User Manual

Anybus® Communicator™ for Modbus RTU

Rev. 2.53

HMS Industrial Networks AB

	☎	✉
Germany	+49 - 721 - 96472 - 0	ge-sales@hms-networks.com
Japan	+81 - 45 - 478 -5340	jp-sales@hms-networks.com
Sweden	+46 - 35 - 17 29 20	sales@hms-networks.com
U.S.A	+1 - 773 - 404 - 3486	us-sales@hms-networks.com

Table of Contents

Chapter 4 Basic Settings

Chapter 5 Nodes

Chapter 6 Transactions

Chapter 7 Frame Objects

Chapter 8 Commands

About This Document

How To Use This Document

This document contains a general introduction as well as a description of the technical features provided by the Anybus Communicator, including the PC-based configuration software.

The reader of this document is expected to be familiar with PLC and software design, as well communication systems in general. The reader is also expected to be familiar with the Microsoft Windows operating system.

Important User Information

The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the application meets all performance and safety requirements including any applicable laws, regulations, codes, and standards.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS cannot assume responsibility or liability for actual use based on these examples and illustrations.

Warning:	This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.
ESD Note:	This product contains ESD (Electrostatic Discharge) sensitive parts that may be damaged if ESD control procedures are not followed. Static control precautions are required when handling the product. Failure to observe this may cause damage to the product.

Related Documents

Document name	Author
ABC-RTU Installation Leaflet	HMS

Document History

Summary of Recent Changes (2.52... 2.53)

Change	Page(s)
Corrected switch information (‘Physical Interface’)	1-4
Misc. corrections and adjustments	-

Revision List

Revision	Date	Author	Chapter	Description
2.00	2005-08-25	PeP	All	Second major release
2.50	2006-04-05	PeP	All	Major update
2.51	2006-06-01	PeP	-	Minor corrections
2.52	2006-12-22	PeP	-	Minor corrections
2.53	2007-11-23	PeP	All	Minor updates

Conventions & Terminology

The following conventions are used throughout this document:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The term 'user' refers to the person or persons responsible for installing the Anybus Communicator in a network.
- The term 'gateway' refers to the Anybus Communicator.
- Hexadecimal values are written in the format 0xNNNN, where NNNN is the hexadecimal value.
- Decimal values are represented as NNNN where NNNN is the decimal value
- As in all communication systems, the terms "input" and "output" can be ambiguous, because their meaning depend on which end of the link is being referenced. The convention in this document is that "input" and "output" are always being referenced to the master/scanner end of the link.

Glossary

Term	Meaning
ABC	Anybus® Communicator™
Broadcaster	A protocol-specific node in the configuration that handles transactions destined to all nodes.
RTU	Modbus RTU
Command	A pre-defined transaction.
Configuration	List of configured nodes with transactions on the sub-network.
Fieldbus	The higher level network to which the communicator is connected.
Fieldbus Control System	Fieldbus master
Frame Object	Low level entities which are used to describe the different parts of a Transaction.
Monitor	A tool for debugging the gateway and the network connections.
Node	A device in the configuration which defines the communication with a node on the sub-network
Sub-network	The network that is logically located on a subsidiary level with respect to the fieldbus, and to which the Anybus Communicator acts as a gateway.
Transaction	A generic building block that is used in the sub-network configuration and defines the data that is sent and received on the sub-network.
User	Person or persons responsible for installing the Anybus Communicator
Higher Level Network	In this case, Modbus RTU
Network	
Fieldbus	

Support

For technical support consult the online FAQ (www.anybus.com), or contact the nearest support centre:

HMS Sweden (Head Office)

E-mail:	support@hms-networks.com
Phone:	+46 (0) 35 - 17 29 20
Fax:	+46 (0) 35 - 17 29 09
Online:	www.anybus.com

HMS America

E-mail:	us-support@hms-networks.com
Phone:	+1-773-404-2271
Toll Free:	888-8-Anybus
Fax:	+1-773-404-1797
Online:	www.anybus.com

HMS Germany

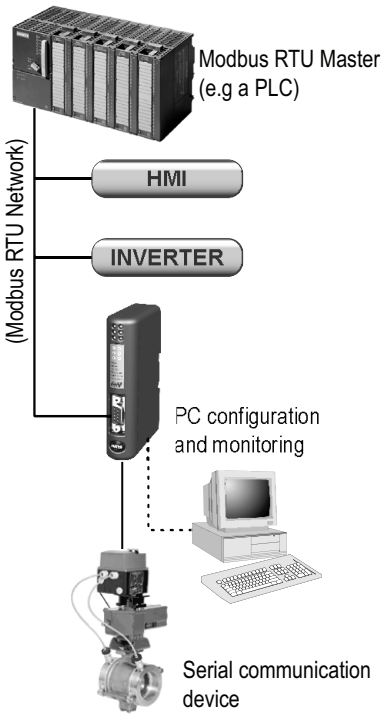
E-mail:	ge-support@hms-networks.com
Phone:	+49-721-96472-0
Fax:	+49-721-964-7210
Online:	www.anybus.com

HMS Japan

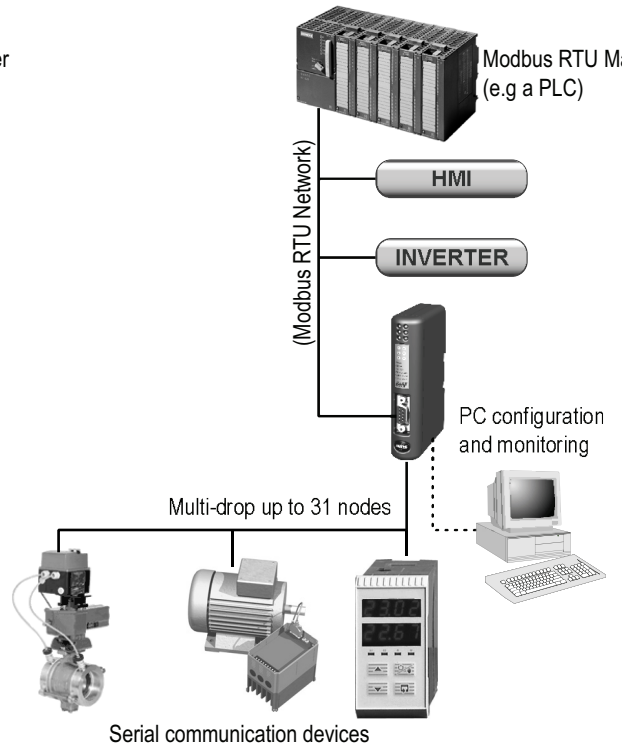
E-mail:	jp-support@hms-networks.com
Phone:	+81-45-478-5340
Fax:	+81-45-476-0315
Online:	www.anybus.com

About the Anybus Communicator for Modbus-RTU

The Anybus Communicator for Modbus RTU acts as a gateway between virtually any serial application protocol and a Modbus RTU-based network. Integration of industrial devices is enabled without loss of functionality, control and reliability, both when retro-fitting to existing equipment as well as when setting up new installations.



Single-Node Serial Sub Network



Multi-Node Serial Sub Network

Sub Network

The gateway can address up to 31 nodes, and supports the following physical standards:

- RS-232
- RS-422
- RS-485

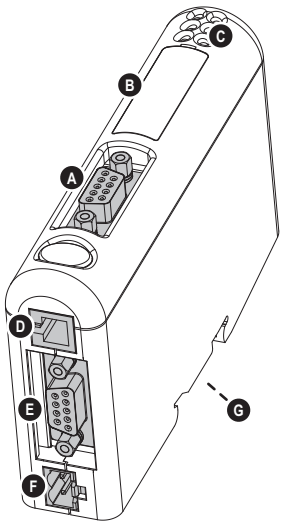
Modbus RTU Interface

Modbus RTU connectivity is provided through patented Anybus technology; a proven industrial communication solution used all over the world by leading manufacturers of industrial automation products.

- Galvanically isolated bus interface
- Coil and Register access
- RS-232 or RS-485 operation
- On-board configuration switches
- 1200... 57600bps operation

External View

For wiring and pin assignments, see A-1 “Connector Pin Assignments”.



A: Modbus RTU Connector

This connector is used to connect the gateway to the fieldbus.

See also...

- A-1 “Fieldbus Connector (Modbus-RTU)”

B: Configuration Switches

See also...

- 1-4 “Configuration Switches”

C: Status LEDs

See also...

- 1-3 “Status LEDs”

D: PC-connector

This connector is used to connect the gateway to a PC for configuration and monitoring purposes.

See also...

- A-2 “PC Connector”

E: Sub-network Connector

This connector is used to connect the gateway to the serial sub-network.

See also...

- A-3 “Sub-network Interface”

F: Power Connector

This connector is used to apply power to the gateway.

See also...

- A-1 “Power Connector”
- B-1 “Technical Specification”

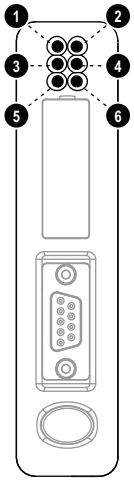
G: DIN-rail Connector

The DIN-rail mechanism connects the gateway to PE (Protective Earth).

See also...

- 1-5 “Hardware Installation”
- B-1 “Technical Specification”

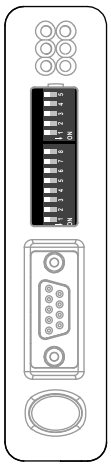
Status LEDs



#	State	Status
1 - Bus Error	Off	Normal operation
	Red	Bus error; CRC mismatch >10%
2 - Bus Ready	Off	Not powered
	Green	Normal operation (bus ready)
	Red	Bus is off line (bus not ready)
3 - Processing	Off	Currently not processing query
	Green, flashing	Currently processing query
4 - Switch Status	Off	Normal operation
	Red	Invalid configuration switch setting
5 - Subnet Status[a]	Off	Power off
	Green, flashing	Initializing and not running
	Green	Running
	Red	Stopped or subnet error, or timeout
6 - Device Status	Off	Power off
	Alternating Red/Green	Invalid or missing configuration
	Green	Initializing
	Green, flashing	Running
	Red, flashing	Contact HMS

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

Configuration Switches

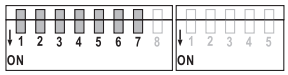


The configuration switches determines the basic communication settings for the Modbus interface. Normally, these switches are covered by a plastic hatch. When removing the hatch, avoid touching the circuit boards and components. If tools are used to open the hatch, be cautious.

Note that these settings cannot be changed during runtime, i.e. the gateway must be restarted in order for any changes to have effect.

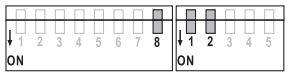
Node Address

Node Address	Sw. 1	Sw. 2	Sw. 3	Sw. 4	Sw. 5	Sw. 6	Sw. 7
(reserved)	OFF	OFF	OFF	OFF	OFF	OFF	OFF
1	OFF	OFF	OFF	OFF	OFF	OFF	ON
2	OFF	OFF	OFF	OFF	OFF	ON	OF
...	...	...	...	...	...	...	...
126	ON	ON	ON	ON	ON	ON	OFF
127	ON	ON	ON	ON	ON	ON	ON



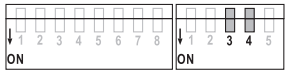
Baudrate Configuration

Baudrate	Sw. 8	Sw. 1	Sw. 2
(reserved)	OFF	OFF	OFF
1200 bps	OFF	OFF	ON
2400 bps	OFF	ON	OFF
4800 bps	OFF	ON	ON
9600 bps	ON	OFF	OFF
19200 bps (standard)	ON	OFF	ON
38400 bps	ON	ON	OFF
57600 bps	ON	ON	ON



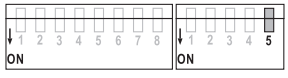
Parity & Stop Bits

Parity	Sw. 3	Sw. 4
(reserved)	OFF	OFF
No parity, 2 stop bits	OFF	ON
Even parity, 1 stop bit	ON	OFF
Odd parity, 1 stop bit	ON	ON



Physical Interface

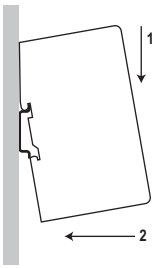
Interface Type	Sw. 5
RS-485	OFF
RS-232	ON



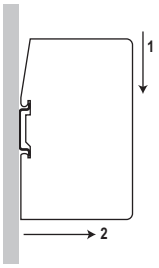
Hardware Installation

Perform the following steps when physically installing the gateway:

1. Snap the gateway on to the DIN-rail (See 1-2 "DIN-rail Connector")
   The DIN-rail mechanism works as follows:



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



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

2. Connect the gateway to the Modbus RTU network
3. Set the Modbus RTU communication settings using the on-board switches
4. Connect the gateway to the serial sub-network
5. Connect the gateway to a free COM-port on the PC via the PC-cable.
6. Connect the power cable and apply power
7. Start the ABC Config Tool program on the PC
   (The ABC Config Tool software attempts to detect the serial port automatically. If not successful, select the correct port manually in the "Port"-menu).
8. Configure the gateway using the ABC Config Tool and download the configuration

Software Installation

ABC Config Tool

System requirements

- Pentium 133 MHz or higher
- 10 MB of free space on the hard drive
- 8 MB RAM
- Screen resolution of 800x600 (16 bit colour) or higher
- Microsoft Windows™ NT4 / 2000 / XP
- Internet Explorer 4.01 SP1 or newer

Installation

- **Anybus Communicator resource CD**

  Insert the CD and follow the on-screen instructions. If the installation does not start automatically, right-click on the CD-drive icon and select Explore. Execute 'setup.exe' and follow the on-screen instructions.

- **From website**

  Download and execute the self-extracting .exe-file from the HMS website (www.anybus.com).

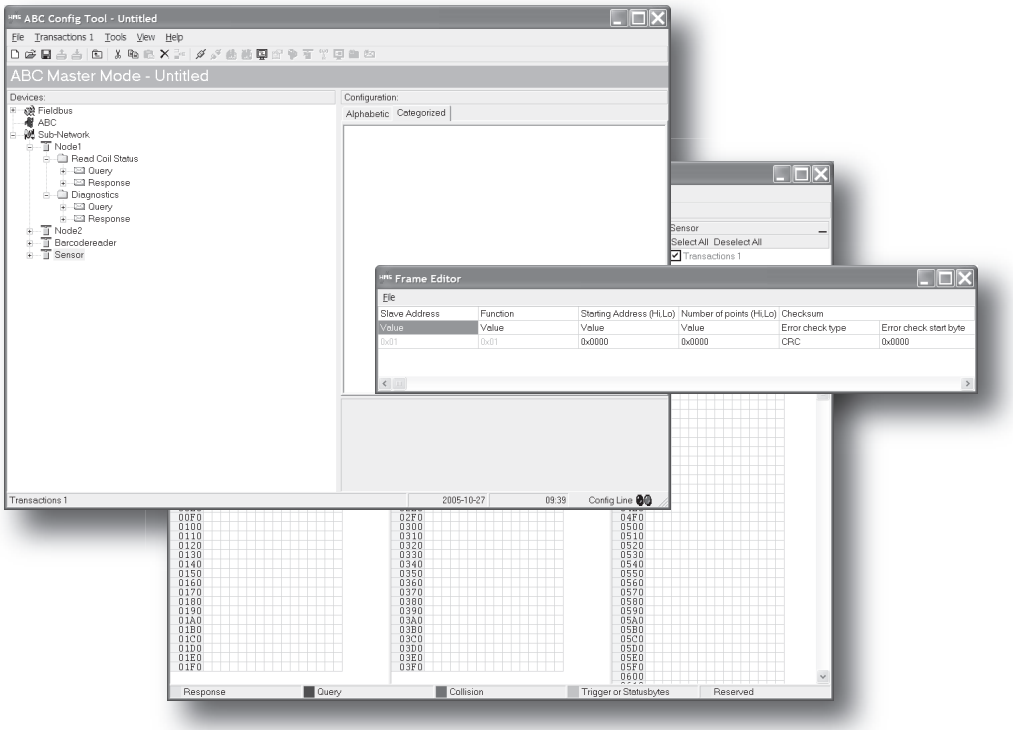
Basic Operation

General

The Anybus Communicator gateway is designed to exchange data between a serial sub-network and a higher level network (in this case Modbus RTU). Unlike most other gateway devices of similar kind, it does not have a fixed protocol for the sub-network, and can be configured to handle almost any form of serial communication.

