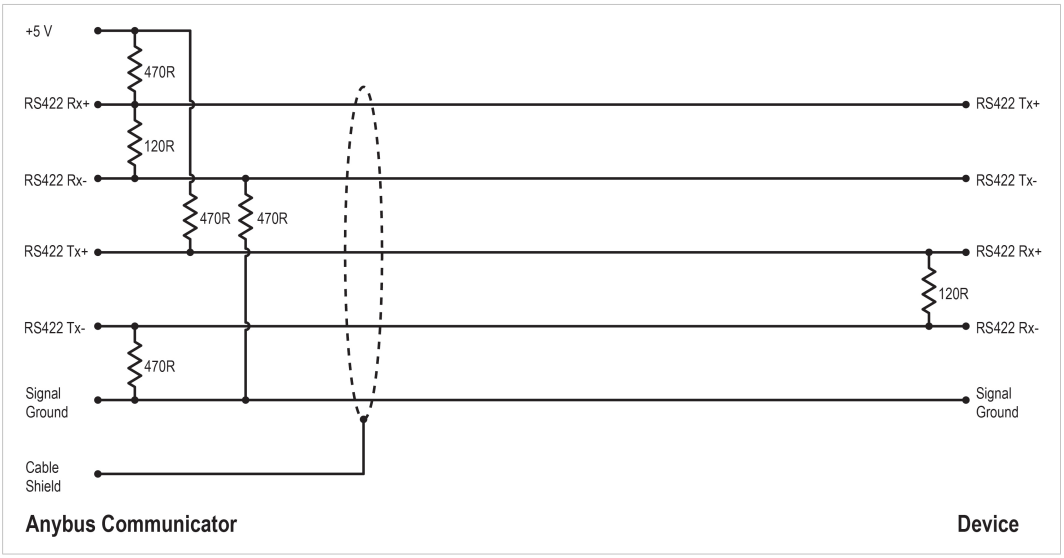


Fig. 11 Typical connection – RS422

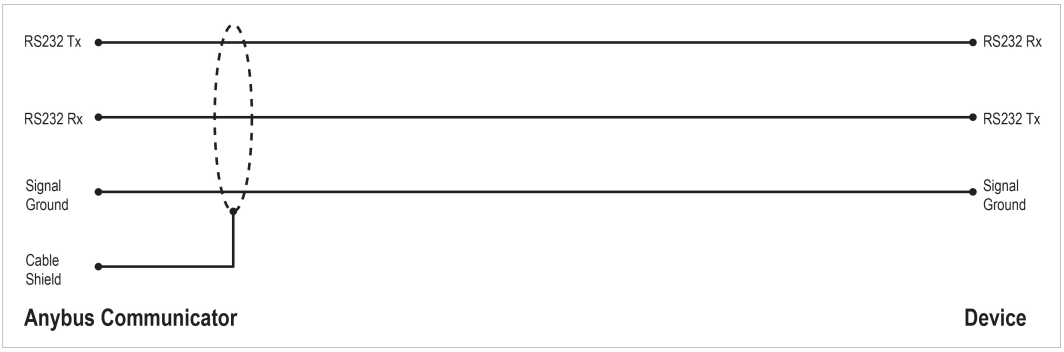
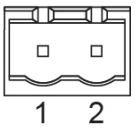


Fig. 12 Typical connection – RS232

3.4.4 Power Connector (2 pin)

> **!** Connecting power with reverse polarity or using the wrong type of power supply may damage the equipment. Make sure that the power supply is connected correctly and of the recommended type.

Pin	Signal
1	+24 VDC
2	Power Ground



3.4.5 PC Connector (RJ11)

Pin	Signal
1	GND (signal ground)
2	GND (signal ground)
3	RS-232 Rx (input)
4	RS-232 Tx (output)

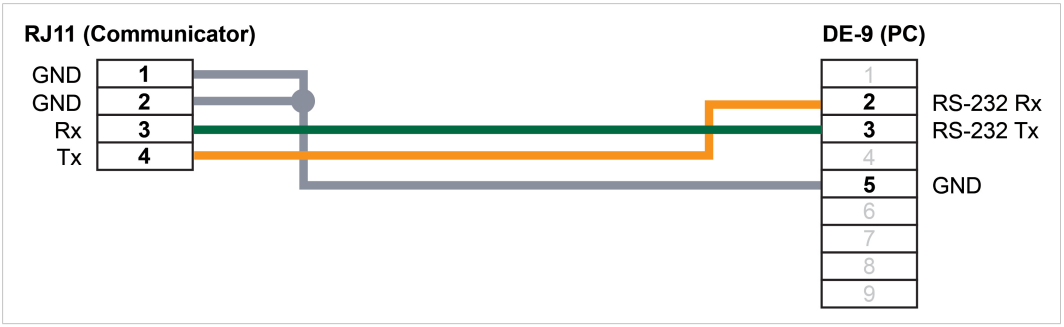
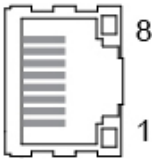


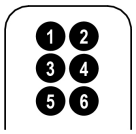
Fig. 13 Configuration cable

3.4.6 Ethernet Connector (RJ45)

Pin	Signal	Description
1	TD+	Transmit data +
2	TD-	Transmit data -
3	RD+	Receive data +
6	RD-	Receive data -
4,5,7,8	-	(reserved)



3.5 LED Indicators



LED 1 to 4 Model-specific information
LED 5 Serial subnetwork status
LED 6 Device status

LED	Indication	Meaning
1 - Network Status	Off	Offline or no power
	Green	Online, IP address assigned
	Green, flashing	No IP address assigned or no link
	Red	Fatal error, IP address conflict
	Alternating Red/Green	Firmware update in progress
2 - Module Status	Off	No power or IP address conflict
	Green	Normal operation
	Red	Fatal error
	Alternating red/green	Firmware update in progress
3 - Link/Activity 1 4 - Link/Activity 2	Off	Link not detected or no power
	Green	Link established (100 Mbit/s)
	Green, flickering	Link activity (100 Mbit/s)
	Red	Link established (10 Mbit/s)
	Red, flickering	Link activity (10 Mbit/s)
5 - Subnet Status	Off	No power
	Green	Running
	Green, flashing	Running, one or more transaction errors
	Red	Transaction error/timeout or subnet stopped
6 - Device Status	Off	No power
	Green	Initializing
	Green, flashing	Running
	Red	Bootloader mode
	Alternating red/green	Configuration invalid or missing

The Link/Activity LED indicators will show a red light for 10 Mbit/s connections. This is normal and does not indicate an error.

4 Configuration

4.1 Configuration Overview

4.1.1 Data Exchange Settings

The data exchange options and subnetwork protocol settings must be configured before communication can be set up. The configuration is created and downloaded to the Anybus Communicator IIoT using Anybus Configuration Manager - Communicator.

See *Anybus Configuration Manager - Communicator, p. 15*.

4.1.2 Network Communication Settings

To be able to communicate over Ethernet the IIoT interface needs a valid TCP/IP network configuration. TCP/IP settings can be configured from the IT network or locally using the IPconfig configuration tool.

TCP/IP settings can be set automatically from a DHCP or BootP server. If no DHCP server is found, the module will fall back on its current settings. If no current settings are available the module will halt and the status LED will indicate a network configuration error. The network configuration may still be accessed using IPconfig.

See *TCP/IP Configuration, p. 50*.

4.1.3 IIoT Configuration

The MQTT and OPC UA data and communication options are configured using Anybus Configuration Manager - IIoT and the web interface of the Anybus Communicator IIoT.

See *Anybus Configuration Manager - IIoT, p. 53* and *Web Interface, p. 55*

4.2 Anybus Configuration Manager - Communicator

Anybus Configuration Manager - Communicator is used to create a configuration and download it to the Anybus Communicator. The configuration can be created manually or using the built-in configuration wizard. The Anybus Communicator must have a valid configuration before you can set up network communication.

This chapter is a general description of the functions in Anybus Configuration Manager - Communicator. Some Anybus Communicator models may not support all described features or allow data flow in both directions.

4.2.1 Configuration Wizard

The configuration wizard will automatically create a basic subnetwork configuration from a predefined template based on input provided by the user.

The wizard will be displayed each time you create a new configuration unless it has been disabled on the **Options** page.

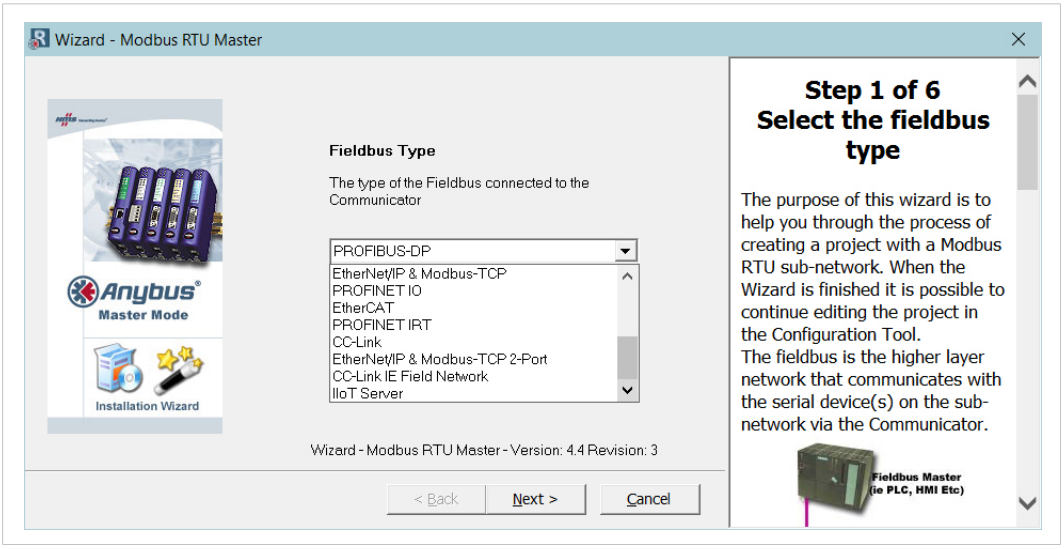


Fig. 14 Configuration wizard

The configuration wizard will help you through each step of creating and saving a basic configuration. You can then continue to edit the configuration in Anybus Configuration Manager - Communicator before downloading it to the Anybus Communicator.

4.2.2 Overview

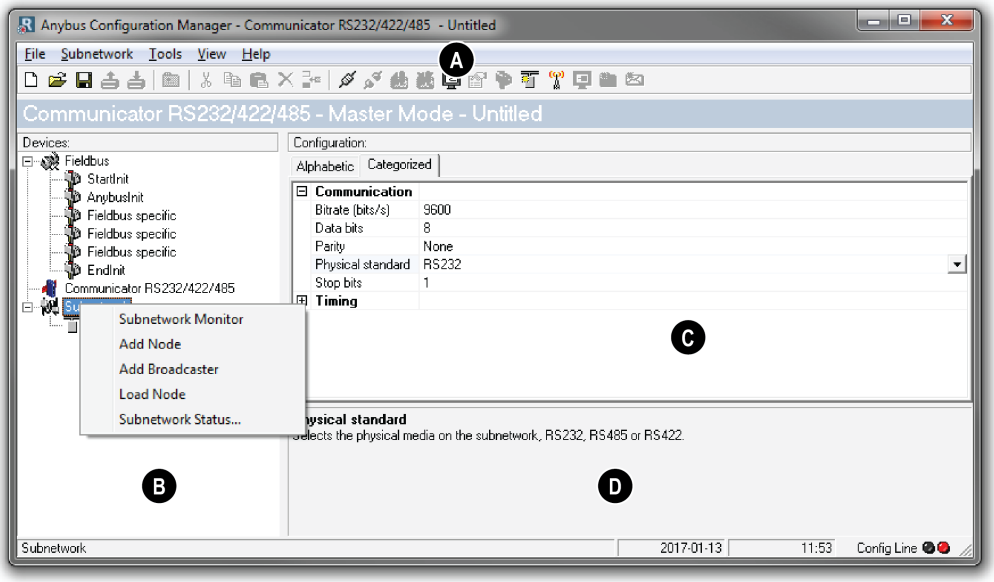


Fig. 15 Anybus Configuration Manager

A: Menus and Toolbar

The toolbar contains shortcut buttons to the most common commands. Each button has a tooltip describing its function.

The second menu (to the right of the File menu) will change depending on what is currently selected in the navigation tree.

B: Navigation Tree

A hierarchic tree view of the configuration, divided into three main sections:

Fieldbus	Communication settings for the higher level network interface
Communicator RS232/422/485	General settings for the Anybus Communicator
Subnetwork	Communication settings for the serial subnetwork

Select an entry to display its available parameters in the Parameter List. Right-click on the entry to show additional options.

C: Parameter List

Lists the parameters or options related to the currently selected entry in the Navigation Tree.

Values can be selected from a dropdown menu or entered manually depending on the parameter. Values can be specified in either decimal or hexadecimal format.

Example: The decimal value `42` can also be entered as `0x2A`.

D: Information Section

Displays information about the currently selected parameter.

Menus and Toolbar

File Menu	
New	Create a new configuration.
Open...	Open a previously saved configuration.
Save	Save the current configuration.
Save as...	Save the current configuration under a different file name.
Print...	Print the current configuration.
Properties...	Set the name and (optional) passwords for the configuration. **Lost passwords cannot be retrieved!**
Exit	Close Anybus Configuration Manager - Communicator.

Tools Menu	
Port	Select the COM port to use for configuration.
Upload configuration from ...	Fetch the active configuration from the Anybus Communicator .
Download configuration to ...	Send the current configuration from Anybus Configuration Manager - Communicator to the Anybus Communicator.
Start (Stop) Logging	Start/stop the Data Logger function.
Options	See *Options Dialog, p. 18*.

View Menu	
Toolbar	Show/hide the toolbar at the top of the main window.
Status Bar	Show/hide the status bar at the bottom of the main window.

Help Menu	
About	General information about Anybus Configuration Manager - Communicator and the currently connected Anybus Communicator. A help system is not included in this version of Anybus Configuration Manager - Communicator.

