Fieldbus Appendix

Anybus-M EtherNet/IP

ABM-EIP - EtherNet/IP Scanner & IT Functionality

Doc.Id. HMSI-168-21
Rev. 1.2

HMS Industrial Networks
Mailing address: Box 4126, 300 04 Halmstad, Sweden
Visiting address: Stationsgatan 37, Halmstad, Sweden

E-mail: info@hms-networks.com
www.anybus.com

Important User Information

This document is intended to provide a good understanding of the functionality offered by Anybus-M EtherNet/IP. The document only describes the features that are specific to the Anybus-M EtherNet/IP. For general information regarding the Anybus-M, consult the Anybus-S/M Parallel Design Guide.

Please consult the general Anybus-S Parallel Design Guide for further information about the Anybus-S platform.

The reader of this document is expected to be familiar with high level software design, and communication systems in general. The use of advanced EtherNet/IP-specific functionality may require in-depth knowledge in EtherNet/IP networking internals and/or information from the official EtherNet/IP specifications. In such cases, the people responsible for the implementation of this product should either obtain the EtherNet/IP specification to gain sufficient knowledge or limit their 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.

Jul 2012 Doc Id HMSI-168-21

Table of Contents

Chapter 6 SMTP Client

Chapter 7 Server Side Include (SSI)

Chapter 8 Ethernet/IP

Chapter 9 Fieldbus Specific Mailbox Commands

P. About This Document

For more information, documentation etc., please visit the HMS website, 'www.anybus.com'.

P.1 Related Documents

Document	Author
RFC 821 (Network Working Group)	www.ietf.org
RFC 896 (Network Working Group)	www.ietf.org
RFC 1918 (Network Working Group)	www.ietf.org
EIP Specifications (ControlNet International and ODVA)	www.odva.org
Anybus-S/M Parallel Design Guide	www.anybus.com

P.2 Document History

Summary of Recent Changes (1.10 ... 1.2)

Change	Page(s)
Corrected description of how to format the file system	145

Revision List

Revision	Date	Author(s)	Chapter(s)	Description
1.00	2005-11-29	PeP	All	1st release
1.01	2005-12-11	PeP	1, 9, A, C	Minor update
1.02	2006-10-19	PeP	B, C	Misc. minor updates
1.03	2008-10-29	HeS	8, 9	Misc. minor updates
1.04	2010-06-21	KeL	2, 4, 7, 8, 9	Misc. minor updates
1.05	2010-12-06	KeL	P, B, E	Misc. minor updates
1.06	2011-08-19	KeL	P, 1,7,8,9	Misc. updates
1.10	2012-07-11	KeL	P,1,2,7,8,9	Misc. updates and additions
1.2	2016-10-07	KeL	A	Correction

P.3 Conventions & Terminology

The following conventions are used throughout this manual:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The terms 'Anybus' or 'module' refers to the Anybus-CompactCom module.
- The terms 'host' or 'host application' refers to the device that hosts the Anybus module.
- Hexadecimal values are written in the format NNNNh, where NNNN is the hexadecimal value.

P.4 Sales and Support

Please contact the sales and support pages at www.anybus.com.

1. About the Anybus-M EtherNet/IP

The Anybus-M EtherNet/IP provides full EtherNet/IP scanner functionality via the patented Anybus-S application interface. Any device that supports this standard can take advantage of the features offered by the module, providing seamless network integration regardless of network type.

This product conforms to all aspects of the application interface defined in the Anybus-S Parallel Design Guide, making it fully interchangeable with any other device following that specification.

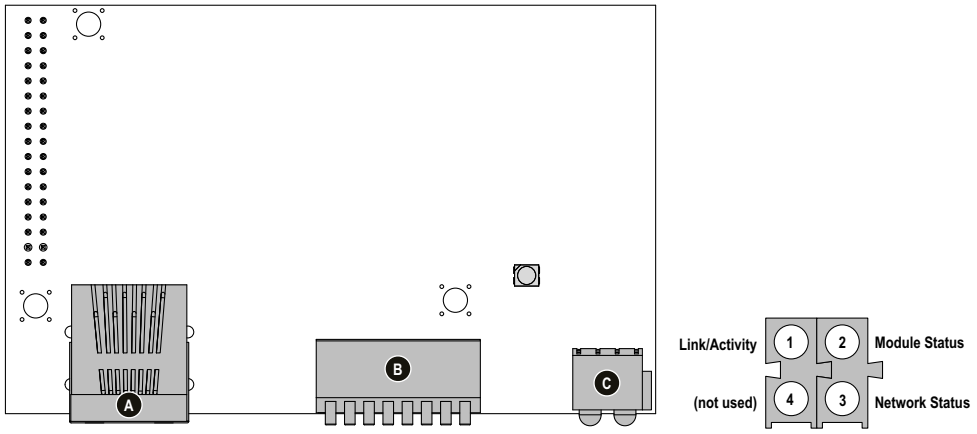
1.1 Features

- Supports shielded (FTP) and unshielded (UTP) cables
- EtherNet/IP scanner
- Exchanges data with up to 64 slaves
- Identity customization
- Built in file system with volatile and non-volatile storage locations
- FTP server
- Web server
- SMTP client
- Server Side Include (SSI) capability

1.2 Fieldbus Conformance Notes

- This product is pre-certified for network compliance. While this is done to ensure that the final product *can* be certified, it does not necessarily mean that the final product will not require recertification. Contact HMS for further information.
- Each device on EtherNet/IP is associated with an Electronic Data Sheet (a.k.a .EDS-file), which holds a description of the device and its functions. HMS supplies a generic .EDS-file which can serve as a basis for new implementations; note however that this file must be altered to match the end product (i.e. vendor and identity settings etc.).
- In order to pass conformance tests, the module should not be initiated to an IO data size larger than 500 bytes in each direction.
- In order to pass conformance tests, the mailbox EXACT_IO_MATCH must be sent during initialization, see "Enable Exact IO Match (EXACT_IO_MATCH)" on page 128.

1.3 Overview



#	Description	
A	Ethernet Connector	For more information, see “Connectors” on page 148
B	Switches	IP configuration switches, see “TCP/IP Settings” on page 10
C	Status Indicators	These LEDs indicate run time status and errors to the user, see below.

Status Indicators

#	Indication	State	Description
1	Link/Activity	Green	Link established
		Green, flashing	Activity; receiving/transmitting data
		Off	No link or power off
2	Module Status	Green	Device operational - Module is operating correctly in Run-state
		Green, flashing	Standby - Module has not been configured - Scanner in Idle-state
		Red	Major fault - Major unrecoverable fault
		Red, flashing	Minor fault - Minor recoverable fault (originated on timeout) - An originated connection could not be opened
		Alternating Red/Green	Self test - Module is performing power up test procedures
		Off	No power
3	Network Status	Green	Connected - The module has at least one established EtherNet/IP connection (target or originated)
		Green, flashing	No connections - There are no EtherNet/IP connections established to the module (Class 1 or Class 3, target or originated)
		Red	Duplicate IP - Configured IP address already in use
		Red, flashing	Connection timeout - One or several EtherNet/IP target connections have timed out - The module can only leave this state if all timed-out target connections are re-established, or if the module is reset.
		Off	No power or no IP address
4	-	-	-

2. Basic Operation

2.1 Data Exchange

Parameter Data Areas (Explicit Data)

By default, the Parameter Data portions of the Anybus Input- and Output areas cannot be accessed via EtherNet/IP. Using the mailbox commands PARAMETER_IN_MAP and PARAMETER_OUT_MAP, it is however possible to map blocks of Parameter Data to the Parameter Data Input- and Output Mapping Objects.

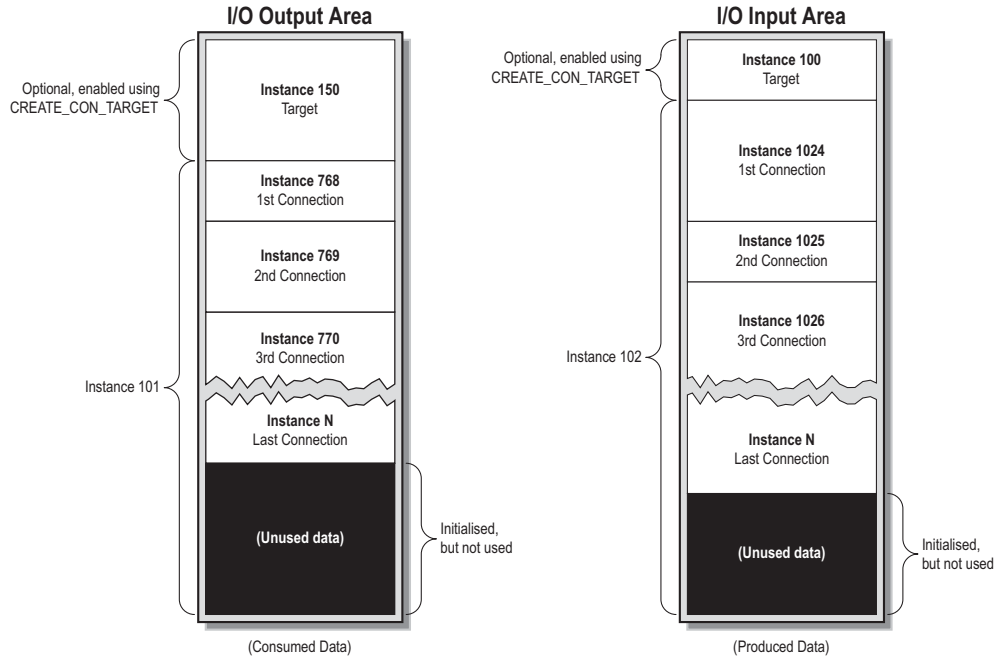
See also...

- "Parameter Data Input Mapping (PARAMETER_IN_MAP)" on page 114
- "Parameter Data Out Area Mapping (PARAMETER_OUT_MAP)" on page 116
- "Parameter Data Input Mapping Object, Class B0h" on page 44
- "Parameter Data Output Mapping Object, Class B1h" on page 45

I/O Data Areas

The Anybus-M EtherNet/IP can exchange I/O data with up to 64 slaves. Additionally, it is possible to allocate a block of I/O data for the scanner itself by issuing the mailbox command CREATE_CON_TARGET, or by setting up a target connection in the scanner configuration, see "Connection Configuration Object, Class F3h" on page 46.

Each connection is represented as two instances in the Assembly Object, which in turn are mapped to the I/O Data portion of the Anybus Input- and Output areas as depicted in the example below. The connections are mapped in the order of their instance number.



See also...

- "Create Connection Target Area (CREATE_CON_TARGET)" on page 132
- "Assembly Object, Class 04h" on page 38
- "Connection Configuration Object, Class F3h" on page 46

2.2 File System

General

The module features a built in file system, which is used to store information such as web files, network communication settings, email messages etc.

The filesystem can be accessed using FTP, HTTP, and by the application via the mailbox interface.

Storage Areas

The filesystem consists of the different storage locations, a.k.a.'Discs':

- **Disc 0 (1856 kByte, Non Volatile)**

  This section is intended for static files such as web files etc. Once the product is released, this section can be locked so that it cannot be altered by the end user.

- **Disc 1 (192 kByte, Non Volatile)**

  This section is intended for configuration files etc. which can be altered by the end user.

- **Disc 2 (1024 kByte, Volatile)**

  This section is intended for temporary storage. Any data placed here will be lost in case of power loss or reset.

Conventions

- '\' (backslash) is used as a path separator
- A 'path' originates from the system root and as such must begin with a '\'
- A 'path' must not end with a '\'
- Names may contain spaces (' ') but must not begin or end with one.
- Names must not contain one of the following characters: '\ / : * ? " < > |'
- Names cannot be longer than 48 characters (plus null termination)
- A path cannot be longer than 256 characters (filename included)
- The maximum number of simultaneously open files is 40
- The maximum number of simultaneously open directories is 40

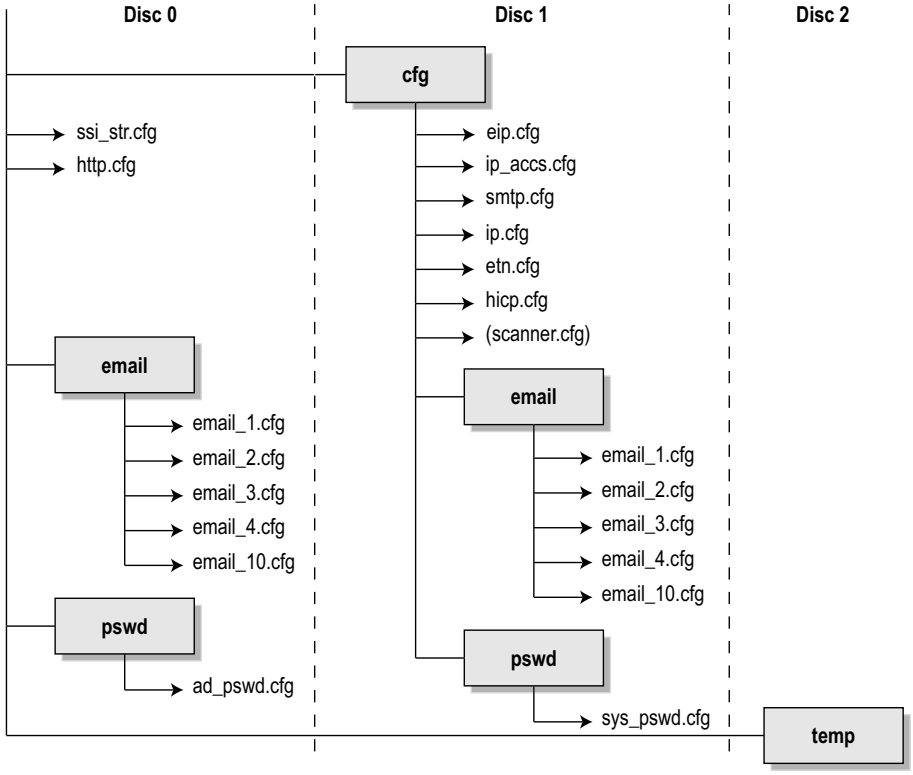
Important Note:

The non-volatile storage is located in FLASH memory. Each FLASH segment can only be erased approximately 100000 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
- Saving scanner configuration

2.2.1 Filesystem Overview



2.2.2 System Files

The filesystem contains a set of files used for system configuration. These files, known as "system files", are regular ASCII files which can be altered using a standard text editor (such as the Notepad in Microsoft Windows™). Note that some of these files may also be altered by the module itself, e.g. when using SSI (see "Server Side Include (SSI)" on page 20).

The format of the system files are based on the concept of 'keys', where each 'key' can be assigned a value, see example below.

Example:

```
[Key1]
value of key1

[Key2]
value of key2
```

Some configuration files can be redirected by placing a file path destination in the file. These are ssi_str.cfg, http.cfg, ip_accs.dfg, smtp.cfg, ip.cfg, etn.cfg, eip.cfg and hicp.cfg.

Example:

```
[file path]
\dest\for\content.cfg
```

The exact format of each system file is described in detail later in this document.

Important Note: Contrary to what is state above, the file '\cfg\scanner.cfg' holds the scanner configuration in binary format. This file is created automatically by the module and must not be altered manually.

2.3 Network Reset Handling

The Identity object provides a reset service. By default, incoming reset requests are handled as follows:

- **Reset Type 0: 'Power Cycling Reset'**
  Module is restarted.
- **Reset Type 1: 'Out of Box Reset'**
  The contents of '\cfg\' is erased, and the module is restarted.

Optionally, the module can forward reset requests to the application. To accomplish this, use the following settings in ANYBUS_INIT:

- Set the RDR-bit (bit 4) in the 'Operation Mode'-word
- Set the RST-bit (bit 3) in the 'Event Notification Config.'-word

This will cause the module to generate an Event Notification (which depending on the implementation may or may not trigger an interrupt) each time a network-reset is received. The application can then retrieve the reset-type using GET_ID_RESET_PARAM and perform the desired actions (e.g. erase the '\cfg\' directory and internal configuration settings + reset itself and the Anybus module).

See also...

- "Identity Object, Class 01h" on page 37
- "Enable Exact IO Match (EXACT_IO_MATCH)" on page 128

2.4 DNS Functionality

General

DNS is a service which translates hostnames into IP addresses. To achieve this, the Anybus module asks a DNS server for the IP address of a specified host. If the hostname isn't recognized by the server, it will forward the request to another DNS server until the corresponding address is found.

The Anybus will use this functionality each time a hostname is used instead of an IP address.

Host / Domain Names Conventions

Prior to issuing a DNS request, the Anybus module will process the specified hostname as follows:

- If the hostname does not contain a dot, the Anybus will append the default domain.
- If the hostname ends with a dot, that dot will automatically be removed.
- If the hostname contains a dot but isn't ending with one, the Anybus will issue a DNS request. If this request fails, the Anybus will append the default domain name and retry.

Examples:

(In the examples below, the default domain is 'hms.com')

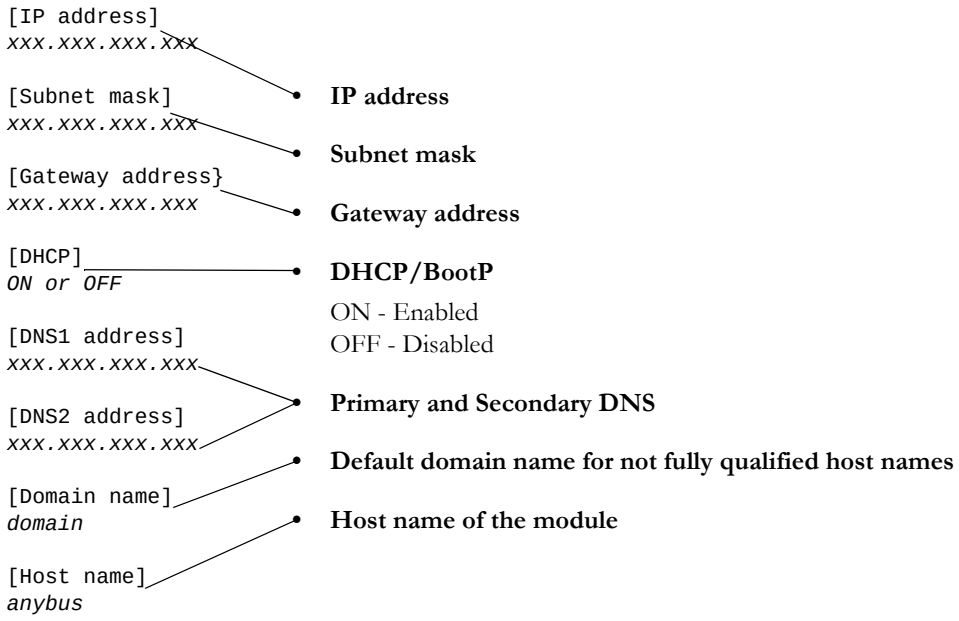
- 'test' becomes 'test.hms.com'
- 'test.hms.com.' becomes 'test.hms.com'
- 'test.a' will if not found become 'test.a.hms.com'

3. Network Configuration

3.1 TCP/IP Settings

To be able to participate on the network, the module needs a valid TCP/IP configuration. These settings are stored in the system file '\cfg\ip.cfg'.

File Format:



The settings in this file may also be affected by...