The gateway can issue serial telegrams cyclically, on change of state, or based on trigger events issued by the control system of the higher level network (i.e. the fieldbus master or PLC). It can also monitor certain aspects of the sub-network communication and notify the higher level network when data has changed.

An essential part of the Anybus Communicator package is the ABC Config Tool, a Windows™ application which is used to supply the gateway with a description of the sub-network protocol. No programming skills are required; instead, a visual protocol description-system is used to specify the different parts of the serial communication.



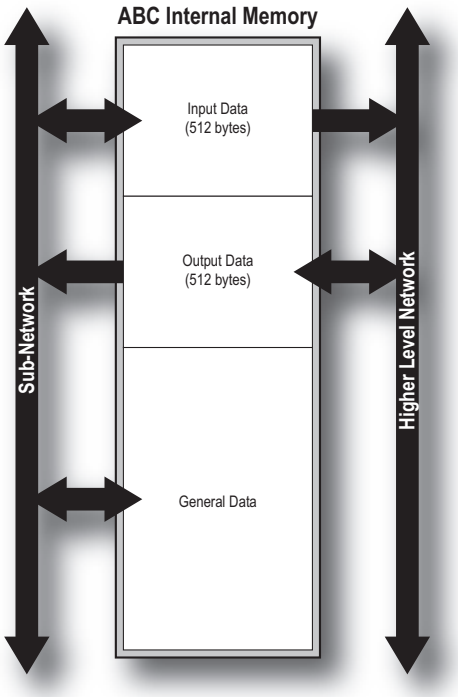
Data Exchange Model

Internally, the data exchanged on the sub-network, and the data exchanged on the higher level network, resides in the same memory.

This means that in order to exchange data with the sub-network, the higher level network simply reads and writes data to memory locations specified using the ABC Config Tool. The very same memory locations can then be exchanged on the sub-network.

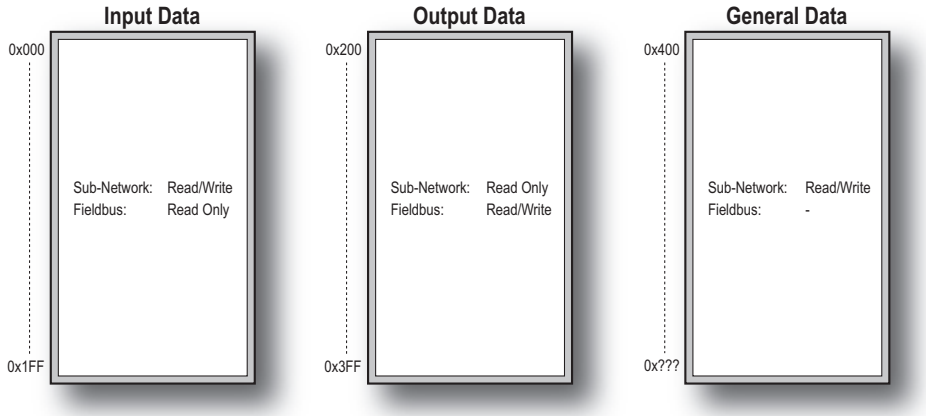
The internal memory buffer is divided into three areas based on their function:

- **Input Data (512 bytes)**
  This area can be read by the higher level network (in this case Modbus RTU).
- **Output Data (512 bytes)**
  This area can be read/written by the higher level network (in this case Modbus RTU).
- **General Data (Up to 1024 bytes)**
  This area cannot be accessed from the higher level network, but may be used for transfers between individual nodes on the sub-network, or as a general "scratch pad" for data. The actual size of this area depends on the amount of data that is exchanged on the sub-network. The gateway can handle up to 1024 bytes of General Data.



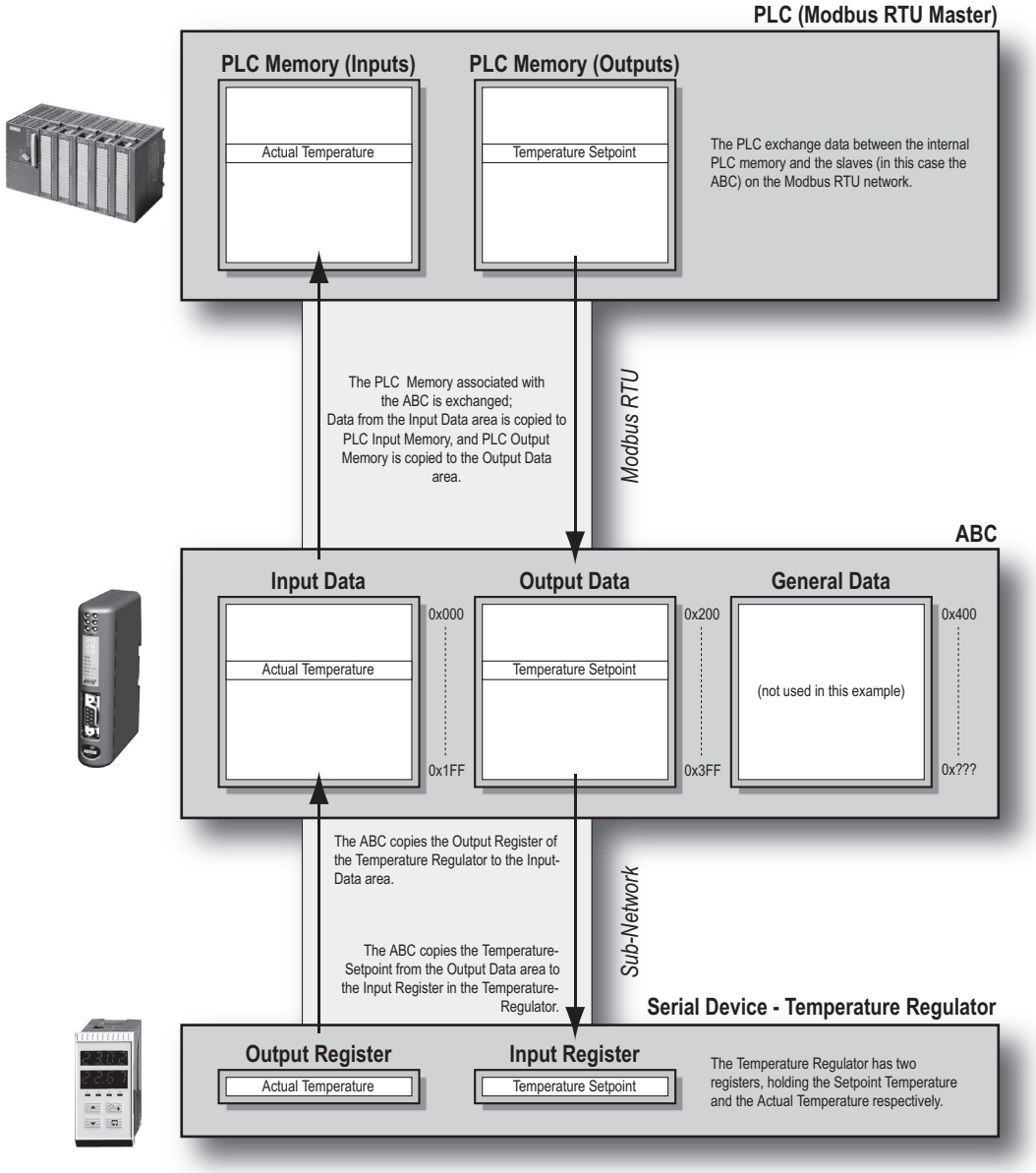
Memory Map

When building the sub-network configuration using the ABC Config Tool, the different areas described above are mapped to the memory locations (addresses) specified below.



Data Exchange Example

In the following example, a temperature regulator on the sub-network exchanges information with a PLC on the higher level network, via the internal memory buffers in the gateway.



Sub-Network Protocol

Protocol Modes

The gateway features two distinct modes of operation regarding the sub-network communication, called 'Master Mode' and 'Generic Data Mode'. Note that the protocol mode only specifies the basic communication model, not the actual sub-network protocol.

- **Master Mode**

  In this mode, the gateway acts as a master on the sub-network, and the serial communication takes place in a Query-Response fashion. The nodes on the network are not permitted to issue messages unless they have been addressed by the gateway first.

  For more information about this mode, see 2-5 "Master Mode".

- **Generic Data Mode**

  In this mode, there is no master-slave relationship between the sub-network nodes and the gateway; any node on the sub-network, including the gateway, may spontaneously produce or consume messages.

  For more information about this mode, see 2-5 "Generic Data Mode".

Protocol Building Blocks

The following building blocks are used in ABC Config Tool to describe the sub-network communication. How these blocks apply to the two protocol modes will be described later in this document.

- **Nodes**

  A node represents a single device on the sub-network. Each node can be associated with a number of Transactions, see below.

- **Transactions**

  A 'Transaction' represents a complete serial telegram, and consists of a number of Frame Objects (below). Each Transaction is associated with a set of parameters controlling how and when to use it on the sub-network.

- **Commands**

  Commands are simply pre-defined Transactions stored in the ABC Config Tool. This simplifies common operations by allowing Transactions to be stored and re-used.

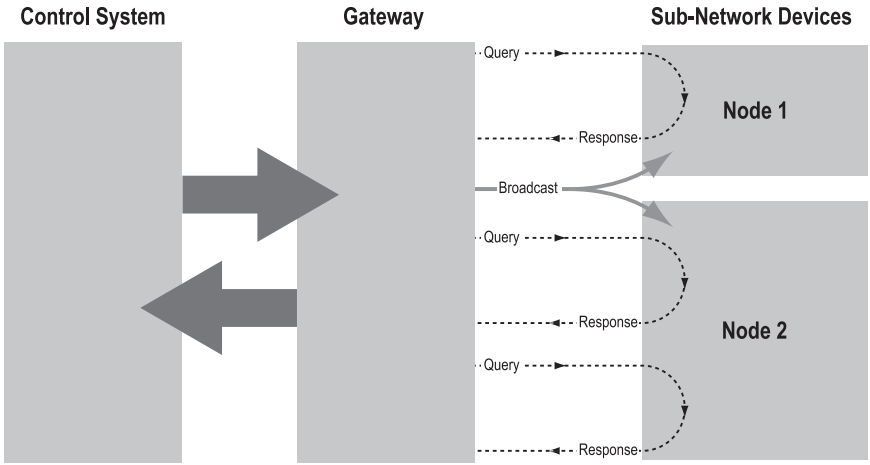
- **Frame Objects**

  Frame Objects are low level entities used to compose Transactions (see above). A Frame Object can represent a fixed value (a constant), a range of values (limit objects), a block of data or a calculated checksum.

Master Mode

In this mode, the communication is based on a Query/Response scheme; when the gateway issues a Query on the sub-network, the addressed node is expected to issue a Response to that Query. Nodes are not permitted issue Responses spontaneously, i.e. without first receiving a Query.

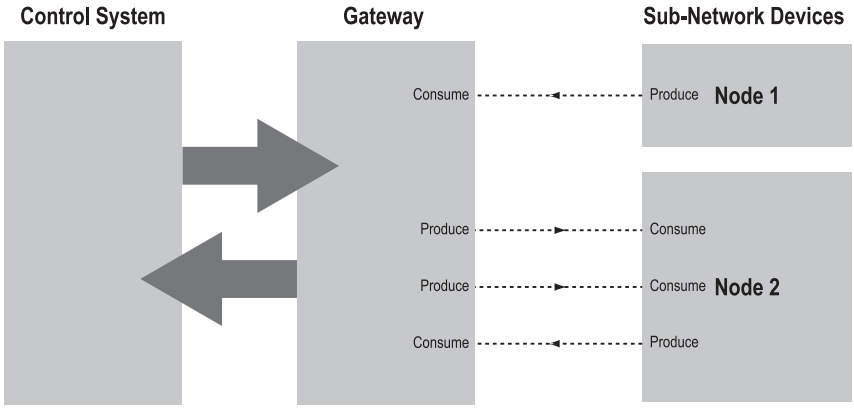
There is one exception to this rule; the Broadcaster. Most protocols offer some way of broadcasting messages to all nodes on the network, without expecting them to respond to the broadcasted message. This is also reflected in the gateway, which features a dedicated Broadcaster node.



In Master Mode, ABC Config Tool comes pre-loaded with most commonly used Modbus RTU commands, which can conveniently be reached by right-clicking on a node in the ABC Config Tool and selecting 'Insert New Command'. Note however that this does not in any way 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 nodes on the sub-network and the gateway. Any node, including the gateway, may spontaneously produce or consume a message. Nodes do not have to respond to messages, nor do they have to wait for a query in order to send one.



In the figure above, the gateway 'Consumes' data that is 'Produced' by a node on the sub-network. This 'Consumed' data can then be accessed from the higher level network. This also works the other way around; the data received from the higher level network is used to 'Produce' a message on the sub-network to be 'Consumed' by a node.

Data Representation on Modbus RTU

General

The Input- and Output Data areas are mapped to Modbus registers 0... 1279 and Coils 0... 20479.

Supported Function Codes

The following function codes are supported:

Function Code	Modbus Function	Associated with Area(s)
1	Read Coil	Input- and Output Data Area (0x000... 0x3FF)
2	Read Input Discretes	
3	Read Holding Registers	
4	Read Input Registers	
5	Write Coil	Output Data Area (0x200... 0x3FF)
6	Write Single Register	
15	Force Multiple Coils	
16	Force Multiple Registers	
22	Mask Write Register	
23	Read/Write Registers	Input- and Output Data Area (0x000... 0x3FF)

Coil & Register Map

The Input & Output Data areas are mapped to coils and registers as follows:

Register #	Coil #	Memory Location	Area	Comments
1	1... 16	0x000... 0x001	Input Data area	-
2	17... 32	0x002... 0x003		
...	...	...		
255	4065... 4080	0x1FC... 0x1FD		
256	4081... 4096	0x1FE... 0x1FF		
257... 1024	4097... 16384	-	-	(reserved)
1025	16385... 16400	0x200... 0x201	Output Data area	-
1026	16401... 16416	0x202... 0x203		
...	...	...		
1279	20449... 20464	0x3FC... 0x3FD		
1280	20465... 20480	0x3FE... 0x3FF		

Note: Coils are mapped MSB first, i.e. coil 0 corresponds to bit 15 of register 0.

