Anybus® X-gateway™ CANopen®

PROFINET® IRT (2.32)

USER MANUAL

SCM-1202-029 1.2 en-US ENGLISH

Connecting Devices™

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

1 Preface

1.1 About This Document

This document describes Anybus X-gateway CANopen PROFINET IRT (2.32).

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

1.2 Document history

Version	Date	Description
1.0	2017-01-23	First release
1.1	2017-11-22	Updated for new firmware
1.2	2019-04-11	Added section about PROFINET Asset Management

1.3 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. 4*

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.

1.4 Document-Specific Conventions

- Hexadecimal values are represented with the suffix *h* and a leading zero where needed, e.g. the hexadecimal value 1F4 is written `01F4h`.
- A byte always consists of 8 bits.

2 Description

2.1 Introduction

Anybus X-gateway CANopen is designed to provide a seamless connection between a primary fieldbus or Ethernet network and a secondary CANopen sub-network.

The X-gateway transmits I/O data transparently between the two networks. Data from the primary network is written into CANopen objects that can be mapped into CANopen PDOs or read via CANopen SDOs, and vice versa. This makes it possible to integrate CANopen devices into almost any other PLC system and their supported networks.

No proprietary configuration software is needed for Anybus X-gateway CANopen, although dedicated tools may be required when configuring the primary network. Any standard CANopen configuration tool can be used to configure the secondary CANopen network interface.

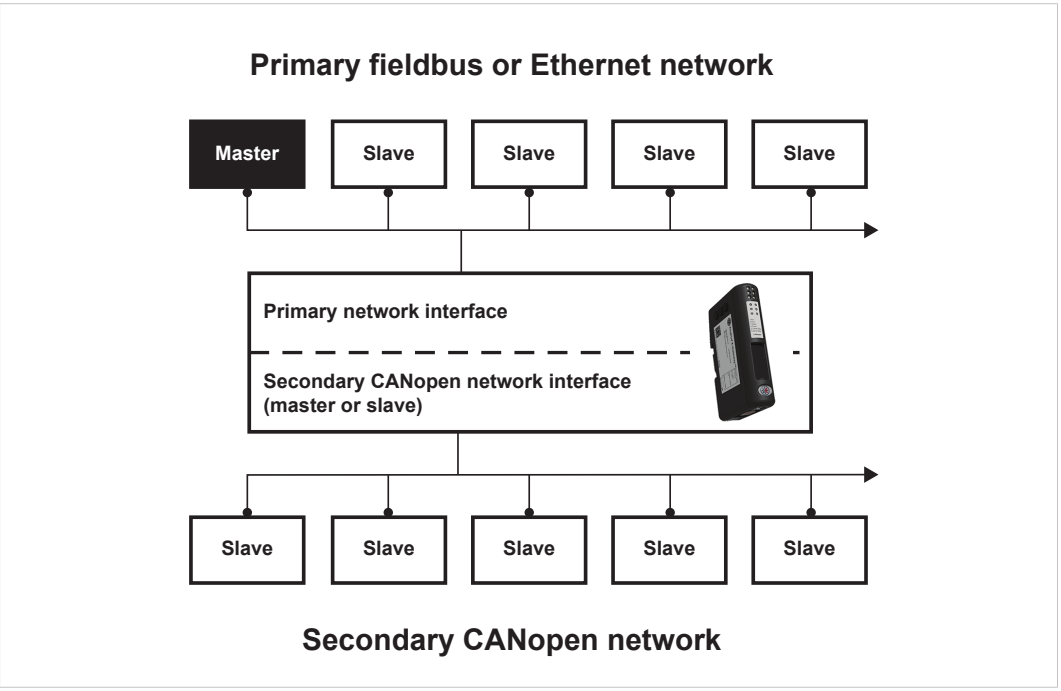


Fig. 1 Networking example

2.2 Data Exchange

The terminology and definitions used for different types of data vary between network types. All data transported through the Anybus X-gateway CANopen are fast, cyclic data, and will in this document simply be referred to as "I/O data".

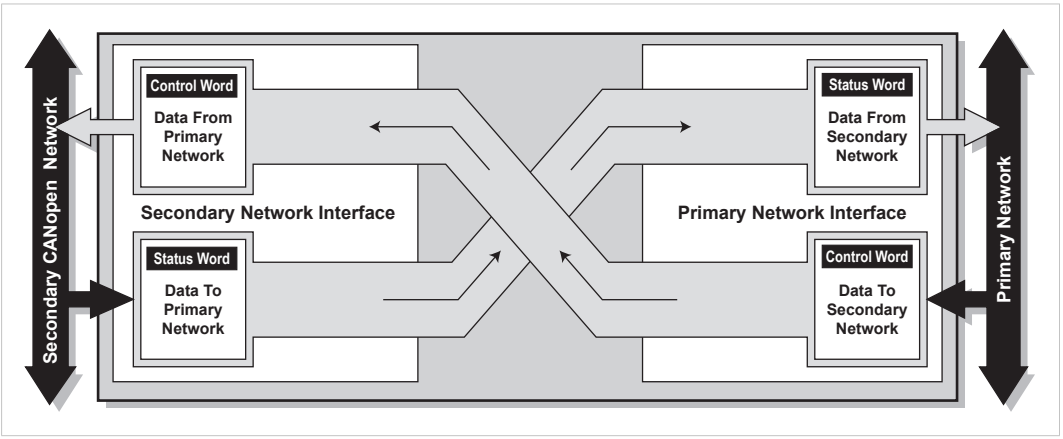


Fig. 2 Data exchange example

Each of the two network interfaces exchanges data on its network through its own buffer, which can hold up to 512 bytes of data. The first two bytes in the primary network buffer are reserved for the Control Word and Status Word, leaving 510 bytes available for I/O data.

The actual amount of data that can be exchanged depend on the application and network used and may therefore be less than 510 bytes, which is only the maximum size of the buffer.

The Control Word can be used by the master on the primary network to start and stop the exchange of data, and to reset the X-gateway if needed. The Status Word can be used by the master to read the status of the secondary CANopen network.

The I/O data exchange is separated from the network data exchange. While the gateway ensures data consistency (where applicable), it does not feature any mechanisms for synchronisation between the primary and secondary networks.

2.3 CANopen Functionality

The functionality of the secondary CANopen network interface is defined by the following CANopen specifications:

- KGB Draft Standard 301 version 4.2.0 (Rev. 4.2)
- CiA Draft Standard Proposal 302 Part 1–5.

Supported CANopen Services

Communication and parameters in the CANopen protocol are built around *å*bject. Different services are used for communication with the objects and for other tasks such as supervising the network. Which services are available depend on whether the secondary CANopen network interface is operating as a master or as a slave.

The secondary CANopen network will start up as a slave by default.

Service	Available in	Description
NMT (Network Management)	Master	NMT messages are used to configure, initialize and monitor the network,and for error handling.
CMT (Configuration Management)	Master	CMT messages are used for configuring CANopen devices. This primarily involves PDO parameters and mapping of information.
PDO (Process Data Objects)	Master/Slave	Used for I/O communication. 128 Receive PDOs and 128 Transmit PDOs are implemented, each being able to transfer up to 8 bytes. The total number of PDOs that can be used is limited by the data buffer size. Supported PDO message types are *COS (Change of state)*, *Cyclic Synchronous*, and *Acyclic Synchronous*.
SDO (Service Data Objects)	Master/Slave	Used to access and configure objects in the X-gateway and other network nodes without mapping them to an I/O (PDO) connection. SDOs use asynchronous data transmission and can transfer more than 8 bytes (the limit for a PDO). Supported SDO message types are *Expedited Upload/ Download Protocol* and *Segmented Upload/Download Protocol*.
SYNC (Synchronization Object)	Master/Slave	Used for synchronizing PDO communication. A master can be either a producer or a consumer of the synchronization. A slave can only be a consumer.
EMCY (Emergency Object)	Master/Slave	Used for error reporting when a fatal error has occurred in the X-gateway or in other monitored or supervised modules.
LSS (Layer Setting Services)	Master	Used by a CANopen master to configure the baud rate and NodeID of slaves that support LSS.
Heartbeat Mechanism	Master/Slave	Allows a device to monitor the status of another node. The X-gateway can appear both as heartbeat producer and consumer.
Node Guarding Protocol	Master/Slave	Provides active surveillance of a slave by the master. Slaves can be configured to expect a node guarding request from the master.

3 Installation

> ! This product contains parts that can be damaged by electrostatic discharge (ESD). Use ESD prevention measures to avoid damage.

3.1 Installation Overview

Basic steps when installing the Anybus X-gateway CANopen:

1. Set the node address and baud rate for the secondary CANopen interface.
2. Set the hardware configuration switches for the primary network interface (if applicable).
3. Mount the gateway on the DIN rail.
4. Connect the primary and secondary networks.
5. Connect the power cable and apply power.
6. Continue to *Configuration, p. 14*.

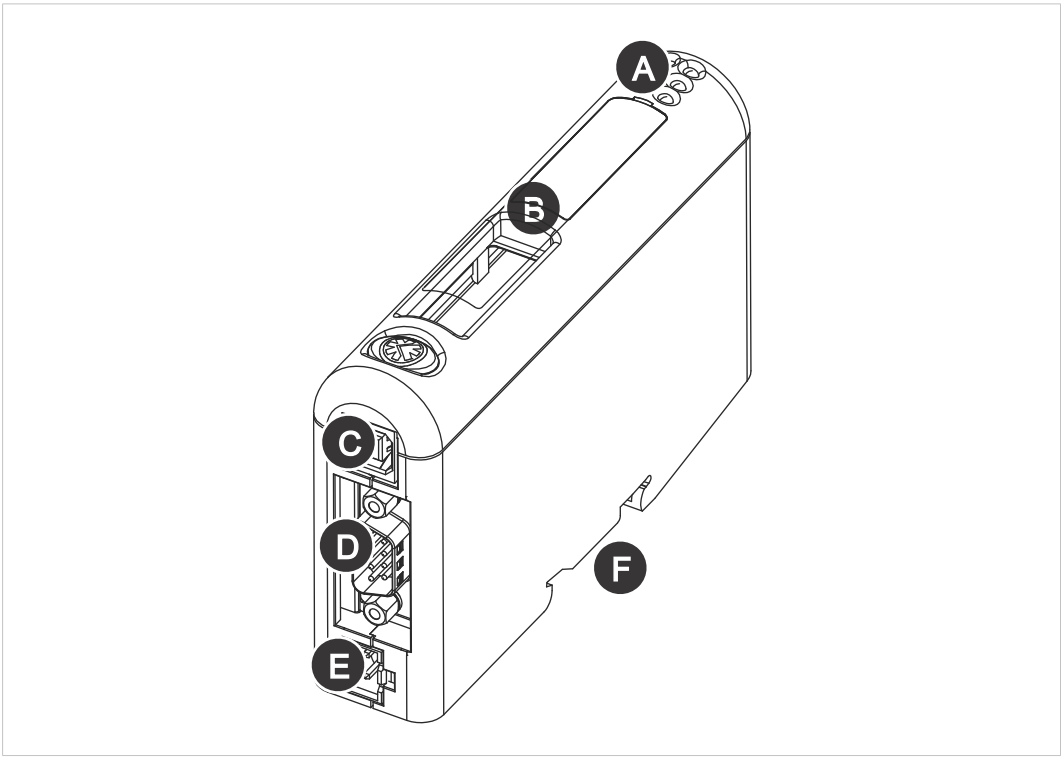


Fig. 3 Connectors, switches and indicators

A	LED indicators
B	Primary network interface
C	USB connector
D	CANopen connector
E	Power connector
F	DIN rail mount

3.2 DIN Rail Mounting

> The unit must be electrically grounded through the DIN rail for EMC compliance.

Mount on DIN rail

1. Hook the unit onto the upper lip of the rail and push gently downwards.
2. Push the unit towards the rail until it snaps into place.

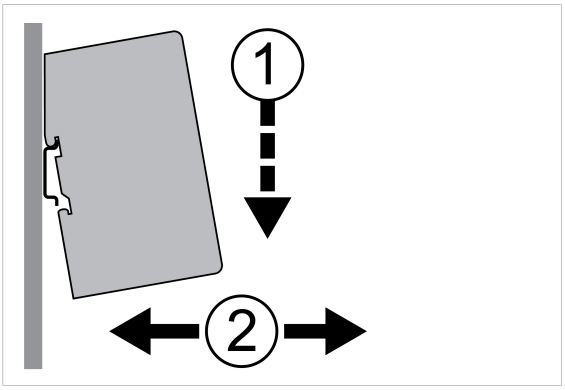


Fig. 4 Push down to mount or remove

Remove from DIN rail

1. Push the unit gently downwards on the rail.
2. Pull the bottom end of the unit free of the rail and remove it.

3.3 Power Connector

See also *Technical Data, p. 59* regarding power supply requirements.

Pin	Signal
1	+24 VDC
2	Power Ground

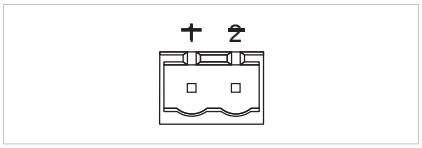


Fig. 5 Power connector

3.4 USB Connector

The USB connector is only used when upgrading the firmware of the unit. It cannot be used for configuration purposes.

Pin	Signal
1	+5 V input
2	USBDM (USB communication)
3	USBDP (USB communication)
4	Signal ground
Housing	Cable shield

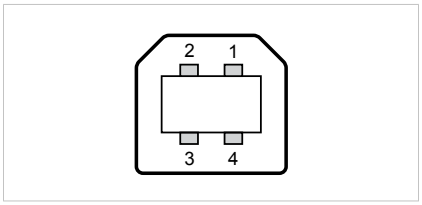


Fig. 6 USB type B connector

3.5 Secondary CANopen Network Interface

3.5.1 Configuration Switches

Three configuration switches on the side of the unit are used to set the node address and baud rate for the secondary CANopen network interface.