Options Dialog

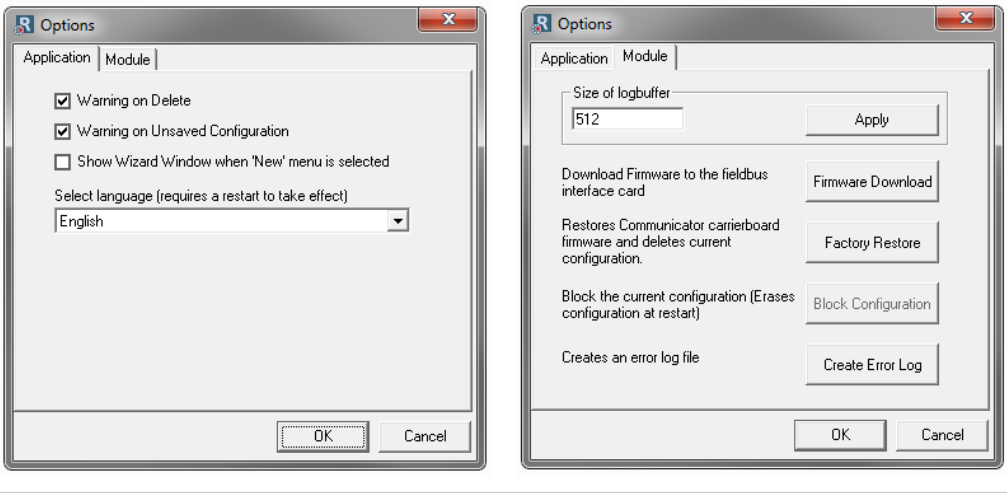


Fig. 16 Options dialog tabs

Application	
Warning on Delete	A confirmation dialog will be displayed when something is deleted.
Warning on Unsaved Configuration	A confirmation dialog is displayed when closing Anybus Configuration Manager - Communicator without saving the configuration.
Show Wizard ...	Open a wizard when creating a new configuration
Select Language	The language to use in Anybus Configuration Manager - Communicator. The program must be restarted for the new language seting to become active.

Module	
Size of logbuffer	The number of entries logged in each direction by the Data Logger. Can be set between 0 (no logging) and 512 (default).
Firmware Download	Used for updating the firmware in the higher level network interface.
Factory Restore	Resets the Anybus Communicator to the factory default settings. The settings in the higher level network interface are **not** affected.
Block Configuration	Prevents the downloaded configuration from being used by the Anybus Communicator. **Use with caution!**
Create Error Log	Create an error log file for troubleshooting.

4.2.3 Fieldbus Settings

When the Anybus Communicator is started, the higher level network interface will be initialized with the settings made in the configuration. For the IIoT interface there are no settings to be made except selection of the type of higher level network.

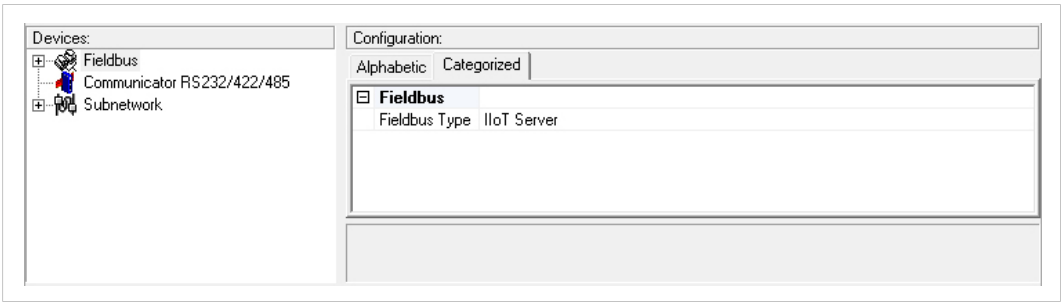


Fig. 17 Fieldbus settings

Fieldbus	
Fieldbus Type	Select the type of higher level network used (**IIoT Server**).

4.2.4 Communicator Settings

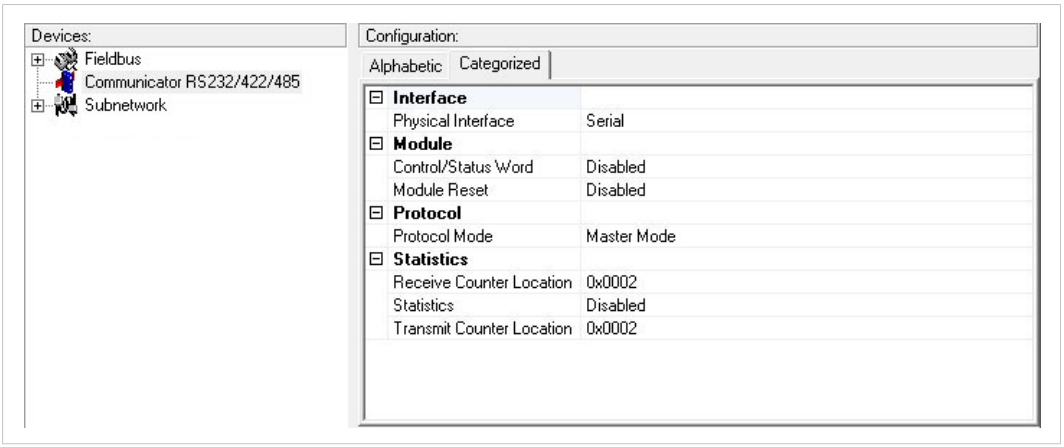


Fig. 18 Communicator parameters

Interface

The physical interface type of the subnetwork. Cannot be changed from **Serial**.

Control/Status Word

Enabled	Enables the Control and Status Registers. The *Data Valid* bit in the Control Register must be set to start the subnetwork communication.
Enabled but no startup lock	Same as Enabled, but the *Data Valid* bit does not have to be set.
Disabled	Control and Status Registers will not be used.

Module Reset

Defines how the gateway behaves in the event of a fatal error.

Enabled	The gateway will be restarted, no error indication.
Disabled	The gateway will halt and indicate an error.

Protocol Mode

Selects the protocol mode for the subnetwork.

Master Mode	Intended for Query/Response based protocols, where a single master exchanges data with a number of slaves.
Generic Data Mode	Intended for Producer/Consumer based protocols, where there is no master-slave relationship between the subnetwork nodes and the gateway.
DF1 Master	Intended for the DF1 protocol. The gateway can only be configured as a Master with half-duplex communication.

Statistics

The Transmit and Receive Counters indicate the number of successful transactions on the network. This parameter is primarily intended for debugging purposes.

Receive Counter Location	Specifies the location of the Receive Counter in the internal memory buffer.
Transmit Counter Location	Specifies the location of the Transmit Counter in the internal memory buffer.
Statistics	Enable/disable the counters.

4.2.5 Subnetwork Settings

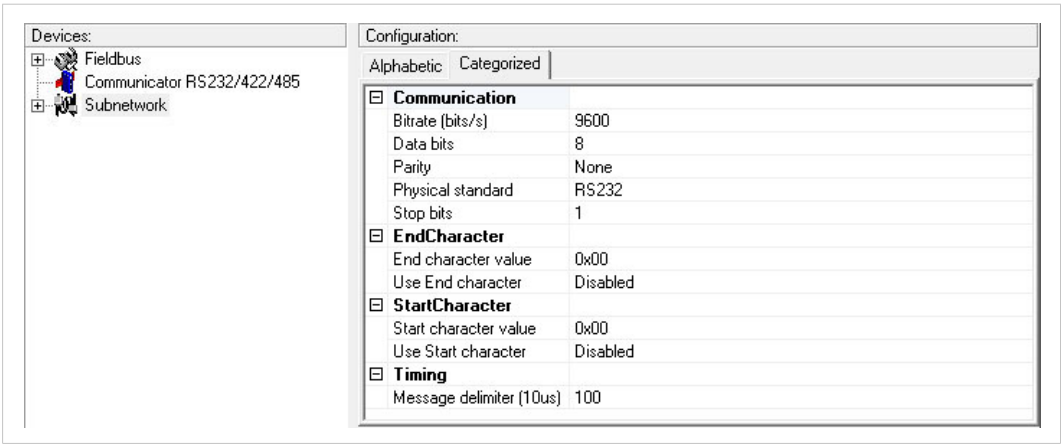


Fig. 19 Subnetwork parameters

Communication

The values that can be selected depend on the selected protocol mode.

Parameter	Description	Possible values
Bitrate (bits/s)	Selects the bit rate	1200 to 57600
Data bits	Selects the number of data bits	7, 8 (only 8 in DF1 Master mode)
Parity	Selects the parity mode	None, Odd, Even
Physical standard	Selects the interface type	RS232, RS422, RS485
Stop bits	Selects the number of stop bits	0, 1 (only 1 in DF1 Master mode)

EndCharacter/StartCharacter

This parameter group is only available in Generic Data Mode.

Start and End Characters are used to indicate the beginning and end of a serial message.

Parameter	Description	Possible values
End character value	End character ASCII code	0x00 to 0xFF
Use End character	Enable or disable use of the End character	Enabled/disabled
Start character value	Start character ASCII code	0x00 to 0xFF
Use Start character	Enable or disable use of the Start character	Enabled/disabled

Example: A message should be initiated with <ESC> and terminated with <LF>. The Start Character should then be 0x1B (ASCII code for <ESC>) and the End Character should be 0x0A (ASCII code for <LF>)

Timing – Message delimiter

This parameter is available in Master Mode and Generic Data Mode.

- **Master Mode**

  Message delimiter specifies the time that separates two messages in steps of **10 ms**.

  If set to 0 (zero), the standard Modbus delimiter of 3.5 characters will be used. The actual time in ms will then be calculated automatically based on the communication settings.

- **Generic Data Mode**

  Message delimiter specifies the time separating two messages in steps of **10 µs**.

DF1 Settings

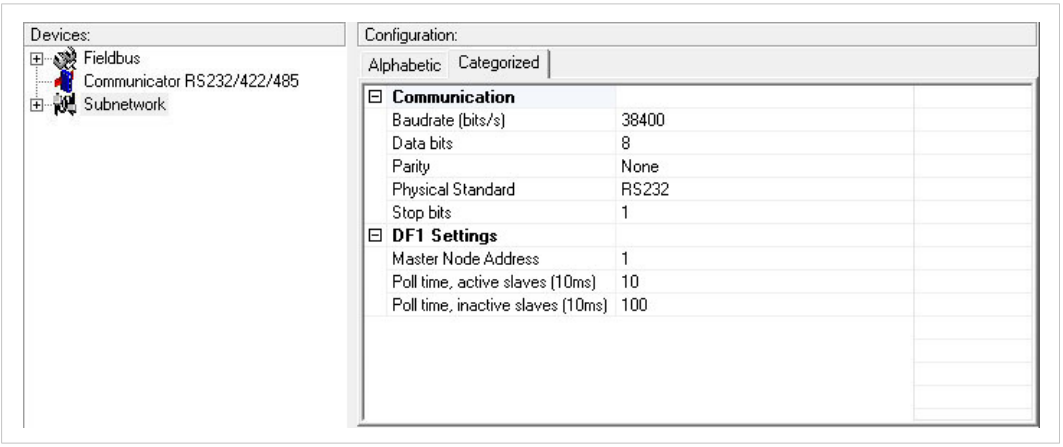


Fig. 20 Subnetwork parameters (DF1 Master mode)

This parameter group is only available in DF1 Master mode.

Parameter	Description	Default value
Master Node Address	Node address of the master. Valid values: 0 to 254	1
Poll time, active slaves (10 ms)	How often active slaves should be polled. Entered in steps of 10 ms	10 (= 100 ms)
Poll time, inactive slaves (10 ms)	How often inactive slaves should be polled. Entered in steps of 10 ms	100 (= 1000 ms)

In the parameter window the poll time value is displayed in steps of 10 ms, which means that the displayed value is a tenth of the actual time in ms.

Incrementing the value by 1 will thus result in a change of 10 ms.

4.2.6 Nodes

Description

A node in Anybus Configuration Manager - Communicator represents a single device on the network. Although the gateway does not feature a scan list in the traditional sense, all nodes and their transactions will be processed in the order they have been defined in the configuration.

A maximum of 31 nodes can be created in Anybus Configuration Manager - Communicator.

Adding and Managing Nodes

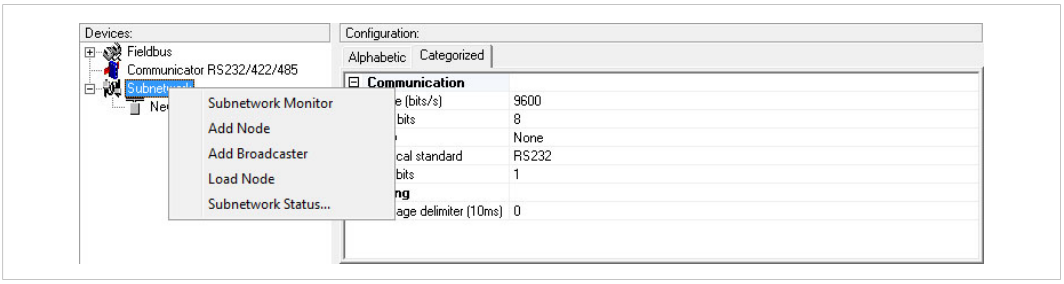


Fig. 21 Adding nodes

Subnetwork Context Menu (right click)	
Menu Command	**Description**
Paste	Paste a node from the clipboard
Subnetwork Monitor	Open the Subnetwork Monitor. See *Subnetwork Monitor, p. 45*.
Add Node	Add a node to the configuration
Add Broadcaster	Add a broadcaster node to the configuration (Master mode only)
Load Node	Add a previously saved node
Subnetwork Status...	View diagnostic information about the subnetwork

Node Parameters

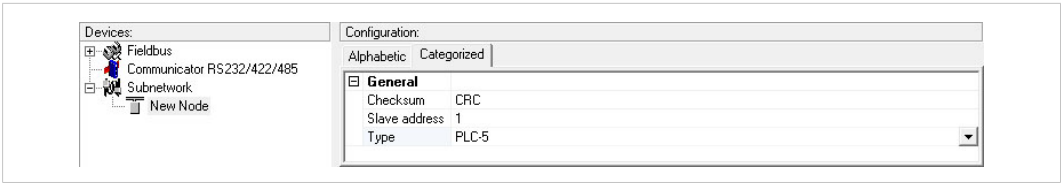


Fig. 22 Node parameters

Master Mode and Generic Data Mode	
Parameter	**Description**
Slave address	This value may be used to set the node address in certain commands. See *Commands (Master Mode & Generic Data Mode), p. 37*.

DF1 Master Mode		
Parameter	**Description**	**Possible values**
Checksum	The type of checksum used	BCC, CRC (default)
Slave address	Sets the node address	0 to 254
Type	The PLC type of the slave	PLC-5, SLC-500, Micrologix

Node Context Menu

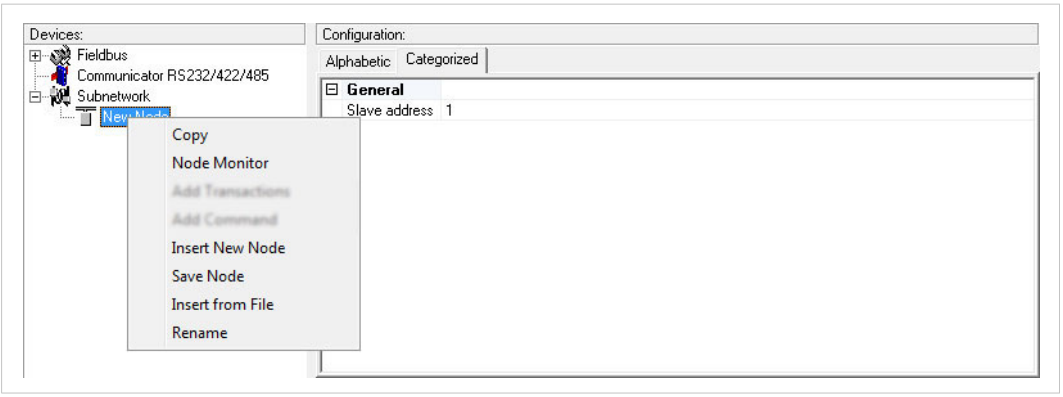


Fig. 23 Node context menu

Right-click on a node to open a context menu with additional commands for managing or adding nodes.