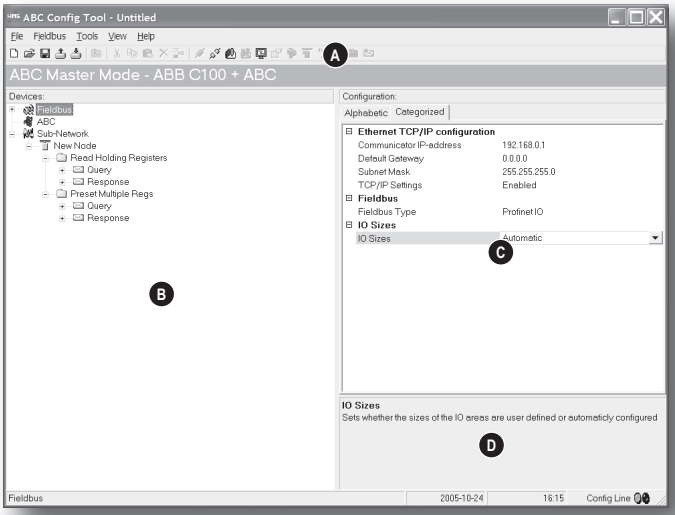
Supported Exception Codes

Exception Code	Name	Description
0x01	Illegal function	Function code not supported
0x02	Illegal data address	Invalid address in query
0x03	Illegal data value	Illegal data in request

Navigating the ABC Config Tool

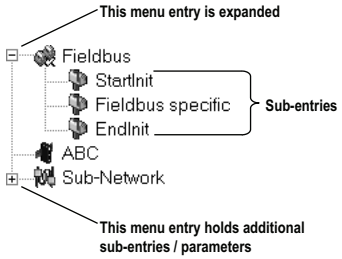
Main Window

The main window in the ABC Config Tool can be divided in 4 sections as follows:



- **A: Pull-down Menus & Tool Bar**

  The second drop-down menu from the left will change depending on the current context. The Tool Bar provides quick access to the most frequently used functions.



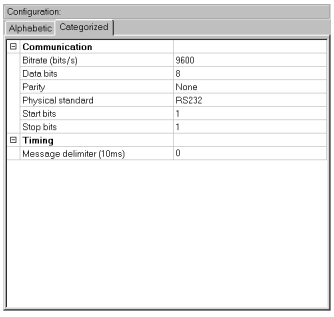
- **B: Navigation Section**

  This section is the main tool for selecting and altering different levels of the sub-network configuration.

  Entries preceded by a '+' holds further configuration parameters or 'sub menus'. To gain access to these parameters, the entry must be expanded by clicking '+'.

  There are three main levels in the navigation window, namely Fieldbus, ABC and Sub-network.

  Right-clicking on entries in this section brings out additional selections related to that particular entry.



Parameter Section

- **C: Parameter Section**

  This section holds a list of parameters or options related to the currently selected entry in the Navigation Section.

  The parameter value may be specified either using a selection box or manually, depending on the parameter itself. Values can be specified in decimal form (e.g. '42'), or in hexadecimal format (e.g. '0x2A').

Information Section

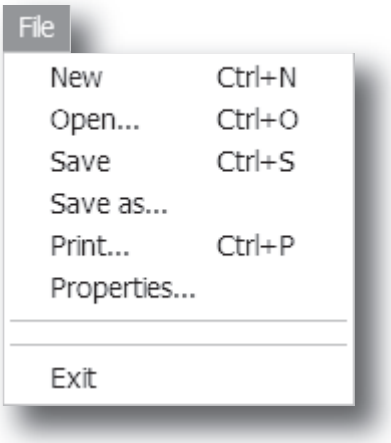
- **D: Information Section**

  This section holds information related to the currently selected parameter.

Pull-down Menu

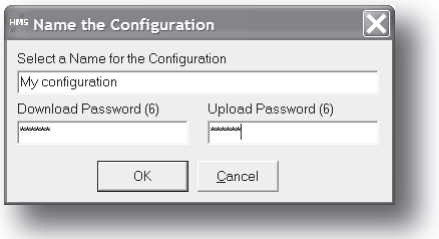
File

This menu features the following entries:



- **New**
  Create a new configuration.
  See also 12-1 “Configuration Wizards”.
- **Open...**
  Open a previously created configuration.
- **Save**
  Save the current configuration.
- **Save As...**
  Save the current configuration under a new name.
- **Print...**
  Send details about the current configuration to a printer.
- **Properties...**
  This brings out the following window:

Item	Description
Select a Name for the Configuration	A name for the configuration may be entered here
Download Password(6)	These fields can be used to password-protect the configuration in the gateway.
Upload Password(6)	

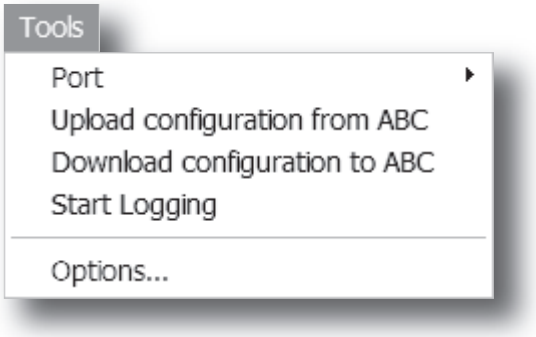


CAUTION: Always keep a copy of the password in a safe place. A lost password cannot be retrieved!

- **Exit**
  Close the ABC Config Tool.

Tools

This menu features the following entries:



- **Port**

  This entry selects the COM-port used for the configuration of the gateway.

- **Upload configuration from ABC**

  Upload the configuration from the gateway to the ABC Config Tool.

- **Download configuration to ABC**

  Download the current configuration into the gateway.

- **Start Logging**

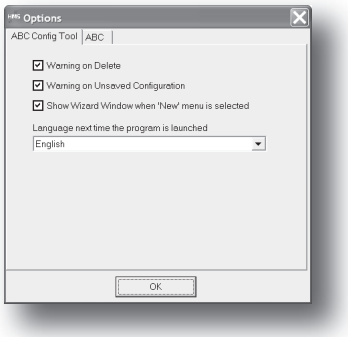
  Start the Data Logger (see 11-1 "Data Logger").

  Note that when the Data Logger is active, this menu-entry is changed to 'Stop Logging'.

- **Options**

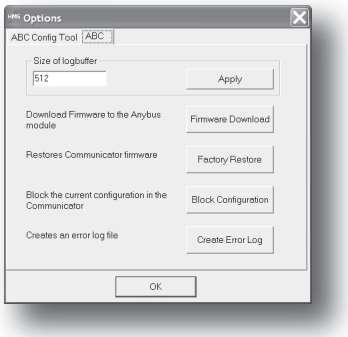
  This will bring out the following window:

Item	Description
Warning on Delete	A confirmation dialog is displayed each time something is deleted.
Warning on unsaved data	A confirmation dialog is displayed when closing the ABC Config Tool with unsaved data.
Show Wizard when "New" menu is selected	The Wizard is displayed each time a new configuration is created.
Language next time the program is launched	Selects which language to use. The new setting will be active the next time the program is launched.



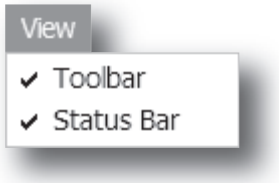
Selecting the 'ABC'-tab will reveal additional properties:

Item	Description
Size of logbuffer	By default, the Data Logger can log up to 512 entries in each direction. If necessary, it is possible to specify a different number of entries (valid settings range from 1...512). Cick 'Apply' to validate the new settings. See also 11-1 "Data Logger".
Firmware Download	Download firmware to the embedded fieldbus interface. **Warning: Use with caution.**
Factory Restore	Restores the gateway firmware to it's original state (does not affect the embedded fieldbus interface).
Block Configuration	When selected, the downloaded configuration will not be executed by the gateway. **Warning: Use with caution.**
Create Error log	Creates an error log file



View

This menu features the following entries:



- **Toolbar**

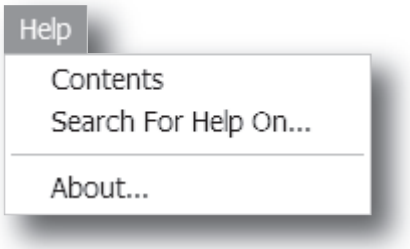
  This entry enables/disables the toolbar icons at the top of the main window.

- **Status Bar**

  This entry enables/disables the status bar at the bottom of the main window.

Help

This menu features the following entries:



- **Contents**

  Display the table of contents of the on-line help system.

  Note: At the time of writing, no on-line help system exists.

- **Search For Help On...**

  Search for a particular topic in the on-line help system.

  Note: At the time of writing, no on-line help system exists.

- **About...**

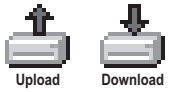
  Display general information about the gateway and the current build of ABC Config Tool.

Toolbar Icons

The toolbar features icons for the most commonly used functions.

- **New, Open & Save**
  See 3-2 "File".

- **Upload from ABC & Download to ABC**
  See 3-3 "Tools".



- **Up one Level**
  Clicking on this icon will move the selection in the navigation section.

- **Cut, Copy, Paste, Delete, Insert**
  These icons are used for common editing functions in the navigation section.

- **Connect**
  Clicking on this icon will cause the ABC Config Tool to attempt to connect to the gateway.

- **Disconnect**
  Clicking on this icon will cause the ABC Config Tool to disconnect from the gateway.

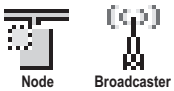
- **Start Logging & Stop Logging**
  See 3-3 "Tools" & 11-1 "Data Logger".

- **Sub-Network Monitor**
  Clicking on this icon will launch the Sub-network Monitor (see 9-1 "Sub Network Monitor").

- **Add Command**
  This icon is used to add commands to the currently selected node.

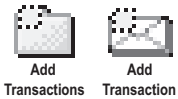
- **Add Mailbox**
  (Advanced functionality, see 14-1 "Mailbox Editor")

- **Add Node & Add Broadcaster**
  These icons are used to add nodes to the configuration.



- **Node Monitor**
  Clicking on this icon will launch the Node Monitor (see 10-1 "Node Monitor")

- **Add Transaction(s)**
  These icons are used to add transactions to the currently selected node.



Basic Settings

Fieldbus Settings

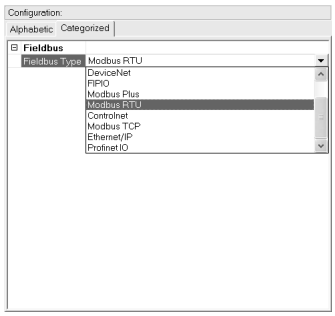
(Select 'Fieldbus' in the Navigation Section to gain access to the parameters described in this section).

General

During start-up the fieldbus interface of the gateway is initialized to fit the configuration created in the ABC Config Tool. Optionally, some initialisation parameters can be set manually to provide better control over how the data shall be treated by the gateway.

Fieldbus Type

The ABC Config Tool supports a wide range of networking systems. Make sure that this parameter is set to 'Modbus RTU'.



Fieldbus Type

ABC Parameters

(Select 'ABC' in the Navigation Section to gain access to the parameters described in this section).

Interface

Currently, only serial communications is supported.

Status / Control Word

(See 13-1 "Control and Status Registers").

Value	Description
Enabled	Enable the Control- and Status Registers. The 'Data Valid'-bit in the Control Register must be set to start the sub-network communication.
Enabled but no startup lock	This setting is similar to 'Enabled', except that the control system is not required to set the 'Data Valid'-bit to start the sub-network communication.
Disabled	This setting completely disables the Control- and Status Registers.

Module Reset

This parameter specifies how the gateway will behave in the event of a fatal error.

Value	Description
Enabled	The gateway will be restarted, and no error will be indicated to the user.
Disabled	The gateway will halt and indicate an error.

Protocol Mode

This parameter specifies which protocol mode to use for the sub-network.

Value	Description
Generic Data Mode	This mode is primarily intended for Produce & Consume-based protocols, where there are no Master-Slave relationship between the gateway and the nodes on the sub-network.
Master Mode	This mode is intended for 'Query & Response'-based protocols, where a single Master exchanges data with a number of Slaves.

See also 2-4 "Protocol Modes".

Statistics

The Transmit- and Receive Counters indicate how many transactions that have successfully been exchanged on the sub-network. This feature 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.

Both counters are enabled by setting 'Statistics' to 'Enabled'.

Sub-Network Parameters

(To gain access to the parameters described in this section, select 'Sub Network' in the Navigation Section).

Communication

These parameters specify the actual communication settings used for the sub-network.

Parameter	Description	Valid Settings
Bit rate	Selects the bit rate	1200...57600
Data bits	Selects the number of data bits	7, 8
Parity	Selects the parity mode	None, Odd, Even
Physical standard	Selects the physical interface type	RS232, RS422, RS485
Start bits	Number of start bits.	1
Stop bits	Number of stop bits.	1, 2

Start- and End Character

Note: These parameters are only available in Generic Data Mode.

Start and end characters are used to indicate the beginning and end of a serial message. For example, a message may be initiated with <ESC> and terminated with <LF>. In this case, the Start character would be 0x1B (ASCII code for <ESC>) and the End character 0x0A (ASCII code for <LF>)

Parameter	Description	Valid settings
End Character Value	End character for the message, ASCII	0x00 - 0xFF
Use End Character	Determines if the End character shall be used or not	Enable / Disable
Start Character Value	Start character for the message, ASCII	0x00 - 0xFF
Use Start Character	Determines if the Start character shall be used or not	Enable / Disable

Timing (Message Delimiter)

The parameters in this category differs slightly between the different Protocol Modes.

- **Master Mode**

  The Message Delimiter specifies the time that separates two messages in steps of 10ms. If set to 0 (zero), the gateway will use the standard Modbus delimiter of 3.5 characters (the actual number of ms will be calculated automatically based on the currently used communication settings).

- **Generic Data Mode**

  The Message Delimiter specifies the time that separates two messages in steps of 10µs.

Nodes

General

In ABC Config Tool, a node represents a single device on the network. While the gateway doesn't feature a scanlist in the traditional sense, all nodes, and their transactions, will be processed in the order they have been defined in the ABC Config Tool.

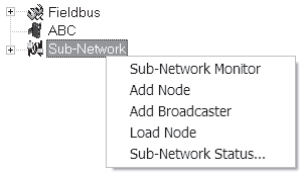
The maximum number of nodes that can be created in the ABC Config Tool is 31.

Adding & Managing Nodes

(Right-click on 'Sub Network' in the Navigation Section to gain access to these functions)

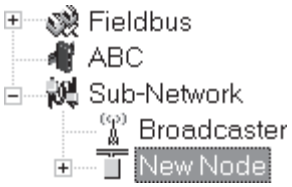
Function	Description
Paste	Paste a node from the clipboard
Sub Network Monitor	Launch the subnet monitor (9-1 "Sub Network Monitor")
Add Node	Add a node to the configuration
Add Broadcaster[a]	Add a broadcaster node to the configuration
Load Node	Add a previously saved node
Sub-Network Status...	View diagnostic information about the sub-network

a. This function is only available in Master Mode.



Node Parameters

(To gain access to the parameters described in this section, select a node in the Navigation Section).



Parameter	Description
Slave Address	The value entered here may be used to set the node address in certain commands. For more information, see 8-3 "The Command Editor".

Transactions

General

As mentioned previously, transactions are representations of the actual serial telegrams exchanged on the serial sub-network. While the gateway doesn't feature a scanlist in the traditional sense, all nodes, and their transactions, will be processed in the order they have been defined in the ABC Config Tool.

Transactions are handled slightly differently in the two protocol modes:

- **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 sub-network. The Broadcaster can only send transactions.

- **Generic Data Mode**

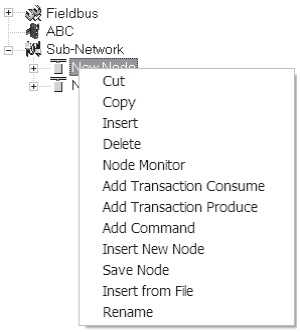
  Transactions can be added as desired for both directions. Transactions sent to the sub-network are called 'Transaction Produce', and transactions issued by other nodes are called 'Transaction Consume'.

Theoretically, the gateway supports up to 100 transactions. The actual number may however be less depending on the memory requirements of the defined transactions.

