Anybus® CompactCom™ 40

Modbus-TCP® Transparent Ethernet

NETWORK GUIDE

SCM-1202-022 1.3 en-US ENGLISH

Important User Information

Disclaimer

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

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

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

Table of Contents

Page

This page intentionally left blank

1 Preface

1.1 About this document

This document is intended to provide a good understanding of the functionality offered by the Anybus CompactCom 40 Modbus-TCP. The document describes the features that are specific to Anybus CompactCom 40 Modbus-TCP. For general information regarding Anybus CompactCom, consult the Anybus CompactCom design guides.

The reader of this document is expected to be familiar with high level software design and communication systems in general. The information in this network guide should normally be sufficient to implement a design. However if advanced Modbus-TCP specific functionality is to be used, in-depth knowledge of Modbus-TCP networking internals and/or information from the official Modbus-TCP specifications may be required. In such cases, the persons responsible for the implementation of this product should either obtain the Modbus-TCP specification to gain sufficient knowledge or limit their implementation in such a way that this is not necessary.

For additional related documentation and file downloads, please visit the support website at www.anybus.com/support.

1.2 Related Documents

Document	Author	Document ID
Anybus CompactCom 40 Software Design Guide	HMS	HMSI-216-125
Anybus CompactCom M40 Hardware Design Guide	HMS	HMSI-216-126
Anybus CompactCom B40 Design Guide	HMS	HMSI-27-230
Anybus CompactCom Host Application Implementation Guide	HMS	HMSI-27-334
Anybus CompactCom 40 Modbus-TCP Network Guide	HMS	SCM-1202-027

1.3 Document History

Version	Date	Description
1.0	2017-01-17	First release
1.1	2017-07-11	Added appendix on backward compatibility Updated description of module status LED
1.2	2018-05-28	Minor update to Network Configuration Object Minor corrections to common files
1.3	2019-06-10	Rebranding

1.4 Document Conventions

Ordered lists are used for instructions that must be carried out in sequence:

1. First do this
2. Then do this

Unordered (bulleted) lists are used for:

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

...and for action-result type instructions:

► This action...

→ leads to this result

Bold typeface indicates interactive parts such as connectors and switches on the hardware, or menus and buttons in a graphical user interface.

```
Monospaced text is used to indicate program code and other
kinds of data input/output such as configuration scripts.
```

This is a cross-reference within this document: *Document Conventions, p. 5*

This is an external link (URL): www.hms-networks.com

This is additional information which may facilitate installation and/or operation.

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

Caution
This instruction must be followed to avoid a risk of personal injury.

WARNING
This instruction must be followed to avoid a risk of death or serious injury.

1.5 Document Specific Conventions

- The terms “Anybus” or “module” refers to the Anybus CompactCom module.
- The terms “host” or “host application” refer to the device that hosts the Anybus.
- Hexadecimal values are written in the format NNNNh or 0xNNNN, where NNNN is the hexadecimal value.
- A byte always consists of 8 bits.
- The terms “basic” and “extended” are used to classify objects, instances and attributes.

1.6 Trademarks

Anybus® is a registered trademark of HMS Industrial Networks.

All other trademarks are the property of their respective holders.

2 About the Anybus CompactCom 40 Modbus-TCP

2.1 General

The Anybus CompactCom 40 Modbus-TCP communication module provides instant Ethernet and Modbus-TCP connectivity via the patented Anybus CompactCom host interface. Any device that supports this standard can take advantage of the features provided by the module, allowing seamless network integration regardless of network type.

> ! This network guide covers the Transparent Ethernet version of the product. Transparent Ethernet has to be enabled during setup, or the device will appear as an Anybus CompactCom 40 with full IT functionality. The IT functionality is described in the network guide for the standard Anybus CompactCom 40.
>
> IT functionality is not available when Transparent Ethernet is enabled.

The modular approach of the Anybus CompactCom 40 platform allows the CIP-object implementation to be extended to fit specific application requirements. Furthermore, the Identity Object can be customized, allowing the end product to appear as a vendor-specific implementation rather than a generic Anybus module.

This product conforms to all aspects of the host interface for Anybus CompactCom 40 modules defined in the Anybus CompactCom 40 Hardware and Software Design Guides, making it fully interchangeable with any other device following that specification. Generally, no additional network related software support is needed, however in order to be able to take full advantage of advanced network specific functionality, a certain degree of dedicated software support may be necessary.

2.2 Features

- Transparent Ethernet
- Two Ethernet ports
- Ethernet connectors
- 10/100 Mbit, full/half duplex operation
- Modbus-TCP server/slave (up to 4 simultaneous connections)
- Max. read process data: 1536 bytes
- Max. write process data: 1536 bytes
- Max. process data (read + write, in bytes): 3072 bytes
- Customizable Identity Information
- Modular Device functionality

2.3 Transparent Ethernet

Transparent Ethernet offers the possibility for a host application, that includes an IT implementation (web pages, file system, a proprietary protocol etc.), to let the Anybus CompactCom handle an industrial Ethernet protocol (in this case Modbus-TCP), without the need for extra Ethernet ports.

Ethernet communication is routed straight to the host application system using an RMII interface. The host application must include an Ethernet controller and a TCP/IP stack. Modbus-TCP protocol messages will be routed to the Anybus CompactCom internal software. Please note that the Transparent Ethernet functionality has to be enabled during startup by setting attribute #16 (instance #1) in the Anybus Object.

16–bit parallell mode can not be used, as specific host application connector pins are reserved for transparent Ethernet. Also TCP/UDP ports may be reserved, and can, in that case, not be used for the transparent Ethernet communication.

See also ...

- *Transparent Ethernet, p. 20*
- Anybus CompactCom 40 Hardware Design Guide
- *Anybus Object (01h), p. 25*

3 Basic Operation

3.1 Software Requirements

No additional network support code needs to be written in order to support the Anybus CompactCom 40 Modbus-TCP, however due to the nature of the Modbus-TCP networking system, certain restrictions must be taken into account:

- The total number of ADIs that can be represented on the network depends on their size. By default, ADIs with instance numbers 1...3839 can be accessed from the network, each with a size of up to 32 bytes.
- ADI names, types and similar attributes cannot be accessed viaModbus-TCP. They are however represented on the network through the built in web server.
- A network write access of an ADI mapped to process data will result in a corresponding write access of the process data buffer of the Anybus CompactCom 40 Modbus-TCP.
- A network read access of an ADI, even if it is mapped to process data, will result in a corresponding Get_Attribute command towards the application.
- Modbus-TCP reset requests are not supported.
- Up to 5 diagnostic instances (See Diagnostic Object) can be created by the host application during normal conditions. An additional 6th instance may be created in event of a major fault.
- Modbus-TCP in itself does not impose any specific timing demands when it comes to acyclic requests (i.e. requests towards instances in the Application Data Object), however it is generally recommended to process and respond to such requests within a reasonable time period (exactly what this means in practice depends on the implementation and the actual installation).
- The use of advanced Modbus-TCP specific functionality may require in-depth knowledge in Modbus-TCP networking internals and/or information from the official Modbus-TCP specifications. In such cases, the people responsible for the implementation of this product is expected either to obtain these specifications to gain sufficient knowledge or limit their implementation is such a way that this is not necessary.

See also...

- *Application Data (ADIs), p. 13*
- *Diagnostic Object (02h), p. 27*
- Anybus CompactCom 40 Software Design Guide, “Application Data Object (FEh)”

For in depth information regarding the Anybus CompactCom software interface, consult the Anybus CompactCom 40 Software Design Guide.

3.2 Device Customization

3.2.1 Modbus-TCP Implementation

By default, a “Read Device Identification” request returns the following information:

Vendor Name	“HMS”
Product Code:	“Anybus CompactCom 40 Modbus-TCP”
Major Minor Rev.:	The current firmware version of the product
Vendor URL:	(no information returned by default)
Product Name:	(no information returned by default)
Model Name:	(no information returned by default)
User Application Name:	(no information returned by default)

It is possible to customize this information by implementing the Modbus Host Object. See *Modbus Host Object (FAh), p. 41* for more information.

3.3 Communication Settings

As with other Anybus CompactCom products, network related communication settings are grouped in the Network Configuration Object (04h).

In this case, this includes...

Ethernet Interface Settings	By default, the module is set to autonegotiate the physical link settings. It is, however, possible to force the module to use a specific setting if necessary.
IP Settings	These settings must be set properly in order for the module to be able to participate on the network. IP settings must be synchronized between the Anybus CompactCom 40 and the host application. The module supports DHCP, which may be used to retrieve the IP settings from a DHCP-server automatically. DHCP is enabled by default, but can be disabled if necessary.
Modbus-TCP Connection Timeout	This setting specifies how long a Modbus-TCP connection may be idle before it is closed by the module (default is 60 seconds).
Process Active Timeout	This value specifies how long the module shall stay in the 'PROCESS_ACTIVE' state after receiving a Modbus-TCP request. See*Network Configuration Object (04h), p. 28* for more information. **Note**: This value can be accessed from the Modbus registers. **Note**: This value affects the behavior of the SUP-bit. See *SUP-Bit Definition, p. 50*.

See also...

Network Configuration Object (04h), p. 28 (Anybus Module Object)

Secure HICP (Secure Host IP Configuration Protocol), p. 51

3.4 Diagnostics

Each instance within the Diagnostic Object (02h) is represented on the network as a dedicated entry in the Modbus register map (see *Input Registers (3x), p. 15*).

Note that since each entry corresponds *directly* to a specific diagnostic instance, it is possible to have "empty" diagnostic entries in the register map (when read, such entries will return zeroes).

See also...

Input Registers (3x), p. 15

Diagnostic Object (02h), p. 27

3.5 Network Data Exchange

3.5.1 General

It is important to notice that various register areas might have different response times. Generally queries directed at the process data registers will be answered more quickly than those directed at the ADI-related registers since the former are directly processed by the module itself whereas the latter are forwarded to the application, which must respond before the module can respond to the master. In the latter case this will influence the allowable timeout time for the master to use for these registers.