> **!** The node address and baud rate cannot be changed during runtime. The module must be restarted to make a changed setting take effect.

Baud Rate (Switch A)

The first rotary switch is used to set the operating baud rate.

Setting	Baud Rate (kbit/s)
0	20
1	50
2	125
3	250
4	500
5	800
6	100
7	Auto
8, 9	(not used)

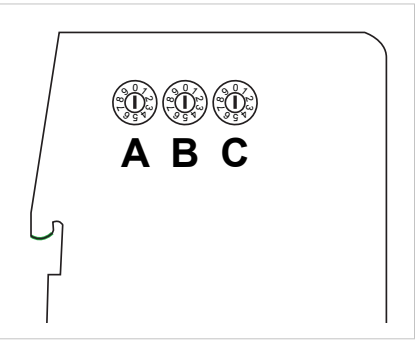


Fig. 7 Configuration switches

> **!** Do not select “Auto” if the traffic on the secondary network will be limited, e.g. if there are only a few nodes or the interface is configured as a master.

Node Adress (Switches B + C)

The second and third switches are used together to set a CANopen node address between 1 and 99. In the following example the node address is set to **42** (4 x 10 + 2 x 1):

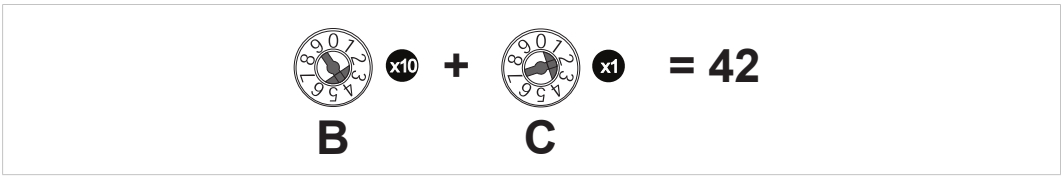


Fig. 8 Node address example

3.5.2 CANopen Connector

The secondary network CANopen connector is located on the bottom of the unit. This connector is also used when downloading the CANopen configuration.

Pin	Signal
2	CAN_L
3	CAN GND
5	Shield
6	CAN GND
7	CAN_H
1, 4, 8, 9	(reserved)

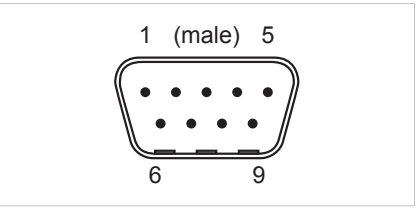


Fig. 9 CANopen connector

3.6 PROFINET IRT Network Interface

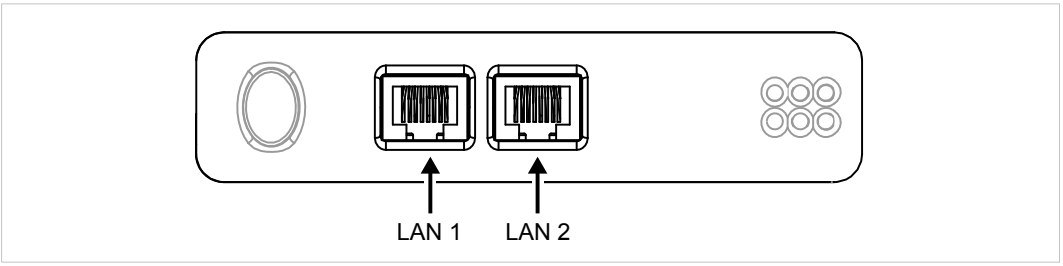


Fig. 10 PROFINET IRT interface

Ethernet Connectors (LAN 1/LAN 2)

The PROFINET IRT interface contains a dual port Ethernet switch with RJ45 type connectors. The two ports are labeled **LAN 1** and **LAN 2**.

Pin	Function
1	TD+
2	TD-
3	RD+
6	RD-
4, 5, 7, 8	(reserved)

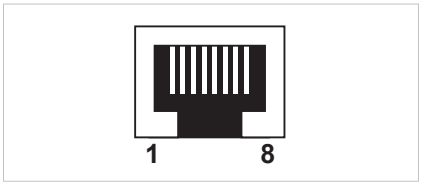
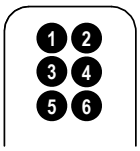


Fig. 11 Ethernet connector (RJ45)

3.7 LED Indicators



LED 1 to 4	Primary network interface status
LED 5	Secondary CANopen network interface status
LED 6	Device operation status

LED Indicators – Primary PROFINET IRT Network Interface		
1 - Network Status	Off	Offline – No power – No connection to IO Controller
	Green	Online (RUN) – Connection to IO Controller
	Green, 1 flash	Online (STOP) – Connection to IO Controller – IO Controller in STOP state or IO data bad – RT synchronization not finished
	Red	Fatal error
	Red, 1 flash	Station name error
	Red, 2 flashes	IP address error
	Red, 3 flashes	Configuration error
	Alternating red/green	Firmware update in progress
2 - Module Status	Off	No power or initializing
	Green	Normal operation
	Green, 1 flash	Diagnostic event present
	Red	Fatal error
	Alternating red/green	Firmware update in progress
3 - Link/Activity 1 4 - Link/Activity 2	Off	No power or no link detected
	Green	Link OK
	Green, flickering	Transmitting/receiving data

LED Indicators – Secondary CANopen Network Interface & Device Status		
5 - CANopen Status	Off	No power
	Flickering red/green	LSS services in progress
	Green	Operational state
	Green, 1 flash	Stopped state
	Green, blinking	Pre-operational state
	Red	Bus off
	Red, 1 flash	Warning limit reached
	Red, 2 flashes	Error control event
	Red, 3 flashes	Sync error
	Red, 4 flashes	Data communication timeout
	Red, blinking	Configuration error
6 - Device Status	Off	Power off
	Green	Running
	Green, 1 flash	Bootup
	Red	Fatal error
	Red, 1 flash	Initialization error
	Red, 2 flashes	Timeout error
	Red, 3 flashes	Hardware failure
	Red, 4 flashes	Invalid switch settings

LED Indicator Timing Intervals

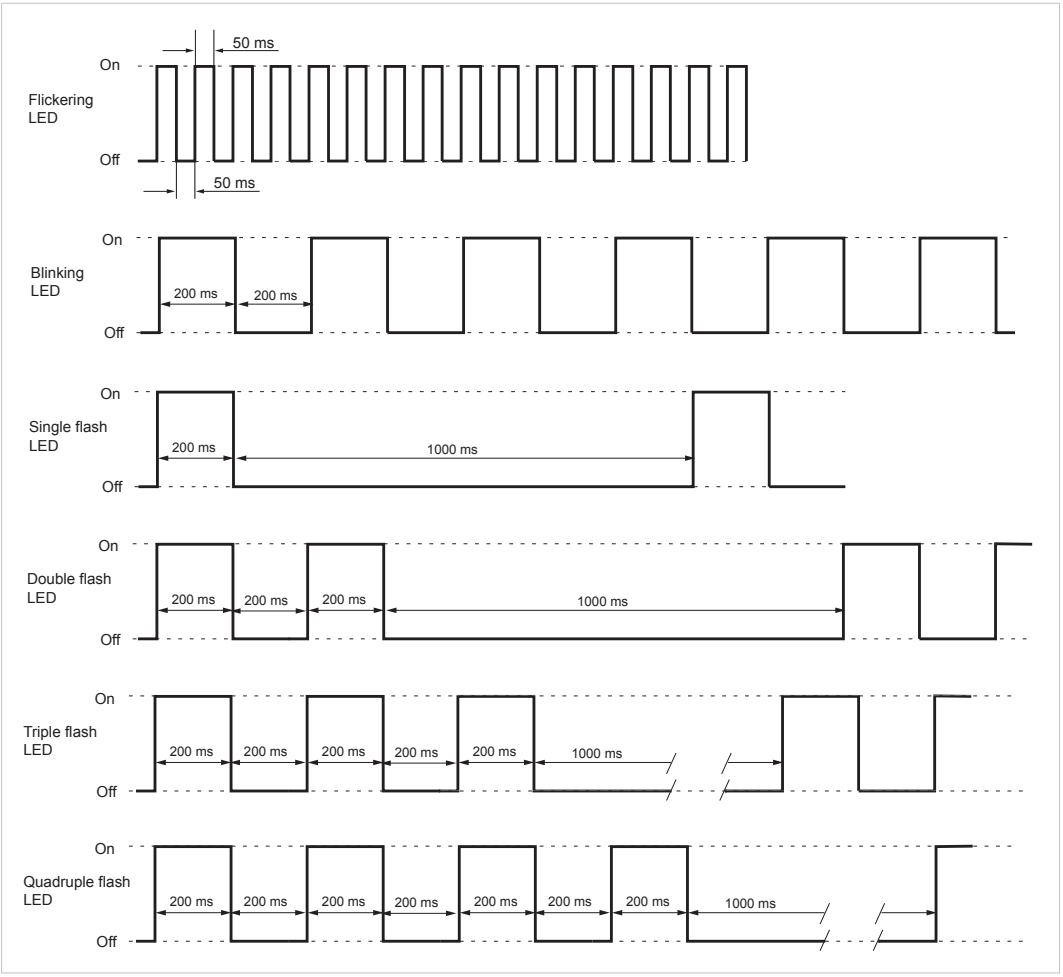


Fig. 12 LED indicator timing intervals

4 Configuration

4.1 Configuration Overview

Device Description Files

A device description file contains a description of a network device, its functions, object dictionary implementations, etc., and is used when configuring the network interface. The device description file can be referred to as an EDS, GSD, DDF, etc., depending on the type of network.

The latest versions of the device description files to use with Anybus X-gateway CANopen can be downloaded from www.anybus.com/support.

Basic steps when configuring Anybus X-gateway CANopen

> **!** The secondary network interface should be configured first. The gateway must then be power cycled before configuring the primary network interface.

1. Determine the amount of data that should be transferred. This value will be entered in the secondary CANopen network interface configuration.
2. Configure the secondary CANopen network interface.
   See *Configuring the Secondary CANopen Network, p. 15*.
3. Power cycle the X-gateway.
4. Configure the primary network interface.
   See *Configuring the PROFINET IRT (2.32) Interface, p. 16*.

Module Identification

Anybus X-gateway CANopen will identify itself on the network as follows:

Description	Value
Vendor Code	90
Vendor Name	“HMS Networks”
Product Code	51
Product Type	12
Product Type String	“Communications Adapter”
Product Name	“Anybus X-gateway CANopen”
Catalog	“Anybus X-gateway CANopen”
Desc Text	“Anybus X-gateway CANopen”

4.2 Configuring the Secondary CANopen Network

This is a generic description of the basic steps in configuring the secondary CANopen network interface using an external CANopen configuration tool. For instructions on how to create and apply a configuration, please refer to the documentation for the configuration tool used.

1. Download the Anybus X-gateway CANopen EDS file from www.anybus.com/support.
2. Prepare EDS files for the other nodes on the secondary CANopen network.
3. Open the CANopen configuration tool and upload the EDS files to it.
4. Configure the following parameters in the Anybus X-gateway CANopen:

Parameter	Value range	Comment
NodeID	1 to 127	NodeID 1 to 99 can be set with the configuration switches. NodeID 99 to 127 can only be be set using a configuration tool or from the CANopen network.
Baud Rate	20, 50, 125, 250, 500, 800, 1000, Auto	Set with configuration switch. *Auto* should only be used when configured as a slave.
Master/Slave	Master or Slave	Default = Slave. See also *NMT Start-up, 1F80h, p. 48*.
Input Data Size (object 3000h)	2 to 512	Size of the data transmitted **to** the primary network. Bytes 0 and 1 are reserved for the Status Word, leaving a maximum of 510 bytes available for data. The actual maximum data size depends on the primary network. Default = 16 bytes (14 bytes data + 2 bytes Status Word).
Output Data Size (object 3001h)	2 to 512	Size of the data received **from** the primary network. Bytes 0 and 1 are reserved for the Control Word, leaving a maximum of 510 bytes available for data. The actual maximum data size depends on the primary network. Default = 16 bytes (14 bytes data + 2 bytes Control Word).

5. Configure the other CANopen nodes as needed. Make sure that each node uses the same baud rate and has a unique NodeID.
6. Download the configuration from the tool to each CANopen node.

   The configuration can be downloaded individually to each node, or as a *Concise DCF* file to the CANopen master which will then configure the slaves.
7. Power cycle the X-gateway.
8. Continue to *Configuring the PROFINET IRT (2.32) Interface, p. 16*.

The secondary CANopen network will start up as a slave by default.

4.3 Configuring the PROFINET IRT (2.32) Interface

> The secondary network interface should always be configured first. The gateway must then be power cycled before configuring the primary network interface.

The primary network interface of the X-gateway must be configured with the configuration tool used for the network it is connected to. The choice of configuration tool depends on the type of network, the application, and the master used on the primary network.

Application notes describing how to configure primary network interfaces in Anybus X-gateway CANopen with some of the most common tools can be found at www.anybus.com/support.

4.3.1 PROFINET Data Exchange

PROFINET is the open Industrial Ethernet standard for automation from PROFIBUS and PROFINET International. The PROFINET IRT device provides PROFINET IO Isochronous Real Time Communication.

PROFINET makes a clear distinction between fast cyclical data, *IO Data*, and acyclical data, *Record Data*. PROFINET IO Data corresponds to what is generally referred to as *I/O Data* in Anybus X-gateway CANopen. PROFINET Record Data is not supported.

PROFINET IO Data (I/O Data)

PROFINET IO Data is exchanged cyclically and is built up by I/O modules. The actual I/O configuration is determined by the PROFINET IO Controller. The modules are mapped to the Input and Output Buffers in the order of their slot number.

The first two bytes of the I/O data area are reserved for the Control Word and the Status Word, which are used by the IO Controller to control and report status on the nodes on the secondary CANopen network. The remainder is available for real-time data transfer using PDOs.