- Mailbox Commands (See "General Configuration Commands" on page 56)
- EtherNet/IP (See "TCP/IP Interface Object, Class F5h" on page 50)
- DHCP/BootP (See "DHCP/BootP" on page 11)
- HICP (See "HICP (Anybus IPconfig)" on page 11)
- SSI (See "Server Side Include (SSI)" on page 20)
- Switches (See below)

  If the on-board switches are set to 0 (zero), the module will only use the settings stored in '\cfg\ip.cfg'. If not, the switch setting overrides certain settings in this file as follows:

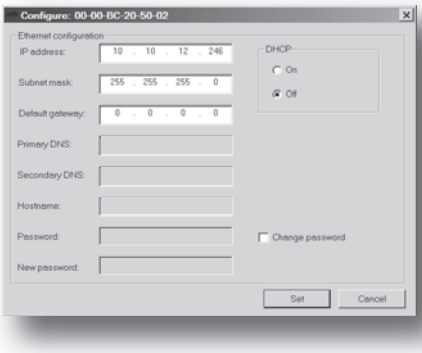
SW1	SW2	SW3	SW4	SW5	SW6	SW7	SW8	DHCP	Subnet	Gateway	IP
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	Determined by '\cfg\ip.cfg'			
OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	Disabled	255.255.255.0	192.168.0.255	192.168.0.1
OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF	Disabled	255.255.255.0	192.168.0.255	192.168.0.2
...	...	...	...	...	...	...	...	...	...	...	...
ON	ON	ON	ON	ON	ON	OFF	ON	Disabled	255.255.255.0	192.168.0.255	192.168.0.253
ON	ON	ON	ON	ON	ON	ON	OFF	Disabled	255.255.255.0	192.168.0.255	192.168.0.254
ON	ON	ON	ON	ON	ON	ON	ON	(invalid setting)			

Note: The switches are read during startup; any changes require a reset in order to have effect.

3.2 HICP (Anybus IPconfig)

The module supports the HICP protocol used by the Anybus IPconfig utility, which can be downloaded free of charge from the HMS website. This utility may be used to access the network settings of any Anybus product connected to the network via UDP port 3250.

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 module will appear as 'ABM-EIP'.



To alter the network settings of the module, double-click on its entry in the list. A window will appear, containing the settings for the module.

Validate the new settings by clicking 'Set'. The new IP configuration will be stored in '\cfg\ip.cfg'.

Optionally, 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'.

The password is stored in the system file '\cfg\hicp.cfg'.

File Format:

```
[Password]
<password>
```

Note: This feature cannot be used to alter the network settings if the mailbox command SET_IP_-CONFIG has been issued (see "Set IP Configuration (SET_IP_CONFIG)" on page 57).

3.3 DHCP/BootP

The module can retrieve the TCP/IP settings from a DHCP or BootP server. If no DHCP server is found, the module will fall back on it's current settings (i.e. the settings currently stored in '\cfg\ip.cfg').

If no current settings are available (i.e. set to 0), the module will halt and indicate an error on the on-board status LEDs (the network configuration can however still be accessed via HICP, see "HICP (Anybus IPconfig)" on page 11.

3.4 Speed and Duplex

The module supports 10 or 100Mbit operation in full or half duplex. These settings are stored in the system file 'cfg\etn.cfg'. The settings can also be altered via the Ethernet Link Object, see "Ethernet Link Object, Class F6h" on page 53.

File Format:

```
[AutoNeg]
xxx
```
Auto-negotiation; valid settings: 'ON' or 'OFF'

```
[Speed]
xxx
```
Speed; valid settings: '100' or '10'

```
[Duplex]
xxxx
```
Duplex; valid settings: 'FULL' or 'HALF'

3.5 IP Access Control

It is possible to specify which IP addresses that are permitted to connect to the module. This information is stored in the system file '\cfg\ip_accs.cfg'.

File Format:

```
[Web]
xxx.xxx.xxx.xxx
```
Nodes listed here may access the web server.

```
[FTP]
xxx.xxx.xxx.xxx
```
Nodes listed here may access the FTP server.

```
[Ethernet/IP]
xxx.xxx.xxx.xxx
```
Nodes listed here may connect to the module via EtherNet/IP.

```
[All]
xxx.xxx.xxx.xxx
```
Fallback setting, used by the module when one or several of the keys above are omitted.

Note: '*' may be used as a wildcard to select IP series.

4. FTP Server

4.1 General

The module features a built in FTP server, which can be used to upload/download files to the filesystem using a standard FTP client.

The FTP server can be disabled using the mailbox command DISABLE_FTP_SERVER, see "Disable FTP server (DISABLE_FTP_SERVER)" on page 66.

The following port numbers are used by the FTP server:

- TCP, port 20 (FTP data port)
- TCP, port 21 (FTP command port)

The FTP server can handle a maximum of 16 simultaneous connections.

Security Levels

The FTP-server features two security levels; admin and normal. Security is set at a per-user basis, or globally using the mailbox command GLOBAL_ADMIN_MODE (see "Global Admin Mode (GLOBAL_ADMIN_MODE)" on page 67).

- **Normal level**

  The root directory will be '\cfg'.

- **Admin level (User or global)**

  The root directory will be '\', i.e. the user has unrestricted access to the file system.

  If the mailbox command GLOBAL_ADMIN_MODE (see "Global Admin Mode (GLOBAL_ADMIN_MODE)" on page 67) is issued during startup, the module will run in global admin mode, causing the FTP-server to accept any username/password combination. Connecting to the module when running in global admin mode provides unrestricted access to the filesystem.

User Accounts

The user accounts are stored in two files, which are protected from web access:

- **'\cfg\pswd\sys_pswd.cfg'**

  This file holds the user accounts for normal-level users.

- **'\pswd\ad_pswd.cfg'**

  This file holds the user accounts for admin-level users.

File Format:

The format of these files are as follows:

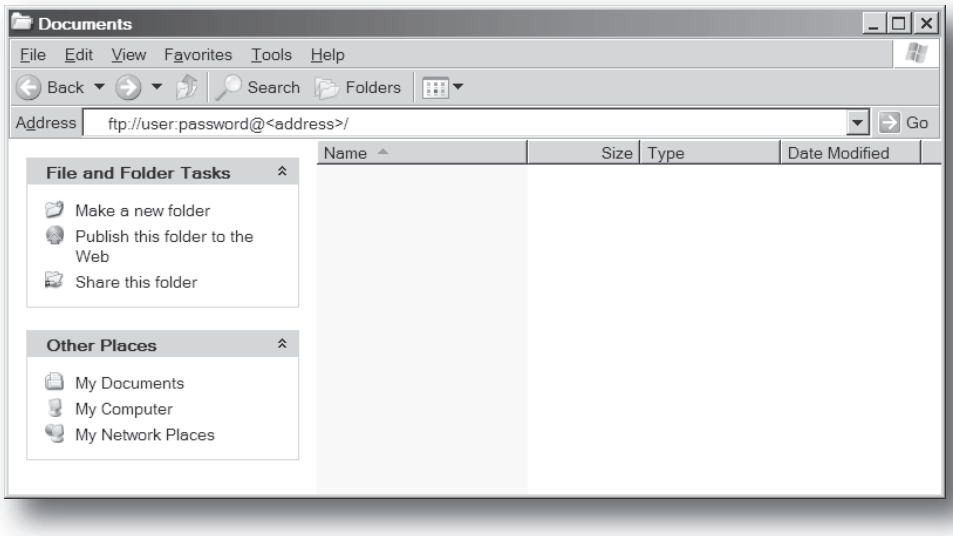
```
Username1:Password1
Username2:Password2
Username3:Password3
```

Note: If no valid user accounts has been defined, the module will fall back to global admin mode.

FTP Connection Example (Windows Explorer)

The built in FTP client in Windows Explorer can easily be used to access the filesystem as follows:

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



5. Web Server

5.1 General

The Anybus module features a flexible web server with SSI capabilities. The built in web pages can be customized to fit a particular application and allow access to I/O data and configuration settings.

The web server communicates through port 80 and can handle a maximum of 48 simultaneous connections.

See also...

- "Disable Web Server (DISABLE_WEB_SERVER)" on page 64

Protected Files

For security reasons, the following files are protected from web access:

- Files located in '\cfg\pswd'
- Files located in '\pswd'
- Files located in a directory which contains a file named 'web_accs.cfg'

Default Web Pages

The Anybus module contains a set of virtual files that can be used when building a web page for configuration of network parameters. These virtual files can be overwritten (not erased) by placing files with the same name in the root of disc 0.

This makes it possible to, for example, replace the HMS logo by uploading a new logo named '\logo.jpg'. It is also possible to make links from a web page to the virtual configuration page. In that case the link shall point to '\config.htm'.

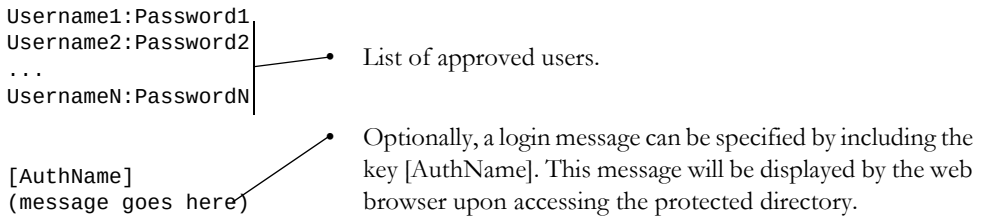
These virtual files are:

```
\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
\index.htm       l          - Points to the contents of config.htm
\eth_stat.html              - Configuration frame page
\cip_stat.html              - Configuration form page
\ip_config.shtm             - Configuration form page
\smtp_config.shtm           - Configuration store page
\style.css                  - HMS logo
\arrow_red.gif              - Configuration picture
```

5.2 Authorization

Directories can be protected from web access by placing a file called 'web_accs.cfg' in the directory to protect. This file shall contain a list of users that are allowed to access the directory and its subdirectories.

File Format:



The list of approved users can optionally be redirected to one or several other files.

Example:

In this example, the list of approved users will be loaded from the files 'here.cfg' and 'too.cfg'.

```
[File path]
\i\put\it\over\here.cfg
\i\actually\put\some\of\it\over\here\too.cfg

[AuthName]
Yeah. Whatsda passwoid?
```

Note that when using this feature, make sure to put the user/password files in a directory that is protected from web access, see "Protected Files" on page 15.

5.3 Content Types

By default, the following content types are recognized by their file 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, *.hpp
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.wmlscriptc	*.wmlsc
text/xml	*.xml
application/pdf	*.pdf

It is possible to configure/reconfigure the reported content types, and which files that shall be scanned for SSI. This is done in the system file '\http.cfg'.

File Format:

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

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

Note: Up to 50 content types and 50 SSI file types may be specified in this file.

6. SMTP Client

6.1 General

The built in email client can send predefined email messages based on trigger-events in the dual port memory (DPRAM). The application can also use the client directly via the mailbox interface.

The client supports SSI, however note that some SSI functions cannot be used in email messages (specified separately for each SSI function).

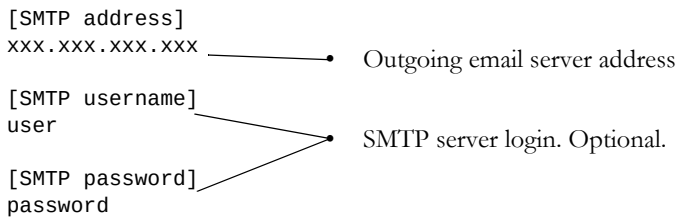
See also...

- "Server Side Include (SSI)" on page 20
- "Send Email (SEND_EMAIL)" on page 139

Server Settings

The module needs a valid SMTP server configuration in order to be able to send email messages. These settings are stored in the system file '\cfg\smtp.cfg'.

File Format:



See also...

- "Set SMTP Configuration (SET_SMTP_CONFIG)" on page 62
- "Get SMTP Configuration (GET_SMTP_CONFIG)" on page 63"Send Email (SEND_EMAIL)" on page 139

Event-Triggered Messages

As mentioned previously, the email client can send predefined message based on events in the DPRAM. In operation, this works as follows:

1. The trigger source is fetched from the dual port memory
2. A logical AND is performed between the trigger source and a mask value
3. The result is compared to a reference value according to a specified operand
4. If the end result is true, the email is sent to the specified recipient(s).

Which events that shall cause a particular message to be sent, is specified separately for each message. For more information, see "Email Definitions" on page 19.

Note that the DPRAM is scanned once every 0.5 second, i.e. a trigger-event must be present longer than 0.5 seconds to ensure that it is detected by the Anybus module.

6.2 Email Definitions

The email definitions are stored in the following two directories:

- **'\cfg\email'**
  This directory holds up to 10 messages which can be altered by normal-level FTP-users.
- **'\email'**
  This directory holds up to 10 messages which can be altered by admin-level FTP-users.

Email definition files must be named 'email_1.cfg', 'email_2.cfg'... 'email_10.cfg' in order to be properly recognized by the module.

File Format:

```
[Register]
Area, Offset, Type

[Register Match]
Value, Mask, Operand

[To]
recipient

[From]
sender

[Subject]
subject line

[Headers]
Optional extra headers

[Message]
message body
```

Key	Value	Scanned for SSI
Area	Source area in DPRAM. Possible values are 'IN' or 'OUT'	No
Offset	Source offset, written in decimal or hexadecimal.	
Type	Source data type. Possible values are 'byte', 'word', and 'long'	
Value	Used as a reference value for comparison.	
Mask	Mask value, applied on the trigger source prior to comparison (logical AND).	
Operand	Possible values are '<', '=' or '>'	
To	Email recipient	Yes
From	Sender email address	
Subject	Email subject. One line only.	
Headers	Optional; may be used to provide additional headers.	
Message	The actual message.	

Note: Hexadecimal values must be written with the prefix '0x' in order to be recognized by the module.

7. Server Side Include (SSI)

7.1 General

Server Side Include (from now on referred to as SSI) functionality enables dynamic content to be used on web pages and in email messages.

SSI are special commands embedded in the source document. When the Anybus module encounters such a command, it will execute it, and replace it with the result (when applicable).

Syntax

The 'X's below represents a command opcode and parameters associated with the command.

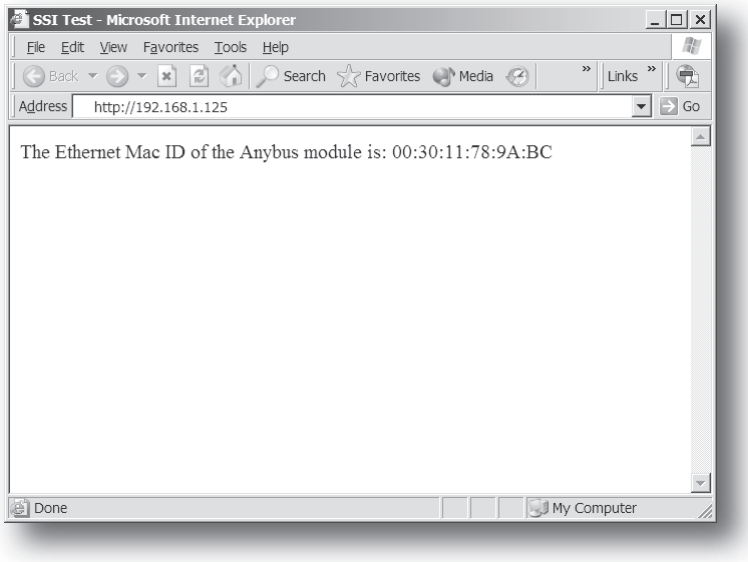
```
<?--#exec cmd_argument='XXXXXXXXXXXXXXXXXXXXX'-->
```

Example

The following example causes a web page to display the Ethernet Mac ID of the module:

```
<HTML>
<HEAD><TITLE>SSI Test</TITLE></HEAD>
<BODY>
The Ethernet Mac ID of the Anybus module is:
<?--#exec cmd_argument='DisplayMacID'-->
</BODY>
</HTML>
```

Resulting web page:



7.2 Functions

DisplayMacID

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

Syntax:

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

DisplaySerial

This function returns the serial number of the Anybus module.

Syntax:

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

DisplayFWVersion

This function returns the main firmware revision of the Anybus module.

Syntax:

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

DisplayBLVersion

This function returns the bootloader firmware revision of the Anybus module.

Syntax:

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

DisplayIP

This function returns the currently used IP address.

Syntax:

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

DisplaySubnet

This function returns the currently used Subnet mask.

Syntax:

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

DisplayGateway

This function returns the currently used Gateway address.

Syntax:

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

DisplayDhcpState

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

Syntax:

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

StoreIPConfig

Note: This function cannot be used in email messages.

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

Syntax:

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

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

Accepted fields in form:

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

Default output:

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

GetText

Note: This function cannot be used in email messages.

This SSI function gets the text from an object and stores it in the OUT area.

Syntax:

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

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

Default output:

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

printf

This SSI function includes a formatted string, which may contain data from the Anybus IN/OUT area, on a web page. The formatting of the string is equal to the standard C function printf().

Syntax:

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

Like the standard C function printf() the "String to write" 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 ABCDEF 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*)	Read a signed byte from position *offset* in the IN area
InReadUByte(*offset*)	Read an unsigned byte from position *offset* in the IN area
InReadSWord(*offset*)	Read a signed word from position *offset* in the IN area
InReadUWord(*offset*)	Read an unsigned word from position *offset* in the IN area
InReadSLong(*offset*)	Read a signed longword from position *offset* in the IN area
InReadULong(*offset*)	Read an unsigned longword from position *offset* in the IN area
InReadString(*offset*)	Read a string (char*) from position *offset* in the IN area
InReadFloat(*offset*)	Read a floating point (float) value from position *offset* in the IN area
OutReadSByte(*offset*)	Read a signed byte from position *offset* in the OUT area
OutReadUByte(*offset*)	Read an unsigned byte from position *offset* in the OUT area
OutReadSWord(*offset*)	Read a signed word (short) from position *offset* in the OUT area
OutReadUWord(*offset*)	Read an unsigned word (short) from position *offset* in the OUT area
OutReadSLong(*offset*)	Read a signed longword (long) from position *offset* in the OUT area
OutReadULong(*offset*)	Read an unsigned longword (long) from position *offset* in the OUT area
OutReadString(*offset*)	Read a null-terminated string from position *offset* in the OUT area
OutReadFloat(*offset*)	Read a floating point (float) value from position *offset* in the OUT area
MbReadSByte(*id*)	Read a signed byte (short) from the application via the mailbox interface
MbReadUByte(*id*)	Read an unsigned byte (short) from the application via the mailbox interface
MbReadSWord(*id*)	Read a signed word from the application via the mailbox interface
MbReadUWord(*id*)	Read an unsigned word from the application via the mailbox interface
MbReadSLong(*id*)	Read a signed longword from the application via the mailbox interface
MbReadULong(*id*)	Read an unsigned longword from the application via the mailbox interface
MbReadString(*id*)	Read a null-terminated string from the application via the mailbox interface
MbReadFloat(*id*)	Read a floating point (float) value from the application via the mailbox interface
CipReadSByte(*class*, *inst*, *attr*)	Read a signed byte from a CIP-object
CipReadUByte(*class*, *inst*, *attr*)	Read an unsigned byte from a CIP-object
CipReadSWord(*class*, *inst*, *attr*)	Read a signed word from a CIP-object

Argument	Description
CipReadUWord(*class*, *inst*, *attr*)	Read an unsigned word from a CIP-object
CipReadSLong(*class*, *inst*, *attr*)	Read a signed longword from a CIP-object
CipReadULong(*class*, *inst*, *attr*)	Read an unsigned longword from a CIP-object
CipReadFloat(*class*, *inst*, *attr*)	Read a floating point value from a CIP-object
CipReadShortString(*class*, *inst*, *attr*)	Read a short string from a CIP-object
CipReadString(*class*, *inst*, *attr*)	Read a null-terminated string from a CIP-object
CipReadUByteArray(*class*, *inst*, *attr*)	Read an unsigned byte-array from a CIP-object
CipReadUWordArray(*class*, *inst*, *attr*)	Read an unsigned word-array from a CIP-object
CipReadULongArray(*class*, *inst*, *attr*)	Read an unsigned longword-array from a CIP-object

scanf

Note: This function cannot be used in email messages.