Node Context Menu (right click)	
Menu Command	**Description**
Copy	Copy the selected node to the clipboard
Delete	Delete the selected node (only available when there is more than one node)
Node Monitor	Open the Node Monitor. See *Node Monitor, p. 46*.
Insert New Node	Insert a new node above the selected node
Save Node	Save the selected node
Insert from File	Insert a previously saved node above the selected node
Rename	Rename the selected node

4.2.7 Transactions (Master Mode and Generic Data Mode)

Description

Transactions in Anybus Configuration Manager - Communicator are representations of the actual serial telegrams exchanged on the serial subnetwork. Although the gateway does not feature a scan list in the traditional sense, all nodes and their transactions will be processed in the order they have been defined in the configuration.

Transactions are only available in Master Mode and Generic Data Mode.

- **Master Mode**

  For regular nodes transactions always come in pairs: a *query* and a *response*. The query is issued by the gateway, while responses are issued by the slaves on the subnetwork. The Broadcaster can only send transactions.

- **Generic Data Mode**

  Transactions can be added as desired for both directions. Transactions sent to the subnetwork are called *Transaction Produce*, and transactions issued by other nodes are called *Transaction Consume*.

The gateway can in theory support up to 150 transactions. The actual number of transactions that can be defined depends on their respective memory requirements, and may therefore be significantly less.

Adding and Managing Transactions

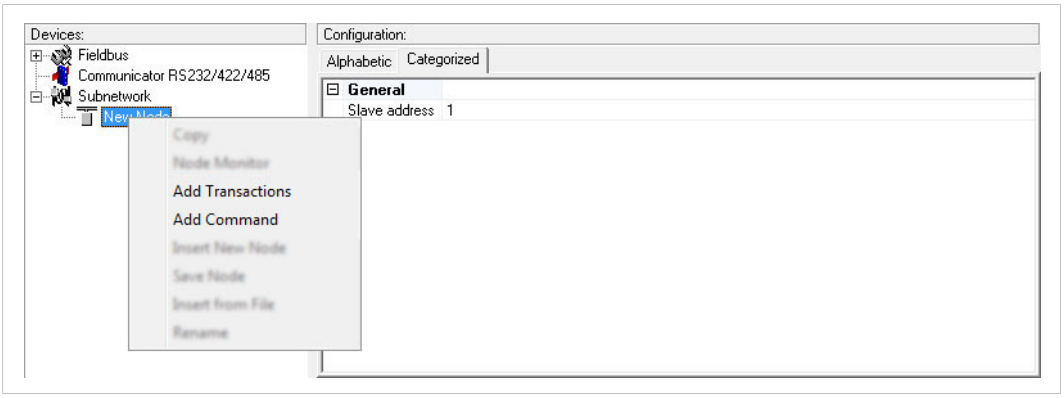


Fig. 24 Node context menu

Right-click on a node to open a context menu with additional commands for adding transactions and commands.

Node Context Menu (right click)	
Menu Command	**Description**
Add Transactions (Master Mode)	On regular nodes: add a Query and a Response (grouped) On the Broadcaster: add a single transaction
Add Transaction Consume (Generic Data Mode)	Add a Consume type transaction
Add Transaction Produce (Generic Data Mode)	Add a Produce type transaction
Add Command	Add a predefined transaction

Query/Broadcast Transactions (Master Mode)

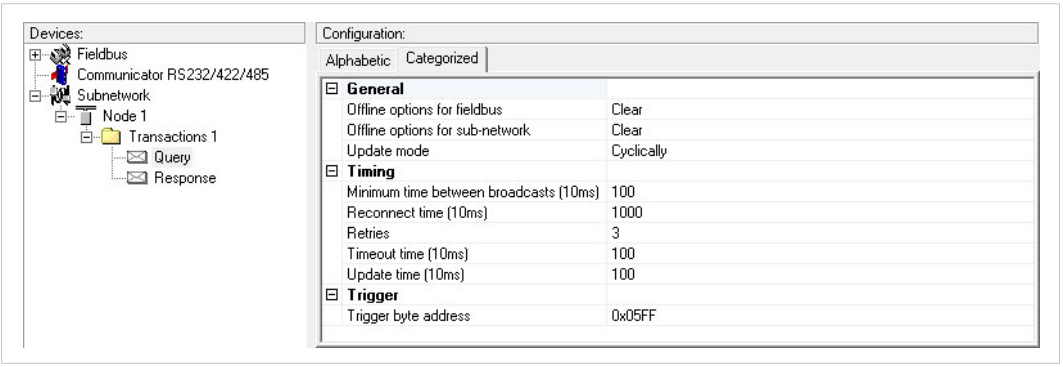


Fig. 25 Transaction parameters

Query/Broadcast Transaction Parameters

Parameter	Description
Minimum time between broadcasts (10 ms)	How long the gateway should wait after transmitting a broadcast transaction before processing the next entry in the scanlist. The value should be set high enough to allow the slave devices time to finish the handling of the broadcast. The entered value is multiplied by 10. A value of 5 means 50 ms. **Note:** This setting is only relevant for the Broadcaster node.
Offline options for fieldbus	The action to take for this transaction if the higher level network goes offline. This affects the data that is sent to the subnetwork. **Clear** = The data destined for the slave devices is cleared (set to zero) **Freeze** = The data destined for the slave device is frozen **NoScanning** = The updating of the subnetwork is stopped
Offline options for subnetwork	The action to take for this transaction if the subnetwork goes offline. This affects the data that is sent to the higher level network. **Clear** = Data is cleared (0) on the higher level network **Freeze** = Data is frozen on the higher level network
Reconnect time (10 ms)	How long the gateway shall wait before attempting to reconnect a disconnected node. A node will be disconnected in case the maximum number of retries (below) has been reached. The entered value is multiplied by 10. A value of 5 means 50 ms. **Note:** This setting is not relevant for the Broadcaster node.
Retries	How many times a timeout may occur in sequence before the node is disconnected.
Timeout time (10 ms)	How long the gateway will wait for a response from a node. If this time is exceeded, the gateway will retransmit the Query until the maximum number of retries (see above) has been reached. The entered value is multiplied by 10. A value of 5 means 50 ms.
Trigger byte address	The location of the trigger byte in the internal memory buffer. Only relevant when *Update mode* is set to *Change of state on trigger.* Valid settings range from 0x200 to 0x3FF and 0x400 to 0xFFF

Query/Broadcast Transaction Parameters (continued)

Parameter	Description
Update mode	Specifies when the transaction shall be sent to the slave. **Cyclically** = The transaction is issued cyclically at the interval specified in the *Update time* parameter. **On data change** = The data area is polled for changes at the time interval defined by the *Update time* parameter. A transaction is issued when a change in data is detected. **Single shot** = The transaction is issued once at start up. **Change of state on trigger** = The transaction is issued when the trigger byte value has changed. This feature enables the control system to notify the gateway when to issue a particular Query. To use this feature correctly, the control system must first update the data area associated with the Query/transaction, then increase the trigger byte by 1. The trigger byte is checked at the interval specified in *Update time*. The location of the trigger byte is specified in *Trigger byte address*.
Update time (10 ms)	How often the transaction will be issued in steps of 10 ms. Relevant only when *Update mode* is set to *Cyclically*, *On data change* or *Change of state on trigger*. The entered value is multiplied by 10. A value of 5 means 50 ms.

Response Transactions (Master Mode)

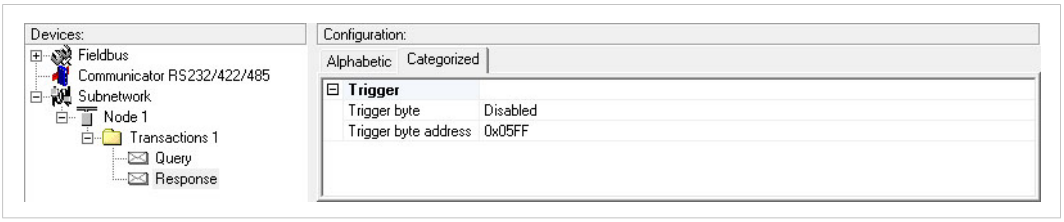


Fig. 26 Parameters

Response Transaction Parameters

Parameter	Description
Trigger byte	Enables/disables the trigger byte function for the response. If enabled, the gateway will increment the trigger byte by 1 when it receives new data from the subnetwork. This can be used to notify the control system of updated data.
Trigger byte address	The location of the trigger byte in the internal memory buffer. Valid settings range from 0x000 to 0x1FF and 0x400 to 0xFFF

Produce Transactions (Generic Data Mode)

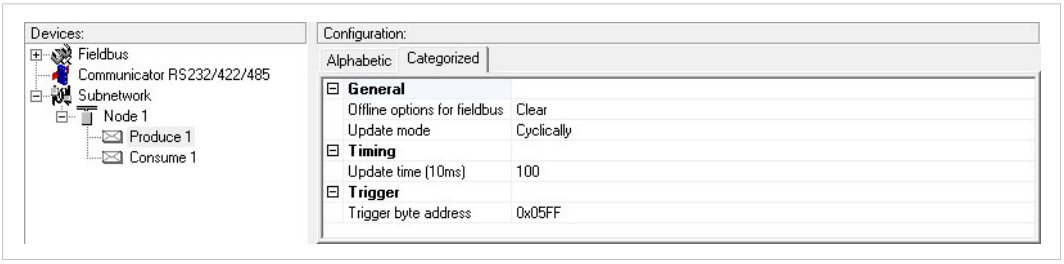


Fig. 27 Parameters

Parameter	Description
Offline options for fieldbus	The action to take for this transaction if the higher level network goes offline. This affects the data that is sent to the subnetwork. **Clear** = The data destined for the slave devices is cleared (set to zero) **Freeze** = The data destined for the slave device is frozen **NoScanning** = The updating of the subnetwork is stopped
Update mode	Specifies when the transaction shall be sent to the slave. **Cyclically** = The transaction is issued cyclically at the interval specified in the *Update time* parameter. **On data change** = The data area is polled for changes at the time interval defined by the *Update time* parameter. A transaction is issued when a change in data is detected. **Single shot** = The transaction is issued once at start up. **Change of state on trigger** = The transaction is issued when the trigger byte value has changed. This feature enables the control system to notify the gateway when to issue a particular Query. To use this feature correctly, the control system must first update the data area associated with the Query/transaction, then increase the trigger byte by 1. The trigger byte is checked at the interval specified in *Update time*. The location of the trigger byte is specified in *Trigger byte address*.
Update time (10 ms)	How often the transaction will be issued in steps of 10 ms. Relevant when *Update mode* is set to *Cyclically*, *On data change* or *Change of state on trigger.* The entered value is multiplied by 10. A value of 5 means 50 ms.
Trigger byte address	The location of the trigger byte in the internal memory buffer. Only relevant when *Update mode* is set to *Change of state on trigger.* The specified memory location is monitored by the gateway. Whenever the trigger byte is updated, the gateway will produce the transaction on the subnetwork. This way, the control system can instruct the gateway to produce a specific transaction on the subnetwork by updating the corresponding trigger byte. The trigger byte address must be unique to each transaction. It can not be shared by multiple transactions. Valid settings range from 0x200 to 0x3FF and 0x400 to 0xFFF

Consume Transactions (Generic Data Mode)

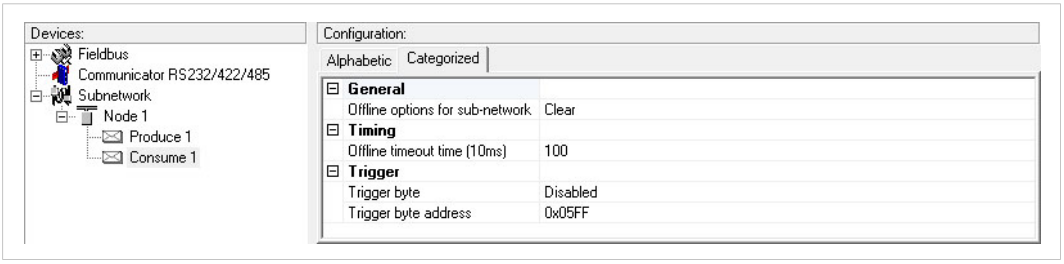


Fig. 28 Parameters

Parameter	Description
Offline options for subnetwork	The action to take for this transaction if the subnetwork goes offline. This affects the data that is sent to the higher level network. **Clear** = Data is cleared (0) on the higher level network **Freeze** = Data is frozen on the higher level network
Offline timeout time (10 ms)	The maximum allowed time in steps of 10 ms between two messages before the subnetwork is considered to be offline. Zero (0) disables the timeout feature. The entered value is multiplied by 10. A value of 5 means 50 ms.
Trigger byte	Enables/disables the trigger byte function. This function can be used to notify the control system of updated data. If enabled, the trigger byte will be incremented each time a valid transaction is consumed by the gateway. The trigger byte will also be incremented if the offline option is set to **Clear** and the offline timeout limit is reached.
Trigger byte address	The location of the trigger byte in the internal memory buffer. The trigger byte address must be unique to each transaction. It can not be shared by multiple transactions. Valid settings range from 0x000 to 0x1FF and 0x400 to 0xFFF

Transaction Editor

The Transaction Editor can be used to edit the individual frame objects of a transaction. The same settings are also available in the parameter section of the main window, however the Transaction Editor presents the frame objects in a more visual manner.

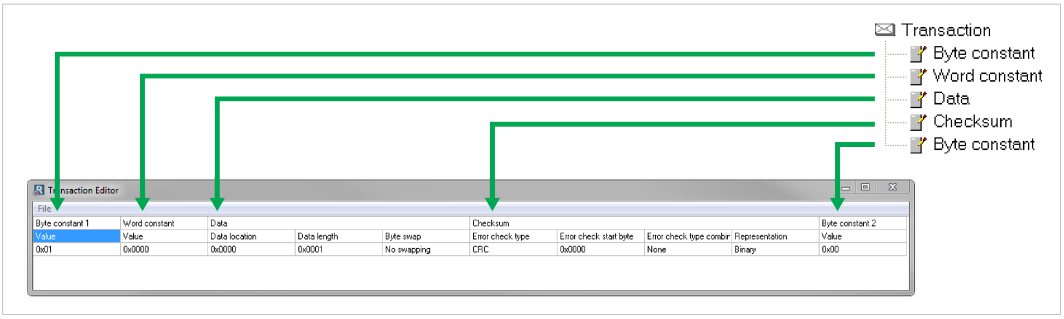


Fig. 29 Transaction Editor

To edit the value of a parameter, click on it and enter a new value, then select **Apply Changes** from the **File** menu in the Transaction Editor. To exit the editor without saving, select **Quit**.

When editing transactions that are based on predefined commands, certain parts of the transaction may not be editable.