Adding & Managing Transactions

(Right-click on a node in the Navigation Section to gain access to these functions)

Function	Description
Cut	Cut a node to the clipboard
Copy	Copy a node to the clipboard
Insert	Insert a node from the clipboard
Delete	Delete a node
Node Monitor	Launch the node monitor (10-1 "Node Monitor")
Add Transaction(s)[a]	On regular nodes, this adds a Query and a Response. The two transactions will be grouped in order to increase readability. On the Broadcaster, a single transaction will be added.
Add Transaction Consume[b]	Add a 'Consume'-transaction
Add transaction Produce[b]	Add a 'Produce'-transaction
Add Command	Add pre-defined transactions to the node
Insert New Node	Insert a new node above the currently selected one
Save Node	Save the selected node
Insert from File	Insert a previously saved node above the currently selected node
Rename	To increase readability, each node can be given a unique name using this function



a. Only available in Master Mode
b. Only available in Generic Data Mode

Transaction Parameters (Master Mode)

Parameters (Query & Broadcast)

(To gain access to these parameters, select a Query- or Broadcast- transaction in the Navigation Section)

Parameter	Description
Minimum time between broadcasts (10ms)	This parameter specifies how long the gateway shall 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 unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms. **Note:** This setting is only relevant for the Broadcaster node.
Offline options for field-bus	This parameter specifies the action to take for this transaction if the higher level network goes off-line. This affects the data that is sent to the sub-network. • **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 sub-network is stopped
Offline options for sub-network	This parameter specifies the action to take for this transaction if the sub-network goes off-line. This affects the data that is reported to the control system. • **Clear** - Data is cleared (0) on the higher level network if the sub-network goes offline • **Freeze** - Data is frozen on the higher level network if the sub-network goes offline
Reconnect time (10ms)	This parameter specifies how long the gateway shall wait before attempting to re-connect a disconnected node. A node will be disconnected in case the maximum number of retries (below) has been reached. The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms. **Note:** This setting is not relevant for the Broadcaster node.
Retries	This parameter specifies how many times a timeout may occur in sequence before the node is disconnected.
Timeout time (10ms)	This parameter specifies how long the gateway will wait for a response from a node. If this time is exceeded, the gateway will re-transmit the Query until the maximum number of retries (see above) has been reached. The unit is milliseconds (ms) and the entered value is multiplied by 10, which means that the shortest time is 10 ms.
Trigger byte address	This parameter specifies the location of the trigger byte in internal memory (only relevant when 'Update mode' is set to 'Change of state on trigger').
Update mode	This parameter is used to specify 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 transaction is issued each time the data area associated with the transaction has changed. • **Single shot** The Query is issued once at start up. • **Change of state on trigger** The Query 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 one. The location of the trigger byte is specified by the 'Trigger byte address' parameter.
Update time (10ms)	This parameter specifies how often the transaction will be issued in steps of 10ms (only relevant when 'Update mode' is set to 'Cyclically').

Parameters (Response)

(To gain access to these parameters, select a Response-transaction in the Navigation Section)

Parameter	Description
Trigger byte	This parameter is used to enable/disable the trigger functionality for the response. If enabled, the gateway will increase the trigger byte by one when the gateway receives new data from the sub-network. This can be used to notify the control system of the updated data. The location of the trigger byte is specified by the 'Trigger byte address' parameter below.
Trigger byte address	This parameter specifies the location of the trigger byte in the internal memory buffer. Valid settings range from 0x000... 0x1FF and 0x400... 0xNNN

Transaction Parameters (Generic Data Mode)

Produce-Transactions

(To gain access to these parameters, select a Produce Transaction in the Navigation Section)

Parameter	Description
Offline options for fieldbus	This parameter specifies the action to take for this transaction if the higher level network goes off-line. This affects the data that is sent to the sub-network. • **Clear** Data is cleared (0) on the sub-network if the higher level network goes offline • **Freeze** Data is frozen on the sub-network if the higher level network goes offline • **NoScanning** Stop sub-net scanning for this transaction if the higher level network goes offline
Update mode	The update mode for the transaction: • **Cyclically** The transaction is sent cyclically at the interval specified in the 'Update Time'-parameter. • **On data change** The transaction is sent each time its data has changed. • **Single shot** The transaction is sent once at startup. • **Change of state on trigger** The transaction is sent when the trigger byte has changed. This feature enables the control system to notify the gateway when to issue a particular transaction. To use this feature correctly, the control system must first update the data area associated with the transaction, then increase the trigger byte by one. The location of the trigger byte is specified by the 'Trigger byte address' parameter.
Update time (10ms)	This parameter specifies how often the transaction will be issued in steps of 10ms (only relevant when 'Update mode' is set to 'Cyclically').

Parameter	Description
Trigger byte address	This parameter specifies location of the trigger byte in the internal memory buffer. If 'Update mode' is set to 'Change of state on trigger', the memory location specified by this parameter is monitored by the gateway. Whenever the trigger byte is updated, the gateway will produce the transaction on the sub-network. This way, the control system can instruct the gateway to produce a specific transaction on the sub-network by updating the corresponding trigger byte. The trigger byte should be incremented by one for each activation. **Note:** This parameter has no affect unless the 'Update mode' parameter is set to 'Change of state on trigger'.

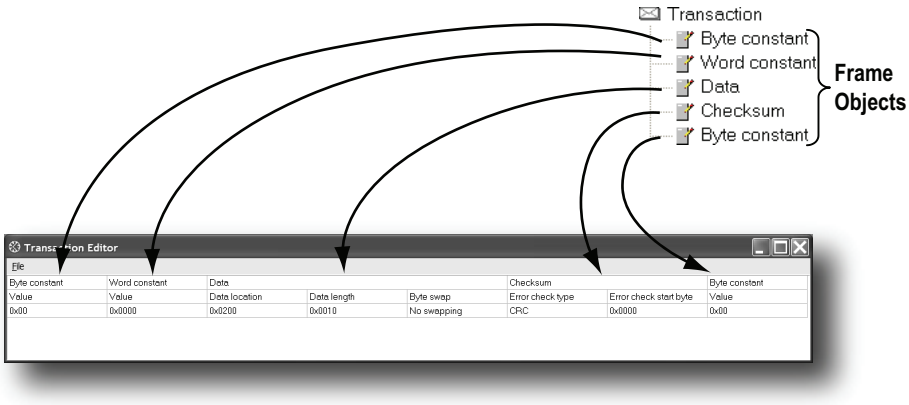
Consume-Transactions

(To gain access to these parameters, select a Consume Transaction in the Navigation Section)

Parameter	Description
Offline options for sub-network	This parameter specifies the action to take for this transaction if the sub-network goes off-line. This affects the data that is sent to the higher level network. • **Clear** Data is cleared (0) on the higher level network if the sub-network goes offline • **Freeze** Data is frozen on the higher level network if the sub-network goes offline
Offline timeout time (10ms)	This parameter specifies the maximum allowed time between two incoming messages in steps of 10ms. If this time is exceeded, the sub-network is considered to be offline. A value of 0 disables this feature, i.e. the sub-network can never go offline.
Trigger byte	• **Enable** Enables the trigger byte. The location of the trigger byte must be specified in the 'Trigger byte address' (below). The trigger byte value will be increased each time a valid transaction has been consumed by the gateway. This feature enables the control system to be notified each time new data has been consumed on the sub-network. • **Disable** Disables the trigger byte functionality.
Trigger byte address	This parameter specifies the location of the trigger byte in the internal memory buffer. Valid settings range from 0x000... 0x1FF and 0x400... 0xNNN

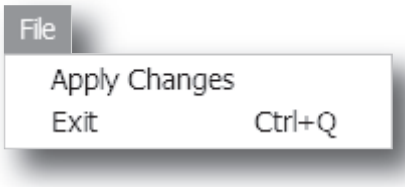
Transaction Editor

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.



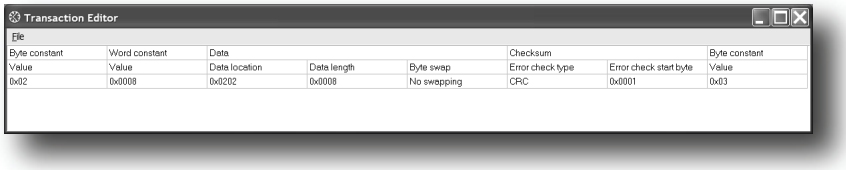
To edit the value of a parameter, click on it and enter a new value using the keyboard. When editing transactions which are based on pre-defined commands, certain parts of the transaction may not be editable.

The File-menu features the following entries:



- **Apply Changes**
  This will save any changes and exit to the main window.
- **Exit**
  Exit without saving.

Example:



The transaction created in this example are built up as follows:

The first byte holds the STX (0x02) followed by two bytes specifying the length of the data field (in this case 8). The next 8 bytes are data and since this is a 'query'-transaction, the data is to be fetched from the Output Area which starts at address location 0x202. No swapping will be performed on the data. This is followed by a two-byte checksum. The checksum calculation starts with the second byte in the transaction.

The transaction ends with a byte constant, the ETX (0x03).

Frame Objects

General

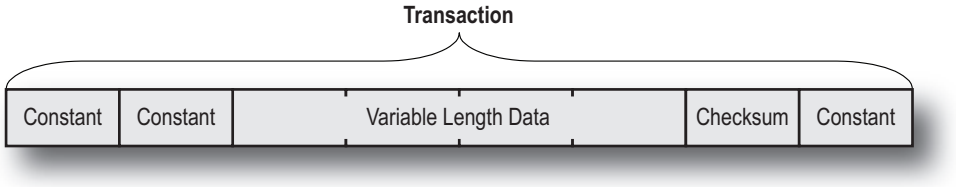
Each transaction consists of Frame Objects which makes up the serial telegram frame. Each Frame Object specifies how the gateway shall interpret or generate a particular part of the telegram.

There are 5 types of frame objects, which are described in detail later in this chapter:

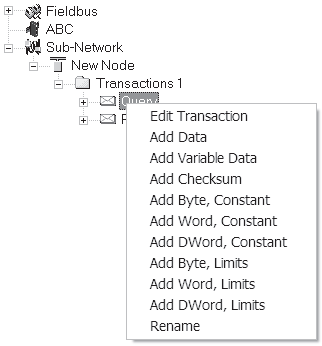
- Constant Objects
- Limit Objects
- Data Objects
- Variable Data Objects
- Checksum Objects

Example:

The following Transaction consists of several frame objects; three constants, a data object, and a checksum object.

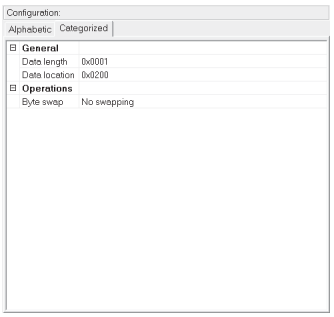


Adding and Editing Frame Objects



To add a frame object to a Transaction, right-click on the Transaction in the Navigation Section and select one of the entries in the menu that appears.

The entry called 'Transaction Editor' will launch the Transaction Editor, which is used to edit transactions and frame objects in a more visual manner. For more information, see 6-5 "Transaction Editor".



Data Object, Parameters

To edit parameters associated with a particular frame object, select the frame object in the Navigation Section. The settings for that frame object will be displayed in the Parameter Section.

It is also possible to edit the frame objects in a transaction in a more visual manner using the Transaction Editor, see 6-5 "Transaction Editor"

Constant Objects (Byte, Word, Dword)

Constant Objects have a fixed value and come in three sizes:

- **Byte**
  8 bits
- **Word**
  16 bits
- **Dword**
  32 bits

Constants are handled differently depending on the direction of the transaction:

- **Produce/Query Transactions**
  The gateway will send the value as it is without processing it.
- **Consume/Response Transactions**
  The gateway will check if the received byte/word/dword matches the specified value. If not, the message will be discarded.

To set the value of the object, select it in the Navigation Section enter the desired value in the Parameter section.

Parameter	Description
Value	Constant value

Limit Objects (Byte, Word, Dword)

Limit Objects have a fixed range and come in three sizes:

- **Byte**
  8 bits
- **Word**
  16 bits
- **Dword**
  32 bits

Limit Objects are handled differently depending on the direction of the transaction:

- **Produce/Query Transactions**
  This object shall not be used for such transactions (value will be undefined)
- **Consume/Response Transactions**
  The gateway will check if the received byte/word/dword fits inside the specified boundaries. If not, the message will be discarded.

There are 3 types of interval objects:

- **Byte**
  8 bit interval
- **Word**
  16 bit interval
- **Dword**
  32 bit interval

To set the range of the object, select it in the Navigation Section enter the desired range in the Parameter section as follows:

Parameter	Description
Maximum Value	This is the largest allowed value for the range. Range: 0x00... 0xFFh (byte) 0x0000... 0xFFFFh (word) 0x00000000... 0xFFFFFFFFh (dword) **Note:** Value must be larger than the Minimum Value (below)
Minimum Value	This is the smallest allowed value for the range. Range: 0x00... 0xFEh (byte) 0x0000... 0xFFFEh (word) 0x00000000... 0xFFFFFFFEh (dword) **Note:** Value must be less than the Maximum Value (above)

Data Object

Data Objects are used to represent raw data as follows:

- **Produce/Query Transactions**
  The specified data block is forwarded from the higher level network to the sub-network.
- **Consume/Response Transactions**
  The specified data block is forwarded from the sub-network to the higher level network.

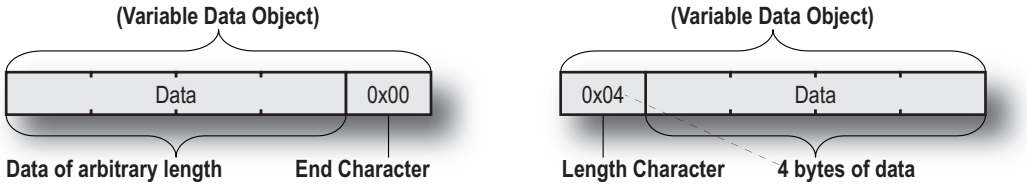
To specify the properties of the object, select it in the Navigation Section enter the desired settings in the Parameter section as follows:

Parameter	Description
Byte Swapping	• **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 case of a Response or Consume transaction, incoming messages where the data size differs from the value specified here will be discarded.
Data Location	The location of the data block in the internal memory buffer

Variable Data Object

Note: Only one Variable Data Object is permitted for each transaction.

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 as follows:



- **Produce/Query Transactions**
  The specified data block will be forwarded from the higher level network to the sub-network. 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 sub-network or discarded.
- **Consume/Response Transactions**
  The specified data block is forwarded from the sub-network 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 higher level network or discarded.

To specify the properties of the object, select it in the Navigation Section enter the desired settings in the Parameter section as follows:

Parameter	Description
Byte Swapping	• **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	• **Enabled**[a] Fill unused data with the value specified in 'Filler byte'. • **Disabled** Don't fill
Filler byte	Filler byte value. Only used if 'Fill unused bytes' has been enabled.
Data Location	The offset in the internal memory buffer where the data shall be read from / written to
Object Delimiter	• **Length Character** Length character is visible in the internal memory buffer but *not* on the sub-network • **Length Character Visible** The length character is visible both in the internal memory buffer *and* on the sub-network. • **End Character** The end character is visible in the internal memory buffer but *not* on the sub-network. • **End Character Visible** The end character is visible both in the internal memory buffer *and* on the sub-network • **No Character**[a] No End- or Length-character is generated in the internal memory buffer.
End Character Value	End Character value[b]
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.

a. Only relevant for Consume/Response transactions
b. Only used if 'Object Delimiter' is set to 'End Character' or 'End Character Visible'

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.