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 OUT area 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*)	Write a byte to position *offset* in the OUT area
OutWriteWord(*offset*)	Write a word to position *offset* in the OUT area
OutWriteLong(*offset*)	Write a long to position *offset* in the OUT area
OutWriteString(*offset*)	Write a string to position *offset* in the OUT area
OutWriteFloat(offset)	Write a floating point value to position *offset* in the OUT area
MbWriteByte(*id*)	Write a byte to the application via the mailbox interface
MbWriteWord(*id*)	Write a word to the application via the mailbox interface
MbWriteLong(*id*)	Write a longword to the application via the mailbox interface
MbWriteString(*id*)	Write a string to the application via the mailbox interface
MbWriteFloat(*id*)	Write a floating point value to the application via the mailbox interface
CipWriteByte(*class, inst, attr*)	Write a byte value to a CIP-object
CipWriteWord(*class, inst, attr*)	Write a word value to a CIP-object
CipWriteLong(*class, inst, attr*)	Write a longword to a CIP-object
CipWriteFloat(*class, inst, attr*)	Write a floating point value to a CIP-object

Default output:

```
Write succeeded
Write failed
```

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>
```

SaveToFile

Note: This function cannot be used in email messages.

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
```

SaveDataToFile

Note: This function cannot be used in email messages.

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
```

DisplayScannerState

This function returns the current scanner mode (run or idle state). It returns the text shown in the brackets below, the first string if the scanner mode is run, the second if the scanner mode is idle.

Syntax:

```
<?--#exec cmd_argument='DisplayScannerState
( "Output when Run", "Output when Idle" )'-->
```

SetScannerState

Note: This function cannot be used in email messages.

This function is used to set the EtherNet/IP Scanner to Run or Idle. A variable called 'scanner_state', with the value 'run' or 'idle' (other values will be ignored), shall be present when the form is submitted.

Syntax:

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

Default output:

```
Failure                     - Change scanner mode not possible
```

See also...

- "Identity Object, Class 01h" on page 37
- "Set Scanner Mode (SET_SCANNER_MODE)" on page 131

StoreUCMMTimeout

This SSI function is used to store a new UCMM timeout (ms) to flash memory. Valid range is 1000 - 30000.

Syntax:

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

Accepted fields in form:

```
SetUCMMTimeout
```

Default output:

```
Success                     - New UCMM timeout successfully stored
Failure                     - Failed to store new UCMM timeout
```

DisplayUCMMTimeout

This SSI function is used to display the currently used UCMM timeout value on a web page.

Syntax:

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

7.2.1 SSI Scanlist Functionality

The possibility to check and change the scanlist of the module from the default web pages, is disabled by default. The functionality can be enabled using the mailbox "Scanlist Configuration Web Page Control", see page 68.

If you replace the default web pages with new ones, the functionality will be enabled by default.

Please note, that if the scanlist is to be changed, the module must be in idle mode. The mode of the module can be changed either by a CIP command, by a the SSI command "SetScannerState" (see page 27) or the mailbox "Set Scanner Mode (SET_SCANNER_MODE, see page 131). The module can also be set in idle mode checking the box at the bottom of the first default web page shown.

GetScanListArray

This SSI function is used to get the configured EtherNet/IP scanlist. The output is a javascript array with the requested name, where the first entry is a header row describing each entry item, followed by one entry for each configured connection.

In combination with "EditScanEntry", this function can be used to build up a scanlist configuration web page. There is a default webpage in the flash of the module that holds scanlist configuration pages. These have to be enabled using the "Scanlist configuration web page control" mailbox, see page 68.

Syntax in Java code:

```
<?--#exec cmd_argument='GetScanListArray( "ScanlistArrayVariableName" )'-->
```

(When this file is read form the module the SSI-command above is replaced with the array variable "ScanlistArrayVariableName" that holds the ScanList.)

Output Example (output for a scanlist containing two entries):

```
ScanlistArrayVariableName[0] = ("SN","IP address","Timeout Mult","O->T
RPI","O->T Param", "T->O RPI","T->O Param","O->T ConnPoint","O->T Offset",
"T->O ConnPoint", "T->O Offset", "Config inst");
ScanlistArrayVariableName[1] =
(1,"10.10.21.28",0,100000,18454,100000,10258,150,0,100,0,1);
ScanlistArrayVariableName[2] =
(1,"10.10.21.29",0,100000,18454,100000,10258,150,8,100,8,1);
```

ScanlistArrayVariableName.length can be used to get the number of rows in the table.

Argument	Description
SN	Connection serial number
IP address	Target IP address
Timeout Mult	Connection timeout multiplier: 0 -> 4 x RPI 1 -> 8 x RPI 2 -> 16 x RPI 3 -> 32 x RPI 4 -> 64 x RPI 5 -> 128 x RPI 6 -> 256 x RPI 7 -> 512 x RPI
O->T RPI T->O RPI	Originator to Target/Target to Originator RPI (Requested packet interval) in µs.
O->T Param T->O Param	Originator to Target/Target to Originator. Information about the connection type and size[a]. Connection type mask: 0x6000 (24576) Multicast connection: 0x2000 (8192) Point-to-point connection: 0x4000 (16384) Connection size mask: 0x01FF (size in bytes, 511) See table below for a detailed description.
O->T ConnPoint T->O ConnPoint	Originator to Target/Target to Originator Connection point (normally the Assembly object instance in the target, to which the connection shall connect).
O->T Offset T->O Offset	Offset (in number of 16 bit words) in DPRAM IN/OUT area, where this connection data will be represented.

a. The data in this argument is presented in decimal form in the messages.

The table below gives descriptions of the bits in the argument O->T Param, T->O Param.

Bits	Parameter	Description
0-8	Connection Size	The maximum size , in bytes, of the data for each direction (where applicable) of the connection. For a variable sized connection, the size shall be the maximum size of the buffer for any transfer. The actual size of the transfer for a variable connection shall be equal to or less than the size specified for the connection. The maximum buffer size shall be dependent on the links that the connection traverses. The connection size includes the sequence count and the 32 bit real time header, if present.
9	Fixed/Variable	The product only support fixed size connection. The amount of data on each transmission shall be the size specified in the Connection Size parameter. This bit shall always be set to 0 (fixed).
10-11	Priority	00: Low 01: High 10: Scheduled (recommended 11: Urgent
12	Reserved	Set to zero.
13-14	Connection Type	00: (reserved) 01: Multicast, only supported for T->O. 10: Point to Point 11: (reserved)
15	Redundant Owner	Always 0, to indicate an exclusive-owner, input-only or listen-only connection.

EditScanEntry

This SSI function is used to edit an entry in the configured EtherNet/IP scanlist. Two variable entries are present in the form when it is submitted, "ActionObjectName", specifying which action to take on an entry and "EntryObjectName", with information about the entry to modify.

In combination with "GetScanListArray", this function can be used to build up a scanlist configuration web page similar to the one enabled by the "Scanlist configuration web page control" mailbox.

Syntax:

```
<?--#exec cmd_argument='EditScanEntry( "ActionObjectName", "EntryObjectName"
)'-->
```

Argument	Description
ActionObjectName	Specifies the name of the HTML form entry holding information about which action to take on a scanlist entry: "delete" request to delete a scanlist entry "add" request to add a scanlist entry "edit" request to edit a scanlist entry
EntryObjectName	Specifies the name of the HTML form entry holding information about the entry the action is valid for. Content differs depending on action: "delete" contains the serial number of the scanlist entry to delete "add" string of entry data (see "GetScanListArray" on page 30 for details). Independent of Set Connection serial number, the first free place is used. Example: "0,"10.10.21.28", 0, 100000, 18454, 100000,10258, 150, 16, 100, 16, 1"[a] "edit" string of entry data Example: "2,"10.10.21.28", 0, 100000, 18454, 100000,10258, 150, 16, 100, 16, 1"[a]

a. Param data has to be in decimal format. (18458 = 4816h, 10258 = 2812h). The command will result in 10 bytes of I/O data in each direction (a two byte counter and a four byte run/idle header occupies 6 bytes of O->T data and a two byte counter occupies two bytes of T->O data) with 16 words offset in DPRAM transferred every 100 ms.

Note: The list is renumbered at repower to remove unused connection serial numbers.

7.3 Changing SSI output

There is 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()".

7.3.1 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 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

7.3.2 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) )'-->
```

8. Ethernet/IP

The module act as a Scanner Class product on the EtherNet/IP network. EtherNet/IP is based on the Common Industrial Protocol (CIP) which is also the application layer used by DeviceNet and ControlNet to exchange data between nodes.

The following port numbers are used for the EtherNet/IP communication:

- **TCP, port 41818**
  Used for encapsulation messages; supports up to 48 concurrent incoming TCP connections.
- **UDP, port 41818**
  Used for encapsulation messaging over UDP.
- **UDP, port 2222**
  Used for I/O data (Class 1, Multicast or Point-to-Point).

System File eip.cfg

EtherNet/IP settings are stored in a system file:

- **'\cfg\eip.cfg'**

File Format:

The file shall contain one or several of the headers below
[UCMM Timeout]
1500
[Select ACD]
1
[ACD activity]
0
[Remote MAC]
00:00:00:00:00:00
[ARP PDU]
00000000000000000000000000000000000000000000000000000000

If the file does not exist or if the UCMM timeout value is outside the valid range (1000 - 30000 ms), the default value (1500 ms) will be used. If the value in this file is changed during runtimem the new value will ot be used until the next restart of the module.

Select ACD is mapped to attribute #10 in the TCP/IP object (F5h). If the eip.cfg file does not exist, ACD will be enabled. Valid values are 1 to enable and 0 to disable ACD.

The remaining parameters in the eip.cfg file are members of the LastConflictDetected structure in attribute #11 of the TCP/IP object (F5h). The parameters are written when an ACD conflict occurs and read when a Get_Attribute_Single or a Get_Attributes_All service is received. The parameters are set to 0 if a Set_Attribute_Single service is received. Each by in Remote MAC and ARP PDU is represented by tow digit hexadecimal number.

See also ...

- "TCP/IP Interface Object, Class F5h" on page 50

8.1 Implemented Objects

EtherNet/IP requires some mandatory objects; these are implemented as well as some vendor specific objects. The mandatory objects are the ones in the specification from ODVA.

The following vendor specific objects are implemented:

- Identity Object, Class 01h
- Message Router, Class 02h
- Assembly Object, Class 04h
- Connection Manager Object, Class 06h
- Diagnostic Object, Class AAh
- Parameter Data Input Mapping Object, Class B0h
- Parameter Data Output Mapping Object, Class B1h
- Connection Configuration Object, Class F3h
- Port Object, Class F4h
- TCP/IP Interface Object, Class F5h
- Ethernet Link Object, Class F6h

8.1.1 Identity Object, Class 01h

Services

Class services: Get Attribute All
Get Attribute Single

Instance services: Get Attribute All
Get Attribute Single
Set Attribute Single
Reset (See "Network Reset Handling" on page 9)

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Instance Attributes

#	Access	Name	Type	Value	Description
1	Get	Vendor ID	UINT	Default: 005Ah	HMS Industrial Networks AB[a]
2	Get	Device Type	UINT	Default: 0000h	Generic Device[a]
3	Get	Product Code	UINT	Default: 0022h	Anybus-M Ethernet[a]
4	Get	Revision	Struct of:		-
			USINT		Major fieldbus version[a]
			USINT		Minor fieldbus version[a]
5	Get	Status	WORD	-	Device status, see table below
6	Get	Serial Number	UDINT	Module serial number	Serial number of the module
7	Get	Product Name	SHORT_STRING	Anybus-M EtherNet/IP	Name of product[a]
103	Set[b]	Scanner Mode	USINT	-	0: Idle mode 1: Run mode

a. Can be customized using mailbox commands, see "EtherNet/IP Specific Commands" on page 111.
b. A 'Set'-request will result in an 'Object state conflict'-response if change over network has been disabled. See also "Set Scanner Mode (SET_SCANNER_MODE)" on page 131

Status Attribute

bit(s)	Name	Description
0	Module Owned	-
1	(reserved)	-
2	Configured	-
3	(reserved)	-
4 - 7	Extended Device Status	(See table on the right)
8	Minor recoverable fault	-
9	Minor recoverable fault	-
10	Major recoverable fault	-
11	Major unrecoverable fault	-
12 - 15	(reserved)	-

Extended Device Status

Value	Meaning
0000b	Unknown
0010b	Faulted I/O Connection
0011b	No I/O connection established
0100b	Non volatile configuration bad
0110b	Connection in Run mode
0111b	Connection in Idle mode

8.1.2 Message Router, Class 02h

Services

Class services: -
Instance services: -

8.1.3 Assembly Object, Class 04h

Services

Class services: Get Attribute Single

Instance services: Get Attribute Single
Set Attribute Single

Description

The Assembly Object uses static assemblies. The instance IDs used are in the vendor specific range.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0002h	Revision 2
2	Get	Max Instance	UINT	-	The highest initiated instance no.

Instances

#	Contents	Description
3	Heartbeat instance	No attributes implemented for this instance. Used as a heartbeat instance for connections only accessing one “real” connection point.
100	Produced Data (target)	This instance only exist if CREATE_CON_TARGET has been issued, and holds data produced by the module.
150	Consumed Data (target)	This instance only exist if CREATE_CON_TARGET has been issued, and holds data consumed by the module. The ‘Run/Idle’ information is discarded.
160 - 164	User configurable consumed data[a]	These instances are created by the end user using a configuration tool. The user can select data size and offset for the instance in the memory area of the scanner. Originators can later connect to a created instance in the O->T direction.
165- 169	User configurable produced data[a]	These instances are created by the end user using a configuration tool. The user can select data size and offset for the instance in the memory area of the scanner. Originators can later connect to a created instance in the T->O direction.
101	Consumed Data (all connections)	Equals the contents of instances 768...832
102	Produced Data (all connections)	Equals the contents of instances 1024...1087
768	Consumed Data, connection 1	Consumed data associated with I/O connection no. #1 (Configuration Object instance 1)
769	Consumed Data, connection 2	Consumed data associated with I/O connection no. #2 (Configuration Object instance 2)
...	...	...
831	Consumed Data, connection 64	Consumed data associated with I/O connection no. #64 (Configuration Object instance 64)
1024	Produced Data, connection 1	Produced data associated with I/O connection no. #1 (Configuration Object, instance 1)
1025	Produced Data, connection 2	Produced data associated with I/O connection no. #2 (Configuration Object, instance 2)
...	...	...
1087	Produced Data, connection 64	Produced data associated with I/O connection no. #64 (Configuration Object, instance 2)

a. Instances 160-164 and 165- 169 adds slave functionality to the module. Any other node in the network, e.g. other masters, can use these instances to open a connection to the module.

Note: It is also possible to create target areas dynamically.

Instance Attributes

#	Access	Name	Type	Value	Description
3	Get/Set	Data	Array of BYTE	-	Data produced/consumed

8.1.4 Connection Manager Object, Class 06h

Services

Class services: Get Attributes All
Forward Open
Forward Close
Unconnected Send

Instance services: Get Attributes All

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Data	UINT	0001h	Revision 1

Instance Attributes, Instance 01h

#	Access	Name	Type	Description
1	Get	Open Requests	UINT	No. of received 'Forward Open'-requests
2	Get	Open Format Rejects	UINT	No. of 'Forward Open'-requests that have been rejected due to bad format.
3	Get	Open Resource Rejects	UINT	No. of 'Forward Open'-requests that have been rejected due to lack of resources
4	Get	Open Other Rejects	UINT	No. of 'Forward Open'-requests that have been rejected for reasons other than bad format or lack of resources.
5	Get	Close Requests	UINT	No. of received 'Forward Close'-requests.
6	Get	Close Format Rejects	UINT	No. of 'Forward Close'-requests that have been rejected due to bad format.
7	Get	Close Other Rejects	UINT	No. of 'Forward Close'-requests that have been rejected for reasons other than bad format.
8	Get	Connection Timeouts	UINT	No. of connection timeouts.

Class 1 Connections

Class 1 connections are used to transfer I/O data. Each class 1 connection establishes two data transports; one consuming, and one producing.

No. of Supported Originated Class 1 Connections:	64
Max. Input Connection size:	509 bytes
Min. Output Connection size:	504 bytes
Supported Packet Rate (API):	2...3200ms
Supported Trigger Types:	Cyclic

- **Producing Assembly Instances 100, 101, 102**

No. of Supported Target Class 1 Connections:	20 per instance (sharing same transport)
No. of Supported Transports	1
Supported Transport Types	Point-to-Point, Multicast
Supported Packet Rate (API):	2...3200ms
Supported Trigger Types:	Cyclic

Once a class 1 connection has been established, a transport is received. This transport may be of Point-to-Point or Multicast type. If Point-to-Point, the data is transferred using UDP unicast messages, and no other connections can access the data. If Multicast, the data is transferred with UDP multicast messages, and other connections may use the same transport accessing the data.

Producing instances can only be assigned one transport. Therefore, if using Point-to-Point connections, only 1 (one) Class 1 connection can be established. However, 20 connections can be linked to each Multicast transport, allowing 20 Class 1 connections to be established if they all use the same transport.

In order for a connection to use an existing transport, the connection data size must match the data size of the existing transport, or an error response will be returned. If the connection RPI (Requested Packet Interval) does not match the existing connections's API (Actual Packet Interval), the connection will still be established using the API of the existing transport. This API will be returned in the response to the 'Forward Open'-request.

- **Consuming Assembly Instance 150**

No. of Supported Target Class 1 Connections:	1 per instance
No. of Supported Transports	1
Supported Transport Types	Point-to-Point only
Supported Packet Rate (API):	Unlimited

Since consuming instances are used to control the outputs, only one connection is allowed to each consuming instance.The transport used for the connection must be Point-to-Point.

Class 3 Target Connections

Class 3 connections are used to establish connections to the message router. Thereafter the connection is used for explicit messaging. Class 3 connections use TCP connections.

Up to 16 simultaneous class 3 connections towards the message router is supported.

Unconnected Send

If additional communication ports have been registered using REGISTER_PORT, unconnected send requests to the module will be routed to the application using the mailbox message ROUTE_REQUEST.

Up to 64 messages can be pending before a client will get a 'No resource'-response to an unconnected send request.

See also...

- "Register Port (REGISTER_PORT)" on page 123
- "Route Unconnected Send (ROUTE_REQUEST)" on page 126

8.1.5 Diagnostic Object, Class AAh

Services

Class services: Get Attribute All

Instance services: Get Attribute Single

Description

This vendor specific object provides diagnostic information from the module.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Instance Attributes, Instance 01h

#	Access	Name	Type	Description
01h	Get	Module serial number	UDINT	Serial number
02h	Get	Vendor ID	UINT	Manufacturer Vendor ID
03h	Get	Fieldbus Type	UINT	Fieldbus Type
04h	Get	Module Software version	UINT	Module software version
0Ah	Get	Module Type	UINT	Module Type
0Fh	Get	IN Cyclic I/O length	UINT	Size of I/O Input area (in bytes)
10h	Get	IN DPRAM length	UINT	Number of valid IN bytes in DPRAM
11h	Get	IN Total length	UINT	Total number of IN bytes supported
12h	Get	OUT Cyclic I/O length	UINT	Size of I/O Output area (in bytes)
13h	Get	OUT DPRAM length	UINT	Number of valid OUT bytes in DPRAM
14h	Get	OUT Total length	UINT	Total number of OUT bytes supported
18h	Get	MAC ID	Array of USINT	Ethernet MAC ID of the module (6 bytes)
19h	Get	IP Address	UDINT	Currently configured IP address
1Ah	Get	Subnet mask	UDINT	Currently configured subnet mask
1Bh	Get	Gateway address	UDINT	Currently configured gateway address
1Ch	Get	SMTP server address	UDINT	SMTP server address
1Dh	Get	DHCP state	UDINT	0=No DHCP, 1=DHCP available
1Eh	Get	Bootloader version	UDINT	Bootloader firmware version
1Fh	Get	Application interface version	UINT	Application interface software version
20h	Get	Fieldbus software version	UINT	Fieldbus interface software version

8.1.6 Parameter Data Input Mapping Object, Class B0h

Services

Class services: Get Attribute All

Instance services: Get Attribute Single

Description

This object is created dynamically by the application, and may be used to access blocks of Parameter Data using connected- and unconnected messages.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Instance Attributes, Instance 01h

Each attribute within this instance corresponds to a block of Input Parameter Data. Note that the size and location of each block must be specified by the application during initialisation. For more information, see "Parameter Data Input Mapping (PARAMETER_IN_MAP)" on page 114.