4.2.8 Frame Objects (Master Mode & Generic Data Mode)

Description

Each transaction consists of *Frame Objects* which make up the serial telegram frame. Each frame object specifies how the gateway shall interpret or generate a particular part of the telegram.

There are 5 types of frame objects:

- Constant Objects
- Limit Objects
- Data Objects
- Variable Data Objects
- Checksum Objects

Example:

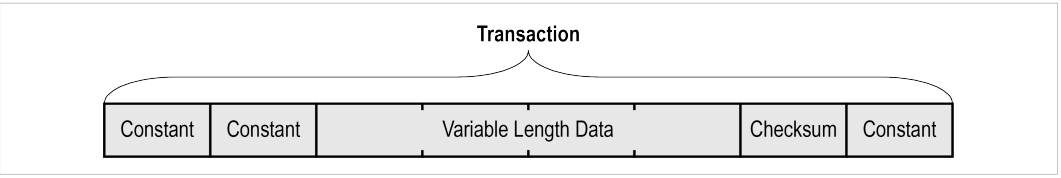


Fig. 30 Frame object example

Adding and Editing Frame Objects

To add a frame object to a transaction, right-click on the transaction and select one of the entries in the context menu.

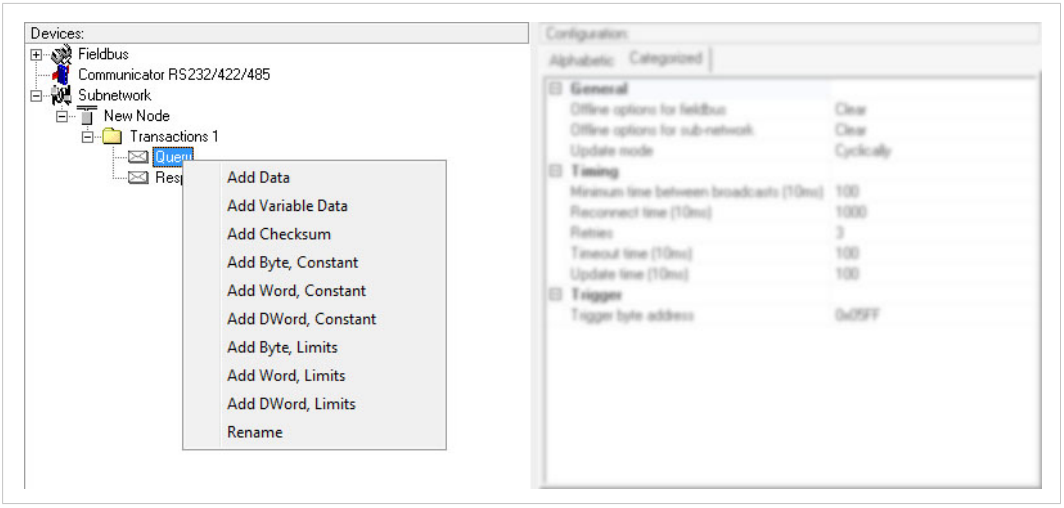


Fig. 31 Adding frame objects

The *Transaction Editor* can also be used to edit transactions and frame objects in a more visual manner. See *Transaction Editor, p. 29*.

Constant Objects (Byte, Word, Dword)

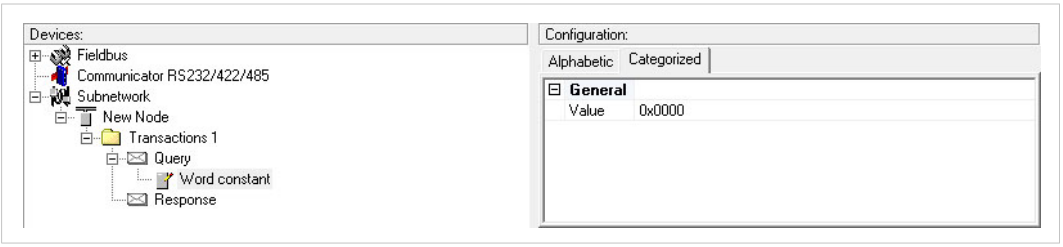


Fig. 32 Constant object

Constant Objects contain a fixed value of a **Byte** (8 bits), **Word** (16 bits) or **DWord** (32 bits).

- **Produce and Query Transactions**

  The gateway will send the value as-is without processing.

- **Consume and Response Transactions**

  The gateway will check if the received value matches the specified value. If not, the message will be discarded.

Constant Object Parameters	
Parameter	**Allowed values**
Value	Byte: 0x00 to 0xFFh Word: 0x0000 to 0xFFFFh DWord: 0x00000000 to 0xFFFFFFFFh

Limit Objects (Byte, Word, Dword)

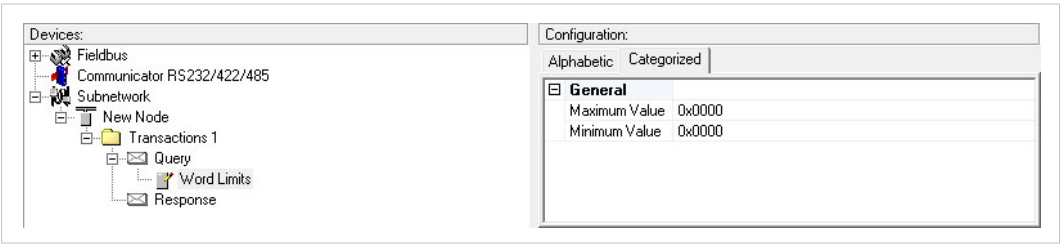


Fig. 33 Limit object

Limit Objects define a value range between a maximum and a minimum value. The two values can be given as a **Byte** (8 bits), **Word** (16 bits) or **DWord** (32 bits).

- **Produce and Query Transactions**

  Not used (the object will be ignored)

- **Consume and Response Transactions**

  The gateway will check if the received value fits inside the specified value interval. If not, the message will be discarded.

Limit Object Parameters	
Parameter	**Allowed values**
Maximum Value	Byte: 0x00 to 0xFFh Word: 0x0000 to 0xFFFFh DWord: 0x00000000 to 0xFFFFFFFFh **Note:** The value must be larger than the Minimum Value.
Minimum Value	Byte: 0x00 to 0xFEh Word: 0x0000 to 0xFFFEh DWord: 0x00000000 to 0xFFFFFFFEh **Note:** The value must be less than the Maximum Value.

Data Object

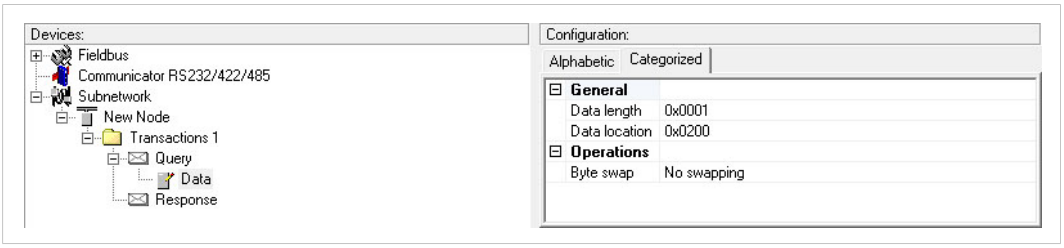


Fig. 34 Data object

Data Objects are used to present raw data.

- **Produce and Query Transactions**

  The specified data block is forwarded from the higher level network to the subnetwork.

- **Consume and Response Transactions**

  The specified data block is forwarded from the subnetwork to the higher level network.

Data Object Parameters	
Parameter	**Description**
Byte Swap	**No swapping** No swapping is performed on the data **Swap 2 bytes** A, B, C, D becomes B, A, D, C **Swap 4 bytes** A, B, C, D becomes D, C, B, A
Data Length	The length of the data block in bytes. In Response/Consume transactions, incoming messages where the data size differs from this value will be discarded. The maximum data length allowed for one frame is 300 bytes.
Data Location	The location of the data block in the internal memory buffer.

Variable Data Object

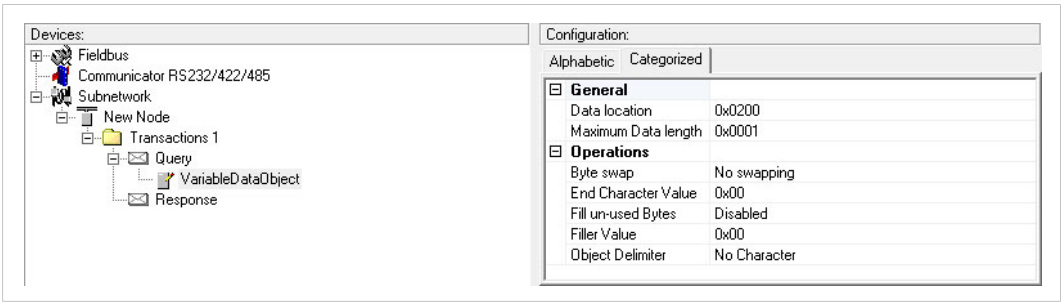


Fig. 35 Variable data object

Each transaction can only contain one variable data object.

This object is similar to the Data Object, except that it has no predefined length. Instead, an End or Length character specifies the size of the data block:

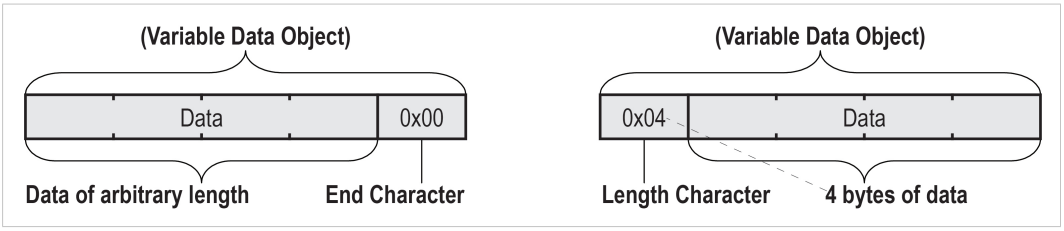


Fig. 36 Example of variable data

- **Produce and Query Transactions**

  The specified data block will be forwarded from the higher level network to the subnetwork. The control system must supply an End or Length character in order for the gateway to know the size of the data block.

  The End or Length Character itself may either be forwarded to the subnetwork or discarded.

- **Consume and Response Transactions**

  The specified data block is forwarded from the subnetwork to the higher level network. The End or Length character will be generated by the gateway automatically (if applicable).

  The End or Length character itself may either be forwarded to the subnetwork or discarded.

Variable Data Object Parameters	
Parameter	**Description**
Byte Swap	**No swapping** No swapping will be performed on the data **Swap 2 bytes** A, B, C, D becomes B, A, D, C **Swap 4 bytes** A, B, C, D becomes D, C, B, A
Fill Unused Bytes	When enabled, unused bytes in Consume/Response transactions will be filled with the value specified in *Filler Value*. This parameter is ignored in Produce/Query transactions.
Filler Value	Filler byte value.
Data Location	The offset in the internal memory buffer that the data shall be read from or written to
Object Delimiter (Produce/Query)	**Length Character** Length character visible in internal memory buffer but *not* sent to the subnetwork **Length Character Visible** Length character visible in internal memory buffer *and* sent to the subnetwork **End Character** End character visible in internal memory buffer but *not* sent to the subnetwork **End Character Visible** End character visible in the internal memory buffer *and* sent to the subnetwork **No Character** No End or Length character generated in the internal memory buffer
Object Delimiter (Consume/Response)	**Length Character** Length character visible in internal memory buffer but *not* received from the subnetwork **Length Character Visible** Length character visible in internal memory buffer *and* received from the subnetwork **End Character** End character visible in internal memory buffer but *not* received from the subnetwork **End Character Visible** End character visible in the internal memory buffer *and* received from the subnetwork **No Character** No End or Length characters included in the received string or generated in the internal memory buffer
End Character Value	End character value
Maximum Data Length	The maximum allowed length (in bytes) of the variable data object. If the actual length of the data exceeds this value, the message will be discarded. The value must not exceed 256 bytes, which is the maximum data length allowed for one frame.

Checksum Object

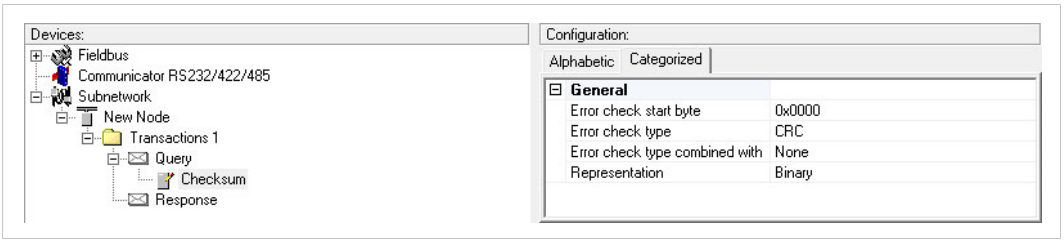


Fig. 37 Checksum object

Most serial protocols features some way of verifying that the data has not been corrupted during transfer. The Checksum Object calculates and includes a checksum in a transaction.

Checksum Object Parameters

Parameter	Description
Error check start byte	Specifies the byte offset in the transaction to start checksum calculations on.
Error check type	This parameter specifies which type of algorithm to use: **CRC** (2 bytes) CRC-16 with 0xA001 polynome (Modbus RTU standard) **LRC** (1 byte) All bytes are added together as unsigned 8-bit values. The two's complement of the result will be used as a checksum. (Modbus ASCII standard with Error Check Start Byte = 0x01 and Representation = ASCII) **XOR** (1 byte) All bytes are logically XOR:ed together. The resulting byte will be used as a checksum. **ADD** (1 byte) All bytes are added together as unsigned 16-bit values. The lowest 8 bits in the result will be used as a checksum.
Error check type combined with	The binary value can be converted to its one's or two's complement. This conversion is carried out before ASCII formatting (see next parameter). **None** The checksum binary value is transmitted without conversion. **One's complement** The checksum value will be converted to its one's complement (inverse code). Example: `00001100` will be transmitted as `11110011` **Two's complement** The checksum value will be converted to its two's complement (complement code). Example: `00001100` will be transmitted as `11110100`
Representation	**Binary** The checksum is transmitted in binary format. **ASCII** All characters in the checksum are converted to ASCII values.

4.2.9 Commands (Master Mode & Generic Data Mode)

Description

Commands are predefined transactions that can be stored and reused. Just like regular transactions, commands consist of frame objects and are representations of the actual serial telegrams exchanged on the serial subnetwork.

Adding a command to a node results in a transaction or transactions being added according to directions specified in the command. The frame objects in such a transaction may retrieve their values not only from parameters in the parameter section, but also from other sources such as the *SlaveAddress* parameter. These parameters cannot be edited directly and will be greyed out in the parameter section of Anybus Configuration Manager - Communicator.

In Master Mode, commands for the most common Modbus RTU functions have been predefined. Additional commands can easily be added using the Command Editor. In Generic Data Mode there are no predefined commands.

Adding and Managing Commands

To add a command to a node, right-click on the node and select **Add Command**.

