Anybus® CompactCom™ 40 Modbus Serial

CANopen

NETWORK GUIDE

SCM-1202-166 1.0 en-US ENGLISH

Important User Information

Disclaimer

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

HMS 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 Networks and is subject to change without notice. HMS 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 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 Networks will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features or functional side effects found outside the documented scope of the product. The effects caused by any direct or indirect use of such aspects of the product are undefined and may include e.g. compatibility issues and stability issues.

Table of Contents

Page

1 Preface 3

1.1 About this Document 3
1.2 Document History 3
1.3 Document Conventions 3
1.4 Document Specific Conventions 4
1.5 Trademarks 4

2 About the Anybus CompactCom 40 Modbus Serial - CANopen 5

2.1 General Information 5
2.2 Features (CANopen) 6
2.3 Overview 6

3 Basic Operation 7

3.1 Electronic Data Sheet (EDS) 7
3.2 Network Identity 7
3.3 Startup and Identity Customization 8
3.4 Node Address & Data Rate Configuration 12
3.5 Data Exchange 13

4 Object Dictionary (CANopen) 16

4.1 Standard Objects 16
4.2 Manufacturer and Profile Specific Objects 17

A LED Indications 19

A.1 RUN LED 19
A.2 ERROR LED 19

B Conformance Test Guide 20

B.1 Introduction 20
B.2 Fieldbus Conformance Notes 20
B.3 Certification 20

This page intentionally left blank

1 Preface

1.1 About this Document

This network guide is intended to provide a good understanding of the functionality offered by the Anybus CompactCom 40 Modbus Serial - CANopen .

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, along with the Anybus CompactCom B40 Modbus Serial user manual should normally be sufficient to implement a design. However, if advanced CANopen specific functionality is required for the network interface of the device, in-depth knowledge of CANopen networking internals and/or information from the official CANopen specifications may be required. In such cases, the persons responsible for the implementation of this product should either obtain the CANopen specification to gain sufficient knowledge or limit their implementation in such a way that this is not necessary.

For additional information, please visit the support website at www.anybus.com/support.

1.2 Document History

Version	Date	Description
1.0	2020-08-31	First release

1.3 Document Conventions

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

1. First do this
2. Then do this

Bulleted lists are used for:

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

► An action

→ and a result

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

```
Program code and script examples
```

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

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

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

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

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

Additional information which may facilitate installation and/or operation.

1.4 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.
- All dimensions in this document have a tolerance of ±0.10 mm unless otherwise stated.
- Outputs are TTL compliant unless otherwise stated.
- Signals which are “pulled to GND” are connected to GND via a resistor.
- Signals which are “pulled to 3V3” are connected to 3V3 via a resistor.
- Signals which are “tied to GND” are directly connected to GND,
- Signals which are “tied to 3V3” are directly connected to 3V3.

1.4.1 Pin Types

The pin types of the connectors are defined in the table below. The pin type may be different depending on which mode is used.

Pin type	Definition
I	Input
O	Output
I/O	Input/Output (bidirectional)
OD	Open Drain
Power	Pin connected directly to module power supply, GND or 3V3

1.5 Trademarks

2 About the Anybus CompactCom 40 Modbus Serial - CANopen

2.1 General Information

The Anybus CompactCom 40 Modbus Serial - CANopen is a communication solution for simple industrial field devices. The host application communicates with the product using the Modbus RTU protocol. The Anybus CompactCom 40 Modbus Serial - CANopen then communicates the data to the network. Typical applications are basic level I/O blocks, temperature controllers, measuring devices, and sensors.

The Anybus CompactCom 40 Modbus Serial - CANopen software interface is designed to be network protocol independent, making it possible to support several networking systems using the same application software code/driver.

The Anybus CompactCom 40 Modbus Serial - CANopen share footprint and electrical interface with the other members of the product family, independent of fieldbus or network. The host application connector provides an interface between the host application (Modbus RTU) and the Anybus CompactCom, while the network connector provides access to the chosen network. The Anybus CompactCom acts as a Modbus RTU slave on the host application side.

The Anybus CompactCom 40 family offers a wide range of functionality. For advanced products and applications, we recommend the standard Anybus CompactCom 40.

For general information about other products using the Anybus CompactCom 40 platform, consult www.anybus.com/support.

In a domestic environment, this product may cause radio interference in which case the user may be required to take adequate measures.