Parameter	Description
Error Check Start byte	This parameter 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 0xFFFF polynome (Modbus RTU standard) • **LRC (1 byte)** All bytes are added together as unsigned 8-bit values. The 2's complement of the result will be used as a checksum. • **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. • **AddInvASCII (2 bytes)** All bytes are added together as unsigned 8-bit values. The lowest 8 bits in the result are inversed and used as a checksum, represented as hexadecimal ASCII (2 bytes).

Commands

General

As mentioned previously, Commands are actually pre-defined transactions that can be stored and re-used. Just like regular transactions, commands consist of frame objects and are representations of the actual serial telegrams exchanged on the serial sub-network.

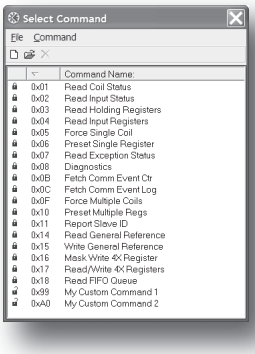
Adding a command to a node actually results in (a) transaction(s) being added according to the directions specified in the command. The Frame Objects in such a transaction may retrieve their values not only from parameters in the parameter section, but also from other sources such as the 'SlaveAddress'-parameter (see 5-1 "Node Parameters"). In such case, the parameters in the parameter section will be greyed out and cannot be edited directly.

In Master Mode, ABC Config Tool comes pre-loaded with commands for most common Modbus RTU functions. Additional commands can easily be added using the Command Editor (see 8-3 "The Command Editor"). In Generic Data Mode, no pre-defined commands exist, but custom ones may be implemented as desired.

Adding & Managing Commands

To add a command to a node, right-click on the node in the Navigation Section and select 'Add Command'.

A list of commands will appear:



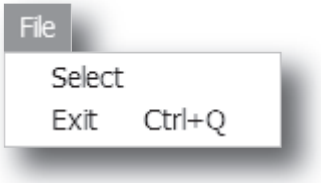
Select the desired command in the list, and select 'Add Command' in the 'Command'-menu. The specified command will be added to the node.

Just like other transactions, the frame objects of 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.

Pull-Down Menu

File

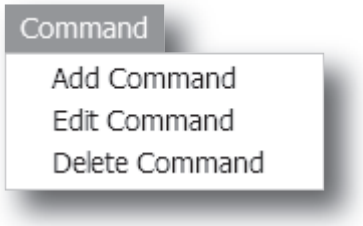
This menu features the following entries:



- **Select**
  Add the currently selected Command to the node.
- **Exit**
  Exit without adding a command to the node.

Command

This menu is used to manage the commands in the list:



- **Add Command**
  Add a custom command to the list, and open the new command in the Command Editor.
  See also 8-3 “The Command Editor”.
- **Edit Command**
  Edit the currently selected command using the Command Editor.
  See also 8-3 “The Command Editor”.
- **Delete Command**
  Delete the currently selected command from the list. Note that some commands are fixed and cannot be deleted.

Toolbar Icons

The toolbar features icons for the most commonly used functions.

- **Add Command**
  (Same as ‘Add Command’ in the ‘Command’-menu).

- **Edit Command**
  (Same as ‘Edit Command’ in the ‘Command’-menu).

- **Delete Command**
  (Same as ‘Delete Command’ in the ‘Command’-menu).

The Command Editor

General

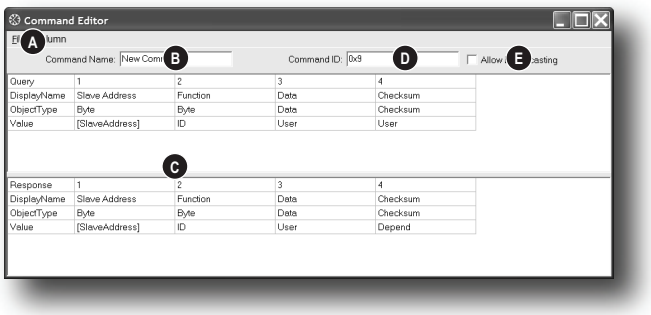
The Command Editor is used to define new commands and edit existing ones. This makes it possible to build a library of commands, which can be stored and re-used at a later stage.

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 uses Master Mode. The procedures involved are similar in General Data Mode, but without the limitations imposed by the Modbus RTU protocol.

Basic Navigation

Open the Command Editor by selecting 'Edit Command' or 'Add Command' from the 'Command'-menu.



A: Pull-down Menu

See 8-4 "Pull-down Menu".

B: Name of Command

Actual name of the command, in text form.

C: Command Transactions

This section holds the actual transactions associated with the command. This can either be a Query-Response pair, or a single transaction, depending on the protocol mode etc.

D: Command ID

This can be used as desired when building the command, e.g. to specify the function code.

E: Other Settings

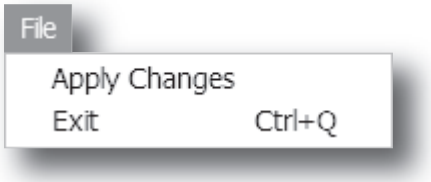
Setting	Description
Allow Broadcasting	Specifies if it is allowed to broadcast the command (only relevant in Master Mode)
Produce	The command is producing data (Generic Data Mode only)
Consume	The command is consuming data (Generic Data Mode only)

Pull-down Menu

File

This menu features the following entries:

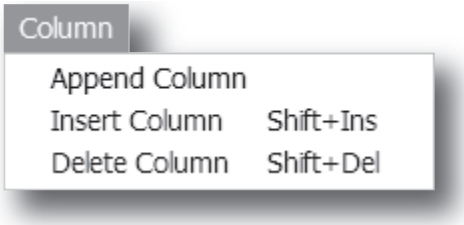
- **Apply Changes**
  Save changes and exit to the main window.
- **Exit**
  Exit without saving.



Column

The functions in this menu alters the structure of the command.

- **Append Column**
  Add another column to the command.
- **Insert Column**
  Insert a column at the selected position.
- **Delete Command**
  Delete the column at the selected position.



Editing a Command

As mentioned previously, the transaction section in the Command Editor represents the actual transactions associated with the command. Each column represents a frame object within the transaction.

Each column features four rows with the following parameters:

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

  The upper right cell indicates the direction of the transaction.

- **DisplayName**

  Each column can be named so that the different parts of the command appears in a more user friendly manner when editing its settings in the Transaction Editor or in the Parameter Section of the Main Window.

- **ObjectType**

  This row specifies the type of frame object that shall be used for the column.

- **Value**

  This row specifies where the frame object shall retrieve its value/settings.

Value	Description
Depend	This setting is only relevant for Responses in Master Mode. The value will be retrieved from the corresponding part of the 'Query'-transaction.
Id	The value will be retrieved from the 'Command ID'-setting (see 8-3 "Basic Navigation").
User	The settings associated with the object can be edited by the user.
[SlaveAddress]	The value will be retrieved from the 'SlaveAddress'-parameter (see 5-1 "Node Parameters").
(other settings)	Other settings are no longer supported.

Example: Specifying a Modbus-RTU Command in Master Mode

In the following example, a Modbus-RTU command is created in Master Mode. In Modbus-RTU, a transaction always feature the following parts:

- Slave Address (1 byte)
- Function Code (1 bytes)
- A data field
- CRC (CRC-16)

Furthermore, each command always consists of a Query and a Response.

- **Example Query**

Query	1	2	3	4
DisplayName	Slave Address	Function	Data	Checksum
Object Type	Byte Object	Byte Object	Data Object	Checksum Object
Value	[SlaveAddress]	ID	User	User
	The value of this byte constant will be set using the 'SlaveAddress' parameter (see 5-1 "Node Parameters").	The value of this byte constant will be set using the 'Command ID'-field.	The size and location of the data associated with this object is determined by the user.	The checksum type etc can be selected by the user. By default, this is set to match the Modbus-RTU standard.

- **Example Response**

Response	1	2	3	4
DisplayName	Slave Address	Function	Data	Checksum
Object Type	Byte Object	Byte Object	Data Object	Checksum Object
Value	[SlaveAddress]	ID	User	Depend
	This value is linked to the 'SlaveAddress' parameter in the parameter window.	The value of this byte constant will be set using the 'Command ID'-field.	The size and location of the data associated with this object is determined by the user.	This object will retrieve its settings from the corresponding object in the Query.

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

Name the new command by entering it's name in the 'Command Name'-field, and enter a suitable function code in the 'Command ID'-field. If the command is allowed to be broadcasted, check the 'Allow Broadcasting'-checkbox.

Sub Network Monitor

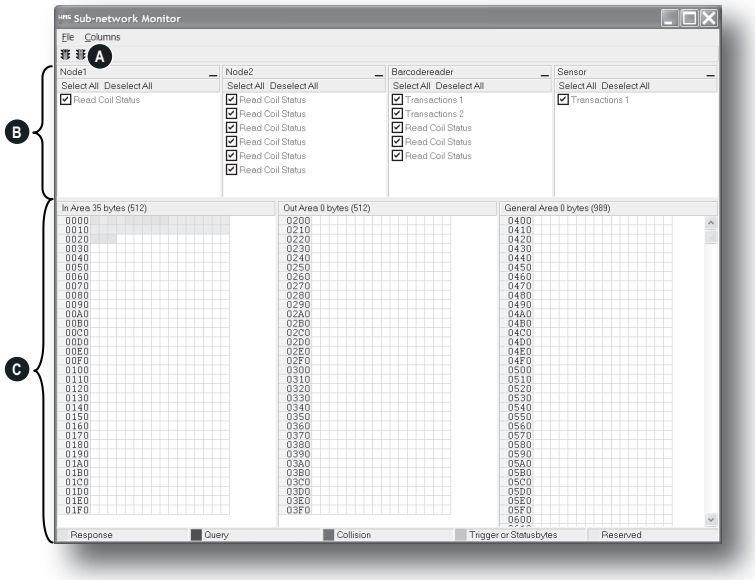
General

The Sub Network Monitor is intended to simplify configuration and troubleshooting of the sub network. It's main function is to display the data allocated for sub-network communication and detect if any area has been allocated twice (i.e if a collision has occurred).

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

Note: The sub-network monitor has a negative influence on the overall performance of the gateway. Therefore the monitor functionality should be used with care.

Operation



A: Start Network & Stop Network Icons

These icons controls the sub-network activity. To stop all sub-network activity, click on the red light. To start the sub-network again, click on the green light.


Start


Stop

B: Nodes / 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).

C: Monitor Section

This section visualises how data is allocated in the Input, Output and General Data areas.

Colour	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.

Node Monitor

General

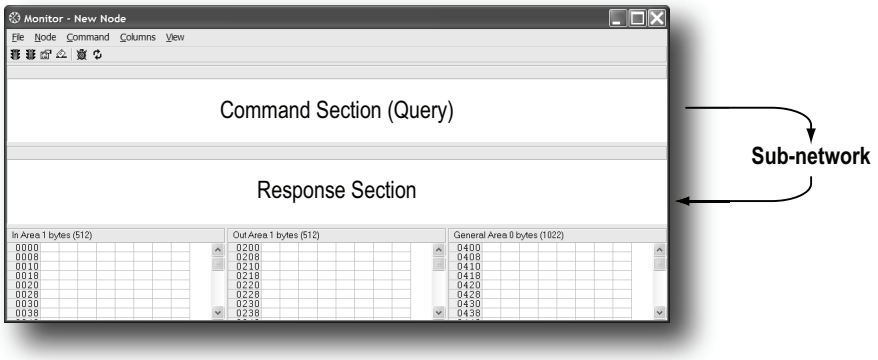
The Node Monitor can provide valuable information when setting up the communication with the sub-network, by allowing individual commands to be issued manually, and monitoring the response (if applicable). It also provides an overview of the memory used by a particular node.

Note: The node monitor has a negative influence on the overall performance of the gateway, i.e. it should be used only when necessary.

The Node Monitor behaves somewhat differently in the two protocol modes:

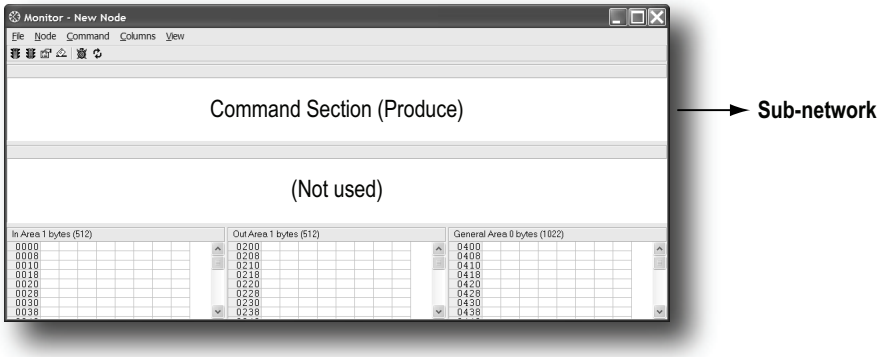
- **Master Mode**

  The selected Command (Query Transaction) is sent to the sub-network. The response to the Query can be monitored in the Response Section.

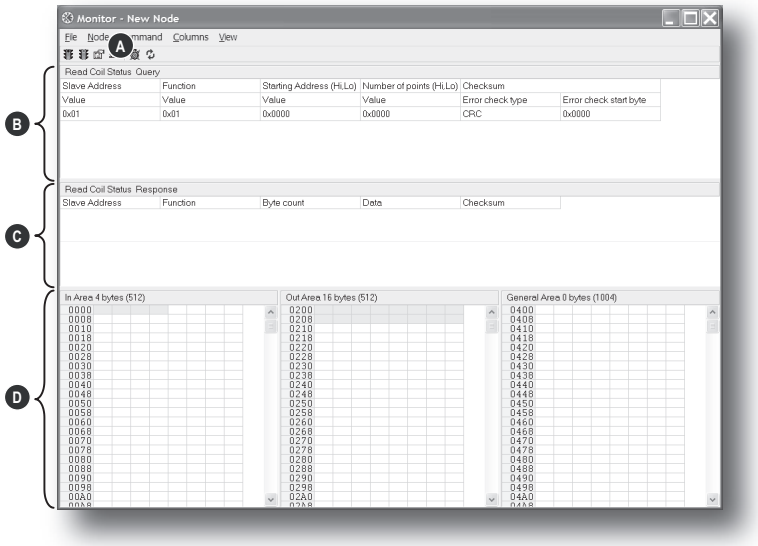


- **Generic Data Mode**

  The selected command (Transaction Produce) is sent to the sub-network. It is not possible to monitor any responses etc. generated by other nodes.



Navigating the Node Monitor



A: Pull-down Menu & Toolbar Icons

See 10-3 "Pull-Down Menu" and 10-4 "Toolbar Icons"

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 only)

This section holds the response to the selected Command.

D: Monitor Section

This section displays the data associated with the node. Areas in dark grey are reserved for the Status & Control Registers, and areas displayed in light grey represents the data that is used by the node.

The data displayed in this section will be refreshed based on the refresh-icons in the toolbar. For more information, see 10-4 "Toolbar Icons"

Pull-Down Menu

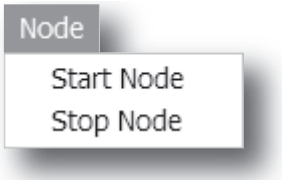
File

There is only one entry in this menu:

- **Exit**
  This will close the Node Monitor. Note however that if the node has been disabled using 'Stop Node' (see below), it will not resume data exchange until enabled again using 'Start node'.

Node

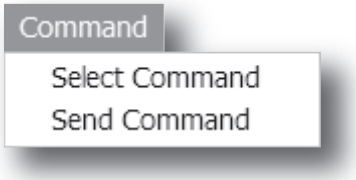
This menu controls the data exchange for the node. This feature can help isolate problems associated with a particular node.