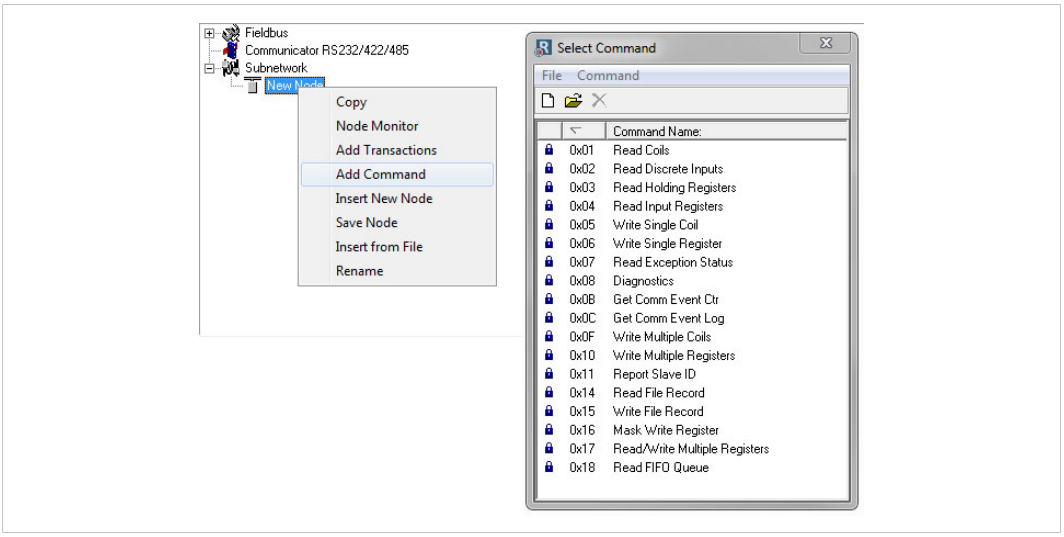


Fig. 38 Adding commands

You can use the **File** and **Command** menus in this window to add, edit and delete custom commands. The predefined commands in Master Mode cannot be edited or deleted.

As with other transactions, the frame objects of each added command may be edited in the Navigation/Parameter Section or using the Transaction Editor. Note however that certain frame objects may be locked for editing.

Command Editor

The Command Editor is used to define new commands and edit existing ones. This makes it possible to build a library of commands, which can be stored and reused at a later stage.

Note that the Command Editor is somewhat protocol-dependent in the sense that certain frame objects may not be deleted or altered.

The examples in this section use Master Mode. The procedures involved are similar in Generic Data Mode, but without the limitations imposed by the Modbus RTU protocol.

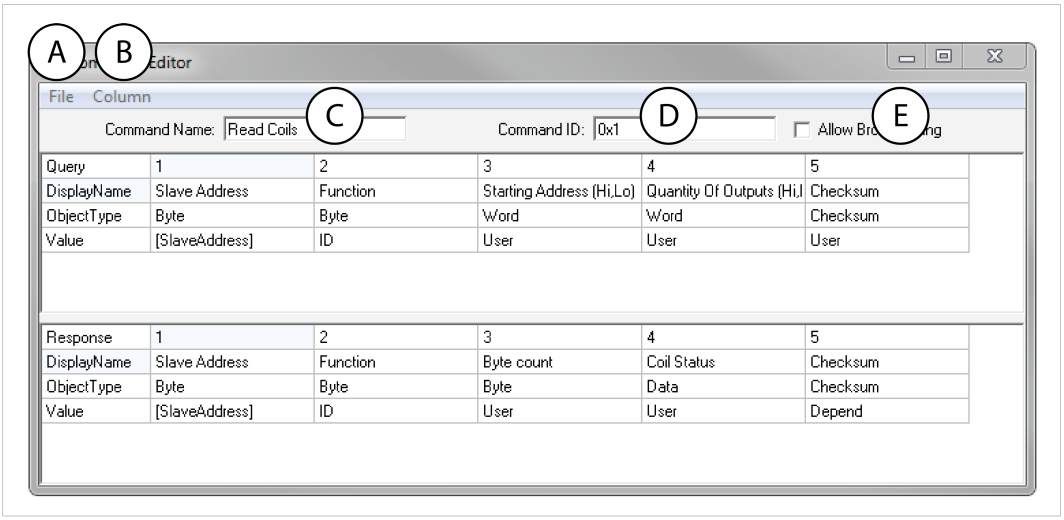


Fig. 39 Command Editor

A. **File menu**

Commands for applying changes or exiting the editor.

B. **Column menu**

Commands for adding and editing columns.

Append Column	Add a new column to the command
Insert Column	Insert a column at the selected position
Delete Column	Delete the column at the selected position

C. **Command Transactions**

The transactions associated with the command. Depending on the protocol mode this can be a query-response pair or a single transaction.

D. **Command ID**

A unique identifier for the command.

E. **Additional settings**

Allow Broadcasting (Master Mode)	Specifies if the command can be broadcasted.
Produce/Consume (Generic Data Mode)	Selects if the command is producing or consuming data.

Editing a Command

The transaction section in the Command Editor represents the transactions associated with the command. Each column represents a frame object within the transaction, and contains the following parameters:

- **Query/Response/Produce/Consume**

  Indicates the direction of the transaction.

- **DisplayName**

  Each column can be named so that the parts of the command are more easily identifiable in the Transaction Editor or in the parameter section of the main window.

- **ObjectType**

  The type of frame object to be used for the column.

- **Value**

  Specifies where the frame object shall retrieve its value.

Value	Description
User	Settings associated with the object can be edited by the user.
ID	Value will be retrieved from the *Command ID* setting in the Command Editor.
[SlaveAddress]	Value will be retrieved from the *SlaveAddress* parameter. See *Nodes, p. 23*.
Depend	Only relevant for Response transactions in Master Mode. The value will be retrieved from the corresponding part of the Query transaction.

Example: Modbus RTU Command in Master Mode

A Modbus RTU transaction always contains the following parts:

- Slave Address (1 byte)
- Function Code (1 byte)
- Data
- Checksum (CRC-16)

The command consists of a Query and a Response. The required Modbus RTU specific frame objects are automatically added, with a data object inserted between the function code and the checksum. These objects cannot be moved or deleted, but additional objects can be added between the function code and the checksum object.

Enter a descriptive name for the command in the *Command Name* field and a suitable function code in the *Command ID* field. If the command is allowed to be broadcasted, check the *Allow Broadcasting* checkbox.

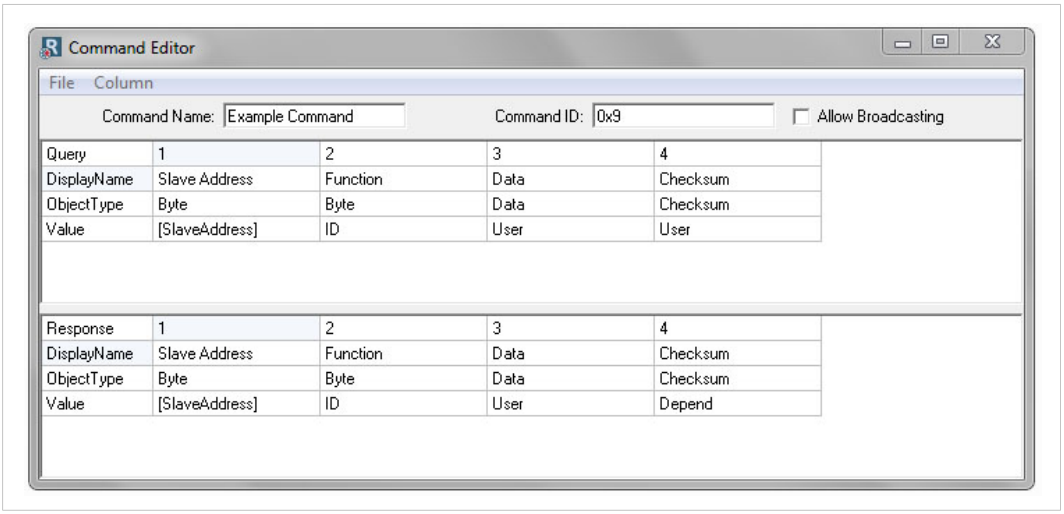


Fig. 40 Example - Modbus RTU Command

Query	**1**	**2**	**3**	**4**
DisplayName	Slave Address	Function	Data	Checksum
ObjectType	Byte	Byte	Data	Checksum
Value	[SlaveAddress]	ID	User	User
	Linked to the *SlaveAddress* parameter.	Retrieved from the *CommandID* field in the Command Editor.	Size and location of this object is set by the user.	Checksum type and parameters can be selected by the user. Default = CRC-16.

Response	**1**	**2**	**3**	**4**
DisplayName	Slave Address	Function	Data	Checksum
ObjectType	Byte	Byte	Data	Checksum
Value	[SlaveAddress]	ID	User	User
	Linked to the *SlaveAddress* parameter.	Retrieved from the *CommandID* field in the Command Editor.	Size and location of this object is set by the user.	Retrieved from the corresponding object in the Query transaction.

4.2.10 Services (DF1 Master Mode)

Description

Services are commands that can be stored and reused. The user configures each slave with services that can be issued from the master. When the Anybus Communicator is going to execute a service, it automatically chooses the appropriate DF1 command(s) used to perform the service on the selected DF1 node type.

A maximum of 50 services are allowed.

Predefined Services

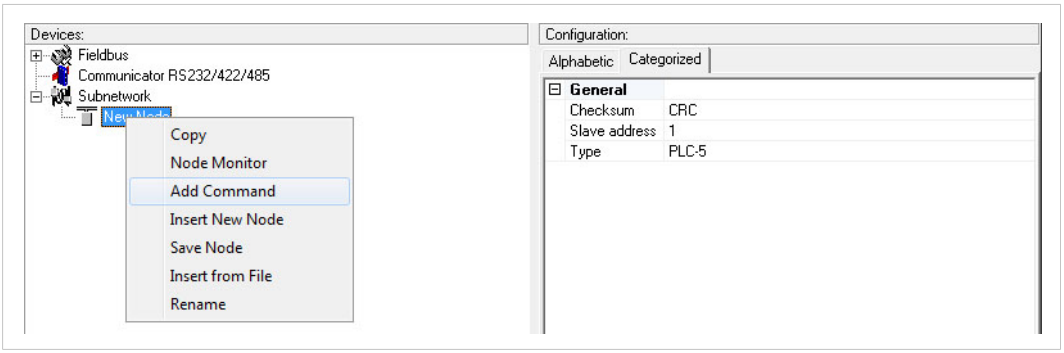


Fig. 41 Adding a service

Right-click on the node and choose **Add Command**. Four different services will be available:

- Integrity Check
- Read Diagnostics
- Read Data
- Write Data

The parameters of the predefined services can be configured to suit the application.

Common Parameters

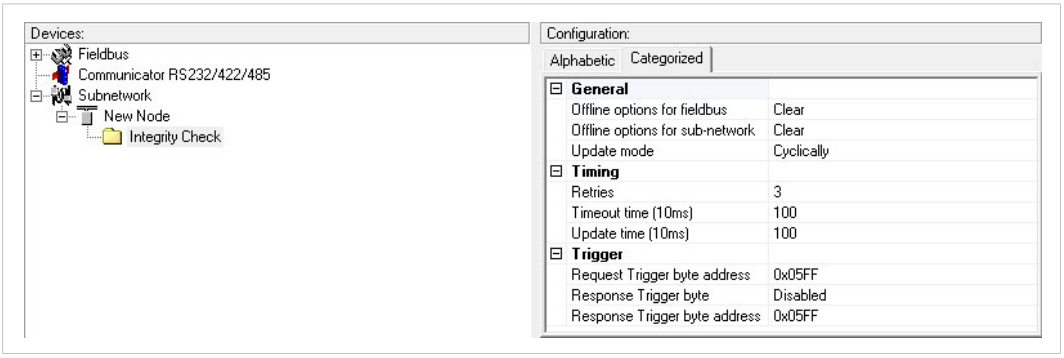


Fig. 42 Common parameters

These parameters are common to all services, but the settings are individual to each instance of a service.

General		
Parameter	**Description**	**Valid settings**
Offline options for fieldbus	The action to take for this service if the fieldbus goes offline. This option affects the data that is sent out to the subnetwork.	Clear Freeze Noscanning
Offline options for subnetwork	The action to take for this service if the subnetwork goes offline. This option affects the data that is reported to the fieldbus master.	Clear Freeze
Update mode	The update mode for this service	Cyclically On data change Single shot Change of state on trigger

Timing		
Parameter	**Description**	**Default**
Retries	The number of times to resend this service before the node is disconnected	3
Timeout time (10 ms)	The time to wait before resending this service (in steps of 10 ms)	100 (= 1000 ms)
Update time (10 ms)	The minimum time between two services of this kind (in steps of 10 ms)	100 (= 1000 ms)

Trigger		
Parameter	**Description**	**Default**
Request Trigger byte address	The memory location of the trigger byte this service uses for updates on trigger byte changes	0x05FF
Response Trigger byte	Enables/disables the trigger byte	Disabled
Response Trigger byte address	The memory location of the trigger byte this service uses for updates on trigger byte changes Valid settings range from 0x200 to 0x3FF and 0x400 to 0xFFF	0x05FF

Integrity Check

This service checks that a node is up and running correctly. A telegram is sent to the node, and the node mirrors and returns the telegram.

This service contains no configuration apart from the common parameters.

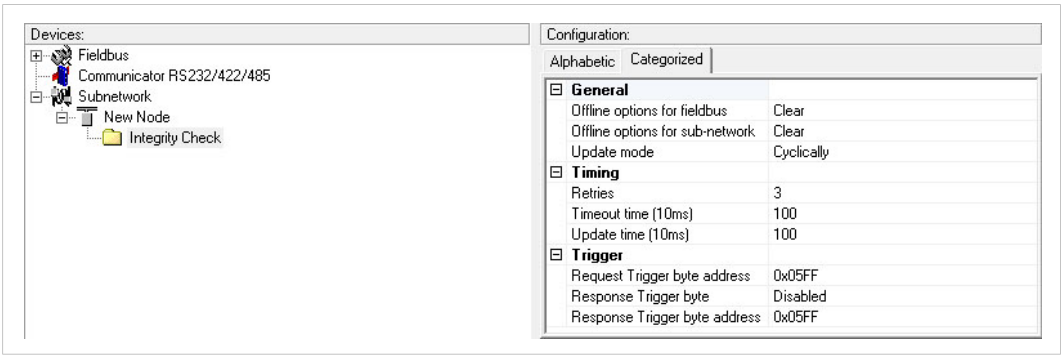


Fig. 43 Integrity Check service

Read Diagnostics

This service reads diagnostic information from the gateway.