This product contains ESD (Electrostatic Discharge) sensitive parts that may be damaged if ESD control procedures are not followed. Static control precautions are required when handling the product. Failure to observe this may cause damage to the product.

2.2 Features (CANopen)

- CiA® 301 version 4.2.0 compliant
- Automatic baud rate detection
- Supports LSS (CANopen Layer Setting Service), according to CiA 305, 3.0.0
- Customizable Identity Information
- Up to 64 TPDO's & 64 RPDO's (Corresponds to a total of 512 bytes of Process Data in each direction)
- Heartbeat functionality supported (Node Guarding not supported)
- Supports Expedited- and Segmented SDO Transfer (Block Transfer not supported)
- Galvanic isolation between the host application and the industrial network available if used with the CompactCom B40 connector board

> **!** All Anybus CompactCom 40 Modbus Serial, where the host is running an example application, will be precertified for network conformance. This is done to ensure that the final product can be certified, but it does not necessarily mean that the final product does not require recertification. Contact HMS Networks for further information.

2.3 Overview

The picture below shows the data flow in the Anybus CompactCom 40 Modbus Serial - CANopen. The Modbus master sets up the Modbus RTU communication, and the Anybus CompactCom maps the process data to the industrial network/fieldbus.

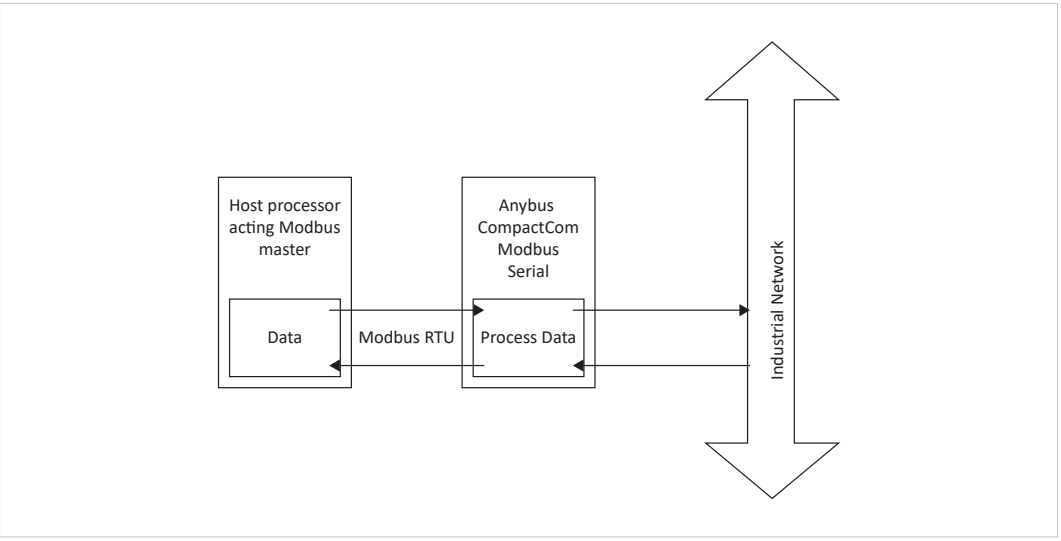


Fig. 1

3 Basic Operation

3.1 Electronic Data Sheet (EDS)

Each device on CANopen is associated with an Electronic Data Sheet (an EDS file), which holds a description of the device and its functions. Most importantly, the file describes the object dictionary implementation in the module.

HMS Networks supplies a generic EDS file which can serve as a basis for new implementations; however this file must be altered to match the end product (process data configuration, identity settings etc.). Process data must be described as specified in the CANopen standard “DS306 Electronic data sheet specification for CANopen” (can be requested from the CiA home page, www.can-cia.org).

To verify the EDS-file, download and run the EDS-file checker program from www.can-cia.org.

3.2 Network Identity

By default, the module uses the following identity settings:

Vendor ID:	0000001Bh (HMS Networks)
Device Type:	00000000h (Generic Device)
Product Code:	000Dh (Anybus CompactCom 40 CANopen)
Product Name:	“Anybus CompactCom 40 CANopen”

Optionally, it is possible to customize the identity of the module.

See also...

- *Startup and Identity Customization, p. 8*

3.3 Startup and Identity Customization

To customize the identity of the Anybus CompactCom (e.g. Vendor ID, Product Code, etc.), Virtual Attributes are used.