- **Start Node**
  Enable the transactions associated with the node.
- **Stop Node**
  Disable the transactions associated with the node.

Command

This menu is used to specify and issue a command manually.



- **Select Command**
  Select a command to be sent on the sub-network.
- **Send Command**
  Send the specified command to the sub-network.

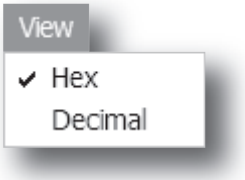
Columns

This menu specifies the number of columns in the Monitor Section.

- **Free**
  The number of columns depends on the width of the window.
- **8 Multiple**
  The number of columns will be fixed to 8.

View

This menu specifies the data representation in the Monitor Section.



- **Hex**
  Display the data in hexadecimal format.
- **Decimal**
  Display the data in decimal format.

Toolbar Icons

The toolbar features icons for the most commonly used functions.

- **Start Node & Stop Node**

  These icons corresponds to the functions in the 'Node'-menu.

  See also 10-3 "Node".


Start


Stop

- **Select Command & Send Command**

  These icons corresponds to the functions in the 'Command'-menu.

  See also 10-3 "Command".


Select


Send

- **Resume Refresh & Stop Refresh**

  When enabled, the data displayed in the Monitor Section will be refreshed cyclically. When disabled, i.e. stopped, the data will have to be refreshed manually using the 'Refresh'-icon (see below).


Stop


Resume

- **Refresh**

  When clicking on this icon, the data displayed in the Monitor Section will be refreshed.


Refresh

Data Logger

General

This feature allows the sub-network traffic to be logged into a buffer for examination. This may provide valuable information when debugging the lowest levels of the sub-network communication.

Note that the logger function is part of the gateway itself and is separate from the ABC Config Tool. This means that logging can be performed even if the gateway is physically disconnected from the PC running the ABC Config Tool.

Operation

Start & Stop Logging

- **Start logging**

  Select 'Start Logging' in the 'Tools'-menu. ABC Config Tool will then prompt for the desired mode of operation, see below.

- **Stop logging**

  Select 'Stop Logging' in the 'Tools'-menu. This will open the log-window, see below.

Modes of Operation

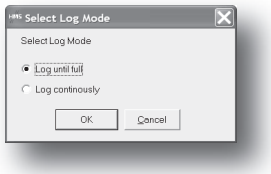
Select the desired mode of operation and click 'OK' to start logging data.

- **Log until full**

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

- **Log continuously**

  Data will be logged continuously until logging is stopped by clicking 'Stop Logging'. The log-buffer will contain the most recent data.

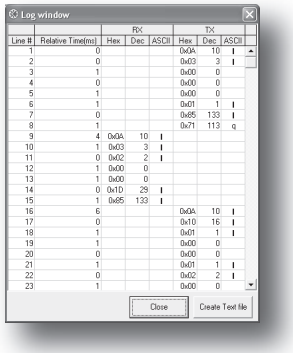


Log Window

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.

The data may optionally be saved in ASCII text format by clicking 'Create Text file'.

Click 'Close' to exit.



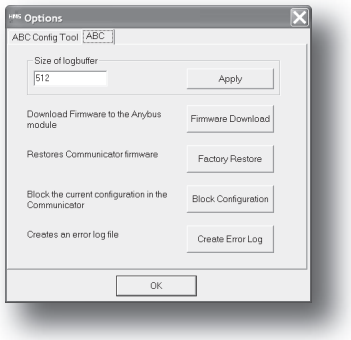
Configuration

By default, the log-buffer can hold 512 bytes of data in each direction. To specify a different size for the buffer, select 'Options' in the 'Tools'-menu.

A window with various settings will appear. Select the 'ABC'-tab, and enter the desired number of buffer entries under 'Size of logbuffer' (valid settings range from 1...512).

Click 'Apply' to validate the new settings.

Click 'OK' to exit.



Configuration Wizards

General

When creating a new sub network configuration, the ABC Config Tool provides a choice between starting out with a blank configuration, or using a predefined template, a.k.a a wizard.

The wizard automatically creates a sub-network configuration based on information supplied by the user, i.e the user simply has to "fill in the blanks". Note however that this will only work when the sub-network fits the wizard profile; in all other cases the 'Blank Configuration' option must be used.

Selecting a Wizard Profile

The following window appears each time the ABC Config Tool is started, or upon selecting the 'New' entry in the 'File'-menu (unless it has been disabled in the 'Options'-menu, see 3-3 "Tools").

Currently, the following wizards are available:

- **ABCC ExtLink Wizard**

  This wizard is intended for use with the Anybus-CompactCom Modbus-RTU fieldbus communication module.

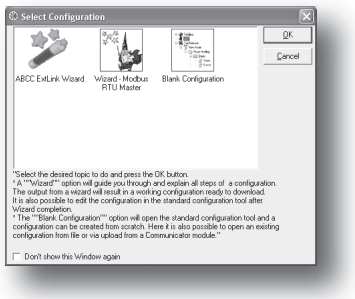
- **Wizard - Modbus RTU Master**

  This option is suitable for Modbus RTU-based networks.

  See also 12-2 "Wizard - Modbus RTU Master".

- **Blank Configuration**

  This option creates an empty configuration.



Highlight the desired wizard and click 'OK' to continue.

Wizard - Modbus RTU Master

This wizard can be used to create a Modbus-RTU-based network configuration based on certain information about the sub-network. The on-line help system explains each configuration step in detail.

- **Important Notes:**

  Many OEM devices do not fully comply with the Modbus standard. For example, they may implement a variation of this standard or be limited to the use of specific Modbus commands other than the ones used by this wizard. In all cases, the user should consult the documentation of the devices that shall be used on the sub-network for information about their serial communication requirements, and if necessary contact the manufacturer of the device to obtain further information about the serial communication protocol.

  In the event that the wizard doesn't handle a particular Modbus command required by a device, it is possible to specify this command manually as a transaction in the ABC Config Tool.

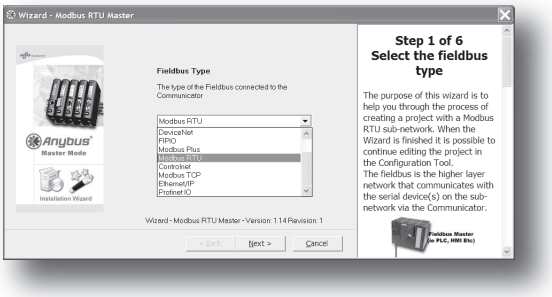
Using this wizard involves the following steps:

Step 1: Communicator Type

Select 'Modbus RTU'.

Click 'Next' to continue.

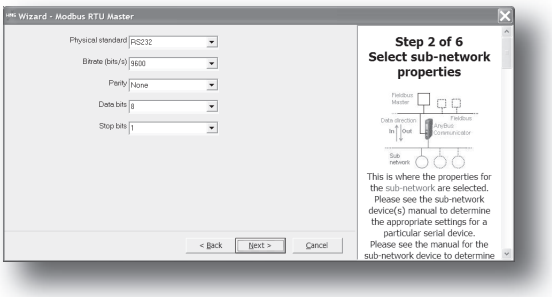
Tip: It is possible to return to a previous menu at any time without losing any settings by clicking 'Previous'.



Step 2: Physical Settings

Select the physical properties of the sub network.

Click 'Next' to continue.



Steps 3 - 6

Consult the on line help system for further information.

Control and Status Registers

General

The Control- and Status Registers are disabled by default, but can be enabled using the ABC Config Tool (see 4-7 “Status / Control Word”). These registers form an interface for exchanging status information between the sub-network and the fieldbus control system.

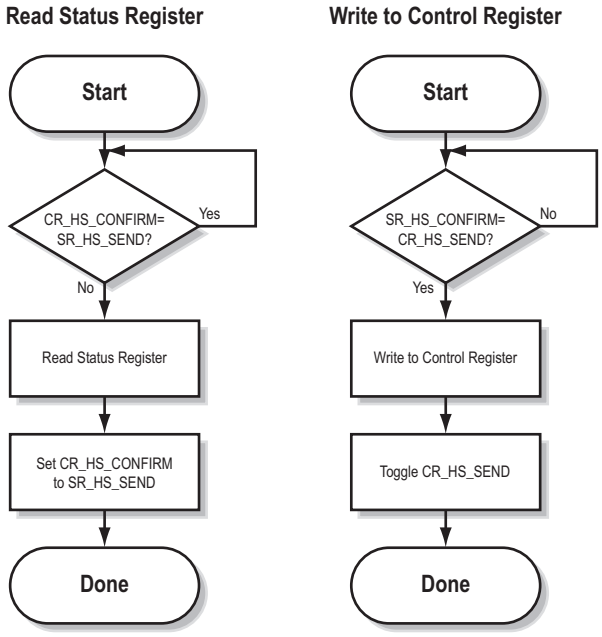
The main purpose of these registers is to...

- Report sub-network related problems to the fieldbus control system
- Ensure that only valid data is exchanged in both directions
- Enable the fieldbus control system to start/stop data exchange with selected nodes on the sub-network

If enabled, these registers are located in the Input and Output data areas, which means that they can be accessed from the fieldbus just like any other data in these areas. Their location can be specified freely, however keep in mind that the memory locations occupied by these registers cannot be used for regular data exchange.

Handshaking Procedure

A special handshaking procedure, which is illustrated in the two flowcharts below, must be followed when accessing these registers to ensure that both parts receive proper information.



Data Consistency

The 'Data Valid'-bits in the Control- and Status Registers are used to ensure data consistency during start-up and fieldbus off-line/on-line transitions.

If the 'Status / Control Word'-parameter in ABC Config Tool is set to 'Enabled', the gateway will wait for the fieldbus control system to set the 'Data Valid'-bit in the Control Register before it starts exchanging data on the sub-network.

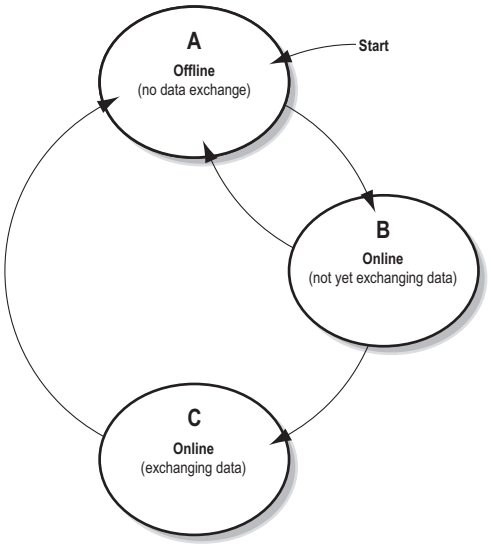
If the same parameter is set to 'Disabled' or 'Enabled but no startup lock', communication will start as soon as the fieldbus goes online.

State Machine

The fieldbus network participation can be described using a state machine as described below.

A: Offline (No data exchange)

1. Clear the 'Data Valid'-bit in the Control Register.
2. Write initial data to the Output Area according to the sub-network configuration.
3. Wait until the fieldbus control system and the gateway are online on the fieldbus network, and shift to state B.



B: Online (Not yet exchanging data)

4. Wait until the 'Data Valid'-bit in the Status Register is cleared by the gateway.
5. Set the 'Data Valid'-bit in the Control Register.
6. When the 'Data Valid'-bit in the Status Register is set by the gateway, shift to state C.
7. If the gateway goes offline on the fieldbus, shift to state A.

C: Online (Exchanging data)

Exchanging valid data in both directions.
If the gateway goes offline on the fieldbus, shift to state A.

Note: The gateway cannot spontaneously clear the 'Data Valid'-bit in the Status Register.

Latency

The 'Data Valid'-bit in the Status Register may in some cases be delayed. This latency can be caused by a missing node or a bad connection to a node with a long timeout value assigned to it.

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

Status Register Contents (Gateway to Control System)

General Information

As mentioned previously, the Status Register indicates the current status of the gateway towards the fieldbus control system.

bit(s)	Name	Description
15	Send (SR_HS_SEND)	These bits control the handshaking towards the fieldbus control system. See also... - 13-1 “Handshaking Procedure” - 13-5 “Control Register Contents (Control System to Gateway)”
14	Confirm (SR_HS_CONFIRM)	
13	Data Valid (Master Mode Only)	This bit is set when all transactions have been executed successfully at least once. Once set, it will not change. 1: Data Valid 0: Data not Valid **Note:** This bit is not used in Generic Data Mode.
12... 8	Status Code	This field holds the last status report from the gateway. See also... - 13-3 “Status Codes in Master Mode” - 13-4 “Status Code in Generic Data Mode”
7... 0	Data	

Note: Internally, this is treated as a Motorola-format word (i.e. MSB first). If the higher level network uses a different byte order, the upper and lower bytes will appear swapped.

Status Codes in Master Mode

(This table is valid only in Master Mode).

Code	Condition	Type	Data	Description
0x00	Re-transmission Counter Updated	Warning	Counter	The number of re-transmissions on the sub-network has increased. If this problem persists, this may eventually trigger a Single- or Multiple Node(s) Missing condition.
0x01	Single Node Missing	Error	Slave address	A single node is missing.
0x02	Multiple Nodes Missing	Error	Number of nodes	Multiple nodes are missing.
0x03	Buffer Overrun	Warning	Slave address	A node returned more data than expected.
0x04	Other Error	Error	Slave address	Undefined error
0x1F	No Error	Warning	-	No errors

Note: Conditions of type ‘Error’ will eventually be followed by a ‘No Error’ condition when the cause has been resolved. Conditions of type ‘Warning’ are however considered informational and may not necessarily be followed by a ‘No Error’ condition later on.

Status Code in Generic Data Mode

(This table is valid only in Generic Data Mode).

Code	Condition	Type	Data	Description
0x00	Invalid Transaction Counter Updated	Error	Counter	The number of invalid transactions (i.e. received transactions which doesn't match any of the Consume-transactions defined in the sub-network configuration) has increased.
0x01	Frame Error	Warning	-	End character is enabled, but a message delimiter timeout occurs prior to receiving it.
0x02	Offline Timeout Counter Updated	Error	Counter	The of number of timed out Consume-transactions has increased. See also... - 6-4 "Consume-Transactions" (Offline timeout time)
0x03	Buffer Overrun	Warning	-	A node returned more data than expected - or - the gateway was unable to finish processing a message prior to receiving a new one.
0x04	Other Error	Error	-	Undefined error
0x1F	No Error	Warning	-	No errors

Note: Conditions of type 'Error' will eventually be followed by a 'No Error' condition when the cause no longer is detected. Conditions of type 'Warning' are however considered informational and may not necessarily be followed by a 'No Error' condition later on.

Control Register Contents (Control System to Gateway)

General Information

As mentioned previously, the Control Register can be used to synchronize data exchange and instruct the gateway to perform certain tasks related to the sub-network communication.

bit(s)	Name	Description
15	Confirm (CR_HS_CONFIRM)	These bits control the handshaking towards the gateway. See also... - 13-1 “Handshaking Procedure” - 13-3 “Status Register Contents (Gateway to Control System)”
14	Send (CR_HS_SEND)	
13	Data Valid	This bit controls data consistency (see 13-2 “Data Consistency”). 1: Output Area valid; exchange data on the sub-network 0: Output Area not valid; do not exchange data on the sub-network **Note:** This bit is only relevant if the Control/Status Registers are set as ‘Enabled’
12	Execute Command	If set, the specified command will be executed by the gateway (see below).
11... 8	Control Code	This field holds commands which can be executed by the gateway (see below). See also... - 13-5 “Control Codes in Master Mode” - 13-5 “Control Codes in Generic Data Mode”
7... 0	Data	

Note: Internally, this is treated as a Motorola-format word (i.e. MSB first). If the higher level network uses a different byte order, the upper and lower bytes will appear to be swapped.