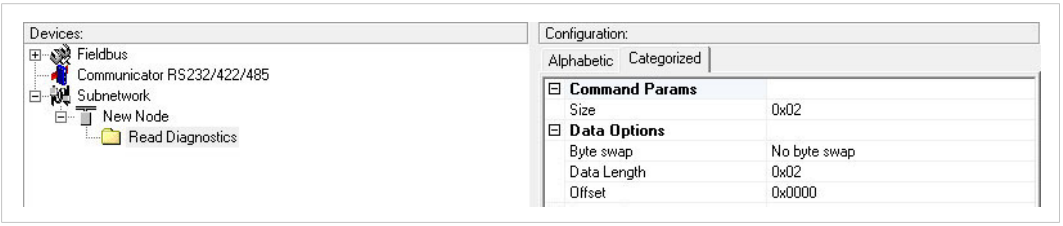


Fig. 44 Read Diagnostics service

Command Parameters		
Parameter	**Description**	**Valid settings**
Size	The number of bytes that can be read from the slave. One bit/integer element is 2 bytes and one float element is 4 bytes. The parameter must have an even value as only whole elements can be read.	PLC-5: 1–26 SLC-500: 1–28 MicroLogix: 1–26

Data Options		
Parameter	**Description**	**Valid settings**
Byte swap	Determines if the data shall be swapped	No byte swap Swap words Swap double words
Data length	The number of bytes, read from the DF1 network, to write to the area determined in *Offset*.	Less or equal to *Size*
Offset	The offset in the internal memory buffer that the data shall be read from.	-

Read Data

This service is used to read data from the nodes in the subnetwork.

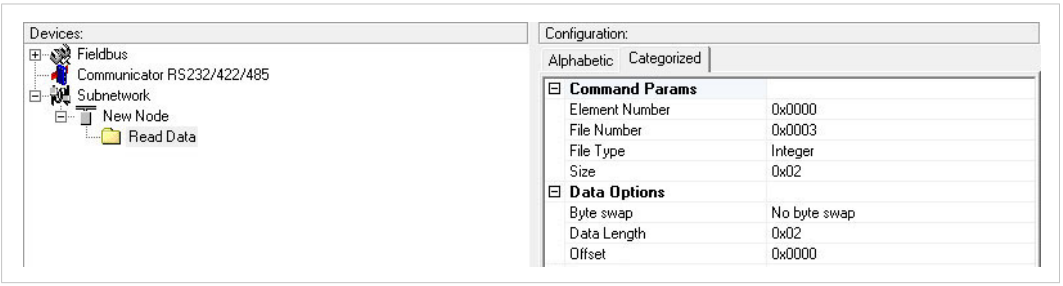


Fig. 45 Read Data service

Command Parameters		
Parameter	**Description**	**Valid settings**
Element Number	The element number of the data file to be accessed within the slave.	PLC-5: 0–999 SLC-500: 0–255 MicroLogix: 0–255
File Number	The file number of the data file to be accessed.	PLC-5: 3, 7, 8, 10–999 SLC-500: 3, 7, 8, 10–255 MicroLogix: 3, 7, 8, 10–255
File Type	The file type of the data to be accessed.	Integer Bit Float
Size	The number of bytes to read from the slave. One bit/integer element is 2 bytes and one float element is 4 bytes. The parameter must have an even value as only whole elements can be read.	PLC-5: 2–240 SLC-500: 2–236 MicroLogix: 2–242

Data Options		
Parameter	**Description**	**Valid settings**
Byte Swap	Determines if the data shall be swapped	No byte swap Swap words Swap double words
Data Length	The number of bytes, read from the DF1 network, to write to the area determined in *Offset*.	Less or equal to *Size*
Offset	The offset in the internal memory buffer that the data shall be read from. **Note:** If the Control and Status registers are enabled (default) the first available data address will be 0x002 in the Input area, and 0x202 in the Output area.	-

Write Data

This service is used to write data to the nodes in the subnetwork. The parameters are the same as for the service *Read Data*. The only difference is that data is read from the internal memory buffer in the gateway and written to the subnetwork bus, instead of vice versa.

4.2.11 Subnetwork Monitor

Description

The Subnetwork Monitor is intended to simplify configuration and troubleshooting of the subnetwork. Its main function is to display the data allocated for subnetwork communication and detect if any area has been allocated twice (address collision).

> ! Using the Subnetwork Monitor may temporarily reduce gateway performance.

Operation

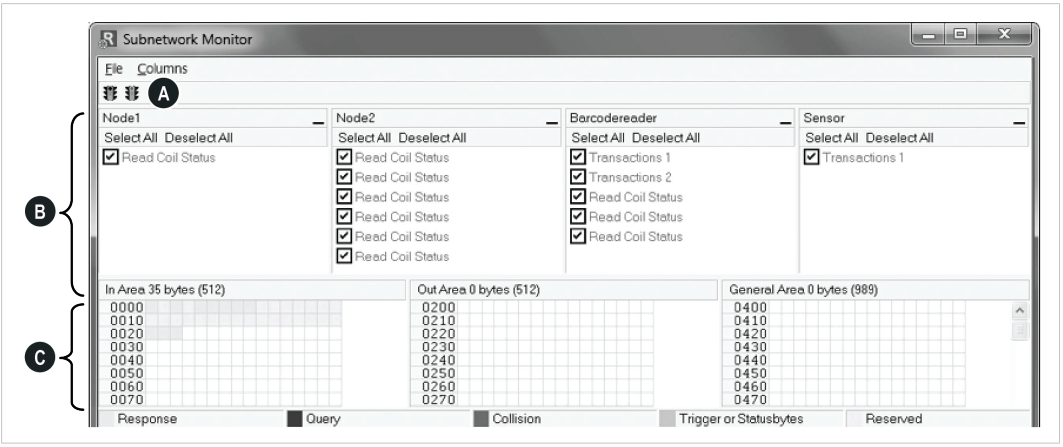


Fig. 46 Subnetwork Monitor

A: Menus and Toolbar Buttons

File	Start Network	Start subnetwork activity	
	Stop Network	Stop all subnetwork activity	
	Exit	Close the Subnetwork Monitor	
Columns	Free	Number of columns depends on window width	
	8 Multiple	Number of columns will be fixed to 8	

B: Nodes/Transactions Section

Lists all the configured nodes and their transactions. To view data blocks associated with a transaction, select the transaction in the list. The corresponding data will then appear in the Monitor Section.

C: Monitor Section

Visualizes how data is allocated in the Input, Output and General Data areas.

Color	Meaning
White	Not allocated
Yellow	Data allocated by a Response or Consume transaction
Blue	Data allocated by a Query or Produce transaction
Red	Collision – area has been allocated more than once
Grey	Reserved (illustrates memory consumption, area can be allocated if necessary)
Green	Data allocated by Trigger byte, Transmit/Receive Counter, or Control/Status Registers

4.2.12 Node Monitor

Description

The Node Monitor can provide valuable information when setting up the communication with the subnetwork, by allowing individual commands to be issued manually, and monitoring the response (if applicable). It also provides an overview of the memory used by a particular node. The behavior of the Node Monitor differs slightly depending on the selected protocol mode.

> **!** Using the Node Monitor may temporarily reduce gateway performance.

Master Mode and DF1 Master Mode

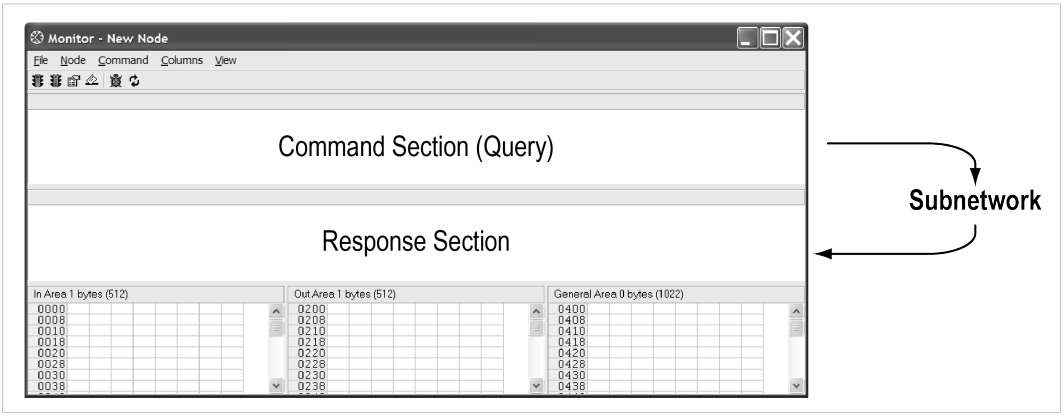


Fig. 47 Node Monitor – Master Mode

The selected Command (Query Transaction) or Service is sent to the subnetwork. The response to the Query can be monitored in the Response Section.

Generic Data Mode

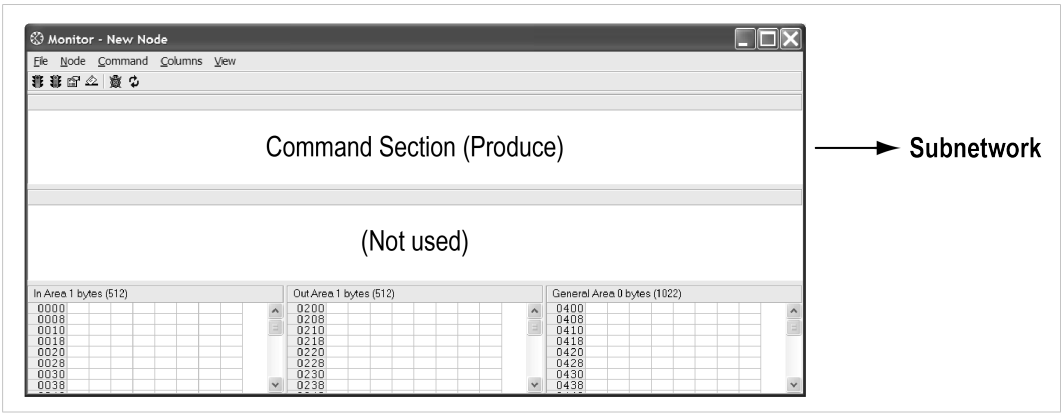


Fig. 48 Node Monitor – Generic Data Mode

The selected command (Produce Transaction) is sent to the subnetwork. It is not possible to monitor responses or other activity generated by other nodes.

Operation

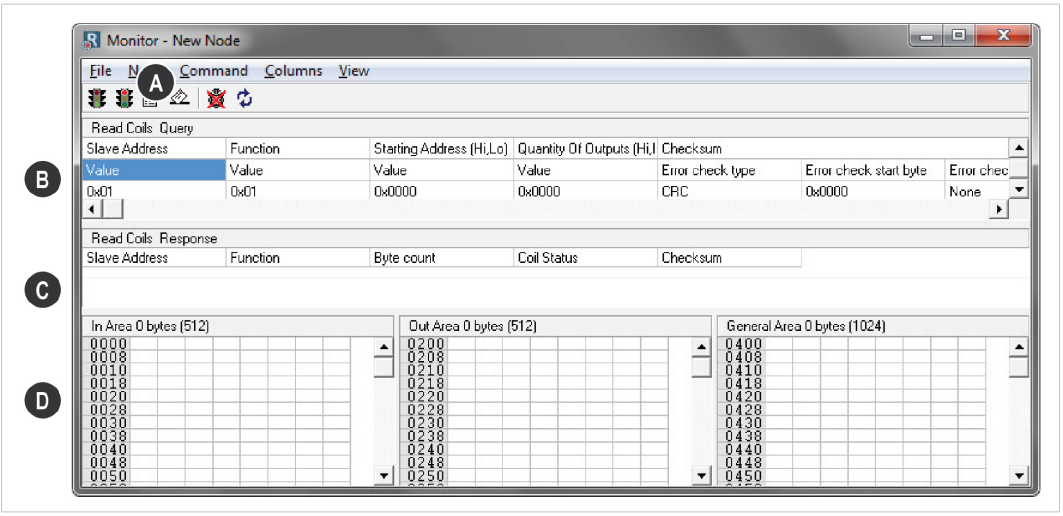


Fig. 49 Node Monitor

A: Menus and Toolbar Buttons

Menu	Command	Description	Button
File	Exit	Closes the Node Monitor	
Node	Start Node	Enable all transactions associated with the node	
	Stop Node	Disable all transactions associated with the node	
Command	Select	Select a command to be sent to the subnetwork	
	Send	Send the selected command to the subnetwork	
Columns	Free	Number of columns depends on window width	
	8 Multiple	Number of columns will be fixed to 8	
View	Hex	Display the data in hexadecimal format	
	Decimal	Display the data in decimal format	
-	-	Stop/Resume refreshing	
	-	Refresh the displayed data	

B: Command Section

This section holds the currently selected command. The individual frame objects in the command can be edited in a similar way as in the Transaction and Command Editors.

C: Response Section (Master Mode and DF1 Master Mode)

This section holds the response to the selected Command.

D: Monitor Section

Displays the data associated with the node. Areas in dark grey are reserved for the Status and Control Registers, and areas displayed in light grey represent data used by the node.

Click on the Refresh icon in the toolbar to update the data.

> **!** The displayed data will not be cleared if the higher level network goes offline.

4.2.13 Data Logger

Description

The subnetwork traffic can be logged into a buffer. This may provide valuable information when debugging the lowest levels of the subnetwork communication.

The logger is built into the gateway and is separate from Anybus Configuration Manager - Communicator. This means that logging can be performed even if the gateway is disconnected from the computer.

The log buffer will hold 512 bytes of data in each direction by default. The size of the log buffer can be changed from the **Options** dialog. See also *Options Dialog, p. 18*.

The Data Logger is not available until a communication port has been selected.

Operation

Select **Start logging** from the **Tools** menu, then choose the operation mode:

- **Log until full**

  Data will only be logged until the log buffer is full.

- **Log continuously**

  Data will be logged continuously until you select **Stop Logging**. The log buffer will contain the most recent data.

In both operation modes, selecting **Stop Logging** will stop the logging and open the log window.

Log Window

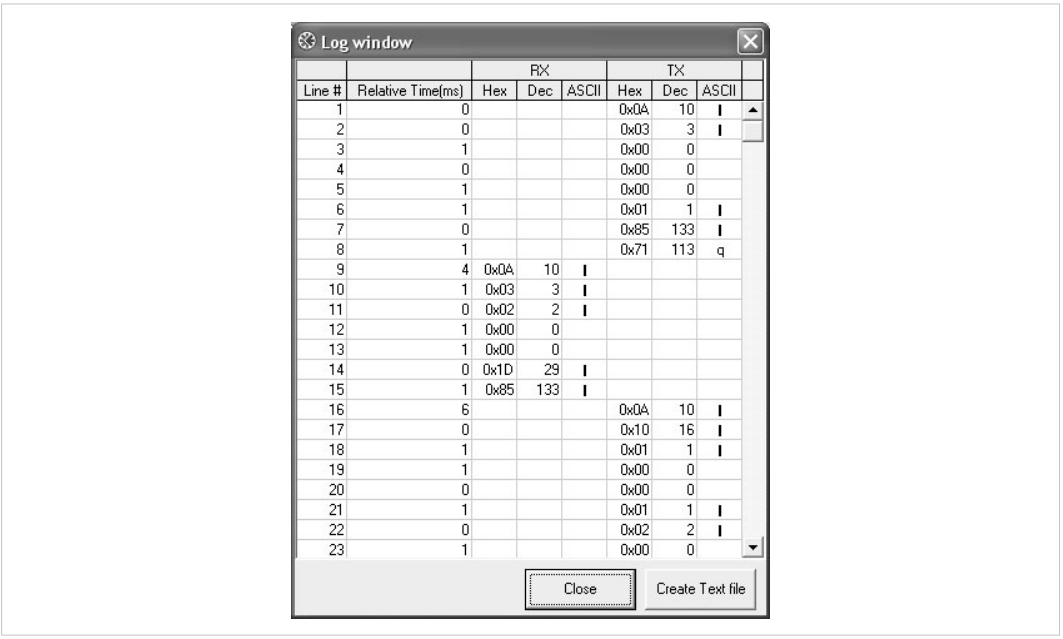


Fig. 50 Log window

The logged data is displayed in hexadecimal, decimal and ASCII format for both directions. The time between the log entries is displayed in a separate column.