3.5.2 Application Data (ADIs)

As mentioned previously, the total number of ADIs that can be represented on the network depends on their size. By default, ADIs with instance numbers 1...3839 can be accessed from the network, each with a size of up to 32 bytes. It is possible to alter this ratio by changing the number of ADI indexing bits. See attribute #9, in the *Modbus Host Object (FAh), p. 41*.).

Example 1 (Default Settings)

In this example, attribute #9 in the Modbus Host Object (FAh) is set to its default value (04h).

Holding Register #	ADI No.
1010h... 101Fh	1
1020h... 102Fh	2
1030h... 103Fh	3
1040h... 104Fh	4
...	...
FFE0h... FFEFh	3838
FFF0h... FFFFh	3839

Each ADI is represented using 16 Modbus registers, which means that up to 32 bytes of an ADI can be accessed from the network.

Example 2 (Customized Implementation)

In this example, attribute #9 in the Modbus Host Object (FAh) is set to 05h.

Holding Register #	ADI No.
1010h... 102Fh	1
1030h... 104Fh	2
1050h... 106Fh	3
1070h... 108Fh	4
...	...
FFB0h... FFCFh	1918
FFD0h... FFEFh	1919

Each ADI is represented using 32 Modbus registers, which means that up to 64 bytes of an ADI can be accessed from the network.

3.5.3 Process Data

Modbus does not feature a dedicated cyclic data channel in the same sense as many other networks. In the Anybus CompactCom 40 implementation, process data can however still be accessed from the network via dedicated entries in the Modbus register map.

Process data can be accessed on a bit by bit basis (as Coils & Discrete Inputs) - *or* - as 16 bit entities (Holding Registers & Input Registers).

ⓘ *For natural reasons, writing to the write process data register area has no effect, and reading unused register locations will return zeroes.*

Example

Each 16-bit Modbus register contains 2 bytes from the process data at the corresponding address, i.e. Modbus register N holds process data byte (N*2) in the low byte and (N*2 + 1) in the high byte.

Process Data			**Modbus Register**		
Byte	**Type**	**Value**	**Register**	**Value**	**Comment**
0	UINT16	1234h	0	1234h	-
1					
2	UINT8	00h	1	FF00h	Two bytes from the process data in one register.
3	UINT8	FFh			
4	UINT32	11223344h	2	3344h	LSB * A 32-bit type occupies two Modbus * registers. MSB
5					
6			3	1122h	
7					
8	BOOL[3]	01h	4	0001h	-
9		00h			
10		01h	5	3401h	-
11	UINT16	1234h			
12			6	0012h	High byte from unmapped process data is set to zero.

4 Modbus-TCP Register Implementation

4.1 Holding Registers (4x)

Range	Contents	Notes
0000h...02FFh	Read Process Data (1536 bytes)	-
0300h...07FFh	Reserved	-
0800h...0AFFh	Write Process Data (1536 bytes)	-
0B00h...0FFFh	Reserved	-
1000h...1002h	Reserved	-
1003h	Process Active Timeout	See *Network Configuration Object (04h), p. 28*
1004h	Enter/Exit Idle Mode	0: Not Idle, >0: Idle
1005h...100Fh	Reserved	-
1010h...101Fh	ADI Number 1	See *Application Data (ADIs), p. 13*
1020h...102Fh	ADI Number 2	
...	...	
FFF0h...FFFFh	ADI Number 3839	

4.2 Input Registers (3x)

Range	Contents	Notes
0000h...02FFh	Write Process Data	-
0300h...07FFh	Reserved	-
0800h	Diagnostic Event Count	Number of pending diagnostic events. There may be "gaps" between active diagnostic events. Inactive diagnostic events return 0000h when read.
0801h	Diagnostic Event #1	These registers corresponds to instances in the Diagnostic Object (02h), see *Diagnostic Object (02h), p. 27*. High byte = Severity Low byte = Event Code
0802h	Diagnostic Event #2	
0803h	Diagnostic Event #3	
0804h	Diagnostic Event #4	
0805h	Diagnostic Event #5	
0806h	Diagnostic Event #6	

4.3 Coils (0x)

Range	Contents	Notes
0000h...2FFFh	Read Process Data	-
3000h...7FFFh	Reserved	-

4.4 Discrete Inputs (1x)

Range	Contents	Notes
0000h...2FFFh	Write Process Data	-
3000h...07FFh	Reserved	-

5 Modbus-TCP Functions

The following Modbus-TCP functions are implemented in the module:

#	Function
1	Read Coils
2	Read Discrete Inputs
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Register
15	Write Multiple Coils
16	Write Multiple Registers
23	Read/Write Multiple Registers
43/14	Read Device Identification

Exception Codes:

Code	Name	Description
0x01	Illegal function	The function code in the query is not supported
0x02	Illegal data address	The data address received in the query is outside the initialized memory area
0x03	Illegal data value	The data in the request is illegal

See also...

Modbus Host Object (FAh), p. 41

5.1 Read Coils

Function Code:	1
Register Type:	0x (Coils)

Details

This function is mapped to the Read Process data as follows:

Coil #	Process Data Byte #	Bit #
0000h	0000h	0
0001h		1
0002h		2
0003h		3
...		...
0007h		7
0008h	0001h	0
0009h		1
000Ah		2
000Bh		3
...		...
000Fh		7
...	...	...
2FF8h	05FFh	0
2FF9h		1
2FFAh		2
2FFBh		3
...		...
2FFFh		7

5.2 Read Discrete Inputs

Function Code:	2
Register Type:	1x (Discrete Inputs)

Details

This function is mapped to the Write Process data; the mapping is otherwise identical to that of the “Read Coils” function described above.

5.3 Read Holding Registers

Function Code:	3
Register Type:	4x (Holding Registers)

Details

Mapped to Read- and Write Process Data, ADIs, and configuration registers. It is allowed to read parts of a larger Anybus CompactCom data type; it is also allowed to read multiple ADIs using a single request.

5.4 Read Input Registers

Function Code:	4
Register Type:	3x (Input Registers)

Details

Mapped to Write Process Data and diagnostic events.

5.5 Write Single Coil

Function Code:	5
Register Type:	0x (Coils)

Details

This function is mapped to the Read Process data, and the mapping is identical to that of the “Read Coils” function described above.

5.6 Write Single Register

Function Code:	6
Register Type:	4x (Holding Registers)

Details

Mapped to Read- and Write Process Data, ADIs and configuration registers. ADIs must be written as a whole, however the Process Data area accepts writes of any size.

5.7 Write Multiple Coils

Function Code:	15
Register Type:	0x (Coils)

Details

This function is mapped to the Read Process data, and the mapping is identical to that of the “Read Coils” function described above.

5.8 Write Multiple Registers

Function Code:	16
Register Type:	4x (Holding Registers)

Details

Mapped to Read- and Write Process Data, ADIs and configuration registers.

ℹ *ADIs must be written as a whole, but the Process Data area accepts writes of any size.*

5.9 Read/Write Multiple Registers

Function Code:	23
Register Type:	4x (Holding Registers)

Details

Mapped to Read- and Write Process Data, ADIs and configuration registers.

ℹ *ADIs must be written as a whole, but the Process Data area accepts writes of any size.*

It is allowed to read parts of larger data types, and to read multiple ADIs using a single request.

The write operation is performed before the read. If there is an overlap in the read and write ranges, the newly written data will be returned by the read operation.

5.10 Read Device Identification

Function Code:	43 (subcode 14)
Register Type:	-

Details

Basic and regular device identification objects are supported according to the Modbus specification. Extended device identification objects are not supported.

Identification strings are extracted from the host application via the *Modbus Host Object (FAh), p. 41*. If this object is not implemented, the default identification strings will be returned

6 Transparent Ethernet

6.1 General Information

Transparent Ethernet offers the possibility for a host application, that includes an IT implementation, to let the Anybus CompactCom handle an industrial Ethernet protocol (in this case Modbus-TCP), without the need for extra Ethernet ports.

Ethernet communication that is not related to Modbus-TCP is internally routed via the RMII interface to the Ethernet port and the TCP/IP stack of the host application. The IP configurations and the MAC addresses of the host application and the Anybus CompactCom must be the same.

The RMII interface is accessed through the host application connector. Please note that the 16 bit parallel interface is not available when transparent Ethernet is enabled. See the *Anybus CompactCom M40 Hardware Design Guide* for more information.

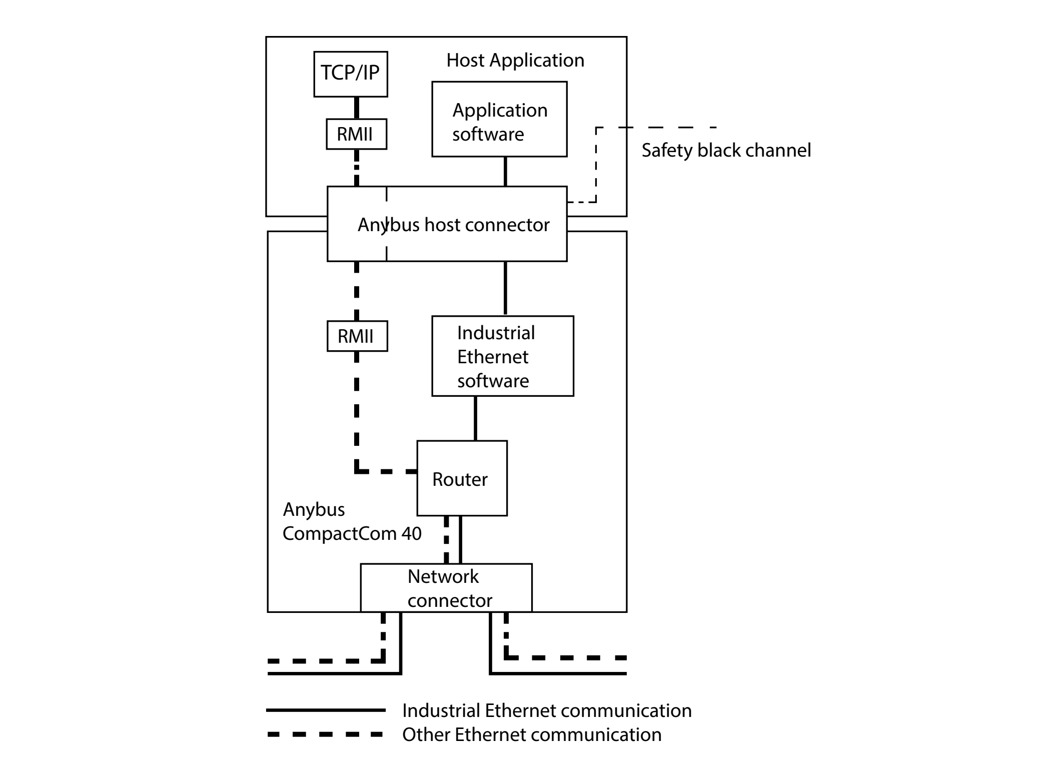


Fig. 1

! Transparent Ethernet has to be enabled by setting instance attribute #16 in the Anybus Object (01h) during setup, see below.