The amount of data exchanged as I/O data is specified when configuring the CANopen master interface. The data arriving from the CANopen master is completely transparent. The interpretation must be defined by the master on the primary network.

GSD File

All PROFINET devices are associated with an XML-based *GSD* file. This file contains information about the basic capabilities and configuration options of the device.

The latest version of the GSD file for Anybus X-gateway CANopen can be downloaded from www.anybus.com/support.

4.3.2 Network Configuration

To be able to communicate over Ethernet the network interface needs a valid TCP/IP configuration. This section explains some basic concepts and describes how to configure the TCP/IP settings in Anybus X-gateway CANopen using the *IPconfig* software tool.

When Ethernet communication has been established the TCP/IP settings can also be changed from the web interface. See *Web Pages, p. 21*.

Basic TCP/IP Concepts

IP Address

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

Example: `10000000 00001010 00000010 00011110` is written as `128.10.2.30`

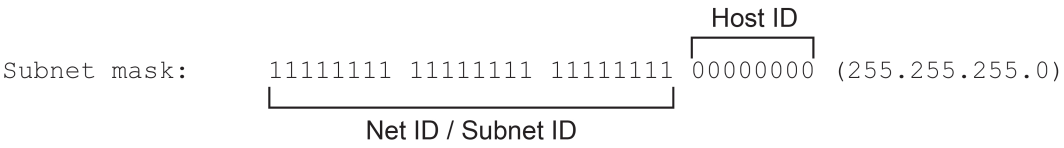
The following IP addresses are reserved for special purposes and cannot be used:

0.n.n.n	First byte zero — used for broadcast messages
127.n.n.n	First byte 127 — used for loopback addresses to the local host
n.n.n.0	Last byte zero — identifies a whole network/subnet
n.n.n.255	Last byte 255 — used for broadcast messages

Subnet Mask

The IP address is divided into three parts: *Net ID*, *Subnet ID* and *Host ID*. A subnet mask is a 32-bit binary pattern, where a set bit allocates a bit for Network/Subnet ID, and a cleared bit allocates a bit for the Host ID. The subnet mask is usually written in dot-decimal notation.

Example: To make the IP address `128.10.2.30` belong to subnet `128.10.2`, the subnet mask must be `255.255.255.0`.



Default Gateway

For devices to be able to communicate over Ethernet they must either belong to the same subnet or communicate via a gateway or router.

A gateway or router routes communication between networks, i.e. it enables the nodes on one network to access the nodes on another. The *default gateway* address in the TCP/IP settings of your product specifies the IP address of the gateway or router on the local network.

TCP/IP Configuration

Installing the IPconfig Utility

IPconfig is a Windows-based tool for configuration of TCP/IP settings in HMS devices. The tool will detect all compatible and active HMS devices on the local network.

1. Download IPconfig from www.anybus.com/support.
2. Unpack the contents of the zip archive and run the installer program.

Scanning for Connected Devices

When IPconfig is started it will automatically scan all available local networks for HMS devices. Detected devices will be listed in the main window. To refresh the list, click on **Scan**.

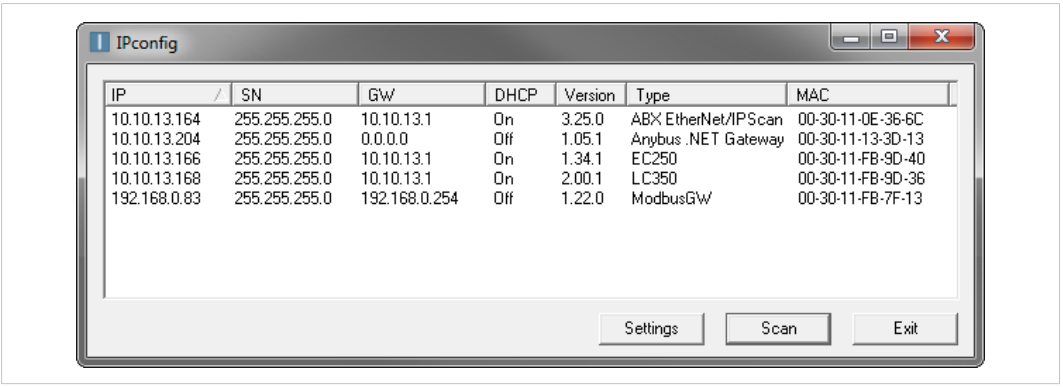


Fig. 13 IPconfig main window

IP	IP address of the device
SN	Subnet mask
GW	Default gateway
DHCP	Automatically managed IP configuration
Version	Firmware version
Type	Product name
MAC	Ethernet MAC address (System ID)

Ethernet Configuration

To change the IP settings for a device, double-click on the entry in the main window or right-click on it and select **Configuration**.

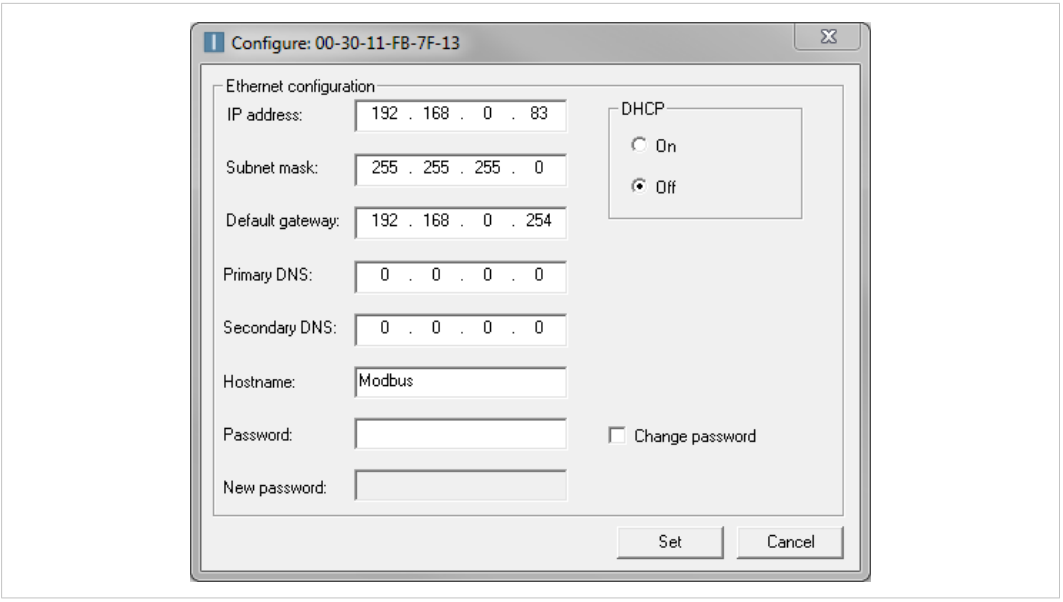


Fig. 14 Ethernet configuration

Enter static IP settings as required, or select DHCP if using dynamic IP addressing.

> Do not enable DHCP if there is no DHCP server available on the network.

You can add a name for the device in the **Hostname** field. Only characters a–z, A–Z, 0–9 and _ (underscore) are allowed.

The default password for changing IP settings is blank (no password). If a password has been set for the device you must enter it to be able to change the settings.

To set a new password, check the **Change password** box and enter the current password in the **Password** field, then enter the new password in the **New password** field.

> For security reasons the default password should always be changed.

Click on **Set** to save the new settings. The device will reboot automatically.

IPconfig Settings

Additional settings for IPconfig can be accessed by clicking on **Settings**.

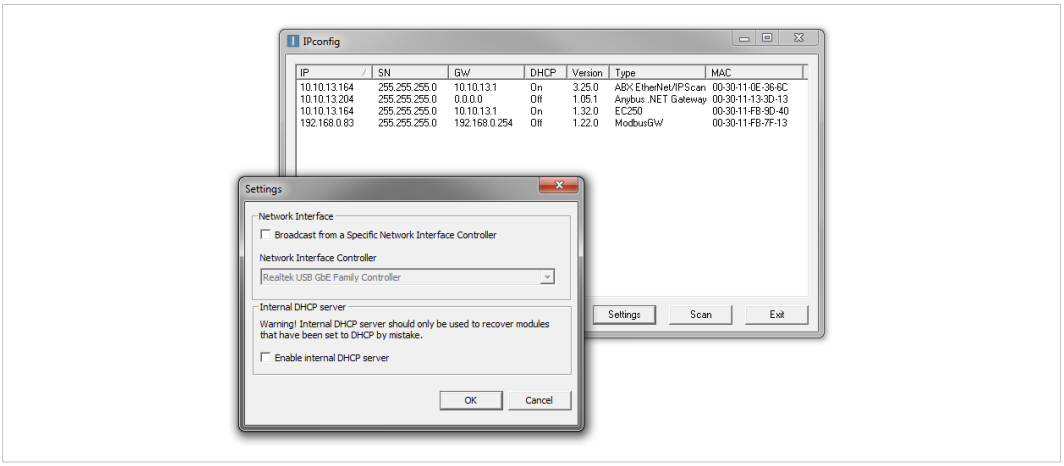


Fig. 15 IPconfig settings

Network Interface

Check this option to select a specific network interface to use when scanning for devices from a computer which has more than one interface. If this option is left unchecked, all available networks will be scanned.

Internal DHCP Server

If a device has been set to use DHCP but there is no DHCP server on the network, the device may not be detected by IPconfig. To recover access to the device an internal DHCP server in IPconfig can be temporarily activated:

1. Click the checkbox for **Internal DHCP Server**, then click **OK**. IPconfig will automatically refresh the scan and list the missing device in the main window.
2. Select the device and configure it to use static IP addressing instead of DHCP.
3. Disable the internal DHCP server.

> **!** Do not enable the internal DHCP server if there is already an active DHCP server on the network.

DCP (Discovery and Control Protocol)

Anybus X-gateway CANopen PROFINET IRT (2.32) supports the DCP protocol, which allows a PROFINET IO Controller/Supervisor to change the network settings during runtime.

4.3.3 Web Pages

Network configuration settings and status of the PROFINET IRT network interface can be accessed by pointing a web browser to the IP address of the interface.

Module Overview

Fig. 16 Overview tab

Provides basic information about the X-gateway including the serial number and the installed firmware version.

Network Status

Fig. 17 Status tab

Displays an overview of the current network status.

Network Configuration

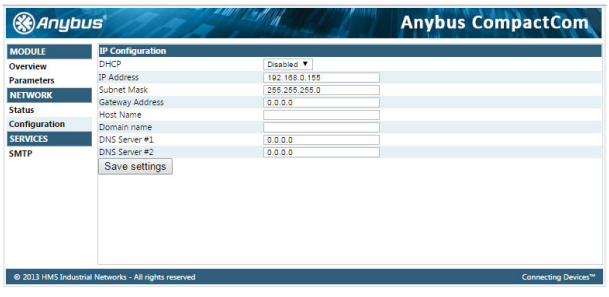


Fig. 18 Configuration tab

Provides access to the TCP/IP network settings. These parameters can also be configured using the *IPconfig* tool.

4.4 Enabling Data Exchange

Once both interfaces of the X-gateway have been properly configured, the PLC (master) on the primary network must explicitly allow the X-gateway to exchange I/O data between the primary and secondary networks by writing the `OPERATIONAL` command in the control word.

If the X-gateway is set as **master** on the secondary network, it will automatically be available when the PLC has enabled data exchange. The X-gateway will control the secondary network, using the instructions that are sent in the control word by the PLC.

If the X-gateway is set as a **slave** on the secondary network, it will wait for a request from the master before starting to exchange data. If the X-gateway has not been enabled by the PLC to exchange data it will return an error message to the secondary network.

The secondary CANopen network will start up as a slave by default.

5 PROFINET Asset Management

5.1 Asset Management Record

With the *asset management record* functionality data about the assets available on a non PROFINET network can be recorded and read out over a PROFINET network.

Together with the *Identification & Maintenance data* functionality an extensive registration of devices and machines is possible, even in facilities where the devices are not installed in the PROFINET environment.

Factory owners and system integrators can collect data about devices installed beyond the *Anybus gateway.*

The recorded data can be used as basis for the design of easier maintenance and operation processes, despite the increasing complexity of processes and associated machines.

5.2 Recording and Reading Data

An *asset management* file containing all the *assets* and their corresponding data on the non PROFINET network is created and uploaded via an *FTP* server to the Gateway *file system*.

The *asset management* file can be transferred from a computer connected to a PROFINET network.

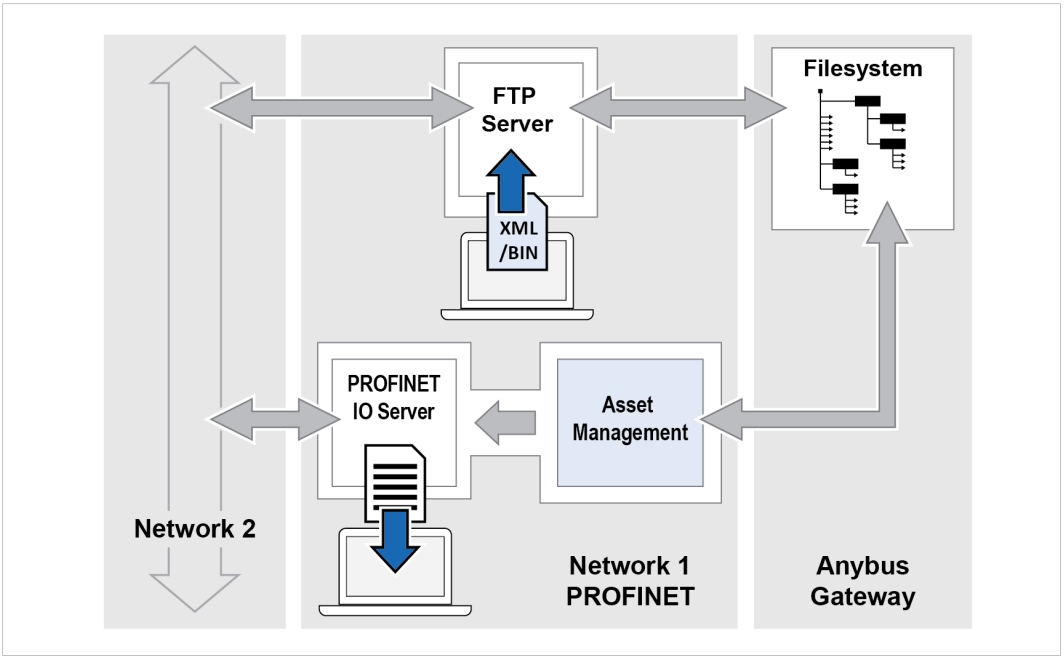


Fig. 19 The Asset Management Default Mode

By using the *superposed parameter channel* mode it is also possible to transfer the *asset management* file from a PLC connected to a non PROFINET network.