Click **Create Text file** to save the data in ASCII text format.

Click **Close** to exit the log window.

4.3 Basic TCP/IP Concepts

IP Address

The IP address is used to identify each node on a TCP/IP network. IP addresses are written as four decimal integers (0–255) separated by dots, where each integer represents the binary value of one byte of the IP address. This is known as *dot-decimal notation*.

Example: `10000000 00001010 00000010 00011110` is written as `128.10.2.30`

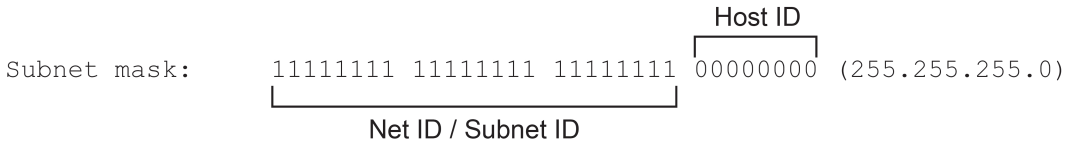
The following IP addresses are reserved for special purposes and cannot be used:

0.n.n.n	First byte zero — used for broadcast messages
127.n.n.n	First byte 127 — used for loopback addresses to the local host
n.n.n.0	Last byte zero — identifies a whole network/subnet
n.n.n.255	Last byte 255 — used for broadcast messages

Subnet Mask

The IP address is divided into three parts: *Net ID*, *Subnet ID* and *Host ID*. A subnet mask is a 32-bit binary pattern, where a set bit allocates a bit for Network/Subnet ID, and a cleared bit allocates a bit for the Host ID. The subnet mask is usually written in dot-decimal notation.

Example: To make the IP address `128.10.2.30` belong to subnet `128.10.2`, the subnet mask must be `255.255.255.0`.



Default Gateway

For devices to be able to communicate over Ethernet they must either belong to the same subnet or communicate via a gateway or router.

A gateway or router routes communication between networks, i.e. it enables the nodes on one network to access the nodes on another. The *default gateway* address in the TCP/IP settings of your product specifies the IP address of the gateway or router on the local network.

4.4 TCP/IP Configuration

4.4.1 Installing the IPconfig Utility

IPconfig is a Windows-based tool for configuration of TCP/IP settings in HMS devices. The tool will detect all compatible and active HMS devices on the local network.

1. Download IPconfig from www.anybus.com/support.
2. Unpack the contents of the zip archive and run the installer program.

4.4.2 Scanning for Connected Devices

When IPconfig is started it will automatically scan all available local networks for HMS devices. Detected devices will be listed in the main window. To refresh the list, click on **Scan**.

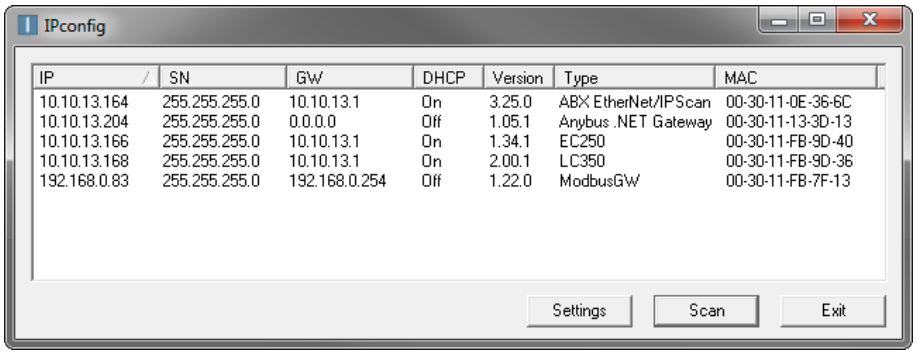


Fig. 51 IPconfig main window

IP	IP address of the device
SN	Subnet mask
GW	Default gateway
DHCP	Automatically managed IP configuration
Version	Firmware version
Type	Product name
MAC	Ethernet MAC address (System ID)

4.4.3 Ethernet Configuration

To change the IP settings for a device, double-click on the entry in the main window or right-click on it and select **Configuration**.

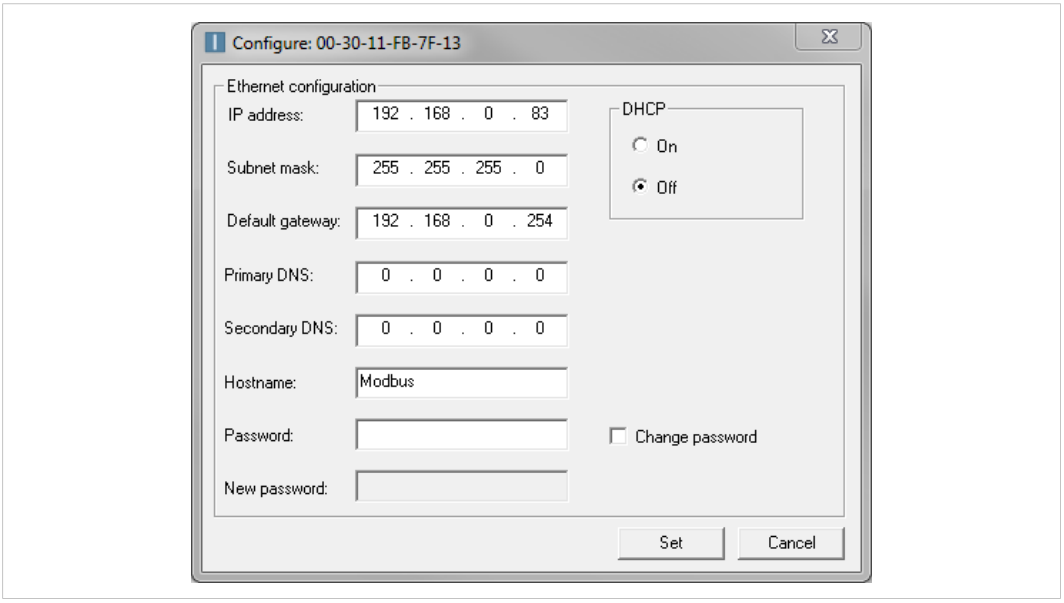


Fig. 52 Ethernet configuration

Enter static IP settings as required, or select DHCP if using dynamic IP addressing.

> Do not enable DHCP if there is no DHCP server available on the network.

You can add a name for the device in the **Hostname** field. Only characters a–z, A–Z, 0–9 and _ (underscore) are allowed.

The default password for changing IP settings is blank (no password). If a password has been set for the device you must enter it to be able to change the settings.

To set a new password, check the **Change password** box and enter the current password in the **Password** field, then enter the new password in the **New password** field.

> For security reasons the default password should always be changed.

Click on **Set** to save the new settings. The device will reboot automatically.

4.4.4 IPconfig Settings

Additional settings for IPconfig can be accessed by clicking on **Settings**.

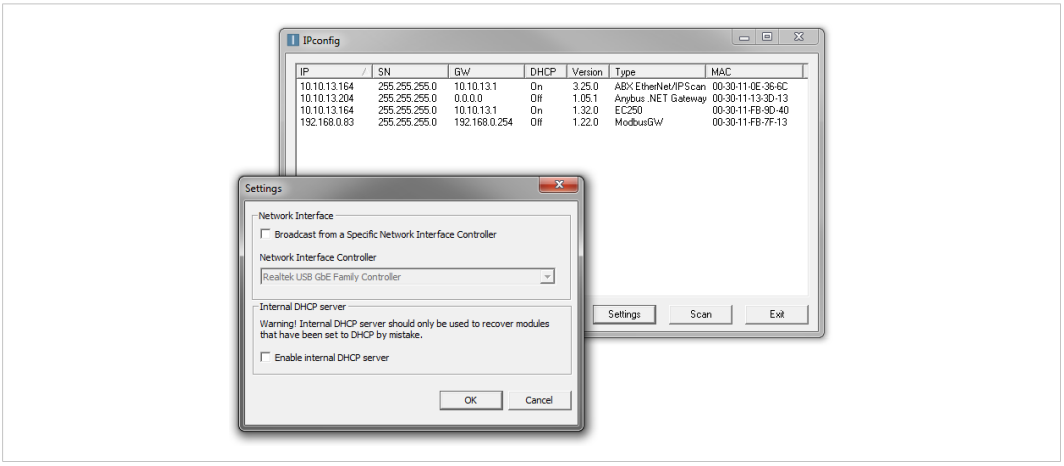


Fig. 53 IPconfig settings

Network Interface

Check this option to select a specific network interface to use when scanning for devices from a computer which has more than one interface. If this option is left unchecked, all available networks will be scanned.

Internal DHCP Server

If a device has been set to use DHCP but there is no DHCP server on the network, the device may not be detected by IPconfig. To recover access to the device an internal DHCP server in IPconfig can be temporarily activated:

1. Click the checkbox for **Internal DHCP Server**, then click **OK**. IPconfig will automatically refresh the scan and list the missing device in the main window.
2. Select the device and configure it to use static IP addressing instead of DHCP.
3. Disable the internal DHCP server.

> **!** Do not enable the internal DHCP server if there is already an active DHCP server on the network.

4.5 Anybus Configuration Manager - IIoT

The Anybus Communicator is configured using *Anybus Configuration Manager - IIoT*, which can be downloaded from www.anybus.com/support.

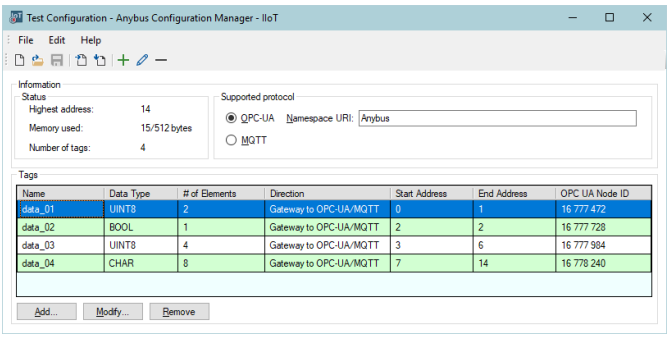


Fig. 54 Anybus Configuration Manager - IIoT

Setting up the IIoT tags

1. Select either OPC UA or MQTT as communication protocol.

   For OPC UA you must also specify a **Namespace URI**. If this field is left empty a default value will be used.

2. Click on **Add** to add IIoT tags as required. The data types and addresses of the tags must match the IO data mapping set up with Anybus Configuration Manager - Communicator.

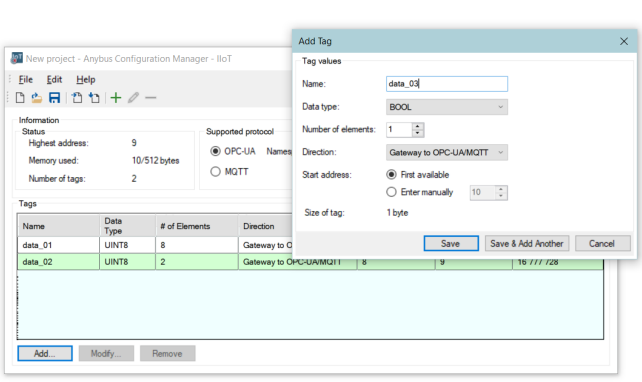


Fig. 55 Adding tags

3. Optionally, select **File ▸ Save** to save the current configuration project.

Note: The project file saves the current settings in Anybus Configuration Manager - IIoT and can be used as a starting point when creating configurations. It cannot be used as a configuration file by the Anybus Communicator IIoT.

When you have finished setting up the IIoT tags you must generate a configuration and apply it to the Anybus Communicator IIoT. This can be done directly from Anybus Configuration Manager - IIoT or by saving a configuration file and using an external FTP client.

Applying the configuration from Anybus Configuration Manager - IIoT

1. Select **File ► Generate and send...**.

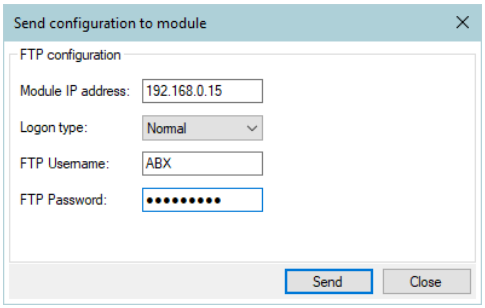


Fig. 56 Sending the configuration

2. Enter the IP address and the FTP username and password for the Anybus Communicator.

   The default username is **ABX** and the password is **FTPAccess**.

3. Click on **Send** to transfer the configuration to the Anybus Communicator.

   When the transfer is finished, click on **OK** to close the confirmation dialog.

   The X-gateway will apply the new configuration and restart automatically.

Generating a configuration file and applying it using an external FTP client

1. Select **File ► Generate configuration...**.
2. Save the generated configuration file to your computer.
3. Open an FTP client and connect to the IP address of the Anybus Communicator.

   The default username is **ABX** and the password is **FTPAccess**.

4. Transfer the saved configuration file from your computer to the Anybus Communicator.
5. Power cycle the X-gateway to apply the new configuration.

4.6 Web Interface

The web interface of the Anybus Communicator IIoT provides additional configuration settings and status information. The web interface is accessed by entering the IP address of the Anybus Communicator IIoT in a web browser. The computer used for access must be in the same Ethernet subnet as the Anybus Communicator.

Overview

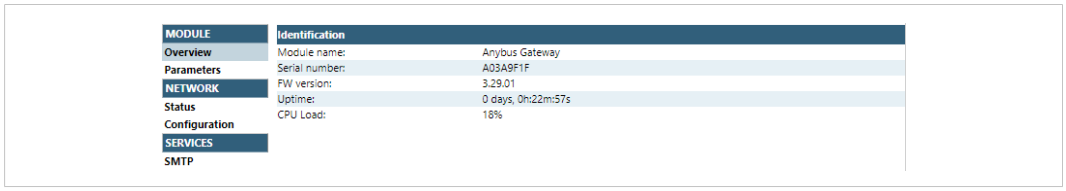


Fig. 57 Overview page

This page provides basic information about the Anybus Communicator including the serial number and the installed firmware version.