Control Codes in Master Mode

(This table is valid only in Master Mode).

Code	Instruction	Data	Description
0x00	Disable Node	Actual node address	Disables the specified node.
0x01	Enable Node	Actual node address	Enables a previously disabled node.
0x02	Enable Nodes	Actual number of nodes to enable	Enables the specified number of nodes, starting from the first node in the configuration. Remaining nodes will be disabled.

Control Codes in Generic Data Mode

(No Control Codes are currently supported in this mode).

Advanced Fieldbus Configuration

General

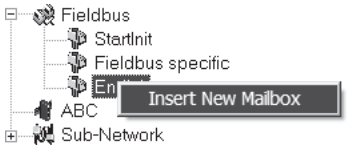
The fieldbus interface of the gateway consists of an embedded Anybus-S communication interface. Normally, the Anybus-S configuration settings are set up automatically by the gateway. However, advanced users can configure the Anybus-S card for specific features. This chapter assumes that the reader is familiar with the Anybus-S and it's application interface. For more information about the Anybus-S platform, consult the Anybus-S Parallel Design Guide.

The standard initialisation parameters are determined by the sub-network configuration. Information about the amount of input- and output data used for sub-network communication is used by ABC Config Tool to create the configuration message that sets the sizes of the input- and output data areas in the Dual Port RAM of the embedded Anybus-S interface. It is possible to add fieldbus specific mailbox messages to customize the initialisation. This is done in the Mailbox Editor, see below.

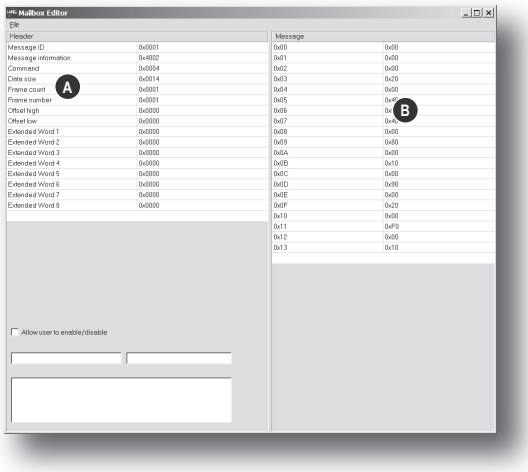
(A mailbox message is a HMS specific command structure used for low-level communication with an Anybus-S interface. Consult the Anybus-S Parallel Design Guide and the fieldbus appendix for the desired fieldbus for further information.)

Mailbox Editor

To add a mailbox message to the configuration, right-click on 'EndInit' and select 'Insert New Mailbox'.



A mailbox message consists of a Header section and a data section where the Header consists of 16 words (32 bytes) and the data section consists of up to 128 words (256 bytes). All fields are editable except the Message information field that is fixed to 0x4002, which means that only fieldbus specific mailbox messages can be entered here.



The mailbox message is presented as two columns; one contains header information (A), the other one contains the message data (B).

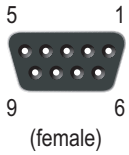
To add message data, simply change the Data size parameter in the header column (A), and the corresponding number of bytes will appear in the message data column (B).

For more information about fieldbus specific mailbox messages, consult the separate Anybus-S Fieldbus Appendix for the fieldbus you are using. For general information about the Anybus-S platform, consult the Anybus-S Design Guide.

Connector Pin Assignments

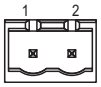
Fieldbus Connector (Modbus-RTU)

Pin	Signal	Description
Housing	Shield	Bus cable shield, connected to PE
1	-	-
2	TxD	Transmit data (RS-232)
3	RxD	Receive data (RS-232)
4	-	-
5	GND	Ground, galvanically isolated
6	+5V	+5V, galvanically isolated
7	A	A-Line (RS-485)
8	B	B-Line (RS-485)
9	CAN_V+	Opional CAN external power supply



Power Connector

Pin	Description
1	+24V DC
2	GND

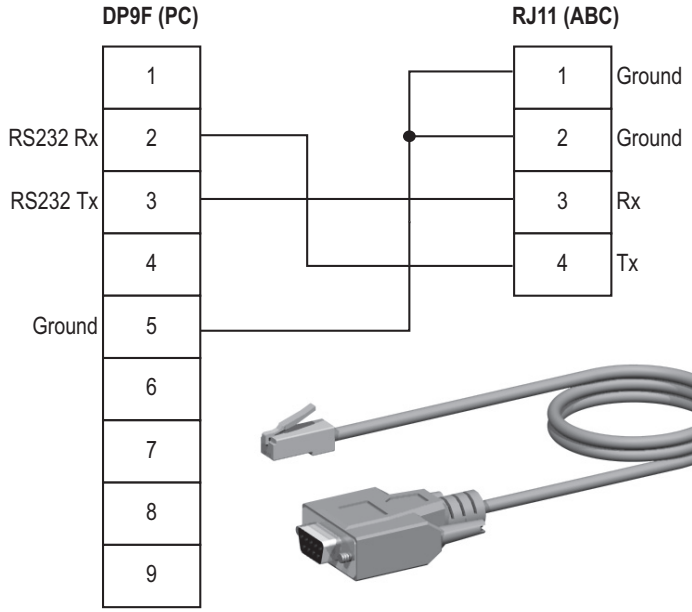


Notes:

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

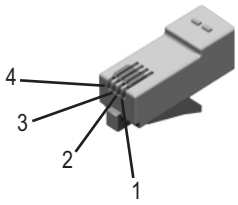
PC Connector

Configuration Cable Wiring



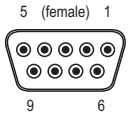
RJ11 (ABC)

Pin	Description
1	Signal ground
2	
3	RS232 Rx (Input)
4	RS232 Tx (Output)



DB9F (PC)

Pin	Description
1	-
2	RS232 Rx (Input)
3	RS232 Tx (Output)
4	-
5	Signal Ground
6 - 9	-



Sub-network Interface

General Information

The sub-network interface provides for RS232, RS422 and RS485 communications. Depending on the configuration specified in the ABC Config Tool, different signals are activated in the sub-network connector.

Bias Resistors (RS485 Only)

When idle, RS485 enters an indeterminate state, which may cause the serial receivers to pick up noise from the serial lines and interpret this as data. To prevent this, the serial lines should be forced into a known state using pull-up and pull-down resistors, commonly known as bias resistors.

The bias resistors forms a voltage divider, forcing the voltage between the differential pair to be higher then the threshold for the serial receivers, typically >200mV.

Note that bias resistors shall only be installed on one node; installing bias resistors on several nodes may compromise the signal quality on the network and cause transmission problems.

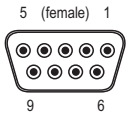
Termination (RS485 & RS422 Only)

To avoid reflections on the serial lines, it is important to properly terminate the sub-network by placing termination resistors between the serial receivers near the end nodes.

The resistor value should ideally match the characteristic impedance of the cable, typically 100... 120R.

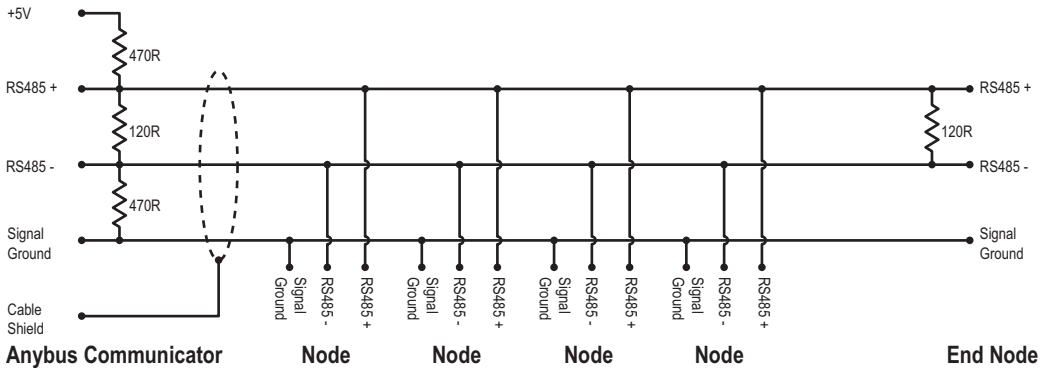
Connector Pinout (DB9F)

Pin	Description	RS232	RS422	RS485
1	+5V Output(100mA max)	✓	✓	✓
2	RS232 Rx	✓		
3	RS232 Tx	✓		
4	(reserved)			
5	Signal Ground[a]	✓	✓	✓
6	RS422 Rx +		✓	
7	RS422 Rx -		✓	
8	RS485 + /RS422 Tx+		✓	✓
9	RS485 - /RS422 Tx-		✓	✓
(housing)	Cable Shield	✓	✓	✓

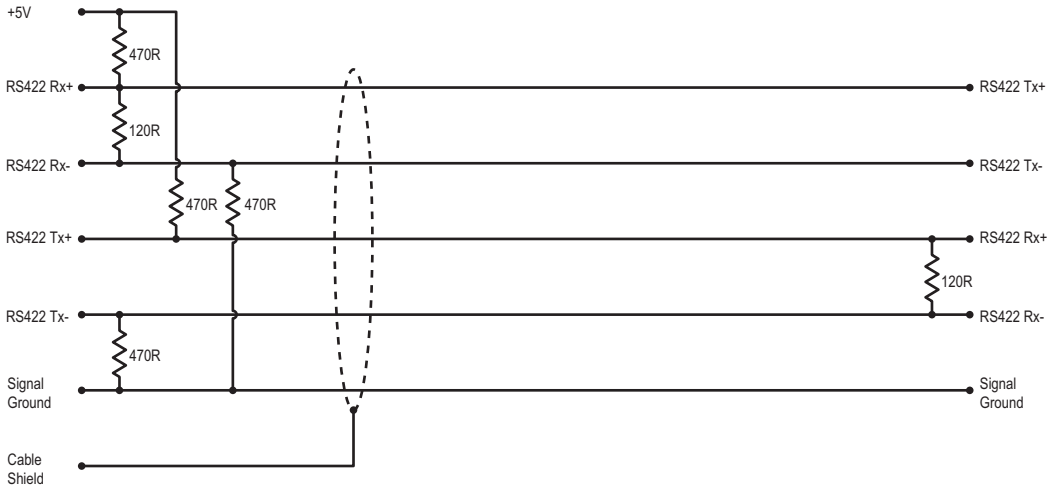


a. Connecting this signal directly Protective Earth (PE) of other nodes may, in case of grounding loops etc., cause damage to the on-board serial transceivers. It is therefore generally recommended to connect it only to Signal Ground (if available) of other nodes.

Typical Connection (RS485)

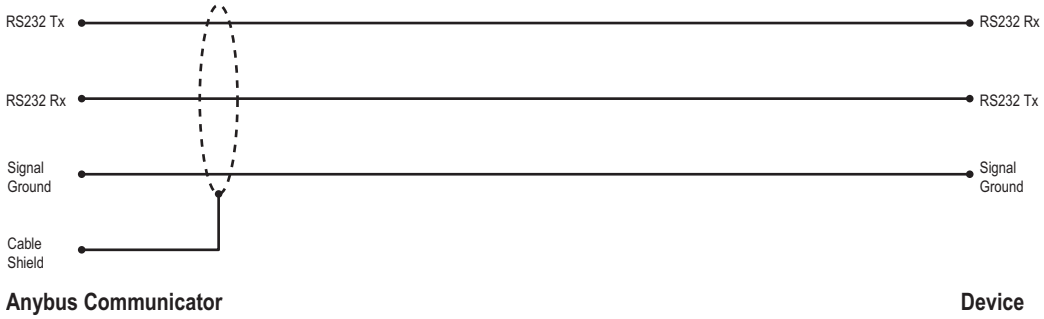


Typical Connection (RS422 & 4-Wire RS485)



Note: Bias resistors are normally not needed on RS422, but may be required when using 4-wire RS485.

Typical Connection (RS232)



Technical Specification

Mechanical Properties

Housing

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

Dimensions

120 mm x 75 mm x 27 mm, L x W x H (inches: 4.72" x 2.95" x 1.06"; L x W x H)

Electrical Characteristics

Power Supply

Power: 24V ± 10%

Power Consumption

Maximum power consumption is 280mA on 24V. Typically around 100mA

Environmental Characteristics

Relative Humidity

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

Temperature

Operating:	±0°C to +55°C
Non Operating:	-25°C to +85°C

Regulatory Compliance

EMC Compliance (CE)

This product is in accordance with the EMC directive 89/336/EEC, with amendments 92/31/EEC and 93/68/EEC through conformance with the following standards:

- **EN 50082-2 (1993)**

EN 55011 (1990)	Class A

- **EN 61000-6-2 (1999)**

EN 61000-4-3 (1996)	10V/m	
EN 61000-4-6 (1996)	10V/m	(all ports)
EN 61000-4-2 (1995)	±8kV	Air Discharge
	±4kV	Contact discharge
EN 61000-4-4 (1995)	±2kV	Power port
	±1kV	Other ports
EN 61000-4-5 (1995)	±0.5kV	Power ports (DM/CM)
	±1kV	Signal ports

UL/c-UL compliance

The certification has been documented by UL in file E214107.

Troubleshooting

Problem	Solution
Problem during configuration Upload / Download. The Config Line “led” turns red in the ABC Config Tool.	• Serial communication failed. Try again
The serial port seems to be available, but it is not possible to connect to the gateway	• The serial port may be in use by another application. Exit the ABC Config Tool and close all other applications including the ones in the system tray. Try again • Select another serial port Try again
Poor performance	• Right click ‘Sub-Network’ in the Navigation window and select ‘Sub-Network Status’ to see status / diagnostic information about the sub network. If the gateway reports very many re-transmissions, check your cabling and / or try a lower baud rate setting for the sub network (if possible). • Is the Sub-Net Monitor in the ABC Config Tool active? The sub-network monitor has a negative influence on the overall performance of the gateway, and should only be used when necessary. • Is the Node Monitor in the ABC Config Tool active? The node monitor has a negative influence on the overall performance of the gateway, and should only be used when necessary.
No sub-network functionality	• Use the ‘Data logger’-functionality to record the serial data communication on the sub-network. • If no data is being transmitted, check the configuration in ABC Config Tool. • If no data is received, check the sub-network cables. Also verify that the transmitted data is correct.

ASCII Table

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NUL 0	SOH 1	STX 2	ETX 3	EOT 4	ENQ 5	ACK 6	BEL 7	BS 8	HT 9	LF 10	VT 11	FF 12	CR 13	SO 14	SI 15
1x	DLE 16	DC1 17	DC2 18	DC3 19	DC4 20	NAK 21	SYN 22	ETB 23	CAN 24	EM 25	SUB 26	ESC 27	FS 28	GS 29	RS 30	US 31
2x	(sp) 32	! 33	" 34	# 35	$ 36	% 37	& 38	' 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47
3x	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63
4x	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79
5x	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95
6x	` 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111
7x	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	DEL 127

AnyBus® Communicator Protocol Appendix

Modbus

DOC. ABC-APPENDIX-MB Rev.0.91

HMS INDUSTRIAL NETWORKS AB
PIELEFELTSGATAN 93 - 95
S - 302 50 HALMSTAD
SWEDEN

PHONE: +46 35 17 29 00
FAX: +46 35 17 29 09
e-mail: info@hms-networks.com
web: www.hms-networks.com

Revision notes

Revision	Date	Description	Responsible
0.9	2001-06-20	Created	Edk/MaB
0.91	2001-07-30	Updated numbering	Edk

Preface

The data and illustrations found in this manual are not binding. We reserve the right to modify our products in line with our policy of continuous product development. The information in this manual is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB.

HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

The product and technology described in this document is patent pending in the following countries:
USA, Canada, Japan, Belgium, Denmark, Finland, France, Greece, Ireland, Italy, Luxemburg, Monaco, Netherlands, Portugal, Switzerland, Lichtenstein, Spain, United Kingdom, Sweden, Germany and Austria.

About the AnyBus Communicator Modbus Appendix

This fieldbus appendix contains fieldbus specific information about the Modbus protocol for the AnyBus Communicator. For more information about the AnyBus Communicator, please refer to the AnyBus Communicator User Manual, DOC. ABC-UM.

If technical support is required, please contact the AnyBus Support Centre:

Europe (Sweden)
Phone: +46 (0) 35 - 17 29 20
E-mail: support@hms.se

Germany
Phone: +49-721-96472-0
E-mail: ge-support@hms-networks.com

North America
Phone: +1-773-404-2271
Toll Free: 888-8-ANYBUS
E-mail: us-support@hms-networks.com

Japan
Phone: +81-45-478-5340
E-mail: jp-support@hms-networks.com

Conditions for trademark use

Please contact HMS for further information.

Related documents

Document name	Author	Document ID	Revision
AnyBus Communicator User Manual	Edk/MaB	ABC-UM	0.91
Modbus Protocol Reference Guide	Modicon	PI-MBUS-300	J

Abbreviations

Important abbreviations used in this manual:

Abbreviation	Description
AB-C	AnyBus Communicator
ABcCon	AnyBus Communicator Configuration Software

1 Appendix for Modbus

1.1 Introduction

When configured for Modbus protocols, the AnyBus Communicator supports Modbus RTU, Modbus ASCII and Modbus Generic. Some basic knowledge is needed, as to understand how to use the Anybus-C for configuration of the Modbus network.

The Modbus standard was created by Modicon for communication between controllers and other devices. The transactions on the Modbus network are of master/slave type, and are named “query” and “response”. One single master sends the queries. All transactions on the network have got a frame structure where one part is common for both Modbus RTU and Modbus ASCII. This is illustrated in figure 1.

Figure 1: Modbus frame layout

The main difference between Modbus RTU and Modbus ASCII is that in RTU all hexadecimal values are represented with one byte and in ASCII they are represented with two bytes. Another difference is the start and stop signs that envelope the frame.

Modbus RTU

Each byte in the Modbus RTU message represents a hexadecimal value between 0 and 255. The frame looks exactly like the one mentioned earlier and there is always an interval of 3.5 silent characters between the frames. CRC (Cyclical Redundancy Check) is used for error checking. Multiple Modbus transactions would appear on the physical interface like this:

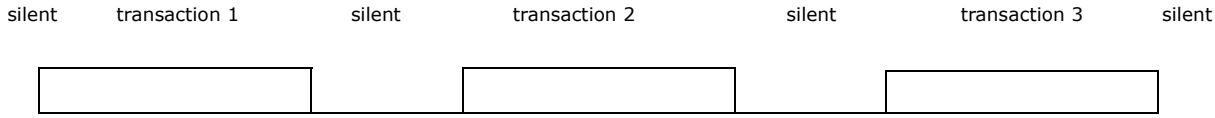


Figure 2: Modbus RTU transactions

Modbus ASCII

Each byte in the Modbus ASCII message represents one hexadecimal digit i.e. 0-9, A-F. This means that two bytes are used to represent each hexadecimal value (0x00-0xFF). For example, the value 0x2A is represented like this: 1st byte: "2" (0x32), 2nd byte: "A" (0x41). Start and stop characters are added to the frame from Figure 1. A colon, ":", is used as start character and "CR""LF" are used as stop characters. Longitudinal Redundancy Check (LRC) is used for error checking. A complete Modbus ASCII transaction would appear on the physical interface like this:

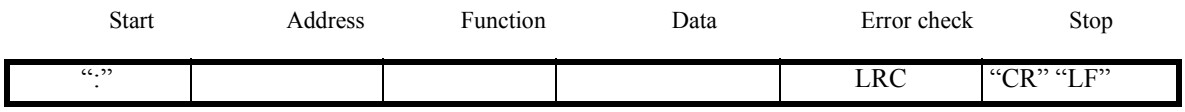


Figure 3: Modbus ASCII transactions

1.2 AB-C as Modbus master

When the AB-C acts as master on the Modbus network it uses a scan-list for communication with the different slaves on the network as described in the "Sub-network, Master" section. The scan-list is defined using AbcCon.When Modbus protocol is selected you add pre-defined modbus commands to the scan-list simply by selecting them from a list. The next chapter describes all the supported Modbus commands and what you need to think about when using them in the AB-C.

What makes the AB-C Modbus specific is the way transactions are used in Modbus commands and what the transactions consist of. One important issue in the AB-C is that the data must be of a pre-defined length. The example below shows how this works.

Basically, if we exclude the start character and stop character in the Modbus frame, each Modbus frame consists of two one-byte objects, one data object, maybe some more one- or two-byte objects and one error-check object. Lets take an example using Modbus RTU. Example. Read Holding Register (0x03) to node 0x05.

This command is built up like this:

Query:

Modbus frame	**Address**	**Function**	**Data**		**Error check**
Frame contents	0x05	0x03	“Starting Address”	“No. of points”	CRC
AB-C frame	“One byte object”	“One byte object”	Two byte object	Two byte object	“Error check object”

Table 1: Query

What you as a user must do here is to enter "Starting address" and "No. of points". These two parameters are represented as two-byte objects that you enter values into using AbcCon. This command will ask the slave for the same registers every time the command is sent and unless something goes wrong, the slave will answer with the same amount of data every time.

Response:

Modbus frame	**Address**	**Function**	**Data**		**Error check**
Frame contents	0x05	0x03	“Byte count”	“Data”	CRC
AB-C frame	“One byte object”	“One byte object”	One byte object	“Data Object”	“Error check object”

Table 2: Response

In the response the data section in the Modbus frame needs to be filled in. Here the data section is represented by a one-byte object (byte count) and a data object (data). To match the query you need to add a value into the one-byte object that is two times the No. of points value you entered in the query. Furthermore, the data object needs a starting address and a length where the length should match the one-byte object (byte count).

As you can see the requested data is always of the same length and therefore the data object in the response is also always of the same length. Should less bytes than specified arrive then the response is considered to have an error and a re-transmission of the query will occur if this command is configured for re-transmission. The same handling is done if more data than expected arrives.

1.3 Modbus commands

The following tables list all Modbus commands that are supported by the AB-C. For each command there is an explanation about what actions you as user need to take on the query and response.

Code	Name
01	Read Coil Status
02	Read Input Status
03	Read Holding Registers
04	Read Input Registers
05	Force Single Coil
06	Preset Single Register
07	Read Exception Status
11	Fetch Comm. Event Ctr
12	Fetch Comm. Event Log
15	Force Multiple Coils
16	Preset Multiple Registers
17	Report Slave ID
20	Read General Reference
21	Write General Reference
22	Mask Write 4X Register
23	Read/Write 4X Register
24	Read FIFO Queue

Table 3: Modbus Commands

1 Read Coil Status		
Query	Starting Address	2 byte value where you enter the first address of the requested coils.
	No. of Points	2 byte value where you enter the number of coils to read.
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object where you enter the length of the received data and the destination address.

Table 4: Read Coil Status

2 Read Input Status		
Query	Starting Address	2 byte value where you enter the first address of the requested discrete inputs.
	No. of Points	2 byte value where you enter the number of inputs to read.
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object where you enter the length of the received data and the destination address.

Table 5: Read Input Status

3 Read Holding Registers		
Query	Starting Address	2 byte value where you enter the first address of the requested registers.
	No. of Points	2 byte value where you enter the number of registers to read.
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object where you enter the length of the received data and the destination address.

Table 6: Read Holding Registers

4 Read Input Registers		
Query	Starting Address	2 byte value where you enter the first address of the requested registers.
	No. of Points	2 byte value where you enter the number of registers to read.
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object where you enter the length of the received data and the destination address.

Table 7: Read Input Registers

5 Force Single Coil		
Query	Data	Data object with 4 bytes. The fieldbus master should enter Coil Address and Force Data in these four bytes.
Response	Data	Data object with 4 bytes. The slave returns Coil Address and Forced Data in these 4 bytes.

Table 8: Force Single Coil

6 Preset Single Register		
Query	Data	Data object with 4 bytes. The fieldbus master should enter Register Address and Preset Data in these four bytes.
Response	Data	Data object with 4 bytes. The slave returns Register Address and Preset Data in these 4 bytes.

Table 9: Preset Single register

7 Read Exception Status		
Query	-	-
Response	Data	Data object with 1 byte. The slave returns the 8 Exception Status Coils in this byte.

Table 10: Read Exception Status

11 Fetch Comm. Event Counter		
Query	-	-
Response	Data	Data object with 4 bytes. The slave returns Status and Event Count in these 4 bytes.

Table 11: Fetch Comm. Event Counter

12 Fetch Comm. Event Log		
Query	-	-
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object with 6-70 bytes. The slave returns Status, Event Count, Message Count and the Event Log in these bytes.

Table 12: Fetch Comm. Event Log

15 Force Multiple Coils		
Query	Coil Address	2-byte value where you enter the reference of the first coil to be forced.
	Quantity of Coils	2-byte value where you enter the number of coils to force.
	Byte Count	1-byte value where you enter the number of data bytes.
	Data	Data object where you enter the length of the data to send and the source address. The fieldbus master should enter the force data in these bytes.
Response	Data	Data object with 4 bytes. The slave returns Coil Address and Quantity of Coils forced in these bytes.

Table 13: Force Multiple Coils

16 Preset Multiple Registers		
Query	Starting Address	2 byte value where you enter the address of the first register to be preset.
	No. of Registers	2 byte value where you enter the number of registers to preset.
	Byte Count	1 byte value where you enter the number of data bytes.
	Data	Data object where you enter the length of the data to send and the source address. The fieldbus master should enter the preset data in these bytes.
Response	Data	Data object with 4 bytes. The slave returns Starting Address and No. of Registers preset in these bytes.

Table 14: Preset Multiple Registers

17 Report Slave ID		
Query	-	-
Response	Byte Count	1 byte value where you enter the number of expected data bytes.
	Data	Data object where you enter the length of the recieved data and the destination address. The slave returns slave ID, Run Indicator Status and Additional Data in these bytes.

Table 15: Report Slave ID

20 Read General Reference		
Query	-	-
Response	-	-

Table 16: Read General reference

21 Write General reference		
Query	-	-
Response	-	-

Table 17: Write General Reference

22 Mask Write 4X Registers		
Query	Data	Data object where you enter the length of the data to send and the source address. The fieldbus master should enter the Reference Address, AND mask and OR mask in these bytes.
Response	Data	Data object where you enter the number of expected data bytes and the destination address. The slave returns Reference Address, AND mask and OR mask in these bytes.

Table 18: Mask Write 4X Registers

23 Read/Write 4X Registers		
Query	-	-
Response	-	-

Table 19: Read/Write 4X Registers

24 read FIFO Queue		
Query	-	-
Response	-	-

Table 20: Read FIFO Queue

This page is intentionally left blank.

If you have any comments about this documentation, please take a few minutes to fill out this form, and let us know about your opinions. These comments will help us improve our work, and make us aware of what customers of our products may find good, faulty or even missing.

Document title and revision:_____

Your name and company:_____

Phone:_____

E-mail:_____

Comments:

Text and illustrations:

_____

_____

_____

What information is missing or unclear?:

_____

_____

_____

Other comments:

_____

_____

_____

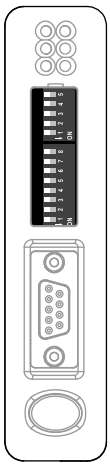
Send your comments to:

HMS Industrial Networks AB
Support Department
Pilefeltsgatan 93-95
302 50 Halmstad
SWEDEN

You may also mail or fax your comments:

E-mail: support@hms-networks.com
Fax: +46 (0)35 172909

Configuration Switches

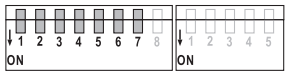


The configuration switches determines the basic communication settings for the Modbus interface. Normally, these switches are covered by a plastic hatch. When removing the hatch, avoid touching the circuit boards and components. If tools are used to open the hatch, be cautious.

Note that these settings cannot be changed during runtime, i.e. the gateway must be restarted in order for any changes to have effect.

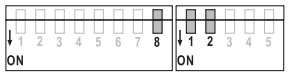
Node Address

Node Address	Sw. 1	Sw. 2	Sw. 3	Sw. 4	Sw. 5	Sw. 6	Sw. 7
(reserved)	OFF	OFF	OFF	OFF	OFF	OFF	OFF
1	OFF	OFF	OFF	OFF	OFF	OFF	ON
2	OFF	OFF	OFF	OFF	OFF	ON	OF
...	...	...	...	...	...	...	...
126	ON	ON	ON	ON	ON	ON	OFF
127	ON	ON	ON	ON	ON	ON	ON



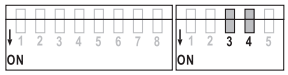
Baudrate Configuration

Baudrate	Sw. 8	Sw. 1	Sw. 2
(reserved)	OFF	OFF	OFF
1200 bps	OFF	OFF	ON
2400 bps	OFF	ON	OFF
4800 bps	OFF	ON	ON
9600 bps	ON	OFF	OFF
19200 bps (standard)	ON	OFF	ON
38400 bps	ON	ON	OFF
57600 bps	ON	ON	ON



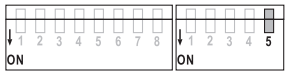
Parity & Stop Bits

Parity	Sw. 3	Sw. 4
(reserved)	OFF	OFF
No parity, 2 stop bits	OFF	ON
Even parity, 1 stop bit	ON	OFF
Odd parity, 1 stop bit	ON	ON



Physical Interface

Interface Type	Sw. 5
RS-485	OFF
RS-232	ON



Data Representation on Modbus RTU

General

The Input- and Output Data areas are mapped to Modbus registers 0... 1279 and Coils 0... 20479.

Supported Function Codes

The following function codes are supported:

Function Code	Modbus Function	Associated with Area(s)
1	Read Coil	Input- and Output Data Area (0x000... 0x3FF)
2	Read Input Discretes	
3	Read Holding Registers	
4	Read Input Registers	
5	Write Coil	Output Data Area (0x200... 0x3FF)
6	Write Single Register	
15	Force Multiple Coils	
16	Force Multiple Registers	
22	Mask Write Register	
23	Read/Write Registers	Input- and Output Data Area (0x000... 0x3FF)

Coil & Register Map

The Input & Output Data areas are mapped to coils and registers as follows:

Register #	Coil #	Memory Location	Area	Comments
1	1... 16	0x000... 0x001	Input Data area	-
2	17... 32	0x002... 0x003		
...	...	...		
255	4065... 4080	0x1FC... 0x1FD		
256	4081... 4096	0x1FE... 0x1FF		
257... 1024	4097... 16384	-	-	(reserved)
1025	16385... 16400	0x200... 0x201	Output Data area	-
1026	16401... 16416	0x202... 0x203		
...	...	...		
1279	20449... 20464	0x3FC... 0x3FD		
1280	20465... 20480	0x3FE... 0x3FF		

Note: Coils are mapped MSB first, i.e. coil 0 corresponds to bit 15 of register 0.

Supported Exception Codes

Exception Code	Name	Description
0x01	Illegal function	Function code not supported
0x02	Illegal data address	Invalid address in query
0x03	Illegal data value	Illegal data in request