#	Access	Name	Type	Description
01h	Get	Data	Array of USINT	Mapped block of Input Data.
02h	Get	Data	Array of USINT	Mapped block of Input Data.
03h	Get	Data	Array of USINT	Mapped block of Input Data.
04h	Get	Data	Array of USINT	Mapped block of Input Data.
05h	Get	Data	Array of USINT	Mapped block of Input Data.
07h	Get	Data	Array of USINT	Mapped block of Input Data.
...	...	...	...	...
32h	Get	Data	Array of USINT	Mapped block of Input Data

8.1.7 Parameter Data Output Mapping Object, Class B1h

Services

Class services: Get Attribute All

Instance services: Get Attribute Single
Set Attribute Single

Description

This object is created dynamically by the application, and may be used to access blocks of Parameter Data using connected- and unconnected messages.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1

Instance Attributes, Instance 01h

Each attribute within this instance corresponds to a block of Output Parameter Data. Note that the size and location of each block must be specified by the application during initialisation. For more information, see "Parameter Data Out Area Mapping (PARAMETER_OUT_MAP)" on page 116.

#	Access	Name	Type	Description
01h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
02h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
03h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
04h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
05h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
07h	Get/Set	Data	Array of USINT	Mapped block of Output Data.
...	...	...	...	...
32h	Get/Set	Data	Array of USINT	Mapped block of Output Data

8.1.8 Connection Configuration Object, Class F3h

Services

Class services:
Create
Delete
Restore
Get Attribute All
Set Attribute Single
Kick Timer (4Bh)
Change Start (4Fh)
Get Status (50h)
Change Complete (51h)
Audit Changes (52h)

Instance services:
Delete
Restore
Get Attribute All
Set Attribute All
Get Attribute Single

Description

This object is used to create and maintain a scanlist for the EtherNet/IP network.

When set in RUN mode, the Anybus-M EtherNet/IP module starts to scan all EtherNet/IP slaves in the network according to the previously configured scanlist. The scanlist can be accessed using Explicit CIP messages via Ethernet or the mailbox interface, and also via the internal web pages using SSI commands, see "SSI Scanlist Functionality" on page 29.

The object specific service parameters can be set and changed by UCMM commands using the mailbox "Send UCMM (SEND_UCMM)", see page 118. The mailbox has to be sent either to the module itself, using the IP address obtained by the mailbox "Get IP configuration (GET_IP_CONFIG)", see page 58, or to the Ethernet port on the module via an external tool.

The module must be in idle mode if the scanlist is to be changed. The mode is controlled by a CIP-command, the SSI command "SetScannerState", see page 27, or the mailbox "Set Scanner Mode (SET_SCANNER_MODE), see page 131.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0002h	Revision 2
2	Get	Max Instance	UDINT	0000 0040h	Maximum instance no. = 64h
3	Get	No. of instances	UDINT	-	-
5	Get	Optional service list	Struct of:		List of optional service codes implemented in this class
		Number services	UINT	2	Number of services in the optional service list
		Optional services	Array of UINT	{ 0Eh, 10h }	The optional service codes
8	Get	Format number	UINT	0065h	Format number for instance attribute 9
9	Set	Edit Signature	UDINT	-	Value written by configuration tool to detect modifications in instance attribute values.
33	Set	Scanner Mode	BOOL	-	0 - Idle mode 1 - Run mode
34	Get	Scanner Capabilities	WORD	00E3h	See CIP specification

Instance Attributes

One instance is created for each connection. All instance attributes except attribute #1 is stored in non-volatile memory. The instances together form the scanlist for the EtherNet/IP network where the module acts as scanner.

#	Access	Name	Type	Description
1	Get	Connection Status	Struct of:	When a connection is not open, this attribute indicates why
		Gen_status	USINT	
		(reserved)	USINT	
		Ext_status	UINT	
2	Set	Connection Flags	WORD	Connection flags
3	Set	Target Device ID	Struct of:	Device identification used by configuration software to identify target devices associated with this instance.
		Vendor_id	UINT	
		Product_code	UINT	
		Major_rev	USINT	
		Minor_rev	USINT	
5	Set	Net connection parameters	Struct of:	-
		Conn_timeout	USINT	Connection Timeout Multiplier
		Xport_class_and_trigger	BYTE	Transport Class and Trigger
		Rpi_OT	UDINT	Originator to Target requested packet interval
		Net_OT	UINT	Originator toTarget network connection parameters *(This attribute specifies the OT connection size)*
		Rpi_TO	UDINT	Target to Originator requested packet interval
		Net_TO	UINT	Target to Originator network target connection parameters *(This attribute specifies the TO connection size)*
6	Set	Connection Path	Struct of:	-
		Open_path_size	USINT	Path size in words (16-bit)
		(reserved)	USINT	Reserved, ignore
		Open connection path	Padded EPATH	Path used in the 'Forward Open'-service of the Connection Manager
7	Set	Config #1 Data	Struct of:	-
		Config_data_size	UINT	Length of Config_data in bytes
		Config_data	Array of Octet	Config #1 data
8	Set	Connection name	Struct of:	-
		Name_size	USINT	Number of characters in the name
		(reserved)	USINT	Reserved, ignore
		Connection_name	STRING2	User assigned connection name encoded in UNICODE
9	Set	Implementation Defined Attribute	Struct of:	-
		Format_number	UINT	0101h
		Impl_defined_data_size	UINT	000Ah
		Impl_defined_data	Array of: UINT, UINT, UINT, UINT, UINT,	 Reserved: (ignore) Offset OT: Offset of data in memory map Reserved: (ignore) Offset TO: Offset of data in memory map Reserved: (ignore)
10	Set	Config #2 Data	Struct of:	-
		Config_data_size	UINT	Length of Config_data in bytes
		Config_data	Array of Octet	Config #2 data

#	Access	Name	Type	Description
11	Set	Proxy Device ID	Struct of:	-
		Vendor_id	UINT	Vendor ID
		Product_type	UINT	Device Type
		Product_code	UINT	Product Code
		Major_rev	USINT	Major Revision
		Minor_rev	USINT	Minor Revision

8.1.9 Port Object, Class F4h

Services

Class services: Get Attribute All
Get Attribute Single

Instance services: Get Attribute All
Get Attribute Single

Description

Instances in this class (New ports) can be registered using the REGISTER_PORT mailbox command, see "Register Port (REGISTER_PORT)" on page 123.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1
2	Get	Max Instance	UINT	0002h	2 is the highest instance number (Default)
3	Get	No. of instances	UINT	0001h	1 instance is implemented (Default)
8	Get	Entry Port	UINT	0002h	Returns the instance of the Port object that describes the port.
9	Get	All Ports	Array of STRUCT {UINT; UINT;}	0000h 0000h 0000h 0000h 0004h 0002h	Array of structure containing attributes 1 and 2 from each instance. Instance 1 is at byte offset 4. Instance 2 is at byte offset 8, etc. The 4 bytes at offset 0 shall be 0. (Default)

Instance Attributes, Instance 02h

#	Access	Name	Type	Value	Description
1	Get	Port Type	UINT	0004h	4 = TCP/IP
2	Get	Port Number	UINT	0002h	Port 2
3	Get	Port Object	Struct of:		-
		Path Size	UINT	0002h	Path Size
		Path	Padded EPATH	20 F5 24 01h	TCP class, Instance 1
4	Get	Port Name	SHORT_STIRNG	'TCP/IP'	Name of port
8	Get	Node Address	Padded EPATH	-	EPATH describing our TCP/IP address

8.1.10 TCP/IP Interface Object, Class F5h

Services

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

Description

This object provides the a mechanism to configure the TCP/IP settings via EtherNet/IP. Note that writing to this object will affect the settings stored in the configuration file 'ip.cfg'.

The TCP/IP settings will be updated immediately when the Interface Configuration (#5) attribute is set.

Class Attributes

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

Instance Attributes

#	Access	Name	Type	Value	Comments
1	Get	Status	DWORD	-	Interface status, see “Status Attribute Description” on page 52
2	Get	Configuration Capability	DWORD	Bit: Value:	
				0 0	BootP client
				1 1	DNS client
				2 -	DHCP Client Value: Meaning: 0 DISABLE_DHCP mailbox received 1 Other
				3 0	DHCP-DNS Update
				4 -	Configuration settable Value: Meaning: 0 TCP_IP_RO mailbox received 1 Other
				5 -	Hardware Configurable Value: Meaning: 0 SET_ETN_CONFIG mailbox 1 Other
				6 0	Interface config requires reset
				7 1	ACD capable
				8 - 31	(not used, set to 0)
3	Get/Set[a]	Configuration Control	DWORD	-	Value: Meaning: 0 Configuration from non-volatile memory (i.e. ‘ethcfg.cfg’) 2 Configuration from DHCP
4	Get	Port Object	Struct of:		Physical link -> Ethernet object
		Path Size	UINT	0002h	2 words
		Path	Padded EPATH	20 F6 24 01h	Ethernet Class, Instance 1
5	Get/Set[a]	Interface Configuration	Struct of:		
		IP Address	UDINT	-	Currently used IP address
		Subnet Mask	UDINT	-	Currently used Subnet mask
		Gateway Address	UDINT	-	Currently used Gateway Address
		Name Server 1	UDINT	-	Primary DNS
		Name Server 2	UDINT	-	Secondary DNS
		Domain Name	STRING	-	Default domain name
6	Get/Set[a]	Host Name	STRING	-	Host name
10	Get/Set	SelectACD	BOOL	-	Enable ACD
11	Get/Set	LastConflictDetected	Struct of:		Last detected address conflict.
		AcdActivity	USINT	-	State of ACD activity when last conflict detected.
		RemoteMAC	Array of 6 USINT	-	MAC address of remote node from the ARP PDU in which a conflict was detected.
		ArpPdu	Array of 28 USINT	-	Copy of the raw ARP PDU in which a conflict was detected.

Status Attribute Description

Bit (s)	Name	Description
0-3	Interface Configuration Status	Indicates the status of the Interface Configuration attribute 0 - The Interface Configuration attribute has not been configured. 1 - The Interface Configuration attribute contains valid configuration obtained from BOOT, DHCP or non-volatile storage. 2- The IP address member of the Interface Configuration attribute contains valid configuration, obtained from hardware settings, e.g. a push-wheel or a thumbwheel.
4	Mcast Pending	Indicates a pending configuration change in the TTL Value and/or Mcast Config attributes. This bit shall be set when either the TTL Value or Mcast Config attribute is set, and shall be cleared the next time the device starts.
6	AcdStatus	Indicates if an address conflict has been detected 0 - No address conflict detected 1 - Address conflict detected
7-31	(reserved)	Set to zero

8.1.11 Ethernet Link Object, Class F6h

Services

Class services: Get Attribute All
Get Attribute Single

Instance services: Get Attribute All
Get Attribute Single
Set Attribute Single
Get and Clear

Description

This object maintains link specific counters and status information for the Ethernet communications interface.

Class Attributes

#	Access	Name	Type	Value	Description
1	Get	Revision	UINT	0001h	Revision 1
2	Get	Max Instance	UINT	0001h	1 is the highest instance number
3	Get	No. of instances	UINT	0001h	1 instance is implemented

Instance Attributes

#	Access	Name	Type	Value	Description
1	Get	Interface Speed	UDINT	10 or 100	The actual speed of the module (MBPS)
2	Get	Interface Flags	DWORD	-	Interface flags
3	Get	Physical Address	Array of 6 USINTS	MAC address	Ethernet MAC address of the module
4	Get	Interface Counters	Struct of:		-
		In Octets	UDINT	-	Octets received on the interface
		In Ucast Packets	UDINT	-	Unicast packets received on the interface
		In NUcast Packets	UDINT	-	Non-unicast packets received on the interface
		In Discards	UDINT	-	Inbound packets with unknown protocol
		In Errors	UDINT	-	Inbound packets that contain errors (In Discards not included)
		In Unknown Protos	UDINT	-	Inbound packets with unknown protocol
		Out Octets	UDINT	-	Octets sent on the interface
		Out Ucast Packets	UDINT	-	Non-unicast packets sent on the interface
		Out Discards	UDINT	-	Outbound packets with unknown protocol
		Out Errors	UDINT	-	Outbound packets that contain errors (Out Discards not included)
5	Get	Media Counters	Struct of:		-
		Alignment Errors	UDINT	-	Frames received that are not an integral number of octets in length.
		FCS Errors	UDINT	-	Frames received that do not pass the FCS check
		Single Collisions	UDINT	-	Successfully transmitted frames which experienced exactly one collision
		Multiple Collisions	UDINT	-	Successfully transmitted frames which experienced more than one collision
		SQE Test Errors	UDINT	-	Number of times SQRE test error message is generated
		Deferred Transmissions	UDINT	-	Frames for which the first transmission attempt is delayed because the medium is busy
		Late Collisions	UDINT	-	Number of times a collision is detected later than 512 bit-times into the transmission of a packet
		Excessive Collisions	UDINT	-	Frames for which a transmission fails due to excessive collisions
		MAC Transmit Errors	UDINT	-	Frames for which transmission fails due to an internal MAC sub-layer receive error
		Carrier Sense Errors	UDINT	-	Times that the carrier sense condition was lost or never asserted when attempted to transmit a frame
		Frame Too Long	UDINT	-	Frames received that exceed the maximum permitted frame size
		MAC Receive Errors	UDINT	-	Frames for which reception on an interface fails due to an internal MAC sub-layer receive error
6	Get/Set	Interface Control	Struct of:		-
		Control Bits	UDINT	-	Interface Control Bits
		Force Interface Speed	UDINT	-	Speed at which the interface shall be forced to operate. Returns 'Object state Conflict' if auto-negotiation is enabled.

9. Fieldbus Specific Mailbox Commands

9.1 Fault Information

When a mailbox command cannot be processed, the Message Information register in the header of the response will indicate that an error occurred (Consult the Anybus-S Parallel Design Guide for more information). If the error code is 'Invalid Other' (Fh), extended error information is available in the Fault Information register (Extended word 8).

The fault codes in the Fault Information register are:

Register Value	Description
0001h	Invalid IP-address or Subnet mask
000Ah	Invalid timeout time
000Ch	Failed to open file or file not found
000Dh	Invalid file descriptor
000Eh	Invalid open method
000Fh	No email server configured
0010h	Command aborted
0011h	Too many registered objects
0012h	Object already registered
0013h	Deregistering invalid object
0015h	Unsupported Command
0016h	Failed to send UCMM command
0017h	No timeout
0018h	Invalid port number
0019h	Duplicate port number
001Ah	EPATH too big
001Bh	Mapping Failed
001Ch	Reset notification unsupported
001Dh	Too many open files
001Eh	Failed to create directory
001Fh	Failed to delete directory
0020h	Failed to rename file
0021h	Failed to move file
0022h	Failed to copy file
0023h	Too many open directories
0024h	Failed to open directory or directory not found
0025h	Failed to resolve hostname with DNS
0026h	Timed out resolving hostname with DNS

9.2 General Configuration Commands

Commands in this category:

Mailbox Commands	Description	Page
Set IP Configuration (SET_IP_CONFIG)	Set network settings	57
Get IP Configuration (GET_IP_CONFIG)	Retrieve the currently used network settings	58
Get MAC Address (GET_MAC_ADDR)	Retrieve the ethernet MAC address from the module	59
Set MAC Address (SET_MAC_ADDR)	Get the ethernet MAC address of the module	60
Disable HICP (HICP_DISABLE)	Disable HICP support	61
Set SMTP Configuration (SET_SMTP_CONFIG)	Configures the SMTP server address	62
Get SMTP Configuration (GET_SMTP_CONFIG)	Configures the SMTP server address	63
Disable Web Server (DISABLE_WEB_SERVER)	This command disables the built in web server	64
Enable Web Server (ENABLE_WEB_SERVER)	This command enables the built in web server	65
Disable FTP server (DISABLE_FTP_SERVER)	This command disables the built in FTP server	66
Global Admin Mode (GLOBAL_ADMIN_MODE)	This command instruct the module to run in global admin mode	67
Scanlist Configuration Web Page Control (SCANLIST_CFG_CTRL)	Enable/disable scanlist configuration web pages	68
Disable Virtual File System (DISABLE_VFS)	Disable the virtual file system	69
Set Speed and Duplex (SET_SPD_CONFIG)	Set speed & duplex	70
Get Speed and Duplex (GET_SPD_CONFIG)	Get speed & duplex	71

9.2.1 Set IP Configuration (SET_IP_CONFIG)

Description

This command may be used to specify the TCP/IP network settings. Note that the specified settings will override the corresponding settings in '\cfg\ip.cfg'.

Note: This command may only be issued during initialisation.

Initiated by	Application
Command no.	0001h
Extended Header	Fault information
Message data	Network settings.
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0001h	0001h	*SET_IP_CONFIG*
Data size	(data size)	(data size)	*Data size in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data word 1	IP address (high)	IP address (high)	
Message data word 2	IP address (low)	IP address (low)	
Message data word 3	Subnet mask (high)	Subnet mask (high)	
Message data word 4	Subnet mask (low)	Subnet mask (low)	
Message data word 5	Gateway address (high)	Gateway address (high)	
Message data word 6	Gateway address (low)	Gateway address (low)	
Message data word 7	Primary DNS (high)	Primary DNS (high)	
Message data word 8	Primary DNS (low)	Primary DNS (low)	
Message data word 9	Secondary DNS (high)	Secondary DNS (high)	
Message data word 10	Secondary DNS (low)	Secondary DNS (low)	
Message data...	Host name (string, null-terminated)	Host name (string, null-terminated)	
	Domain name (string, null-terminated)	Domain name (string, null-terminated)	

9.2.2 Get IP Configuration (GET_IP_CONFIG)

Description

This command returns the currently used network settings.

Note: This command may only be issued during runtime.

Initiated by	Application
Command no.	0002h
Extended Header	-
Message data	-
Response data	Currently used network settings.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0002h	0002h	*GET_IP_CONFIG*
Data size	0000h	(data size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		IP address (high)	Response dataword 1
		IP address (low)	Response dataword 2
		Subnet mask (high)	Response dataword 3
		Subnet mask (low)	Response dataword 4
		Gateway address (high)	Response dataword 5
		Gateway address (low)	Response dataword 6
		Host name (string, null-terminated)	Response data...
		Domain name (string, null-terminated)	

9.2.3 Get MAC Address (GET_MAC_ADDR)

Description

This command returns the MAC address of the module.

Initiated by	Application
Command no.	0018h
Extended Header	-
Message data	-
Response data	MAC Address, 6 bytes

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0018h	0018h	*GET_MAC_ADDR*
Data size	0000h	0006h	*6 bytes of data (3 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		MAC Address (high)	Response dataword 1
		MAC Address (mid)	Response dataword 2
		MAC Address (low)	Response dataword 3

9.2.4 Set MAC Address (SET_MAC_ADDR)

Description

This command temporarily changes the MAC address of the module.