Parameters

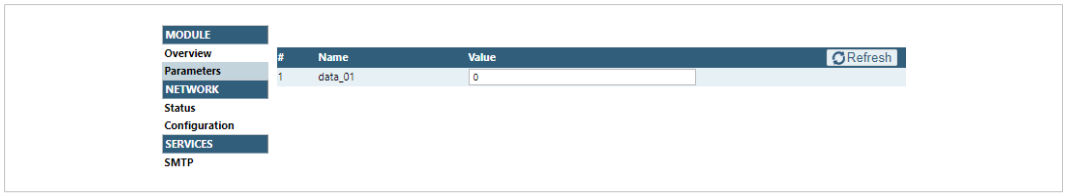


Fig. 58 Parameter page

This page shows the current data values for the configured IIoT tags.

Network Status

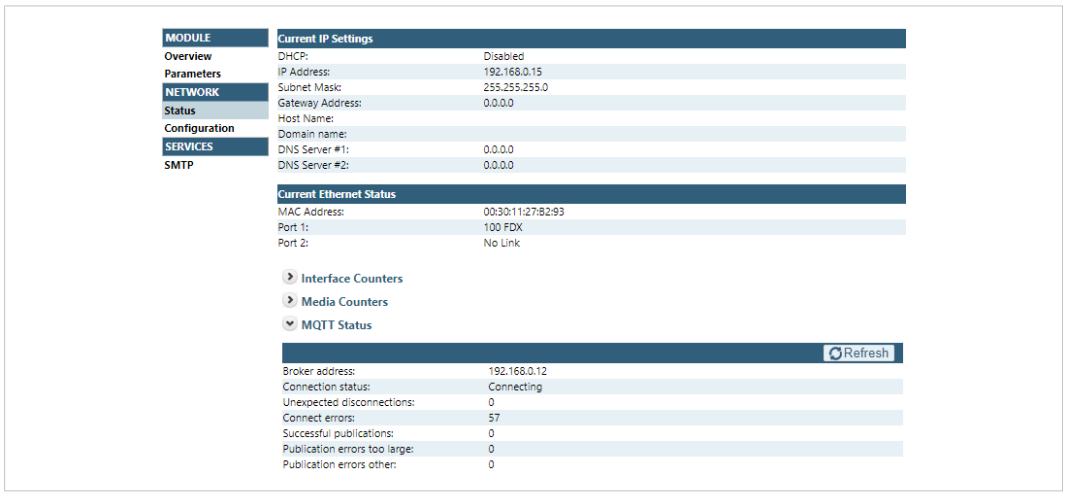


Fig. 59 Status page

This page shows the current network settings and communication status. This information is mainly intended for troubleshooting.

Network Configuration

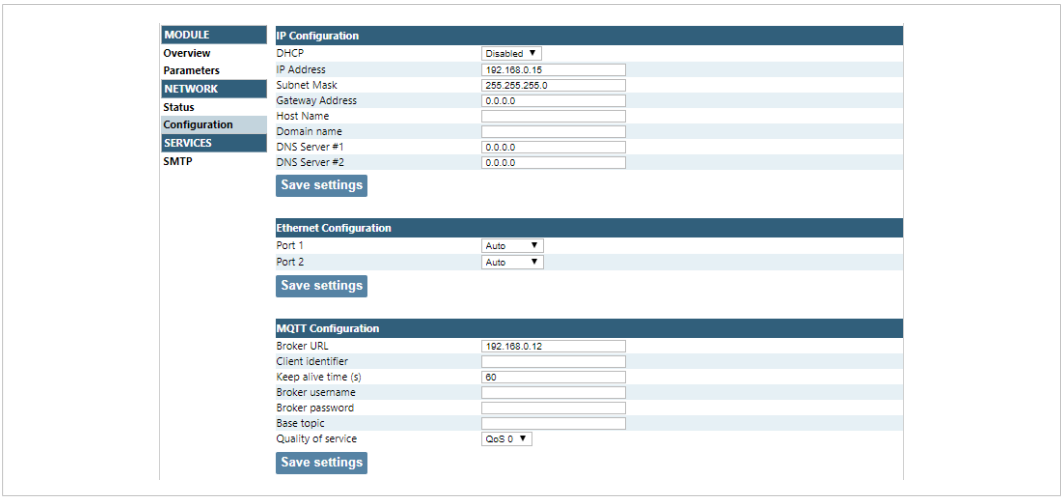


Fig. 60 Configuration page

IP Configuration

TCP/IP network settings. These parameters can also be configured using *IPconfig*.

Ethernet Configuration

Port 1/2

10 HDX = 10 Mbit/s half duplex

10 FDX = 10 Mbit/s full duplex

100 HDX = 100 Mbit/s half duplex

100 FDX = 10 Mbit/s full duplex

Auto = Automatic mode (default)

MQTT Configuration

Broker URL The URL of the MQTT broker.

Client identifier The MQTT ClientID of the Anybus Communicator IIoT.

Keep alive time (s) The time interval in seconds between the keep alive messages sent from the client to the broker.

Broker username/password Authentication to send to the MQTT broker (if required).

Base topic The base topic name to use in MQTT messages.

Quality of service The MQTT QoS level to use.

OPC UA Configuration

TCP port The TCP port to use for OPC UA. Default = 4840.

Discovery server URL The URL of the OPC UA discovery server (required for timestamp information).

A Technical Data

A.1 General Specifications

Model name	Anybus Communicator IIoT
Order code	AB7079-B
Dimensions (L x W x H)	120 x 75 x 27 mm
Weight	150 g
Operating temperature	0 to +55 °C (IEC 60068-2-1 and IEC 60068-2-2)
Storage temperature	-40 to +85 °C (IEC 60068-2-1 and IEC 60068-2-2)
Humidity range	5–95 % RH, non-condensing (IEC 60068-2-30)
Power supply	24 V ±10 % DC regulated power source
Current consumption	Typical: 100 mA @ 24 VDC Maximum: 200 mA @ 24 VDC
Galvanic isolation	Yes, on both network sides
Mechanical rating	IP20, NEMA rating 1
Mounting	DIN rail (EN 50022) Network shield conductance via DIN rail
Certifications	See datasheet at www.anybus.com/support

A.2 Serial Interface

Serial application Interface	Selectable RS-232, RS-422, RS-485
Maximum number of stations	31 nodes via RS-422 or RS-485
Protocol: Modbus RTU	Modbus RTU Master - Query/Response
Protocol: ASCII/Vendor Specific	Request/Response or Produce/Consume
Protocol: Rockwell DF1	DF1 Master

A.3 IIoT Interface

OPC UA functionality

- Support for micro-embedded profile
- Supports Discovery Services
- Timestamp supported via discovery server
- User name and password authentication
- Supports DataChange Subscription
- Maximum 80 data point tags

  (max. 2 clients with up to 40 tags per client)

MQTT functionality

- MQTT client acting as publisher
- MQTT version 3.1.1 supported
- Json data encoding supported
- QoS 0-2 supported
- User name and password authentication
- Maximum 256 data point tags

Ethernet

- 100 Mbit/s, full duplex (fixed)
- Dual port cut-through switch, RJ45 connectors

B License Information

LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

****

FatFs - FAT file system module R0.09b   (C)ChaN, 2013

FatFs module is a generic FAT file system module for small embedded systems.
This is a free software that opened for education, research and commercial
developments under license policy of following trems.

Copyright (C) 2013, ChaN, all right reserved.

The FatFs module is a free software and there is NO WARRANTY.
No restriction on use. You can use, modify and redistribute it for
personal, non-profit or commercial products UNDER YOUR RESPONSIBILITY.
Redistributions of source code must retain the above copyright notice.

****

Format - lightweight string formatting library.
Copyright (C) 2010-2013, Neil Johnson
All rights reserved.

Redistribution and use in source and binary forms,
with or without modification,
are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice,
  this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimer in the
  documentation and/or other materials provided with the distribution.
* Neither the name of nor the names of its contributors
  may be used to endorse or promote products derived from this software
  without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

****

Source code for the "strtod" library procedure.

Copyright (c) 1988-1993 The Regents of the University of California.
Copyright (c) 1994 Sun Microsystems, Inc.

This software is copyrighted by the Regents of the University of

California, Sun Microsystems, Inc., Scriptics Corporation,
and other parties. The following terms apply to all files associated
with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute,
and license this software and its documentation for any purpose, provided
that existing copyright notices are retained in all copies and that this
notice is included verbatim in any distributions. No written agreement,
license, or royalty fee is required for any of the authorized uses.
Modifications to this software may be copyrighted by their authors
and need not follow the licensing terms described here, provided that
the new terms are clearly indicated on the first page of each file where
they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY
FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES
ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY
DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE
IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE
NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR
MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the
U.S. government, the Government shall have only "Restricted Rights"
in the software and related documentation as defined in the Federal
Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you
are acquiring the software on behalf of the Department of Defense, the
software shall be classified as "Commercial Computer Software" and the
Government shall have only "Restricted Rights" as defined in Clause
252.227-7013 (c) (1) of DFARs. Notwithstanding the foregoing, the
authors grant the U.S. Government and others acting in its behalf
permission to use and distribute the software in accordance with the
terms specified in this license.

****

Copyright 2013 jQuery Foundation and other contributors
http://jquery.com/

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
"Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND

NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

****

rsvp.js

Copyright (c) 2013 Yehuda Katz, Tom Dale, and contributors

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies
of the Software, and to permit persons to whom the Software is furnished to do
so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

****

libb (big.js)

The MIT Expat Licence.

Copyright (c) 2012 Michael Mclaughlin

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
'Software'), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED 'AS IS', WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

****

The "inih" library is distributed under the New BSD license:

```
are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products
   derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT
SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

Author: David van Moolenbroek <david@minix3.org>

*****************************************************************************

MD5 routines

Copyright (C) 1999, 2000, 2002 Aladdin Enterprises.  All rights reserved.

This software is provided 'as-is', without any express or implied
warranty.  In no event will the authors be held liable for any damages
arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
   claim that you wrote the original software. If you use this software
   in a product, an acknowledgment in the product documentation would be
   appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be
   misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

L. Peter Deutsch
ghost@aladdin.com

*****************************************************************************

open62541 is licensed under the Mozilla Public License v2.0

This Source Code Form is subject to the terms of the Mozilla Public
License, v. 2.0. If a copy of the MPL was not distributed with this
file, You can obtain one at http://mozilla.org/MPL/2.0/.

To obtain customized changes please contact foss@anybus.com.

*****************************************************************************
```

```
musl as a whole is licensed under the following standard MIT license:


Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

For additional information about the PCG random number generation scheme,
including its license and other licensing options, visit

    http://www.pcg-random.org

******************************************************************************

queue.h

Copyright (c) 1991, 1993
The Regents of the University of California.  All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

@(#)queue.h 8.5 (Berkeley) 8/20/94

******************************************************************************

Buffer-based memory allocator

Copyright (C) 2006-2015, ARM Limited, All Rights Reserved
SPDX-License-Identifier: Apache-2.0
```

Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

****

Eclipse Distribution License - v 1.0
Copyright (c) 2007, Eclipse Foundation, Inc. and its licensors.

All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this
list of conditions and the following disclaimer.
Redistributions in binary form must reproduce the above copyright notice, this
list of conditions and the following disclaimer in the documentation and/or
other materials provided with the distribution.
Neither the name of the Eclipse Foundation, Inc. nor the names of its
contributors may be used to endorse or promote products derived from this
software without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAG

****

Eclipse Public License - v 1.0
THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC
LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM
CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

1. DEFINITIONS

"Contribution" means:

a) in the case of the initial Contributor, the initial code and documentation
distributed under this Agreement, and

b) in the case of each subsequent Contributor:

i) changes to the Program, and

ii) additions to the Program;

where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program.

"Contributor" means any person or entity that distributes the Program.

"Licensed Patents" mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

"Program" means the Contributions distributed in accordance with this Agreement.

"Recipient" means anyone who receives the Program under this Agreement, including all Contributors.

2. GRANT OF RIGHTS

a) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form.

b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder.

c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise.
As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any.
For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.

d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement.

3. REQUIREMENTS

A Contributor may choose to distribute the Program in object code form under

its own license agreement, provided that:

a) it complies with the terms and conditions of this Agreement; and

b) its license agreement:

i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions
of merchantability and fitness for a particular purpose;

ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits;

iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and

iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange.

When the Program is made available in source code form:

a) it must be made available under this Agreement; and

b) a copy of this Agreement must be included with each copy of the Program.

Contributors may not remove or alter any copyright notices contained within the Program.

Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution.

4. COMMERCIAL DISTRIBUTION

Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program,
the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program
in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs
(collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor
in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement.
In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial
Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that

Commercial Contributor then makes performance claims, or
offers warranties related to Product X, those performance claims and warranties
are such Commercial Contributor's responsibility alone. Under this section, the
Commercial Contributor would have to defend claims against
the other Contributors related to those performance claims and warranties, and
if a court requires any other Contributor to pay any damages as a result, the
Commercial Contributor must pay those damages.

5. NO WARRANTY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN
"AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR
IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR
CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A
PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the
appropriateness of using and distributing the Program and assumes
all risks associated with its exercise of rights under this Agreement ,
including but not limited to the risks and costs of program errors, compliance
with applicable laws, damage to or loss of data, programs or equipment,
and unavailability or interruption of operations.

6. DISCLAIMER OF LIABILITY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY
CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT
LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM
OR THE EXERCISE OF ANY RIGHTS GRANTED HEREUNDER, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGES.

7. GENERAL

If any provision of this Agreement is invalid or unenforceable under applicable
law, it shall not affect the validity or enforceability of the remainder of the
terms of this Agreement, and without further action by the
parties hereto, such provision shall be reformed to the minimum extent
necessary to make such provision valid and enforceable.

If Recipient institutes patent litigation against any entity (including a
cross-claim or counterclaim in a lawsuit) alleging that the Program itself
(excluding combinations of the Program with other software or hardware)
infringes such Recipient's patent(s), then such Recipient's rights granted
under Section 2(b) shall terminate as of the date such litigation is filed.

All Recipient's rights under this Agreement shall terminate if it fails to
comply with any of the material terms or conditions of this Agreement and does
not cure such failure in a reasonable period of time after becoming
aware of such noncompliance. If all Recipient's rights under this Agreement
terminate, Recipient agrees to cease use and distribution of the Program as
soon as reasonably practicable. However, Recipient's obligations under
this Agreement and any licenses granted by Recipient relating to the Program
shall continue and survive.

Everyone is permitted to copy and distribute copies of this Agreement, but in
order to avoid inconsistency the Agreement is copyrighted and may only be
modified in the following manner. The Agreement Steward reserves the
right to publish new versions (including revisions) of this Agreement from time
to time. No one other than the Agreement Steward has the right to modify this

Agreement. The Eclipse Foundation is the initial Agreement Steward.
The Eclipse Foundation may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number.
The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version.
Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise.
All rights in the Program not expressly granted under this Agreement are reserved.

This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation.

Box 4126
300 04 Halmstad, Sweden

info@hms.se