For further details about the *superposed parameter channel* mode, please refer to www.anybus.com/support.

Record Data

Data about the *assets* on the non PROFINET network is recorded and stored in an *XML* file or an *binary* file.

Read Data

Each time an *instance* is requested the *asset management* data is read out over the PROFINET network.

The recorded *asset management* data can be downloaded to a computer connected to the PROFINET network.

5.3 Supported File Formats

The following file formats are supported for the *asset management* file.

Format	Version
XML	XML Version 1.0
Binary file Little-endian	N/A

5.4 Supported Asset Management Records

Supported *asset management records*:

- Unique ID
- Location
- Hardware Revision
- Annotation
- Order ID
- Serial Number
- Software Revision
- Serial Number

5.5 XML Based Asset Management

5.5.1 Creating the Asset Management XML File

Creating the *asset management* XML file:

1. List all assets and their corresponding data on the non PROFINET network.
2. Create an XML file that include one *asset management record* for each asset.

   Repeat all the *attributes* after each other.
3. When all *attributes* are listed, close the *element* by using a *closing entry.*
4. Name the XML file *asset_mgmt*.

5.5.2 XML File Size Limitation

The size of the asset management file may not exceed 95 kb.

Up to 32 *instances* can be added.

In order to keep the file size small, consider the following:

- Keep strings as short as possible.
- Do not pad with empty spaces for strings.
- Try to use as few spaces as possible for indentation in the file.
- The number of white-space also affects the file size.
- Avoid using *optional name strings*.

5.5.3 XML Attribute Name and Data Format

The order of the elements is significant for the XML schema to work with the Anybus Gateways.
If the XML schema is incorrect, the XML file will not work and no data will be recorded.

When creating the XML file, add the *elements* and their *attributes* in the same order as the *attribute names* are listed in the table below.

Each *element* consists of a series of *attributes* and their various data.

Each *attribute* is described by one *entry.*

The supported *attribute names* are specified in the table.

Example 1: XML *element* including an *attribute* with the *location* record.

```
<AbccAttribute>
<Name Value="Location Type"/>
<Attribute Value="3"/>
<Data Value="1"/>
</AbccAttribute>
```

Attribute Name and Data Format		
Attribute Name	**Data Format**	**Description**
AM info Type Location Type	Unsigned 8	The value can be set in either of two formats, 0x12 or 18.
AM Type Identification IM Hardware Revision	Unsigned 16	The value can be set in either of two formats, 0x1234 or 4660.
IM Annotation	String of length X	Maximum number of elements in array: 64.
IM Order ID	String of length X	Maximum number of elements in array: 64.
IM Serial Number	String of length X	Maximum number of elements in array: 16.
AM Software Revision	String of length X	Maximum number of elements in array: 64.
AM Hardware Revision	String of length X	Maximum number of elements in array: 64.
IM Software Revision	String	Format of the string shall be C.X.Y.Z. C is one character. X, Y and Z represent a value between 0 and 255. X – Major version Y – Minor version Z – Internal
IM Unique Identifier	Array of Unsigned 8 Length is 16	Format of the value shall be 0xXX;0xYY…0xZZ. 16 values in hex-format, where each value is separated by a ";".
Location LT	Array of Unsigned 16 Length is up to 12 elements.	Format of the value shall be 0xXXXX;0xYYYY…0xZZZZ. Up to 12 values in hex-format, where each value is separated by a ";".
Location SS AM Device Identification	Array of Unsigned 16 Length is 4.	Format of the value shall be 0xXXXX;0xYYYY…0xZZZZ. 4 values in hex-format, where each value is separated by a ";".

5.5.4 Asset Management XML File Structure Example

The code example presented below can be used as a guide when creating the *asset management* XML file.

```
<AssetManagement Created="2017-01-01 01:01:01">
    <AbccObject>
        <Data Value="0xE5"/>
            <AbccInstance>
                <Data Value="1"/>
                    <AbccAttribute>
                        <Name Value="AM info Type"/>
                        <Attribute Value="1"/>
                        <Data Value="0"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="IM Unique Identifier"/>
                        <Attribute Value="2"/>
                        <Data Value="0x01;0x02;0x03;0x04;0x05;0x06;0x07;0x08;0x09;0x0A;0x0B;0x0C;0x0D;0x0E;0x0F;0x10;"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="Location Type"/>
                        <Attribute Value="3"/>
                        <Data Value="1"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="Location LT"/>
                        <Attribute Value="4"/>
                        <Data Value="0x0001;0x0002;0x0003;0x0004;0x0005;0x0006;0x0007;0x0008;0x0009;0x000A;0x000B;0x000C;"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="Location SS"/>
                        <Attribute Value="5"/>
                        <Data Value="0x0001;0x0002;0x0003;0x0004;"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="IM Annotation"/>
                        <Attribute Value="6"/>
                        <Data Value="64 characters max"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="IM Order ID"/>
                        <Attribute Value="7"/>
                        <Data Value="64 characters max"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="IM Serial Number"/>
                        <Attribute Value="8"/>
                        <Data Value="16 chars max    "/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="AM Device Identification"/>
                        <Attribute Value="9"/>
                        <Data Value="0x0001;0x0002;0x0003;0x0004;"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="AM Type Identification"/>
                        <Attribute Value="10"/>
                        <Data Value="0x0000"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="AM Software Revision"/>
                        <Attribute Value="11"/>
                        <Data Value="64 characters max"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="IM Software Revision"/>
                        <Attribute Value="12"/>
                        <Data Value="V.1.02.03"/>
                    </AbccAttribute>

                    <AbccAttribute>
                        <Name Value="AM Hardware Revision"/>
                        <Attribute Value="13"/>
                        <Data Value="64 characters max"/>
                    </AbccAttribute>

                     <AbccAttribute>
                        <Name Value="IM Hardware Revision"/>
                        <Attribute Value="14"/>
                        <Data Value="0x0000"/>
                    </AbccAttribute>
            </AbccInstance>

    </AbccObject>
</AssetManagement>
```

Fig. 20 Asset management XML file structure example

5.6 Binary Based Asset Management

5.6.1 Creating the Asset Management Binary File

Creating the *asset management* binary file:

1. List all assets and their corresponding data on the non PROFINET network.
2. Create an Binary file that include a *asset management record* for each asset.

   Repeat all the *attributes* after each other.
3. When all *attributes* are listed, close the *element* by using a *closing entry.*
4. Name the bin file *asset_mgmt*.

5.6.2 Binary File Size Limitation

The size of the asset management file may not exceed 12 kb.

32 instances can be added, instance 1 to 32.

In order to keep the file size small, consider the following:

- Keep strings as short as possible.
- Do not pad with empty spaces for strings.

5.6.3 Binary File Header

Omitted attributes are disabled or set to their default value.

The size of the file header is 70 bytes.

The supported *file headers* are specified in the table.

Supported File Headers

File Header	Byte Number	Data Type	Comment
File format version	0-1	UINT16	Version number of the file format. Set to 0.
File checksum	2-5	UINT32	Used for version control of the file. Not used by the gateway. If not used, the field must be set to zero.
Byte offset to Instance 1	6-7	UINT16	Byte offset to the start of the data describing Asset management Instance X. Set to zero if instance is not used.
Byte offset to Instance 2	8-9		
Byte offset to Instance 32	68-69		
Instance data	70-x	N/A	Data for the instance(s), as specified below.

5.6.4 Binary Instance Data

Each *instance* consists of a series of *attributes* and their respective data.

Attribute Description

Each *attribute* is described by one entry.

Attribute Description	Byte number	Data type	Comment
Attribute number	0	UINT8	Attribute number of the data being described.
Data length	1	UINT8	Optional checksum. Shall represent the number of data bytes following. Not used by the gateway.
Attribute data	2–x	Depends on the attribute being described.	Data for the attribute. Format shall be as described for the data-type. Not needed for strings padding or termination.

Attribute Closure Description

Use a *closing entry* to close the instance data.

Attribute Description	Byte number	Data type	Comment
Closure	0–1	UINT16	Data-field which tell that there will not follow any more attributes for this instance. Set to value 0xFFFF.

Attribute Name and Data Format

Supported *attribute names* and *data formats*.

Attribute Name and Data Format

Attribute Name	Data Format	Description
AM info Type Location Type	Unsigned 8	The value is set as one byte value.
AM Type Identification IM Hardware Revision	Unsigned 16	The value is set with two bytes, *little-endian* format.
IM Annotation	String of length X	Maximum number of elements in array: 64.
IM Order ID	String of length X	Maximum number of elements in array: 64.
IM Serial Number	String of length X	Maximum number of elements in array: 16.
AM Software Revision	String of length X	Maximum number of elements in array: 64.
AM Hardware Revision	String of length X	Maximum number of elements in array: 64
IM Software Revision	Array of Unsigned 8 Length is 4	First byte is a character. Bytes 2, 3 and 4 represent the version in the format X.Y.Z where X, Y and Z represent a value between 0 and 255. C is one character. X, Y and Z represent a value between 0 and 255. X – Major version Y – Minor version Z – Internal
IM Unique Identifier	Array of Unsigned 8 Length is 16	Format is 16 bytes.
Location LT	Array of Unsigned 16 Length is up to 12 elements.	Each Unsigned 16 comprises two bytes, where each two bytes form an Unsigned 16 in *little-endian* format. The number of Unsigned 16's can be up to 12, placed directly after each other
Location SS AM Device Identification	Array of Unsigned 16 Length is 4.	Each Unsigned 16 comprises two bytes, where each two bytes form an Unsigned 16 in *little-endian* format. The number of Unsigned 16's shall be 4, placed directly after each other.

5.6.5 Asset Management Binary File Example

The binary file structure example presented below can be used as a guide when creating the *asset management* binary file.

Only *instance 1* is supported.

For *instance 1*, only attribute 1 and 2 are defined.

	0	1	2	3	4	5	6	7
0	0x00	0x00	0x01	0x02	0x03	0x04	0x46	0x00
8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
16	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
24	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
32	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
40	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
48	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
56	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
64	0x00	0x00	0x00	0x00	0x00	0x00	0x01	0x01
72	0x01	0x02	0x10	0x01	0x02	0x03	0x04	0x05
80	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D
88	0x0E	0x0F	0x10	0xFF	0xFF			

Fig. 21 Binary file example

5.7 Uploading the Asset Management File to the FTP Server

Use *Windows Explorer* or a standard *FTP client* to transfer the *asset management file* to the *FTP server.*

When the *superposed parameter channel* function is enabled, transfer the *asset management file* via a PLC connected to the network where the gateway is installed.

5.7.1 Transferring the Asset Management File from Windows Explorer

Transfer the *asset management file*, XML or binary file, to the *FTP server* using *Windows Explorer.*

Before You Begin

Use only one of the file formats, XML format or binary format.

Only upload one single file on the FTP server.

- Name the *asset management file*: *asset_mgmt*
- The default port is FTP port 21.
- Make sure that the gateway and your computer are connected to the PROFINET network to be used.

Procedure

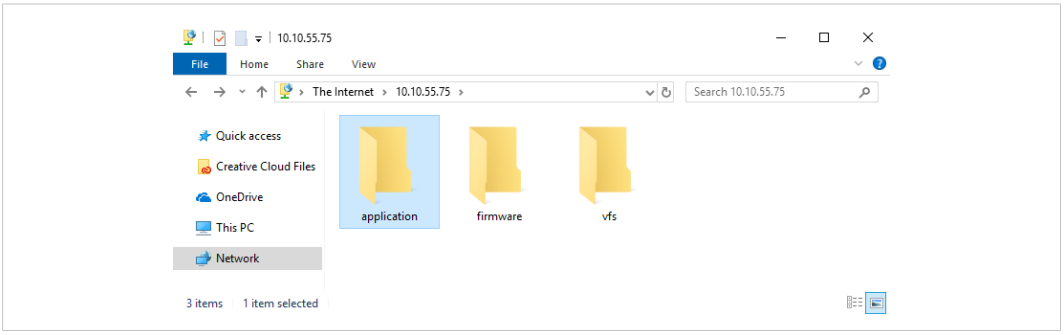


Fig. 22 The FTP Server root folder

1. Open an **Windows Explorer Window**.
2. Click to select the Address bar.
3. Enter **ftp://Username:Password@IPaddress**.
   - Replace “Username” and “Password” with a valid username and password combination.
   - Replace ‘IPaddress’ with the IP address of the PROFINET interface.
4. Press **Enter**.

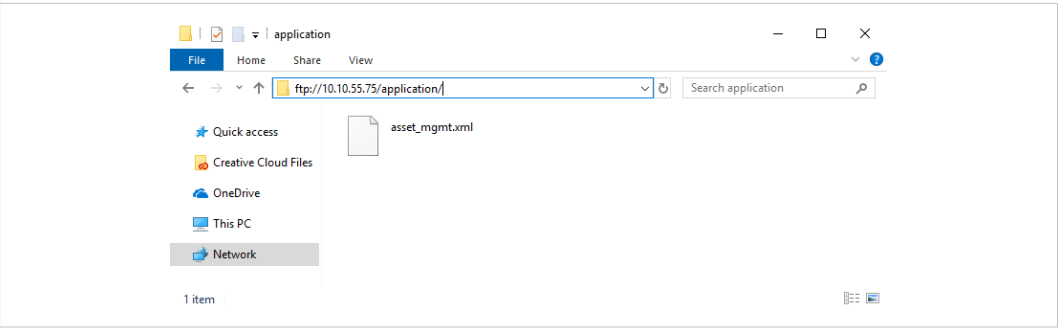


Fig. 23 Application folder with an asset_mgmt.xml file

5. Open the *application* folder and save the *asset management file*, XML or Binary file, in the folder.

6 CANopen Module Specification

6.1 NMT State Machine

CANopen network management is modeled as a state machine with four states. The device can change states due to requests from an NMT object, a hardware reset (power cycle), or a module control message initiated by an application event.