Note: This command may only be issued during initialization.

Initiated by	Application
Command no.	0019h
Extended Header	-
Message data	MAC Address, 6 bytes
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0019h	0019h	*GET_MAC_ADDR*
Data size	0000h	0006h	*6 bytes of data (3 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message dataword 1	MAC Address (high)	MAC Address (high)	
Message dataword 2	MAC Address (mid)	MAC Address (mid)	
Message dataword 3	MAC Address (low)	MAC Address (low)	

9.2.5 Disable HICP (HICP_DISABLE)

Description

This command disables support for HICP (Anybus IP Config).

Initiated by	Application
Command no.	0013h
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0013h	0013h	*HICP_DISABLE*
Data size	0000h	0000h	*(no data)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.6 Set SMTP Configuration (SET_SMTP_CONFIG)

Description

This command may be used to specify the email account settings & the address to the SMTP server. Note that the specified settings will override the settings in '\cfg\smtp.cfg'.

Initiated by	Application
Command no.	0016h
Extended Header	Fault information
Message data	Account settings & SMTP server address
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0016h	0016h	*SET_SMTP_CFG*
Data size	(data size)	(data size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	SMTP Server (String, null-terminated)	SMTP Server (String, null-terminated)	
	Login (String, null-terminated)	Login (String, null-terminated)	
	Password (String, null-terminated)	Password (String, null-terminated)	

- **SMTP Server**

  ASCII string, null terminated (e.g. "192.168.1.42" or "smtp.server.com")

- **Login**

  ASCII string, null terminated (e.g. "Charles_M" or "Edward_G")

- **Password**

  ASCII string, null terminated (e.g. "zombie_001" or "chainsaw")

9.2.7 Get SMTP Configuration (GET_SMTP_CONFIG)

Description

This command returns the currently used email account settings & the address to the SMTP server.

Initiated by	Application
Command no.	0017h
Extended Header	Fault information
Message data	-
Response data	Account settings & SMTP server address

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0017h	0017h	*GET_SMTP_CFG*
Data size	0000h	(data size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		SMTP Server (String, null-terminated)	
		Login (String, null-terminated)	Response data
		Password (String, null-terminated)	

- **SMTP Server**

  ASCII string, null terminated (e.g. “192.77.6.15” or “smtp.server.com”)

- **Login**

  ASCII string, null terminated (e.g. “Charles_M” or “Edward_G”)

- **Password**

  ASCII string, null terminated (e.g. “zombie_001” or “chainsaw”)

This command may be used to specify the address to the SMTP server in ASCII form.

9.2.8 Disable Web Server (DISABLE_WEB_SERVER)

Description

This command disables the onboard web server. The web server is enabled by default.

Initiated by	Application
Command no.	0004h
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0004h	0004h	*DISABLE_WEB_SERVER*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.9 Enable Web Server (ENABLE_WEB_SERVER)

Description

This command enables the onboard web server. The web server is enabled by default.

Initiated by	Application
Command no.	0005h
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0005h	0005h	*ENABLE_WEB_SERVER*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.10 Disable FTP server (DISABLE_FTP_SERVER)

Description

This command disables the FTP server.

Initiated by	Application
Command no.	0006h
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0006h	0006h	*DISABLE_FTP_SERVER*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.11 Global Admin Mode (GLOBAL_ADMIN_MODE)

Description

This command instructs the module to run in Global Admin Mode. For more information, see "Security Levels" on page 13.

Note: This command may only be issued during initialization.

Initiated by	Application
Command no.	000Bh
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	000Bh	000Bh	*GLOBAL_ADMIN_MODE*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.12 Scanlist Configuration Web Page Control (SCANLIST_CFG_CTRL)

Description

This command enables/disables the scanlist configuration web pages in the VFS file system.

Initiated by	Application
Command no.	000Ch
Extended Header	-
Message data	Enable/Disable
Response data	The response indicates if the command was accepted

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	000Ch	000Ch	*SCANLIST_CFG_CTRL*
Data size	0001h	0001h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data	0: Disable Other: Enable		

9.2.13 Disable Virtual File System (DISABLE_VFS)

Description

This command disables the virtual files in the file system.

Note: This command may only be issued during initialization.

Initiated by	Application
Command no.	0011h
Extended Header	-
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0011h	0011h	*DISABLE_VFS*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.2.14 Set Speed and Duplex (SET_SPD_CONFIG)

Description

This command may be used to set the speed and duplex of the ethernet interface. Note that this setting overrides the settings in 'etn.cfg'.

Initiated by	Application
Command no.	009Bh
Extended Header	-
Message data	Configuration
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	009Bh	009Bh	*SET_SPD_CONFIG*
Data size	0001h	0001h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte	Configuration	Configuration	

- **Configuration**

  00h: Auto-negotiation
  01h: 10Mbit, half duplex
  02h: 10Mbit, full duplex
  03h: 100Mbit, half duplex
  04h: 100Mbit, full duplex

9.2.15 Get Speed and Duplex (GET_SPD_CONFIG)

Description

This command retrieves the currently used speed and duplex settings for the ethernet interface.

Initiated by	Application
Command no.	009Ch
Extended Header	-
Message data	-
Response data	Configuration

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	009Ch	009Ch	*GET_SPD_CONFIG*
Data size	0000h	0001h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		Configuration	Response data byte

- **Configuration**

  00h: Auto-negotiation
  01h: 10Mbit, half duplex
  02h: 10Mbit, full duplex
  03h: 100Mbit, half duplex
  04h: 100Mbit, full duplex

9.3 Mailbox Socket Interface

The Anybus module features a transparent socket interface, allowing the application to send and receive transparent data via TCP/IP or UDP/IP. The mailbox socket interface can be used in two modes:

- **Non-blocking**

  All mailbox operations on these sockets will respond directly - not block until the command is performed. Up to 16 simultaneous non-blocking sockets are supported.

  Note: Status information for all non-blocking sockets are available in the fieldbus specific area, see "Fieldbus Specific Area" on page 143.

- **Blocking**

  Blocking sockets means that the Anybus will not respond to further socket commands until the previous one has been completed (However, non-socket related commands can still be processed as normal). Up to 32 simultaneous blocking sockets are supported.

  Note: Blocking sockets do *not* have any status information in the fieldbus specific area.

Extensive use of this functionality may affect the performance of the module negatively.

Commands in this category:

Mailbox Command	Description	Page
Socket non-blocking (SOCKET_NB)	Creates a socket in non-blocking mode.	73
Socket blocking (SOCKET_B)	Creates a socket in blocking mode.	74
Listen (LISTEN)	Starts to listen on a socket for incoming connections.	75
Accept (ACCEPT)	Accepts connections for sockets in blocking mode.	76
Connect (CONNECT)	Tries to connect a socket to a client.	77
Send (SEND)	Sends a message to a connected socket.	79
Receive (RECEIVE)	Receives a message form a connected socket.	80
Send To (SEND_TO)	Sends a message to an unconnected UDP socket to a specified host.	81
Receive From (RECV_FROM)	Receives a message from an unconnected UDP socket.	82
Close (CLOSE)	Closes a socket (and connection).	83
Send Fragment (SEND_FRAG)	Sends a fragment of a message with a maximum total size of 1460 bytes.	84
Receive Fragment (RECV_FRAG)	Receives a fragment of a message with a total maximum size of 140 bytes.	85
Send Fragment To (SEND_FRAG_TO)	Sends a fragment of a message with a total maximum size of 1460 bytes to an unconnected UDP socket.	87
Receive Fragment From (RECV_FRAG_FROM)	Receives a fragment of a message with a total maximum size of 1460 bytes from an unconnected UDP socket.	88
Get Socket Option (GET_SOCKET_OPTION)	Read options from a socket.	90
Set Socket Option (SET_SOCKET_OPTION)	Sets options to a socket	91

9.3.1 Socket Non-Blocking (SOCKET_NB)

Description

This mailbox command creates a socket in non-blocking mode and associates it to a specific port number. If the specified port number is 0, the Anybus module selects a free port.

The response message contains a socket descriptor and the port number. The socket descriptor shall be used on all following operations on the socket. It indicates at which position in the fielbus specific area, information about this socket can be found.

Initiated by	Application
Command no.	0040h
Extended Header	-
Message data	The socket type (TCP or UDP) and the port number to bind the socket to.
Response data	The response indicates if the command was accepted. The response indicates which socket descriptor that is used and the port number the socket is associated to.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0040h	0040h	*SOCKET_NB*
Data size	0004h	0004h	*4 bytes of data (2 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message dataword 1	Socket type	Socket descriptor	
Message dataword 2	Port number	Port number	

- **Socket Type**

Value	Socket type
0001h	TCP socket
0002h	UDP socket

9.3.2 Socket Blocking (SOCKET_B)

Description

This mailbox command creates a socket in blocking mode and associates it to a specific port number. If the specified port number is 0, the Anybus module selects a free port.

The response message contains a socket descriptor and the port number. This descriptor shall be used on all following operations on this socket.

Initiated by	Application
Command no.	003Fh
Extended Header	-
Message data	The socket type (TCP or UDP) and the port number to bind the socket to.
Response data	The response indicates if the command was accepted. The response indicates which socket descriptor that is used and the port number the socket is associated to.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	003Fh	003Fh	*SOCKET_B*
Data size	0004h	0004h	*4 bytes of data (2 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message dataword 1	Socket type	Socket descriptor	
Message dataword 2	Port number	Port number	

- **Socket Type**

Value	Socket type
0001h	TCP socket
0002h	UDP socket

9.3.3 Listen (LISTEN)

Description

This mailbox command makes a socket listen for new connections. If the Anybus module detects a connection request on the specified socket, a new connected socket will be created, and the current socket will continue listening for new connections. This means that multiple hosts can connect to one listening socket simultaneously.

Note: This command can only be used on a TCP socket.

- **Non-blocking sockets**

  Information about active connections on this socket can be read in the fieldbus specific area, see 11-1 "Memory Map" and 11-3 "Socket Status Structure".

- **Blocking sockets**

  Socket descriptors for new connections connected to this socket can be received by the mailbox command ACCEPT, see 9-76 "Accept (ACCEPT)".

Initiated by	Application
Command no.	0041h
Extended Header	Socket Descriptor, Fault Information
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0041h	0041h	*LISTEN*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket descriptor	
Extended word 2	(reserved, set to 0000h)	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	

9.3.4 Accept (ACCEPT)

Description

When a connection request to a listening socket in blocking mode is received, this command receives the socket descriptor of the newly created connected socket.

This command is blocking and will not respond until a connection request is received.

Initiated by	Application
Command no.	0050h
Extended Header	Socket Descriptor, Fault Information, Local Port no, Host Port no, Host IP
Message data	-
Response data	New socket descriptor

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0050h	0050h	*ACCEPT*
Data size	0000h	0002h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	(reserved, set to 0000h)	Local Port No.	
Extended word 3	-	Host Port No.	
Extended word 4	-	Host IP-address word 1	
Extended word 5	-	Host IP-address word 2	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		New socket descriptor	Response dataword

9.3.5 Connect (CONNECT)

Description

This mailbox command tries to establish a connection to a specified IP address and port number.

If the socket is of UDP type this command specifies the peer with which the socket is to be associated, the address is to which datagrams are sent and the only address from which datagrams are received.

If the socket is of TCP type this command attempts to make a connection to another socket. TCP sockets may CONNECT only once, while UDP sockets may use CONNECT multiple times to change their association.

- **Non-blocking sockets**

  If this command is correctly sent, it will be accepted regardless of whether it is possible to establish a connection or not. The result of the operation is available in the fieldbus specific area, see 11-1 “Fieldbus Specific Area”.

- **Blocking sockets**

  This command will block until a connection is established or the connection request is cancelled due to timeout or connection error.

Initiated by	Application
Command no.	0042h
Extended Header	Socket Descriptor, Fault Information, Connection Result
Message data	IP address, Port number
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0042h	0042h	*CONNECT*
Data size	0006h	0006h	*6 bytes of data (3 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	New Socket Descriptor	
Extended word 2	(reserved, set to 0000h)	Connection result	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data word 1	IP address (high)	IP address (high)	
Message data word 2	IP address (low)	IP address (low)	
Message data word 3	Port number	Port number	

- **Connection Result Code (Only for blocking sockets)**

Code	Status
0003h	Connected
0004h	Connection Refused
0005h	Connection Timeout
0006h	Connection Failed

9.3.6 Send (SEND)

Description

This mailbox command writes data to a connected socket. A maximum of 256 bytes of data can be sent using this command.

- **Non-blocking sockets**

  If there isn't enough space available for the data in the output buffers, the response will indicate that the amount of data actually sent was less than requested.

- **Blocking sockets**

  If there isn't buffer space available for the data in the output buffers this command will block until there is.

Initiated by	Application
Command no.	0043h
Extended Header	Socket Descriptor, Fault Information
Message data	Data to send
Response data	(the response holds a copy of the command data)

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0043h	0043h	*SEND*
Data size	(size)	(size)	*Max. 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	(reserved, set to 0000h)	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Data to send	Sent data	

9.3.7 Receive (RECV)

Description

This mailbox command receives data from a connected socket.

If the specified socket is of TCP type this command will return the requested number of bytes from the received data stream. If the available data is less than requested, all available data will be returned.

If the specified socket is of UDP type this command will return the requested amount of data from the next received datagram. If the datagram is smaller than requested, the entire datagram will be returned in the response message. If the datagram is larger than requested, the excess bytes will be discarded.

A maximum of 256 bytes of data can be received using this command.

- **Non-blocking sockets**

  If no data is available on the socket the response will indicate that 0 bytes of data was received.

- **Blocking sockets**

  If this command is called and no data is available the command will block until there is. If the response indicates that 0 bytes of data was received the connection has been closed by the host. The socket however is still valid and must be closed using the mailbox command CLOSE.

Initiated by	Application
Command no.	0044h
Extended Header	Socket Descriptor, Bytes to receive, Fault Information
Message data	-
Response data	Received data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0044h	0044h	*RECV*
Data size	0000h	(size)	*Maximum 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	Bytes to receive (in bytes)	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Received data	Response data

9.3.8 Send To (SEND_TO)

Description

This mailbox command sends a UDP datagram to a specified IP address and port number. A maximum of 256 bytes of data can be sent using this command. (Unconnected UDP sockets only)

Initiated by	Application
Command no.	0045h
Extended Header	Socket Descriptor, IP-address, Port number, Fault Information
Message data	Data to send
Response data	Sent data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0045h	0045h	*SEND_TO*
Data size	(size)	(size)	*Maximum 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket descriptor	
Extended word 2	IP-address (high)	IP-address (high)	*Destination IP address*
Extended word 3	IP-address (low)	IP-address (low)	
Extended word 4	Port number	Port number	*Port number*
Extended word 5	(reserved, set to 0000h)	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Data to send	Sent data	

9.3.9 Receive From (RECV_FROM)

Description

This mailbox command reads the next received datagram from a UDP type socket. The response message contains the IP address and port number of the sender.

If the received datagram is smaller than requested, the entire datagram will be returned in the response message. If the received datagram is larger than requested, the excess bytes will be discarded.

A maximum of 256 bytes of data can be received using this command.

- **Non-blocking sockets**

  If the amount of data available on the socket is less than requested, this is reflected in the data size of the response.

- **Blocking sockets**

  If this command is called and no data is available the command will block until there is.

Initiated by	Application
Command no.	0046h
Extended Header	Socket Descriptor, Bytes to receive, IP-address, Port number, Fault Information
Message data	-
Response data	Received data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0046h	0046h	*RECV_FROM*
Data size	0000h	(size)	*Maximum 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket descriptor	
Extended word 2	Receive data size	IP address (high)	*Senders IP-address*
Extended word 3	(reserved, set to 0000h)	IP address (low)	
Extended word 4	-	Port number	*Sender port number*
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Received data	Response data

9.3.10 Close (CLOSE)

Description

This mailbox command causes a connected socket to shut down and release its socket descriptor.

- **Blocking sockets**

  Commands still blocking on the socket when it is closed will be aborted and return indicating 0010h (Command aborted)

Note: If a host closes a TCP connection while there is still data available to read on the socket in the client, the client socket will be indicated as connected until all data is read. In this case, if the client tries to send data the mailbox response will report "Can't send more".

Initiated by	Application
Command no.	0047h
Extended Header	Socket Descriptor, Fault Information
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0047h	0047h	*CLOSE*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket descriptor	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	

9.3.11 Send Fragment (SEND_FRAG)

Description

This mailbox command is used when sending messages larger than 256 bytes. Internally the fragments are stored in a buffer until the last fragment is received. The message is then sent to the socket. The maximum size of a fragmented message is 1460 bytes.

It is not possible to send multiple fragmented messages simultaneously. A fragmented message must be completely sent before another fragmented message can be sent on the same or another socket.

- **Non-blocking sockets**

  If there isn't enough space available for the data in the output buffers, the response will indicate that the amount of data actually sent was less than requested.

- **Blocking sockets**

  If there isn't buffer space available for the data in the output buffers this command will block until there is.

Initiated by	Application
Command no.	005Eh
Extended Header	Socket descriptor, Fragment Type
Message data	Data to send
Response data	Sent Data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	005Eh	005Eh	*SEND_FRAG*
Data size	(size)	(size)	*Max. 256 bytes/fragment*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket descriptor	
Extended word 2	Fragment type	Fragment type	*See below*
Extended word 3	(reserved, set to 0000h)	No. of sent bytes	*(Only in last fragment)*
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Data to send	Sent data	

- **Fragment Type Value**

Value	Description
0000h	First fragment of a new message
0001h	Subsequent fragment of the message
0002h	Last fragment of the message. When this fragment is sent the entire message will be sent to the socket.

9.3.12 Receive Fragment (RECV_FRAG)

Description

This mailbox command is used to receive fragmented messages larger than 256 bytes from a connected socket. Internally the entire message will be read from the socket to a buffer. The fragments of the message can then be read from the buffer using this command.

If the specified socket is of TCP type this command will return the requested number of bytes from the received data stream. If the available data is less than requested, all available data will be returned.

If the specified socket is of UDP type this command will return the requested amount of data from the next received datagram. If the datagram is smaller than requested, the entire datagram will be returned in the response message. If the datagram is larger than requested, the excess bytes will be discarded.

The maximum size of a fragmented message is 1460 bytes.

- **Non-blocking sockets**

  If no data is available on the socket the response will indicate that 0 bytes of data was received.

- **Blocking sockets**

  If no data is available the command will block until there is. If the response indicates that 0 bytes of data was received the connection has been closed by the host. The socket however is still valid and must be closed using the mailbox command CLOSE.

Initiated by	Application
Command no.	005Fh
Extended Header	Socket descriptor, Fragment Type, Receive Data Size, Bytes Remaining, Fault information
Message data	-
Response data	Received Data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	005Fh	005Fh	*RECV_FRAG*
Data size	0000h	(size)	*Max. 256 bytes/fragment*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	Fragment type	Fragment type	*See below*
Extended word 3	Receive data size[a]	Bytes remaining	
Extended word 4	(reserved, set to 0000h)	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Received data	Response data

a. The receive data size is only used if the Fragment type = 0000h

- **Fragment Type Value**

Value	Description
0000h	Receive first fragment of a new message. This receives a new message from the network. Any unread fragments from earlier received datagrams will be overwritten.
0001h	Receive the next fragment of the message.

9.3.13 Send Fragment To (SEND_FRAG_TO)

Description

This mailbox command sends a UDP datagram to a specified IP address and port number. This command is used when sending a fragment of a message larger than 256 byte. Internally the fragments are stored in a buffer until the last fragment is received. The message is then sent to the socket. The maximum size of a fragmented message is 1460 bytes.

