X-gateway Interface Addendum
Profinet IO Slave

HMS

Connecting Devices™

Important User Information

This document is intended to provide a good understanding of the functionality offered by the Interface described here.

The reader is expected to be familiar with high level software design, and communication systems in general. The use of advanced interface-specific functionality may require in-depth knowledge of networking internals and/or information from the network specifications. In such cases, the persons responsible for the implementation of this product should either obtain the necessary specifications to gain sufficient knowledge, or alternatively limit the implementation in such a way that this is not necessary.

Liability

Every care has been taken in the preparation of this manual. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. 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 applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

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 Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the US and other countries.

Trademark Acknowledgements

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

!	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.

Table of Contents

Appendix A Customization

P. About This Document

P.1 How To Use This Document

This document describes network specific features and procedures needed when operating the PROFI-NET IO interface for the Anybux X-Gateway. For general information and operating instructions for the Anybus X-Gateway, consult the Anybus-X Generic Gateway User Manual.

The reader of this document is expected to be familiar with PROFINET networking technology, local area networks, and communication systems in general.

For further information, documentation etc., please visit www.anybus.com.

P.2 Related Documents

Document	Author
Anybus-X Generic Gateway User Manual	HMS
Anybus-S PROFINET IO Fieldbus Appendix	HMS
Anybus-X PROFINET IO Adapter Installation Sheet	HMS

P.3 Document History

Revision List

Revision	Date	Author(s)	Chapter(s)	Description
1.00	2005-10-04	PeP	-	First official release
1.01	2007-03-13	PeP	D	Added information about transport provider + misc. minor updates
1.02	2008-10-14	HeS	-	Misc. minor updates
2.00	May 2014	SDa	Multiple	New hardware and Anybus Configfiguration Manager

P.4 Conventions & Terminology

The following conventions are used throughout this manual:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The term 'Anybus-X' is used when referring to the Anybus-X Generic Gateway
- The term 'Adapter' is used when referring to the Anybus-X PROFINET IO Adapter.
- The term 'user manual' is used when referring to the Anybus-X Generic Gateway User Manual.
- Hexadecimal values are written in the format NNNNh, where NNNN is the hexadecimal value.
- 16/32 bit values are generally stored in Motorola (big endian) format unless otherwise stated.

P.5 Support

For general contact information and support, please refer to the contact and support pages at www.anybus.com.

1. About the Anybus-X PROFINET IO Adapter

1.1 General Description

The Anybus-X PROFINET IO Adapter is a multi purpose communication solution for the Anybus-X Generic Gateway; offering industrial protocol support as well as web and email capabilities.

The adapter exchanges data via two memory buffers a.k.a. the Input and Output Buffers. These buffers can be accessed via a built in webserver, or via industrial protocols such as PROFINET IO and Modbus/TCP.

Dynamic content capabilities allows data from the input/output buffers to be monitored on web pages, or included in email messages.

- **Input Buffer**

  This buffer holds data *from* the gateway, i.e. data *from* another network.

- **Output Buffer**

  This buffer holds data that shall be sent *through* the gateway *to* another network.
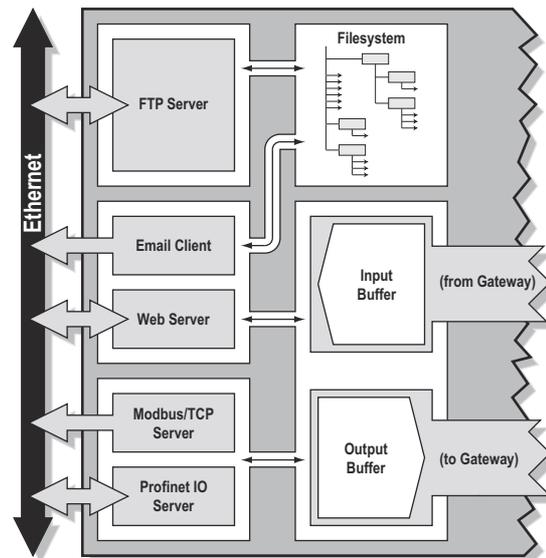
1.2 Features

General

- 10/100Mbit operation in full or half duplex
- Built in filesystem (1.4MB) w. per-user security framework
- Webserver with dynamic data capability
- Email Client with dynamic data capability
- FTP Server
- PROFINET IO Server
- Modbus/TCP Server (Read-only)
- Supports the Ethernet Transport Provider

Supported PROFINET Features

- Soft Real-Time (RT) communication
- Cyclic data exchange (10ms cycle time)
- Acyclic Data exchange (Record Data Requests)
- Up to 64 slots / 1 sub-slot
- Up to 512 bytes of I/O in each direction
- DCP support (Discovery and Configuration Protocol)

1.3 External View

Front View

1.3.1 PROFINET Status LED:s

LED	Colour	Indication
Gateway Status	See the gateway user manual for further details.	
LINK	Green	Link established
	Green, flashing	Receiving/Transmitting data
	Off	No link or power off
CS	Green	On line, Run • Connection with IO Controller established • IO Controller is in RUN state
	Green, 1 flash	On line, STOP • Connection with IO Controller established • IO Controller in STOP state
	Off	Off line • No connection with IO Controller
MS	Green	Initialized, no error
	Green, 1 flash	Diagnostic data available
	Green, 2 flashes	Blink. Used by engineering tools to identify the adapter.
	Red, 1 flash	Configuration Error • Too many modules/sub-modules • /O size derived from IO Controller configuration is too large • Configuration mismatch (no module, wrong module)
	Red, 3 flashes	No Station Name or no IP address assigned
	Red, 4 flashes	Internal error
	Off	No power or not initialized

Top-mounted Profinet interface
Gateway status
LINK
CS
MS
Not used
Gateway status
LINK
CS
MS
Not used
USB Gateway Config Connector
Bottom-mounted Profinet interface

Top-mounted interface

LAN
Gateway Power Connector
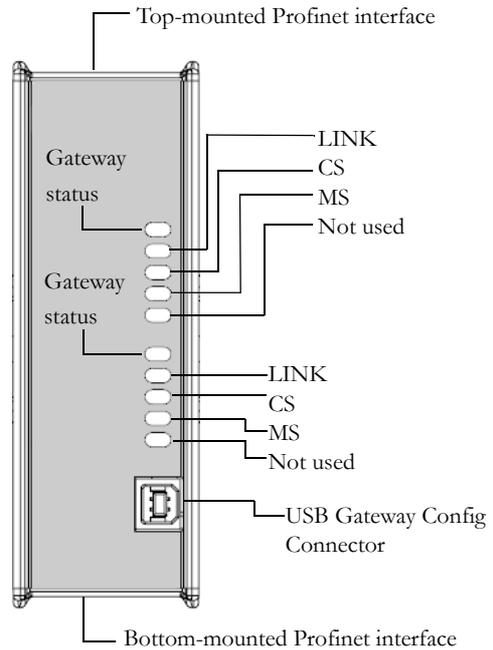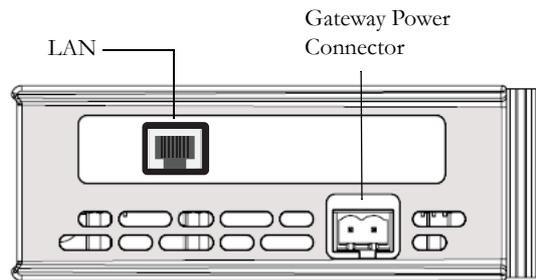
Bottom-mounted interface

LAN
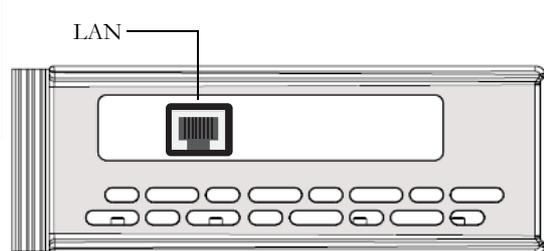
1.3.2 Connectors

Ethernet Connector (RJ45)

See E-50 "Ethernet Connector Pinout (RJ45)".

Gateway power connector

See the X-gateway User Manual for further details.

Gateway config connector

See the X-gateway User Manual for further details.

2. TCP/IP Configuration

2.1 Introduction

To be able to participate on the ethernet network, the adapter needs a valid TCP/IP configuration. The following sections describes some basic settings and their meaning.

Experienced users may wish to proceed to the following page, 2-13 "Configuring the TCP/IP Settings".

2.1.1 IP Address

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

Example:

```
IP Address 10000000 00001010 00000010 00011110 is written as 128.10.2.30
```

Certain IP address are reserved for special purposes and must not be used:

0.x.x.x	- (IP address where the first byte is zero)
127.x.x.x	- (IP address where the first byte is 127)
x.x.x.0	- (IP address where the last byte is zero)
x.x.x.255	- (IP address where the last byte is 255)

2.1.2 Subnet Masks

The IP address is divided into three parts - *net ID, subnet ID* and *host ID*. The subnet mask is a 32-bit binary pattern, where a set bit allocates a bit for network/subnet ID, and a cleared bit allocates a bit for the host ID. Just like the IP address, the subnet mask is commonly written in dotted-decimal notation.

Example:

To make the IP address 128.10.2.30 belong to subnet 128.10.2, the subnet mask shall be set to 255.255.255.0.

```
                                            Host ID
                                          ┌──────────┐
Subnet Mask:   11111111 11111111 1111111 00000000(255.255.255.0)
               └─────────────────────────┘
                    Net ID / Subnet ID
```

2.1.3 Gateways

To be able to establish communication between two devices both devices must belong to the same subnet. If not, the communication must be done through a gateway.

The gateway routes communication between two networks, i.e. it enables nodes on one network to access nodes on another. The gateway address specifies the IP address of the gateway/router.

2.2 Configuring the TCP/IP Settings

2.2.1 General

Internally, the network configuration is stored in a configuration file (called 'ethcfg.cfg'), which is read by the adapter during startup. In the unlikely event that this file is missing, it will attempt to retrieve its settings via DHCP or HICP. If no configuration has been received within 30 seconds, the adapter will halt and indicate an error on its status LEDs (the network configuration can however still be set using HICP, see below).

The adapter offers several ways to configure it's network settings:

- DHCP/BootP (See 2-14 "DHCP/BootP")
- Anybus IPconfig (HICP client) (See below)
- Via the Web Interface (See 4-18 "Web Interface")
- DCP (See 2-14 "DCP (Discovery and Control Protocol)")

2.2.2 HICP (Anybus IPconfig)

The adapter supports the HICP protocol used by the Anybus IPconfig utility from HMS, which can be downloaded free of charge from the HMS website. This utility may be used to configure the network settings of any Anybus product connected to the network. Note that if successful, this will replace the settings currently stored in the configuration file ('ethcfg.cfg').
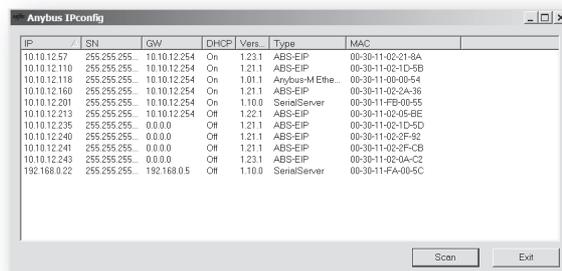
Upon starting the program, the network is scanned for Anybus products. The network can be rescanned at any time by clicking 'Scan'. In the list of detected devices, the Anybus-X adapter will appear as 'ABS-PRT'. To alter its network settings, double-click on its entry in the list.

A window will appear, containing the IP configuration and password settings. Validate the new settings by clicking 'Set', or click 'Cancel' to abort.

The configuration can be protected from unauthorized access by a password. To enter a password, click on the 'Change password' checkbox, and enter the password under 'New password'. When protected, any changes in the configuration requires that the user supplies a valid password.

When done, click 'Set'. The new IP configuration will now be stored in the configuration file ('ethcfg.cfg').
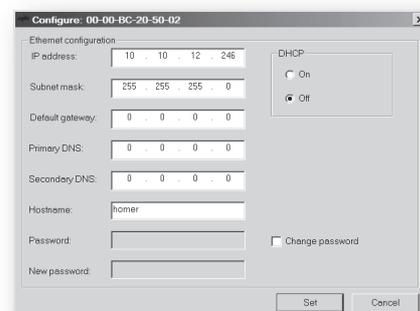
2.2.3 DHCP/BootP

The adapter can retrieve its TCP/IP settings from a DHCP or BootP server. Note that if successful, this will replace the settings currently stored in the configuration file ('ethcfg.cfg').

If no DHCP server is found, the adapter will fall back on it's current settings (i.e.the settings currently stored in the configuration file). If no current settings are available (i.e. in the unlikely event that the configuration file 'ethcfg.cfg' is missing), the adapter will halt and indicate an error on its status LEDs (the network configuration can however still be set using HICP, see 2-13 "HICP (Anybus IPconfig)").

2.2.4 DCP (Discovery and Control Protocol)

The adapter supports the DCP protocol, allowing a PROFINET IO Controller/Supervisor to change the network settings during runtime. If successful, this will replace the settings currently stored in 'ethcfg.cfg'.

2.2.5 Startup Sequence

During startup, the adapter establishes the TCP/IP configuration according to the following flowchart:
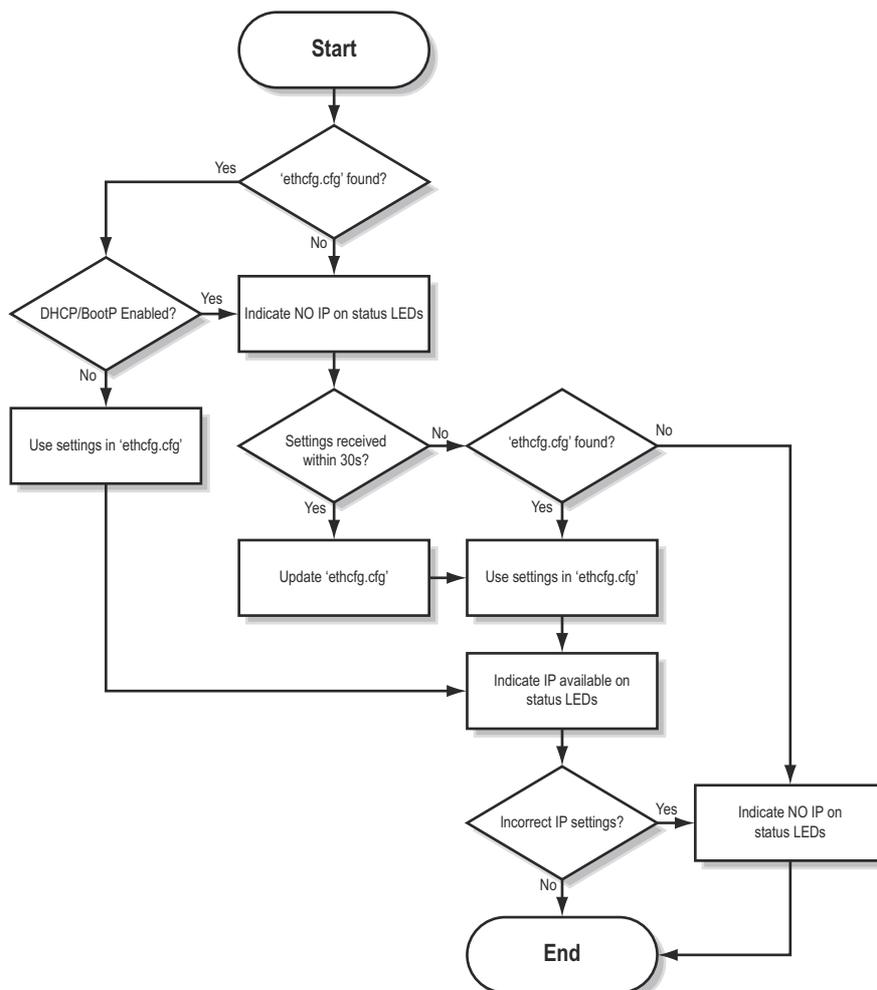
3. IO Data Exchange

3.1 PROFINET-IO

3.1.1 General

PROFINET is the open Industrial Ethernet standard for Automation from Profibus International. The PROFINET IO adapter provides PROFINET IO Soft Real-Time Communication.

As with most fieldbus systems, PROFINET makes a distinction between fast cyclical data, a.k.a. 'IO Data', and acyclical data, called 'Record Data'. PROFINET IO Data corresponds to what is called 'I/O Data' in the Anybus-X Generic Gateway User Manual.

- **PROFINET IO Data (I/O Data)**

  PROFINET IO Data is exchanged cyclically and is built up by I/O modules. In the case of the Anybus-X, the actual I/O module configuration is adopted from the I/O Controller/Supervisor, provided that their total size does not exceed the IO sizes specified in the Gateway Config Interface.

- **PROFINET Record Data (Parameter Data)**

  Record Data corresponds to what is called 'Parameter Data' in the Anybus-X Generic Gateway User Manual. Record Data is exchanged using acyclic Record Data Read/Write requests.

For information about how the IO- and Record Data relates to the Input- and Output Buffers, see 3-16 "Data Representation (IO Data & Record Data)".

3.1.2 GSDML-File

On PROFINET, all devices are associated with a GSDML-file. The GSDML-file is the equivalent of the Profibus GSD-file, and is based on the EXtensible Markup Language (XML).

This file holds information about the device (in this case the Anybus-X adapter), it's features, and possible I/O configurations. The latest version of the GSDML-file for the adapter can be downloaded from the HMS website, 'www.anybus.com'.

3.1.3 Data Representation (IO Data & Record Data)

As mentioned previously, the actual I/O configuration is determined by the IO Controller. The modules are mapped to the Input- and Output Buffers in the order of their slot number.

Example:

In this example, the I/O sizes in the Gateway Config Interface has been set to the following values:

Input I/O Data Size= 256 bytes
Output I/O Data Size= 400 bytes
Input Parameter Data Size= 256 bytes
Output Parameter Data Size= 112bytes

The following modules are specified in the IO Controller:

Slot	Module Size	Direction	Notes
0	0	-	(Device Access Point, DAP)
1	16 bytes	Input	-
2	16 bytes	Input/Output	-
3	128 bytes	Input	-
4	64 bytes	Input/Output	-
5	32 bytes	Input	-
6	64 bytes	Output	-
7	128 bytes	Output	-

Resulting memory layout:
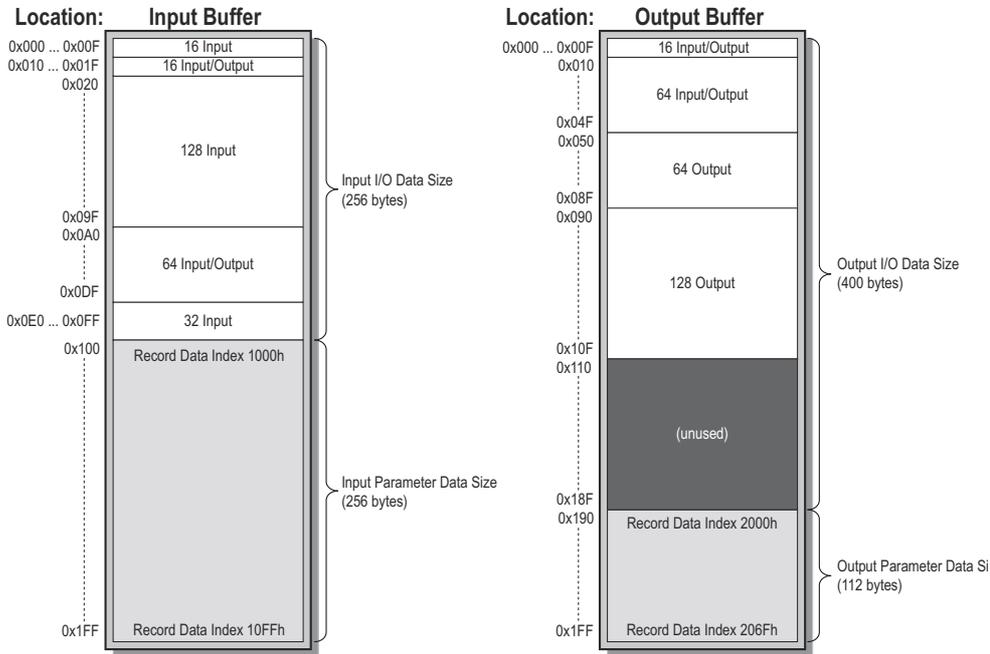
Note the 'unused'-part of the Output Buffer. The reason for this is that although IO Size Out is set to 400, only 272 bytes (128+64+64+16) are actually used in the I/O module configuration, hence it will not be exchanged on PROFINET.

Note: The Control/Status Words and the Live List is not considered in this example. For more information, consult the general Anybus X-gateway User Manual.

3.2 Modbus/TCP (Read-Only)

3.2.1 General

The Modbus/TCP protocol is an implementation of the standard Modbus protocol running on top of TCP/IP. The same function codes and addressing model are used. The built in Modbus/TCP server provides read-only access to the Input- and Output Buffers via a subset of the functions defined in the Modbus/TCP specification.

All Modbus/TCP messages are received/transmitted on TCP port no. 502. For detailed information regarding the Modbus/TCP protocol, consult the Open Modbus Specification.

3.2.2 Data Representation (Modbus/TCP Register Map)

The following function codes are implemented:

Modbus Function	Function Code	Associated with...
Read Input Registers	4	Input Buffer
Read Holding Registers	3	Output Buffer

The Input & Output Buffers are mapped to Modbus registers as follows:

Register Type	Register #	Associated with...	Location
Input Registers (3xxxx)	0x0000[a]	Input Buffer	0x000...0x001
	0x0001[b]		0x002...0x003
	0x0002[b]		0x004...0x005
	0x0003[b]		0x006...0x007
	0x0004[b]		0x008...0x009
	...		...
	0x00FF		0x1FE...0x1FF
Output Registers (4xxxx)	0x0000[c]	Output Buffer	0x000...0x001
	0x0001		0x002...0x003
	0x0002		0x004...0x005
	0x0003		0x006...0x007
	0x0004		0x008...0x009
	...		...
	0x00FF		0x1FE...0x1FF

a. Input Register 0x0000 may be occupied with the Status Word (if enabled). For more information, consult the Anybus-X Generic Gateway User Manual.
b. Input Registers 0x0001...0x0004 may be occupied with the Live List (if enabled). For more information, consult the general Anybus-X Generic Gateway User Manual.
c. Output Register 0x0000 may be occupied with the Control Word (if enabled). For more information, consult the Anybus-X Generic Gateway User Manual.

4. Web Interface

The web interface provides access to most common settings via any standard web browser. The pages differs slightly depending on network type (i.e. slave, master, fieldbus type etc), however some basic functions are essentially the same.

- **Device Diagnostics**

    These pages are unique to each network type, and is only available on network master interfaces.

    The diagnostic pages for each supported network are described separately later in this document (see C-44 "Master Specific Details").

- **General Status**

    This page provides an overview of the gateway initialisation parameters and general gateway diagnostics (these values corresponds to the values set using the Gateway Config Interface).

- **IP Config**

    This page holds the current network settings.

- **About**

    This page holds the software version numbers and serial numbers of the different components of the gateway. This page also holds the Ethernet MAC-ID of the adapter.
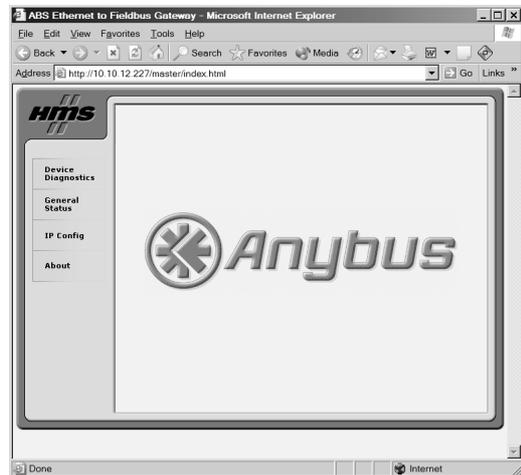
Start Page
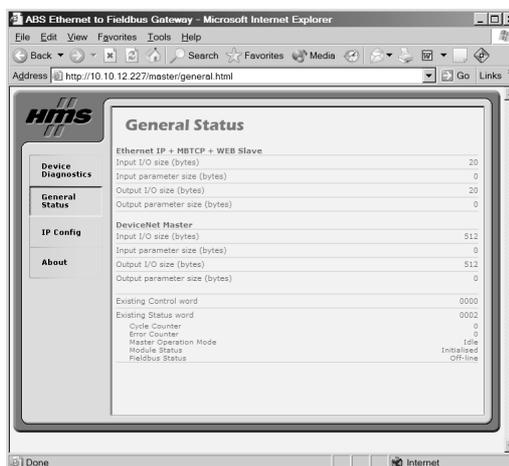
General Status Page
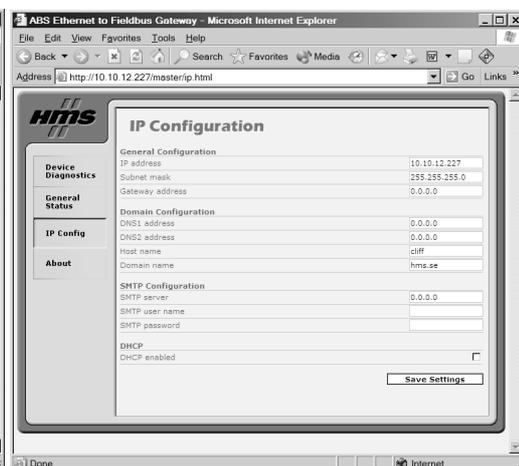
IP Configuration Page

A. Customization

A.1 Introduction

By default, the adapter is locked to the web interface described in 4-18 "Web Interface" and C-44 "Master Specific Details". To be able to customize the web interface and use advanced features such as the built in email client, the adapter must be unlocked. Contact HMS for further information.

Unlocking the adapter opens a wide range of possibilities:

- **Web Interface**

  It is possible to build customized web pages which display and control the factory process. Using SSI (Server Side Include), it is possible to monitor data in a user friendly manner, or to alter data based on user input.

  For more information, see A-20 "Webserver" and A-24 "Server Side Include (SSI)".

- **Email**

  The built in email client can send email messages based on events in the Input- and Output buffers.Using SSI (Server Side Include), it is possible to include data from the Input- and Output Buffers, allowing for example alarms containing diagnostic information to be sent via email.

  For more information, see A-20 "Webserver" and A-24 "Server Side Include (SSI)".

- **Multi-User Environments**

  The built in filesystem features a multi-user security framework, enabling different users to have access to different information.

  For more information, see B-36 "Filesystem".

- **FTP**

  The built in FTP server is used to transfer files to/from the filesystem, making it easy to develop and maintain a customized system.

  For more information, see A-35 "FTP Server".

A.2 Webserver

A.2.1 General

The built in webserver allows data and configuration settings to be accessed in a user friendly manner. A default set of web pages provides access to most common settings, however it possible to customize these pages to fit a particular application. The webserver supports SSI, which enables web pages to include dynamic content or affect the data in the Input- and Output Buffers (see A-24 "Server Side Include (SSI)").

The web pages are stored in the filesystem (see B-36 "Filesystem"), and new pages can easily be uploaded using FTP (see A-35 "FTP Server"). Certain directories within the filesystem, such as directories containing password files, are protected from web access. It is possible to protect additional directories by placing a file called 'web_accs.cfg' in the directory to protect. For more information, see B-40 "'web_accs.cfg'".

The following virtual files may optionally be used when building a web page for configuration of network parameters. These virtual files are:

```
\index.htm      - Points to the contents of config.htm
\config.htm     - Configuration frame page
\configform.htm- Configuration form page
\configform2.htm- Configuration form page
\store.htm      - Configuration store page
\logo.jpg       - HMS logo
\configuration.gif- Configuration picture
\boarder.bg.gif- picture
\boarder_m_bg.gif- picture
```

For more information about virtual files, see B-38 "Virtual Files".

A.2.2 Default Content Types

By default, the following content types are recognized by on their filename extension:

Content Type	File extension
text/html	*.htm; *.html; *.shtm
image/gif	*.gif
image/jpeg	*.jpeg; *.jpg; *.jpe
image/x-png	*.png
application/x-javascript	*.js
text/plain	*.bat; *.txt; *.c; *.h; *.cpp
application/x-zip-compressed	*.zip
application/octet-stream	*.exe; *.com
text/vnd.wap.wml	*.wml
application/vnd.wap.wmlc	*.wmlc
image/vnd.wap.wbmp	*.wbmp
text/vnd.wap.wmlscript	*.wmls
application/vnd.wap.wmlscript	*.wmlsc
text/xml	*.xml
application/pdf	*.pdf

If the extension of a file is not recognized the content type is set to binary data "/".

A.2.3 Configuration

The web server configuration is stored in the system file '\http.cfg'. In this file, it is possible to configure reported content types, and which files that shall be scanned for SSI.

Settings contained in '\http.cfg':

- **Content Types**

    There are a number of file types that by default will return predefined content types when requested through the webserver (see A-20 "Default Content Types"). When a file is requested through the webserver it will first search for the file types specified in this file. If it's not found in this file it will search for it in its predefined content types. This means that adding file type in this file will replace it's predefined type. File types shall be added under the heading [FileTypes], see A-21 "File Format". Up to 50 additional file types can be defined.

- **SSI File Types**

    By default, all files with the extensions '.htm', '.html', '.wml' and '.wmls' are scanned for SSI. It is possible to add additional file types to be scanned under the heading [SSIFileTypes], see A-21 "File Format". Up to 50 additional SSI file types can be defined.

File Format

```
[FileTypes]
FileType1:ContentType1
FileType2:ContentType2
...
FileTypeN:ContentTypeN

[SSIFileTypes]
FileType1
FileType2
...
FileTypeN
```

Example

```
[FileTypes]
tif:image/tiff
tiff:image/tiff
doc:application/msword
avi:video/x-msvideo

[SSIFileTypes]
htm
html
xml
```

A.3 Email Client

A.3.1 General

The adapter features a built in email client, capable of sending messages based on trigger-events in the Input- and Output Buffers. Messages may contain SSI functions, however due to obvious reasons certain functions (i.e. functions which requests input from the user) cannot be used.

A trigger-event must be present at least 0.5 seconds to ensure that it is detected by the adapter. Likewise, the trigger must be false for at least that time period in order for the adapter to recognize the next trigger event.

Trigger events are specified separately for each message in the email configuration files (see B-43 "Email Configuration Files (email_1.cfg, email_2.cfg ... email_10.cfg)").

The files have the following format:

```
[Register]
Buffer, Offset, Type

[Register match]
Match Value, Mask, Match operand

[To]
Recipient(s)

[From]
Sender

[Subject]
Subject line

[Headers]
Extra Headers

[Message]
Message body
```

Parameter	Description
Buffer	Source buffer. Possible values are 'IN' (Input Buffer) or 'OUT' (Output Buffer)
Offset	Source offset in the specified buffer, written in decimal or hex
Type	Source data type. Possible values are 'byte', 'word', and 'long'
Match Value	Value to compare with the source data. Shall be written in decimal or hexadecimal.
Mask	The Anybus performs a logical 'and' on the source data and this Mask before the value is compared with the Match Value. The value shall be written in decimal or hexadecimal.
Match Operand	Specifies how the data shall be compared with the Match Value. Possible values: '<', '=', '>'
Recipient(s)	Destination email addresses, semicolon separated
Sender	Sender email address
Subject line	Email subject (One line only)
Extra Headers	Optional. May be useful for advanced users when for example sending HTML emails etc.
Message Body	The actual email message.

Note 1: If the [Register] or [Register match] information is changed, a reset is required for changes to take effect. Other changes will have immediate effect.

Note 2: Hexadecimal values must be written in the format 0xN where 'N' is the hexadecimal value.

A.3.2 Operation

In operation, event-triggered email messages works as follows:

1. The source of the trigger event is read from the location specified by the Buffer, Offset and Type

2. A logical 'AND' is performed between the read trigger data and the Mask parameter.

3. The result is compared with the parameter Match Value according to the Match Operand.

4. If the result is true, the email message will be sent to the specified recipient(s).

Example:

```
[Register]
OUT, 0x0003, byte

[Register match]
0x20, 0x7F, >

[To]
support@hms-networks.com

[From]
Anybus@hms-networks.com

[Subject]
Status

[Message]
All data correct.
```

A byte is read from the Output Buffer at offset 0003h

A logical AND is performed between the data and 7Fh

If the result is larger than 20h, the email message is sent to support@hms-networks.com

A.4 Server Side Include (SSI)

A.4.1 General

Server Side Include (SSI) functionality enables dynamic content to be used on web pages and in email messages. SSI's are simply predefined commands embedded within the source document. The adapter interprets these commands prior to sending the document to the network.

Basic Syntax

```
<?--#exec cmd_argument='SSI COMMAND'-->
```

Example

When included in a .HTML document, the following line will be replaced by the Ethernet MacID of the adapter:

```
<?--#exec cmd_argument='DisplayMacId'-->
```

A.4.2 Data Consistency

The Input- and Output Buffers can be accessed via PROFINET IO, Modbus/TCP (read-only) and SSI. When multiple sources access the same memory locations, data consistency issues may arise. It is important to keep this in mind when designing the web interface.

IO Data

Cyclic PROFINET IO Data exchange has priority over SSI. This means that data written to the IO Data portion of the Output Buffer using SSI will be discarded.

Reading from the IO Data portion of the Input Buffer can be done freely, however.

Parameter Data

Parameter Data can be accessed freely, as long as the same memory location is not written to by more than one source. In such case, the data in that memory location is undefined.

Input/Output Data Buffers

Cyclic Data Exchange

IO Data

SSI Write

SSI Read

Record Data Read

SSI Write

Parameter Data

Record Data Write

SSI Read
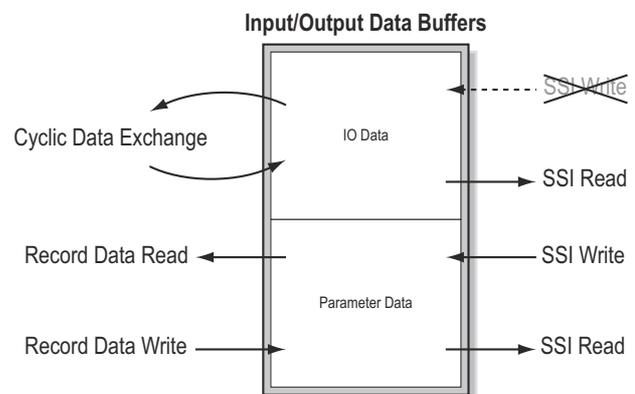
A.4.3 DisplayMacID

This function returns the MAC ID in format xx:xx:xx:xx:xx:xx.

Syntax:

```
<?--#exec cmd_argument='DisplayMacId'-->
```

A.4.4 DisplaySerial

This function returns the serial number of the adapter.

Syntax:

```
<?--#exec cmd_argument='DisplaySerial'-->
```

A.4.5 DisplayFWVersion

This function returns the main firmware revision of the adapter.

Syntax:

```
<?--#exec cmd_argument='DisplayFWVersion'-->
```

A.4.6 DisplayBLVersion

This function returns the bootloader firmware revision of the adapter.

Syntax:

```
<?--#exec cmd_argument='DisplayBLVersion'-->
```

A.4.7 DisplayIP

This function returns the currently used IP address.

Syntax:

```
<?--#exec cmd_argument='DisplayIP'-->
```

A.4.8 DisplaySubnet

This function returns the currently used Subnet mask.

Syntax:

```
<?--#exec cmd_argument='DisplaySubnet'-->
```

A.4.9 DisplayGateway

This function returns the currently used Gateway address.

Syntax:

```
<?--#exec cmd_argument='DisplayGateway'-->
```

A.4.10 DisplayDNS1

This function returns the address of the primary DNS server.

Syntax:

```
<?--#exec cmd_argument='DisplayDNS1'-->
```

A.4.11 DisplayDNS2

This function returns the address of the secondary DNS server.

Syntax:

```
<?--#exec cmd_argument='DisplayDNS2'-->
```

A.4.12 DisplayHostName

This function returns the hostname.

Syntax:

```
<?--#exec cmd_argument='DisplayHostName'-->
```

A.4.13 DisplayDomainName

This function returns the default domain name.

Syntax:

```
<?--#exec cmd_argument='DisplayDomainName'-->
```

A.4.14 DisplayDhcpState

This function returns whether DHCP/BootP is enabled or disabled.

Syntax:

```
<?--#exec cmd_argument='DisplayDhcpState( "Output when ON", "Output when OFF"
)'-->
```

A.4.15 DisplayEmailServer

This function returns the currently used SMTP server address.

Syntax:

```
<?--#exec cmd_argument='DisplayEmailServer'-->
```

A.4.16 DisplaySMTPUser

This function returns the username used for SMTP authentication.

Syntax:

```
<?--#exec cmd_argument='DisplaySMTPUser'-->
```

A.4.17 DisplaySMTPPswd

This function returns the password used for SMTP authentication.

Syntax:

```
<?--#exec cmd_argument='DisplaySMTPPswd'-->
```

A.4.18 DisplayStationName

This function returns the PROFINET Station Name.

Syntax:

```
<?--#exec cmd_argument='DisplayStationName'-->
```

A.4.19 DisplayStationType

This function returns the PROFINET Station Type.

Syntax:

```
<?--#exec cmd_argument='DisplayStationType'-->
```

A.4.20 DisplayVendorID

This function returns the PROFINET Vendor ID.

Syntax:

```
<?--#exec cmd_argument='DisplayVendorId'-->
```

A.4.21 DisplayDeviceID

This function returns the PROFINET DeviceID.

Syntax:

```
<?--#exec cmd_argument='DisplayDeviceId'-->
```

A.4.22 StoreEtnConfig[1]

This SSI function stores a passed IP configuration in the configuration file 'ethcfg.cfg'.

Syntax:

```
<?--#exec cmd_argument='StoreEtnConfig'-->
```

Include this line in a HTML page and pass a form with new IP settings to it.

Accepted fields in form:

```
SetIp
SetSubnet
SetGateway
SetEmailServer
SetDhcpState - value "on" or "off"
SetDNS1
SetDNS2
SetHostName
SetDomainName
SetSMTPUser
SetSMTPPswd
```

Default output:

```
Invalid IP address!
Invalid Subnet mask!
Invalid Gateway address!
Invalid IP address or Subnet mask!
Invalid Email Server IP address!
Invalid DHCP state!
Invalid DNS1!
Invalid DNS2!
Configuration stored correctly.
Failed to store configuration.
```

For information about how to change the SSI output, see A-33 "Changing the SSI Output Strings"

Note: The gateway must be restarted in order for the new settings to have effect. For more information, see A-34 "Restarting the Gateway".

---

1. This function cannot be used within email messages

A.4.23 GetText[1]

This SSI function gets the text from an object and stores it in the Output Buffer.

Syntax:

```
<?--#exec cmd_argument='GetText( "ObjName", OutWriteString ( offset ), n )'-->
```

ObjName   - Name of object.
offset   - Specifies the offset from the beginning of the Output Buffer.
n   - Specifies maximum number of characters to read (Optional)

Default output:

```
Success       - Write succeeded
Failure       - Write failed
```

For information about changing the default SSI output, see A-33 "Changing the SSI Output Strings".

A.4.24 printf

This SSI function outputs a formatted string, which may contain data from the Input/Output Buffers. The formatting of the string is equal to the standard C-function printf().

Syntax:

```
<?--#exec cmd_argument='printf("String to print", Arg1, Arg2, ..., ArgN)'-->
```

Like the standard C function printf() the "String to print" for this SSI function contains two types of objects: Ordinary characters, which are copied to the output stream, and conversion specifications, each of which causes conversion and printing of the next successive argument to printf. Each conversion specification begins with the character % and ends with a conversion character. Between the % and the conversion character there may be, in order:

- Flags (in any order), which modify the specification:

    -        which specifies left adjustment of the converted argument in its field.
    +   which specifies that the number will always be printed with a sign
    (space) if the first character is not a sign, a space will be prefixed.
    0   for numeric conversions, specifies padding to the field with leading zeroes.
    #   which specifies an alternate output form. For o, the first digit will be zero. For x or X, 0x or 0X will be prefixed to a non-zero result. For e, E,f, g and G, the output will always have a decimal point; for g and G, trailing zeros will not be removed.

- A number specifying a minimum field width. The converted argument will be printed in a field at least this wide, and wider if necessary. If the converted argument has fewer characters than the field width it will be padded on the left (or right, if left adjustment has been requested) to make up the field width. The padding character is normally space, but can be 0 if the zero padding flag is present.

- A period, which separates the field width from the precision.

- A number, the precision, that specifies the maximum number of characters to be printed from a string, or the number of digits to be printed after the decimal point for e, E, or F conversions, or the number of significant digits for g or G conversion, or the minimum number of digits to be printed for an integer (leading 0s will be added to make up the necessary width)

- A length modifier h, l (letter ell), or L. "h" Indicates that the corresponding argument is to be printed as a short or unsigned short; "l" indicates that the argument is along or unsigned long.

The conversion characters and their meanings are shown below. If the character after the % is not a conversion character, the behaviour is undefined.

Character	Argument type, Converted to
d, i	byte, short; decimal notation (For signed representation. Use signed argument)
o	byte, short; octal notation (without a leading zero).
x, X	byte, short; hexadecimal notation (without a leading 0x or 0X), using abcdef for 0x or ABC-DEF for 0X.
u	byte, short; decimal notation.
c	byte, short; single character, after conversion to unsigned char.
s	char*; characters from the string are printed until a "\0" is reached or until the number of characters indicated by the precision have been printed
f	float; decimal notation of the form [-]mmm.ddd, where the number of d's is specified by the precision. The default precision is 6; a precision of 0 suppresses the decimal point.
e, E	float; decimal notation of the form [-]m.dddddd e+-xx or[-]m.ddddddE+-xx, where the number of d's specified by the precision. The default precision is 6; a precision of 0 suppresses the decimal point.
g, G	float; %e or %E is used if the exponent is less than -4 or greater than or equal to the precision; otherwise %f is used. Trailing zeros and trailing decimal point are not printed.
%	no argument is converted; print a %

The arguments that can be passed to the SSI function *printf* are:

Argument	Description
InReadSByte(*offset*)	Reads a signed byte from position *offset* in the Input Buffer
InReadUByte(*offset*)	Reads an unsigned byte from position *offset* in the Input Buffer
InReadSWord(*offset*)	Reads a signed word (short) from position *offset* in the Input Buffer
InReadUWord(*offset*)	Reads an unsigned word (short) from position *offset* in the Input Buffer
InReadSLong(*offset*)	Reads a signed longword (long) from position *offset* in the Input Buffer
InReadULong(*offset*)	Reads an unsigned longword (long) from position *offset* in the Input Buffer
InReadString(*offset*)	Reads a string (char*) from position *offset* in the Input Buffer
InReadFloat(*offset*)	Reads a floating point (float) value from position *offset* in the Input Buffer
OutReadSByte(*offset*)	Reads a signed byte from position *offset* in the Output Buffer
OutReadUByte(*offset*)	Reads an unsigned byte from position *offset* in the Output Buffer
OutReadSWord(*offset*)	Reads a signed word (short) from position *offset* in the Output Buffer
OutReadUWord(*offset*)	Reads an unsigned word (short) from position *offset* in the Output Buffer
OutReadSLong(*offset*)	Reads a signed longword (long) from position *offset* in the Output Buffer
OutReadULong(*offset*)	Reads an unsigned longword (long) from position *offset* in the Output Buffer
OutReadString(*offset*)	Reads a NULL-terminated string (char*) from position *offset* in the Output Buffer
OutReadFloat(*offset*)	Reads a floating point (float) value from position *offset* in the Output Buffer

A.4.25 scanf[1]

This SSI function reads a string passed from an object in a HTML form, interprets the string according to the specification in format, and stores the result in the Output Buffer according to the passed arguments. The formatting of the string is equal to the standard C function call scanf()

Syntax:

```
<?--#exec cmd_argument='scanf( "ObjName", "format", Arg1, ..., ArgN), ErrVal1,
..., ErrvalN'-->
```

ObjName   - The name of the object with the passed data string
format   - Specifies how the passed string shall be formatted
Arg1 - ArgN   - Specifies where to write the data
ErrVal1 -ErrValN   - Optional; specifies the value/string to write in case of an error.

Character	Input, Argument Type
d	Decimal number; byte, short
i	Number, byte, short. The number may be in octal (leading 0(zero)) or hexadecimal (leading 0x or 0X)
o	Octal number (with or without leading zero); byte, short
u	Unsinged decimal number; unsigned byte, unsigned short
x	Hexadecimal number (with or without leading 0x or 0X); byte, short
c	Characters; char*. The next input characters (default 1) are placed at the indicated spot. The normal skip over white space is suppressed; to read the next non-white space character, use %1s.
s	Character string (not quoted); char*, pointing to an array of characters large enough for the string and a terminating "\0" that will be added.
e, f, g	Floating-point number with optional sign, optional decimal point and optional exponent; float*
%	Liteal %; no assignment is made.

The conversion characters d, i, o, u and x may be preceded by l (letter ell) to indicate that a pointer to 'long' appears in the argument list rather than a 'byte' or a 'short'

The arguments that can be passed to the SSI function scanf are:

Argument	Description
OutWriteByte(*offset*)	Writes a byte to position *offset* in the Output Buffer
OutWriteWord(*offset*)	Writes a word (short) to position *offset* in the Output Buffer
OutWriteLong(*offset*)	Writes a long to position *offset* in the Output Buffer
OutWriteString(*offset*)	Writes a string to position *offset* in the Output Buffer
OutWriteFloat(offset)	Writes a floating point (float) value to position *offset* in the Output Buffer

Default output:

```
Write succeeded
Write failed
```

For information about changing the default SSI output, see A-33 "Changing the SSI Output Strings".

---

1. This function cannot be used within email messages

A.4.26 IncludeFile

This SSI function includes the contents of a file on a web page.

Syntax:

```
<?--#exec cmd_argument='IncludeFile( "File name" )'-->
```

Default output:

```
Success        - <File content>
Failure        - Failed to open <filename>
```

For information about changing the default SSI output, see A-33 "Changing the SSI Output Strings".

A.4.27 SaveToFile[1]

This SSI function saves the contents of a passed form to a file. The passed name/value pair will be written to the file "File name" separated by the "Separator" string. The [Append|Overwrite] parameter determines if the specified file shall be overwritten, or if the data in the file shall be appended.

Syntax:

```
<?--#exec cmd_argument='SaveToFile( "File name", "Separator",[Append|Over-
write] )'-->
```

Default output:

```
Success        - Form saved to file
Failure        - Failed to save form
```

For information about changing the default SSI output, see A-33 "Changing the SSI Output Strings".

A.4.28 SaveDataToFile[1]

This SSI function saves the data of a passed form to a file. The "Object name" parameter is optional, if specified, only the data from that object will be stored. If not, the data from all objects in the form will be stored.

The [Append|Overwrite] parameter determines if the specified file shall be overwritten, or if the data in the file shall be appended.

Syntax:

```
<?--#exec cmd_argument='SaveDataToFile( "File name", "Object name",[Ap-
pend|Overwrite] )'-->
```

Default output:

```
Success        - Form saved to file
Failure        - Failed to save form
```

For information about changing the default SSI output, see A-33 "Changing the SSI Output Strings".

---

1. This function cannot be used within email messages

A.5 Changing the SSI Output Strings

There are two methods of changing the output strings from SSI functions:

1. Changing SSI output defaults by creating a file called "\ssi_str.cfg" containing the output strings for all SSI functions in the system

2. Temporary changing the SSI output by calling the SSI function "SsiOutput()".

A.5.1 SSI Output String File

If the file "\ssi_str.cfg" is found in the filesystem and the file is correctly according to the specification below, the SSI functions will use the output strings specified in this file instead of the default strings.

The files shall have the following format:

```
[StoreEtnConfig]
Success: "String to use on success"
Invalid IP: "String to use when the IP address is invalid"
Invalid Subnet: "String to use when the Subnet mask is invalid"
Invalid Gateway: "String to use when the Gateway address is invalid"
Invalid Email server: "String to use when the SMTP address is invalid"
Invalid IP or Subnet: "String to use when the IP address and Subnet mask does
not match"
Invalid DNS1: "String to use when the primary DNS cannot be found"
Invalid DNS2: "String to use when the secondary DNS cannot be found"
Save Error: "String to use when storage fails"
Invalid DHCP state: "String to use when the DHCP state is invalid"

[scanf]
Success: "String to use on success"
Failure: "String to use on failure"

[IncludeFile]
Failure: "String to use when failure"¹

[SaveToFile]
Success: "String to use on success"
Failure: "String to use on failure"¹

[SaveDataToFile]
Success: "String to use on success"
Failure: "String to use on failure"¹

[GetText]
Success: "String to use on success"
Failure: "String to use on failure"
```

The contents of this file can be redirected by placing the line '[File path]' on the first row, and a file path on the second.

Example:

```
[File path]
\user\ssi_strings.cfg
```

In this example, the settings described above will be loaded from the file 'user\ssi_strings.cfg'.

---

1. '%s' includes the filename in the string

A.5.2 Temporary SSI Output change

The SSI output for the next called SSI function can be changed with the SSI function "SsiOutput()" The next called SSI function will use the output according to this call. Thereafter the SSI functions will use the default outputs or the outputs defined in the file '\ssi_str.cfg'. The maximum size of a string is 128 bytes.

Syntax:

```
<?--#exec cmd_argument='SsiOutput( "Success string", "Failure string" )'-->
```

Example:

This example shows how to change the output strings for a scanf SSI call.

```
<?--#exec cmd_argument='SsiOutput ( "Parameter1 updated", "Error" )'-->
<?--#exec cmd_argument='scanf( "Parameter1", "%d", OutWriteByte(0) )'-->
```

A.6 Gateway Control

A.6.1 Refreshing Dynamic Gateway Status Information

The system files 'dynamic.txt' and 'master.txt' (master/scanner configurations only) holds dynamic status information from the gateway and the onboard network interfaces. In order to provide accurate information, these files must be refreshed prior to accessing their contents.

The following SSI command sequence will instruct the gateway to refresh it's status files:

Syntax:

```
<?--#exec cmd_argument='printf( "Data: %u", MbReadSWord( ID ) )'-->
```
(Substitute 'ID' with a value from the table below)

ID	Action
20	Refresh 'master.txt'
21	Refresh 'dynamic.txt'

A.6.2 Restarting the Gateway

The following command SSI restarts the gateway.

Syntax:

```
<?--#exec cmd_argument='printf( "Data: %u", MbReadSWord( 1 ) )'-->
```

A.7 FTP Server

A.7.1 General

Note: By default, the FTP server is protected from access via a password. To obtain this password, contact HMS.

The filesystem can be accessed using a common FTP client. Depending on security settings, different parts of the filesystem may be accessed by the user:

- **Normal users**

  The root directory will be '\user' unless the user has Admin access rights, see below.

- **Admin users**

  The user will have unrestricted access to the filesystem, i.e. the root directory will be '\'.

- **Global Admin Mode**

  Any username/password combination will be accepted. All users has unrestricted access to the filesystem, i.e. the root directory will be '\'.

For more information about the security framework in the device, see B-37 "Security Framework".

A.7.2 Connection Example (Windows Explorer)

The built in FTP client in Windows Explorer can easily be used to access the filesystem.

1. Open a Windows Explorer by right-clicking on the 'Start' button and selecting 'Explore'.
2. In the address field, type FTP://<user>:<password>@xxx.xxx.xxx.xxx
   - Substitute xxx.xxx.xxx.xxx with the IP address of the adapter
   - Substitute <user> with the username to use
   - Substitute <password> with the password to use
3. Press enter. The Explorer will now attempt to connect to the adapter with the specified settings.
4. If successful, the filesystem of the adapter shall now be displayed in the Explorer window.

B. Filesystem

B.1 General

The built in filesystem is a fixed-size storage area with a hierarchical directory structure. Files can be grouped in directories for increased readability.

- **Case Sensitivity**

  The filesystem is case sensitive. This means that the filename 'Anybus.txt' is not identical to the file 'Anybus.TXT'.

- **Filename / Pathname length**

  Filenames can be a maximum of 48 characters long. Pathnames can be 256 characters in total, filename included.

- **File size**

  File size is not restricted. However, the size cannot exceed the space available in the filesystem.

- **Free space**

  Approximately 2.4MB (1.4MB non-volatile & 1.0MB volatile).

Important Note

The non-volatile part of the filesystem (i.e. all directories except '\tmp\') is located in FLASH memory. Each FLASH segment can only be erased approximately 1000000 times due to the nature of this type of memory.

The following operations will erase one or more FLASH segments:

- Deleting, moving or renaming a file or directory
- Writing or appending data to an existing file
- Formatting the filesystem

Generally, it is recommended to put frequently altered data in the '\tmp\' directory to prevent data corruption due to FLASH wearout. Note that since this folder resides in volatile memory, any data stored herein will be erased in the event of a power cycle or reset.

B.2 Security Framework

B.2.1 General

The filesystem features two security levels; Admin and Normal. Security level is set at a per user basis.

- **Admin Mode**

  Admin users has full access to the filesystem via FTP. This enables the user to access areas of the filesystem, that is restricted or inaccessible in Normal mode.

  The Admin user accounts are defined in the file 'ad_pswd.cfg'.

- **Normal Mode**

  This mode is recommended for normal operation, so that web pages and other settings are protected from FTP access.

  The accounts for normal users are defined in the file 'sys_pswd.cfg'.

Files within the filesystem can be protected from web access through username/password authorization, see B-40 "Password Files" and B-40 "'web_accs.cfg'".

B.2.2 Normal Mode

In this mode, the FTP server is enabled only if there is a subdirectory called "\user". When a normal user connects via FTP, this directory will be their root directory. The user will not be able to access files outside this directory and it's subdirectories.

If user/password protection for FTP is required in normal mode, a file called "sys_pswd.cfg" must be placed in the directory "\user\pswd\". Files in this directory cannot be accessed from a web browser.

The adapter will operate in this mode if a valid password file (See A-6 "Password Files") is found.

B.2.3 Global Admin Mode

If no admin password file (See A-6 "Password Files") is found, the adapter will run in Global Admin Mode; i.e. all users will have Admin access rights (the FTP server accepts any username/password combination)

Note that this mode is primarily intended for product configuration and development.

B.3 Structure

B.3.1 Overview

The figure below illustrates the structure of the filesystem, where the system files are located, and which areas that can be accessed by Normal/Admin users.

Root directory for Admin users

Root directory for Normal users

| user |

| pswd | *(Files in this directory and its subdirectories are protected from web access)*

sys_pswd.cfg *(Passwords for Normal mode users)*

| email |

ssi_str.cfg *(SSI output strings)*   email_1.cfg
http.cfg *(Web server settings)*   *(User defined email files)*
ethcfg.cfg *(Network settings)*   email_10.cfg
pniocfg.cfg *(Profinet settings)*
static.txt *(General gateway information)*
(Virtual files, see below)

| email |

email_1.cfg
(Admin defined email files)
email_10.cfg

| pswd | *(Files in this directory and its subdirectories are protected from web access)*

ad_pswd.cfg *(Passwords for Admin mode users)*

| tmp | *(temporary files, volatile)*

dynamic.txt *(Gateway status information)*
master.txt *(Master status information. Master/Scanner configurations only)*

B.3.2 Virtual Files

The filesystem contains a set of virtual files used to build the default configuration webpage. These files can be overwritten or disabled, but not erased; A file with the same name and location temporarily replaces the corresponding virtual file until it has been removed again.

For more information about these files and their contents, see A-20 "Webserver".

B.4 System Files

The filesystem contains a set of ASCII-based configuration files, a.k.a System Files, used to configure various parts of the system. Depending on security settings, the files may be inaccessible for normal users. Generally, the gateway has to be restarted in order for any configuration changes to have effect.

Note: It is very important to follow the exact syntax specifications for each file, or the adapter may have trouble interpreting it, resulting in faulty or non-expected behaviour.

B.4.1 Configuration files

'ethcfg.cfg'

This file is read during startup and contains network related configuration settings. For more information about network configuration see A-1 "Network Configuration Basics".

The format of the file is the following:

```
[IP address]
10.10.12.212

[Subnet mask]
255.255.255.0

[Gateway address]
0.0.0.0

[DHCP/BOOTP]
OFF

[SMTP address]
0.0.0.0

[SMTP username]
username

[SMTP password]
password

[DNS1 address]
0.0.0.0

[DNS2 address]
0.0.0.0

[Domain name]
hms.se

[Host name]
Anybus

[HICP Password]
password
```

TCP/IP Settings

DHCP/BootP
ON - Enabled
OFF - Disabled

SMTP server/login settings
Username and Password is only necessary if required by the server.

Primary and Secondary DNS
Required in order to be able to resolve host names

Default Domain Name (Optional)

Host Name (Optional)

HICP Password (Optional)
If this entry is present, users connecting to the adapter using AnybusIPconfig must provide a valid password in order to be able to alter the TCP/IP settings.

Note: This file can be redirected. For more information, see B-44 "System File Redirection".

'http.cfg'

This file holds webserver configuration data. For more information, see A-21 "Configuration".

Note: This file can be redirected. For more information, see B-44 "System File Redirection".

B.4.2 Password Files

Note: Password files are case-sensitive, i.e. "cliff" does not equal "Cliff".

'ad_pswd.cfg' & 'sys_pswd.cfg'

User/password information for FTP is stored in the files 'sys_pswd.cfg' (Normal users) and 'ad_pswd.cfg' (Admin users). These files must be placed in '\user\pswd' and '\pswd\ respectively. These directories are protected from web browser access.

The file format is the following:

```
User1:password1
User2:password2
...
User3:password3
```

Example:
```
Cliff:KenSentMe
```

In this example, the username is 'Cliff', and the password is 'KenSentMe'.

If no ':' is present, the password will be equal to the username.

Example:
```
Flanders
```

In this example, both username and password will be 'Flanders'.

'web_accs.cfg'

To protect a directory from web access, a file called 'web_accs.cfg' must be placed in the directory to protect. This file shall contain a list of users that are allowed to browse the protected directory and its

subdirectories. Multiple of these password files may be present in the system, allowing different users to access different files and directories.

The file format is the same as for the 'ad_pswd.cfg' and 'sys_pswd.cfg' files, except that the optional parameter 'AuthName' can be added. The value of this parameter will be presented in the login window. If it is not given, the requested file/pathname will be presented instead.

File format:

```
User:Password

[AuthName]
(Message goes here)
```


The contents of this file can be redirected by placing the line '[File path]' on the first row, followed by a list of password files.

Example:

```
[File path]
\user\pswd\my_passwords\web_pswd.cfg
\user\pswd\my_passwords\more_pswd.cfg

[AuthName]
(Message goes here)
```

In this example, the accepted user/passwords will be loaded from the files '\user\pswd\my_passwords\web_pswd.cfg' and '\user\pswd\my_passwords\more_pswd.cfg'

B.4.3 Gateway Status Files

General

These files are made up of lists of keys and their values, in the following format:

Format:

```
<Key Name>=<Value>
```

- <Key Name> is a unique identifier.
- <Value> is a value associated with the <Key Name>. It can either be a single hexadecimal value, a list of hexadecimal values, or a binary list:

Type	Meaning
Hex8	Single hexadecimal value, 8 bit Example: 42
Hex16	Single hexadecimal value, 16 bit Example: 1234
Hex32	Single hexadecimal value, 32 bit Example: 12345678
Hex48	Single hexadecimal value, 48 bit Example: 123456789ABCDEF
Hex8 List	Multiple hexadecimal values, separated by dots (.) Example: 48.4F.4D.45.52.20.53.49.4D.53.4F.4E

Type	Meaning
Binary list	Multiple boolean values (no separation). Example: 110101101101100100011

'dynamic.txt'

This file contains the value of the Control- and Status words, and must be refreshed before use (See A-34 "Refreshing Dynamic Gateway Status Information").

Key Name	Meaning	Type
ModuleInfoAbs1.Controlword	Anybus-X PROFINET IO Adapter controlword value	Hex16
ModuleInfoAbs1.Statusword	Anybus-X PROFINET IO Adapter statusword value	Hex16
ModuleInfoAbs2.Controlword	Other network adapter controlword value	Hex16
ModuleInfoAbs2.Statusword	Other network adapter controlword value	Hex16

'master.txt'

This file is only present in master/scanner configuration and contains master/scanner related status information. This file must be refreshed before use in order to provide accurate information (see A-34 "Refreshing Dynamic Gateway Status Information").

The contents of this file depends on the type of gateway that is used; for more information, see C-44 "Master Specific Details".

'static.txt'

This file is updated during startup and contains the current I/O configuration and miscellaneous information about the onboard network adapters.

Key Name	Meaning	Type
ModuleInfoAbs1.InputIoSize	Anybus-X PROFINET IO Adapter Input I/O size	Hex16
ModuleInfoAbs1.InputParSize	Anybus-X PROFINET IO Adapter Input Parameter Data size	-
ModuleInfoAbs1.OutputIoSize	Anybus-X PROFINET IO Adapter Output I/O size	Hex16
ModuleInfoAbs1.OutputParSize	Anybus-X PROFINET IO Adapter Output Parameter Data size	-
ModuleInfoAbs1.FieldbusType	Fixed value of 0084 (PROFINET IO)	Hex16
ModuleInfoAbs1.ModuleType	Fixed value of 0101 (Slave)	Hex16
ModuleInfoAbs1.SerialNumber	Anybus-X PROFINET IO Adapter serial number.	Hex32
ModuleInfoAbs1.BootloaderVersion	Anybus-X PROFINET IO Adapter bootloader revision.	Hex16
ModuleInfoAbs1.SoftwareVersion	Anybus-X PROFINET IO Adapter software revision.	Hex16
ModuleInfoAbs1.EthernetMacId	Anybus-X PROFINET IO Adapter MacID.	Hex48
ModuleInfoAbs2.InputIoSize	Other network adapter Input I/O size.	Hex16
ModuleInfoAbs2.InputParSize	Other network adapter Input Parameter Data size.	Hex16
ModuleInfoAbs2.OutputIoSize	Other network adapter Output I/O size.	Hex16
ModuleInfoAbs2.OutputParSize	Other network adapter Output Parameter Data size.	Hex16
ModuleInfoAbs2.FieldbusType	Other network adapter type: 0001 = Profibus DP 0040 = Modbus Plus 0005 = Profibus DPV1 0045 = Modbus RTU 0011 = Interbus 0065 = ControlNet 0015 = LonWorks 0083 = Ethernet 0020 = CANOpen 0084 = PROFINET IO 0025 = DeviceNet 0090 = CC-Link 0035 = FIP IO 0091 = AS-Interface	Hex16
ModuleInfoAbs2.ModuleType	Other network adapter class: 0101 = Slave 0201 = Master	Hex16

Key Name	Meaning	Type
ModuleInfoAbs2.SerialNumber	Other network adapter serial number.	Hex32
ModuleInfoAbs2.BootloaderVersion	Other network adapter bootloader revision.	Hex16
ModuleInfoAbs2.SoftwareVersion	Other network adapter software revision.	Hex16
ModuleInfoAbs2.EthernetMacId	Other network adapter Ethernet MacID (if applicable)	Hex48
Gateway.SerialNumber	Generic gateway serial number	Hex32
Gateway.BootloaderVersion	Generic gateway bootloader revision	Hex16
Gateway.LibraryVersion	Generic gateway library revision.	Hex16
Gateway.ApplicationVersion	Generic gateway application revision.	Hex16
Gateway.ProductVersion	Generic gateway product revision.	Hex16

B.4.4 Other

'pniocfg.cfg'

This file holds basic PROFINET configuration settings. The file is read once during startup, i.e. the gateway must be restarted on order for any changes to have effect (Unless it's contents has been changed by an IO Controller/Supervisor via the DCP protocol. In such case, the settings will have effect immediately).

Note: Do not alter the Station Type, Vendor ID or Device ID.

Example:

```
[Station Name]
Nice Device

[Station Type]
ABS-PRT

[Vendor ID]
0x010C

[Device ID]
0x0001
```

Station Name

Station name as ASCII string, maximum 64 characters.

Station Type

Important: Do not alter this setting.

Vendor ID

Important: Do not alter this setting.

Device ID

Important: Do not alter this setting.

Email Configuration Files (email_1.cfg, email_2.cfg ... email_10.cfg)

These files contain predefined email messages and information on how and when to send them. It is possible to have a maximum of 10 admin defined email files and 10 user defined email files. The files must be named 'email_1.cfg'... 'email_10.cfg', and placed in the folders '\email' and '\user\email' re

For a specification of the file format, see 6-1 "Email Client".

B.5 System File Redirection

The contents of certain system files can be redirected by placing the line '[File path]' on the first row, and a file path on the second. The files which supports redirection are marked with a note indicating this.

Example:

```
[File path]
\user\eth_settings.cfg
```

In this example, the settings described above will be loaded from the file 'user\eth_settings.cfg'. This permits normal users to access the network configuration settings.

C. Master Specific Details

C.1 Profibus Master Gateway

The information in this section only applies to Anybus-X PROFINET IO <> Profibus Master gateways.

C.1.1 Device Diagnostics Webpage

The diagnostic page for this device holds the current status of each slave on the Profibus network.

- **'Slave'**

  (Slave number).

- **'Slave Configured'**

  1: Configured
  0: Not Configured

- **'Slave in Data Transfer'**

  1: Slave in Data Transfer
  0: Slave not in Data Transfer

- **'Slave with Diagnostics'**

  1: Diagnostics available
  0: Diagnostics not available

Profibus Master Diagnostic Page

C.1.2 Contents of 'master.txt'

Note: The following information is only relevant when creating a customized web interface.

Key Name	Meaning	Type
ModuleInfoAbs2.Configured	List of configured slaves (128 entries) 1: Configured, 0: Not Configured	Binary List
ModuleInfoAbs2.DataTransfer	List of slaves in data transfer (128 entries) 1: Slave in Data Transfer, 0: Slave not in Data Transfer	Binary List
ModuleInfoAbs2.Diagnostics	List of slaves with diagnostics (128 entries) 1: Slave Diagnostics available, 0: Slave Diagnostics not available	Binary List

Note: In order to provide accurate information, this file must be refreshed prior to accessing it. For more information, see A-34 "Refreshing Dynamic Gateway Status Information".

C.2 DeviceNet Scanner Gateway

The information in this section only applies to Anybus-X PROFINET IO <> DeviceNet Scanner gateways.

C.2.1 Device Diagnostics Webpage

General

The diagnostic webpage for this device holds the current status of each node on the DeviceNet network.

- **'Slave'**

  (Node number).

- **'Node Configured'**

  1: Node Active
  0: Node not active

- **'Node Idle'**

  1: Node Idle
  0: Node not Idle

- **'Node Faulted'**

  1: Node Faulted
  0: Node not Faulted

- **'Node Status'**

  Diagnostic information associated with the node.

DeviceNet Scanner Diagnostic Page

Advanced

Additional diagnostics can be viewed by clicking 'Advanced' link in the upper right corner.

- **'Connections'**

  Number of active connections.

- **'Packetrate'**

  Expected packet rate.

- **'Baudrate'**

  Currently used operating baudrate on the DeviceNet network.

- **'MacID'**

  Currently used MacID of the DeviceNet Scanner.

C.2.2 Contents of 'master.txt'

Note: The following information is only relevant when creating a customized web interface.

Key Name	Meaning	Type
ModuleInfoAbs2.NodeActive	List of active nodes (64 entries). 1: Node Active, 0: Node not Active	Binary List
ModuleInfoAbs2.NodeIdle	List of idle nodes (64 entries). 1: Node Idle, 0: Node not Idle	Binary List
ModuleInfoAbs2.NodeFaulted	List of faulted nodes (64 entries). 1: Faulted, 0: Not faulted	Binary List
ModuleInfoAbs2.NodeStatus	List containing the status of each node 00h: No error, Node is not in the scanlist 46h: Duplicate MACID failure 47h: Scanner configuration error 48h: Device communication error 49h: Incorrect device identity 4Ah: Data overrun error 4Bh: No network traffic detected 4Ch: No network traffic to the scanner detected 4Dh: Incorrect I/O data size 4Eh: Device does not exist 4Fh: Transmit failure 50h: Device is in 'Idle' mode 51h: Device is in 'Fault' mode 52h: Fragmentation error 53h: Unable to initialise device 54h: Node not yet initialised 55h: Receive buffer overflow 56h: Node changed to 'Idle' mode 57h: Shared master error 58h: Shared choice error 59h: ADR failed 5Ah: CAN port disabled by application 5Bh: Bus-off condition detected 5Ch: No bus power detected 5Fh: Updating flash 60h: In test mode 61h: Scanner halted by application 62h: Unrecoverable firmware failure 63h: Unrecoverable hardware failure	Hex8 List
ModuleInfoAbs2.Connections	Current number of established connections towards other nodes.	Hex8
ModuleInfoAbs2.PacketRate	Current expected packet rate.	Hex16
ModuleInfoAbs2.Baudrate	Currently used baudrate. 00: 125kbps, 01: 250kbps, 02: 500kbps.	Hex8
ModuleInfoAbs2.MacId	DeviceNet MacID	Hex8

Note: In order to provide accurate information, this file must be refreshed prior to accessing it. For more information, see A-34 "Refreshing Dynamic Gateway Status Information".

C.3 AS-Interface Master Gateway

The information in this section only applies to Anybus-X PROFINET IO <> AS-Interface Master gateways.

C.3.1 Device Diagnostics Webpage

General

The diagnostic page for this device holds the current status of each node on the AS-Interface network.

- **'Slave'**

    AS-Interface slave number.

- **'Configured slaves'**

    1: Configured
    0: Not configured

- **'Activated slaves'**

    1: Activated
    0: Not activated

- **'Detected slaves'**

    1: Detected
    0: Not detected

- **'Peripheral fault'**

    1: Peripheral fault
    0: No peripheral fault

- **'IO Configurations'**

    This column holds the IO Configuration of each AS-Interface slave.

- **'ID Code'**

    This column holds the ID Code of each AS-Interface slave.

AS-Interface Diagnostic (Standard)

Advanced

Additional diagnostics can be viewed by clicking 'Advanced' link in the upper right corner.

- **'Status'**

    This entry shows the state of several AS-Interface specific flags. For more information regarding these flags, consult the Anybus-X AS-Interface Network Interface Addendum.

- **Configmode**

    This entry reflects the current mode of operation. For more information, consult the Anybus-X AS-Interface Network Interface Addendum.

- **Byte / Nibble Mode**

    This entry reflects how data is represented in the Input- and Output Buffers. For more information, consult the Anybus-X AS-Interface Network Interface Addendum.

C.3.2 Contents of 'master.txt'

Note: The following information is only relevant when creating a customized web interface.

Key Name	Meaning	Type
ModuleInfoAbs2.Configured	List of configured slaves (64 entries) 1: Slave Configured, 0: Slave not Configured	Binary List
ModuleInfoAbs2.Active	List of Active Slaves (64 entries) 1: Slave Active, 0: Slave not Active	Binary List
ModuleInfoAbs2.Detected	List of Detected Slaves (64 entries) 1: Slave Detected, 0: Slave not Detected	Binary List
ModuleInfoAbs2.Fault	List of Peripheral Fault (64 entries) 1: Peripheral Fault, 0: No fault	Binary List
ModuleInfoAbs2.IOConfig	List w. I/O Configuration of each slave (64 entries).	Hex8 List
ModuleInfoAbs2.IDCode	List w. ID Codes of each slave (64 entries).	Hex8 List
ModuleInfoAbs2.Status	Bit field containing misc. status information: Bit 0: Offline mode Bit 1: (reserved) Bit 2: EEPROM OK Bit 3 Automatic addressing enabled (set by the user) Bit 4: Periphery fault Bit 5: (reserved) Bit 6: (reserved) Bit 7: (reserved) Bit 8: Offline phase active Bit 9: Voltage on AS-Interface too low Bit 10: Normal operation Bit 11: 1: Configuration Mode Bit 12: Automatic programming possible Bit 13: (reserved) Bit 14: Slave with address 0 exists Bit 15: Actual configuration matches configured configuration	Hex16
ModuleInfoAbs2.ConfigMode	Current operation mode 1: Configuration Mode, 0. Protected Mode	Hex8
ModuleInfoAbs2.NibbleMode	Data Format 1: Nibble Mode, 0: Byte Mode	Hex8

Note: In order to provide accurate information, this file must be refreshed prior to accessing it. For more information, see A-34 "Refreshing Dynamic Gateway Status Information".

D. Ethernet Transport Provider

D.1 General Information

The Ethernet interface supports the Transport Provider protocol, which allows a host to control the network interface on the other side of the gateway using the standardized Anybus-S API.

This includes...

- Anybus OPC Server
- Anybus NetTool for DeviceNet
- Anybus NetTool for PROFIBUS
- Custom applications based on the Anybus-S API
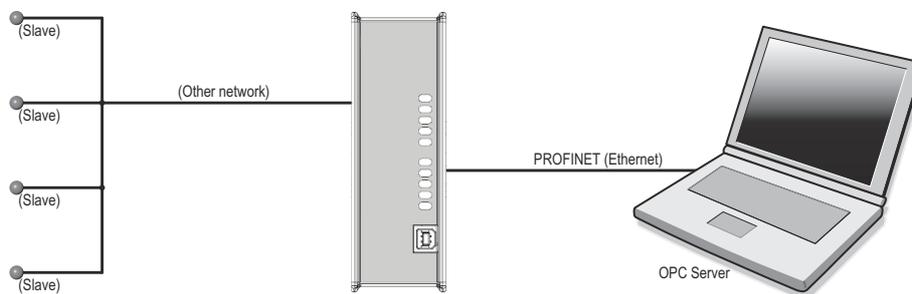

D.2 Allocation of I/O Data

The Transport Provider uses parts of the Input- and Output Buffers to transfer I/O data. The amount of data allocated for the Transport Provider is defined as the difference in I/O sizes between the two network interfaces.

Example:

Transport Provider I/O Size =(I/O Size, Interface A) - (I/O Size, Interface B)
I/O Size, Interface A =30 bytes

I/O Size, Interface B =20 bytes
Transport Provider I/O Size =10 bytes

Interface A
I/O Size = 30 bytes

Interface B
I/O Size = 20 bytes

20 bytes

20 bytes

This data is available as I/O Data
on the PROFINET (ethernet) side

10 bytes

10 bytes

This data is transported as I/O Data
through the Ethernet Transport Provider
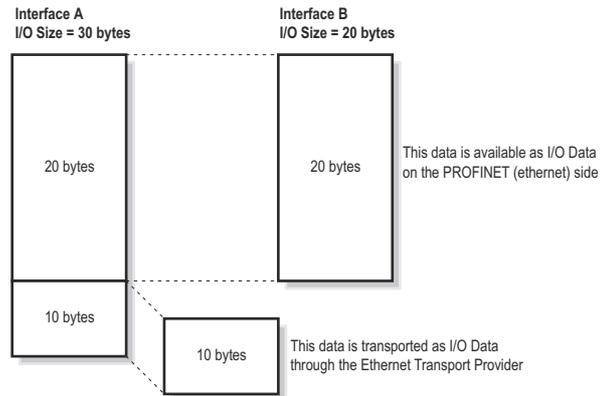
Note: In case the Transport Provider shall handle the complete I/O image towards the other network, set the I/O size to 0 (zero) on the Ethernet interface.

E. Technical Specification

E.1 Network Interface Details

General

- 100mbit operation, full duplex
- Twisted-pair cables
- Flexible filesystem providing both volatile and non-volatile storage areas
- Security framework
- Server Side Include (SSI) capability
- FTP server
- Webserver
- Email client (Messages triggered by I/O data events)
- DHCP/DCP/HICP support
- DNS support

Industrial Protocols

- **Modbus/TCP**

  Read-only access, using a subset of the Modbus/TCP specification.

- **PROFINET IO**
  - Soft Real-Time (RT) communication
  - Cyclic data exchange (10ms cycle time)
  - Acyclic Data exchange (Record Data Requests)
  - Up to 64 slots / 1 sub-slot
  - Up to 512 bytes of I/O in each direction
  - DCP support (Discovery and Configuration Protocol)

E.2 Ethernet Connector Pinout (RJ45)

Pin	Signal
1	TD+
2	TD-
3	RD+
4	Termination
5	Termination
6	RD-
7	Termination
8	Termination