16–bit parallell mode cannot be used when using transparent Ethernet.

MAC addresses and IP configurations have to be synchronized, see below.

Some EtherTypes ,TCP/UDP ports and multicast MAC addresses may be reserved for use by the industrial Ethernet network. These must not be used for transparent Ethernet communication. See below for more information.

The Transparent Ethernet interface only supports 100 Mbit, full duplex operation.

6.2 Enabling Transparent Ethernet

Transparent Ethernet is not enabled at delivery. Attribute #16 (instance #1) in the Anybus Object (01h) has to be set to 0002h during setup. If this attribute is not changed, the Anybus

CompactCom 40 Modbus-TCP will start up with full IT functionality instead of transparent Ethernet functionality. Transparent Ethernet cannot be enabled after setup is finished. Once Transparent Ethernet is enabled, no IT functionality is enabled.

6.3 MAC Address Synchronization

The host application and the Anybus CompactCom must use the same MAC address when communicating on Ethernet. The host application must make sure that this is the case. This can be accomplished in ether of the two ways described below:

- The pre-programmed MAC address in attribute #1 (instance #1) in the Network Ethernet Object (0Ch) is read and used by the host application when communicating on Ethernet.
- The Ethernet Host Object (F9h, instance #1, attribute #1) is implemented in the application, set with a MAC address provided and used by the application. At initialization, the Anybus CompactCom will read and then use the application provided MAC address from this object.

6.4 IP Configuration Synchronization

The host application TCP/IP stack and the Anybus CompactCom 40 Modbus-TCP TCP/IP stack must use the same IP configuration when communicating on Ethernet. The Anybus CompactCom 40 Modbus-TCP will write its currently used IP configuration to instance attribute #16 in the Ethernet Host Object (F9h) whenever the configuration is assigned or changed. The host application must use this configuration. DNS server and domain names can be read from the Network Configuration Object (04h) after an IP configuration update.

6.5 Routing Restrictions

The internal router receives all frames from the network. The frames that are intended for the industrial Ethernet network internal software, are recognized and routed to the Anybus CompactCom. The remaining Ethernet frames will be routed to the host application. Some restrictions apply to the use of e.g. UDP and TCP ports, sometimes also depending on industrial Ethernet network. If the host application is intended only for use with Modbus-TCP, the restrictions for the other networks can be ignored.

> **!** The host application is responsible for taking the following restrictions into consideration. If they are not followed, the Ethernet communication will not work correctly.

6.5.1 EtherTypes

The Anybus CompactCom internally uses bit 12 and bit 13 (mask 3000h) in the EtherType. Thus the host application cannot implement protocols using Ethertype, where these bits are used..

The following EtherTypes are used by PROFINET and should not be used by the host application:

- 8892h (PNIO)
- 88CCh (LLDP)
- 88E3h (MRP)

6.5.2 Multicast MAC Addresses

The host application must not transmit or receive any data from and to the following MAC addresses as they may be used by the industrial Ethernet network (PROFINET):

- 01-0E-CF-XX-XX-XX
- 01-80-C2-00-00-0E
- 01-00-5E-40-F8-00 ... 01-00-5E-40-FB-FF
- X3-XX-00-00-00-00

(X: any number 0-F)

6.5.3 UDP/TCP Ports

The following ports may be used by the Anybus CompactCom, and must not be used by the host application:

- UDP 67 & 68 (DHCP)
- UDP 161 (SNMP)
- UDP 3250 (HICP)

The following ports are reserved for use by PROFINET:

- UDP 34962 (PROFINET RT Unicast)
- UDP 34963 (PROFINET RT Multicast)
- UDP 34969 (PROFINET RPC Context Manager)
- UDP 53247 (PROFINET RPC Client/Server)

The following ports are reserved for use by EtherNet/IP:

- UDP 2222 (Implicit messaging)
- UDP & TCP 44818 (Explicit messaging)

The following port is reserved for use by Modbus TCP:

- TCP 502 (Modbus messaging)

7 Firmware Upgrade

The Anybus CompactCom 40 Modbus-TCP firmware can be updated either by running the Firmware Manager II tool (FMII), available at www.anybus.com/support, or by downloading the firmware upgrade file directly to the host application file system. For any of these methods to work the following needs to be implemented and/or performed:

- HICP needs to be enabled (FMII only).
- An FTP server needs to be implemented in the host application.
- A directory named “firmware” in the host application FTP root.
- The file module.nfo in the “firmware” directory in theAnybus CompactCom file system has to be copied to the “firmware” directory in the host application file system (FMII only). The Anybus CompactCom file system is accessed via the Anybus File System Object (0Ah).

Once a firmware file has been downloaded, the host application must be able to:

- detect a new file in the “firmware” directory
- download this file to the “firmware” directory in the Anybus CompactCom (The Anybus CompactCom file system is accessed via the Anybus File System Object (0Ah).)

The firmware will be updated upon the next reset of the Anybus CompactCom 40 Modbus-TCP.

8 Anybus Module Objects

8.1 General Information

This chapter specifies the Anybus Module Object implementation and how they correspond to the functionality in the Anybus CompactCom 40 Modbus-TCP.

Standard Objects:

- *Anybus Object (01h), p. 25*
- *Diagnostic Object (02h), p. 27*
- *Network Object (03h), p. 28*
- *Network Configuration Object (04h), p. 28*

Network Specific Objects:

- *Anybus File System Interface Object (0Ah), p. 38*
- *Network Ethernet Object (0Ch), p. 39*

8.2 Anybus Object (01h)

Category

Basic, extended

Object Description

This object assembles all common Anybus data, and is described thoroughly in the general *Anybus CompactCom 40 Software Design Guide*.

> **!** Instance attribute #16 has to be set to 0002h during SETUP state to enable Transparent Ethernet functionality.

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Set_Attribute Get_Enum_String

Object Attributes (Instance #0)

(Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.)

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	Module type	Get	UINT16	0403h (Standard Anybus CompactCom 40)
2... 11	-	-	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
12	LED colors	Get	struct of: UINT8 (LED1A) UINT8 (LED1B) UINT8 (LED2A) UINT8 (LED2B)	Value: Color: 01h Green 02h Red 01h Green 02h Red
13... 15	-	-	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
16	GPIO configuration	Get/Set	UINT16	Configuration of the host interface GIO pins. To enable Transparent Ethernet, this attribute has to be set to 0002h during SETUP state.

Extended

#	Name	Access	Type	Value
17	Virtual attributes	Get/Set	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
18	Black list/White list	Get/Set		
19	Network time	Get	UINT64	0 (Not supported)

8.3 Diagnostic Object (02h)

Category

Basic

Object Description

This object provides a standardized way of handling host application events & diagnostics, and is thoroughly described in the general *Anybus CompactCom 40 Software Design Guide*.

See also ...

- *Diagnostics, p. 12*

Supported Commands

Object:	Get_Attribute Create Delete
Instance:	Get_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1... 4	-	-	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
11	Max no. of instances	Get	UINT16	5+1 (Of the maximum number of instances there should always be one instance reserved for an event of severity level “Major, unrecoverable”, to force the module into the state EXCEPTION.)
12	Supported functionality	Get	BITS32	Bit 0: “0” (Latching events are not supported) Bit 1 - 31: reserved (shall be “0”)

Instance Attributes (Instance #1)

Extended

#	Name	Access	Data Type	Value
1	Severity	Get	UINT8	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
2	Event Code	Get	UINT8	
3	-	-	-	Not implemented in product
4	Slot	Get	UINT16	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
5	ADI	Get	UINT16	
6	Element	Get	UINT8	
7	Bit	Get	UINT8	

8.4 Network Object (03h)

Category

Basic

Object Description

For more information regarding this object, consult the general *Anybus CompactCom 40 Software Design Guide*.

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Set_Attribute Get_Enum_String Map_ADI_Write_Area Map_ADI_Read_Area Map_ADI_Write_Ext_Area Map_ADI_Read_Ext_Area

Object Attributes (Instance #0)

(Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.)

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	Network type	Get	UINT16	0093h
2	Network type string	Get	Array of CHAR	“Ethernet Modbus-TCP”
3	Data format	Get	ENUM	00h (LSB first)
4	Parameter data support	Get	BOOL	True
5	Write process data size	Get	UINT16	Current write process data size (in bytes) Updated on every successful Map_ADI_Write_Area. (Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.)
6	Read process data size	Get	UINT16	Current read process data size (in bytes) Updated on every successful Map_ADI_Read_Area. (Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.)
7	Exception Information	Get	UINT8	Additional information available if the module has entered the EXCEPTION state. Value: Meaning: 00h No information available 01h Invalid assembly instance mapping 02h Missing MAC address (Only valid for Anybus IP)

8.5 Network Configuration Object (04h)

Category

Extended

Object Description

This object holds network specific configuration parameters that may be set by the end user. A reset command (factory default) issued towards this object will result in all instances being set to their default values.

If the settings in this object do not match the configuration used, the Module Status LED will flash red to indicate a minor error.

As soon as the used combination of IP address, Subnet mask and Gateway is changed, the module informs the application by writing the new set to instance #1, attribute #16 in the Ethernet Host Object (F9h).

The object is described in further detail in the Anybus CompactCom 40 Software Design Guide.

See also...

- *Communication Settings, p. 12*
- *Ethernet Host Object (F9h), p. 44*

Supported Commands

Object:	Get_Attribute Reset
Instance:	Get_Attribute Set_Attribute Get_Enum_String

Object Attributes (Instance #0)

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Network Configuration”
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0012h (18)
4	Highest instance number	Get	UINT16	0016h (22)

(Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.)