Initiated by	Application
Command no.	005Ch
Extended Header	Socket descriptor, Fragment Type, IP-address, Port number, No. of sent bytes, Fault information
Message data	Data to send
Response data	Sent data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	005Ch	005Ch	*SEND_FRAG_TO*
Data size	(size)	(size)	*Max. 256 bytes/fragment*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	Fragment type	Fragment type	*See below*
Extended word 3	IP-address (high)[a]	IP-address (high)[a]	*Destination IP address*
Extended word 4	IP-address (low)[a]	IP-address (low)[a]	
Extended word 5	Port number[a]	Port number[a]	*Destination Port number*
Extended word 6	(reserved, set to 0000h)	No. of sent bytes	*(Only in last fragment)*
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Data to send	Sent data	

a. IP-address and Port Number shall only be given in the first fragment.

- **Fragment Type Value**

Value	Description
0000h	First fragment of a new message.
0001h	Subsequent fragment of the message
0002h	Last fragment of the message. When this fragment is sent the entire message will be sent to the socket.

9.3.14 Receive Fragment From (RECV_FRAG_FROM)

Description

This mailbox command reads the next received datagram from a UDP type socket. The response message contains the IP address and port number of the sender.

This command is used to receive a fragment of a message larger than 256 bytes. The maximum total size of a fragmented message is 1460 bytes. The maximal size of each fragment is 256 bytes.

If the received datagram is smaller than requested, the entire datagram will be returned in the response message. If the received datagram is larger than requested, the excess bytes will be discarded.

For blocking sockets, the first fragment will block until there is data available on the socket.

Internally the entire message is read from the socket to a buffer. The fragments can then be read from the buffer using this command.

- **Non-blocking sockets**

  If no data is available on the socket the response will indicate that 0 bytes of data was received.

- **Blocking sockets**

  If this command is called but there is no data available on the socket the command will block and not return until there is data available.

Initiated by	Application
Command no.	005Dh
Extended Header	Socket descriptor, Fragment Type, Received data size, Bytes remaining, IP-address, port number, Fault information
Message data	-
Response data	Received data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	005Dh	005Dh	*RECV_FRAG_FROM*
Data size	0000h	(size)	*Max. 256 bytes/fragment*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket descriptor	Socket Descriptor	
Extended word 2	Fragment type	Fragment type	
Extended word 3	Receive data size	Bytes remaining	
Extended word 4	(reserved, set to 0000h)	IP-address (high)	*The senders IP address*
Extended word 5	-	IP-address (low)	
Extended word 6	-	Port number	*The senders port number*
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Received data	Response data

- **Fragment Type Value**

Value	Description
0000h	Receive first fragment of a new message. This receives a new message from the network. Any unread fragments from earlier received datagrams will be overwritten.
0001h	Receive the next fragment of the message.

9.3.15 Get Socket Option (GET_SOCKET_OPTION)

Description

This command reads options from a socket.

Initiated by	Application
Command no.	0051h
Extended Header	Socket descriptor, Socket Option
Message data	-
Response data	Option Data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0051h	0051h	*GET_SOCKET_OPTION*
Data size	0000h	Option data size	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket Descriptor	Socket Descriptor	
Extended word 2	Socket Option HI	Socket Option HI	
Extended word 3	Socket Option LO	Socket Option LO	
Extended word 4			
Extended word 5			
Extended word 6			
Extended word 7			
Extended word 8			
		Option Data	Response data

Socket Options

The following options are used to Get settings from a socket:

SO_LINGER
SO_KEEPALIVE
SO_REUSEADDR
IP_MULTICAST_TTL
IP_MULTICAST_LOOP

For more information see section “Socket Options” on page 92.

9.3.16 Set Socket Option (SET_SOCKET_OPTION)

Description

This command changes the settings for a specified socket.

Initiated by	Application
Command no.	0052h
Extended Header	Socket descriptor, Socket Option
Message data	Option Data
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0052h	0052h	*SET_SOCKET_OPTION*
Data size	Option data size	Option data size	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Socket Descriptor	Socket Descriptor	
Extended word 2	Socket Option HI	Socket Option HI	
Extended word 3	Socket Option LO	Socket Option LO	
Extended word 4			
Extended word 5			
Extended word 6			
Extended word 7			
Extended word 8		Fault Information	
Message data (size depending on Socket Option)	Option Data	Option Data	

Socket Options

The following options are used to Set settings on a socket:

SO_LINGER
SO_KEEPALIVE
SO_REUSEADDR
IP_MULTICAST_TTL
IP_MULTICAST_LOOP
IP_ADD_MEMBERSHIP
IP_DROP_MEMBERSHIP
TCP_NODELAY

For more information see section “Socket Options” on page 92.

9.3.17 Socket Options

Name	Socket Option Value	Data Type	Description
SO_LINGER	00000080h	Struct of: UINT32 (l_onoff) UINT32 (l_linger)	Controls the action taken when unsent data is queued on a socket that is being closed. This option is only valid for TCP sockets. l_onoff 0:Linger OFF (default) Other:Linger ON l_linger Normally defines the linger timeout. NOT SUPPORTED, ALWAYS SET TO 0. If SO_LINGER is disabled, Socket Close returns immediately and the connection is closed in the back-ground. If SO_LINGER is enabled with a zero timeout, Socket Close returns immediately and the connection is reset.
SO_KEEPALIVE	00000008h	UINT32 (l_keepalive)	Enables/disables keep alive probes on a socket. This option is only valid for TCP sockets. l_keepalive 0:Keep alive OFF (default) Other:Keep alive ON Keep alive can be used to detect if the host is still active, and if it is not, close down the connection. If keep alive is enabled a keep alive probe will be sent to the host after 2 hours with no data being sent or received on a connection. This packet is designed to provoke an ACK response from the host. If no ACK is received another 8 keep alive probes will be sent with 75 seconds interval, and if none of them is ACKed the connection will be reset.
SO_REUSEADDR	00000004h	UINT32 (l_reuseaddr)	Enables/disables reuse address option on a socket. This option is only valid for TCP sockets. l_reuseaddr 0:Reuse address OFF (default) Other:Reuse address ON When reuse address option is enabled it is possible to reuse a TCP port even if the port is busy in TIME_WAIT state. If the port is busy in other states an error will still be generated. This can be useful for a server implementation that is shut down and directly restarted while sockets are still active on its port.

IP_MULTICAST_TTL	0000000Ah	UINT8 (b_ttl)	Sets the TTL value for multicast packets. This option is only valid for UDP sockets. b_ttl 1-255 (Default 1) The TTL value is part of the IP packet header and specifies the number of routers a packet is allowed to pass before it shall be deleted. The default value of 1 prevents multicast packets from being forwarded beyond the local network.
IP_MULTICAST-_LOOP	0000000Bh	UINT8 (b_multicastloop)	Enables/disables multicast packet loopback. This option is only valid for UDP sockets. l_reuseaddr 0:Multicast loopback OFF 1:Multicast loopback ON (default)
IP_ADD_MEMBER-SHIP	0000000Ch	Struct of: UINT32 (l_multiaddr) UINT32 (l_interface)	Adds membership to a multicast group. This option is only valid for UDP sockets. l_multiaddr IP address of multicast group to join. l_interface IP address of interface to join (own IP address) By joining a multicast group the local multicast router will be notified about the multicast membership (using IGMP) and the local interface network driver will enable reception of multicast datagrams destined for this multicast address.
IP_DROP_MEMBER-SHIP	0000000Dh	Struct of: UINT32 (l_multiaddr) UINT32 (l_interface)	Drops membership from a multicast group. This option is only valid for UDP sockets. l_multiaddr IP address of multicast group to leave. l_interface IP address of interface (own IP address) By leaving a multicast group the local multicast router will be notified and the local interface network driver will disable reception of multicast datagrams destined for this multicast address.
TCP_NODELAY	00002002h	UINT32 (l_nodelay)	Enables/disables the Nagle algorithm on a socket. This option is only valid on TCP sockets. l_nodelay 0:Nagle algorithm ON (default) l_nodelay Other:Nagle algorithm OFF For some applications, especially request/response applications, the performance over a TCP connection may be poor due to the interaction between the Nagle algorithm and the delayed acknowledgment functionality. Then the TCP_NODELAY option can be used to disable the Nagle algorithm to increase performance. For more information about Nagle algorithm see RFC 896.

9.4 Mailbox File System Interface

The filesystem is available to the application through the mailbox interface. Commands in this category:

Mailbox Command	Description	Page
Open File (FILE_OPEN)	Open a file for reading, writing, or appending.	95
Close File (FILE_CLOSE)	Close a file previously opened using FILE_OPEN	96
Read File (FILE_READ)	Read data from a file	97
Write File (FILE_WRITE)	Write data to a file.	98
Delete File (FILE_DELETE)	Delete a file	99
Move File (FILE_MOVE)	Moves a file	100
Rename File (FILE_RENAME)	Rename a file	101
Copy File (FILE_COPY)	Copy a file	102
Create Directory (DIR_CREATE)	Create a new directory	103
Delete Directory (DIR_DELETE)	Delete an empty directory	104
Open Directory (DIR_OPEN)	Open a directory	105
Read Directory (DIR_READ)	Read contents of a directory previously opened using DIR_OPEN	106
Close Directory (DIR_CLOSE)	Close a directory previously opened using DIR_OPEN	108
Format File System (FORMAT_FS)	Formats the filesystem.	109
File system Checksum (CRC_FS)	Calculate and return the CRC for the File System	110

9.4.1 Open File (FILE_OPEN)

Description

This command opens a file for reading, writing, or appending. If the specified file does not exist, it will be created.

Initiated by	Application
Command no.	0060h
Extended Header	Mode, Filesize & Fault information
Message data	Name and path to the file to open (NULL terminated)
Response data	File Handle

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0060h	0060h	*FILE_OPEN*
Data size	(size)	0004h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Mode	Filesize (high)	
Extended word 2	-	Filesize (low)	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Path + filename (String, null-terminated)	File Handle (high)	Response data word 1
		File Handle (low)	Response data word 2

- **Mode**

Value	Mode
0000h	Open a file in read mode
0001h	Open a file in write mode. If the specified file does not exist, it will be created. If the specified file already exists, it will be overwritten.
0002h	Open a file in append mode. If the specified file does not exist, it will be created. If the specified file exists, any data written to the file will be appended at end-of-file.

- **Filesize**

  Current filesize (if applicable).

- **File Handle**

  Unique identifier which must be used on all further operations associated with the file.

9.4.2 Close File (FILE_CLOSE)

Description

This command closes a file previously opened using FILE_OPEN.

Initiated by	Application
Command no.	0061h
Extended Header	File Handle, Filesize & Fault information.
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0061h	0061h	*FILE_CLOSE*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	File Handle (high)	File Handle (high)	
Extended word 2	File Handle (low)	File Handle (low)	
Extended word 3	-	Filesize (high)	
Extended word 4	-	Filesize (low)	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	

- **File Handle**

  Handle of the file to close. See also "File Handle" on page 95.

- **Filesize**

  Size of the file.

9.4.3 Read File (FILE_READ)

Description

This command reads data from a file previously opened in read mode using FILE_OPEN.

Initiated by	Application
Command no.	0062h
Extended Header	File Handle, no. of bytes to read & Fault information
Message data	-
Response data	The read data is returned in the response data field.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0062h	0062h	*FILE_READ*
Data size	0000h	(size)	*Bytes read*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	File Handle (high)	File Handle (high)	
Extended word 2	File Handle (low)	File Handle (low)	
Extended word 3	No. of bytes	No. of bytes	*Maximum 256 bytes.*
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Data	Response data

- **File Handle**

  File handle of the file to read data from. See also "File Handle" on page 95.

- **No. of bytes**

  Number of bytes to read minus 1 (i.e. a value of 42 will read 43 bytes).

- **Data**

  The actual data read from the file (if applicable).

9.4.4 Write File (FILE_WRITE)

Description

This mailbox command writes data to a file previously opened in write or append mode using FILE_OPEN.

Initiated by	Application
Command no.	0063h
Extended Header	File Handle & Fault information
Message data	Data to write
Response data	A 'Data size' value of 0 (zero) indicates that the command was unsuccessful, possibly due to a faulty handle, or that the module has run out of storage.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0063h	0063h	*FILE_WRITE*
Data size	(number of bytes to write)	(number of written bytes)	*Max. 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	File Handle (high)	File Handle (high)	
Extended word 2	File Handle (low)	File Handle (low)	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Data	Data	

- **File Handle**

  File handle of the file to write data to. See also "File Handle" on page 95.

- **No. of bytes**

  Number of bytes to write minus 1 (i.e. a value of 42 will read 43 bytes).

- **Data**

  The actual data that shall be written.

9.4.5 Delete File (FILE_DELETE)

Description

This mailbox command deletes a file from the file system.

Initiated by	Application
Command no.	0064h
Extended Header	Fault information
Message data	Name and path to the file to delete (NULL terminated)
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0064h	0064h	*FILE_DELETE*
Data size	(size)	(size)	*Maximum 256 bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Path + filename (String, null-terminated)	Path + filename (String, null-terminated)	

9.4.6 Move File (FILE_MOVE)

Description

This command renames a file in the filesystem.

Note: Although the filesystem supports path lengths of up to 256 characters, the total length of the source and destination paths summed together must be less than 256 characters when using this command due to limitations in the mailbox command structure.

Initiated by	Application
Command no.	0065h
Extended Header	Fault information
Message data	Name + Path of source and destination, both NULL terminated
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0065h	0065h	*FILE_MOVE*
Data size	(size)	(size)	*Size of path strings*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Source: Path + filename (String, null-terminated)	Source: Path + filename (String, null-terminated)	
	Destination: Path + filename (String, null-terminated)	Destination: Path + filename (String, null-terminated)	

9.4.7 Rename File (FILE_RENAME)

Description

This command renames a file in the filesystem.

Note: Although the filesystem supports path lengths of up to 256 characters, the total length of the two pathnames summed together must be less than 256 characters when using this command due to limitations in the mailbox command structure.

Initiated by	Application
Command no.	0066h
Extended Header	Fault information
Message data	Name + Path of source and destination, both NULL terminated
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0066h	0066h	*FILE_RENAME*
Data size	(size)	(size)	*Size of path strings*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Old: Path + filename (String, null-terminated)	Old: Path + filename (String, null-terminated)	
	New: Path + filename (String, null-terminated)	New: Path + filename (String, null-terminated)	

9.4.8 Copy File (FILE_COPY)

Description

This command copies a file in the filesystem to a specified location.

Note: Although the filesystem supports path lengths of up to 256 characters, the total length of the source and destination paths summed together must be less than 256 characters when using this command due to limitations in the mailbox command structure.

Initiated by	Application
Command no.	0067h
Extended Header	Fault information
Message data	Name + Path of source and destination, both NULL terminated
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0067h	0067h	*FILE_COPY*
Data size	(size)	(size)	*Size of path strings*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Source: Path + filename (String, null-terminated)	Source: Path + filename (String, null-terminated)	
	Destination: Path + filename (String, null-terminated)	Destination: Path + filename (String, null-terminated)	

9.4.9 Create Directory (DIR_CREATE)

Description

This command creates a directory in the file system.

Initiated by	Application
Command no.	0068h
Extended Header	Fault information
Message data	Path and name of the new directory, null terminated.
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0068h	0068h	*DIR_CREATE*
Data size	(size)	(size)	*Size of path string*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Path + name (String, null-terminated)	Path + name (String, null-terminated)	

9.4.10 Delete Directory (DIR_DELETE)

Description

This command deletes an empty directory from the file system.

Initiated by	Application
Command no.	0069h
Extended Header	-
Message data	Path and name of the directory, null terminated.
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0069h	0069h	*DIR_DELETE*
Data size	(size)	(size)	*Size of path string*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Path + name (String, null-terminated)	Path + name (String, null-terminated)	

9.4.11 Open Directory (DIR_OPEN)

Description

This command opens a directory and returns a descriptor that should be used on all further operations on the directory.

See also "Reading the Contents of a Directory" on page 107.

Initiated by	Application
Command no.	006Ah
Extended Header	-
Message data	Path and name of the directory, null terminated.
Response data	Directory handle & Fault information

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	006Ah	006Ah	*DIR_OPEN*
Data size	(size)	0004h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data	Path + name (String, null-terminated)	Directory Handle (high)	Response data word 1
		Directory Handle (low)	Response data word 2

- **Directory Handle**

  Unique identifier which must be used on all further operations associated with the directory.

9.4.12 Read Directory (DIR_READ)

Description

This command reads the contents of a directory previously opened using DIR_OPEN. This must be repeated until the response to the command is empty (i.e. until the response data size equals zero).

See also "Reading the Contents of a Directory" on page 107.

Initiated by	Application
Command no.	006Bh
Extended Header	Directory Handle & Fault information
Message data	-
Response data	Details about one object in the directory.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	006Bh	006Bh	*DIR_READ*
Data size	0000h	(size)	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Directory Handle (high)	Directory Handle (high)	*(See DIR_OPEN)*
Extended word 2	Directory Handle (low)	Directory Handle (low)	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Object Size (long)	Response data byte 1 Response data byte 2 Response data byte 3 Response data byte 4
		Object Flags	Response data byte 5
		Object Name (string, null-terminated)	Response data...

- **Directory Handle**

  Unique identifier which must be used on all further operations associated with the directory.

- **Object Size**

  Size of object (i.e. filesize).

- **Object Flags**

  Various flags specifying the nature of the object:

b7	b6	b5	b4	b3	b2	b1	b0
(reserved)				SYS	H	RO	DIR

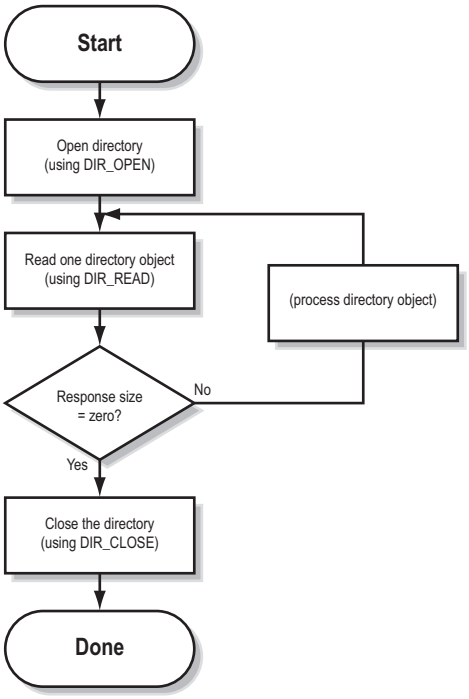
Bit	Description	
DIR	Directory flag	0: Object is a file 1: Object is a directory
RO	Read only	0: Object can be read or written 1: Object is read-only
H	Hidden	0: Object is visible 1: Object is hidden
SYS	System	0: User object 1: System object

- **Object Name**

  Name of object, null-terminated (e.g. filename or directory name).

Reading the Contents of a Directory

The following flowchart illustrates the process of reading the contents of a directory:



9.4.13 Close Directory (DIR_CLOSE)

Description

This command closes a directory previously opened using DIR_OPEN.

See also "Reading the Contents of a Directory" on page 107.

Initiated by	Application
Command no.	006Ch
Extended Header	Directory Handle & Fault information
Message data	-
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	006Ch	006Ch	*DIR_CLOSE*
Data size	0000h	0000h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Directory Handle (high)	Directory Handle (high)	*(See DIR_OPEN)*
Extended word 2	Directory Handle (low)	Directory Handle (low)	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	

9.4.14 Format File System (FORMAT_FS)

Description

This command formats the file system.