The most common customizations will be described here. For more detailed information, see the related documents listed in the beginning of this document.

Setting up the virtual attributes in the Anybus CompactCom can be accomplished in two different ways.

- Using the user-defined Modbus function code (Function code 70).

  The use of Function code 70 can be included in the Modbus master. Hence the CompactCom does not need to be preprogrammed before mounting it in the host application.

- Using the Anybus Virtual Attributes Manager.

  The Virtual Attributes Manager is recommended for use during development and for low volume production, since manual user operations are needed for every Anybus CompactCom that shall be programmed.

Once the virtual attributes are written to the Anybus CompactCom, they are saved in non-volatile memory. It is not necessary to write the virtual attributes at each startup.

3.3.1 Virtual Attributes with Specific Modbus Function Code 70

With Modbus function code 70, the Modbus master has access to the Anybus CompactCom internal messaging protocol. This means that all attributes within the Anybus CompactCom are potentially accessible.

When writing the virtual attributes to the Anybus CompactCom, the Anybus object, Object 01h, Instance 1, Attribute 17 is used. All information relevant for the basic virtual attributes will be covered here. For more information, refer to the related documents section in this document.

The example shows example values to the basic virtual attributes:

Virtual Attribute	Example Value
Vendor ID:	0x0000001B
Product Code:	0x0000000D
Major Revision:	1
Minor Revision:	2
Serial Number:	0x12345678
Product Name:	Product Name
Firmware Version:	1.2.3
Hardware Version:	3

To set the virtual attributes in the Anybus CompactCom to these values, using the Modbus function 70, create the request below:

Modbus function 70 Request

	Value	Note
Modbus Address	0xXX	
Function Code	0x46	FC70
Command	0x42	Set_Attribute
Object	0x01	Anybus Object
Instance	0x01	
	0x00	
Ext0	0x11	Attribute 17
Ext1	0x00	Not used
Data Size	0x51	The data size in this example is 81 bytes
	0x00	
Data	0xFB 0x01 0x00 0x01 0x04 0x00 0x1B 0x00 0x00 0x00 0xFB 0x01 0x00 0x02 0x04 0x00 0x0D 0x00 0x00 0x00 0xFB 0x01 0x00 0x03 0x02 0x00 0x01 0x00 0xFB 0x01 0x00 0x04 0x02 0x00 0x02 0x00 0xFF 0x01 0x00 0x03 0x04 0x00 0x78 0x56 0x34 0x120xFF 0x01 0x00 0x09 0x0C 0x00 0x50 0x72 0x6F 0x64 0x75 0x63 0x74 0x20 0x4E 0x61 0x6D 0x65 0xFF 0x01 0x00 0x0A 0x03 0x00 0x01 0x02 0x03 0xFF 0x01 0x00 0x0B 0x02 0x00 0x03 0x00	Vendor ID Product Code Major Rev. Minor Rev. Serial Number Product Name Firmware Ver. Hardware Ver.
CRC	0xXX	CRC-16
	0xXX	

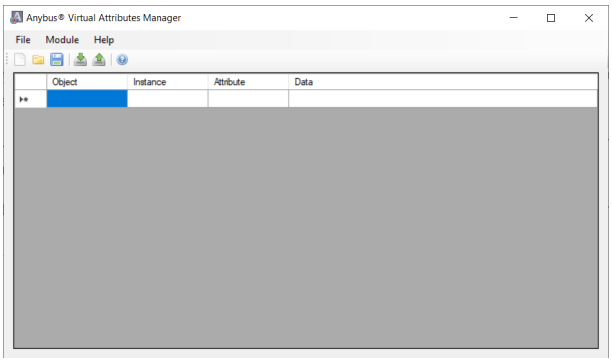
Response

	Value	Note
Modbus Address	0xXX	
Function Code	0x46	FC70
Command	0x02	Set_Attr_Resp
Object	0x01	Anybus Object
Instance	0x01	
	0x00	
Ext0	0x11	Attribute 17
Ext1	0x00	Not used
Data Size	0x00	
	0x00	
CRC	0xXX	CRC-16
	0xXX	

Requests with a size larger than 244 bytes will return Modbus exception code ILLEGAL DATA VALUE.