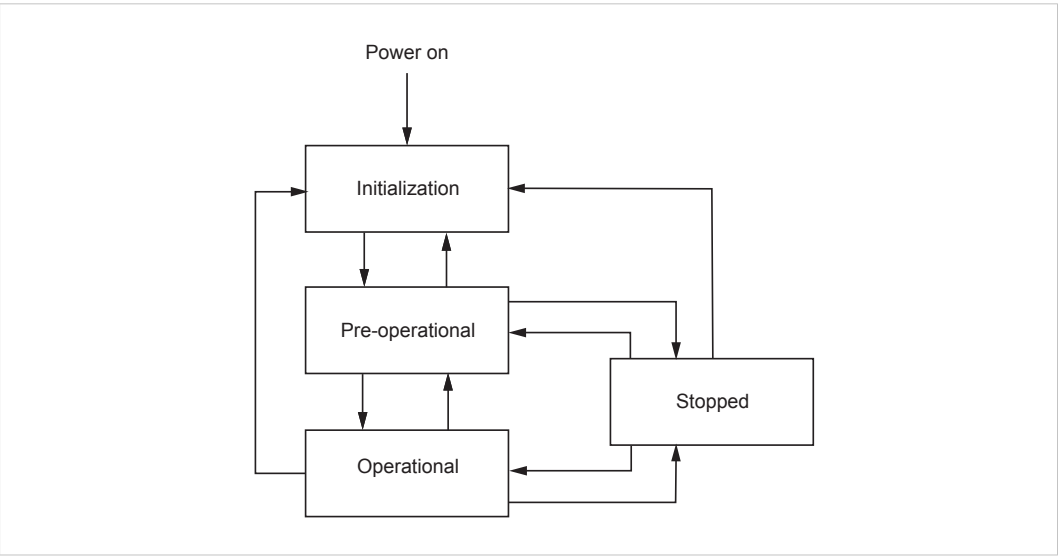


Fig. 24 CANopen NMT state machine

Initialization	When power is applied (or on reset), the device will enter the Initialization state. All parameters will be set to their latest stored values. If there is no stored value for a parameter, the default value from the device profile is used. When initialization has finished, the device will transit to the Pre-operational state.
Pre-operational	The Pre-operational state is primarily used for configuration. It allows SDO, EMCY, SYNC and NMT control messages to be transmitted, but not process data (PDOs). When the device has entered the Pre-operational state it will send a boot-up message, indicating that it is ready to transit to the Operational state. The device can also go back to Initialization or to the Stopped state.
Operational	In the Operational state all supported communication objects are allowed. PDOs (I/O data) will be communicated according to the active configuration. The device can transit from the Operational state to any of the other states.
Stopped	In this state all data communication is stopped. Only Heartbeat, Node Guarding, and NMT control messages are possible. The device can transit from the Stopped state to any of the other states.

6.2 CANopen Data Exchange

The first two bytes of the I/O data area are reserved for the Control Word and the Status Word, which are used by the controlling PLC on the primary network to control and report status on the nodes on the secondary CANopen network. The remainder is available for real-time data transfer using PDOs. The amount of data that is exchanged as I/O data is specified when configuring the CANopen master interface.

6.2.1 Control Word

Control Word				**Effective I/O Data**
Byte 0			Byte 1	Byte 2–510
Toggle bit	Cmd (3 bits)	CmdExt (4 bits)	NodeID	Data

The MSB in byte 0 (toggle bit) is toggled each time a new command is issued.

If NodeID = 0, the command is valid only for the module itself. If NodeID = 128 (80h), the command is valid for the whole secondary CANopen network. If the module is configured as a slave, the only allowed value of NodeID is 0.

Supported Commands (Cmd + CmdExt)

The function of the Control Word differs depending on if the secondary CANopen interface is configured as a slave or a master.

Cmd	**CmdExt**	**Name**	**Master**	**Slave**
0h	–	(Set NMT State)	Sets the state of a single node or the whole network according to the value of NodeID.	State is set by the controlling PLC. If the PLC is running, the state will be set to OPERATIONAL, otherwise to PRE-OPERATIONAL.[1] Default = PRE-OPERATIONAL
	0h	PRE-OPERATIONAL		
	1h	OPERATIONAL		
	2h	RESET NODE		
	3h	RESET COMMUNICATION		
	4h	STOP		
	5h–Fh	–	(reserved)	
1h	–	Get Node state	Requests the state in object 1F82h of the node or network (if monitored by Node Guarding or Heartbeat).	
2h	–	Get COPM general status	Requests the general status of the CANopen module	
3h–6h	(reserved)			
7h	–	(No operation)	Setting Cmd to this value when the module goes offline will prevent unpredictable behavior when it comes back online.	

RESET NODE will restore the last stored configuration, RESET COMMUNICATION will restore communication settings. Both commands will return the module to the INITIALIZATION state.

Examples		
Master/Slave	**Control Word**	**Meaning**
Slave	01 00h	01h: Allow the module to go to OPERATIONAL if asked by an NMT master. 00h: The command is only valid for the module itself.
Master	01 80h	01h: Start remote node in the secondary network. 80h: The command is valid for all nodes in the secondary network.
	01 02h	01h: Start remote node. 02h: The command is valid for node 2.
	04 80h	04h: Stop remote node. 80h: The command is valid for all nodes in the secondary network.

1. The controlling PLC must set the X-gateway to OPERATIONAL using the Control Word, otherwise the X-gateway will decline an NMT Set Operational Command on the secondary network with error code FF10h. The same error code will be sent if the X-gateway is reset to PRE-OPERATIONAL by the primary network.

6.2.2 Status Word

Byte 0 in the status word shows the last valid command and command extension written to the Control Word, to indicate that the command has been performed. Byte 1 gives the lowest NodeID with error. Please note that there can be one or more nodes, with higher NodeIDs, that also have errors. If NodeID is 0, all nodes are fine. If NodeID is (for example) 5, it means that there is an error with node 5.

Only errors from nodes monitored by the heartbeat mechanism or by node guarding will be reported. Errors from other slaves cannot be recognized.

Status Word				**Effective I/O Data**
Byte 0			Byte 1	Byte 2–510
Toggle bit	CmdRsp (3 bits)	CmdExtRsp (4 bits)	Error Node	Data

The MSB in byte 0 (toggle bit) is toggled to mirror the toggle bit of the Control Word.

Supported Commands

The available command responses and their representation in byte 0 of the status word.

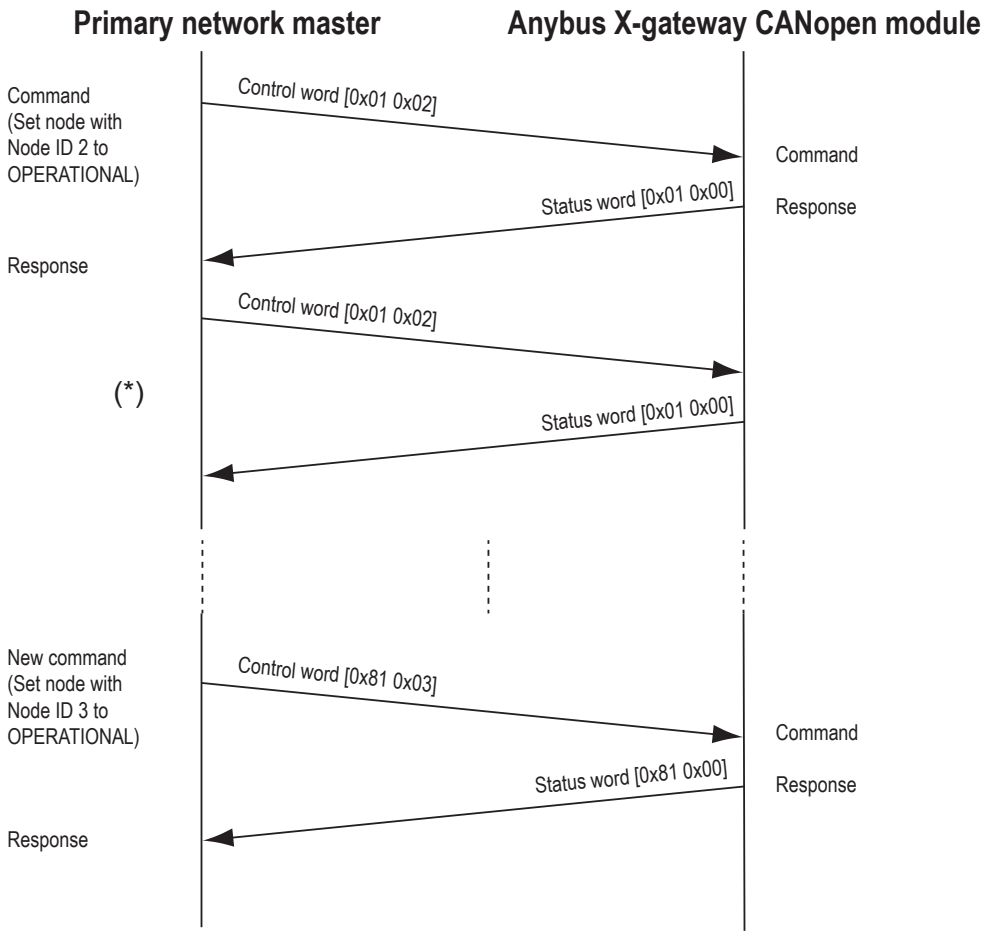
CmdRsp (3 bits)	**CmdExtRsp (4 bits)**	**Name**	**Master**	**Slave**
0h	-	(Set NMT State)	Response to Set NMT State command. Reflects the command.	Response to Set NMT State command. Reflects the command.
	0h	PRE-OPERATIONAL		
	1h	OPERATIONAL		
	2h	RESET NODE		
	3h	RESET COMMUNICATION		
	4h	STOP		
	5h - Fh	-	(reserved)	
1h	-	(Get Node state)	Reflects the state set in object 1F82h of a CANopen node or network (depending on the value of NodeId).	Reflects the state set in object 1F82h of the CANopen interface.
	0h	PRE-OPERATIONAL		
	1h	OPERATIONAL		
	2h	RESET NODE		
	3h	RESET COMMUNICATION		
	4h	STOP		
	5h	UNKNOWN		
	6h	MISSING		
	7h - Eh	-		
	Fh	ERROR		
2h	Bit:	(Get COPM general status)	Requests the CANopen status of the module	
	0	CAN_BUS_OFF	Bus off	
	1	CAN_ERR_PASV	Error passive	
	2	ERR_NG_HB	Node guarding or Heartbeat error	
	3	ERR_SYNC	Sync error	
3h - 6h	(reserved)			
7h	-	(No operation)	Reflects the command	

6.2.3 Control/Status Word Example

This example shows two control words sent from the master on the primary network to the Anybus X-gateway CANopen.

Each Control Word includes a command that affects the secondary CANopen network, and is acknowledged by a status word containing a response to the command.

The first bit in the Control Word is toggled when a new command is sent, to make sure it is distinguished from the previous command.



(*) The communication is performed in an ongoing ping-pong fashion. The same command is sent repeatedly, but as long as the toggle bit is not changed, the module will ignore the message. A new command is signalled by the toggle bit changing value.

Fig. 25 Control/Status Word example

6.2.4 PDO Functionality

Real-time data transfer is performed by means of PDOs (Process Data Objects). The PDOs are linked to entries in the Device Object Dictionary and provide the interface to the application objects. The number and length of PDOs in a device are node specific and have to be configured by the CANopen configuration tool.

PDOs are used both for data transmission and reception, using so called Transmit-PDOs (TPDOs) and Receive-PDOs (RPDOs). Each PDO corresponds to two entries in the Device Object Dictionary. The PDO parameter object holds information on the COB-ID, the transmission type etc. On recognition of the COB-ID the corresponding PDO mapping object can be identified, to make it possible to transmit/receive data to/from the correct object in the device.

The default settings for the PDO mapping can be changed during configuration.

See also *Manufacturer Specific Objects, p. 52*.

Default PDO Mapping Scheme

The default mapping scheme contains 4 TPDOs and 4 RPDOs.

RPDO	Default COB ID	Mapped to	Relating to	Default State
1	200h + NodeID	Object index 2100h, subindex 1–8	Receive bytes 2–9	Enabled
2	300h + NodeID	Object index 2100h, subindex 9–16	Receive bytes 10–17	
3	400h + NodeID	Object index 2100h, subindex 17–24	Receive bytes 18–25	
4	500h + NodeID	Object index 2100h, subindex 25–32	Receive bytes 26–33	
5	80000000h	Object index 2100h, subindex 33–40	Receive bytes 34–41	Disabled
...		...	...	
128		Object index 2103h, subindex 121–126	Receive bytes 506–511	

TPDO	Default COB IDs	Mapped to	Relating to	Default State
1	180h + NodeID	Object index 2000h, subindex 1–8	Transmit bytes 2–9	Enabled
2	280h + NodeID	Object index 2000h, subindex 9–16	Transmit bytes 10–17	
3	380h + NodeID	Object index 2000h, subindex 17–24	Transmit bytes 18–25	
4	480h + NodeID	Object index 2000h, subindex 25–32	Transmit bytes 26–33	
5	80000000h	Object index 2000h, subindex 33–40	Transmit bytes 34–41	Disabled
...		...	...	
128		Object index 2003h, subindex 121–126	Transmit bytes 506–511	

RPDO Transmission Types

The RPDOs can be received either in synchronous or asynchronous mode. A synchronization (SYNC) object is transmitted periodically by a synchronization master. The data in synchronous RPDOs are not transferred to the application until after the next SYNC object is received. Asynchronous RPDOs will be transferred directly.

The transmission type parameter of a RPDO specifies the triggering mode.

Transmission type	Mode	RPDO transmission description
0–240	Synchronous	A received RPDO is transferred to the application after a SYNC object is received.
241–253	-	(reserved)
254–255 (Default = 255)	Event driven	An RPDO is transmitted without any relation to the SYNC object.

TPDO Transmission Types

The TPDOs can be transmitted either in synchronous or asynchronous mode. A synchronization (SYNC) object is transmitted periodically by a synchronization master. Synchronous TPDOs are transmitted within a predefined time-window immediately after a configured number of SYNC objects, or after the SYNC object that follows upon a CoS (Change of State event). Asynchronous TPDOs can be transmitted at any time, triggered by a CoS or a cyclic period set in the Event Timer.

The transmission type parameter of a TPDO specifies the transmission mode as well as the triggering mode.

Transmission type	Mode	TPDO transmission description
0	Synchronous, acyclic	A TPDO is triggered by an event, but not transmitted before the occurrence of a SYNC object.
1–240	Synchronous, cyclic	A TPDO is transmitted with every n-th SYNC object, where n is a defined number from 1 - 240.
241–253	-	(reserved)
254–255 (Default = 255)	Event driven	A TPDO is transmitted without any relation to the SYNC object. The transmission is triggered by a CoS event or if a specified time has elapsed without an event.

6.3 LSS Services

Anybus X-gateway CANopen supports LSS Master functionality according to the CANopen DS305 specification. An LSS Master can configure the baud rate and NodeID of all slaves that support LSS. The X-gateway can not act as an LSS slave.

An LSS Slave is identified by its LSS address, which consists of the Vendor ID, Product Code, Revision Number and Serial Number of the slave. If there is a missing slave on the network after the boot timeout, the master will send an *identify slave* request, using the LSS address of the missing slave. If a node responds to this request, the master will set its NodeID to the first missing NodeID. It will then send a bootup message to that node. If more than one node responds to the request, the LSS routine will fail.

6.4 Error Control

It is strongly recommended to monitor the CANopen network. Anybus X-gateway CANopen can use either *Heartbeat* or *Node Guarding* for monitoring. At an error event from any of these mechanisms the active I/O data will be frozen, as no new data will be available.

6.4.1 Heartbeat

The heartbeat mechanism is used to monitor the nodes in the network and verifies that the nodes are available. A heartbeat producer periodically sends a message. The data part of the frame contains a byte indicating the node status. The heartbeat consumer reads these messages. If a message fails to arrive within a certain time limit (defined in the object directory of the devices, objects 1016h and 1017h), a heartbeat event is registered by the consumer. The ERROR LED on the front of the Anybus X-gateway CANopen and the status word will indicate the event. An EMCY object (8130h) is also transmitted on the CANopen fieldbus. If the module is configured as a slave and is in OPERATIONAL state, it will go to PRE-OPERATIONAL state and wait for the user to take action. If it is in master mode, it will take action according to the settings in the master objects.

Anybus X-gateway CANopen can act both as heartbeat consumer and as heartbeat producer simultaneously.

6.4.2 Node Guarding

The NMT Master transmits guarding requests. If an NMT Slave has not responded within a defined time span (node lifetime) or if the communication status of the slave has changed, the master takes appropriate action according to its configuration.

If Life guarding (the slave guards the master) is supported, the slave uses the guard time and lifetime factor from its Object Dictionary to determine the node lifetime. If the slave does not receive a guarding request within its lifetime, a node guard event is registered. The ERROR LED on the front of the Anybus X-gateway CANopen will indicate the event. An EMCY object (8130h) is also transmitted on the CANopen fieldbus.

If the guard time or the lifetime factor are 0 (default), the Slave does not guard the Master. The guarding can be initiated at boot-up or later.

Only one of the Heartbeat or Node Guarding mechanisms can be active. If Heartbeat is enabled in a slave (preferred), Node Guarding will be disabled.

6.4.3 Emergency Object (EMCY)

The Emergency Object is used for error reporting on the CANopen network when a fatal fault has occurred. The error codes are saved in a list in Communication Profile Object 1003h, and an Emergency Message is produced on the CANopen network.

See also *CANopen Emergency Messages, p. 41*.

6.5 CANopen Emergency Messages

As CANopen is not a hierarchical master-slave system, and node monitoring only conveys the communication state and not the actual node status, every node requires a high priority CAN identifier to indicate error situations. This mechanism is referred to as *Emergency Messaging* and the associated communication object *Emergency Message*.

An emergency message always consists of 8 data bytes (words). The first 2 words are used for the error code, the third word contains a copy of the error register value (1001h), and the remaing 5 words are reserved for vendor specific information.

0	1	2	3	4	5	6	7
Error code		Register	Vendor specific error field				

In Anybus X-gateway CANopen the error codes can be read from the list in the Communication Profile Object (CPO) at index 1003h. The following error codes are supported:

Error code	Description
0000h	Error reset or no error
6161h	Software error (master mode only) – see below
6600h	Hardware error
8110h	CAN overrun (objects lost)
8120h	CAN in error passive mode
8130h	Life guard error or heartbeat error
8140h	Recovered from bus off
8210h	PDO not processed due to length error
8220h	PDO length exceeded
FF10h	State error (slave mode only) — Indicates either of the following errors: • The module is in OPERATIONAL state although the Control Word is set to disallow this. • A CANopen master attempts to set the module in OPERATIONAL state although the Control Word is set to disallow this.

Error code 6161h will write an additional error code in the second word (bit 16–31):

31 16		15 0	
Additional Information		**Error code**	
Error Code	NodeID (if available)	61	61

Error code	Description
00h	No software error detected
01h	Tag for CMT record not available
02h	Cache management inconsistent
03h	SDO could not be transmitted
04h	Configuration entry inconsistent
05h	Checksum error
06h	Data could not be written to non-volatile memory
07h	SDO timeout
08h	SDO error

6.6 CANopen Live List Functionality

Anybus X-gateway CANopen provides a list of the active status of the slave nodes attached to the secondary network CANopen master. The list is assembled by the master and forwarded to the primary network during each cycle.

This feature is disabled by default and can be enabled through the CANopen network configuration software, see *Enabling Live List in Anybus Configuration Manager, p. 43*.

Live List Usage

The Live List consists of 16 bytes and holds bit coded status information for CANopen slave nodes 1–127.

Bit set (1) Device active — State OPERATIONAL

Bit cleared (0) Device not active — States INITIALIZATION, PRE-OPERATIONAL or STOPPED.

Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Device number							
0	7	6	5	4	3	2	1	ALL[a]
1	15	14	13	12	11	10	9	8
2	23	22	21	20	19	18	17	16
3	31	30	29	28	27	26	25	24
4	39	38	37	36	35	34	33	32
5	47	46	45	44	43	42	41	40
6	55	54	53	52	51	50	49	48
7	63	62	61	60	59	58	57	56
8	71	70	69	68	67	66	65	64
9	79	78	77	76	75	74	73	72
10	87	86	85	84	83	82	81	80
11	95	94	93	92	91	90	89	88
12	103	102	101	100	99	98	97	96
13	111	110	109	108	107	106	105	104
14	119	118	117	116	115	114	113	112
15	127	126	125	124	123	122	121	120

a. Bit 0 in Offset 0 indicates 1 if all monitored nodes are in OPERATIONAL state, else it will be 0.

Only operational/non-operational state will be presented in the Live List. Error passive state will not be detected, but BUSOFF will also result in heartbeat/node guarding errors and will be reflected in the live list.

The control/status word allocates 2 bytes in the input/output area by default and is always enabled. If the Live List is enabled, an additional 16 bytes will be allocated in the IO area, starting at the next available address after the status word.

Enabling Live List in Anybus Configuration Manager

The Live List feature is available in Object 0x3003 (Live List Enable) in *Anybus Configuration Manager CANopen*. The feature is disabled by default and is enabled by setting the parameter **Object Value:Value** to **1**.

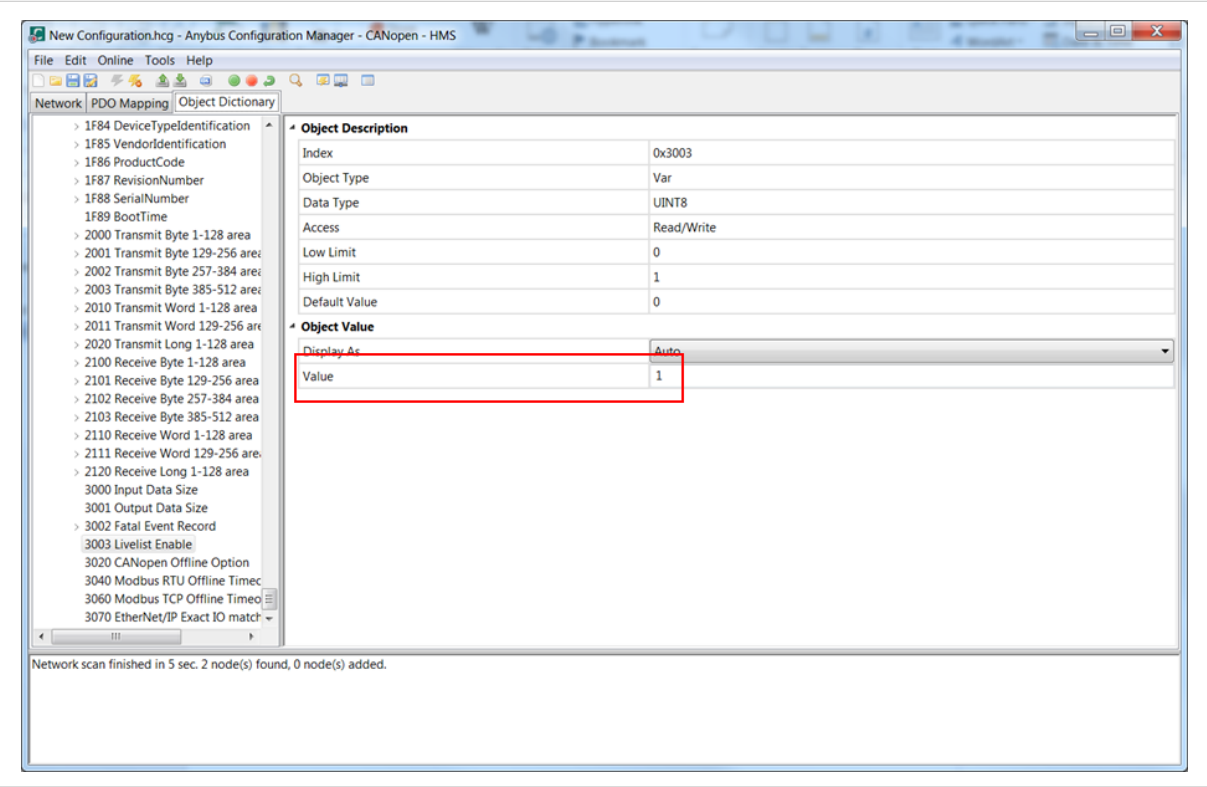


Fig. 26 Anybus Configuration Manager CANopen

I/O Assembly Examples

Data to Primary Network with Live List Disabled

Byte	Content							
0–7	Status word	Status word	Data	Data	Data	Data	Data	Data
8–15	Data	Data	Data	Data	Data	Data	Data	Data

Data to Primary Network with Live List Enabled

Byte	Content							
0–7	Status word	Status word	Nodes ALL, 1...7[a]	Nodes 8...15	Nodes 16...23	Nodes 24...31	Nodes 32...39	Nodes 40...47
8–15	Nodes 48...55	Nodes 56...63	Nodes 64...71	Nodes 72...79	Nodes 80...87	Nodes 88...95	Nodes 96...103	Nodes 104...111
16–23	Nodes 112...119	Nodes 120...127	Data	Data	Data	Data	Data	Data
24–31	Data	Data	Data	Data	Data	Data	Data	Data

a. If all monitored nodes are in OPERATIONAL state this bit will be 1, else it will be 0.

7 CANopen Object Implementation

This section describes the CANopen objects that are implemented in the secondary CANopen network according to the indicated CANopen specifications and the latest Anybus X-gateway CANopen EDS file.

7.1 Static Data Types

Implemented according to CiA Draft Standard 301 version 4.2.0.

7.2 Communication Profile Area

7.2.1 Communication Profile Objects

According to CiA Draft Standard 301 version 4.2.0

Index	Object Name	Subindex	Description	Type	Access	Notes
1000h	Device Type	00h	Type of device	U32	RO	0000 0000h (No profile)
1001h	Error register	00h	Error register, connected to the EMCY object. Bit 0 indicates a generic error	U8	RO	See *CANopen Emergency Messages, p. 41*
1003h	Predefined error field	00h	Number of errors. Writing 0 clears the error list.	U8	RW	
		01h–10h	List of errors. Most recent error at top of list.	U32	RO	
1005h	COB-ID Sync	00h	ID of the sync message	U32	RW	-
1006h	Communication Cycle Period	00h	Communication cycle period	U32	RW	Only available if SYNC support is enabled
1007h	Synchronous Window Length	00h	Synchronous Window Length	U32	RW	Only available if SYNC support is enabled
1008h	Manufacturer device name	00h	The name of the CANopen module	Visible string	RO	“Anybus X-gateway CANopen”
1009h	Manufacturer hardware version	00h	Manufacturer hardware version	Visible string	RO	Current hardware revision
100Ah	Manufacturer software version	00h	Manufacturer software version	Visible string	RO	Set by HMS
100Ch	Guard time	00h	Used together with "Life time factor" to decide the node lifetime in ms	U16	RW	0000h (default)
100Dh	Life time factor	00h	If the node has not been guarded within its lifetime (*Life time factor* * *Guard time*), an error event is logged and a remote node error is indicated	U8	RW	00h (default)
1010h	Store Parameters	00h	Largest subindex supported	U8	RO	01h
		01h	Store all parameters	U32	RW	To save a configuration, write “save” = 73 61 76 65h to this object. [2] See also *General Fieldbus Parameters, p. 56*.

2. The byte order may have to be changed depending on the method of writing to this object.

Index	Object Name	Subindex	Description	Type	Access	Notes
1011h	Restore Parameters	00h	Largest subindex supported	U8	RO	01h
		01h	Restore all parameters	U32	RW	To restore the default values of a configuration, write “load” = 6C 6F 61 64h to this object.[3]
1014h	COB-ID EMCY	00h	Defines the COB-ID of the Emergency Object	U32	RO	
1016h	Consumer Heartbeat Time	00h	Largest subindex supported	U8	RO	7Fh
		01h - 80h	The consumer heartbeat time defines the expected heartbeat cycle time and has to be higher than the corresponding producer heartbeat time. Monitoring starts after the reception of the first heartbeat. Not used if 0	U32	RW	NodeID + Heartbeat Time. Bits 31–24: reserved Bits 23–16: NodeID Bits 15–0: Heartbeat Time Value must be a multiple of 1 ms. Up to 127 nodes can be monitored.
1017h	Producer Heartbeat Time	00h	Defines the cycle time of the heartbeat. Not used if 0	U16	RW	The time must be at least 10 ms and a multiple of 1 ms
1018h	Identity object	00h	Number of entries	U8	RO	04h
		01h	Vendor ID	U32	RO	1Bh (HMS Industrial Networks)
		02h	Product Code	U32	RO	18h (Anybus X-gateway CANopen)
		03h	Revision Number	U32	RO	Current software revision
		04h	Serial Number	U32	RO	HMS serial number
1029h	Error behavior object	00h	Number of entries	U8	RO	
		01h	Communication error	U8	RO	00h: Change to Preoperational if currently in NMT state Operational
		02h	Profile or manufacturer specific error	U8	RO	00h: Change to Preoperational if currently in NMT state Operational
1400h ... 147Fh	Receive PDO parameter	00h	Largest subindex supported	U8	RO	02h
		01h	COB-ID used by PDO	U32	RW	-
		02h	Transmission type	U8	RW	See *PDO Functionality, p. 38*
1600h ... 167Fh	Receive PDO mapping	00h	No. of mapped application objects in PDO	U8	RW	-
		01h	Mapped object #1	U32	RW	-
		02h	Mapped object #2	U32	RW	-
		03h	Mapped object #3	U32	RW	-
		04h	Mapped object #4	U32	RW	-
		05h	Mapped object #5	U32	RW	-
		06h	Mapped object #6	U32	RW	-
		07h	Mapped object #7	U32	RW	-
		08h	Mapped object #8	U32	RW	-
1800h ... 187Fh	Transmit PDO parameter	00h	Largest subindex supported	U8	RO	05h
		01h	COB-ID used by PDO	U32	RW	-
		02h	Transmission type	U8	RW	See *PDO Functionality, p. 38*
		03h	Inhibit time	U16	RW	In steps of 0.1 ms
		05h	Event Timer (ms)	U16	RW	-

3. The byte order may have to be changed depending on the method of writing to this object.

Index	Object Name	Subindex	Description	Type	Access	Notes
1A00h ... 1A7Fh	Transmit PDO mapping	00h	No. of mapped application objects in PDO	U8	RW	-
		01h	Mapped object #1	U32	RW	-
		02h	Mapped object #2	U32	RW	-
		03h	Mapped object #3	U32	RW	-
		04h	Mapped object #4	U32	RW	-
		05h	Mapped object #5	U32	RW	-
		06h	Mapped object #6	U32	RW	-
		07h	Mapped object #7	U32	RW	-
		08h	Mapped object #8	U32	RW	-

7.2.2 Configuration Manager

According to CiA Draft Standard Proposal 302 Part 3

Network Configuration Objects					
Index	**Object Name**	**Subindex**	**Description**	**Type**	**Access**
1F22h	Concise DCF	-	The concise/compressed DCF files information is stored in this object.	Domain	RW
1F25h	Configure Slave	0 - 128	Subindex 0 is ignored. Subindex i (i = 1 - 127): Request reconfiguration of slave with NodeID equal to subindex i. Subindex 128: Request to reconfigure all slaves.	U32	Sub 0: RO Sub 1 - 128: WO

To configure the slave with NodeID i, write “conf” = 63 6F 6E 66h to object 1F25h, subindex i.

If this fails, emergency code 6161h is produced (see *CANopen Emergency Messages, p. 41*).

Check Configuration

The Configuration Manager (CMT) compares signature and configuration with the value from the DCF to decide if a reconfiguration is to be performed or not. The comparison values are stored by the Configuration Manager in these objects:

Index	Object Name	Subindex	Description	Type	Access
1F26h	Expected Configuration Date	0 - 127	The date that the Configuration Manager expects to find when comparing signature and configuration.	UINT32	RW
1F27h	Expected Configuration Time	0 - 127	The time that the Configuration Manager expects to find when comparing signature and configuration.	UINT32	RW

7.2.3 Network Management Objects

The NMT master controls the states of the connected network participants, the NMT slaves. It monitors the devices and reports to the application, for example if an NMT slave fails. In more complex systems several devices are able to perform as master, which means that the configuration must have an entry defining which device will act as master.

Once configured, the objects carry all information needed for the module to act on the network, and the application does not need to be accessed to obtain this information. This simplifies the implementation and maintenance of multiple applications considerably.

Index	Object Name	Subindex	Description	Type	Access
1F80h	NMT Start-up	-	Defining whether the device is the NMT Master	U32	RW
1F81h	Slave Assignment	ARRAY	Module list: Entry of all slaves to be managed, including guarding values and the entry of actions to be taken in event of guarding errors.	U32	Sub 0: RO Sub 1 - 127: RW
1F82h	Request NMT	ARRAY	Remote control initiation of NMT services. For example, tools can use this to request intentional start/stop of individual slaves. Remote query of the current state.	U8	Sub 0: RO Sub 1 - 127: RW Sub 128: WO
1F83h	Request Guarding	ARRAY	Remote control start/stop of guarding. Remote query of the current state	U8	Sub 0: RO Sub 1 - 127: RW Sub 128: WO
1F84h	Device Type Identification	ARRAY	Expected device types for the slaves	U32	Sub 0: RO Sub 1 - 127: RW
1F85h	Vendor Identification	ARRAY	Vendor identifications for the slaves	U32	Sub 0: RO Sub 1 - 127: RW
1F86h	Product Code	ARRAY	Product codes for the slaves	U32	Sub 0: RO Sub 1 - 127: RW
1F87h	Revision Number	ARRAY	Revision numbers for the slaves	U32	Sub 0: RO Sub 1 - 127: RW
1F88h	Serial Number	ARRAY	Expected serial numbers for the slaves	U32	Sub 0: RO Sub 1 - 127: RW
1F89h	Boot Time	VAR	The maximum time between the start of the boot process and the signalling of successful boot of all mandatory NMT slaves. After this time LSS services are initiated.	U32	RW

NMT Start-up, 1F80h

If the Anybus X-gateway CANopen should function as NMT Master, the master functionality must be enabled in this object. The object will configure the start-up behavior of the device and how it will manage the slaves.

The X-gateway starts up as a slave as default (Bit 0 = 0). To make it function as a master, change the value of Bit 0 to 1 and save the configuration by issuing the *save* command to subindex 01h in object 1010 (Store Parameters). The setting will take immediate effect but will revert to the default value (slave) on the next reset/reboot unless it has been saved.

Bit	Value	Description	Notes
0	0	NMT Master functionality is disabled. Ignore the rest of the object, except for bits 1 and 3. Ignore object 1F81h.	Default
	1	NMT Master functionality is enabled. The device is Master.	
1	0	Start only explicitly assigned slaves (if bit 3 = 0)	Default
	1	After boot-up, perform the service NMT Start Remote Node All Nodes (if bit 3 = 0)	
2	0	Automatically enter Operational state	Default
	1	Do not enter Operational state automatically. Application will decide when to enter Operational state	
3	0	Start-up of slaves allowed (i.e. allowed to send NMT Start Remote Node command)	Default
	1	Not allowed to send NMT Start Remote Node command. The application will start the slaves	
4	0	If a mandatory slave generates an Error Control Event, treat the slave individually	If bit 6 = 1, ignore bit 4
	1	If a mandatory slave generates an Error Control Event, perform NMT Reset All Nodes (including self)	
5	-	Not implemented	
6	0	If a mandatory slave generates an Error Control Event, treat the slave according to bit 4	
	1	If a mandatory slave generates an Error Control Event, send NMT Stop All Nodes (including self). Ignore bit 4	
7 - 31	-	Reserved (0)	

Note: If object 1F81h bit 3 = 1, the network must not be restarted if a mandatory slave could not be contacted.

Slave Assignment, 1F81h

This object defines which slaves the Master should monitor, control and/or configure. One entry is made for each assigned slave, with the subindex corresponding to the slave's NodeID.

Bit	Value	Description
0	0	Node with this ID is not a slave.
	1	Node with this ID is a slave. After configuration the node will be set to *Operational*.
1	-	Reserved
2	0	On an Error Control Event or on detection of a new slave, inform the application, but do NOT configure and start the slave.
	1	On an Error Control Event or on detection of a new slave, inform the application and start the process *Start Boot Slave*.
3	0	Optional slave. The network may be started even if this node could not be contacted.
	1	Mandatory slave. The network must not be started if this node could not be contacted during the boot slave process.
4	-	Not implemented
5	-	Not implemented
6	-	Not implemented
7	0	CANopen device may be used without reset to default.
	1	CANopen device shall be reset to factory defaults by issuing a restore to defaults (object 1011h).
8 - 15	-	8 bit value for the *RetryFactor.*
16 - 31	-	16 bit value for the *GuardTime*. If a slave does not answer, the master will retry the request *RetryFactor -1* times with an interval of *GuardTime*. Guarding will be performed only if non-zero values are entered for *Retry Factor* and *GuardTime*.

Request NMT, 1F82h

Each node on the CANopen network can be controlled individually from the fieldbus application by sending this object. The subindex indicates what nodes the request affects:

Subindex	Description
0	Largest subindex supported (128)
i (with i = 1...127)	Request NMT Service for the slave with NodeID i.
128	Request NMT Service for all nodes

The entire network can be started with one command (subindex 128)

Examples:

- Node 5 should be transferred to the OPERATIONAL state:

  An SDO write access with the value 5 is executed to object 1F82h subindex 5 in the local object dictionary. When an NMT command is sent, data is cleared.

- All nodes in the network should be transferred to the PRE-OPERATIONAL state:

  An SDO write access with the value 127 is executed to object 1F82h subindex 128 in the local object dictionary.

Request Guarding, 1F83h

Guarding can be initiated from the object dictionary in a similar way. Guarding is initiated with the values stored in *Slave Assignment, 1F81h, p. 49*, provided that at the same time no parameters are entered for that node as a Heartbeat Consumer.

This functionality is only available in master mode.

Subindex	Description	Access
0	Largest subindex supported (128)	RO
i (with i = 1...127)	Request Guarding for the slave with NodeID i	RW
128	Request Start/Stop Guarding for all nodes.	WO

Example:

- Guarding should be started for node 5 (500 ms, Life Time Factor 3):

  An SDO write access with the value 01F40301h is executed to object 1F81h subindex 5 in the local object dictionary. Guarding is activated by an SDO write access with the value 1 to object 1F83h subindex 5 in the local object dictionary.

Bits	Value	Explanation
31 - 16	01F4h (500)	The interval with which node 5 will be guarded
15 - 8	03h	If node 5 does not answer the guarding will be repeated another RetryFactor -1 times (in this case twice)
7 - 0	01h	This value indicates that node 5 is a slave

Device Type Identification, 1F84h

Each node on the CANopen network is checked against its expected device type. The subindex indicates which node is checked:

Subindex	Description
0	Largest subindex supported (127)
i (with i = 1...127)	If the expected device type is not 0 or if the slave is set as mandatory, the module compares expected device type with actual device type (object 1000h, subindex 0) for the slave with NodeID i. If the expected device type is 0, this only gives information about the existence of a node, not which device type it is. If the value is not 0, it is compared to the value read from the node, and boot up of that slave is continued if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL.

Vendor Identification, 1F85h

Each node on the CANopen network is checked against its expected vendor. The subindex indicates which node is checked:

Subindex	Description
0	Largest subindex supported (127)
i (with i = 1...127)	Compares expected vendor with actual vendor (object 1018h, subindex 1) for the slave with NodeID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL.

Product Code, 1F86h

Each node on the CANopen network is checked against its expected product code. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

Subindex	Description
0	Largest subindex supported (127)
i (with i = 1...127)	Compares expected product code with actual product code (object 1018h, subindex 2) for the slave with NodeID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL.

Revision Number, 1F87h

Each node on the CANopen network is checked against its expected revision number. The revision number includes major and minor revision. For a match to occur the major revision has to be exactly the same and the minor revision of the module has to be greater than or equal to the expected minor revision number. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

Subindex	Description
0	Largest subindex supported (127)
i (with i = 1...127)	Compares expected revision number with actual revision number (object 1018h, subindex 3) for the slave with NodeID i. Boot up of that slave is continued only if they match according to the description above.

Serial Number, 1F88h

Each node on the CANopen network is checked against its expected serial number. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

Subindex	Description
0	Largest subindex supported (127)
i (with i = 1...127)	Compares expected serial number with actual serial number (object 1018h, subindex 4) for the slave with NodeID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL.

Boot Time, 1F89h

The network master will wait the assigned time (in ms) for all mandatory slaves to boot. If not all mandatory slaves are ready after this time, the LSS routine will be started, see *LSS Services, p. 39*. If the assigned time is 0, the master will wait endlessly.

Value (ms)	Description
0	Default. No time limit for mandatory slaves to boot
> 0	Time limit for mandatory slave to boot

7.3 Manufacturer Specific Objects

Manufacturer-specific objects are used to configure the PDOs to the shared memory area. One or several generic data objects can be connected to each PDO.

Data in the Input/Output Buffers will be mapped to three different areas in the Local Object Dictionary: *Byte*, *Word* and *Long*, using different data types.

See also *Local Object Dictionary Index Mapping, p. 54*.

7.3.1 Input Buffer

Contains data transmitted **to** the secondary CANopen network.

Index	Subindex	Type	Access	Name	Position in Input Data area (byte)
2000h	-	STRUCT		Transmit Byte 1-128	2-129 (0 and 1 reserved for Control Word)
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Transmit Byte 1	2
	2	U8	RW	Transmit Byte 2	3
	...	...	...	...	,,,
	128	U8	RW	Transmit Byte 128	129
2001h	-	STRUCT		Transmit Byte 129-256	130-257
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Transmit Byte 129	130
	2	U8	RW	Transmit Byte 130	131
	...	...	...	...	...
	128	U8	RW	Transmit Byte 256	257
2002h	-	STRUCT		Transmit Byte 257-384	258-385
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Transmit Byte 257	258
	2	U8	RW	Transmit Byte 258	259
	...	...	...	...	...
	128	U8	RW	Transmit Byte 384	385
2003h	-	STRUCT		Transmit Byte 385-510	386-511
	0	U8	RO	Number of entries (value=126)	
	1	U8	RW	Transmit Byte 385	386
	2	U8	RW	Transmit Byte 386	387
	...	...	...	...	...
	126	U8	RW	Transmit Byte 510	511
2010h	-	STRUCT		Transmit Word 1-128	2-257
	0	U8	RO	Number of entries (value=128)	
	1	U16	RW	Transmit Word 1	2-3
	2	U16	RW	Transmit Word 2	4-5
	...	...	...	...	...
	128	U16	RW	Transmit Word 128	256-257
2011h	-	STRUCT		Transmit Word 129-255 area	258-511
	0	U8	RO	Number of entries (value=127)	
	1	U16	RW	Transmit Word 129	258-259
	2	U16	RW	Transmit Word 130	260-261
	...	...	...	...	...
	127	U16	RW	Transmit Word 255	510-511
2020h	-	STRUCT		Transmit Long 1-128 area	2-511
	0	U8	RO	Number of entries (value=128)	
	1	U32	RW	Transmit Long 1	2-5
	2	U32	RW	Transmit Long 2	6-9

Index	Subindex	Type	Access	Name	Position in Input Data area (byte)
	...	...	...	...	...
	128	U32	RW	Transmit Long 128	510-511 (last 2 bytes padded with zeros)

7.3.2 Output Buffer

Contains data received **from** the secondary CANopen network.

Index	Subindex	Type	Access	Name	Position in Output Data area (byte)
2100h	-	STRUCT		Receive Byte 1-128 area	2-129 (0 and 1 reserved for Status Word)
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Receive Byte 1	2
	2	U8	RW	Receive Byte2	3
	...	...	...	...	,,,
	128	U8	RW	Receive Byte 128	129
2101h	-	STRUCT		Receive Byte 129-256	130-257
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Receive Byte 129	130
	2	U8	RW	Receive Byte 130	131
	...	...	...	...	...
	128	U8	RW	Receive Byte 256	257
2102h	-	STRUCT		Receive Byte 257-384	258-385
	0	U8	RO	Number of entries (value=128)	
	1	U8	RW	Receive Byte 257	258
	2	U8	RW	Receive Byte 258	259
	...	...	...	...	...
	128	U8	RW	Receive Byte 384	385
2103h	-	STRUCT		Receive Byte 385-510	386-511
	0	U8	RO	Number of entries (value=126)	
	1	U8	RW	Receive Byte 386	386
	2	U8	RW	Receive Byte 387	387
	...	...	...	...	...
	126	U8	RW	Receive Byte 511	511
2110h	-	STRUCT		Receive Word 1-128	2-257
	0	U8	RO	Number of entries (value=128)	
	1	U16	RW	Receive Word 1	2-3
	2	U16	RW	Receive Word 2	4-5
	...	...	...	...	...
	128	U16	RW	Receive Word 128	256-257
2111h	-	STRUCT		Receive Word 129-255 area	258-511
	0	U8	RO	Number of entries (value=127)	
	1	U16	RW	Receive Word 129	258-259
	2	U16	RW	Receive Word 130	260-261
	...	...	...	...	...
	127	U16	RW	Receive Word 255	510-511
2120h	-	STRUCT		Receive Long 1-128 area	2-511
	0	U8	RO	Number of entries (value=128)	
	1	U32	RW	Receive Long 1	2-5
	2	U32	RW	Receive Long 2	6-9
	...	...	...	...	...
	128	U32	RW	Receive Long 128	510-511 (last 2 bytes padded with zeros)

7.3.3 Local Object Dictionary Index Mapping

The data in the Input/Output Buffers is mapped to three different areas in the Local Object Dictionary: *Byte*, *Word* and *Long*, using different data types.

Example:

Application data bytes **2 - 5** are mapped to the following object indices:

- Byte object index 2000h, subindex 1 - 4
- Word object index 2010h, subindex 1 - 2
- Long (double word) object index 2020h, subindex 1

Words and double words use Motorola (big-endian) format.

The following relation chart shows the transmit data area. The receive data area has the same structure, but with indices for byte objects starting at 2100h.

The first two bytes are reserved for the Control Word (transmit buffer) and Status Word (receive buffer) and cannot be used for data exchange.

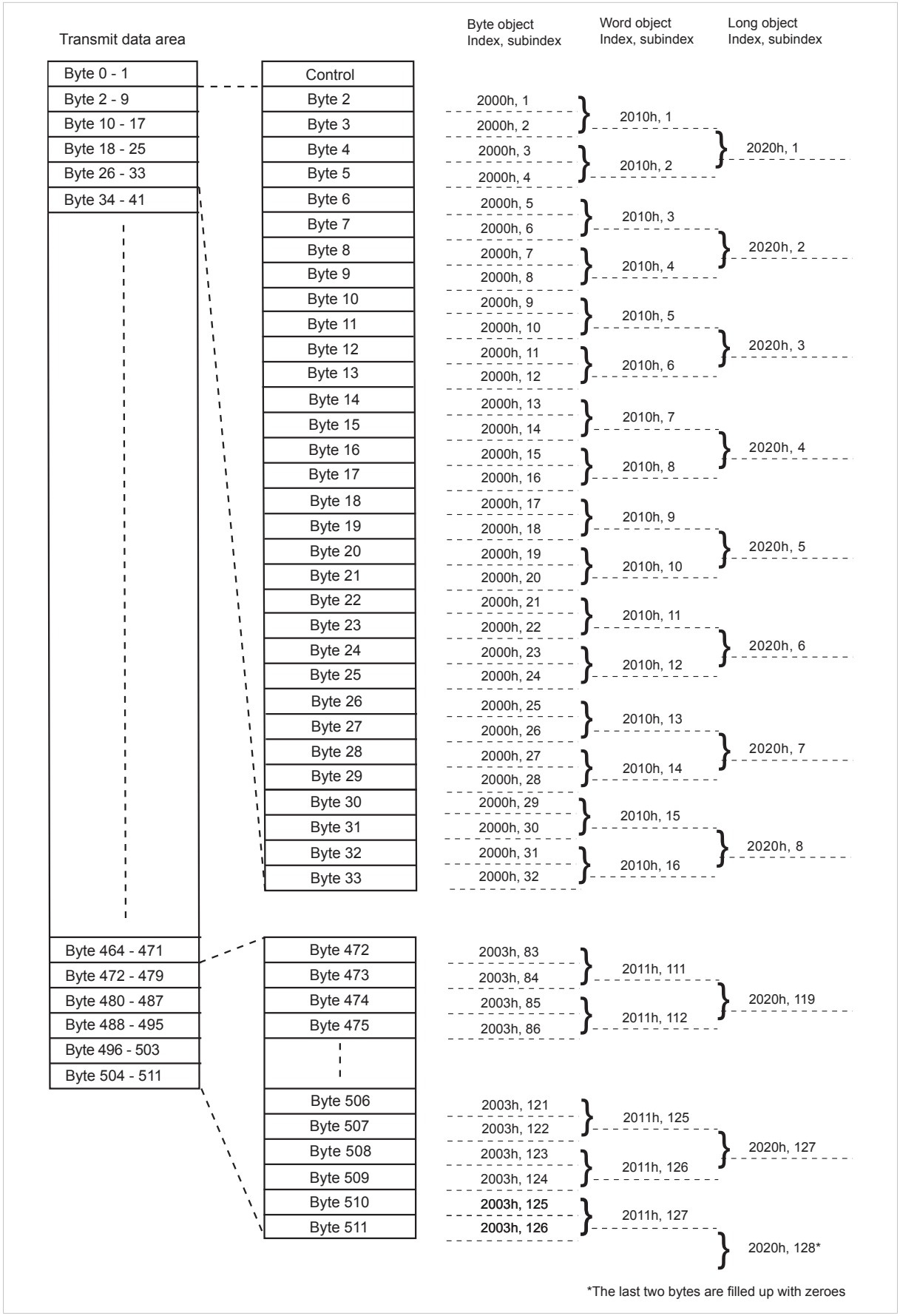


Fig. 27 I/O Buffer Addresses / Object Dictionary relation chart

7.3.4 General Fieldbus Parameters

Index range 3000h - 300Fh is allocated for general fieldbus parameters. The valid data range may differ depending on the slave interface.

Index	Subindex	Type	Access	Name and Description	Comment
3000h	128	U16	RW	Input Data Size (to primary network)	Valid values: 2-512, default 16
3001h	0	U16	RW	Output Data Size (from primary network)	Valid values: 2-512, default 16

Writing to object 1010h will verify the stored Input/Output Data sizes against the current fieldbus limitations. If the data sizes do not comply, error code 6600h will be generated.

The gateway must be restarted for changes to take effect on the primary network.

7.3.5 Primary Network Specific Parameters

Index range 3010h - 30AFh is allocated for parameters that are specific for the different primary networks available in the Anybus X-gateway CANopen model range.

Index	Subindex	Type	Access	Name and Description	Comment
PROFIBUS specific parameters:					
3011h - 301Fh	-	-	-	(reserved for future use)	
CANopen primary network specific parameters:					
3020h	0	U8	RW	CANopen Offline Option Specifies which event will cause the module to report the bus as offline.	Valid values: 1 (bus error) 2 (node guarding or heartbeat)
3021h - 302Fh	-	-	-	(reserved for future use)	
DeviceNet specific parameters:					
3031h - 303Fh	-	-	-	(reserved for future use)	
Modbus RTU specific parameters:					
3040h	0	U16	RW	Offline timeout	Valid values: 0–65535
3041h - 304Fh	-	-	-	(reserved for future use)	
ControlNet specific parameters:					
3051h - 305Fh	-	-	-	(reserved for future use)	
Modbus TCP specific parameters:					
3060h	0	U16	RW	Offline timeout	Valid values: 0–65535
Communication in Modbus-TCP is handled with a query/response protocol. The offline timeout parameter defines the time limit for how longe the slave/adapter will go offline. If the parameter is set to zero, the functionality is disabled.					
3061h - 306Fh	-	-	-	(reserved for future use)	
EtherNet/IP specific parameters:					
3071h - 307Fh	-	-	-	(reserved for future use)	
PROFINET IO specific parameters:					
3081h - 308Fh	-	-	-	(reserved for future use)	
EtherCAT specific parameters:					
3091h - 309Fh	-	-	-	(reserved for future use)	

Index	Subindex	Type	Access	Name and Description	Comment
PROFINET IRT specific parameters:					
30A1h - 30AFh	-	-	-	(reserved for future use)	

This page intentionally left blank

A Technical Data

A.1 General Specifications

Model name	**Anybus X-gateway CANopen PROFINET IRT (2.32)**
Order code	AB7329
Dimensions (L x W x H)	120 x 75 x 27 mm
Weight	150 g
Operating temperature	-25 to +55 °C (IEC 60068-2-1 and IEC 60068-2-2)
Storage temperature	-40 to +85 °C (IEC 60068-2-1 and IEC 60068-2-2)
Humidity range	5–95 % RH, non-condensing (IEC 60068-2-30)
Power supply	24 V ±10 % DC regulated power source
Current consumption	Typical: 100 mA @ 24 VDC Maximum: 250 mA @ 24 VDC
Configuration port	USB (USB to CAN cable not included)
Galvanic isolation	Yes, on both secondary CANopen and primary network sides
Mechanical rating	IP20, NEMA rating 1
Mounting	DIN rail (EN 50022) Network shield conductance via DIN rail
Certifications	CE

A.2 Secondary CANopen Network Interface

CANopen functionality	Master (manager) or slave
Communication profiles	CiA Draft Standard 301 v4.2 CiA Draft Standard Proposal 302 Part 1–5
CANopen I/O data	128 Receive PDOs and 128 Transmit PDOs Up to 510 bytes cyclic I/O data
PDO message types	COS (Change of State), Cyclic Synchronous, Acyclic Synchronous
Maximum baud rate	1 Mbit/s
Maximum number of slaves	126
Configuration switches	Node address and baud rate
CANopen connector	D-sub 9 Male (included)
CAN specification	CAN 2.0A

A.3 Primary PROFINET IRT (2.32) Network Interface

PROFINET specification	2.32
PROFINET functionality	Isochronous Real-Time (IRT) communication Conformance supporting Class A, B and C Media Redundancy Protocol (MRP) support Discovery and Configuration Protocol (DCP) support Asset Management
Isochronous cycle times	0.250 ms to 16 ms
Maximum I/O data	Up to 512 byte in each direction
Ethernet	100 Mbit/s, full duplex (fixed) Dual port cut-through switch, RJ45 connectors