Initiated by	Application
Command no.	006Dh
Extended Header	-
Message data	Which disc to format
Response data	The response indicates if the command was accepted

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	006Dh	006Dh	*FORMAT_FS*
Data size	0000h[a] or 0001h	0000h	*No additional data*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message databyte 1	(optional) Disc to format: 0: Root disc 1: cfg disc 2: ram disc		

a. If Data size is set to 0000h, all discs will be formatted

9.4.15 File system Checksum (CRC_FS)

Description

This command calculates the checksum for the the file system disc area.

Initiated by	Application
Command no.	006Eh
Extended Header	-
Message data	Which disc to calculate CRC for
Response data	The response indicates if the command was accepted

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	006Eh	006Eh	*CRC_FS*
Data size	0001h	0004h	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data	Disc ID: 0: Root disc 1: cfg disc 2: ram disc	DISC CRC	

9.5 EtherNet/IP Specific Commands

Commands in this category:

Mailbox Command	Description	Page
Set Product Info (SET_PRODUCT_INFO)	Used for vendor customization	112
Get Product Info (GET_PRODUCT_INFO)	Used for vendor customization	113
Parameter Data Input Mapping (PARAMETER_IN_MAP)	Maps parameter input data into the parameter input object	114
Parameter Data Out Area Mapping (PARAMETER_OUT_MAP)	Maps parameter output data into the parameter input object	116
Send UCMM (SEND_UCMM)	Sends an unconnected message to another EthernNet/IP module	118
UCMM Request (UCMM_REQUEST)	Generated by the module when an UCMM request is addressed to a user-registered object	120
Register Class (REGISTER_CLASS)	Registers an EtherNet/IP object in the message router object	121
Deregister Class (DEREGISTER_CLASS)	Deregisters a class that is already registered within the message router	122
Register Port (REGISTER_PORT)	Registers a port in the port object	123
Route Unconnected Send (ROUTE_REQUEST)	Generated by the module when it receives an unconnected send message.	126
Enable Exact IO Match (EXACT_IO_MATCH)	Only accept IO connection requests with the exact same sizes as the module has been initiated to.	128
Get Reset Parameter (GET_ID_RESET_PARAM)	Get reset parameter	129
Change Ethernet Port Number (CHANGE_ETH_PORT_NO)	Change the port number reported by the Port Object for the Ethernet port	130
Set Scanner Mode (SET_SCANNER_MODE)	Set scanner to Run or Idle	131
Create Connection Target Area (CREATE_CON_TARGET)	Use part of the Anybus IO area as connection target and create instances in the Assembly Object	132
Set UCMM Timeout (SET_UCMM_TIMEOUT)	Set UCMM timeout	133
Get UCMM Timeout (GET_UCMM_TIMEOUT)	Get UCMM timeout for display on e.g. a web page	133
Set Minimum Class1 O->T Timeout (SET_MIN_CLASS1_OT_TIMEOUT)	Set minimum timeout for Class1 O->T Connections	135

9.5.1 Set Product Info (SET_PRODUCT_INFO)

Description

This command may be used to customize the settings in the Identity Object. Note that the .EDS-file must be adjusted accordingly.

Note: This command may only be issued during initialisation.

Command initiator	Application
Command number	0082h
Extended Header data	-
Message data	Identity object information
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0082h	0082h	*SET_PRODUCT_INFO*
Data size	(data size)	(data size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Vendor ID	Vendor ID	
Message data word 2	Device Type	Device Type	
Message data word 3	Product Code	Product Code	
Message data word 4	Major & Minor Revision	Major & Minor Revision	
Message data word 5	Serial Number (high)	Serial Number (high)	
Message data word 6	Serial Number (low)	Serial Number (low)	
Message data...	Product Name (string)	Product Name (string)	

- **Vendor ID, Device Type & Product Code**

  Vendor & product information.

- **Major & Minor Revision**

  Revision no. (major revision = high byte, minor revision = low byte)

- **Serial Number**

  Product serial number

- **Product Name**

  1st byte holds the length of the name in bytes, followed by the actual name as pure ASCII.

9.5.2 Get Product Info (GET_PRODUCT_INFO)

Description

This command returns information from the Identity Object.

Command initiator	Application
Command number	0083h
Extended Header data	-
Message data	-
Response message	Identity object information

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0083h	0083h	*GET_PRODUCT_INFO*
Data size	0000h	(data size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		Vendor ID	*Response data word 1*
		Device Type	*Response data word 2*
		Product Code	*Response data word 3*
		Major & Minor Revision	*Response data word 4*
		Serial Number (high)	*Response data word 5*
		Serial Number (low)	*Response data word 6*
		Product Name (string)	*Response data....*

- **Vendor ID, Device Type & Product Code**
  Vendor & product information.
- **Major & Minor Revision**
  Revision no. (major revision = high byte, minor revision = low byte)
- **Serial Number**
  Product serial number
- **Product Name**
  1st byte holds the length of the name in bytes, followed by the actual name as pure ASCII.

9.5.3 Parameter Data Input Mapping (PARAMETER_IN_MAP)

Description

This mailbox command is used to set up blocks of data in the input data area to be used with the Class B0h "Parameter data input mapping object" (See "Parameter Data Input Mapping Object, Class B0h" on page 44). This way, a 'Get_Atrribute_Single' command from the fieldbus can return a specified block of data from the input parameter area. It is possible to map up to 50 attributes.

Attributes are mapped beginning with attribute 1, followed by attributes 2 through 50. Offset in the input parameter area and number of bytes to map is specified for each attribute.

If zero length is specified, the attribute will not be mapped. This way, it is for example possible to map only attributes 1 and 10 by specifying zero length for attributes 2 through 9. It is only necessary to include information in the telegram up to the last used attributes; the remaining attributes will not be mapped.

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

To access the parameters from EtherNet/IP use Class attribute B0h, Instance attribute 01h and Attribute ID 01h through 32h. (See "Parameter Data Input Mapping Object, Class B0h" on page 44)

Note: This command may only be issued during module initialisation, after ANYBUS_INIT.

Command initiator	Application
Command number	0084h
Extended Header data	Fault information
Message data	Offset and length of the attributes to map
Response message	The response holds a copy of the command data.

Command and response layout (example when only setting attribute 1-5)

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0084h	0084h	*PARAMETER_IN_MAP*
Data size	0014h	0014h	*20 bytes of data (10 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data word 1	Offset	Offset	*Offset*
Message data word 2	Length	Length	*Number of bytes to map*
Message data word 3	Offset	Offset	*Attribute 2*
Message data word 4	Length	Length	
Message data word 5	Offset	Offset	*Attribute 3*
Message data word 6	Length	Length	
Message data word 7	Offset	Offset	*Attribute 4*
Message data word 8	Length	Length	
Message data word 9	Offset	Offset	*Attribute 5*
Message data word 10	Length	Length	

9.5.4 Parameter Data Out Area Mapping (PARAMETER_OUT_MAP)

Description

This mailbox command is used to set up blocks of data in the output data area to be used with the Class B1h "Parameter data output mapping object" (See "Parameter Data Output Mapping Object, Class B1h" on page 45). This way, a Get_Attribute_Single or Set_Attribute_Single command from the fieldbus can return/write a specified block of data from/to the output parameter area. It is possible to map up to 50 attributes.

Attributes are mapped beginning with attribute 1, followed by attributes 2 through 50. Offset in the input parameter area and number of bytes to map is specified for each attribute.

If zero length is specified, the attribute will not be mapped. This way, it is for example possible to map only attributes 1 and 10 by specifying zero length for attributes 2 through 9. It is only necessary to include information in the telegram up to the last used attributes; the remaining attributes will not be mapped.

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

To access the parameters from EtherNet/IP use Class attribute B1h, Instance attribute 01h and Attribute ID no. (See "Parameter Data Output Mapping Object, Class B1h" on page 45)

Note: This command may only be issued during module initialisation, after ANYBUS_INIT.

Command initiator	Application
Command number	0085h
Extended Header data	-
Message data	Offset and length of the attributes to map
Response message	The response holds a copy of the command data.

Command and response layout (example when only setting attribute 1-5)

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0085h	0085h	*PARAMETER_OUT_MAP*
Data size	0014h	0014h	*20 bytes of data (10 words)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data word 1	Offset	Offset	*Offset*
Message data word 2	Length	Length	*Number of bytes to map*
Message data word 3	Offset	Offset	*Attribute 2*
Message data word 4	Length	Length	
Message data word 5	Offset	Offset	*Attribute 3*
Message data word 6	Length	Length	
Message data word 7	Offset	Offset	*Attribute 4*
Message data word 8	Length	Length	
Message data word 9	Offset	Offset	*Attribute 5*
Message data word 10	Length	Length	

9.5.5 Send UCMM (SEND_UCMM)

Description

This command may be used to send an explicit unconnected message directly to a node on the network.(For more information, consult the EtherNet/IP Specification volume 1, section 2-4). This command will not respond until the response from the remote host is received or a timeout has occurred.

Command initiator	Application
Command number	008Ah
Extended Header data	Destination IP address
Message data	Unconnected message[a]
Response message	(service dependant)

a. For more information, consult the EtherNet/IP Specification volume 1, section 2-4

Command and response layout

	Command	Expected response	
Message ID	ID	ID	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Ah	008Ah	*SEND_UCMM*
Data size	(data size)	(data size)	*Size of data*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Destination IP high word	Destination IP high word	*Destination IP high word*
Extended word 2	Destination IP low word	Destination IP low word	*Destination IP low word*
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault Information	
Message data byte 1	Service Request	(Service Dependant)	
Message data byte 2	Request Path Size (in words)		
Message data byte 3	Padded EPATH[a]		
...			
...	(optional service data)		
Message data byte n			

a.EPATH is in little-endian format. See EtherNet/IP Specification, Appendix C - 'Data Management'

The following example (see next page) uses the SEND_UCMM command to retrieve data from a node on the network. The remote node is an Allen Bradley ControlLogix5000 with a 1756-ENBT/A Ethernet/IP module. It has VendorID 0001h, Product Type 000C, Product Code 003Ah, Version 1.33 and Serial Number 00121E63h. The request that is sent to the remote node is 'Get_Attribute_All' (0x01) to Class 0x01 and Instance 0x01. This is the identity object, see 5-2.2 in the EtherNet/IP specification for more information about the response.

Send UCMM Example

	Command	Expected response	
Message ID	ID	ID	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Ah	008Ah	*SEND_UCMM*
Data size	0006h	001Eh	*Size of data*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	0A0Ah	0A0Ah	*Destination IP (high word)*
Extended word 2	0E50h	0E50h	*Destination IP (low word)*
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault Information	
Data byte 1 (Service Request)	01h (Get_Attribute_All)	81h	*Get_Attiribute_All reply*
Data byte 2 (Request Path Length)	02h (2 words)	00h	*(reserved)*
Data byte 3 (Segment Type)	20h[a]	00h	*General Status*
Data byte 4 (Segment Data)	01h (Class #1)	00h	*Additional Status*
Data byte 5 (Segment Type)	24h[b]	01h	*Vendor ID (LSB)*
Data byte 6 (Segment Data)	01h (Instance #1)	00h	*Vendor ID (MSB)*
		0Ch	*Product Type (LSB)*
		00h	*Product Type (MSB)*
		3Ah	*Product Code (LSB)*
		00h	*Product Code (MSB)*
		01h	*Version (Major)*
		21h	*Version (Minor)*
		30h	*Status (LSB)*
		00h	*Status (MSB)*
		63h	*Serial no. (LSB)*
		1Eh	*Serial no.*
		12h	*Serial no.*
		00h	*Serial no. (MSB)*
		0Bh (11)	*Product Name Length*
		31h (‘1’)	*Product Name Char #1*
		37h (‘7’)	*Product Name Char #2*
		35h (‘5’)	*Product Name Char #3*
		36h (‘6’)	*Product Name Char #4*
		2Dh (‘-’)	*Product Name Char #5*
		45h (‘E’)	*Product Name Char #6*
		4Eh (‘N’)	*Product Name Char #7*
		42h (‘B’)	*Product Name Char #8*
		54h (T1’)	*Product Name Char #9*
		2Fh (‘/’)	*Product Name Char #10*
		41h (‘A’)	*Product Name Char #11*

a.Segment Type= Logical Segment
Logical Type= Class ID
Logical Format= 8bit Logical Address

b.Segment Type= Logical Segment
Logical Type= Instance ID
Logical Format= 8bit Logical Address.

9.5.6 UCMM Request (UCMM_REQUEST)

Description

This message is used when the application has registered an EtherNet/IP class (See "Register Class (REGISTER_CLASS)" on page 121) in the module, and an explicit message request has been generated to this class from a node in the EtherNet/IP network.

The format of the message is the message router / request format. (See EtherNet/IP Specification volume 1 section 2-4). The application will have to process the message, and respond to the module with the data necessary to generate a response on the explicit message request for the object.

Command initiator	Anybus
Command number	008Dh
Extended Header data	-
Message data	Explicit message data
Response message	Requested data or error code

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Dh	008Dh	*UCMM_REQUEST*
Data size	(size)	(size)	*Data size*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte 1	Service Request	Reply Service	
Message data byte 2	Request Path Size (in words)	(reserved, ignore)	
Message data byte 3	Padded EPATH[a]	General Status	
...		Size of Additional Status	
...	(optional service data)	Additional Status	
Message data byte n		Response_data	

a.Request path according to the EtherNet/IP Specification, Appendix C - 'Data Management'

9.5.7 Register Class (REGISTER_CLASS)

Description

This mailbox command makes it possible for the application to register objects inside the message router object. If there is a node on the network that sends an explicit message request to the module, address to the registered class, the explicit message will generate an explicit message request telegram, which will be sent from the module to the application. See "UCMM Request (UCMM_REQUEST)" on page 120.

Command initiator	Application
Command number	008Bh
Extended Header data	Fault information
Message data	Class ID
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Bh	008Bh	*REGISTER_CLASS*
Data size	0002h	0002h	*2 bytes of data (1 word)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data byte 1	Class ID (high byte)	Class ID (high byte)	
Message data byte 2	Class ID (low byte)	Class ID (low byte)	

- **Class ID**

  ID of the class to be registered in the Anybus module.

9.5.8 Deregister Class (DEREGISTER_CLASS)

Description

This mailbox command makes it possible for the application to deregister objects inside the message router object.

The following classes cannot be deregistered with this command;

- Class 02h - Message Router
- Class 04h - Assembly Object

Command initiator	Application
Command number	008Eh
Extended Header data	Fault information
Message data	Class ID
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Eh	008Eh	*DEREGISTER_CLASS*
Data size	0002h	0002h	*2 bytes of data (1 word)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
Message data byte 1	Class ID (high byte)	Class ID (high byte)	
Message data byte 2	Class ID (low byte)	Class ID (low byte)	

- **Class ID**

  ID of the class to be deregistered.

9.5.9 Register Port (REGISTER_PORT)

Description

This command may be used to register a port in the Port Object (See EtherNet/IP specification vol. 1 chapters 3 - 7). This must be done for each port in the application if routing has been enabled. The message shall contain the instance attributes 1, 2, 3, 4 and 7, in that order. The class attributes will be updated automatically after each received mailbox command. Port 2 is reserved for the anybus module, and it is not possible to register the same port twice.

Note: This command can only be sent during module initialization.

Command initiator	Application
Command number	0090h
Extended Header data	Fault information
Message data	Instance attributes 1,2,3,4 and 7.
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0090h	0090h	*REGISTER_PORT*
Data size	(size)	(size)	*Size of data in bytes*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault Information	
Data byte 1	Port Type (high)	Port Type (high)	
Data byte 2	Port Type (low)	Port Type (low)	
Data byte 3	Port Number (high)	Port Number (high)	
Data byte 4	Port Number (low)	Port Number (low)	
Data byte 5	Port Object Size (high)	Port Object Size (high)	
Data byte 6	Port Object Size (low)	Port Object Size (low)	
Data byte 7...n	Port Object EPATH	Port Object EPATH	
Data byte n	(Padded)[a]	(Padded)[a]	
Data byte n+1	Port Name Length	Port Name Length	
Data byte n+2	Port Name Char #1	Port Name Char #1	
Data byte n+3	Port Name Char #2	Port Name Char #2	
...	...	...	
Data byte z	Port Name Char #n	Port Name Char #n	
Data byte z+1	Node address	Node address	
Data byte z+2	Node address	Node address	

a.See EtherNet/IP Specification, Appendix C - 'Data Management'

Register Port Example

The following example registers a ControlNet redundant port (3) with port number 3. The port object points to class F0h (ControlNet object) instance 01h. The name of the port is "ControlNet", and the node address is 8 on port 3.

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0090h	0090h	*REGISTER_PORT*
Data size	0017h	0017h	*23 bytes of data*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	

Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault Information	
Data byte 1	00h	00h	*Port Type (high)*
Data byte 2	03h	03h	*Port Type (low)*
Data byte 3	00h	00h	*Port Number (high)*
Data byte 4	03h	03h	*Port Number (low)*
Data byte 5	00h	00h	*Port Object Size (high)*
Data byte 6	02h	02h	*Port Object Size (low)*
Data byte 7	20h	20h	*Port Object EPATH*
Data byte 8	F0h	F0h	*Port Object EPATH*
Data byte 9	24h	24h	*Port Object EPATH*
Data byte 10	01h	01h	*Port Object EPATH*
Data byte 11	0Ah	0Ah	*Port Name Length*
Data byte 12	43h	43h	*Port name: “C”*
Data byte 13	6Fh	6Fh	*Port name: “o”*
Data byte 14	6Eh	6Eh	*Port name: “n”*
Data byte 15	74h	74h	*Port name: “t”*
Data byte 16	72h	72h	*Port name: “r”*
Data byte 17	6Fh	6Fh	*Port name: “o”*
Data byte 18	6Ch	6Ch	*Port name: “l”*
Data byte 19	4Eh	4Eh	*Port name: “N”*
Data byte 20	65h	65h	*Port name: “e”*
Data byte 21	74h	74h	*Port name: “t”*
Data byte 22	03h	03h	*Port to leave node = 3*
Data byte 23	08h	08h	*On ControlNet = 8*

Continued on next page...

The [Port] section in the .EDS file shall look like this to suit the example above:

```
[Port]
                Port1 =
                          TCP,$ Port type
                          “TCP/IP”,$ Port name
                          “20 F5 24 01”,$ Path to object supporting this
port
                          2;$ Port number

                Port2 =
                          ControlNet,$ Port type
                          “ControlNet”,$ Port name
                          “20 F0 24 01”,$ Path to object supporting this
port
                          3;$ Port number
```

9.5.10 Route Unconnected Send (ROUTE_REQUEST)

Description

This mailbox message is generated by the module when it receives a valid unconnected send message, i.e. a message addressed to a port registered by the application. The message data contains the whole unconnected send message (See EtherNet/IP spec. Vol. 1 3-5.5.4). The response from the application must contain either a successful or unsuccessful unconnected send response (See EtherNet/IP spec. Vol. 1 3-5.5.4).

If 16 or more message requests are waiting to be processed by the application, the module will respond with a "No resource" error code for all new requests until there are less than 16 unprocessed requests.

Command initiator	Anybus
Command number	008Fh
Extended Header data	-
Message data	The unconnected send message received by the module
Response message	Unconnected send response message.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	008Fh	008Fh	*ROUTE_REQUEST*
Data size	(size)	(size)	*Datasize*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte 1	Transaction ID (low)		
Message data byte 2	Transaction ID (high)	*Successful response:*	
...	Priority / Time tick	Transaction ID (low)	Response data byte 1
	Time-out ticks	Transaction ID (high)	Response data byte 2
	Msg. req. size (low)	00h (General Status)	...
	Msg. req. size (high)	00h (Reserved)	
Message Request	Service Code	Service Response Data	...
	Request Path Size		Response data byte n
	Req. Path (Padded EPATH)		
	Request Data	*Unsuccessful response:*	
		Transaction ID (low)	Response data byte 1
		Transaction ID (high)	Response data byte 2
	00h (Pad) (Only if Msg.req.size is odd)	General Status	...
		Size of additional status	
	Route Path Size	Additional status	
...	00h (reserved)		...
Message databyte n	Route Path	Remaining path size	Response data byte n

9.5.11 Enable Exact IO Match (EXACT_IO_MATCH)

Description

If this mailbox command is sent, the module will only accept IO connection requests with the exact same IO sizes as those that have been configured for the module.