Instance Attributes (Instance #3, IP Address)

Value is used after module reset.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“IP address” (Multilingual, see page 37)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	04h (four elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of UINT8	Any change is valid after reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)

Instance Attributes (Instance #4, Subnet Mask)

Value is used after module reset.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Subnet mask” (Multilingual, see page 37)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	04h (four elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of UINT8	Any change is valid after reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)

Instance Attributes (Instance #5, Gateway Address)

Value is used after module reset.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Gateway” (Multilingual, see page 37)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	04h (four elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of UINT8	Any change is valid after reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)

Instance Attributes (Instance #6, DHCP Enable)

Value is used after module reset.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“DHCP” (Multilingual, see page *37*)
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	ENUM	Any change is valid after reset. (Multilingual, see page *37*) Value String Meaning 00h “Disable” DHCP disabled 01h “Enable” DHCP enabled (default)
6	Configured Value	Get	ENUM	Holds the configured value, which will be written to attribute #5 after the module has been reset. Value String Meaning 00h “Disable” DHCP disabled 01h “Enable” DHCP enabled (default)

Instance Attributes (Instance #7 Ethernet Communication Settings 1)

Changes have immediate effect.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Comm 1” (Multilingual, see page *37*)
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	ENUM	Value String Meaning (Multilingual, see page *37*) 00h “Auto” Auto negotiation (default) 01h “10 HDX” 10Mbit, half duplex 02h “10 FX” 10Mbit, full duplex 03h “100HDX” 100Mbit, half duplex 04h “100FX” 100Mbit, full duplex
6	Configured Value	Get	ENUM	Holds the configured value, which will be written to attribute #5 after the module has been reset. Value String Meaning (Multilingual, see page *37*) 00h “Auto” Auto negotiation (default) 01h “10 HDX” 10Mbit, half duplex 02h “10 FX” 10Mbit, full duplex 03h “100HDX” 100Mbit, half duplex 04h “100FX” 100Mbit, full duplex

Instance Attributes (Instance #8 Ethernet Communication Settings 2)

Changes have immediate effect.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	"Comm 2" (Multilingual, see page 37)
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	ENUM	Value / String / Meaning (Multilingual, see page 37) 00h "Auto" Auto negotiation (default) 01h "10 HDX" 10Mbit, half duplex 02h "10 FX" 10Mbit, full duplex 03h "100HDX" 100Mbit, half duplex 04h "100FX" 100Mbit, full duplex
6	Configured Value	Get	ENUM	Holds the configured value, which will be written to attribute #5 after the module has been reset. Value / String / Meaning (Multilingual, see page 37) 00h "Auto" Auto negotiation (default) 01h "10 HDX" 10Mbit, half duplex 02h "10 FX" 10Mbit, full duplex 03h "100HDX" 100Mbit, half duplex 04h "100FX" 100Mbit, full duplex

Instance Attributes (Instance #9, DNS1)

This instance holds the address to the primary DNS server. Changes are valid after reset..

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	"DNS1" (Multilingual, see page 37)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	04h (four elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of UINT8	Any change is valid after reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)

Instance Attributes (Instance #10, DNS2)

This instance holds the address to the secondary DNS server. Changes are valid after reset..

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“DNS2” (Multilingual, see page *37*)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	04h (four elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of UINT8	Any change is valid after reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0)

Instance Attributes (Instance #11, Host name)

This instance holds the host name of the module. Changes are valid after reset..

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Host name” (Multilingual, see page *37*)
2	Data type	Get	UINT8	07h (= CHAR)
3	Number of elements	Get	UINT8	40h (64 elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of CHAR	Any change is valid after reset. Host name, 64 characters
6	Configured Value	Get	Array of CHAR	Holds the configured value, which will be written to attribute #5 after the module has been reset. Host name, 64 characters

Instance Attributes (Instance #12, Domain name)

This instance holds the domain name. Changes are valid after reset..

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Host name” (Multilingual, see page *37*)
2	Data type	Get	UINT8	07h (= CHAR)
3	Number of elements	Get	UINT8	30h (48 elements)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	Array of CHAR	Any change is valid after reset. Domain name, 48 characters
6	Configured Value	Get	Array of CHAR	Holds the configured value, which will be written to attribute #5 after the module has been reset. Domain name, 48 characters

Instance Attributes (Instances #13 - #15)

(Reserved)

Instance Attributes (Instance #16, MDI 1 Settings)

This instance holds the settings for MDI/MDIX 1. Changes have immediate effect.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“MDI 1”
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	ENUM	Value (ENUM): String: Meaning: 00h “Auto” (default) 01h “MDI” 02h “MDIX”
6	Configured Value	Get	ENUM	Holds the configured value, which will be written to attribute #5 after the module has been reset. Value (ENUM): String: Meaning: 00h “Auto” (default) 01h “MDI” 02h “MDIX”

Instance Attributes (Instance #17, MDI 2 Settings)

This instance holds the settings for MDI/MDIX 2. Changes have immediate effect.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“MDI 2”
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	ENUM	Value (ENUM): String: Meaning: 00h “Auto” (default) 01h “MDI” 02h “MDIX”
6	Configured Value	Get	ENUM	Holds the configured value, which will be written to attribute #5 after the module has been reset. Value (ENUM): String: Meaning: 00h “Auto” (default) 01h “MDI” 02h “MDIX”

Instance Attributes (Instances #18 and #19)

These instances are reserved for future attributes.

Instance Attributes (Instance #20, Modbus connection timeout)

This instance holds the settings for the Modbus connection timeout. Changes will be applied to new connections. Existing connections will use the previous timeout value.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Conn tmo” (Multilingual, see page *37*)
2	Data type	Get	UINT8	05h (= UINT16)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	UINT16	Value: Meaning (seconds): 0 Timeout disabled 60 Default
6	Configured Value	Get	UINT16	Holds the configured value, which will be written to attribute #5. Value: Meaning (seconds): 0 Timeout disabled 60 Default

Instance Attributes (Instance #21, Process active timeout)

This instance holds the settings for the Process active timeout. Changes have immediate effect. See *Communication Settings, p. 12* for more information.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Process tmo” (Multilingual, see page *37*)
2	Data type	Get	UINT8	05h (= uint16)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	UINT16	Default = 0 (milliseconds, disable timeout)
6	Configured Value	Get	UINT16	Holds the configured value, which will be written to attribute #5. Default = 0 (milliseconds, disable timeout)

Instance Attributes (Instance #22, Word order)

This instance holds the Word order settings. Value is used after module reset.

#	Name	Access	Data Type	Description
1	Name	Get	Array of CHAR	“Word order” (Multilingual, see page *37*)
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (read/write/shared access)
5	Value	Get/Set	UINT8	Value: Meaning: 0 Little endian (default) 1 Big endian Other values will be translated to 0 (default).
6	Configured Value	Get	UINT8	Holds the configured value, which will be written to attribute #5. Value: Meaning: 0 Little endian (default) 1 Big endian Other values will be translated to 0 (default).

Multilingual Strings

The instance names and enumeration strings in this object are multilingual, and are translated based on the current language settings as follows:

Instance	English	German	Spanish	Italian	French
3	IP address	IP-Adresse	Dirección IP	Indirizzo IP	Adresse IP
4	Subnet mask	Subnetzmaske	Masac. subred	Sottorete	Sous-réseau
5	Gateway	Gateway	Pasarela	Gateway	Passerelle
6	DHCP	DHCP	DHCP	DHCP	DHCP
	Enable	Einschalten	Activado	Abilitato	Activé
	Disable	Ausschalten	Desactivado	Disabilitato	Désactivé
7	Comm 1	Komm 1	Comu 1	Connessione 1	Comm 1
	Auto	Auto	Auto	Auto	Auto
	10 HDX	10 HDX	10 HDX	10 HDX	10 HDX
	10 FDX	10 FDX	10 FDX	10 FDX	10 FDX
	100 HDX	100 HDX	100 HDX	100 HDX	100 HDX
	100 FDX	100FDX	100 FDX	100 FDX	100 FDX
8	Comm 2	Komm 2	Comu 2	Connessione 2	Comm 2
	Auto	Auto	Auto	Auto	Auto
	10 HDX	10 HDX	10 HDX	10 HDX	10 HDX
	10 FDX	10 FDX	10 FDX	10 FDX	10 FDX
	100 HDX	100 HDX	100 HDX	100 HDX	100 HDX
	100 FDX	100FDX	100 FDX	100 FDX	100 FDX
9	DNS1	DNS 1	DNS Primaria	DNS1	DNS1
10	DNS2	DNS 2	DNS Secundia.	DNS2	DNS2
11	Host name	Host name	Nombre Host	Nome Host	Nom hôte
12	Domain name	Domain name	Nobre Domain	Nome Dominio	Dom Domaine
13	SMTP Server	SMTP Server	Servidor SMTP	Server SMTP	SMTP serveur
14	SMTP User	SMTP User	Usuario SMTP	Utente SMTP	SMTP utilisa.
15	SMTP Pswd	SMTP PSWD	Clave SMTP	Password SMTP	SMTP mt passe
20	Conn tmo	Verb. Tmo	Tout Conexion	Tout Conn.	Conn tmo
21	Process tmo	Prozess Tmo	Tout Proceso	Tout Processo	Process tmo
22	Word order	Wortfolge	Orden palabra	Ordine "word"	Ordre - mots

8.6 Anybus File System Interface Object (0Ah)

Category

Extended

Object Description

This object provides an interface to the built-in file system. Each instance represents a handle to a file stream and contains services for file system operations.

This provides the host application with access to the built-in file system of the module, e.g. when application specific web pages are to be installed.

Instances are created and deleted dynamically during runtime.

This object is thoroughly described in *Anybus CompactCom 40 Software Design Guide*.

8.7 Network Ethernet Object (0Ch)

Category

Extended

Object Description

This object provides Ethernet-specific information to the application.

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	“Network Ethernet”
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	-
4	Highest instance no.	Get	UINT16	-

Instance Attributes (Instance #1)

#	Name	Access	Data Type	Description
1	MAC Address	Get	Array of UINT8	Current MAC address. See also “Ethernet Host Object (F9h)”

9 Host Application Objects

9.1 General Information

This chapter specifies the host application object implementation in the module. The Application Data Object is mandatory to implement. The other objects listed here may optionally be implemented within the host application firmware to expand the implementation.

Standard Objects:

- Application Object (FFh) - (see Anybus CompactCom 40 Software Design Guide)
- Application Data Object (FEh) - (see Anybus CompactCom 40 Software Design Guide)

Network Specific Objects:

- *Ethernet Host Object (F9h), p. 44*
- *Modbus Host Object (FAh), p. 41*

9.2 Modbus Host Object (FAh)

Category

Extended

Object Description

This object implements Modbus related settings in the host application.

The implementation of this object is optional; the host application can support none, some, or all of the attributes specified below. The module will attempt to retrieve the values of these attributes during startup; if an attribute is not implemented in the host application, simply respond with an error message (06h, “Invalid CmdExt[0]”). In such case, the module will use its default value.

If the module attempts to retrieve a value of an attribute not listed below, respond with an error message (06h, “Invalid CmdExt[0]”).

See also ...

- Anybus CompactCom 40 Software Design Guide, “Error Codes”

Supported Commands

Object:	Get_Attribute Process-modbus-message
Instance:	Get_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	“Modbus”
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

Extended

Changes to these attributes during runtime will have no effect. If an attribute is not implemented the default value will be used.

#	Name	Access	Data Type	Default Value	Comment
1	Vendor name	Get	Array of CHAR	“HMS”	These settings will be returned in response to a “Read Device Identification” request. Attribute 2 and 3 will also be used as identification on the web site and for SHICP (IPconfig). The maximum allowed length of each string is 244 bytes; strings exceeding this length will be ignored and their default value will be used instead. See also ... • • *Read Device Identification, p. 19*
2	Product Code	Get	Array of CHAR	“Anybus CompactCom 40 Modbus-TCP”	
3	Major Minor Revision	Get	Array of CHAR	(firmware rev.)	
4	Vendor URL	Get	Array of CHAR	“ ”	
5	Product name	Get	Array of CHAR	“ ”	
6	Model name	Get	Array of CHAR	“ ”	
7	User Application Name	Get	Array of CHAR	“ ”	
8	Device ID	Get	Array of UINT8	-	Not used
9	No. of ADI indexing bits	Get	UINT8	04h	Value: Meaning: 00h each ADI = 1 Modbus register 01h each ADI = 2 Modbus registers 02h each ADI = 4 Modbus registers 03h each ADI = 8 Modbus registers 04h each ADI = 16 Modbus registers 05h each ADI = 32 Modbus registers 06h each ADI = 64 Modbus registers 07h each ADI = 128 Modbus registers (other) (invalid) (see *Application Data (ADIs), p. 13*)
10	Enable Modbus message forwarding	Get	Bool	False	If true, all Modbus messages, addressed (or broadcast) to this node are routed to the application.
11	Modbus read/write registers command offset	Get	SINT16[2]	[0x0000, 0x0000]	These values provides possibility to use offsets for the various read/write holding register commands Format: [READ, WRITE]

Command Details: Process-modbus-message

Category

Extended

Details

Command Code:	10h
Valid for:	Object

Description

If enabled, this command routes Modbus/TCP communication to the host application.

- Command Details

Field	Contents	Comments
CmdExt[0]	(reserved)	(ignore)
CmdExt[1]		
MsgData[0... n]	Modbus message frame (Query)	-

- Response Details

Field	Contents	Comments
CmdExt[0]	(reserved)	(set to zero)
CmdExt[1]		
MsgData[0... n]	Modbus message frame (Response)	-

ⓘ *The response data size must not exceed 254 bytes, if more data is returned, no Modbus response message will be sent to the originator of the request.*

If the response contains no data, no Modbus response will be sent to the originator of the request.

9.3 Ethernet Host Object (F9h)

Object Description

This object implements Ethernet features in the host application.

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Set_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	“Ethernet”
2	Revision	Get	UINT8	02h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

- If an attribute is not implemented, the default value will be used.
- The module is preprogrammed with a valid MAC address. To use that address, do not implement attribute #1.
- Do not implement attributes #9 and #10, only used for PROFINET devices, if the module shall use the preprogrammed MAC addresses.
- If new MAC addresses are assigned to a PROFINET device, these addresses (in attributes #1, #9, and #10) have to be consecutive, e.g. (xx:yy:zz:aa:bb:01), (xx:yy:zz:aa:bb:02), and (xx:yy:zz:aa:bb:03) with the first five octets not changing.

#	Name	Access	Data Type	Default Value	Comment
1	MAC address	Get	Array of UINT8	-	6 byte physical address value; overrides the preprogrammed Mac address. Note that the new Mac address value must be obtained from the IEEE. Do not implement this attribute if the preprogrammed Mac address is to be used.
2	Enable HICP	Get	BOOL	True (Enabled)	Enable/Disable HICP
3	Enable Web Server	Get	BOOL	True (Enabled)	Enable/Disable Web Server (Not used if Transparent Ethernet is enabled.)
4	(reserved)				Reserved for Anybus CompactCom 30 applications.
5	Enable Web ADI access	Get	BOOL	True (Enabled)	Enable/Disable Web ADI access (Not used if Transparent Ethernet is enabled.)
6	Enable FTP server	Get	BOOL	True (Enabled)	Enable/Disable FTP server (Not used if Transparent Ethernet is enabled.)
7	Enable admin mode	Get	BOOL	False (Disabled)	Enable/Disable FTP admin mode (Not used if Transparent Ethernet is enabled.)
8	Network Status	Set	UINT16	-	See below.

#	Name	Access	Data Type	Default Value	Comment
9	Port 1 MAC address	Get	Array of UINT8	-	**Note**: This attribute is only valid for PROFINET devices. 6 byte MAC address for port 1 (mandatory for the LLDP protocol). This setting overrides any Port MAC address in the host PROFINET IO Object. Do not implement this attribute if the preprogrammed Mac address is to be used.
10	Port 2 MAC address	Get	Array of UINT8	-	**Note**: This attribute is only valid for PROFINET devices. 6 byte MAC address for port 2 (mandatory for the LLDP protocol). This setting overrides any Port MAC address in the host PROFINET IO Object. Do not implement this attribute if the preprogrammed Mac address is to be used.
11	Enable ACD	Get	BOOL	True (Enabled)	Enable/Disable ACD protocol. If ACD functionality is disabled using this attribute, the ACD attributes in the CIP TCP/IP object (F5h) are not available.
12	Port 1 State	Get	ENUM	0 (Enabled)	The state of Ethernet port 1. • This attribute is not read by EtherCAT and Ethernet POWERLINK devices, where Port 1 is always enabled. 00h: Enabled 01h: Disabled. The port is treated as existing. References to the port can exist, e.g. in network protocol or on website.
13	Port 2 State	Get	ENUM	0 (Enabled)	The state of Ethernet port 2. • This attribute is not read by EtherCAT and Ethernet POWERLINK devices, where Port 2 is always enabled. 00h: Enabled 01h: Disabled. The port is treated as existing. References to the port can exist, e.g. in network protocol or on website. 02h: Inactive. The attribute is set to this value for a device that only has one physical port. All two-port functionality is disabled. No references can be made to this port. **Note:** This functionality is available for PROFINET, Ethernet/IP and Modbus-TCP devices.
14	(reserved)				
15	Enable reset from HICP	Get	BOOL	0 = False	Enables the option to reset the module from HICP.
16	IP configuration	Set	Struct of: UINT32 (IP address) UINT32 (Subnet mask) UINT32 (Gateway)	N/A	Whenever the configuration is assigned or changed, the Anybus CompactCom module will update this attribute.

#	Name	Access	Data Type	Default Value	Comment
17	IP address byte 0–2	Get	Array of UINT8 [3]	[0] = 192 [1] = 168 [2] = 0	First three bytes in IP address. Used in standalone shift register mode if the configuration switch value is set to 1-245. In that case the IP address will be set to: Y[0].Y[1].Y[2].X Where Y0-2 is configured by this attribute and the last byte X by the configuration switch.
18	Ethernet PHY Configuration	Get	Array of BITS16	0x0000 for each port	Ethernet PHY configuration bit field. The length of the array shall equal the number of Ethernet ports of the product. Each element represents the configuration of one Ethernet port (element #0 maps to Ethernet port #1, element #1 maps to Ethernet port #2 and so on). **Note**: Only valid for EtherNet/IP and Modbus-TCP devices. Bit 0: Auto negotiation fallback duplex 0 = Half duplex 1 = Full duplex Bit 1–15: Reserved
20	SNMP read-only community string	Get	Array of CHAR	“public”	**Note**: This attribute is only valid for PROFINET devices. Sets the SNMP read-only community string. Max length is 32.
21	SNMP read-write community string	Get	Array of CHAR	“private”	**Note**: This attribute is only valid for PROFINET devices. Sets the SNMP read-write community string. Max length is 32.
22	DHCP Option 61 source	Get	ENUM	0 (Disabled)	**Note**: This attribute is currently only valid for Ethernet/IP devices. See below (DHCP Option 61, Client Identifier)
23	DHCP Option 61 generic string	Get	Array of UINT8	N/A	**Note**: This attribute is currently only valid for Ethernet/IP devices. See below (DHCP Option 61, Client Identifier)
24	Enable DHCP Client	Get	BOOL	1 = True	**Note**: This attribute is currently valid for Ethernet/IP and PROFINET devices. Enable/disable DHCP Client functionality 0: DHCP Client functionality is disabled 1: DHCP Client functionality is enabled

Network Status

This attribute holds a bit field which indicates the overall network status as follows:

Bit	Contents	Description	Comment
0	Link	Current global link status 1= Link sensed 0= No link	
1	IP established	1 = IP address established 0 = IP address not established	
2	(reserved)	(mask off and ignore)	
3	Link port 1	Current link status for port 1 1 = Link sensed 0 = No link	EtherCAT only: This link status indicates whether the Anybus CompactCom is able to communicat using Ethernet over EtherCAT (EoE) or not. That is, it indicates the status of the logical EoE port link and is not related to the link status on the physical EtherCAT ports.
4	Link port 2	Current link status for port 2 1 = Link sensed 0 = No link	Not used for EtherCAT
5... 15	(reserved)	(mask off and ignore)	

DHCP Option 61 (Client Identifier)

ⓘ *Only valid for EtherNet/IP devices*

The DHCP Option 61 (Client Identifier) allow the end-user to specify a unique identifier, which has to be unique within the DHCP domain.

Attribute #22 (DHCP Option 61 source) is used to configure the source of the Client Identifier. The table below shows the definition for the Client identifier for different sources and their description.

Value	Source	Description
0	Disable	The DHCP Option 61 is disabled. This is the default value if the attribute is not implemented in the application.
1	MACID	The MACID will be used as the Client Identifier
2	Host Name	The configured Host Name will be used as the Client Identifier
3	Generic String	Attribute #23 will be used as the Client Identifier

Attribute #23 (DHCP Option 61 generic string) is used to set the Client Identifer when Attribute #22 has been set to 3 (Generic String). Attribute #23 contains the Type field and Client Identifier and shall comply with the definitions in RFC 2132. The allowed max length that can be passed to the module via attribute #23 is 64 octets.

Example:

If Attribute #22 has been set to 3 (Generic String) and Attribute #23 contains 0x01, 0x00, 0x30, 0x11, 0x33, 0x44, 0x55, the Client Identifier will be represented as an Ethernet Media Type with MACID 00:30:11:33:44:55.

Example 2:

If Attribute #22 has been set to 2 (Host Name) Attribute #23 will be ignored and the Client Identifier will be the same as the configured Host Name.

This page intentionally left blank

A Categorization of Functionality

The objects, including attributes and services, of the Anybus CompactCom and the application are divided into two categories: basic and extended.

A.1 Basic

This category includes objects, attributes and services that are mandatory to implement or to use. They will be enough for starting up the Anybus CompactCom and sending/receiving data with the chosen network protocol. The basic functions of the industrial network are used.

Additional objects etc, that will make it possible to certify the product also belong to this category.

A.2 Extended

Use of the objects in this category extends the functionality of the application. Access is given to the more specific characteristics of the industrial network, not only the basic moving of data to and from the network. Extra value is given to the application.

Some of the functionality offered may be specialized and/or seldom used. As most of the available network functionality is enabled and accessible, access to the specification of the industrial network may be required.

B Implementation Details

B.1 SUP-Bit Definition

The supervised bit (SUP) indicates that the network participation is supervised by another network device.

This bit is set when in PROCESS_ACTIVE, and only if the Process active timeout value is greater than zero (0).

B.2 Anybus State Machine

The table below describes how the Anybus state machine relates to the Modbus-TCP network

Anybus State	Implementation	Comment
WAIT_PROCESS	Waiting for Modbus requests. The module shifts to PROCESS_ACTIVE when a Modbus request is received.	-
ERROR	IP address conflict.	This state is only possible if Address Conflict Detection (ACD) is enabled in the Ethernet Host object (enabled by default).
PROCESS_ACTIVE	The module shifts to WAIT_PROCESS if no requests are received within the time stated by Process Active Timeout (see Instance #21 in *Network Configuration Object (04h), p. 28*.	-
IDLE	The IDLE state can be entered/exited by writing to the Modbus Enter/Exit idle state register at address 1004h.	
EXCEPTION	Any Modbus requests will be ignored.	

B.3 Application Watchdog Timeout Handling

Upon detection of an application watchdog timeout, the module will cease network participation and shift to state EXCEPTION. No other network specific actions are performed.

C Secure HICP (Secure Host IP Configuration Protocol)

C.1 General

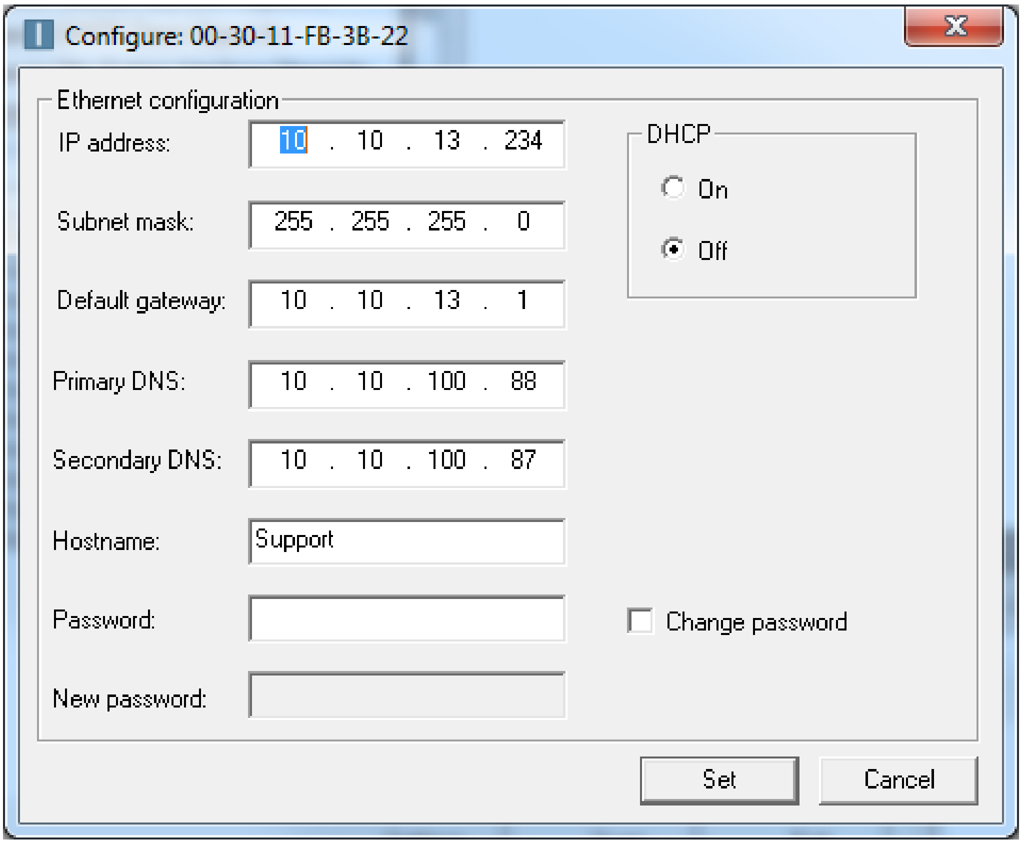
The Anybus CompactCom 40 Modbus-TCP supports the Secure HICP protocol used by the Anybus IPconfig utility for changing settings, e.g. IP address, Subnet mask, and enable/disable DHCP. Anybus IPconfig can be downloaded free of charge from the HMS website, www.anybus.com. This utility may be used to access the network settings of any Anybus product connected to the network via UDP port 3250.

The protocol offers secure authentication and the ability to restart/reboot the device(s).

C.2 Operation

When the application is started, the network is automatically scanned for Anybus products. The network can be rescanned at any time by clicking **Scan**.

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



Configure: 00-30-11-FB-3B-22

Ethernet configuration

IP address: 10 . 10 . 13 . 234

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 10 . 10 . 13 . 1

Primary DNS: 10 . 10 . 100 . 88

Secondary DNS: 10 . 10 . 100 . 87

Hostname: Support

Password:

New password:

DHCP

○ On

● Off

□ Change password

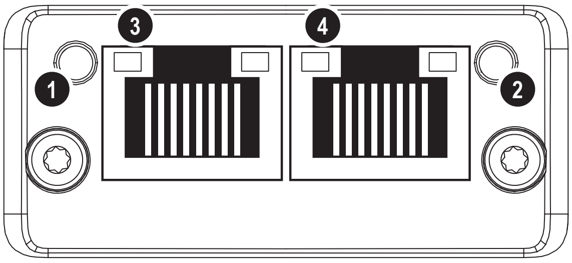
Fig. 2

Validate the new settings by clicking **Set**, or click **Cancel** to cancel all changes. Optionally, the configuration can be protected from unauthorized access by a password. To enter a password, check the **Change password** checkbox and enter the password in the **New password** text field.

D Technical Specification

D.1 Front View

D.1.1 Front View (Ethernet Connectors)

#	Item	Connector	
1	Network Status LED	Ethernet, 45	
2	Module Status LED		
3	Link/Activity LED (port 1)		
4	Link/Activity LED (port 2)		

Test sequences are performed on the Network and Module Status LEDs during startup.

D.1.2 Network Status LED

LED State	Description
Off	No IP address or in state EXCEPTION
Green	At least one Modbus message received
Green, flashing	Waiting for first Modbus message
Red	IP address conflict detected, FATAL ERROR
Red, flashing	Connection timeout. No Modbus message has been received within the configured “process active timeout” time

A test sequence is performed on this LED during startup.

D.1.3 Module Status LED

LED State	Description
Off	No power
Green	Normal operation
Red	Major fault (including Anybus exception), FATAL
Red, flashing	Minor fault
Alternating red/green	Firmware update from file system in progress

A test sequence is performed on this LED during startup.

D.1.4 LINK/Activity LED 3/4

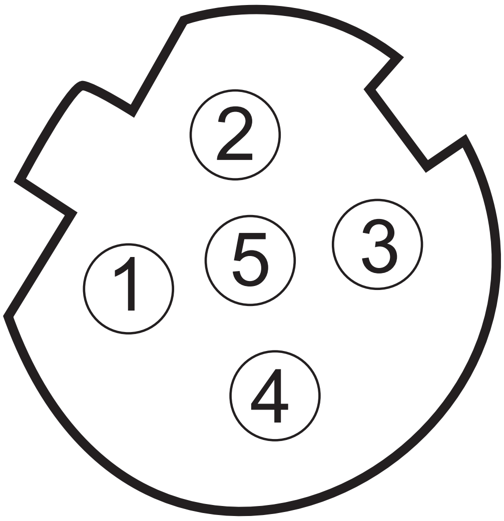
LED State	Description
Off	No link, no activity
Green	Link (100 Mbit/s) established
Green, flickering	Activity (100 Mbit/s)
Yellow	Link (10 Mbit/s) established
Yellow, flickering	Activity (10 Mbit/s)

D.1.5 Fatal Error

If both the Network Status LED and the Module Status LED are red, a fatal error has occurred.

D.1.6 Ethernet Interface

The Ethernet interface 10/100Mbit, full or half duplex operation.

Pin	Name	Description	
1	TXD+	Transmit positive	
2	RXD+	Receive positive	
3	TXD-	Transmit negative	
4	RXD-	Receive negative	
5 (Thread)	Shield	Shield	

D.2 Functional Earth (FE) Requirements

In order to ensure proper EMC behavior, the module must be properly connected to functional earth via the FE pad/FE mechanism described in the *Anybus CompactCom 40 Hardware Design Guide*. Proper EMC behavior is not guaranteed unless these FE requirements are fulfilled.

D.3 Power Supply

D.3.1 Supply Voltage

The Anybus CompactCom 40 Modbus-TCP requires a regulated 3.3 V power source as specified in the general *Anybus CompactCom 40 Hardware Design Guide*.

D.3.2 Power Consumption

TheAnybus CompactCom 40 Modbus-TCP is designed to fulfil the requirements of a Class B module.

In line with HMS policy of continuous product development, we reserve the right to change the exact power requirements of this product without prior notification. However, in any case, the Anybus CompactCom 40 Modbus-TCP will remain as a Class B module.

For more information about the power consumption classification used on the Anybus CompactCom 40 platform, consult the general *Anybus CompactCom 40 Hardware Design Guide*.

> ℹ *It is strongly advised to design the power supply in the host application based on the power consumption classifications described in the general Anybus CompactCom 40 Hardware Design Guide, and not on the exact power requirements of a single product.*

D.4 Environmental Specification

Consult the *Anybus CompactCom 40 Hardware Design Guide* for further information.

D.5 EMC Compliance

Consult the *Anybus CompactCom 40 Hardware Design Guide* for further information.

E Backward Compatibility

The Anybus CompactCom M40 series of industrial network modules have significantly better performance and include more functionality than the modules in the Anybus CompactCom 30 series. The 40 series is backward compatible with the 30 series in that an application developed for the 30 series should be possible to use with the 40 series, without any major changes. Also it is possible to mix 30 and 40 series modules in the same application.

This appendix presents the backwards compatibility issues that have to be considered for Anybus CompactCom 40 Modbus-TCP, when designing with both series in one application, or when adapting a 30 series application for the 40 series.

E.1 Initial Considerations

There are two options to consider when starting the work to modify a host application developed for Anybus CompactCom 30-series modules to also be compatible with the 40-series modules:

- Add support with as little work as possible i.e. reuse as much as possible of the current design.
  - This is the fastest and easiest solution but with the drawback that many of the new features available in the 40-series will not be enabled (e.g. enhanced and faster communication interfaces, larger memory areas, and faster communication protocols).
  - You have to check the hardware and software differences below to make sure the host application is compatible with the 40-series modules. Small modifications to your current design may be needed.
- Make a redesign and take advantage of all new features presented in the 40-series.
  - A new driver and host application example code are available at www.anybus.com/starterkit40 to support the new communication protocol. This driver supports both 30-series and 40-series modules.
  - You have to check the hardware differences below and make sure the host application is compatible with the 40-series modules.

This information only deals with differences between the 30-series and the 40-series.

Link to support page: www.anybus.com/support.

E.2 Hardware Compatibility

Anybus CompactCom is available in three hardware formats; Module, Chip, and Brick.

E.2.1 Module

The modules in the 30-series and the 40-series share physical characteristics, like dimensions, outline, connectors, LED indicators, mounting parts etc. They are also available as modules without housing.

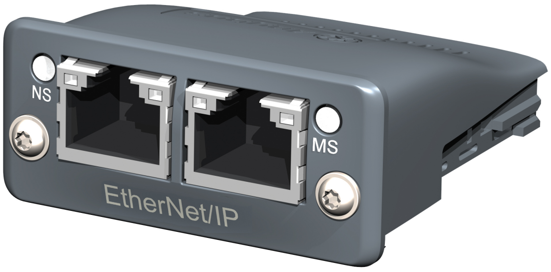


Fig. 3 Anybus CompactCom M30/M40

E.2.2 Chip

The chip (C30/C40) versions of the Anybus CompactCom differ completely when it comes to physical dimensions.

! There is no way to migrate a chip solution from the 30-series to the 40-series without a major hardware update.

E.2.3 Brick

The Anybus CompactCom B40-1 does not share dimensions with the Anybus CompactCom B30. The B40-1 is thus not suitable for migration. However HMS Industrial Networks has developed a separate brick version in the 40-series, that can be used for migration. This product, B40-2, shares dimensions etc. with the B30. Please contact HMS Industrial Networks for more information on the Anybus CompactCom B40-2.

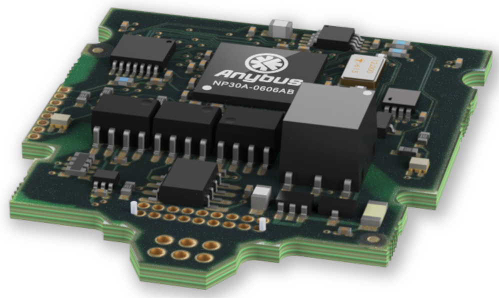


Fig. 4 Anybus CompactCom B30

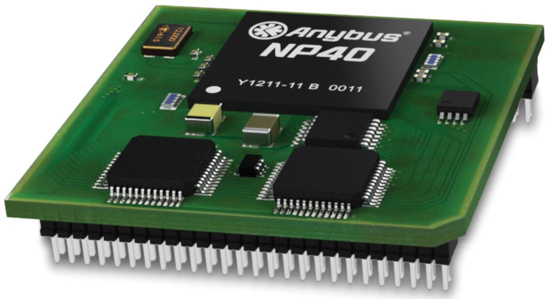


Fig. 5 Anybus CompactCom B40–1 (not for migration)

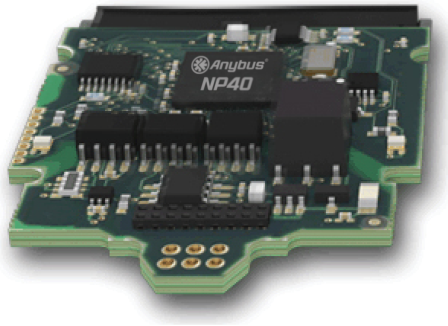


Fig. 6 Anybus CompactCom B40–2

E.2.4 Host Application Interface

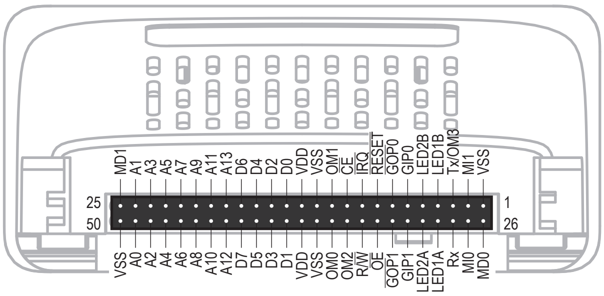


Fig. 7

Some signals in the host application interface have modified functionality and/or functions which must be checked for compatibility. See the following sections.

Tx/OM3

In the 30-series, this pin is only used for Tx. It is tri-stated during power up, and driven by the Anybus CompactCom UART after initialization. In the 40-series this pin is used as a fourth operating mode setting pin (OM3). During startup after releasing the reset, this pin is read to determine the operating mode to use. The pin is then changed to a Tx output.

In the 40-series, this pin has a built-in weak pull-up. If this pin, on a 30-series module or brick is unconnected, pulled high, or connected to a high-Z digital input on the host processor, it will be compatible with the 40-series. An external pull-up is recommended, but not required.

! If this pin is pulled low by the host during startup in a 30-series application, any 40-series module or brick, substituted in the application, will not enter the expected operating mode.

Related Information: Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126), Section “Application Connector Pin Overview”

Module Identification (MI[0..1])

These pins are used by the host application (i.e. your product) to identify what type of Anybus CompactCom that is mounted. The identification differs between the 30-series and the 40-series.

If your software use this identification you need to handle the new identification value.

MI1	MI0	Module Type
LOW	LOW	Active Anybus CompactCom 30
HIGH	LOW	Active Anybus CompactCom 40

MI[0..1] shall only be sampled by the application during the time period from power up to the end of SETUP state. The pins are low at power up and before reset release.

Related Information: *Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126)*, Section “Settings/Sync”.

GIP[0..1]/LED3[A..B]

These pins are tri-stated inputs by default in the 30-series. In the 40-series, these pins are tri-stated until the state NW_INIT. After that they become open-drain, active low LED outputs (LED3A/LED3B).

No modification of the hardware is needed, if your current design has

- tied these pins to GND
- pulled up the pins
- pulled down the pins
- left the pins unconnected

However, if the application drive the pins high, a short circuit will occur.

If you connect the pins to LEDs, a pull-up is required.

In the 40-series, there is a possibility to set the GIP[0..1] and GOP[0..1] in high impedance state (tri-state) by using attribute #16 (GPIO configuration) in the Anybus object (01h). I.e. if it is not possible to change the host application hardware, this attribute can be configured for high impedance state of GIP and GOP before leaving NW_INIT state.

Related Information: *Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126)*, Section “LED Interface/D8-D15 (Data Bus)”.

GOP[0..1]/LED4[A..B]

These pins are outputs (high state) by default in the 30-series. In the 40-series, these pins are tri-stated until the state NW_INIT, and after that they become push-pull, active low LED outputs (LED4A/LED4B).

This change should not affect your product.

Related Information: *Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126)*, Section 3.2.3, “LED Interface/D8-D15 (Data Bus)”.

Address Pins A[11..13]

The address pins 11, 12, and 13 are ignored by the 30-series. These pins must be high when accessing the 40-series module in backwards compatible 8-bit parallel mode. If you have left these pins unconnected or connected to GND, you need to make a hardware modification to tie them high.

Max Input Signal Level (V_{IH})

The max input signal level for the 30-series is specified as V_{IH}=V_{DD}+0,2 V, and for the 40-series as V_{IH}=3.45 V. Make sure that you do not exceed 3.45 V for a logic high level.

RMII Compatibility

If the RMII mode is being used on an Anybus CompactCom 40 module and it is desired to remain compatible with the 30 series, it is important to disable this connection when switching to an Anybus CompactCom 30 module due to pin conflicts. The RMII port of the host processor should be set to tristate by default, and only be enabled if an RMII capable Anybus CompactCom 40 is detected. In case the RMII connection cannot be disabled through an internal hardware control on the host processor, it will be necessary to design in external hardware (i.e. a FET bus switch) to prevent short circuits

Related Information: *Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126)*, Section 3.2.5, “RMII — Reduced Media-Independent Interface”.

E.3 General Software

E.3.1 Extended Memory Areas

The memory areas have been extended in the 40-series, and it is now possible to access larger sizes of process data (up to 4096 bytes instead of former maximum 256 bytes) and message data (up to 1524 bytes instead of former maximum 255 bytes). The 30-series has reserved memory ranges that the application should not use. The 40-series implements new functionality in some of these memory areas.

ⓘ *To use the extended memory areas you need to implement a new communication protocol which is not part of this document.*

Memory areas not supported by the specific network cannot be used. Make sure you do not access these areas, e.g. for doing read/write memory tests.

Related Information: *Anybus CompactCom 40 Software Design Guide (HMSI-216-125)*, Section “Memory Map”

E.3.2 Faster Ping-Pong Protocol

The ping-pong protocol (the protocol used in the 30-series) is faster in the 40-series. A 30-series module typically responds to a so called ping within 10-100 µs. The 40-series typically responds to a ping within 2 µs.

Interrupt-driven applications (parallel operating mode) may see increased CPU load due to the increased speed.

E.3.3 Requests from Anybus CompactCom to Host Application During Startup

All requests to software objects in the host application must be handled and responded to (even if the object does not exist). This applies for both the 30-series and the 40-series. The 40-series introduces additional objects for new functionality.

There may also be additional commands in existing objects added to the 40-series that must be responded to (even if it is not supported).

If your implementation already responds to all commands it cannot process, which is the expected behavior, you do not need to change anything.

E.3.4 Anybus Object (01h)

Attribute	30-series	40-series	Change/Action/Comment
#1, Module Type	0401h	0403h	Make sure the host application accepts the new module type value for the 40-series.
#15, Auxiliary Bit	Available	Removed	It is not possible to turn off the “Changed Data Indication” in the 40-series. Also see “Control Register CTRL_AUX-bit” and “Status Register STAT_AUX-bit” below.
#16, GPIO Configuration	Default: General input and output pins	Default: LED3 and LED4 outputs	See also .. • *GIP[0..1]/LED3[A..B], p. 59* • *GOP[0..1]/LED4[A..B], p. 59*

E.3.5 Control Register CTRL_AUX-bit

30-series The CTRL_AUX bit in the control register indicates to the Anybus CompactCom if the process data in the current telegram has changed compared to the previous one.

40-series The value of the CTRL_AUX bit is always ignored. Process data is always accepted.

All released Anybus CompactCom 30 example drivers from Anybus CompactCom comply with this difference.

Related Information: *Anybus CompactCom 40 Software Design Guide (HMSI-216-125)*, section “Control Register”.

E.3.6 Status Register STAT_AUX-bit

30-series The STAT_AUX bit in the status register indicates if the output process data in the current telegram has changed compared to the previous one. This functionality must be enabled in the Anybus object (01h), Attribute #15. By default, the STAT_AUX bit functionality is disabled.

40-series The STAT_AUX bit indicates updated output process data (not necessarily changed data) from the network compared to the previous telegram. The functionality is always enabled.

All released Anybus CompactCom 30 example drivers from HMS Industrial Networks comply with this difference.

Related Information: *Anybus CompactCom 40 Software Design Guide (HMSI-216-125)*, section “Status Register”.

E.3.7 Control Register CTRL_R-bit

30-series The application may change this bit at any time.

40-series For the 8-bit parallel operating mode, the bit is only allowed to transition from 1 to 0 when the STAT_M-bit is set in the status register. When using the serial operating modes, it is also allowed to transition from 1 to 0 in the telegram immediately after the finalizing empty fragment.

All released Anybus CompactCom 30 example drivers from HMS Industrial Networks comply with this difference.

Related Information: *Anybus CompactCom 40 Software Design Guide (HMSI-216-125)*, section “Control Register”.

E.3.8 Modifications of Status Register, Process Data Read Area, and Message Data Read Area

In the 40-series, the Status Register, the Process Data Read Area, and the Message Data Read Area are write protected in hardware (parallel interface). If the software for some reason writes to any of those areas, a change is needed.

All releasedAnybus CompactCom 30 example drivers from HMS Industrial Networks comply with this difference.

E.4 Network Specific — Modbus-TCP

E.4.1 Modbus Registers

Rearrangements have been made in the Modbus register map, because process data sizes have been increased to 1536 bytes in each direction. An existing PLC configuration need to be changed to use the new addresses. **No difference on the application side.**

Contents	30-series Modbus Address	40-series Modbus Address
Holding Registers (4x)		
Read Process Data	0000h-00FFh	0000h-02FFh
Write Process Data	0100h-01FFh	0800h-0AFFh
Process Active Timeout	0203h	1003h
Enter/Exit Idle Mode	0204h	1004h
ADI Number 1	0210h-021Fh	1010h-101Fh
ADI Number 2	0220h-022Fh	1020h-102Fh
ADI Number 3839		FFF0h-FFFFh
Input Registers (3x)		
Write Process Data	0000h-00FFh	0000h-02FFh
Diagnostic Event Count	0100h	0800h
Diagnostic Event #1	0101h	0801h
Diagnostic Event #2	0102h	0802h
Diagnostic Event #3	0103h	0803h
Diagnostic Event #4	0104h	0804h
Diagnostic Event #5	0105h	0805h
Diagnostic Event #6	0106h	0806h
Coils (0x)		
Read Process Data	0000h-0FFFh	0000h-2FFFh
Discrete Inputs (1x)		
Write Process Data	0000h-0FFFh	0000h-2FFFh

E.4.2 BOOL arrays

Process data mapped BOOL arrays are not compressed to bit-fields on the network in the 40-series, but handled as a normal 8-bit datatype. To create bit-arrays in the 40-series, use the new datatypes BITx instead.

E.4.3 Network Configuration Object (04h)

The instances in the Network Configuration Object have been rearranged for the Ethernet based modules for consistency. Network specific instances are moved to instance number 20 and onwards. This is done to increase the number of instances in the section that is not network specific.

If the host application is using any of the parameters below, the software must be updated to use the new instance numbers.

Parameter Name	30-series Instance #	40-series Instance #
Modbus Connection Timeout	9	20
Process Active Timeout	10	21
DNS1	11	9
DNS2	12	10
Host Name	13	11
Domain Name	14	12
SMTP Server	15	13
SMTP User	16	14
SMTP Password	17	15
Word Order	18	22

E.4.4 Modbus Host Object (FAh)

Attribute	30-series	40-series	Change/Action/Comment
#2, Product Code	Default: “Anybus-CC Modbus-TCP (2-Port)”	Default: “Anybus CompactCom 40 Modbus TCP”	If the attribute is implemented in the host application, it overrides the default value and there is no difference between the 30-series and the 40-series. If the attribute is not implemented, the default value is used.
#11,Modbus read/write registers command offset	-	-	In the 30-series, this register address offset is only applied when accessing holding registers with the command Read/Write Multiple registers (23). The 40-series applies this register offset to all holding register access, i.e. commands 3, 6, 16 and 23.

E.4.5 Ethernet Host Object (F9h)

Attribute	30-series	40-series	Change/Action/Comment
#4, Enable Modbus-TCP	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. Nothing needs to be changed in the host application.

E.4.6 Process data

In the 30-series modules, writing to the ADI register area would only result in a Set_Attribute command to the application (Application Data Object (FEh)) if the ADI was not mapped to read process data. For the 40-series, all register writes to the ADI area also results in a corresponding Set_Attribute command to the host application (Application Data Object (FEh)), as well as updating of the process data.

F License Information

Copyright 2013 jQuery Foundation and other contributors

http://jquery.com/

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

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

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

**-
*

rsvp.js

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

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

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

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

**-
*

libb (big.js)

The MIT Expat Licence.

Copyright (c) 2012 Michael Mclaughlin

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

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

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

***-
*

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

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

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

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

***-
*

Copyright (c) 2002 Florian Schulze.

All rights reserved.

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

1. Redistributions of source code must retain the above copyright

notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright

notice, this list of conditions and the following disclaimer in the

documentation and/or other materials provided with the distribution.

3. Neither the name of the authors nor the names of the contributors

may be used to endorse or promote products derived from this software

without specific prior written permission.

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

tpd.c - This file is part of the FTP daemon for lwIP

***-
*

Format - lightweight string formatting library.

Copyright (C) 2010-2013, Neil Johnson

All rights reserved.

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

* Redistributions of source code must retain the above copyright notice,

this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice,

this list of conditions and the following disclaimer in the

documentation and/or other materials provided with the distribution.

* Neither the name of nor the names of its contributors

may be used to endorse or promote products derived from this software

without specific prior written permission.

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

****

Print formatting routines

Copyright (C) 2002 Michael Ringgaard. All rights reserved.

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

1. Redistributions of source code must retain the above copyright

notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright

notice, this list of conditions and the following disclaimer in the

documentation and/or other materials provided with the distribution.

3. Neither the name of the project nor the names of its contributors

may be used to endorse or promote products derived from this software

without specific prior written permission.

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

***-
*

lwIP is licenced under the BSD licence:

Copyright (c) 2001-2004 Swedish Institute of Computer Science.

All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice,

this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,

this list of conditions and the following disclaimer in the documentation

and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products

derived from this software without specific prior written permission.

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

***-
*

MD5 routines

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

This software is provided “as-is”, without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software. Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not

claim that you wrote the original software. If you use this software

in a product, an acknowledgment in the product documentation would be

appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be

misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

L. Peter Deutsch

ghost@aladdin.com

Box 4126
300 04 Halmstad, Sweden

info@hms.se