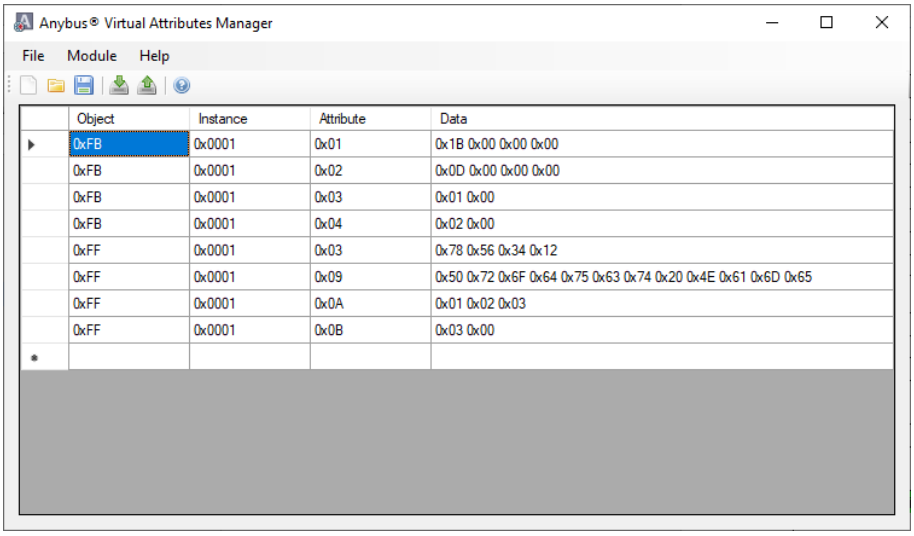
3.3.2 Virtual Attributes with Anybus Virtual Attributes Manager

1. Start the Anybus Virtual Attributes Manager



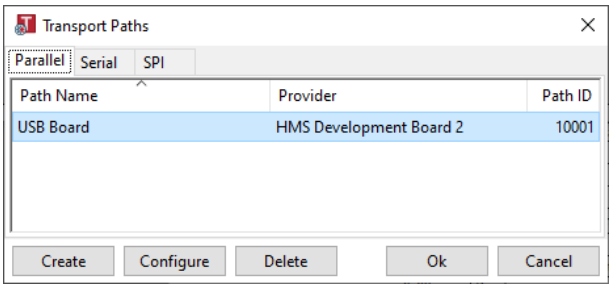
2. Enter the virtual attributes data for the attributes needed. The example below is setting up the attributes with the following values:

Virtual Attribute	Example Value
Vendor ID:	0x0000001B
Product Code:	0x0000000D
Major Revision:	1
Minor Revision:	2
Serial Number:	0x12345678
Product Name:	Product Name
Firmware Version:	1.2.3
Hardware Version:	3



3. Mount the Anybus CompactCom to the USB starterkit board.

4. Select Module->Download and select the correct Transport Path to your USB board.



5. The virtual attributes will be programmed and saved in non-volatile memory.

3.4 Node Address & Data Rate Configuration

3.4.1 General

The Anybus CompactCom supports automatic data rate detection, i.e. if no valid data rate is set, the Anybus CompactCom will measure the bus traffic at different speeds until the correct data rate has been established. Under normal conditions, i.e. with cyclic bus traffic above 2 Hz, the data rate should be detected within 5 seconds. Note that the automatic data rate detection will not work if there is no traffic on the network.

Layer Setting Services (LSS)

The Anybus CompactCom supports LSS (CANopen Layer Setting Service), according to CiA 305, 3.0.0.

This service can be used to set the data rate and node address via the network, and may address the module by its Vendor-ID, Product Code, Revision number and serial number.

It is possible to enable LSS during startup. To do this, use the following settings for application switch 1 & 2:

Application Switch	Switch Value
1	255
2	10

For more information about the application switches, see *Communication Settings, p. 13*.

3.4.2 Communication Settings

The node address is configured using the “Application switch 1” register. An application may select to write the value from a physical DIP switch, rotary switch or similar, to this register, or it can assign it by other means, see below.

Application switch 1 value	Used node address settings	Comment
1-127	Node address X	X is the “Application switch 1” value. The resulting node address is stored and will still be used if “Application switch 1” value is set to 128-255.
0, 128-255	LSS	Factory default settings: Node address: 255 **Note:** Node address may be set from the network.

The baud rate is configured by the “Application switch 2” register. An application may select to write the value from a physical DIP switch, rotary switch or similar, to this register or it can assign it by other means, see below.

Application switch 2 value	Used baud rate settings	Comment
0-10	Baud rate: X	X is the “Application switch 2” value. Resulting baud rate is stored and will still be used if “Application switch 2” value is set to 11-255. 0 = "10 kbps" 1 = "20 kbps" 2 = "50 kbps" 3 = "Reserved" 4 = "125 kbps" 5 = "250 kbps" 6 = "500 kbps" 7 = "800 kbps" 8 = "1 Mbps" 9 = "Auto" 10 = "LSS"
11-255	Use currently stored baud rate	Factory default settings: Baud rate: 9 (auto baud) **Note:** Baud rate may be set from the network.

3.5 Data Exchange

3.5.1 Parameter Data

Parameter Data can be accessed from the network via dedicated object entries in the Manufacturer Specific range and the Profile range (2001h - FFFFh).

Manufacturer and Profile Specific Objects

General

Each object entry in the manufacturer specific range (2001h...FFFFh) corresponds to a Modbus register, i.e. network accesses to these objects result in access towards the corresponding Modbus register. In case of an error, a descriptive abort code will be returned.

Network Data Format

Data is translated between the native network format and the Anybus CompactCom 40 data format as follows:

Anybus Data Type	Network Data Type	
UINT8	UNSIGNED8	
UINT16	UNSIGNED16	

3.5.2 Process Data

Modbus register	Content	Comment
0x5100	Data Type	0x0004 (UINT8) will result in objects of CANopen data type 0x0005 (UNSIGNED8, USINT). 0x0005 (UINT16) will result in objects of CANopen data type 0x0006 (UNSIGNED16, UINT).
0x5102	No of Write Parameters	Maximum 512 parameters when using data type UINT8. Maximum 256 parameters when using data type UINT16.
0x5103	No of Read Parameters	Maximum 512 parameters when using data type UINT8. Maximum 256 parameters when using data type UINT16.
0x0000 – (Depending on Data type and No of Parameters)	Write Process Data	The first write parameter is represented in object entry 0x2001, sub-index 0, the second write parameter is represented in object entry 0x2002, sub-index 0 and so on. This is valid regardless of the used data type.
0x1000 – (Depending on Data type and No of Parameters)	Read Process Data	The first read parameter is represented in object entry (0x2001 + NumberOfWriteParameters), sub-index 0, the second read parameter is represented in object entry (0x2002 + NumberOfWriteParameters), sub-index 0 and so on. This is valid regardless of the used data type.
0x5004	Network Type	0x0020 (CANopen) This register is used to identify the connected module.

The module supports up to 64 TxPDOs and up to 64 RxPDOs, each supporting up to 8 SDO mappings. Each SDO equals one Process Data mapped parameter.

Preferably, the CANopen EDS file should be altered to match the actual Process Data implementation. This is not a general requirement, but it has a positive impact on compatibility with 3rd party masters.

See also...

- *Standard Objects, p. 16*
- *Manufacturer and Profile Specific Objects, p. 13*

Example 1

No of Write parameters: 3 (0x5102)

No of Read parameters: 4 (0x5103)

Data type: (0x0004, UINT8)

Write parameter No	Modbus register	(CANopen Object entry)	TxPDO data byte offset
1	0x0000, LSB	0x2001:0	0x00
2	0x0000, HSB	0x2002:0	0x01
3	0x0001, LSB	0x2003:0	0x02

Total TxPDO length: 3 bytes

Read parameter No	Modbus register	XXX Object entry	RxPDO data byte offset
1	0x1000, LSB	0x2004:0	0x00
2	0x1000, HSB	0x2005:0	0x01
3	0x1001, LSB	0x2006:0	0x02
4	0x1001, HSB	0x2007:0	0x03

Total RxPDO length: 4 bytes

Example 2

No of Write parameters: 3 (0x5102)

No of Read parameters: 4 (0x5103)

Data type: (0x0005, UINT16)

Write parameter No	Modbus register	Object entry	TxPDO data byte offset
1	0x0000	0x2001:0	0x00
2	0x0001	0x2002:0	0x02
3	0x0002	0x2003:0	0x04

Total TxPDO length: 6 bytes.

The data written to the Modbus registers is swapped to little endian before it is sent on CANopen.

Read parameter No	Modbus register	Object entry	RxPDO data byte offset
1	0x1000	0x2004:0	0x00
2	0x1001	0x2005:0	0x02
3	0x1002	0x2006:0	0x04
4	0x1003	0x2007:0	0x06

Total RxPDO length: 8 bytes.

The data sent from the master on CANopen is swapped to big endian when written to the Modbus registers.

4 Object Dictionary (CANopen)

4.1 Standard Objects

4.1.1 General

The standard object dictionary is implemented according to the CiA 302 4.2.0 from CiA (CAN in Automation).

4.1.2 Object Entries

Index	Object Name	Subin-dex	Description	Type	Access	Notes
1000h	Device Type	00h	Device Type	U32	RO	Default 0000 0000h (No profile)
1001h	Error register	00h	Error register	U8	RO	
1003h	Pre-defined error field	00h	Number of errors	U8	RW	
		01h...0-5h	Error field	U32	RO	
1008h	Manufacturer device name	00h	Manufacturer device name	Visible string	RO	These entries are managed through virtual attributes, see *Startup and Identity Customization, p. 8*
1009h	Manufacturer hardware version	00h	Manufacturer hardware version	Visible string	RO	
100Ah	Manufacturer software version	00h	Manufacturer Software version	Visible string	RO	
1011h	Restore parameters	00h	Largest sub index supported	U8	RO	01h
		01h	Restore all default parameters	U32	RW	-
1014h	COB ID EMCY	00h	COB ID EMCY	U32	RW	Default value is 0000 0080h + NodeId
1015h	Inhibit Time EMCY	00h	Inhibit Time EMCY	U16	RW	Default value is 0000h
1016h	Consumer Heartbeat Time	00h	Number of entries	U8	RO	01h
		01h	Consumer Heartbeat Time	U32	RW	Node ID + Heartbeat Time. Value must be a multiple of 1 ms.
1017h	Producer Heartbeat Time	00h	Producer Heartbeat Time	U16	RW	-
1018h	Identity object	00h	Number of entries	U8	RO	Number of entries
		01h	Vendor ID	U32	RO	These entries are managed through virtual attributes, see *Startup and Identity Customization, p. 8*
		02h	Product Code	U32	RO	
		03h	Revision Number	U32	RO	
		04h	Serial Number	U32	RO	
1400h - 14xxh	RPDO communication parameter	00h	Largest sub-index supported	U8	RO	02h
		01h	COB ID used by RPDO	U32	RW	-
		02h	Transmission type.	U8	RW	-

Index	Object Name	Subin-dex	Description	Type	Access	Notes
1600h - 16xxh	Receive PDO mapping	00h	No. of mapped application objects in PDO	U8	RO	No. of mapped objects (0.. 8), see *Process Data, p. 14* for more information.
		01h	Mapped object #1	U32	RO	-
		02h	Mapped object #2	U32	RO	-
		...	...	...	...	-
		NNh	Mapped object #NN	U32	RO	-
1800h - 18xxh	TPDO communication parameter	00h	Largest sub-index supported	U8	RO	05h
		01h	COB ID used by TPDO	U32	RW	-
		02h	Transmission type	U8	RW	-
		03h	Inhibit time	U16	RW	-
		05h	Event Timer (ms)	U16	RW	-
1A00h - 1Axxh	Transmit PDO mapping	00h	No. of mapped application objects in PDO	U8	RO	No. of mapped objects (0.. 8), see *Process Data, p. 14* for more information.
		01h	Mapped object #1	U32	RO	-
		02h	Mapped object #2	U32	RO	-
		...	...	...	...	-
		NNh	Mapped object #NN	U32	RO	-

4.2 Manufacturer and Profile Specific Objects

4.2.1 General

Each object entry in the manufacturer specific range (2001h...FFFFh) corresponds to a Modbus register, i.e. network accesses to these objects results in object requests towards the host application. In case of an error, the status (or error) code returned in the response from the host application will be translated into the corresponding CANopen abort code.

Object Range	Description
2001h... 5FFFh	Manufacturer specific objects
6000h... 9FFFh	Profile specific objects
A000h... BFFFh	Standardized variable objects
C000h... FFFFh	Reserved

This page intentionally left blank

A LED Indications

See Anybus CompactCom B40 Modbus Serial User Manual for more information.

A.1 RUN LED

LED State	Description	Comments
Off	-	No power.
Green	OPERATIONAL	The module is in the state OPERATIONAL.
Green, blinking	PRE-OPERATIONAL	The module is in the state PRE-OPERATIONAL.
Green, 1 flash	STOPPED	The module is in the state STOPPED.
Green, flickering	Autobaud	Baud rate detection in progress or LSS in progress (alternately flickering with ERROR LED)
Red	EXCEPTION state (Fatal Event)	The module has shifted into the state EXCEPTION.

If both LEDs turns red, this indicates a fatal event; the bus interface is shifted into a physically passive state.

A.2 ERROR LED

LED State	Description	Comments
Off	-	No power or the device is in working condition.
Red, single flash	Warning limit reached	A bus error counter reached or exceeded its warning level.
Red, flickering	LSS	LSS services in progress (alternately flickering with RUN LED).
Red, double flash	Error Control Event	A heartbeat event (Heartbeat consumer) has occurred.
Red	Bus off (Fatal Event)	Bus off

If both LEDs turns red, this indicates a fatal event; the bus interface is shifted into a physically passive state.

B Conformance Test Guide

B.1 Introduction

This chapter includes network specific settings that are needed for a host application to be up and running and possible to certify for use on CANopen networks.

B.2 Fieldbus Conformance Notes

- This product is pre-certified for network compliance. While this is done to ensure that the final product can be certified, it does not necessarily mean that the final product will not require recertification. Contact HMS Networks for further information.
- The .EDS file associated with this product must be altered to match the final implementation. See also *Electronic Data Sheet (EDS), p. 7*.
- HMS Networks recommends that the device identity information is customized to ensure interoperability. CiA (CAN in Automation) members should apply for a unique Vendor ID; non-members may contact HMS Networks to obtain a custom Product ID. Note however that a unique Vendor ID is required when certifying the final product.
- The module supports CAN Standard Frames with 11-bit Identifier Field, see CiA 301 v4.2.0. 29-bit Identifier Fields are not allowed.

B.3 Certification

When using the default settings of all parameters, the Anybus CompactCom 40 Modbus Serial - CANopen is precertified for network compliance. This precertification is done to ensure that your product can be certified, but it does not mean that your product will not require certification.

Any change in the parameters in the EDS file, supplied by HMS Networks, will require a certification. A Vendor ID can be obtained from CiA (CAN in Automation) and is compulsory for certification. This section provides a guide for a successful conformance testing of your product, containing the Anybus CompactCom 40 Modbus Serial - CANopen, to comply with the demands for network certification set by CiA (CAN in Automation).

Independent of selected operation mode, the actions described in this section have to be accounted for in the certification process. The identity of the product needs to be changed to match your company and device.

> **!** This section provides guidelines and examples of what is needed for certification. Depending on the functionality of your application, there may be additional steps to take. Please contact HMS Networks at www.anybus.com for more information.

B.3.1 Reidentifying Your Product

The identification attributes listed below shall be implemented and proper values returned. See *Startup and Identity Customization, p. 8* for more information.

Attribute	Explanation	Default	Customer sample	Comment
#1, Vendor ID	With this attribute you set the Vendor ID of the device.	Vendor ID: 0000 001Bh	Vendor ID: 1111h	This information must match the keyword values of the “Device” section in the EDS file.
#2, Product Code	With this attribute you set the Product Code of the device	0000 000Dh	0000 2222h	
#3, Major Revision	With this attribute you set the Major Revision of the device.		0001	
#4, Minor Revision	With this attribute you set the Minor Revision of the device.		0001	
#6, Manufacturer Device Name	With this attribute you set the Product Name of the device.	Anybus CompactCom 40 CANopen	“Widget”	This information must match the keyword values of the Device section in the EDS file. See CANopen object 1008h, *Standard Objects, p. 16*.

B.3.2 Factory Default Reset

Factory Default Reset command must be supported

When Anybus CompactCom 40 Modbus Serial - CANopen products are delivered, they are required to be in their Factory Default state. During runtime, the Modbus master must continuously read Modbus register 0x0FFF. If bit 14 is set, the application must be reset to factory default values.

B.3.3 Modify the EDS File

Modify the Anybus CompactCom CANopen EDS file so that it corresponds to the vendor product (e.g. Vendor ID, Product Name and Product Number along with ADI object names, that correspond to descriptive names in the application. Also the ADI information must correspond.). The EDS file has to contain all ADIs created by the application. Run the EDS file checker program from www.can-cia.org.

See also *Electronic Data Sheet (EDS), p. 7* .

Box 4126
300 04 Halmstad, Sweden

info@hms.se

SCM-1202-166 1.0 en-US / 2020-08-31 / 19639