Note 1: This command can only be sent during module initialization.

Note 2: This command has to be sent if the application is to pass the conformance test.

Command initiator	Application
Command number	0094h
Extended Header data	-
Message data	-
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0094h	0094h	*EXACT_IO_MATCH*
Data size	0000h	0000h	*0 bytes of data*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	

9.5.12 Get Reset Parameter (GET_ID_RESET_PARAM)

Description

Using this command, it is possible to determine which type of reset that was received via EtherNet/IP

Command initiator	Application
Command number	0095h
Extended Header data	-
Message data	-
Response message	Reset Type

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0095h	0095h	*GET_ID_RESET_PARAM*
Data size	0000h	0001h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		Reset Type	Response Databyte

- **Reset Type**

  00h: Power on reset

  01h: Out of box configuration

9.5.13 Change Ethernet Port Number (CHANGE_ETH_PORT_NO)

Description

This command may be used to change the port number reported in the Port Object.

Command initiator	Application
Command number	0096h
Extended Header data	-
Message data	Port no.
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0096h	0096h	*CHANGE_ETH_PORT_NO*
Data size	0001h	0001h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte 1	Port no.	Port no.	

- **Port no.**

  Port number, range 1...14.

9.5.14 Set Scanner Mode (SET_SCANNER_MODE)

Description

This command changes the scanner mode to Run or Idle.

See also...

- "SetScannerState" on page 27
- "Identity Object, Class 01h" on page 37

Command initiator	Application
Command number	0098h
Extended Header data	-
Message data	Port no.
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0098h	0098h	*SET_SCANNER_MODE*
Data size	0002h	0002h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte 1	Scanner Mode	Scanner Mode	
Message data byte 2	Network Change	Network Change	

- **Scanner Mode**

  00h: Idle
  01h: Run

- **Network Change**

  00h: Allowed (Scanner mode may also be changed via SSI or EtherNet/IP)
  01h: Disallowed (Scanner mode may not be changed via SSI or EtherNet/IP)

9.5.15 Create Connection Target Area (CREATE_CON_TARGET)

Description

This command may be used to set up a part of the IO area as a target for an EtherNet/IP connection. When issued, instances 100 and 150 will be created in the Assembly Object accordingly.

Note: This command may only be issued during initialisation, after ANYBUS_INIT.

Command initiator	Application
Command number	0097h
Extended Header data	-
Message data	Number of words (Produce & Consume)
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0097h	0097h	*CREATE_CON_TARGET*
Data size	0004h	0004h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	No. of words (Produce)	No. of words (Produce)	
Message data word 2	No. of words (Consume)	No. of words (Consume)	

- **No. of words (Produce)**

  Number of words to use for Produce Area (Instance 100)

- **No. of words (Consume)**

  Number of words to use for Consume Area (Instance 150)

9.5.16 Set UCMM Timeout (SET_UCMM_TIMEOUT)

Description

This command is used to set the UCMM timeout in milliseconds. Valid range is 1000 - 30000.

NOTE: The module has to be restarted for a change to the timeout value from this mailbox to take effect.

Command initiator	Application
Command number	0099h
Extended Header data	-
Message data	UCMM timeout in milliseconds. Valid values: 1000 - 30000
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0099h	0099h	*SET_UCMM_TIMEOUT*
Data size	0002h	0002h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data byte 1	UCMM timeout (high byte)	UCMM timeout (high byte)	UCMM timeout
Message data byte 2	UCMM timeout (low byte)	UCMM timeout (low byte)	

9.5.17 Get UCMM Timeout (GET_UCMM_TIMEOUT)

Description

This command reads the currently used UCMM timeout (ms) and makes it available for display on a web page.

Command initiator	Application
Command number	009Ah
Extended Header data	-
Message data	UCMM timeout in milliseconds
Response message	Contains currently used UCMM timeout.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	009Ah	009Ah	*GET_UCMM_TIMEOUT*
Data size	0000h	0002h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1		UCMM timeout (high byte)	UCMM timeout
Message data word 2		UCMM timeout (low byte)	

9.5.18 Set Minimum Class1 O->T Timeout (SET_MIN_CLASS1_OT_TIMEOUT)

Description

This command is used to set the minimum timeout for the Class1 connections in the Originator to Target direction (connections originated by the module). If this mailbox is used the Connection Timeout Multiplier attribute (conn_timeout) in the Connection Configuration object will be altered to conform to the value set in this command.

Valid range is 8 - 1024.

See also...

- "Connection Configuration Object, Class F3h" on page 46

Note: This command can only be sent during module initialization.

Command initiator	Application
Command number	00B0h
Extended Header data	-
Message data	Minimum timeout in milliseconds. Valid range is: 8 - 1024.
Response message	The response holds a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	00B0h	00B0h	*SET_MIN_CLASS1_OT_TIMEOUT*
Data size	0002h	0002h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data word 1	Timeout (high byte)	Timeout (high byte)	Minimum timeout
Message data word 2	Timeout (low byte)	Timeout (low byte)	

9.6 Other Commands

Commands in this category:

Mailbox Command	Description	Page
Get DIP Switch GET_DIP_SWITCH	Returns the setting of the onboard DIP switch	137
DNS Request (DNS_REQUEST)	Asks the configured DNS server for the IP address of a specified host	138
Send Email (SEND_EMAIL)	Sends an email message to a specified recipient	139
Request SSI Data (REQUEST_SSI_DATA)	Requests SSI data from the application (issued by the Anybus module)	141
Write SSI Data (WRITE_SSI_DATA)	Writes SSI data to the application (issued by the Anybus module)	142

9.6.1 Get DIP Switch (GET_DIP_SWITCH)

Description

This command returns the setting of the onboard switch.

Initiated by	Application
Command no.	0012h
Extended Header	-
Message data	-
Response data	Switch value

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0012h	0012h	*GET_DIP_SWITCH*
Data size	0000h	0001h	*1 data byte*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	Fault information	
		Switch Value	Response databyte

- **Switch Value**

b7	b6	b5	b4	b3	b2	b1	b0
Switch 1	Switch 2	Switch 3	Switch 4	Switch 5	Switch 6	Switch 7	Switch 8

A set bit indicates that the switch is in ON position.

9.6.2 DNS Request (DNS_REQUEST)

Description

This command sends a request to the configured DNS server for the IP address of a specified host.

Initiated by	Application
Command no.	0030h
Extended Header	-
Message data	Host (string, null-terminated)
Response data	IP address of host, or 0.0.0.0 if not found.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0030h	0030h	*DNS_REQUEST*
Data size	(size)	0004h	
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message data	Host (string, null-terminated)	IP address (high)	Response data word 1
		IP address (low)	Response data word 2

9.6.3 Send Email (SEND_EMAIL)

Description

This command sends an email to a specified recipient. The message data is sent as several fragments, with a total maximum size of 1024 bytes. The maximum size of each fragment is 256 bytes.

Initiated by	Application
Command no.	0070h
Extended Header	Fault information
Message data	Email message specification, fragmented.
Response data	The response data is a copy of the command data.

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	0070h	0070h	*SEND_EMAIL*
Data size	(fragment size)	(fragment size)	*Max. 256 bytes / fragment*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	Fragment Type	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	SMTP Error	*(Last fragment only)*
Extended word 8	-	Fault information	
Message data	Fragment Data	Fragment Data	

- **Fragment Type**

  This value must match the sequence of the fragments as follows:

Value	Description
0000h	This is the first fragment
0001h	This is a subsequent fragment
0002h	This is the last fragment

- **SMTP Error**

  If an SMTP error occurred, the 'SMTP Error' word contains the error code from the SMTP server, see RFC 821 "Simple Mail Transfer Protocol" for more information.

- **Fragment Data**

  The different parts of the email message shall be sent in the following order:

Fragment no.	Fragment Type	Description
1st	0000h	Recipient(s), separated by semicolon (string, null-terminated)
2nd	0001h	Sender address (string, null-terminated)
3rd		Subject line (string, null-terminated)
4th		Message body
...		
...		
...		
...		
(last fragment)	0002h	

9.6.4 Request SSI Data (REQUEST_SSI_DATA)

Description

This message is issued by the Anybus module when a SSI has requested data from the application.

Example:

The following SSI...

```
<?--#exec cmd_argument='printf( "Data: %u", MbReadWord( 42 ) )'-->
```

... will cause the module to issues a REQUEST_SSI_DATA message. The value '42' will be passed to the application.

See also "printf" on page 23.

Initiated by	Anybus
Command no.	00A0h
Extended Header	SSI Identifier
Message data	SSI Data
Response data	-

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	00A0h	00A0h	*REQUEST_SSI_DATA*
Data size	0000h	(data size)	*(size of data)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	SSI Identifier	SSI Identifier	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
		SSI Data	Response Data

- **SSI Identifier**

  Identifier which can be used as desired by the application to address a specific block of data.

- **SSI Data**

  Data associated with the specified SSI Identifier.

9.6.5 Write SSI Data (WRITE_SSI_DATA)

Description

This message is issued by the Anybus module when a SSI writes data to the application.

Example:

The following SSI...

```
<?--#exec cmd_argument='scanf( "Input", "%i", MbWriteWord( 24 ) )'-->
```

... will cause the module to issues a WRITE_SSI_DATA message each time a form with an object named "Input" is sent to the web server. The value '24' will be passed to the application.

See also "scanf" on page 25.

Initiated by	Anybus
Command no.	00A0h
Extended Header	SSI Identifier
Message data	-
Response data	SSI Data

Command and response layout

	Command	Expected response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	*Fieldbus Specific Message*
Command	00A1h	00A1h	*WRITE_SSI_DATA*
Data size	(data size)	0000h	*(size of data)*
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	SSI Identifier	SSI Identifier	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	-	
Extended word 8	-	-	
Message Data	SSI Data		

- **SSI Identifier**

  Identifier which can be used as desired by the application to address a specific block of data.

- **SSI Data**

  Data associated with the specified SSI Identifier.

10. Fieldbus Specific Area

10.1 Memory Map

Address	Contents	Access
640h - 641h	No. of configured originated connections	RO
642h - 643h	No. of active originated connections	RO
644h - 645h	No. of erroneous originated connections	RO
646h - 64Fh	(reserved)	-
650h	Scanner Mode	RO

- **No. of Configured Originated Connection**
  Number of connections that are configured in the Connection Object.
- **No. of Active Originated Connection**
  Number of connections that are configured and opened without errors.
- **No. of Erroneous Originated Connection**
  Number of connections that cannot be opened because of errors.
- **Scanner Mode**
  0: Idle
  1: Run

A. Miscellaneous

A.1 Control Register Area

Fieldbus Type

The fieldbus type value for this product is 0085h.

Module Type

The module type value for this product is 0201h (Anybus-M).

Watchdog Counter Input (7D2h... 7D3h)

If the application has enabled the Watchdog Counter Input and doesn't update it properly, the module will cease all network participation and indicate an internal error on the Module Status LED.

Event Notification Cause/Source Registers

- **ON/OFF Line Indication (FBON/FBOF)**
  The module is considered on-line when a link has been established on the Ethernet interface.
- **Network Reset Functionality (RST)**
  The application can be notified of network reset requests via the Event Notification register.

A.2 Firmware Upgrade

The Anybus module supports firmware updates via FTP. Follow the steps below:

1. As a precaution, make a backup copy of the filesystem contents before proceeding.
2. Initialise the module
3. Upload the new firmware file(s) to the system root ("\"), or to the 'cfg\'-directory.
4. Reset the Anybus module and wait until the on board LED (watch dog LED) flashes 2Hz green (This may take up to 1 minute).
5. Reset the Anybus module.
   The new firmware is now operational.

A.3 Formatting the File System

In case of major file system damage, it is possible to reinitialize the file system as follows:

1. Attach a short jumper to the PCB as shown in figure. Apply power.

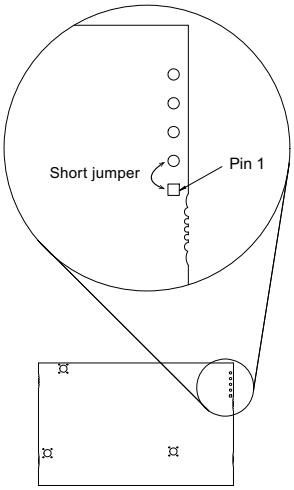
2. Wait until the watchdog LED turns red.

3. Disconnect power.

4. Remove jumper.

5. Apply power.

6. Wait approx. 1 minute while the filesystem is being formatted.



B. Technical Specification

B.1 Electrical Specification

B.1.1 Protective Earth (PE) Requirements

All Anybus-S/M modules feature cable shield filters designed in accordance with each network standard. To be able to support this, the application *must* provide a connection to PE (Protective Earth) as described in the general Anybus-S Parallel Design Guide. HMS cannot guarantee proper EMC behaviourAnybus-M EtherNet/IP unless this requirement is fulfilled.

B.1.2 Isolation

Isolation between the application, the network, and protective earth (PE):

Isolation Barrier	Working Voltage		Distance	
	Creepage	Clearance	External	Internal
Application to PE	200V	2500V	2.0mm	0.4mm
Application to Network	250V	2500V	2.5mm	0.4mm
Network to PE	100V	1500V	1.4mm	0.4mm

(Tests performed according to EN 60950-1)

B.1.3 Power Supply

Supply Voltage

The module requires a regulated 5V power supply as specified in the Anybus-S Parallel Design Guide.

Power Consumption

The maximum power consumption is 450mA.

B.2 Environmental Specification

Temperature

Tests performed according to IEC-60068-2-1, IEC-60068-2-2 and IEC 60068-2-14.

Operating:	0 to 70°C(32 to 158°F)
Storage:	-25 to 85°C(-13 to 185°F)

Humidity

The product is designed for a relative humidity of 5 to 95% non-condensing.

Tests performed according to EN 60068.

B.3 EMC (CE) Pre-compliance

EMC pre-compliance testing has been conducted according to the Electromagnetic Compatibility Directive 2004/108/EC. For more information please consult the EMC pre-compliance document, see product/support pages for Anybus-M EtherNet/IP at www.anybus.com.

C. Connectors

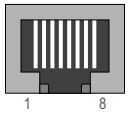
C.1 Application Connector

(Consult the general Anybus-S Parallel Design Guide for more information)

C.2 Ethernet

RJ45 (Standard Connector)

Pin	Signal	Notes
1	TD+	-
2	TD-	-
3	RD+	-
4	-	Normally left unused; to ensure signal integrity, these pins are tied together and terminated to PE via a filter circuit in the module.
5	-	
6	RD-	-
7	-	Normally left unused; to ensure signal integrity, these pins are tied together and terminated to PE via a filter circuit in the module.
8	-	

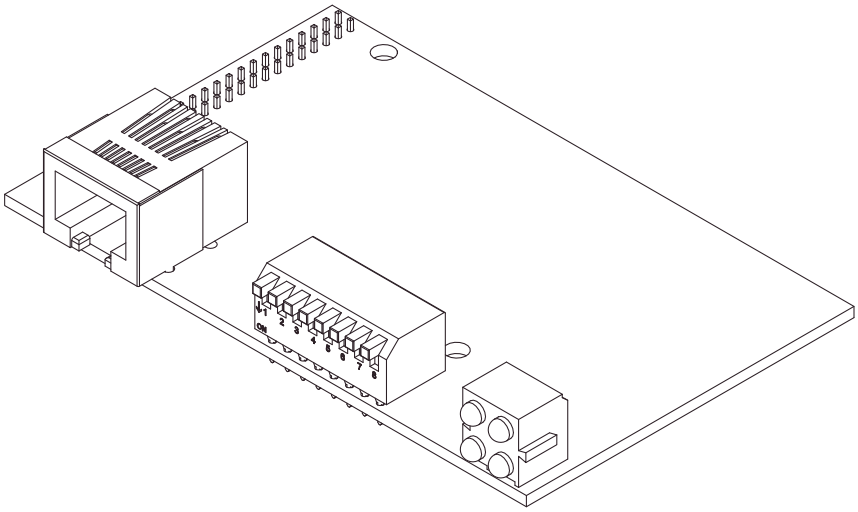


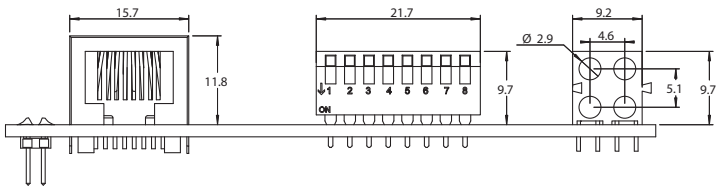
Board to Board

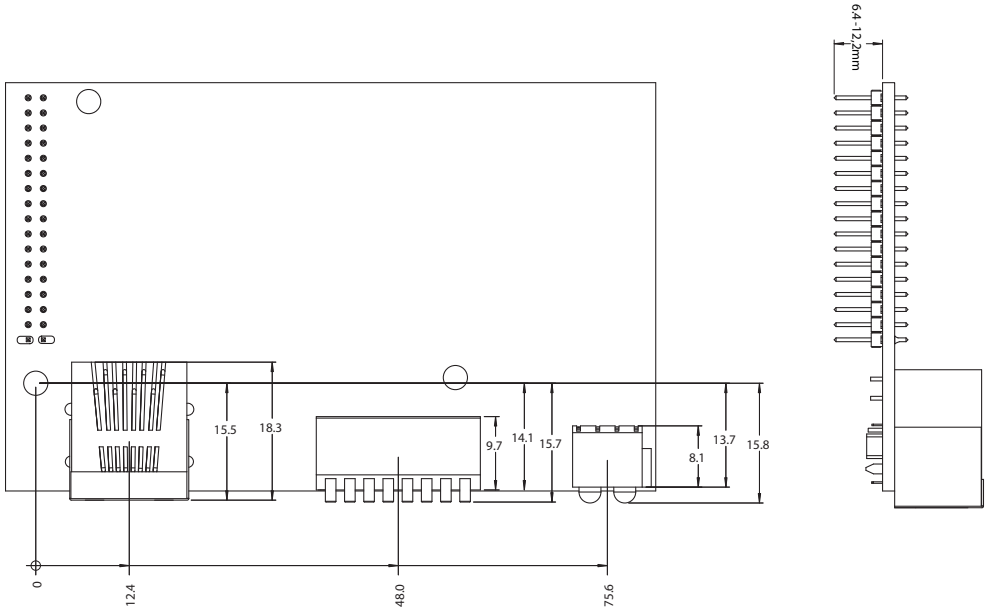
Pin	Signal	Connect to RJ45 pin...	Notes
1	Shield	Housing	-
2	-	4	(See notes for pins 4 and 5 in RJ45 connector)
3	-	5	
4	-	-	(not used)
5	TD+	1	-
6	TD-	2	-
7	RD+	3	-
8	-	7	(See notes for pins 7 and 8 in RJ45 connector)
9	RD-	6	-
10	-	8	(See notes for pins 7 and 8 in RJ45 connector)

D. Mechanical Specification

D.1 Measurements, Connectors & LEDs







E. Copyright Notices

This product includes software developed by Carnegie Mellon, the Massachusetts Institute of Technology, the University of California, and RSA Data Security: