# Anybus CompactCom 40

EtherNet/IP Transparent Ethernet

# 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

This page intentionally left blank

# 1    Preface

## 1.1    About this document

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

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

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

## 1.2    Related Documents

| Document | Author | Document ID |
| --- | --- | --- |
| Anybus CompactCom 40 Software Design Guide | HMS | HMSI-216-125 |
| Anybus CompactCom M40 Hardware Design Guide | HMS | HMSI-216-126 |
| Anybus CompactCom B40 Design Guide | HMS | HMSI-27-230 |
| Anybus CompactCom Host Application Implementation Guide | HMS | HMSI-27-334 |
| Anybus CompactCom 40 EtherNet/IP Network Guide | HMS | SCM-1202–031 |
| CIP specification, Volumes 1 (CIP Common) and 2 (EtherNet/IP) | ODVA | |

## 1.3    Document History

| Version | Date | Description |
| --- | --- | --- |
| 1.0 | 2016-10-22 | First release |
| 1.1 | 2017-01-18 | Minor corrections and updates |
| 1.2 | 2017-05-23 | Ethernet Host Object updated (disabling of DHCP)<br>Port Object updated |
| 1.3 | 2017-07-11 | Added appendix for backward compatibility |
| 1.4 | 2017-11-28 | Added Assembly Mapping Object guide |
| 1.5 | 2017-12-15 | Updated Copyright Appendix |
| 1.6 | 2018-05-07 | Added Conformance Test Guide<br>Updates to CIP objects ADI and QoS<br>Misc updates |
| 1.7 | 2018-09-19 | Updated Set_Configuration_Data descriptions in EtherNet/IP Host object |
| 1.8 | 2018-10-23 | Minor update |
| 1.9 | 2019-06-10 | Rebranding<br>CIP Sync added<br>Misc updates |
| 2.0 | 2016-10-11 | Translated into Japanese |
| 2.1 | 2020-09-23 | Translated into Japanese |
| 2.2 | 2021–05–26 | Updated CIP Identity Object |
| 2.3 | 2021-09-21 | Added Time Object<br>Minor updates |

## 1.4 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. 6*

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

| ⚠ | **WARNING**<br>Instruction that must be followed to avoid a risk of death or serious injury. |
|---|---|

| ⚠ | **Caution**<br>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.* |
|---|---|

## 1.5 Document Specific Conventions

• The terms "Anybus" or "module" refers to the Anybus CompactCom module.

• The terms "host" or "host application" refer to the device that hosts the Anybus.

• Hexadecimal values are written in the format NNNNh or 0xNNNN, where NNNN is the hexadecimal value.

• A byte always consists of 8 bits.

• The terms "basic" and "extended" are used to classify objects, instances and attributes.

## 1.6 Abbreviations

| Abbreviation | Meaning |
|---|---|
| API | assigned packet interval |
| RPI | requested packet interval |
| T | target (in this case the module) |
| O | origin (in this case the master) |

## 1.7        Trademarks

Anybus® is a registered trademark of HMS Networks.

EtherNet/IP is a trademark of ODVA, Inc.

All other trademarks are the property of their respective holders.

# 2 About the Anybus CompactCom 40 EtherNet/IP

## 2.1 General

The Anybus CompactCom 40 EtherNet/IP communication module provides instant EtherNet/IP conformance tested connectivity via the patented Anybus CompactCom host interface. Any device that supports this standard can take advantage of the features provided by the module, allowing seamless network integration regardless of network type. The module supports both linear and ring network topology (DLR, Device Level Ring).

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

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

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

## 2.2        Features

- Transparent Ethernet

- Two EtherNet/IP ports

- Ethernet connectors

- Max. read process data: 1448 bytes

- Max. write process data: 1448 bytes

- Max. process data (read + write, in bytes): 2896 bytes

- Beacon Based DLR (Device Level Ring) and linear network topology supported

- Black channel interface, offering a transparent channel supporting CIP Safety.

- 10/100 Mbit, full/half duplex operation

- CIP Sync functionality

- Customizable Identity Information

- Up to 65535 ADIs

- CIP Parameter Object support

- Expandable CIP-object implementation

- Supports unconnected CIP routing

- Modular Device functionality

- QuickConnect supported

- Multiple IO assembly instances can be created

## 2.3        Transparent Ethernet

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

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

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

See also ...

-

- Anybus CompactCom 40 Hardware Design Guide

-

# 3 Basic Operation

## 3.1 General Information

### 3.1.1 Software Requirements

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

- Certain functionality in the module requires that the command Get_Instance_Number_By_ Order (Application Data Object, FEh) is implemented in the host application.

- Up to 5 diagnostic instances (See *Diagnostic Object (02h), p. 66*) can be created by the host application during normal conditions. An additional 6th instance may be created in event of a major fault. This limit is set by the module, not by the network.

- EtherNet/IP in itself does not impose any specific timing demands when it comes to acyclic requests (i.e. requests towards instances in the Application Data Object), however it is generally recommended to process and respond to such requests within a reasonable time period. The application that sends the request, also decides the timeout, e.g. EIPScan employs a timeout of 10 seconds.

- The use of advanced CIP-specific functionality may require in-depth knowledge in CIP networking internals and/or information from the official CIP and EtherNet/IP specifications. In such cases, the people responsible for the implementation of this product is expected either to obtain these specifications to gain sufficient knowledge or limit their implementation is such a way that this is not necessary.

See also...

- *Diagnostic Object (02h), p. 66* (Anybus Module Objects)

- Anybus CompactCom 40 Software Design Guide, "Application Data Object (FEh)"

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

### 3.1.2     Electronic Data Sheet (EDS)

On EtherNet/IP, the characteristics of a device is stored in an ASCII data file with the suffix EDS. This file is used by configuration tools etc. when setting up the network configuration. HMS Networks supplies a standard (generic) EDS file, which corresponds to the default settings in the module. However, due to the flexible nature of the Anybus CompactCom concept, it is possible to alter the behavior of the product in ways which invalidate the generic EDS file. In such case, a custom EDS file needs to be created, which in turn invalidates the default identity information and require re-certification of the product.

Since the module implements the Parameter Object, it is possible for configuration tools such as RSNetWorx to automatically generate a suitable EDS-file. Note that this functionality requires that the command Get_Instance_Number_By_Order (Application Data Object, FEh) has been implemented in the host application.

See also..

- *Parameter Object (0Fh), p. 38* (CIP object)

- Anybus CompactCom 40 Software Design Guide, "Application Data Object (FEh)"

> **!** HMS Industrial Networks approves use of the standard EDS-file only under the condition that it matches the actual implementation and that the identity information remains unchanged.

## 3.2     Network Identity

By default, the module uses the following identity settings:

| | |
|---|---|
| **Vendor ID:** | 005Ah (HMS Industrial Networks) |
| **Device Type:** | 002Bh (Generic Device) |
| **Product Code:** | 0037h (Anybus CompactCom 40 EtherNet/IP) |
| **Product Name:** | "Anybus CompactCom 40 EtherNet/IP(TM)" |

Optionally, it is possible to customize the identity of the module by implementing the corresponding instance attributes in the EtherNet/IP Host Object.

See also...

- *Identity Object (01h), p. 27* (CIP object)

- *EtherNet/IP Host Object (F8h), p. 102* (Host Application Object)

> **!** According to the CIP specification, the combination of Vendor ID and serial number <u>must</u> be unique. It is <u>not</u> permitted to use a custom serial number in combination with the HMS Vendor ID (005Ah), nor is it permitted to choose Vendor ID arbitrarily. Failure to comply to this requirement will induce interoperability problems and/or other unwanted side effects.
>
> To obtain a Vendor ID, contact the ODVA.

## 3.3     Communication Settings

Network related communication settings are grouped in the Network Configuration Object (04h), and includes:

| **IP settings** | These settings must be set properly in order for the module to be able to participate on the network. |
| | IP settings must be synchronized between the Anybus CompactCom 40 and the host application. |
| | The module supports DHCP, which may be used to retrieve the IP settings from a DHCP-server automatically. DHCP is enabled by default, but can be disabled if necessary. |
| **Physical Link Settings** | By default, the module uses auto negotiation to establish the physical link settings, however it is possible to force a specific setting if necessary. |

The parameters in the Network Configuration Object (04h) are available from the network through the TCP/IP Interface Object (CIP).

See also...

- *TCP/IP Interface Object (F5h), p. 55* (CIP object)

- *Ethernet Link Object (F6h), p. 58* (CIP object)

- *Network Configuration Object (04h), p. 68* (Anybus Module Object)

- *Secure HICP (Secure Host IP Configuration Protocol), p. 119*

## 3.4        Beacon Based DLR (Device Level Ring)

Device Level Ring (DLR) is a network technology for industrial applications that uses embedded switch functionality in automation end devices, such as programmable automation controllers and I/O modules, to enable Ethernet ring network topologies at the device level. DLR technology adds network resilience to optimize machine operation. Beacon based DLR networks consist of a ring supervisor and a number of ring nodes, and use "beacons" to detect breaks in the ring. When a DLR network detects a break in the ring, it provides ways to alternatively route the data to recover the network. Diagnostics built into DLR products can identify the point of failure, thus helping to speed maintenance and reduce repair time. The Anybus CompactCom 40 EtherNet/IP implements the DLR protocol, and it is enabled by default. The device is able to process and act on beacon frames sent by ring supervisors, and supports beacon rates down to 100 μs. If needed, the DLR functionality can be disabled. This can be done by setting attribute #31 (Enable DLR) in the EtherNet/IP Host Object to False. See *EtherNet/IP Host Object (F8h), p. 102*.

## 3.5        Network Data Exchange

### 3.5.1     Application Data

Application Data Instances (ADIs) are represented through the ADI Object (CIP). Each instance within this objects corresponds directly to an instance in the Application Data Object on the host application side.

Accessible range of ADIs is 1 to 65535.

See also...

- *Parameter Object (0Fh), p. 38* (CIP object)

- *ADI Object (A2h), p. 51* (CIP object)

### 3.5.2     Process Data

Process Data is represented as dedicated instances in the Assembly Object (CIP).

See also...

- *Assembly Object (04h), p. 31* (CIP object)

- *Connection Manager (06h), p. 34* (CIP object)

### 3.5.3     Translation of Data Types

The Anybus data types are translated to CIP-standard and vice versa as follows:

| Anybus Data Type | CIP Data Type | Comments |
|---|---|---|
| BOOL | BOOL | Each ADI element of this type occupies one byte. |
| ENUM | USINT | |
| SINT8 | SINT | |
| UINT8 | USINT | |
| SINT16 | INT | Each ADI element of this type occupies two bytes. |
| UINT16 | UINT | |
| SINT32 | DINT | Each ADI element of this type occupies four bytes. |
| UINT32 | UDINT | |
| FLOAT | REAL | |
| CHAR | SHORT_STRING | SHORT_STRING consists of a single-byte length field (which in this case represents the number of ADI elements) followed by the actual character data (in this case the actual ADI elements). This means that a 10-character string occupies 11 bytes. |
| SINT64 | LINT | Each ADI element of this type occupies eight bytes. |

| Anybus Data Type | CIP Data Type | Comments |
|---|---|---|
| UINT64 | ULINT | |
| BITS8 | BYTE | Each ADI element of this type occupies one byte. |
| BITS16 | WORD | Each ADI element of this type occupies two bytes. |
| BITS32 | DWORD | Each ADI element of this type occupies four bytes. |
| OCTET | USINT | |
| BITS1-7 | BYTE | Bit fields of size 1 - 7 |
| PAD0-8 | BYTE | Bit fields of size 0 - 8 used for padding |
| PAD9-16 | BYTE | Bit fields of size 9 - 16 used for padding |
| BOOL1 | BOOL | |

## 3.6    Modular Device Functionality

Modular devices consist of a backplane with a certain number of slots. The first slot is occupied by the "coupler" which contains the Anybus CompactCom module. All other slots may be empty or occupied by modules.

When mapping ADIs to process data the application shall map the process data of each module in slot order.

A list of modules in a Modular Device is available to the EtherNet/IP network master by a request to the CIP Identity object.

See also ...

- "Modular Device Object (ECh)" (see Anybus CompactCom 40 Software Design Guide)

- *Identity Object (01h), p. 27* (CIP object)

# 4        EtherNet/IP Implementation Details

## 4.1        General Information

This chapter covers EtherNet/IP specific details in the Anybus implementation. Note that the use of such functionality may require in-depth knowledge in EtherNet/IP networking internals and/or information from the official EtherNet/IP and CIP specifications. In such cases, the people responsible for the implementation of this product are expected either to obtain these specifications to gain sufficient knowledge or limit their implementation in such a way that this is not necessary. The EDS file must be changed to reflect all changes.

## 4.2        EtherNet/IP & CIP Implementation

By default, the module supports the generic CIP profile. Optionally, it is possible to re-route requests to unimplemented CIP objects to the host application, thus enabling support for other profiles etc.

To support a specific profile, perform the following steps:

1.  Set up the identity settings in the EtherNet/IP Host Object according to profile requirements.

2.  Implement the Assembly Mapping Object in the host application.

3.  Set up the Assembly Instance Numbers according to profile requirements.

4.  Enable routing of CIP messages to the host application in the EtherNet/IP Host Object.

5.  Implement the required CIP objects in the host application.

See also...

- *Using the Assembly Mapping Object (EBh), p. 16*

- *EtherNet/IP Host Object (F8h), p. 102* (Host Application Object), details for the command Process_CIP_Object_Request.

## 4.3     Using the Assembly Mapping Object (EBh)

### 4.3.1     Introduction

This guide will describe how to map CIP instances to ADI data, using the assembly mapping object (EBh).

### 4.3.2     Adding Data - The Application Data Object

According to the Anybus object model, all data that is used in the application must be represented by application data instances (ADIs). ADIs are small portions of structured data, each representing only one of three possible different types: variable, array or structure.

See the Application Data Object (FEh) in the Software Design Guide for more information.

Below is an example with 30 ADIs. Instances 1 - 6 and 30 are implemented in the application, and 7 - 29 are not implemented.

**Application Data Object (FEh) Instances**

| Instance # | Implemented | Order # |
|------------|-------------|---------|
| 1          | Yes         | 1       |
| 2          | Yes         | 2       |
| 3          | Yes         | 3       |
| 4          | Yes         | 4       |
| 5          | Yes         | 5       |
| 6          | Yes         | 6       |
| 7...29     | No          | -       |
| 30         | Yes         | 7       |

### 4.3.3     Grouping Data - The Assembly Mapping Object

The assembly mapping object makes it possible to create an arbitrary number of process data sets, called assembly mappings. Each assembly mapping instance represents a different logical set of process data, that can be chosen by the network and received over a single connection.

Every instance of the assembly mapping object, as seen below, contains an ADI map, referring to an arbitrary number of ADIs.

The instance numbers can be set freely.

| Assembly Mapping Object (EBh) Instances | | |
| --- | --- | --- |
| Instance # | Type | ADI Map |
| 1 | Read | 1, 2 |
| 2 | Read | 2, 3 |
| 10 | Write | 3, 4, 30 |
| 11 | Write | 4, 5 |
| 30 | Read | 5, 6 |
| 51 | Write | 6, 30 |

There are two object instance attributes in the assembly mapping object, called Write PD Instance List and Read PD Instance List. These two attributes contain references to all read instances and all write instances, respectively. The example above will automatically generate the following content in these two attributes.

| Name | Attribute | Values |
| --- | --- | --- |
| Write PD Instance List | 11 | 10, 11, 51 |
| Read PD Instance List | 12 | 1, 2, 30 |

> ⓘ    *The attributes Write PD Instance List and Read PD Instance List adopts the view of the network, e.g. an input will produce data on the network and an output will consume data on the network.*
>
> *Write PD Instance List will contain all assembly mapping object instances with type "Read". Read PD Instance List will contain all assembly mapping object instances with type "Write".*

### 4.3.4        Configuring CIP Assembly Numbers

The read and write instance list attributes in the assembly mapping object are bound to two corresponding attributes in the EtherNet/IP host object, according to the following table.

This routes application data to CIP assembly data, by linking CIP instance numbers to assembly mapping object instances.

> (i)   *The lists are matched index-wise, and must thus be of equal length.*

| Assembly Mapping Object Attribute | Value | | Value | EtherNet/IP Host Object Instance Attribute |
|---|---|---|---|---|
| 11 - Write PD Instance List | 10 | <—> | 10 | 7 - Producing Instance Number |
| | 11 | <—> | 22 | |
| | 51 | <—> | 100 | |
| 12 - Read PD Instance List | 1 | <—> | 1 | 8 - Consuming Instance Number |
| | 2 | <—> | 2 | |
| | 30 | <—> | 150 | |

> **!**   For conformity with the CIP specification, both the Write_Assembly_Data and the Read_Assembly_Data services must be implemented.

### 4.3.5        Going Forward

During the initialization phase, in the NW_INIT state, all write assemblies (e.g. the instances of the assembly mapping object with type"write") will be remapped to the write process data area. For this to happen, the device will issue the Remap_ADI_Write_Area command to the application data object in the host.

See the appendix about "Runtime Remapping of Process Data" in the Anybus CompactCom 40 Software Design Guide for more information.

When the network has been initialized, the device transitions from NW_INIT to the WAIT_PROCESS state. When the device receives a forward open request, the producing/consuming parameters in the request are verified and matched against the EtherNet/IP Host Object instance numbers (producing/consuming)

If the verification is successful, the read process data is remapped and the device transitions to the PROCESS_ACTIVE state. The I/O connection will then be established, and data can be exchanged over the network.

## 4.4 Diagnostics

The severity value of all pending events are combined (using logical OR) and copied to the corresponding bits in the "Status" attribute of the Identity Object (CIP).

See also...

- *Identity Object (01h), p. 27* (CIP Object)
- *Diagnostic Object (02h), p. 66* (Anybus Module Object)

## 4.5 QuickConnect

The module supports the QuickConnect functionality. It is enabled in the EtherNet/IP Host Object. The module fulfills Class A with a startup time of less than 180 ms, with 16 bytes of I/O data mapped with parallel, SPI or shift register application interface.

See also ...

- *EtherNet/IP Host Object (F8h), p. 102* (Host Application Object)
- *TCP/IP Interface Object (F5h), p. 55* (CIP object)

## 4.6 CIP Safety

The Anybus CompactCom 40 EtherNet/IP supports the CIP safety profile. This profile makes it possible for a user to send data on a black channel interface, i.e. a safe channel over EtherNet/IP using an add-on safety module, e.g. the IXXAT Safe T100. For an application to support CIP safety, the Functional Safety Object (E8h, host application object) has to be implemented. The Anybus CompactCom serial channel is used for the functional safety communication. When this channel is used for the host application, a second separate serial channel is implemented for the functional safety communication. See the Anybus CompactCom Hardware Design Guide for more information.

See ...

- *Functional Safety Module Object (11h), p. 81*
- *Functional Safety Object (E8h), p. 91*

### 4.6.1 Safety Module Firmware Upgrade

The firmware of the connected safety module can be upgraded through the Anybus CompactCom. The safety firmware (hiff file) has to be downloaded to the firmware directory in the Anybus CompactCom. At restart, the Anybus CompactCom detects and validates the firmware. Firmware upgrade in progress is indicated to the application by attribute #5 (instance #1) in the Functional Safety Object (E8h), which is set to TRUE during the firmware upgrade. The Anybus CompactCom will need more time to initialize , please do not restart the module during this time.

### 4.6.2 Reset Request from Network

When a reset request arrives from the network, a delay of 1 s is introduced before the Anybus CompactCom 40 EtherNet/IP is reset, if CIP safety is enabled.

## 4.7 CIP Sync

The time synchronization technology for the Common Industrial Protocol (CIP) is called CIP Sync. This technology allows accurate real-time synchronization of devices and controllers connected over CIP networks that require time stamping, sequence of events recording, distributed motion control, and support for highly distributed applications requiring increased control coordination. CIP Sync is based on the IEEE 1588 (IEC 61588) standard – Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, commonly referred to as the Precision Time Protocol (PTP). The protocol is designed for, but not limited to, local area networks like Ethernet. The protocol provides a standard mechanism to synchronize clocks across a network of distributed devices. A copy of the standard may be obtained by ordering the standard IEEE 1588-2008 from the IEEE Standards Organization Committee. The Time Sync Object (43h, CIP object) provides a CIP interface to the IEEE 1588 (IEC 61588) Standard. CIP Sync is enabled by setting attribute #32 in the EtherNet/IP Host Object (F8h)

See ...

- *EtherNet/IP Host Object (F8h), p. 102*

- *Time Sync Object (43h), p. 41* (CIP object)

# 5 Transparent Ethernet

## 5.1 General Information

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

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

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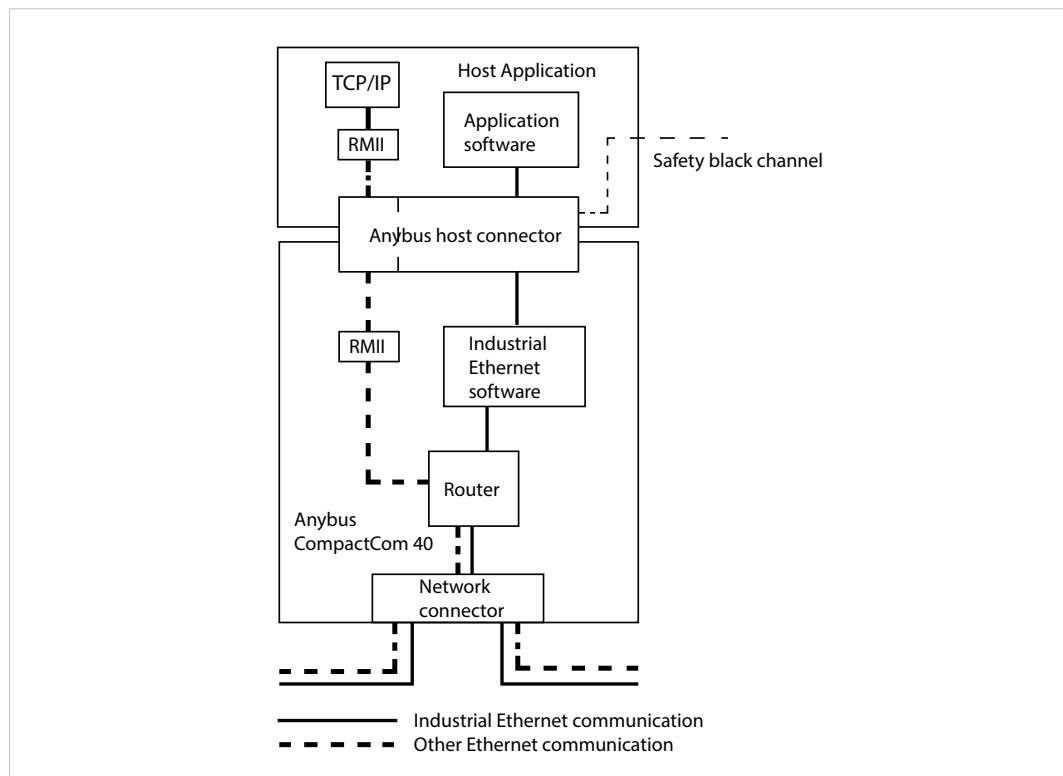


**Fig. 1**

> **!** Transparent Ethernet has to be enabled by setting instance attribute #16 in the Anybus Object (01h) during setup, see below.
>
> 16–bit parallell mode cannot be used when using transparent Ethernet.
>
> MAC addresses and IP configurations have to be synchronized, see below.
>
> Some EtherTypes ,TCP/UDP ports and multicast MAC addresses may be reserved for use by the industrial Ethernet network. These must not be used for transparent Ethernet communication. See below for more information.
>
> The Transparent Ethernet interface only supports 100 Mbit, full duplex operation.

## 5.2 Enabling Transparent Ethernet

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

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

## 5.3      MAC Address Synchronization

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

- The pre-programmed MAC address in attribute #1 (instance #1) in the Network Ethernet Object (0Ch) is read and used by the host application when communicating on Ethernet.

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

## 5.4      IP Configuration Synchronization

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

## 5.5      Routing Restrictions

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

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

### 5.5.1        EtherTypes

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

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

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

### 5.5.2        Multicast MAC Addresses

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

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

(X: any number 0-F)

### 5.5.3        UDP/TCP Ports

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

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

The following ports are reserved for use by PROFINET:

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

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

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

The following port is reserved for use by Modbus TCP:

- TCP 502 (Modbus messaging)

# 6 Firmware Upgrade

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

- HICP needs to be enabled (FMII only).

- An FTP server needs to be implemented in the host application.

- A directory named "firmware" in the host application FTP root.

- The file module.nfo in the "firmware" directory in theAnybus CompactCom file system has to be copied to the "firmware" directory in the host application file system (FMII only). The Anybus CompactCom file system is accessed via the Anybus File System Object (0Ah).

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

- detect a new file in the "firmware" directory

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

The firmware will be updated upon the next reset of the Anybus CompactCom 40 EtherNet/IP.

# 7 CIP Objects

## 7.1 General Information

This chapter specifies the CIP-object implementation in the module. These objects can be accessed from the network, but not directly by the host application.

Mandatory objects

- *Identity Object (01h), p. 27*

- *Message Router (02h), p. 30*

- *Assembly Object (04h), p. 31*

- *Connection Manager (06h), p. 34*

- *QoS Object (48h), p. 46*

- *TCP/IP Interface Object (F5h), p. 55*

- *Ethernet Link Object (F6h), p. 58*

CIP Energy Objects:

- *Base Energy Object (4Eh), p. 47*

- *Power Management Object (53h), p. 49*

Optional Objects:

- *Parameter Object (0Fh), p. 38*

- *Time Sync Object (43h), p. 41*

- *DLR Object (47h), p. 45*

- *Port Object (F4h), p. 53* (Optional)

Vendor Specific Objects:

- *ADI Object (A2h), p. 51*

It is possible to implement additional CIP-objects in the host application using the CIP forwarding functionality, see *EtherNet/IP Host Object (F8h), p. 102* and commend details for Process_CIP_ Object_Request.

Unconnected CIP routing is supported, which means that a message can be sent to a device without first setting up a connection.

## 7.2        Translation of Status Codes

If an error occurs when an object is requested from the application, an error code is returned. These Anybus CompactCom error codes are translated to CIP status codes according to the table below.

| Anybus CompactCom 40 Error Code | | CIP Status Code | |
|---|---|---|---|
| Value | Error | Value | Status |
| 00h | Reserved | 1Eh | Embedded service error |
| 01h | Reserved | 1Eh | Embedded service error |
| 02h | Invalid message format | 1Eh | Embedded service error |
| 03h | Unsupported object | 05h | Path destination unknown |
| 04h | Unsupported instance | 05h | Path destination unknown |
| 05h | Unsupported Command | 08h | Service not supported |
| 06h | Invalid CmdExt(0) | 14h | Depending on Anybus CompactCom Service returning this reply, e.g. attribute not supported |
| 07h | Invalid CmdExt(1) | - | Depending on Anybus CompactCom Service returning this reply |
| 08h | Attribute not settable | 0Eh | Attribute not settable |
| 09h | Attribute not gettable | 2Ch | Attribute not gettable |
| 0Ah | Too Much Data | 15h | Too much data |
| 0Bh | Not Enough Data | 13h | Not enough data |
| 0Ch | Out of range | 09h | Invalid attribute value |
| 0Dh | Invalid state | 0Ch | Object state conflict |
| 0Eh | Out of resources | 02h | Resource unavailable |
| 0Fh | Segmentation failure | 1Eh | Embedded service error |
| 10h | Segmentation buffer overflow | 23h | Buffer overflow |
| 11h | Value too high | 09h | Invalid attribute value |
| 12h | Value too low | 09h | Invalid attribute value |
| 13h | Attribute controlled | 0Fh | A permission/privilege check failed |
| 14h | Message channel too small | 11h | Reply data too large |
| FFh | Object Specific Error | 1Fh | Vendor specific error. No additional error codes will be sent on EtherNet/IP |
| Other | - | 1Eh | Embedded service error |

For further information about the Anybus CompactCom error codes, please consult the Anybus CompactCom 40 Software Design Guide.

## 7.3        Identity Object (01h)

### Category

Extended

### Object Description

The Identity Object provides identification of and general information about the module.

The object supports multiple instances. Instance 1, which is the only mandatory instance, describes the whole product. It is used by applications to determine what nodes are on the network and to match an EDS file with a product on the network. The other (optional) instances describe different parts of the product, e.g. the software.

If modular device functionality is enabled, a list of the modules in the slots can be retrieved and made available to the network master by sending a get request to class attribute 100.

Instance attributes 1 - 7 can be customized by implementing the EtherNet/IP Host Object.

Additional identity instances can be registered by implementing the CIP Identity Host Object (host application object).

See also ....

- *EtherNet/IP Host Object (F8h), p. 102*
- *CIP Identity Host Object (EDh), p. 93*

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| | Get_Attribute_All |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |
| | Get_Attribute_All |
| | Get_Attribute_List |
| | Set_Attribute_List |
| | Reset |

### Class Attributes

| # | Name | Access | Type | Value |
|---|---|---|---|---|
| 1 | Revision | Get | UINT | 0002h (Object revision) |
| 2 | Max instance | Get | UINT | Maximum instance number |
| 3 | Number of instances | Get | UINT | Number of instances |
| 100 | Module ID List | Get | Array of UINT32 | If modular device functionality is enabled, a request to this attribute will generate a Get_List request to the Modular Device Object in the host application. |

## Instance Attributes

Attributes #1–4 and #6–7 can be customized by implementing the EtherNet/IP Host Object, see *EtherNet/IP Host Object (F8h), p. 102*

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Vendor ID | Get | UINT | 005Ah (HMS Industrial Networks) |
| 2 | Device Type | Get | UINT | 002Bh (Generic Device) |
| 3 | Product Code | Get | UINT | 0037h (Anybus CompactCom 40 EtherNet/IP) |
| 4 | Revision | Get | Struct of:<br>　USINT<br>　USINT | Major and minor firmware revision |
| 5 | Status | Get | WORD | See Device Status table below |
| 6 | Serial Number | Get | UDINT | Unique serial number (assigned by HMS) |
| 7 | Product Name | Get | SHORT_STRING | "Anybus CompactCom 40 EtherNet/IP (TM)" |
| 11 | Active language | Set | Struct of:<br>　USINT<br>　USINT<br>　USINT | Requests sent to this instance are forwarded to the Application Object. If the request is accepted, the module will update the language accordingly. |
| 30 | Supported Language List 2 | Get | Struct of:<br>　UINT<br>Array of:<br>　USINT<br>　USINT<br>　USINT | List of languages supported by the host application. The list is read from the Application Object and translated to CIP standard. By default the only supported language is English. The application has to implement the corresponding attributes in the application object to enable more languages. |

## Device Status

| bit(s) | Name |
|--------|------|
| 0 | Module Owned |
| 1 | (reserved) |
| 2 | Configured<br>This bit shows if the product has other settings than "out-of-box". The value is set to true if the configured attribute in the Application Object is set and/or the module's NV storage is changed from default. |
| 3 | (reserved) |
| 4... 7 | Extended Device Status:<br><br>Value:    Meaning:<br>0000b      Unknown<br>0010b      Faulted I/O Connection<br>0011b      No I/O connection established<br>0100b      Non volatile configuration bad<br>0101b      Major fault<br>0110b      Connection in Run mode<br>0111b      Connection in Idle mode<br>(other)    (reserved) |
| 8 | Set for minor recoverable faults. See *Diagnostic Object (02h), p. 66* |
| 9 | Set for minor unrecoverable faults. See *Diagnostic Object (02h), p. 66* |
| 10 | Set for major recoverable faults. See *Diagnostic Object (02h), p. 66* |
| 11 | Set for major unrecoverable faults. See *Diagnostic Object (02h), p. 66* |
| 12... 15 | (reserved) |

## Service Details: Reset

> ⓘ  *This service is not supported if safety is enabled in the Functional Safety Object (E8h).*

The module forwards reset requests from the network to the host application. For more information about network reset handling, consult the general Anybus CompactCom 40 Software Design Guide.

There are two types of network reset requests on EtherNet/IP. Please note that both types must be supported by the application.

| | |
|---|---|
| **Type 0: Power Cycling Reset** | This service emulates a power cycling of the module, and corresponds to Anybus reset type 0 (Power cycling). For further information, consult the general Anybus CompactCom 40 Software Design Guide. |
| **Type 1: Out of box reset** | This service sets a "out of box" configuration and performs a reset, and corresponds to Anybus reset type 2 (Power cycling + factory default). For further information, consult the general Anybus CompactCom 40 Software Design Guide. |

## 7.4　　　Message Router (02h)

### Category

Extended

### Object Description

The Message Router Object provides a messaging connection point through which a client may address a service to any object class or instance residing in the physical module.

In the Anybus CompactCom module it is used internally to direct object requests.

### Supported Services

**Class:** -

**Instance:** -

### Class Attributes

-

### Instance Attributes

-

## 7.5        Assembly Object (04h)

### Category

Extended

### Object Description

The Assembly object uses static assemblies and holds the Process Data sent/received by the host application. It allows data to and from each object to be sent or received over a single connection. The default assembly instance IDs used are in the vendor specific range.

It is possible for the application to create and support up to six consuming and six producing instances if the Assembly Mapping Object is implemented.

The terms "input" and "output" are defined from the network's point of view. An input will produce data on the network and an output will consume data from the network.

See also ....

- <span style="color:blue">*EtherNet/IP Host Object (F8h), p. 102*</span>

- Assembly Mapping Object (see Anybus CompactCom 40 Software Design Guide)

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0002h (Object revision) |
| 2 | Max instance | Get | UINT | Maximum instance number |

### Instance 03h Attributes (Heartbeat, Input-Only)

This instance is used as heartbeat for Input-Only connections. The data size of the Heartbeat instance in the Forward_Open-request should be 0 bytes, however other values are also permitted.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Data | Set | N/A | - (The data size of this attribute is zero) |
| 4 | Size | Get | UINT | 0 (Number of bytes in attribute 3) |

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

### Instance 04h Attributes (Heartbeat, Listen-Only)

This instance is used as heartbeat for listen-only connections. The data size of the Heartbeat instance in the Forward_Open-request should be 0 bytes, however other values are also permitted.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Data | Set | N/A | - (The data size of this attribute is zero) |
| 4 | Size | Get | UINT | 0 (Number of bytes in attribute 3) |

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

### Instance 05h Attributes (Configuration Data)

Configuration Data that is sent through the service Forward_Open will be written to this instance.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Data | Set | N/A | - (Configuration data written to the application when the forward open command has the configuration data included)- (The data size of this attribute is zero) |
| 4 | Size | Get | UINT | 0 (Number of bytes in attribute 3) |

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

See command details for Set_Configuration_Data nad Get_Contfiguration_Data in the *EtherNet/IP Host Object (F8h), p. 102*.

### Instance 06h Attributes (Heartbeat, Input-Only Extended)

This instance is used as heartbeat for input-only extended connections, and does not carry any attributes. The state of connections made to this instance does not affect the state of the Anybus CompactCom module, i.e. if the connection times out, the module does not switch to the Error state. The data size of the Heartbeat instance in the Forward_Open-request should be 0 bytes, however other values are also permitted.

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Data | Set | N/A | - (The data size of this attribute is zero) |
| 4 | Size | Get | UINT | 0 (Number of bytes in attribute 3) |

### Instance 07h Attributes (Heartbeat, Listen-Only Extended)

This instance is used as heartbeat for listen-only extended connections, and does not carry any attributes. The state of connections made to this instance does not affect the state of the Anybus CompactCom 40 module, i.e. if the connection times out, the module does not switch to the Error state. The data size of the Heartbeat instance in the Forward_Open-request should be 0 bytes, however other values are also permitted.

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Data | Set | N/A | - (The data size of this attribute is zero) |
| 4 | Size | Get | UINT | 0 (Number of bytes in attribute 3) |

## Instance 64h Attributes (Producing Instance)

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Produced Data | Get | Array of BYTE | This data corresponds to the Write Process Data. |
| 4 | Size | Get | UINT | Number of bytes in attribute 3 |

See also...

*Network Data Exchange, p. 13*

*EtherNet/IP Host Object (F8h), p. 102*(Instance attribute #7)

## Instance 96h Attributes (Consuming Instance)

The instance number for this instance can be changed by implementing the corresponding attribute in the EtherNet/IP Host Object.

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 3 | Produced Data | Set | Array of BYTE | This data corresponds to the Read Process Data. |
| 4 | Size | Get | UINT | Number of bytes in attribute 3 |

See also...

*Network Data Exchange, p. 13*

*EtherNet/IP Host Object (F8h), p. 102* (Instance attribute #8)

## 7.6        Connection Manager (06h)

### Category

Extended

### Object Description

This object is used for connection and connectionless communications, including establishing connections across multiple subnets.

### Supported Services

| | |
|---|---|
| **Class:** | - |
| **Instance:** | Get Attribute All |
| | Get Attribute Single |
| | Set Attribute Single |
| | Large_Forward_Open |
| | Forward_Open |
| | Forward_Close |
| | Unconnected Send (when unconnected routing is enabled) |

### Class Attributes

(No supported class attributes)

### Instance Attributes

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Open Requests | Set | UINT | Number of Forward Open service requests received. |
| 2 | Open Format Rejects | Set | UINT | Number of Forward Open service requests which were rejected due to bad format. |
| 3 | Open Resource Rejects | Set | UINT | Number of Forward Open service requests which were rejected due to lack of resources. |
| 4 | Open Other Rejects | Set | UINT | Number of Forward Open service requests which were rejected for reasons other than bad format or lack of resources. |
| 5 | Close Requests | Set | UINT | Number of Forward Close service requests received. |
| 6 | Close Format Rejects | Set | UINT | Number of Forward Close service requests which were rejected due to bad format. |
| 7 | Close Other Rejects | Set | UINT | Number of Forward Close service requests which were rejected for reasons other than bad format. |
| 8 | Connection Timeouts | Set | UINT | Total number of connection timeouts that have occurred in connections controlled by this Connection Manager. |

## Class 0 Connection Details

### General

Class 0 connections are only supported for safety connections. The Anybus CompactCom device will act as a transparent bridge for safety connections, meaning that open and close requests for safety connections and safety I/O data will be forwarded to the safety module. Class 0 connections use UDP transport.

| | |
|---|---|
| **Total number of supported class 0 connections:** | 2 |
| **Max input connection size:** | 241 bytes |
| | (Including the Mode Byte, Actual, Complement and Time stamp sections.) |
| **Max output connection size:** | 239 bytes |
| | (Including the Mode Byte, Actual, Complement and Time stamp sections.) |
| **Supported RPI (Requested Packet Interval):** | 1... 20000 ms |

## Class 1 Connection Details

### General

Class 1 connections are used to transfer I/O data, and can be established to instances in the Assembly Object. Each Class 1 connection will establish two data transports; one consuming and one producing. The heartbeat instances can be used for connections that shall only access inputs. Class 1 connections use UDP transport. Null forward open is supported.

| | |
|---|---|
| **Total number of supported class 1 connections:** | 4 |
| **Max input connection size:** | 1448 bytes with Large_Forward_Open, 509 bytes with Forward_Open |
| **Max output connection size:** | 1448 bytes with Large_Forward_Open, 505 bytes with Forward_Open |
| **Supported RPI (Requested Packet Interval):** | 1... 3200ms |
| **T→O Connection type:** | Point-to-point, Multicast, Null |
| **O→-T Connection type:** | Point-to-point, Null |
| **Supported trigger types:** | Cyclic, CoS (Change of State) |
| **Supported priorities:** | Low, High, Scheduled, Urgent |

| | |
|---|---|
| **T** | Target, in this case the module |
| **O** | Origin, in this case the master |

**Connection Types**

- **Exclusive-Owner connection**

  This type of connection controls the outputs of the Anybus module and does not depend on other connections.

  | | |
  |---|---|
  | **Max. no. of Exclusive-Owner connections:** | 1 |
  | **Connection point O →T:** | Assembly Object, instance 96h (Default) |
  | **Connection point T →O:** | Assembly Object, instance 64h (Default) |

- **Input-Only connection**

  This type of connection is used to read data from the Anybus module without controlling the outputs. It does not depend on other connections.

  | | |
  |---|---|
  | **Max. no. of Input-Only connections:** | Up to 4 |
  | | (Shared with Exclusive-Owner and Listen-Only connections) |
  | **Connection point O →T:** | Assembly Object, instance 03h (Default) |
  | **Connection point T →O:** | Assembly Object, instance 64h (Default) |

  Please not that if an Exclusive-Owner connection has been opened towards the module and times out, the Input-Only connection times out as well. If the Exclusive-Owner connection is properly closed, the Input-Only connection remains unaffected.

- **Input-Only Extended connection**

  This connections functionality is the same as the standard Input-Only connection. However when this connection times out it does not affect the state of the application.

  | | |
  |---|---|
  | **Connection point O →T:** | Assembly Object, instance 06h (Default) |
  | **Connection point T →O:** | Assembly Object, instance 64h (Default) |

- **Listen-Only connection**

  This type of connection requires another connection in order to exist. If that connection (Exclusive-Owner or Input-Only) is closed, the Listen-Only connection will be closed as well.

  | | |
  |---|---|
  | **Max. no. of Input-Only connections:** | Up to 4 |
  | | (Shared with Exclusive-Owner and Input-Only connections) |
  | **Connection point O →T:** | Assembly Object, instance 04h (Default) |
  | **Connection point T →O:** | Assembly Object, instance 64h (Default) |

- **Listen-Only Extended connection**

  This connections functionality is the same as the standard Listen-Only connection. However when this connection times out it does not affect the state of the application.

  | | |
  |---|---|
  | **Connection point O →T:** | Assembly Object, instance 07h (Default) |
  | **Connection point T →O:** | Assembly Object, instance 64h (Default) |

- **Redundant-Owner connection**

  This connection type is not supported by the module.

## Class 3 Connection Details

### General

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

| | |
|---|---|
| **No. of simultaneous Class 3 connections:** | 6 |
| **Supported RPI (Requested Packet Interval):** | 100... 10000 ms |
| **T→O Connection type:** | Point-to-point |
| **O→-T Connection type:** | Point-to-point |
| **Supported trigger type:** | Application |
| **Supported connection size:** | 1526 bytes |

## 7.7          Parameter Object (0Fh)

### Category

Extended

### Object Description

The Parameter Object provides an interface to the configuration data of the module. It can provide all the information necessary to define and describe each of the module configuration parameters, as well as a full description of each parameter, including minimum and maximum values and a text string describing the parameter. Configuration tools, such as RSNetworx, can extract information about the Application Data Instances (ADIs) and present them with their actual name and range to the user.

Since this process may be somewhat time consuming, especially when using the serial host interface, it is possible to disable support for this functionality in the EtherNet/IP Host Object.

Each parameter is represented by one instance. Instance numbers start at 1, and are incremented by one, with no gaps in the list. Due to limitations imposed by the CIP standard, ADIs containing multiple elements (i.e. arrays etc.) cannot be represented through this object. In such cases, default values will be returned.

See also ....

- *ADI Object (A2h), p. 51* (CIP Object)

- *EtherNet/IP Host Object (F8h), p. 102* (Host Application Object)

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |
| | Get_Attributes_All |
| | Get_Enum_String |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0001h (Object revision) |
| 2 | Max instance | Get | UINT | Maximum created instance number = class attribute 3 in the Application Data Object (see Anybus CompactCom 40 Software Design Guide) |
| 8 | Parameter Class Descriptor | Get | WORD | Default: 0000 0000 0000 1011b |
| | | | | <table><tr><td>Bit:</td><td>Contents:</td></tr><tr><td>0</td><td>Supports parameter instances</td></tr><tr><td>1</td><td>Supports full attributes</td></tr><tr><td>2</td><td>Must do non-volatile storage save command</td></tr><tr><td>3</td><td>Parameters are stored in non-volatile storage</td></tr></table> |
| 9 | Configuration Assembly Instance | Get | UINT | 0000h (Application does not support configuration data) <br> 0005h (If the application supports configuration data, unless the configuration instance number has been changed using attribute 15 in the EtherNet/IP Host Object.) |

## Instance Attributes

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Parameter Value | Get/Set | Specified in attributes 4, 5 & 6. | Actual value of parameter<br>This attribute is read-only if bit 4 of Attribute #4 is true |
| 2 | Link Path Size | Get | USINT | 0007h (Size of link path in bytes) |
| 3 | Link Path | Get | Packed EPATH | 20 A2 25 nn nn 30 05h<br>(Path to the object from where this parameter's value is retrieved, in this case the ADI Object) |
| 4 | Descriptor | Get | WORD | Bit:     Contents:<br>0       Supports Settable Path (N/A)<br>1       Supports Enumerated Strings<br>2       Supports Scaling (N/A)<br>3       Supports Scaling Links (N/A)<br>4       Read only Parameter<br>5       Monitor Parameter (N/A)<br>6       Supports Extended Precision Scaling (N/A) |
| 5 | Data Type | Get | USINT | Data type code |
| 6 | Data Size | Get | USINT | Number of bytes in parameter value |
| 7 | Parameter Name String | Get | SHORT_STRING | Name of the parameter, truncated to 16 chars |
| 8 | Units String | Get | SHORT_STRING | "" (default string) |
| 9 | Help String | Get | SHORT_STRING | |
| 10 | Minimum Value | Get | (Data type) | Minimum value of parameter<br>The Data Type is defined in attribute 5. |
| 11 | Maximum Value | Get | (Data type) | Maximum value of parameter<br>The Data Type is defined in attribute 5. |
| 12 | Default Value | Get | (Data type) | Default value of parameter<br>The Data Type is defined in attribute 5. |
| 13 | Scaling Multiplier | Get | UINT | 0001h |
| 14 | Scaling Divisor | Get | UINT | |
| 15 | Scaling Base | Get | UINT | |
| 16 | Scaling Offset | Get | INT | 0000h |
| 17 | Multiplier Link | Get | UINT | |
| 18 | Divisor Link | Get | UINT | |
| 19 | Base Link | Get | UINT | |
| 20 | Offset Link | Get | UINT | |
| 21 | Decimal Precision | Get | USINT | 00h |

## Default Values

| #  | Name                    | Value       | Comments                 |
|----|-------------------------|-------------|--------------------------|
| 1  | Parameter Value         | 0           | -                        |
| 2  | Link Path Size          | 0           | Size of link path in bytes. |
| 3  | Link Path               | -           | NULL Path                |
| 4  | Descriptor              | 0010h       | Read only Parameter      |
| 5  | Data type               | C6h         | USINT                    |
| 6  | Data size               | 1           | -                        |
| 7  | Parameter Name String   | "Reserved"  | -                        |
| 8  | Units String            | ""          | -                        |
| 9  | Help String             | ""          | -                        |
| 10 | Minimum value           | N/A         | 0                        |
| 11 | Maximum value           | N/A         | 0                        |
| 12 | Default value           | N/A         | 0                        |
| 13 | Scaling Multiplier      | N/A         | 1                        |
| 14 | Scaling Divisor         | N/A         | 1                        |
| 15 | Scaling Base            | N/A         | 1                        |
| 16 | Scaling Offset          | N/A         | 0                        |
| 17 | Multiplier Link         | N/A         | 0                        |
| 18 | Divisor Link            | N/A         | 0                        |
| 19 | Base Link               | N/A         | 0                        |
| 20 | Offset Link             | N/A         | 0                        |
| 21 | Decimal Precision       | N/A         | 0                        |

## 7.8        Time Sync Object (43h)

### Category

Extended

### Object Description

The Time Sync Object provides a CIP interface to the IEEE 1588 (IEC 61588) Standard, the Precision Time Protocol (PTP). This protocol provides a standard mechanism to synchronize clocks across a network of distributed devices. CIP Sync is disabled by default, but can be enabled by setting attribute #32 in the Ethernet/IP Host Object (F8h).

See also ....

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |
| | Get_Attributes_List |
| | Set_Attributes_List |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0004h (Object revision) |

### Instance Attributes

| # | Name | Access | Type | Value | Description |
|---|------|--------|------|-------|-------------|
| 1 | PTPEnable | Get/Set | BOOL | - | 0 = Disabled<br>1 = Enabled (Default) |
| 2 | IsSynchronized | Get | BOOL | - | 0 = Not synchronized<br>1 = Synchronized |
| 3 | SystemTimeMicroseconds | Get | ULINT | - | Current value of system_time($\mu$s) |
| 4 | SystemTimeNanoseconds | Get | ULINT | - | Current value of system_time in nanoseconds (ns) |
| 5 | OffsetFromMaster | Get | LINT | - | Offset between local clock and master clock |
| 6 | MaxOffsetFromMaster | Get/Set | ULINT | - | Maximum offset between local clock and master clock. Monitored by the module<br>Set 0 to clear |
| 7 | MeanPathDelayToMaster | Get | LINT | - | Mean path delay to master |

| # | Name | Access | Type | Value | Description |
|---|------|--------|------|-------|-------------|
| 8 | GrandMasterClockInfo | Get | STRUCT of: | | Grandmaster Clock Info |
| | ClockIdentity | | USINT[8] | - | Clock Identity |
| | ClockClass | | UINT | - | Clock class |
| | TimeAccuracy | | UINT | - | Time Accuracy |
| | OffsetScaledLogVariance | | UINT | - | Offset scaled log variance |
| | CurrentUtcOffset | | UINT | - | Current UTC offset |
| | TimePropertyFlags | | WORD | - | Time property flags |
| | TimeSource | | UINT | - | Time source |
| | Priority1 | | UINT | - | Priority1 |
| | Priority2 | | UINT | - | Priority2 |
| 9 | ParentClockInfo | Get | STRUCT of: | | Parent Clock Info |
| | ClockIdentity | | USINT[8] | - | Clock Identity |
| | PortNumber | | UINT | - | PTP Port Number |
| | ObservedOffsetScaledLogVariance | | UINT | - | Observed offset scaled log variance |
| | ObservedPhaseChangeRate | | UDINT | - | Observed phase change rate |
| 10 | LocalClockInfo | Get | STRUCT of: | | Local Clock Info |
| | ClockIdentity | | USINT[8] | - | Clock Identity |
| | ClockClass | | UINT | - | Clock Class |
| | TimeAccuracy | | UINT | - | Time accuracy |
| | OffsetScaledLogVariance | | UINT | - | Offset scaled log variance |
| | CurrentUtcOffset | | UINT | - | Current UTC offset |
| | TimePropertyFlagsNone | | WORD | - | Time property flags |
| | TimeSource | | UINT | - | Time source |
| 11 | NumberOfPorts | Get | UINT | 1 | The device contains a hybrid clock which contains an ordinary clock and an end-to-end transparent clock. |
| 12 | PortStateInfo | Get | STRUCT of: | | PTP port state info |
| | NumberOfPorts | | UINT | 1 | Number of PTP ports. See description of attribute #11 |
| | | | ARRAY of STRUCT: | | |
| | PortNumber | | UINT | - | PTP port number |
| | PortState | | UINT | - | PTP port state |
| 13 | PortEnableCfg | Get/Set | STRUCT of: | | PTP port enable configuration |
| | NumberOfPorts | | UINT | 1 | Number of PTP ports. See description of attribute #11 |
| | | | ARRAY of STRUCT: | | |
| | PortNumber | | UINT | - | PTP port number |
| | PortEnable | | UINT | - | 0 = Disabled 1 = Enabled |
| 14 | PortLogAnnounceIntervalCfg | Get/Set | STRUCT of: | | PTP port log announce interval config |
| | NumberOfPorts | | UINT | 1 | Number of PTP ports. See description of attribute #11 |
| | | | ARRAY of STRUCT: | | PTP port log announce interval of each port |
| | PortNumber | | UINT | - | PTP port number |
| | PortLogAnnounceInterval | | UINT | - | PTP port log announce interval |

| # | Name | Access | Type | Value | Description |
|---|------|--------|------|-------|-------------|
| 15 | PortLogSyncIntervalCfg | Get/Set | STRUCT of: | | PTP port log sync interval config |
| | NumberOfPorts | | UINT | - | Number of PTP ports.<br>See description of attribute #11 |
| | | | ARRAY of STRUCT: | | |
| | PortNumber | | UINT | - | PTP port number |
| | PortLogSyncInterval | | INT | - | PTP port log sync interval |
| 18 | DomainNumber | Get/Set | USINT | - | Domain number |
| 19 | ClockType | Get | WORD | - | Clock type |
| 20 | ManufactureIdentity | Get | USINT[4] | - | The first 3 octets hold the first 3 octets of the MAC address. The 4th octet is reserved (set to 0).<br>Default: [0x00, 0x30, 0x11, 0x00] |
| 21 | ProductDescription | Get | STRUCT of: | | Contains vendor name, Product name and serial number (hexadecimal representation). Maximum length of the description is 64 bytes. The serial number plus the semicolons occupies 10 bytes. If the length of the vendor name and the product name together is longer than 54 bytes the vendor name will be truncated to not override the maximum length of 64 bytes. |
| | Size | | UDINT | 10-64 | |
| | Description | | ARRAY of USINT | "<vendor name>;<product name>;<serial number>" | |
| 22 | RevisionData | Get | STRUCT of: | | Revision data |
| | Size | | UDINT | - | Size of revision data |
| | Revision | | ARRAY of USINT | <HDL clock version>;<clock fw component version>;<clock sw component version> | Revision |
| 23 | UserDescription | Get/Set | STRUCT of: | | Format: <device name>;<location>. |
| | Size | | UDINT | 1 | |
| | Description | | ARRAY of USINT | "" | |
| 24 | PortProfileIdentityInfo | Get | STRUCT of: | | PTP port profile identity info |
| | NumberOfPorts | | UDINT | 1 | Number of PTP ports<br>See description of attribute #11 |
| | | | ARRAY of STRUCT: | | |
| | PortNumber | | UINT | - | PTP port number |
| | PortProfileIdentity | | USINT[8] | - | PTP port profile identity |
| 25 | PortPhysicalAddressInfo | Get | STRUCT of: | | PTP port physical address info |
| | NumberOfPorts | | UINT | 1 | Number of PTP ports<br>See description of attribute #11 |
| | | | ARRAY of STRUCT: | | |
| | PortNumber | | UINT | - | PTP port number |
| | PhysicalProtocol | | USINT[16] | - | Physical protocol |
| | SizeOfAddress | | UINT | - | Size of address |
| | PortPhysicalAddress | | USINT[16] | - | PTP port physical address |

| # | Name | Access | Type | Value | Description |
|---|------|--------|------|-------|-------------|
| 26 | PortProtocolAddressInfo | Get | STRUCT of: | | PTP port protocol address info |
| | NumberOfPorts | | UINT | 1 | Number of PTP ports<br>See description of attribute #11 |
| | | | ARRAY of STRUCT: | - | |
| | PortNumber | | UINT | - | PTP port number |
| | NetworkProtocol | | UINT | - | Network protocol |
| | SizeOfAddress | | UINT | - | Size of address |
| | PortProtocolAddress | | USINT[16] | - | PPT port protocol address |
| 27 | StepsRemoved | Get | UINT | - | Steps removed |
| 28 | SystemTimeAndOffset | Get | STRUCT of: | | System time and offset |
| | SystemTime | | ULINT | - | System time |
| | SystemOffset | | ULINT | - | System offset |
| 29 | Associated Interface Objects | Get | STRUCT of: | | This attribute is only implemented if the CIP Port object (F4h) is enabled by the host application.The attribute associates the PTP port with the CIP port instance #1 which represents the EtherNet/IP port of the device. |
| | Number of Ports | | UINT | 1 | |
| | | | ARRAY of STRUCT of: | | |
| | Port Number | | UINT | 1 | |
| | Associated Object Path Size | | USINT | 2 | |
| | Associated Object | | Padded EPATH | [0x20, 0xF4, 0x24, 0x01] | |

## 7.9　　　DLR Object (47h)

### Category

Extended

### Object Description

The Device Level Ring (DLR) Object provides the status information interface for the DLR protocol. This protocol enables the use of an Ethernet ring topology, and the DLR Object provides the CIP application-level interface to the protocol.

This object is not available if DLR is disabled in the EtherNet/IP Host Object, see *Ethernet Host Object (F9h), p. 111*

.

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| | Get_Attributes_All |
| **Instance:** | Get_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0003h (Object revision) |

### Instance Attributes

Attributes #1–4 and #6–7 an be customized by implementing the EtherNet/IP Host Object, see *EtherNet/IP Host Object (F8h), p. 102*

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Network Topology | Get | USINT | Bit:　Contents:<br>0　　"Linear"<br>1　　"Ring" |
| 2 | Network Status | Get | USINT | Bit:　Contents:<br>0　　"Normal" (N/A)<br>1　　"Ring Fault"<br>2　　"Unexpected Loop Detected"<br>3　　"Partial Network Fault"<br>4　　"Rapid Fault/Restore Cycle" |
| 10 | Active Supervisor Address | Get | Struct of: UDINT Array of: 6 USINTs | This attribute holds the IP address (IPv4) and/or the Ethernet Mac address of the active ring supervisor. |
| 12 | Capability Flags | Get | DWORD | 82h (Beacon-based ring node, Flush_Table frame capable) |

## 7.10        QoS Object (48h)

### Category

Extended

### Object Description

Quality of Service (QoS) is a general term that is applied to mechanisms used to treat traffic streams with different relative priorities or other delivery characteristics. Standard QoS mechanisms include IEEE 802.1D/Q (Ethernet frame priority) and Differentiated Services (DiffServ) in the TCP/IP protocol suite.

The QoS Object provides a means to configure certain QoS related behaviors in EtherNet/IP devices.

The object is required for devices that support sending EtherNet/IP messages with nonzero DiffServ code points (DSCP), or sending EtherNet/IP messages in 802.1Q tagged frames.

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0001h (Object revision) |

### Instance Attributes

Attributes #1–4 and #6–7 an be customized by implementing the EtherNet/IP Host Object, see *EtherNet/IP Host Object (F8h), p. 102*

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | 802.1Q Tag Enable | Set | USINT | Enables or disables sending 802.1Q frames.<br><br>Value:    Contents:<br>0         Disabled (Default)<br>1         Enabled |
| 2 | DSCP PTP Event | Set | USINT | DSCP value for PTP (IEEE 1588) event messages.<br>Default value = 59 |
| 3 | DSCP PTP General | Set | USINT | DSCP value for PTP (IEEE 1588) general messages.<br>Default value = 47 |
| 4 | DSCP Urgent | Set | USINT | CIP transport class 1 messages with priority Urgent<br>Default: 55 |
| 5 | DSCP Scheduled | Set | USINT | CIP transport class 1 messages with priority Scheduled<br>Default: 47 |
| 6 | DSCP High | Set | USINT | CIP transport class 1 messages with priority High<br>Default: 43 |
| 7 | DSCP Low | Set | USINT | CIP transport class 1 messages with priority Low<br>Default: 31 |
| 8 | DSCP Explicit | Set | USINT | CIP UCMM and CIP class 3<br>Default: 27 |

## 7.11 Base Energy Object (4Eh)

### Category

Extended

### Object Description

The Base Energy Object acts as an "Energy Supervisor" for CIP Energy implementations. It is responsible for providing a time base for energy values, provides energy mode services, and can provide aggregation services for aggregating energy values up through the various levels of an industrial facility. It also provides a standard format for reporting energy metering results. The object is energy type independent and allows energy type specific data and functionality to be integrated into an energy system in a standard way. The Anybus CompactCom 40 EtherNet/IP supports one instance of the Base Energy Object. For instance, an electric power monitor may count metering pulse output transitions of a separate metering device. The count of such transitions, represented by a Base Energy Object instance, would reflect the energy consumption measured by the separate metering device.

An instance of the Base Energy Object may exist as a stand-alone instance, or it may exist in conjunction with an Electrical and/or Non-Electrical Energy Object instance (These objects are not implemented in the Anybus CompactCom 40 EtherNet/IP). If an instance of any of these objects is implemented in a device, it must be associated with a Base Energy Object instance in the device.

For this object to be able to access the network, the Energy Reporting Object (E7h) must be implemented in the host application, see *Energy Reporting Object (E7h), p. 90*.

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0002h (Object revision) |

## Instance Attributes

Attributes #1–4 and #6–7 an be customized by implementing the EtherNet/IP Host Object, see *EtherNet/IP Host Object (F8h), p. 102*

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Energy/ Resource Type | Get | UINT | Type of energy managed by this instance Always 0 (Generic) |
| 2 | Base Energy Object Capabilities | Get | UINT | Always 0 (Energy measured) |
| 3 | Energy Accuracy | Get | UINT | Specifies the accuracy of power and energy metering results, either in 0.01 percent of reading (default) or 0.01 of other units specified in attribute #4. If 0, unknown. |
| 4 | Energy Accuracy Basis | Get | UINT | Always 0 (Percent of reading) |
| 7 | Consumed Energy Odometer | Get | ODOMETER | The value of the consumed energy. |
| 8 | Generated Energy Odometer | Get | ODOMETER | The value of the generated energy. |
| 12 | Energy Type Specific Object Path | Get | Struct of: UINT (Path size) padded EPATH (Path) | NULL path |

- Depending on whether the instance reports consumed or generated energy, either attribute #7 or attribute #8 is required.

- The struct data type ODOMETER makes it possible to represent very large values, for more information please consult the CIP specification Volume 1 (CIP Common).

## 7.12      Power Management Object (53h)

### Category

Extended

### Object Description

The Power Management Object provides standardized attributes and services to support the control of devices into and out of paused or sleep states. The Energy Control Object (F0h) has to be implemented for this object to gain access to the network.

See also ..

- Energy Control Object (F0h) (Anybus CompactCom 40 Software Design Guide)

### Supported Services

| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Power_Management |
| | Set_Pass_Code |
| | Clear_Pass_Code |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0002h (Object revision) |

## Instance Attributes

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Power Management Command | Get | DWORD | Collection of bit fields comprising the most recent power management request. |
| 2 | Power Management Status | Get | DWORD | Collection of bit fields providing Power Management status information. |
| 3 | Client Path | Get | Struct of: | Specifies the EPATH from this instance (server) to its current owner (client). |
|   |   |   | UINT (Path Size) | Size of path (in words |
|   |   |   | Padded EPATH (Path) | |
| 4 | Number of Power Management Modes | Get | UINT | Number of Power Management Mode array entries in attribute 5. |
| 5 | Power Management Nodes | Get | Array of: | Array of low power modes |
|   |   |   | Struct of: | Modes (Array of mode structures) |
|   |   |   | USINT | Minimum Pause Units (Specifies the unit of Minimum Pause Time) |
|   |   |   | UINT | Minimum Pause Time |
|   |   |   | USINT | Resume Units (Specifies the unit of Resume Time) |
|   |   |   | UINT | Resume Time (Required time to transition from the paused stated to the owned state. |
|   |   |   | REAL | Power Level (Power in kW for this mode) |
|   |   |   | BOOL | Availability (Specifies whether this mode can be entered given the current device state) |
| 6 | Sleeping State Support | Get | BOOL | 0 (Sleeping state not supported) |

## 7.13        ADI Object (A2h)

### Category

Extended

### Object Description

This object maps instances in the Application Data Object to EtherNet/IP. All requests to this object will be translated into explicit object requests towards the Application Data Object in the host application; the response is then translated back to CIP-format and sent to the originator of the request.

The ADI Object can be disabled using attribute 30 in the EtherNet/IP Host Object (F8h). This attribute can also be used to change the ADI Object number.

See also ..

•    Application Data Object (see Anybus CompactCom 40 Software Design Guide)

•    *Parameter Object (0Fh), p. 38* (CIP Object)

•    *EtherNet/IP Host Object (F8h), p. 102*

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_Single |
| **Instance:** | Get_Attribute_Single |
| | Set_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0002h (Object revision) |
| 2 | Max Instance | Get | UINT | Equals attribute #4 in the Application Data Object |
| 3 | Number of instances | Get | UINT | Equals attribute #3 in the Application Data Object |

For information about the Application Data Object, please consult the Anybus CompactCom 40 Software Design Guide.

## Instance Attributes

Each instance corresponds to an instance within the Application Data Object (for more information, please consult the general Anybus CompactCom 40 Software Design Guide).

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Name | Get | SHORT_STRING | Parameter name (Including length) |
| 2 | ABCC Data type | Get | Array of USINT | Data type of instance value |
| 3 | No. of Elements | Get | USINT | Number of elements of the specified data type |
| 4 | Descriptor | Get | Array of USINT | Bit field describing the access rights for this instance<br><br>Bit:    Meaning:<br>0     1 = Get Access<br>1     1 = Set Access<br>2     (reserved, set to 0)<br>3     1 = Write process data mapping possible<br>4     1 = Read process data mapping possible<br>5     1 = NVS parameter<br>6     1 = Data notification enabled |
| 5 | Value | Get/Set | Determined by attributes #2, #3 and #9 | Instance value |
| 6 | Max Value | Get | | The maximum permitted parameter value. |
| 7 | Min Value | Get | | The minimum permitted parameter value. |
| 8 | Default Value | Get | | The default parameter value. |
| 9 | Number of subelements | Get | Array of UINT | Number of subelements in the ADI. Default value is 1 unless implemented in the application.<br>The size of the array depends on attribute #3. |

Attributes #5–8 are converted to/from CIP standard by the module

## 7.14        Port Object (F4h)

### Category

Extended

### Object Description

The Port Object describes the CIP ports present on the device. Each routable CIP port is described in a separate instance. Non-routable ports may be described. Devices with a single CIP port are not required to support this object.

The object exists only if enabled in the EtherNet/IP Host Object (Instance Attribute #17).

See also ..

- *EtherNet/IP Host Object (F8h), p. 102* (Anybus Module Object)

- *CIP Port Configuration Object (0Dh), p. 79* (Host Application Object)

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attributes_All |
| | Get_Attribute_Single |
| **Instance:** | Get_Attributes_All |
| | Get_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0002h (Object revision) |
| 2 | Max Instance | Get | UINT | Max. instance number |
| 3 | Number of Instances | Get | UINT | Number of ports currently created. |
| 8 | Entry Port | Get | UINT | Returns the instance of the Port Object that describes the port through which this request entered the device. |
| 9 | Port Instance Info | Get | Array of: | Array of structures containing instance attributes 1 and 2 from each instance. The array is indexed by instance number, up to the maximum number of instances. The value at index 1 (offset 0) and any non-instantiated instances will be zero. |
| | | | Struct of: UINT (Type) UINT (Number) | Enumerates the type of port (see instance attribute #1) CIP port number associated with this port (see instance attribute #2) |

## Instance Attributes (Instance #1)

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Port Type | Get | UINT | 0h (default)<br>4h (if the application registers a port) |
| 2 | Port Number | Get | UINT | 2h |
| 3 | Link Object | Get | Struct of:<br>  UINT<br>  Padded EPATH | -<br>2h (Path Length)<br>20 F5 24 01h (Link Path) |
| 4 | Port Name | Get | SHORT_STRING | "EtherNet/IP" |
| 5 | Port Type Name | Get | SHORT_STRING | "" |
| 6 | Port Description | Get | SHORT_STRING | "" |
| 7 | Node Address | Get | Padded EPATH | - |
| 10 | Port Routing Capabilities | Get | UDINT | 1h (Routing of incoming Unconnected Messaging supported) |

See also...

*CIP Port Configuration Object (0Dh), p. 79*

## Instance Attributes (Instances #2... #8)

| # | Name | Access | Type | Value/Description |
|---|------|--------|------|-------------------|
| 1 | Port Type | Get | UINT | Enumerates the type of port |
| 2 | Port Number | Get | UINT | CIP port number associated with this port |
| 3 | Link Object | Get | Struct of:<br>  UINT<br>  Padded EPATH | -<br>Path length (number of 16-bit words)<br>Logical path segments which identify the object for this port. The path must consist of one logical class segment and one logical instance segment. The maximum size is 12 bytes. |
| 4 | Port Name | Get | SHORT_STRING | Name of port, e.g. "Port A". Max. 64 characters. |
| 5 | Port Type Name | Get | SHORT_STRING | "" |
| 6 | Port Description | Get | SHORT_STRING | "" |
| 7 | Node Address | Get | Padded EPATH | Node number of this device on port. The range within this data type is restricted to a Port Segment. |
| 8 | Port Node Range | Get | Struct of:<br>  UINT (Min.)<br>  UINT (Max.) | -<br>Min. node number on port<br>Max. node number on port |
| 10 | Port Routing Capabilities | Get | UDINT | 1h (Routing of incoming Unconnected Messaging supported) |

See also...

*CIP Port Configuration Object (0Dh), p. 79* , "Instance Attributes.".

## 7.15        TCP/IP Interface Object (F5h)

### Category

Extended

### Object Description

This object provides the mechanism to configure the TCP/IP network interface of the module. It groups the TCP/IP-related settings in one instance for each TCP/IP capable communications interface.

See also ..

* *Communication Settings, p. 11*

* *Network Configuration Object (04h), p. 68* (Anybus Module Object)

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attribute_All |
| | Get_Attribute_Single |
| **Instance:** | Get_Attribute_All |
| | Get_Attribute_Single |
| | Set_Attribute_Single |

### Class Attributes

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Revision | Get | UINT | 0004h (Object revision) |
| 2 | Max instance | Get | UINT | 1 (Maximum instance number) |
| 3 | Number of instances | Get | UINT | 1 (Number of instances) |
| 6 | Maximum ID Number Class Attributes | Get | UINT | 7 (The attribute number of the last implemented class attribute) |
| 7 | Maximum ID Number Instance Attributes | Get | UINT | 13 (The attribute number of the last implemented instance attribute) |

## Instance Attributes

| # | Name | Access | Type | Value | Comments |
|---|------|--------|------|-------|----------|
| 1 | Status | Get | DWORD | - | Bit:     Meaning:<br>(reserved, set to 0)<br><br>0–3     When set to h, attribute #5 contains valid configuration from DHCP or non-volatile storage. When set to 2h, attribute #5 contains valid configuration from hardware settings. Remaining values are reserved for future use.<br><br>4     Multicast pending if set to 1.<br><br>5     Interface configuration pending if set to 1. A new configuration will be loaded at the next reset.<br><br>6     AcdStatus. Set to 1 if an address conflict is detected. Address conflict detection is enabled/ disabled in attribute #10.<br><br>7     AcdFault<br><br>8–31     (reserved, set to 0) |
| 2 | Configuration Capability | Get | DWORD | - | Bit:     Meaning:<br><br>0-1:     Always 0. For more information, consult the CIP specifications.<br><br>2:     If set to 1, the device is capable of acting as a DHCP client. The bit is set to 0 if attribute #24 (Enable DHCP Client) is disabled in the *Ethernet Host Object (F9h), p. 111*<br><br>3:     Always 0. For more information, consult the CIP specifications.<br><br>4:     The 'Configuration Settable'-bit reflects the value of instance attribute #9 in the "EtherNet/IP Host Object (F8h)" on page 161.<br><br>5:     The module is hardware configurable when this bit is set to 1. The bit will be set if any of the address attributes is set in the Network Configuration Object (04h) during setup or if attribute #6 (Hardware configurable address) in the Application Object (FFh) is set.<br><br>6:     Always 0. For more information, consult the CIP specifications.<br><br>7:     If set to 1, the device is capable of detecting address conflicts. The bit is set to 0 if address conflict detection is disabled in the Ethernet Host Object, see page *111*<br><br>8 - 31:     (reserved, set to 0) |
| 3 | Configuration Control | Get/Set | DWORD | - | Value:     Meaning<br><br>0:     Configuration from non-volatile memory<br><br>2:     Configuration from DHCP |
| 4 | Physical Link Object | Get | Struct of:<br>UINT (Path size)<br>Padded EPATH | -<br>0002h<br>20 F6 24 03h | -<br>-<br>Path to Ethernet Link Object, Instance #3 |
| 5 | Interface Configuration | Get/Set | Struct of:<br>  UDINT (IP)<br>  UDINT (Mask)<br>  UDINT (GW)<br>  UDINT (DNS1)<br>  UDINT (DNS2)<br>  STRING (Domain) | | -<br>IP address<br>Subnet mask<br>Default gateway<br>Primary DNS<br>Secondary DNS<br>Default domain |
| 6 | Host Name | Get/Set | STRING | - | Host name of Anybus module |
| 8 | TTL Value | Get/Set | USINT | 1 | TTL value for EtherNet/IP multicast packets |
| 9 | Mcast Config | Set | Struct of: | | IP multicast configuration. |

| # | Name | Access | Type | Value | Comments |
|---|------|--------|------|-------|----------|
| | Alloc Control | | USINT | 0 | **Value:** **Meaning:** <br> 0: Use default allocation algorithm to generate multicast addresses <br> 1: Allocate multicast addresses according to the values in the "Num Mcast"- and "Mcast Start Addr"-fields. |
| | (reserved) | | USINT | 0 | Set to zero. Do not change. |
| | Num Mcast | | UINT | -1 | Number of multicast addresses to allocate for EtherNet/IP |
| | Mcast Start Addr | | UDINT | - | Starting multicast address from which to begin allocation |
| 10 | SelectAcd | Set | Bool | 1 | **Value:** **Meaning:** <br> 0: Disable ACD <br> 1: Enable ACD (Default). <br> If ACD (address conflict detection) is enabled, bit 6 in attribute #1 will be set if an ACD conflict is detected. The Network Status LED will also indicate a detected conflict, see *Front View, p. 120* . |
| 11 | LastConflictDetected | Set | Struct of: | | ACD Diagnostic parameters Related to the last conflict detected. |
| | AcdActiviity | | USINT | - | State of ACD activity when last conflict detected. |
| | RemoteMAC | | ARRAY of 6 USINT | - | MAC address of remote node form the ARP PDU in which a conflict was detected. |
| | ArpPdu | | ARRAY of 28 USINT | - | Copy of the raw ARP PDU in which a conflict was detected. |
| 12 | EIP QuickConnect | Set | Bool | 0 | **Value:** **Meaning:** <br> 0: Disable EIP QuickConnect (Default) <br> 1: Enable EIP QuickConnect <br> If EIP QuickConnect is enabled, the QuickConnect feature will direct EtherNet/IP target devices to quickly power up and join an EtherNet/IP network. |
| 13 | Encapsulation inactivity timeout | Set | UINT | 0 - 3600 | Number of seconds of inactivity before a TCP connection is closed. <br> 0: Disabled |

- Support for configuring network settings (attributes #3 and #5) from the network can be disabled by implementing attribute #9 in the EtherNet/IP Host Object, see *EtherNet/IP Host Object (F8h), p. 102*

- Attributes #10 and #11 will not be available if ACD is disabled using attribute #11 in the Ethernet Host Object (F9h).

- Attribute #12:

  - If the module is configured to use EIP QuickConnect functionality, the EDS file has to be changed. As the EDS file is changed, the identity of the module has to be changed and the module will require certification.

  - This attribute exists if attribute #26 in the EtherNet/IP Host Object is implemented, see *EtherNet/IP Host Object (F8h), p. 102*.

## 7.16 Ethernet Link Object (F6h)

### Category

Extended

### Object Description

This object maintains link specific counters and status information for an IEEE 802.3 communications interface. Exactly one instance for each communications interface on the module is supported. Instances for internally accessible interfaces can also be supported.

See also ..

- *Communication Settings, p. 11*
- *Network Configuration Object (04h), p. 68* (Anybus Module Object)

### Supported Services

| | |
|---|---|
| **Class:** | Get_Attributes_All |
| | Get_Attribute_Single |
| **Instance:** | Get_Attributes_All |
| | Get_Attribute_Single |
| | Set_Attribute_Single |
| | Get_And_Clear |

### Class Attributes

By default, three instances (port 1, port 2 and the internal port) are implemented, meaning that two ports are activated.

If port 2 is inactivated in the Port 2 State attribute of the Ethernet Host Object (F9h), only one instance (port 1) should be implemented.

| # | Name | Access | Type | Value |
|---|---|---|---|---|
| 1 | Revision | Get | UINT | 0004h (Object revision) |
| 2 | Max Instance | Get | UINT | 1 or 3 (Maximum instance number) |
| 3 | Number of Instances | Get | UINT | 1 or 3 (Number of instances) |
| 6 | Maximum ID Number Class Attributes | Get | UINT | 7 (The attribute number of the last implemented class attribute.) |
| 7 | Maximum ID Number Instance Attributes | Get | UINT | 11 (The attribute number of the last implemented instance attribute.) |

## Instance Attributes

| # | Name | Access | Type | Value | Comments |
|---|------|--------|------|-------|----------|
| 1 | Interface Speed | Get | UDINT | 10 or 100 | Actual Ethernet interface speed. |
| 2 | Interface Flags | Get | DWORD | - | See table "Interface Flags" below. |
| 3 | Physical Address | Get | Array of 6 USINTs | (MAC ID) | Physical network address, i.e. assigned MAC address. |
| 4 | Interface Counters | Get | Struct of: | | |
| | In Octets | | UDINT | N/A | Octets received on the interface |
| | In Ucast Packets | | UDINT | N/A | Unicast packets received on the interface |
| | In NUcast Packets | | UDINT | N/A | Nonunicast packets received on the interface |
| | In Discards | | UDINT | N/A | Inbound packets with unknown protocol |
| | In Errors | | UDINT | N/A | Inbound packets that contain errors (does not include In discards) |
| | In Unknown Protos | | UDINT | N/A | Inbound packets with unknown protocol |
| | Out Octets | | UDINT | N/A | Octets sent on the interface |
| | Out Ucast Packets | | UDINT | N/A | Unicast packets sent on the interface |
| | Out NUcast Packets | | UDINT | N/A | Nonunicast packets sent on the interface |
| | Out Discards | | UDINT | N/A | Outbound packets with unknown protocol |
| | Out Errors | | UDINT | N/A | Outbound packets that contain errors (does not include Out discards) |
| 5 | Media Counters | Get | Struct of: | | Media specific counters |
| | Alignment Errors | | UDINT | N/A | Frames received that are not an integral number of octets in length |
| | FCS Errors | | UDINT | N/A | Frames received that do not pass the FCS check |
| | Single Collisions | | UDINT | N/A | Successfully transmitted frames that have experienced exactly one collision |
| | Multiple Collisions | | UDINT | N/A | Successfully transmitted frames that have experienced more than one collision |
| | SQE Test Errors | | UDINT | 0 | The number of times the SQE test error message is generated(Counter not provided with current PHY interface) |
| | Deferred Transmissions | | UDINT | N/A | Frames for which the first transmission attempt is delayed because the medium is busy |
| | Late Collisions | | UDINT | N/A | The number of times a collision is detected later than 512 bit-times into the transmission of a packet |
| | Excessive Collisions | | UDINT | N/A | Frames for which a transmission fails due to excessive collisions |
| | MAC Transmit Errors | | UDINT | N/A | Frames for which a transmission fails due to an internal MAC sublayer receive error |
| | Carrier Sense Errors | | UDINT | N/A | The number of times that the carrier sense condition was lost or never asserted when attempting to transmit a frame |
| | Frame Too Long | | UDINT | N/A | Frames received that exceed the maximum permitted frame size |
| | MAC Receive Errors | | UDINT | N/A | Frames for which reception on an interface fails due to an internal MAC sublayer receive error |

| # | Name | Access | Type | Value | Comments |
|---|------|--------|------|-------|----------|
| 6 | Interface Control | Get/Set | Struct of: | | |
| | Control Bits | | WORD | - | Interface control bits |
| | Forced Interface Speed | | UINT | - | Speed at which the interface shall be forced to operate. Returns 'Object state Conflict' if auto-negotiation is enabled |
| 7 | Interface Type | Get | USINT | - | See table "Interface State" below. |
| 8 | Interface State | Get | USINT | - | See table "Interface Type" below. |
| 9 | Admin State | Get/Set | USINT | - | See table "Admin State" below. |
| 10 | Interface Label | Get | SHORT_STRING | — | See table "Interface Label" below. |
| 11 | Interface Capability | Get | Struct of: | - | Indication of the capabilities of the interface |
| | Capability Bits | | DWORD | - | Interface capabilities, other than speed/duplex See table "Interface Capability" below. |
| | Speed/Duplex Options | | Struct of: | - | Indicates speed/duplex pairs supported in the Interface Control Attribute |
| | | | USINT | - | Speed/duplex array count |
| | | | Array of Struct of: | - | Speed/duplex array |
| | | | UINT | - | Interface speed |
| | | | USINT | - | Interface Duplex Mode 0 = half duplex 1 = full duplex 2 - 255 = Reserved |

- Support for attribute #6 can be disabled by implementing attribute #9 in the EtherNet/IP Host Object (F8h). see *EtherNet/IP Host Object (F8h), p. 102*

- Support for attribute #9 can be disabled by implementing the port state attributes (#12 or #13) in the Ethernet Host object (F9h) see *Ethernet Host Object (F9h), p. 111*

### Interface Flags

| Bit | Name | Description | | |
| --- | --- | --- | --- | --- |
| 0 | Link status | Indicates whether or not the Ethernet 802.3 communications interface is connected to an active network. | | |
| | | Value: | Meaning: | |
| | | 0 | Inactive link | |
| | | 1 | Active link | |
| 1 | Half/full duplex | Indicates the duplex mode currently in use. | | |
| | | Value: | Meaning: | |
| | | 0 | Half duplex | |
| | | 1 | Full duplex | |
| 2 - 4 | Negotiation Status | Indicates the status of link auto-negotiation. | | |
| | | Value: | Meaning: | |
| | | 0 | Auto-negotiation in progress. | |
| | | 1 | Auto-negotiation and speed detection failed (using default values) (Recommended default values are 10 Mbps, half duplex) | |
| | | 2 | Auto negotiation failed but detected speed (using default duplex value) | |
| | | 3 | Successfully negotiated speed and duplex. | |
| | | 4 | Auto-negotiation not attempted. Forced speed and duplex. | |
| 5 | Manual Setting requires Reset | Value: | Meaning: | |
| | | 0 | Interface can activate changes to link parameters during runtime | |
| | | 1 | Reset is required in order for changes to have effect | |
| 6 | Local Hardware Fault | Value: | Meaning: | |
| | | 0 | No local hardware fault detected | |
| | | 1 | Local hardware fault detected | |
| 7-31 | (reserved) | Set to 0. | | |

### Interface State

This attribute indicates the current operational state of the interface.

| Value | Description |
| --- | --- |
| 0 | Unknown interface state. |
| 1 | The interface is enabled and is ready to send and receive data. |
| 2 | The interface is disabled. |
| 3 | The interface is testing. |

### Admin State

This attribute controls the administrative setting of the interface state.

| Value | Description |
| --- | --- |
| 0 | (reserved) |
| 1 | Enable the interface. |
| 2 | Disable the interface. |
| 3-255 | (reserved) |

### Interface Label

This attribute is configurable via the EtherNet/IP Host Object, see page

| Instance | Value |
|---|---|
| 1 | Port 1 |
| 2 | Port 2 |
| 3 | Internal |

### Interface Type

| Instance | Value | Description |
|---|---|---|
| 1 | 2 | Twisted-pair |
| 2 | 2 | Twisted-pair |
| 3 | 1 | Internal interface |

### Interface Capability

| Bit | Name | Description | | Implementation |
|---|---|---|---|---|
| 0 | Manual setting requires reset | Indicates whether or not the device requires a reset to apply changes made to the Interface Control attribute (#6). | | Return 0 |
| | | 0 | Indicates that the device automatically applies changes made to the Interface Control attribute (#6) and, therefore, does not require a reset in order for changes to take effect. This bit shall have this value when the Interface Control attribute (#6) is not implemented. | |
| | | 1 | 1 = Indicates that the device does not automatically apply changes made to the Interface Control attribute (#6) and, therefore, will require a reset in order for changes to take effect. Note: this bit shall also be replicated in the Interface Flags attribute (#2), in order to retain backwards compatibility with previous object revisions. | |
| 1 | Auto-negotiate | 0 | Indicates that the interface does not support link auto-negotiation | 0 for internal interface, 1 for external interfaces |
| | | 1 | Indicates that the interface supports link auto-negotiation | |
| 2 | Auto-MDIX | 0 | Indicates that the interface does not support auto MDIX operation | 0 for internal interface, 1 for external interfaces |
| | | 1 | Indicates that the interface supports auto MDIX operation | |
| 3 | Manual speed/ duplex | 0 | Indicates that the interface does not support manual setting of speed/duplex. The Interface Control attribute (#6) shall not be supported. | 0 for internal interface, 1 for external interfaces |
| | | 1 | Indicates that the interface supports manual setting of speed/duplex via the Interface Control attribute (#6) | |
| 4 - 31 | Reserved | Shall be set to 0 | | Return 0 |

# 8      Anybus Module Objects

## 8.1      General Information

This chapter specifies the Anybus Module Object implementation and how they correspond to the functionality in the Anybus CompactCom 40 EtherNet/IP.

Standard Objects:

Network Specific Objects:

## 8.2        Anybus Object (01h)

### Category

Basic

### Object Description

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

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

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |
| | Set_Attribute |
| | Get_Enum_String |

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

Basic

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

Extended

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 17 | Virtual attributes | Get/Set | - | Consult the general Anybus CompactCom 40 Software Design Guide for further information. |
| 18 | Black list/White list | Get/Set | | |
| 19 | Network time | Get | UINT64 | If CIP sync is enabled and the device is synchronized to a PTP master, the network time will presented here in the format 64-bit value based on IEEE-1588. Expressed as 64-bit nanoseconds. Base 23:59:51. 51.999918, December 31, 1969. In all other cases the value is 0 (Not supported). |

## 8.3          Diagnostic Object (02h)

### Category

Basic

### Object Description

This object provides a standardized way of handling host application events & diagnostics, and is

thoroughly described in the general *Anybus CompactCom 40 Software Design Guide*.

### Supported Commands

| **Object:** | Get_Attribute |
|---|---|
| | Create |
| | Delete |
| **Instance:** | Get_Attribute |

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

Extended

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

Attributes #2 and #4–7 can not be represented on the network and are ignored by the module.

In this implementation, the severity level of all instances are combined (using logical OR) and represented on the network through the CIP Identity Object.

## 8.4        Network Object (03h)

### Category

Basic

### Object Description

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

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |
| | Set_Attribute |
| | Get_Enum_String |
| | Map_ADI_Write_Area |
| | Map_ADI_Read_Area |
| | Map_ADI_Write_Ext_Area |
| | Map_ADI_Read_Ext_Area |

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

Basic

| # | Name | Access | Type | Value |
|---|------|--------|------|-------|
| 1 | Network type | Get | UINT16 | 009Bh (EtherNet/IP Beacon Based 2–port) |
| 2 | Network type string | Get | Array of CHAR | "Ethernet/IP(TM)" |
| 3 | Data format | Get | ENUM | 00h (LSB first) |
| 4 | Parameter data support | Get | BOOL | True |
| 5 | Write process data size | Get | UINT16 | Current write process data size (in bytes)<br>Updated on every successful Map_ADI_Write_Area. (Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.) |
| 6 | Read process data size | Get | UINT16 | Current read process data size (in bytes)<br>Updated on every successful Map_ADI_Read_Area. (Consult the general *Anybus CompactCom 40 Software Design Guide* for further information.) |
| 7 | Exception Information | Get | UINT8 | Additional information available if the module has entered the EXCEPTION state.<br><br>Value:    Meaning:<br>00h    No information available<br>01h    Invalid assembly instance mapping<br>02h    Missing MAC address (Only valid for Anybus IP) |

## 8.5　　　　Network Configuration Object (04h)

### Category

Extended

### Object Description

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

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

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

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

See also...

- *Communication Settings, p. 11*

- *TCP/IP Interface Object (F5h), p. 55* (CIP-object)

- *Ethernet Link Object (F6h), p. 58*

- *Ethernet Host Object (F9h), p. 111*

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| | Reset |
| **Instance:** | Get_Attribute |
| | Set_Attribute |
| | Get_Enum_String |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value | Description |
|---|------|--------|-----------|-------|-------------|
| 3 | Number of instances | Get | UINT16 | 13 | Supported number of instances |
| 4 | Highest instance number | Get | UINT16 | 20 | Highest instance number |

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

## Instance Attributes (Instance #3, IP Address)

Value is used after module reset.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "IP address" <br> (Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 04h (= UINT8) |
| 3 | Number of elements | Get | UINT8 | 04h (four elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of UINT8 | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |
| 6 | Configured Value | Get | Array of UINT8 | Holds the configured value, which will be written to attribute #5 after the module has been reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |

## Instance Attributes (Instance #4, Subnet Mask)

Value is used after module reset.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Subnet mask" <br> (Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 04h (= UINT8) |
| 3 | Number of elements | Get | UINT8 | 04h (four elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of UINT8 | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |
| 6 | Configured Value | Get | Array of UINT8 | Holds the configured value, which will be written to attribute #5 after the module has been reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |

## Instance Attributes (Instance #5, Gateway Address)

Value is used after module reset.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Gateway" <br> (Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 04h (= UINT8) |
| 3 | Number of elements | Get | UINT8 | 04h (four elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of UINT8 | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |
| 6 | Configured Value | Get | Array of UINT8 | Holds the configured value, which will be written to attribute #5 after the module has been reset. <br> Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |

## Instance Attributes (Instance #6, DHCP Enable)

Value is used after module reset.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "DHCP"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 08h (= ENUM) |
| 3 | Number of elements | Get | UINT8 | 01h (one element) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | ENUM | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset.<br>(Multilingual, see page *75*)<br><br><u>Value</u>   <u>String</u>   <u>Meaning</u><br>00h   "Disable"   DHCP disabled<br>01h   "Enable"   DHCP enabled (default) |
| 6 | Configured Value | Get | ENUM | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br><br><u>Value</u>   <u>String</u>   <u>Meaning</u><br>00h   "Disable"   DHCP disabled<br>01h   "Enable"   DHCP enabled |

## Instance Attributes (Instance #7 Ethernet Communication Settings 1)

Changes have immediate effect.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Comm 1"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 08h (= ENUM) |
| 3 | Number of elements | Get | UINT8 | 01h (one element) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | ENUM | <u>Value</u>   <u>String</u>   <u>Meaning</u><br>     (Multilingual, see page *75*)<br><br>00h   "Auto"   Auto negotiation (default)<br>01h   "10 HDX"   10Mbit, half duplex<br>02h   "10 FX"   10Mbit, full duplex<br>03h   "100HDX"   100Mbit, half duplex<br>04h   "100FX"   100Mbit, full duplex |
| 6 | Configured Value | Get | ENUM | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br><br><u>Value</u>   <u>String</u>   <u>Meaning</u><br>     (Multilingual, see page *75*)<br><br>00h   "Auto"   Auto negotiation<br>01h   "10 HDX"   10Mbit, half duplex<br>02h   "10 FX"   10Mbit, full duplex<br>03h   "100HDX"   100Mbit, half duplex<br>04h   "100FX"   100Mbit, full duplex |

### Instance Attributes (Instance #8 Ethernet Communication Settings 2)

Changes have immediate effect.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Comm 2"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 08h (= ENUM) |
| 3 | Number of elements | Get | UINT8 | 01h (one element) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | ENUM | <u>Value</u>  <u>String</u>  <u>Meaning</u><br>(Multilingual, see page *75*)<br><br>00h  "Auto"  Auto negotiation (default)<br>01h  "10 HDX"  10Mbit, half duplex<br>02h  "10 FX"  10Mbit, full duplex<br>03h  "100HDX"  100Mbit, half duplex<br>04h  "100FX"  100Mbit, full duplex |
| 6 | Configured Value | Get | ENUM | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br><u>Value</u>  <u>String</u>  <u>Meaning</u><br>(Multilingual, see page *75*)<br><br>00h  "Auto"  Auto negotiation<br>01h  "10 HDX"  10Mbit, half duplex<br>02h  "10 FX"  10Mbit, full duplex<br>03h  "100HDX"  100Mbit, half duplex<br>04h  "100FX"  100Mbit, full duplex |

### Instance Attributes (Instance #9, DNS1)

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

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "DNS1"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 04h (= UINT8) |
| 3 | Number of elements | Get | UINT8 | 04h (four elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of UINT8 | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset.<br>Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |
| 6 | Configured Value | Get | Array of UINT8 | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br>Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |

## Instance Attributes (Instance #10, DNS2)

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

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "DNS2"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 04h (= UINT8) |
| 3 | Number of elements | Get | UINT8 | 04h (four elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of UINT8 | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset.<br>Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |
| 6 | Configured Value | Get | Array of UINT8 | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br>Valid range: 0.0.0.0 - 255.255.255.255 (Default =0.0.0.0) |

## Instance Attributes (Instance #11, Host name)

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

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Host name"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 07h (= CHAR) |
| 3 | Number of elements | Get | UINT8 | 40h (64 elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of CHAR | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset.<br>Host name, 64 characters |
| 6 | Configured Value | Get | Array of CHAR | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br>Host name, 64 characters |

## Instance Attributes (Instance #12, Domain name)

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

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "Host name"<br>(Multilingual, see page *75*) |
| 2 | Data type | Get | UINT8 | 07h (= CHAR) |
| 3 | Number of elements | Get | UINT8 | 30h (48 elements) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | Array of CHAR | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset.<br>Domain name, 48 characters |
| 6 | Configured Value | Get | Array of CHAR | Holds the configured value, which will be written to attribute #5 after the module has been reset.<br>Domain name, 48 characters |

## Instance Attributes (Instances #13 - #15)

(Reserved)

## Instance Attributes (Instance #16, MDI 1 Settings )

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

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

## Instance Attributes (Instance #17, MDI 2 Settings )

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

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

## Instance Attributes (Instances #18 and #19)

These instances are reserved for future attributes.

## Instance Attributes (Instance #20, QuickConnect)

This instance enables or disables the QuickConnect functionality from the application. Changes are valid after reset or power cycle. The value of the QuickConnect attribute (#12) in the TCP/IP Interface object (F5h), will change immediately.

This instance has no effect unless QuickConnect is enabled in the EtherNet/IP host object. If QuickConnect is disabled in the EtherNet/IP host object the application is advised to hide this instance to the end-user.

See also...

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Name | Get | Array of CHAR | "QuickConnect" |
| 2 | Data type | Get | UINT8 | 08h (= ENUM) |
| 3 | Number of elements | Get | UINT8 | 01h (one element) |
| 4 | Descriptor | Get | UINT8 | 07h (read/write/shared access) |
| 5 | Value | Get/Set | ENUM | If read, the actual value will be received. If written, the written value is reflected in attribute #6 until a reset. <br><br> Value:         Meaning: <br> 00h           Disable (default) <br> 01h           Enable |
| 6 | Configured Value | Get | ENUM | Holds the configured value, which will be written to attribute #5 after the module has been reset. <br><br> Value:         Meaning: <br> 00h           Disable <br> 01h           Enable |

## Multilingual Strings

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

| Instance | English | German | Spanish | Italian | French |
|---|---|---|---|---|---|
| 3 | IP address | IP-Adresse | Dirección IP | Indirizzo IP | Adresse IP |
| 4 | Subnet mask | Subnetzmaske | Masac. subred | Sottorete | Sous-réseau |
| 5 | Gateway | Gateway | Pasarela | Gateway | Passerelle |
| 6 | DHCP | DHCP | DHCP | DHCP | DHCP |
| | Enable | Einschalten | Activado | Abilitato | Activé |
| | Disable | Ausschalten | Desactivado | Disabilitato | Désactivé |
| 7 | Comm 1 | Komm 1 | Comu 1 | Connessione 1 | Comm 1 |
| | Auto | Auto | Auto | Auto | Auto |
| | 10 HDX | 10 HDX | 10 HDX | 10 HDX | 10 HDX |
| | 10 FDX | 10 FDX | 10 FDX | 10 FDX | 10 FDX |
| | 100 HDX | 100 HDX | 100 HDX | 100 HDX | 100 HDX |
| | 100 FDX | 100FDX | 100 FDX | 100 FDX | 100 FDX |
| 8 | Comm 2 | Komm 2 | Comu 2 | Connessione 2 | Comm 2 |
| | Auto | Auto | Auto | Auto | Auto |
| | 10 HDX | 10 HDX | 10 HDX | 10 HDX | 10 HDX |
| | 10 FDX | 10 FDX | 10 FDX | 10 FDX | 10 FDX |
| | 100 HDX | 100 HDX | 100 HDX | 100 HDX | 100 HDX |
| | 100 FDX | 100FDX | 100 FDX | 100 FDX | 100 FDX |
| 9 | DNS1 | DNS 1 | DNS Primaria | DNS1 | DNS1 |
| 10 | DNS2 | DNS 2 | DNS Secundia. | DNS2 | DNS2 |
| 11 | Host name | Host name | Nombre Host | Nome Host | Nom hôte |
| 12 | Domain name | Domain name | Nobre Domain | Nome Dominio | Dom Domaine |

## 8.6        Anybus File System Interface Object (0Ah)

**Category**

Extended

**Object Description**

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

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

Instances are created and deleted dynamically during runtime.

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

## 8.7        Network Ethernet Object (0Ch)

### Category

Extended

### Object Description

This object provides Ethernet-specific information to the application.

The object has three instances, each corresponding to a port:

| Instance # | Port |
|---|---|
| 1 | Internal port |
| 2 | Port 1 |
| 3 | Port 2 |

Each instance provides statistic counters for the port. This information can e.g be presented on internal web pages, if present, using the JSON script language.

**ⓘ**    *Instance attribute #1 is reserved and used for backwards compatibility with earlier applications.*

### Supported Commands

**Object:**                    Get_Attribute

**Instance:**                  Get_Attribute

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

| # | Name | Access | Data Type | Description |
|---|---|---|---|---|
| 1 | MAC Address | Get | Array of UINT8 | Reserved, used for backwards compatibility.<br>(Device MAC address.)<br>(See also *Ethernet Host Object (F9h), p. 111*) |
| 2 | (Reserved) | | | |
| 3 | (Reserved) | | | |
| 4 | MAC Address | Get | Array of UINT8 | Device MAC address |
| 5 | Interface Counters | Get | Array of UINT32 | Array containing MIB-II interface counters (rfc1213)<br>See table below for array indices. |
| 6 | (Reserved) | | | |

## Instance Attributes (Instances #2 - #3)

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 - 4 | (Reserved) | | | |
| 5 | Interface Counters | Get | Array of UINT32 | Array containing MIB-II interface counters (rfc1213) See table below for array indices. |
| 6 | Media Counters | Get | Array of UINT32 | Array containing Ethernet-Like MIB counters for the port. See table below for array indices. |

## Interface Counters

Array indices of Interface Counters attribute (#5)

| Index | Name | Description |
|-------|------|-------------|
| 0 | In octets | Octets received on the interface |
| 1 | In Unicast Packets | Unicast packets received on the interface |
| 2 | In Non-Unicast Packets | Non-unicast packets (multicast/broadcast) packets received on the interface |
| 3 | In Discards | Inbound packets received on the interface but discarded |
| 4 | In Errors | Inbound packets that contain errors (does not include In Discards) |
| 5 | In Unknown Protos | Inbound packets with unknown protocol |
| 6 | Out Octets | Octets transmitted on the interface |
| 7 | Out Unicast packets | Unicast packets transmitted on the interface |
| 8 | Out Non-Unicast Packets | Non-unicast (multicast/broadcast) packets transmitted on the interface |
| 9 | Out Discards | Outbound packets discarded |
| 10 | Out Errors | Outbound packets that contain errors |

## Media Counters

Array indices of Media Counters attribute (#6)

| Index | Name | Description |
|-------|------|-------------|
| 0 | AlignmentErrors; | Frames received that are not an integral number of octets in length |
| 1 | FCSErrors; | Frames received that do not pass the FCS check |
| 2 | SingleCollisions; | Successfully transmitted frames which experienced exactly one collision |
| 3 | MultipleCollisions; | Successfully transmitted frames which experienced more than one collision |
| 4 | SQETestErrors; | Number of times SQE test error is generated |
| 5 | DeferredTransmissions; | Frames for which first transmission attempt is delayed because the medium is busy |
| 6 | LateCollisions; | Number of times collision is detected later than 512 bit-times into the transmission of a packet |
| 7 | ExcessiveCollisions; | Frames for which transmission fails due to excessive collisions |
| 8 | lMACTransmitErrors; | Frames for which transmission fails due to an internal MAC sublayer transmit error |
| 9 | lCarrieSenseErrors; | Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame |
| 10 | lFrameTooLong; | Frames received that exceed the maximum permitted frame size |
| 11 | lMACRecieveErrors; | Frames for which reception on an interface fails due to an internal MAC sublayer receive error |

## 8.8 CIP Port Configuration Object (0Dh)

### Category

Extended

### Object Description

This object is used to populate and enumerate the CIP Port Object (see *Port Object (F4h), p. 53*) on the network side. Basically, this is a matter of creating and updating instances and attributes which shall represent a CIP Port within the host application. This process is necessary in case support for Unconnected CIP Routing has been enabled (see *EtherNet/IP Host Object (F8h), p. 102*, Instance Attribute #17).

Each instance within this object corresponds to an instance in the CIP Port Object. The object supports up to 8 instances, where instance #1 is dedicated to the local TCP port, enabling the host application to implement up to 7 additional ports. Instance #1 will automatically be populated with default values, however it is possible for the host application to customize instance attributes #2 and #4.

Apart from attribute #7, it is possible to write to the instance attributes only during setup. The host application is responsible for keeping instance attribute #7 updated for all ports located within the host application.

> ! Note that the module does not take over the host application responsibility for error control; the module will not verify that the data set by the host application is correct.

### Supported Commands

| Object: | Get_Attribute |
| | Create |
| | Delete |
| Instance: | Get_Attribute |
| | Set_Attribute |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value |
|----|---------------------|--------|---------------|------------------------|
| 1 | Name | Get | Array of CHAR | "CIP Port Configuration" |
| 2 | Revision | Get | UINT8 | 01h |
| 3 | Number of instances | Get | UINT16 | - |
| 4 | Highest instance no. | Get | UINT16 | - |
| 11 | Max. no. of instances | Get | UINT16 | 0008h |

## Instance Attributes (Instance #1)

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Port Type | Set | UINT16 | Enumerates the port (See CIP specification, available from www.odva.org) |
| 2 | Port Number | Set | UINT16 | CIP port number associated with this port |
| 3 | Link Path | Set | Array of UINT8 | Logical path segments which identify the object for this port. |
| 4 | Port Name | Set | Array of CHAR | String (max. no. of characters is 64) which names the port. |
| 5 | - | - | - | (reserved) |
| 6 | - | - | - | (reserved) |
| 7 | Node Address | Set | Array of UINT8 | Node number of this device on port. The data type restricts the range to a Port Segment. The encoded port number must match the value specified in attribute #2.<br>A device which does not have a node number on the port can specify a zero length node address within the Port Segment (i.e. 10h 00h).<br>In case the node address changes during runtime, the host application is responsible for updating this attribute as well. |
| 8 | Port Node Range | Set | Struct of:<br>UINT16 (Min)<br>UINT16 (Max) | Minimum and maximum node number on port.<br>Support for this attribute is conditional; the attribute shall be supported provided that the node number can be reported within the range of the data type (e.g. DeviceNet). If not (as is the case with networks such as EtherNet/IP which uses a 4 byte IP address), the attribute shall not be supported. |

## 8.9        Functional Safety Module Object (11h)

**Category**

Extended

**Object Description**

This object contains information provided by the Safety Module connected to the Anybus CompactCom module. Please consult the manual for the Safety Module used, for values of the attributes below.

**Supported Commands**

| | |
|---|---|
| **Object:** | Get_Attribute |
| | Error_Confirmation |
| | Set_IO_Config_String |
| | Get_Safety_Output_PDU |
| | Get_Safety_Input_PDU |
| **Instance:** | Get_Attribute |

**Object Attributes (Instance #0)**

| # | Name | Access | Data Type | Value |
|---|---|---|---|---|
| 1 | Name | Get | Array of CHAR | "Functional Safety Module" |
| 2 | Revision | Get | UINT8 | 01h |
| 3 | Number of instances | Get | UINT16 | 0001h |
| 4 | Highest instance no. | Get | UINT16 | 0001h |

## Instance Attributes (Instance #1)

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | State | Get | UINT8 | Current state of the Safety Module<br>Please consult the manual for the Safety Module used. |
| 2 | Vendor ID | Get | UINT16 | Identifies vendor of the Safety Module.<br>E.g. 0001h (HMS Industrial Networks)<br>Please consult the manual for the Safety Module used. |
| 3 | IO Channel ID | Get | UINT16 | Describes the IO Channels that the Safety Module is equipped with.<br>Please consult the manual for the Safety Module used. |
| 4 | Firmware version | Get | Struct of<br>UINT8 (Major)<br>UINT8 (Minor)<br>UINT8 (Build) | Safety Module firmware version.<br>Format: version "2.18.3" would be represented as: first byte = 0x02, second byte = 0x12, third byte = 0x03. |
| 5 | Serial number | Get | UINT32 | 32 bit number, assigned to the Safety Module at production.<br>Please consult the manual for the Safety Module used. |
| 6 | Output data | Get | Array of UINT8 | Current value of the Safety Module output data, i.e. data FROM the network<br>Note: This data is unsafe, since it is provided by the Anybus CompactCom module. |
| 7 | Input data | Get | Array of UINT8 | Current value of the Safety Module input data, i.e. data sent TO the network.<br>**Note**: This data is unsafe, since it is provided by the Anybus CompactCom module. |
| 8 | Error counters | Get | Struct of<br>UINT16 (ABCC DR)<br>UINT16 (ABCC SE)<br>UINT16 (SM DR)<br>UINT16 (SM SE) | Error counters (each counter stops counting at FFFFh)<br><br>ABCC DR:   Responses (unexpected) from the Safety Module, discarded by the Anybus CompactCom module.<br><br>ABCC SE:   Serial reception errors detected by the Anybus CompactCom module.<br><br>SM DR:   Responses (unexpected) from the Anybus CompactCom module, discarded by the Safety Module.<br><br>SM SE:   Serial reception errors detected by the Safety Module. |
| 9 | Event log | Get | Array of UINT8 | Latest Safety Module event information (if any) is logged to this attribute. Any older event information is erased when a new event is logged.<br>For evaluation by HMS support. |
| 10 | Exception information | Get | UINT8 | If the Exception Code in the Anybus object is set to "Safety communication error" (09h), additional exception information is presented here, see table below. |
| 11 | Bootloader version | Get | Struct of<br>UINT8 Major<br>UINT8 Minor | Safety Module bootloader version.<br>Format: version "2.12" would be represented as: first byte = 0x02, second byte = 0x0C |
| 12 | Vendor block safe uc1 | Get | Array of UINT8 | The Safety Module may supply additional vendor-specific data to the Anybus CompactCom. If such data is available it is presented in this attribute. |
| 13 | Vendor block safe uc2 | Get | Array of UINT8 | The Safety Module may supply additional vendor-specific data to the Anybus CompactCom. If such data is available it is presented in this attribute. |

### Exception Information

If Exception Code 09h is set in the Anybus object, there is an error regarding the functional safety module in the application. Exception information is presented in instance attribute #10 according to this table:

| Value | Exception Information |
| --- | --- |
| 00h | No information |
| 01h | Baud rate not supported |
| 02h | No start message |
| 03h | Unexpected message length |
| 04h | Unexpected command in response |
| 05h | Unexpected error code |
| 06h | Safety application not found |
| 07h | Invalid safety application CRC |
| 08h | No flash access |
| 09h | Answer from wrong safety processor during boot loader communication |
| 0Ah | Boot loader timeout |
| 0Bh | Network specific parameter error |
| 0Ch | Invalid IO configuration string |
| 0Dh | Response differed between the safety microprocessors (e.g. different module types) |
| 0Eh | Incompatible module (e.g. supported network) |
| 0Fh | Max number of retransmissions performed (e.g. due to CRC errors) |
| 10h | Firmware file error |
| 11h | The cycle time value in attribute #4 in the Functional Safety Host Object can not be used with the current baud rate |
| 12h | Invalid SPDU input size in start-up telegram |
| 13h | Invalid SPDU output size in start-up telegram |
| 14h | Badly formatted input SPDU |
| 15h | Anybus to safety module initialization failure |

## Command Details: Error_Confirmation

**Category**

Extended

**Details**

| | |
|---|---|
| **Command Code** | 10h |
| **Valid for:** | Object |

**Description**

When the Safety Module has entered the Safe State, for any reason, it must receive an error confirmation before it can leave the Safe State. With this command it is possible to reset all safety channels of the safety which, for any reason, are in the Safe State at the same time. The application issues this command to the Anybus CompactCom module, when an error has been cleared by for example an operator. The Anybus CompactCom forwards the command to the Safety Module.

The channel Safe State can also be confirmed by the safety PLC or by the safety module.

With this command

- Command Details

  (no data)

- Response Details

  (no data)

## Command Details: Set_IO_Config_String

### Category

Extended

### Details

| | |
|---|---|
| **Command Code** | 11h |
| **Valid for:** | Object |

### Description

This command is sent from the host application when there is a need to change the default configuration of the safety inputs and outputs. This string is used by networks where there are no other means (e.g. PLC or some other tool) to provide the configuration to the safety module. Consult the specification of the safety module for more information. The byte string passed is generated by HMS and need to be passed unmodified using this command.

Information about this string is located in the specification of the safety module to which the string shall be sent.

- Command Details

| Field | Contents |
|---|---|
| CmdExt[0] | (not used) |
| CmdExt[1] | |
| Data[0... n] | Data (byte string) <br> The data consists of an IO configuration string, where the data format depends on the safety network. |

- Response Details

  (no data)

## Command Details: Get_Safety_Output_PDU

### Category

Extended

### Details

| Command Code | 12h |
|---|---|
| Valid for: | Object |

### Description

This command can be issued by the application to get the complete safety output PDU sent by the PLC. The Anybus CompactCom 40 EtherNet/IP will respond with the complete safety PDU, that the application then has to interpret.

- Command Details

  (no data)

- Response Details

| Field | Contents |
|---|---|
| CmdExt[0] | (not used) |
| CmdExt[1] | |
| Data[0... n] | Safety PDU from PLC |

## Command Details: Get_Safety_Input_PDU

### Category

Extended

### Details

| Command Code | 13h |
|---|---|
| Valid for: | Object |

### Description

This command can be issued by the application to get the complete safety input PDU sent by the safety module. The Anybus CompactCom 40 EtherNet/IP will respond with the complete safety PDU, that the application then has to interpret.

- Command Details

  (no data)

- Response Details

| Field | Contents |
|---|---|
| CmdExt[0] | (not used) |
| CmdExt[1] | |
| Data[0... n] | Safety PDU from safety module |

## Object Specific Error Codes

| Error Code | Description | Comments |
|---|---|---|
| 01h | The safety module rejected a message. | Error code sent by safety module is found in MsgData[2] and MsgData[3]. |
| 02h | Message response from the safety module has incorrect format (for example, wrong length). | - |

## 8.10        Time Object (13h)

### Category

Extended

### Object Description

In some networks there are multiple possible time sources. This object is used to present all known time sources using a common format. The quality of the different time sources may vary, which the host application has to consider when using the time value.

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value | Description |
|---|------|--------|-----------|-------|-------------|
| 1 | Name | Get | STRING | "Time Object" | Object name |
| 2 | Revision | Get | UINT8 | 1 | Revision of object |
| 3 | Number of instances | Get | UINT16 | N/A | Supported number of instances |
| 4 | Highest instance number | Get | UINT16 | N/A | Highest implemented instance |
| 11 | Protocols | Get | Array of: Struct of: UINT16 Instance ENUM Protocol UINT8 Reserved | N/A | Array of available time protocols. Instance: Corresponding instance number. Protocol: Enumeration of time protocols. See *Time Protocols, p. 88*. Reserved: Should not be used. |

### Instance Attributes (Instance #n)

Instance 1 is dedicated to CIP Sync.

> **!**  To make the instances available they have to be activated:
>
> CIP Sync is activated with attribute 32 in EtherNet/IP Host Object. See *EtherNet/IP Host Object (F8h), p. 102*.

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | Protocol | Get | ENUM | Enumeration identifier of the time protocol. See *Time Protocols, p. 88* for supported protocols. |
| 2 | Current time | Get | UINT64 | Current time in protocol specific format. If the time is not valid the value will be set to 0. See *Time Protocols, p. 88* for protocol formats. |

### Time Protocols

| Enum value | Priority | Protocol | Format | Epoch |
|------------|----------|----------|--------|-------|
| 0 | 0 | CIP Sync (IEEE 1588 PTP) | 64 bit nanoseconds | 23:59:51. 51.999918, December 31, 1969 |

# 9 Host Application Objects

## 9.1 General Information

This chapter specifies the host application object implementation in the module. The objects listed here may optionally be implemented within the host application firmware to expand the EtherNet/IP implementation.

Standard Objects:

- *Energy Control Object (F0h), p. 96*

- Assembly Mapping Object (EBh) - (see Anybus CompactCom 40 Software Design Guide)

- Modular Device Object (ECh) - (see Anybus CompactCom 40 Software Design Guide)

- *Sync Object (EEh), p. 95*

- *Energy Reporting Object (E7h), p. 90*

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

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

Network Specific Objects:

- *Functional Safety Object (E8h), p. 91*

- *CIP Identity Host Object (EDh), p. 93*

- *EtherNet/IP Host Object (F8h), p. 102*

- *Ethernet Host Object (F9h), p. 111*

## 9.2        Energy Reporting Object (E7h)

**Category**

Extended

**Object Description**

Using this object, the host application has a standardized way of reporting its energy consumed or produced. The reporting capabilities of this object are limited. On networks providing more elaborate reporting functionality, the reporting functionality will have to be implemented in a transparent manner by the application.

**Supported Commands**

| **Object:** | Get_Attribute |
|---|---|
| **Instance:** | Get_Attribute |

**Object Attributes (Instance #0)**

| # | Name | Access | Data Type | Value |
|---|---|---|---|---|
| 1 | Name | Get | Array of CHAR | "Energy Reporting" |
| 2 | Revision | Get | UINT8 | 01h |
| 3 | Number of instances | Get | UINT16 | 0001h |
| 4 | Highest instance no. | Get | UINT16 | 0001h |

**Instance Attributes (Instance #1)**

Extended

| # | Name | Access | Type | Description |
|---|---|---|---|---|
| 1 | Energy Reading | Get | Struct of:<br>UINT32<br>UINT32 | Amount of energy (Wh) consumed or produced by the application. Stored in nonvolatile memory.<br>The first UINT32 represents the lower part of the Energy Reading, the second UINT32 represents the higher part of the Energy Reading |
| 2 | Direction | Get | BOOL | Indicates if the host is consuming or producing energy.<br><br>Value:        Meaning:<br>0:              Producing<br>1:              Consuming |
| 3 | Accuracy | Get | UINT16 | Accuracy in 0.01% of reading<br>0: Unknown |
| 4 | Current Power Consumption | Get | UINT16 | The current power consumption in 0.01% of the Nominal Power consumption |
| 5 | Nominal Current Consumption | Get | UINT32 | The nominal power consumption in mW |

## 9.3        Functional Safety Object (E8h)

**Category**

Extended

**Object Description**

> **!** Do not implement this object if a safety module is not used.

This object specifies the safety settings of the application. It is mandatory if Functional Safety is to be supported and a Safety Module is connected to the Anybus CompactCom module.

**Supported Commands**

| Object: | Get_Attribute |
|---|---|
| **Instance:** | Get_Attribute |

**Object Attributes (Instance #0)**

| # | Name | Access | Data Type | Value |
|---|---|---|---|---|
| 1 | Name | Get | Array of CHAR | "Functional Safety" |
| 2 | Revision | Get | UINT8 | 01h |
| 3 | Number of instances | Get | UINT16 | 0001h |
| 4 | Highest instance no. | Get | UINT16 | 0001h |

**Instance Attributes (Instance #1)**

| # | Name | Access | Data Type | Default Value | Comment |
|---|---|---|---|---|---|
| 1 | Safety enabled | Get | BOOL | - | When TRUE, enables communication with the Safety Module. **Note**: If functional safety is not supported, this attribute must be set to FALSE. |
| 2 | Baud Rate | Get | UINT32 | 1020 kbit/s | This attribute sets the baud rate of the communication in bits/s between the Anybus CompactCom and the Safety Module. Valid values: <br>• 625 kbit/s <br>• 1000 kbit/s <br>• 1020 kbit/s (default) <br>Any other value set to this attribute, will cause the module to enter the EXCEPTION state. The attribute is optional. If not implemented, the default value will be used. **Note**: The host application shall never implement this attribute when using the IXXAT Safe T100. |
| 3 | (reserved) | | | | |

| # | Name | Access | Data Type | Default Value | Comment |
|---|------|--------|-----------|---------------|---------|
| 4 | Cycle Time | Get | UINT8 | - | Communication cycle time between the Anybus and the Safety module in milliseconds. **Note**: The host application shall never implement this attribute when using the IXXAT Safe T100. Valid values:<br><br>• 2 ms<br><br>• 4 ms<br><br>• 8 ms<br><br>• 16 ms<br><br>If another value is set in this attribute the Anybus will enter Exception state. Optional attribute; If not implemented the minimum cycle time for the chosen baud rate will be used:<br><br>• 2 ms for 1020 kbit/s<br><br>• 2 ms for 1000 kbit/s<br><br>• 4 ms for 625 kbit/s<br><br>The Anybus CompactCom validates the cycle time according to the minimum values above. If e.g. baud rate is 625 kbit/s and the cycle time is set to 2 ms the Anybus CompactCom will enter the EXCEPTION state. |
| 5 | FW upgrade in progress | Set | BOOL | False | Indicates if the Anybus CompactCom is upgrading the connected Safety module firmware. This means that the Anybus CompactCom will stay in the NW_INIT state longer than normal. |

## 9.4 CIP Identity Host Object (EDh)

### Category

Extended

### Object Description

This object allows for applications to support additional CIP identity instances. It is used to provide additional product identity information, e.g. concerning the software installed.

The first instance in the CIP identity object will not change its behavior. When implementing instances in the CIP identity host object, they will be mapped to the CIP identity object starting at instance 2. Instance no. 1 in the CIP identity host object will be mapped to instance no. 2 in the CIP identity object and so on.

See also ...

- *Identity Object (01h), p. 27* (CIP object)

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |
| | Get_Attribute_All |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value |
|---|---|---|---|---|
| 1 | Name | Get | Array of CHAR | "CIP Identity" |
| 2 | Revision | Get | UINT8 | 01h |
| 3 | Number of instances | Get | UINT16 | Depends on application |
| 4 | Highest instance no. | Get | UINT16 | Depends on application |

### Instance Attributes (Instance #1)

| # | Name | Access | Data Type | Description |
|---|---|---|---|---|
| 1 | Vendor ID | Get | UINT16 | These values replace the values for the CIP identity object instance #2 and upwards. See also... *Identity Object (01h), p. 27* (CIP-object) |
| 2 | Device Type | Get | UINT16 | |
| 3 | Product Code | Get | UINT16 | |
| 4 | Revision | Get | struct of: UINT8 Major UINT8 Minor | |
| 5 | Status | Get | UNIT16 | |
| 6 | Serial Number | Get | UINT32 | |
| 7 | Product Name | Get | Array of CHAR | |

## Command Details: Get_Attribute_All

### Category

Extended

### Details

| Command Code: | 10h |
|---|---|
| **Valid for:** | Object |

### Description

This service must be implemented by the application for all instances that exist in the CIP identity host object. If identity data is requested from the network the Anybus module will issue this command to the application. The application will then respond with a message containing a struct of all attributes in the requested instance.

- Command Details

    (no data)

- Response Details

| Field | Contents | Comments |
|---|---|---|
| MsgData[0, 1] | Vendor ID | ABCC CIP identity data |
| MsgData[2, 3] | Device type | |
| MsgData[4, 5] | Product code | |
| MsgData[6] | Major revision | |
| MsgData[7] | Minor revision | |
| MsgData[8,9] | Status | |
| MsgData[10 .. .13] | Serial number | |
| MsgData[14 .... n] | Product name | |

## 9.5          Sync Object (EEh)

### Category

Extended

### Object Description

This object is only used to store the cycle time for the last established IO connection that consumes data.

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |
| | Set_Attribute |

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

The attributes are represented on EtherNet/IP as follows:

| # | Name | Access | Data Type | Description |
|---|---|---|---|---|
| 1 | Cycle time | Get/Set | UINT32 | The RPI for the last established IO connection that consumes data (O→T RPI) |
| 2–8 | (not implemented) | | | |

## 9.6        Energy Control Object (F0h)

**Category**

Extended


**Object Description**

This object implements energy control functionality, i.e. energy specific settings, in the host application. The implementation of this object is optional. All instance attributes shall be seen as required and must be implemented in the application. If the Anybus module detects that an attribute is missing during run time an appropriate network error is sent and the Discard Responses counter is increased in the Anybus Object instance attribute Error Counter.

Each enabled instance in the object corresponds to an Energy saving mode. The number of available modes is device specific, and must be defined by the application. The higher the instance number, the more energy is saved. The instance with the highest number always corresponds to the "Power off" mode, i.e. the state where the device is essentially shut down. Instance 1 of the object represents "Ready to operate", i.e. the mode where the device is fully functional and does not save energy at all. Consequently a meaningful implementation always contains at least two instances, one for energy saving and one for operating. If this object is implemented for PROFINET, at least three instances are needed: "Ready to operate", "Energy saving mode 1", and "Power off".

Highest number of instances is 8. Please note that these modes are always present – they are not dynamically created or deleted. It is not allowed to leave holes in the list of instances.
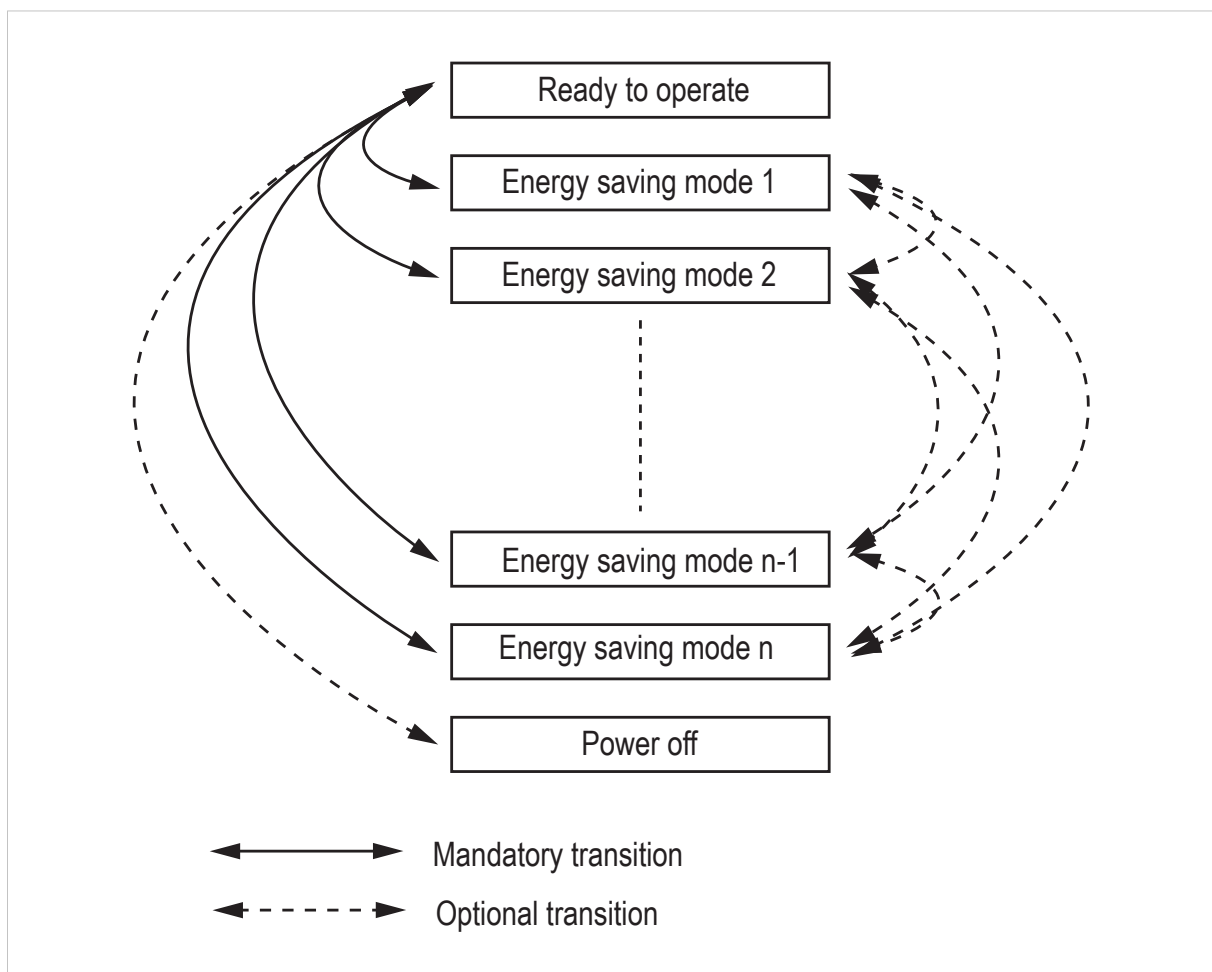


**Fig. 2**

### Supported Commands

| Object: | Get_Attribute |
|---|---|
| | StartPause |
| | EndPause |
| | Preview_Pause_Time (not PROFINET) |
| **Instance:** | Get_Attribute |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value |
|---|---|---|---|---|
| 1 | Name | Get | Array of CHAR | "Energy Control" |
| 2 | Revision | Get | UINT8 | 02h |
| 3 | Number of instances | Get | UINT16 | - |
| 4 | Highest instance no. | Get | UINT16 | Highest created instance number. Maximum value is 8. |
| 11 | Current Energy Saving Mode | Get | UINT16 | Instance number of the currently used Energy saving mode. During a mode transition the new Energy saving mode shall be presented. "Ready to operate" will equal instance #1, and "Power off" mode will equal Highest instance number. |
| 12 | Remaining time to destination | Get | UINT32 | When changing mode this parameter will reflect the actual time (in milliseconds) remaining until the shift is completed.<br>If a dynamic value cannot be generated the static value for the transition from the source to destination mode shall be used.<br>If the value is infinite, or unknown, the maximum value 0xFFFFFFFF shall be used. |
| 13 | Energy consumption to destination | Get | FLOAT | When changing mode this parameter will reflect the actual energy (in kWh) which will be consumed until the shift is completed.<br>If a dynamic value cannot be generated the static value for the transition from the source to destination mode shall be used.<br>If the value is undefined the value 0.0 shall be used. |
| 14 | Transition to "Power off" mode supported | Get | BOOL | Indicates whether transition to "Power off" mode is supported or not.<br>0:	Not supported<br>1:	Supported |

## Instance Attributes (Instance #1 - #8)

| # | Name | Access | Data Type | Description |
|---|------|--------|-----------|-------------|
| 1 | ModeAttributes | Get | BITS16 | Bit 0:        Meaning: <br><br> 0:      Only static time and energy values are available (Value of bit 0 attribute is not implemented) <br><br> 1:      Dynamic time and energy values are available <br><br> Bit 1-15:      Reserved |
| 2 | TimeMinPause | Get | UINT32 | Minimum pause time in milliseconds. ($t_{pause}$) <br> If the value is infinite, or unknown, the maximum value 0xFFFFFFFF shall be used. |
| 3 | TimeToPause | Get | UINT32 | Maximum time to go to this Energy saving mode.(ms, $t_{off}$) <br> If the value is infinite, or unknown, the maximum value 0xFFFFFFFF shall be used. |
| 4 | TimeToOperate | Get | UINT32 | Maximum time needed to go to the "Ready to operate" mode. (ms, $t_{on}$) <br> If the value is infinite, or unknown, the maximum value 0xFFFFFFFF shall be used. |
| 5 | TimeMinLengthOfStay | Get | UINT32 | The minimum time that the device must stay in this mode. In milliseconds.(ms, $t_{off\_min}$) <br> If the value is infinite, or unknown, the maximum value 0xFFFFFFFF shall be used. |
| 6 | TimeMaxLengthOfStay | Get | UINT32 | Maximum time that is allowed to stay in this mode. In milliseconds. <br> If no maximum value is available or if not implemented, the maximum value FFFFFFFFh shall be used. |
| 7 | ModePowerConsump-tion | Get | FLOAT | Amount of power consumed in this mode. (kW) <br> If the value is undefined the value 0.0 shall be used. |
| 8 | EnergyConsumptionTo-Pause | Get | FLOAT | Amount of energy required to go to this mode. (kWh) <br> If the value is undefined the value 0.0 shall be used. |
| 9 | EnergyConsumption-ToOperate | Get | FLOAT | Amount of energy required to go to the "Ready to operate" mode from this mode. (kWh) <br> If the value is undefined the value 0.0 shall be used. |
| 10 | Availability | Get | BOOL | Indicates if this energy saving mode is available given the current device state. <br> Not used for PROFINET. <br><br> False      Not available <br><br> True      Available (Value if attribute not implemented) |
| 11 | Power Consumption | Get | UINT32 | Indicates the power consumption of the device when in this state. <br> Not used for PROFINET. |

**Command Details: Start_Pause**

**Details**

| | |
|---|---|
| **Command Code** | 10h |
| **Valid for:** | Object |

**Description**

This command is sent to the host application when the system wants to initialize a pause of the system. The length of the pause is specified in milliseconds. The response of the message contains the destination mode (i.e. the instance number of the selected energy saving mode).

- Command Details

| Field | Contents | Comments |
|---|---|---|
| Data[0] | Pause time (low word, low byte) | Pause time (ms) |
| Data[1] | Pause time (low word, high byte) | |
| Data[2] | Pause time (high word, low byte) | |
| Data[3] | Pause time (high word, high byte) | |

- Response Details

| Field | Contents | Comments |
|---|---|---|
| Data[0] | Instance number (low byte) | Instance number of selected Energy mode |
| Data[1] | Instance number (low byte) | |

If the application is unable to select a state, given the requested pause time, it shall return one of the error codes in the table below.

| # | Error code | Description |
|---|---|---|
| 0x0D | Invalid state | Given the state of the device and the requested pause time it is currently not possible to enter any energy saving mode |
| 0x12 | Value too low | The requested pause time is too short |

**Command Details: End_Pause**

**Details**

| | |
|---|---|
| **Command Code** | 11h |
| **Valid for:** | Object |

**Description**

This command is sent to the host application when the system wants to return the system from a pause mode back to "Ready to operate" mode. In the response message the number of milliseconds to actualize the switch is returned.

- Command Details

  (none)

- Response Details

| Field | Contents | Comments |
|---|---|---|
| Data[0] | Time To Operate (low word, low byte) | Time needed to switch to "Ready to operate" |
| Data[1] | Time To Operate (low word, high byte) | |
| Data[2] | Time To Operate (high word, low byte) | |
| Data[3] | Time To Operate (high word, high byte) | |

If the application is unable to end the pause it shall return the error code in the table below.

| # | Error code | Description |
|---|---|---|
| 0x0D | Invalid state | Given the state of the device, it is currently not possible to end the pause |

**Command Details: Preview_Pause_Time**

**Details**

| | |
|---|---|
| **Command Code** | 12h |
| **Valid for:** | Object |

**Description**

Not used for PROFINET devices.

This command is sent to the host application when the system wants to preview the application's choice of Energy saving mode. The length of the pause is specified in milliseconds. The response shall contain the destination mode the application would have chosen if the StartPause service was sent (that is, the instance number of the selected energy saving mode). No transition to an Energy saving mode occurs.

- Command Details

| Field | Contents | Comments |
|---|---|---|
| Data[0] | Pause time (low word, low byte) | Pause time (ms) |
| Data[1] | Pause time (low word, high byte) | |
| Data[2] | Pause time (high word, low byte) | |
| Data[3] | Pause time (high word, high byte) | |

- Response Details

| Field | Contents | Comments |
|---|---|---|
| Data[0] | Instance number (low byte) | Instance number of selected Energy mode |
| Data[1] | Instance number (low byte) | |

If the application is unable to select a state, given the requested pause time, it shall return one of the error codes in the table below.

| # | Error code | Description |
|---|---|---|
| 0x0D | Invalid state | Given the state of the device and the requested pause time it is currently not possible to enter any energy saving mode |
| 0x12 | Value too low | The requested pause time is too short |

## 9.7 EtherNet/IP Host Object (F8h)

### Category

Basic, Extended

### Object Description

This object implements EtherNet/IP specific features in the host application. Note that this object must not be confused with the Ethernet Host Object, see *Ethernet Host Object (F9h), p. 111*.

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

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

Note that some of the commands used when accessing this object may require segmentation. For more information, see .

If the module is configured to use EIP QuickConnect functionality, the EDS file has to be changed. As the EDS file is changed, the identity of the module has to be changed and the module will require certification..

See also ...

- *Identity Object (01h), p. 27* (CIP object)

- *Assembly Object (04h), p. 31* (CIP object)

- *Port Object (F4h), p. 53* (CIP object)

- *CIP Port Configuration Object (0Dh), p. 79*

- Anybus CompactCom 40 Software Design Guide, "Error Codes"

### Supported Commands

| **Object:** | Get_Attribute |
| | Process_CIP_Object_Request |
| | Set_Configuration_Data |
| | Process_CIP_Routing_Request |
| | Get_Configuration_Data |
| **Instance:** | Get_Attribute |

### Object Attributes (Instance #0)

| # | Name | Access | Data Type | Value |
|---|------|--------|-----------|-------|
| 1 | Name | Get | Array of CHAR | "EtherNet/IP" |
| 2 | Revision | Get | UINT8 | 02h |
| 3 | Number of instances | Get | UINT16 | 0001h |
| 4 | Highest instance no. | Get | UINT16 | 0001h |

### Instance Attributes (Instance #1)

Basic

| # | Name | Access | Data Type | Default Value | Comment |
|---|------|--------|-----------|---------------|---------|
| 1 | Vendor ID | Get | UINT16 | 005Ah | These values are set in the Identity Object (CIP) at startup.<br>See also...<br>• *Network Identity, p. 11*<br>• *Identity Object (01h), p. 27*<br>Please note that changing any of these attributes requires a new Vendor ID. |
| 2 | Device Type | Get | UINT16 | 002Bh | |
| 3 | Product Code | Get | UINT16 | 0037h | |
| 4 | Revision | Get | struct of:<br>UINT8 Major<br>UINT8 Minor | (software revision) | |
| 5 | Serial Number | Get | UINT32 | (set at production) | |
| 6 | Product Name | Get | Array of CHAR | "Anybus CompactCom 40 EtherNet/IP(TM)" | |

## Extended

| # | Name | Access | Data Type | Default Value | Comment |
|---|------|--------|-----------|---------------|---------|
| 7 | Producing Instance No. | Get | Array of UINT16 | - | The values in this array are the EtherNet/IP Assembly instance numbers that matches the host application Assembly Mapping Object instances that are listed in attribute #11 (Write PD Instance List). If the Assembly Mapping Object is not implemented, one element in this array is allowed, to set the producing instance number. The maximum number of entries in the array is 6.<br>See "Multiple Assembly Instances" below for an example. |
| 8 | Consuming Instance No. | Get | Array of UINT16 | - | The values in this array are the EtherNet/IP Assembly instance numbers that matches the host application Assembly Mapping Object instances that are listed in attribute #12 (Read PD Instance List). If the Assembly Mapping Object is not implemented, one element in this array is allowed, to set the consuming instance number. The maximum number of entries in the array is 6.<br>See "Multiple Assembly Instances" below for an example. |
| 9 | Enable communication settings from net | Get | BOOL | True | Value   Meaning<br>True   Can be set from network<br>False   Cannot be set from network<br>See also ...<br>• *TCP/IP Interface Object (F5h), p. 55* (CIP-object)<br>• *Ethernet Link Object (F6h), p. 58* CIP-object)<br>• *Network Configuration Object (04h), p. 68*(Anybus Module Object) |
| 11 | Enable CIP forwarding | Get | BOOL | False | Value   Meaning<br>True   Requests to unknown CIP objects and unknown assembly object instances are routed to the application.<br>False   Requests to unknown CIP objects and unknown assembly object instances are not routed to the application.<br>See also.command details for Process _CIP_Object_Request below |
| 12 | Enable Parameter Object | Get | BOOL | True | Value   Meaning<br>True   Enable CIP Parameter Object<br>False   Disable CIP Parameter Object |
| 13 | Input-Only heartbeat instance number | Get | UINT16 | 0003h | See "Instance 03h Attributes (Heartbeat, Input-Only)" in *Assembly Object (04h), p. 31* (CIP-object). |
| 14 | Listen-Only heartbeat instance number | Get | UINT16 | 0004h | See "Instance 04h Attributes (Heartbeat, Listen-Only)" in *Assembly Object (04h), p. 31* (CIP-object). |
| 15 | Assembly object Configuration instance number | Get | UINT16 | 0005h | See "Instance 05h Attributes (Configuration Data)" in *Assembly Object (04h), p. 31* (CIP-object). |
| 16 | Disable Strict IO Match | Get | BOOL | False | If true, the module will accept Class1 connection requests that have sizes that's less than or equal to the configured IO sizes. |

| # | Name | Access | Data Type | Default Value | Comment |
|---|------|--------|-----------|---------------|---------|
| 17 | Enable unconnected routing | Get | BOOL | False | If true, the module enables unconnected CIP routing. This also triggers an initial upload of the contents of the CIP Port Mapping object. |
| 18 | Input-Only extended heartbeat instance number | Get | UINT16 | 0006h | See "Instance 06h Attributes (Heartbeat, Input-Only Extended)" in *Assembly Object (04h), p. 31* (CIP-object). |
| 19 | Listen-Only extended heartbeat instance number | Get | UINT16 | 0007h | See "Instance 06h Attributes (Heartbeat, Listen-Only Extended)" in *Assembly Object (04h), p. 31* (CIP-object). |
| 20 | Interface label port 1 | Get | Array of CHAR | Port 1 | The value of this attribute is used to change the interface label for Ethernet Link Object Instance #1 |
| 21 | Interface label port 2 | Get | Array of CHAR | Port 2 | The value of this attribute is used to change the interface label for Ethernet Link Object Instance #2 |
| 22 | Interface label internal port | Get | Array of CHAR | Internal | The value of this attribute is used to change the interface label for Ethernet Link Object Instance #3 |
| 23 - 25 | (reserved) | | | | |
| 26 | Enable EtherNet/IP QuickConnect | Get | BOOL | False | Value  Meaning<br><br>True  EtherNet/IP QuickConnect functionality enabled.<br><br>False  False  EtherNet/IP QuickConnect functionality disabled.<br><br>If the module is configured to use EIP QuickConnect functionality, the EDS file has to be changed. As the EDS file is changed, the identity of the module has to be changed and the module will require certification. |
| 27 - 28 | (reserved) | | | | |
| 29 | Ignore Sequence Count Check | Get | BOOL | False | Setting this attribute to "true" makes the module ignore the Sequence Count Check for consumed Class 1 data. This means that all data, not just changed/new data, received from the Originator, will be copied to the application. Copying all data and not just changed data is a violation of the CIP specification. It will also affect the performance of the module.<br>Use precaution when setting this flag to"true".<br>HMS Networks will do NO performance measurements and states NO guarantees about how performance will be affected when copying all data. |
| 30 | ABCC ADI Object Number | Get | UINT16 | 00A2h | This attribute either changes the object number of theADI Object (CIP object) or disables the ADI Object (CIP object). Valid object numbers are within the vendor specific ranges (0064h - 00C7h and 0300h - 04FFh). Any other value will disable the ADI object. |
| 31 | Enable DLR | Get | BOOL | True | Value  Meaning<br><br>True  DLR functionality enabled<br><br>False  DLR functionality disabled |
| 32 | Enable CIP Sync | Get | BOOL | False | Value  Meaning<br><br>True  CIP Sync functionality enabled<br><br>False  CIP Sync functionality disabled |

## Multiple Assembly Instances

The Assembly Mapping Object has two arrays on class level (Write PD Instance List and Read PD Instance List) listing instances defined by the application. The arrays of attributes 7 and 8 in the EtherNet/IP host object (Producing Instance Number and Consuming Instance number) are bound to the instance lists in the Assembly Mapping Object. The arrays list the corresponding CIP instance numbers representing each assembly instance defined by the application.

For more information, see

- *Using the Assembly Mapping Object (EBh), p. 16*

- Anybus CompactCom 40 Software Design Guide, "Assembly Mapping Object (EBh)"

## Command Details: Process_CIP_Object_Request

### Category

Extended

### Details

| | |
|---|---|
| **Command Code:** | 10h |
| **Valid for:** | Object |

### Description

By setting the 'Enable CIP Request Forwarding'-attribute (#11), all requests to unimplemented CIP-objects and unknown assembly object instances, will be forwarded to the host application through this command. The application then has to evaluate the request and return a proper response. The module supports one CIP-request; additional requests will be rejected by the module.

Note that since the telegram length on the host interface is limited, the request data size must not exceed 1524 bytes. If it does, the module will send a 'resource unavailable' response to the originator of the request and the message will not be forwarded to the host application.

> ⓘ *1524 bytes are available for class 3 connections using Large Forward Open. If UCMM is used, the total size (Message Router request/response) is limited to 504 bytes.*

> ⓘ *If the legacy message channel is used, message data is limited to 255 bytes.*

This command is similar - but <u>not</u> identical - to the 'Process_CIP_Request'-command in the Anybus CompactCom 40 DeviceNet.

- Command Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | CIP Service Code | CIP service code from original CIP request |
| CmdExt[1] | Request Path Size | Number of 16-bit words in the Request Path field |
| MsgData[0... m] | Request Path | CIP EPATH (Class, Instance, Attr. etc.) |
| MsgData[m... n] | Request Data | Service-specific data |

- Response Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | CIP Service Code | (Reply bit set) |
| CmdExt[1] | 00h | (reserved, set to zero) |
| MsgData[0] | General Status | CIP General Status Code |
| MsgData[1] | Size of Additional Status | Number of 16-bit words in Additional Status array |
| MsgData[2... m] | Additional Status | Additional Status, if applicable |
| MsgData[m... n] | Response data | Actual response data, if applicable |

> ❗ When using this functionality, make sure to implement the common CIP Class Attribute (attribute #1, Revision) for all objects in the host application firmware. Failure to observe this will prevent the module from successfully passing conformance tests.

## Command Details: Set_Configuration_Data

### Category

Extended

### Details

| Command Code: | 11h |
|---|---|
| Valid for: | Object |

### Description

If the data segment in the CIP "Forward_Open" service contains Configuration Data, this will be forwarded to the host application through this command. If implemented, the host application should evaluate the request and return a proper response. Segmentation is used, see "Message Segmentation" on page 189 for more information. The maximum total amount of configuration data that will be accepted by the module is 458 bytes.

This command must be implemented in order to support Configuration Data. If not implemented, "Forward_Open" requests will be rejected.

- Command Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | - | (reserved, ignore) |
| CmdExt[1] | Segmentation Control bits | See |
| MsgData[0 - 1] | Producing connection point | Producing connection point, requested by the originator. |
| MsgData[2 - 3] | Consuming connection point | Consuming connecition point, requested by the originator. |
| MsgData[4... n] | Data | Actual configuration data |

When the Set_Configuration_Data command is sent to the application as a result of a CIP Forward_open service containing Configuration Data, the producing connection point will be indicated by MsgData[0-1], and the consuming connection point by MsgData[2-3]. However, the Set_Configuration command may also come as a result of a CIP Set_Attribute_Single service to the CIP Assembly Object or a non matching NULL Forward Open service request. For both cases, MsgData[0-1] and MsgData[2-3] will contain 0 (zero).

- Response Details (Success)

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | (reserved, set to zero) |
| CmdExt[1] | 00h | (reserved, set to zero) |

- Response Details (Error)

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | (reserved, set to zero) |
| CmdExt[1] | 00h | (reserved, set to zero) |
| MsgData[0] | Error code | Anybus error code |
| MsgData[1] | Extended error code | If the Anybus error code is set to FFh, the extended error code shall be translated as shown in the table below. |
| MsgData[2... 3] | Index | If the Extended error code is set to 02h (invalid configuration), this parameter points to the attribute that failed. |

### Extended Error Code

If the Error code equals FFh (Object specific error), the extended code will be translated as below:

| Code | Contents | CIP no. | CIP status code | Additional Information |
|------|----------|---------|-----------------|------------------------|
| 01h | Ownership conflict | 01h | Connection failure | The configuration data was supplied in a forward open request. |
| | | 10h | Device State conflict | The configuration data was supplied in a set request to the Assembly object. |
| 02h | Invalid configuration | 09h | Bad attribute data | CIP extended error code: Use value from MsgData[2 - 3]. The extended error code shall only be used if the request originated from a Forward Open request, not for explicit set requests. |

- *Connection Manager (06h), p. 34* (CIP object)

- Message segmentation

## Command Details: Process_CIP_Routing_Request

### Category

Extended

### Details

| | |
|---|---|
| **Command Code:** | 12h |
| **Valid for:** | Object |

### Description

The module will strip the first path within the "Unconnected_Send" service and evaluate whether or not it's possible to continue with the routing (e.g. check that the requested port exists within the port object). If the stripped path was the last path the contents delivered to the application will be the CIP request sent to the destination node, otherwise it will be an "Unconnected_Send" service with updated route path information.

The module supports one pending request. Additional requests will be rejected by the module.

Please note that since the telegram length on the host interface is limited, the data must not exceed 1524 bytes in length. If it does, the module will reject the originator of the request ("Resource unavailable"), and this command will not be issued towards the host application.

> **ⓘ** *If the legacy message channel is used, message data is limited to 255 bytes.*

- Command Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | - | (reserved, ignore) |
| CmdExt[1] | - | (reserved, ignore) |
| MsgData[0... n] | Destination Path | Destination path encoded as an EPATH. |
| MsgData[n+1] | Time_tick | Valid after timeout parameters have been updated |
| MsgData[n+2] | Time-out_ticks | Valid after timeout parameters have been updated |
| MsgData[n+3... m] | CIP message | CIP message to route |

- Response Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | (reserved, set to zero) |
| CmdExt[1] | 00h | (reserved, set to zero) |
| MsgData[0] | CIP Service | Actual CIP service code, response bit set |
| MsgData[1] | 00h | (reserved, set to zero) |
| MsgData[2] | General Status | Actual CIP General status code |
| MsgData[3] | Size of Additional Status | No. of 16-bit words in Additional Status Array |
| MsgData[4... n] | Additional Status Array | Additional status, if applicable |
| MsgData[n+1... m] | Response Data | Actual response data |

See also..

**Command Details: Get_Configuration_Data**

**Category**

Extended

**Details**

| Command Code: | 13h |
|---|---|
| Valid for: | Object |

**Description**

If the configuration data is requested from the network, the Anybus will issue this command to the application. The application shall send the stored configuration data in the response message.

Segmentation is used since the telegram length on the host interface is limited. The maximum total amount of configuration data that will be accepted by the module is 458 bytes.

This command must be implemented in order to support Configuration Data. If not implemented, the request will be rejected by the Anybus module.

- Command Details

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | - |
| CmdExt[1] | 00h | - |
| MsgData[0... n] | - | No extended message data |

- Response Details (Success)

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | (reserved, set to zero) |
| CmdExt[1] | Segmentation Control bits | See |
| MsgData[0 - n] | Status | Configuration data from the application |

- Response Details (Error)

| Field | Contents | Comments |
|---|---|---|
| CmdExt[0] | 00h | (reserved, set to zero) |
| CmdExt[1] | Segmentation Control bits | See |
| MsgData[0] | Status | Anybus protocol error code |

## 9.8        Ethernet Host Object (F9h)

### Object Description

This object implements Ethernet features in the host application.

### Supported Commands

| | |
|---|---|
| **Object:** | Get_Attribute |
| **Instance:** | Get_Attribute |
| | Set_Attribute |

### Object Attributes (Instance #0)

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

### Instance Attributes (Instance #1)

- If an attribute is not implemented, the default value will be used.

- The module is preprogrammed with a valid MAC address. To use that address, do not implement attribute #1.

- Do not implement attributes #9 and #10, only used for PROFINET devices, if the module shall use the preprogrammed MAC addresses.

- If new MAC addresses are assigned to a PROFINET device, these addresses (in attributes #1, #9, and #10) have to be consecutive, e.g. (xx:yy:zz:aa:bb:01), (xx:yy:zz:aa:bb:02), and (xx:yy:zz:aa:bb:03).

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

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

| # | Name | Access | Data Type | Default Value | Comment |
|---|------|--------|-----------|---------------|---------|
| 17 | IP address byte 0–2 | Get | Array of UINT8 [3] | [0] = 192 [1] = 168 [2] = 0 | First three bytes in IP address. Used in standalone shift register mode if the configuration switch value is set to 1-245. In that case the IP address will be set to: Y[0].Y[1].Y[2].X Where Y0-2 is configured by this attribute and the last byte X by the configuration switch. |
| 18 | Ethernet PHY Configuration | Get | Array of BITS16 | 0x0000 for each port | Ethernet PHY configuration bit field. The length of the array shall equal the number of Ethernet ports of the product. Each element represents the configuration of one Ethernet port (element #0 maps to Ethernet port #1, element #1 maps to Ethernet port #2 and so on). **Note**: Only valid for EtherNet/IP and Modbus-TCP devices. Bit 0:        Auto negotiation fallback duplex 0 = Half duplex 1 = Full duplex Bit 1–15:        Reserved |
| 20 | SNMP read-only community string | Get | Array of CHAR | "public" | **Note**: This attribute is only valid for PROFINET devices. Sets the SNMP read-only community string. Max length is 32. |
| 21 | SNMP read-write community string | Get | Array of CHAR | "private" | **Note**: This attribute is only valid for PROFINET devices. Sets the SNMP read-write community string. Max length is 32. |
| 22 | DHCP Option 61 source | Get | ENUM | 0 (Disabled) | **Note**: This attribute is currently only valid for Ethernet/IP devices. See below (DHCP Option 61, Client Identifier) |
| 23 | DHCP Option 61 generic string | Get | Array of UINT8 | N/A | **Note**: This attribute is currently only valid for Ethernet/IP devices. See below (DHCP Option 61, Client Identifier) |
| 24 | Enable DHCP Client | Get | BOOL | 1 = True | **Note**: This attribute is currently valid for Ethernet/IP and PROFINET devices. Enable/disable DHCP Client functionality 0:        DHCP Client functionality is disabled 1:        DHCP Client functionality is enabled |
| 25 | Enable WebDAV Server | Get | BOOL | 1 = True | **Note**: This attribute is currently valid for devices with IIoT Secure functionality. Enable/disable WebDAV server 0:        WebDAV functionality is disabled 1:        WebDAV functionality is enabled |

## Network Status

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

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

| Bit | Contents | Description | Comment |
|---|---|---|---|
| 4 | Link port 2 | Current link status for port 2<br>1 = Link sensed<br>0 = No link | Not used for EtherCAT |
| 5... 15 | (reserved) | (mask off and ignore) | |

## DHCP Option 61 (Client Identifier)

ⓘ   *Only valid for EtherNet/IP devices*

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

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

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

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

Example:

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

Example 2:

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

This page intentionally left blank

# A        Categorization of Functionality

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

## A.1        Basic

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

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

## A.2        Extended

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

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

# B        Implementation Details

## B.1        SUP-Bit Definition

The supervised bit (SUP) indicates that the network participation is supervised by another network device. In the case of EtherNet/IP, this means that the SUP-bit is set when one or more CIP (Class 1 or Class 3) connections has been opened towards the module.

## B.2        Anybus State Machine

The table below describes how the Anybus Statemachine relates to the EtherNet/IP network

| Anybus State | Implementation | Comment |
|---|---|---|
| WAIT_PROCESS | The module stays in this state until a Class 1 connection has been opened. | - |
| ERROR | Class 1 connections errors<br>Duplicate IP address detected | - |
| PROCESS_ACTIVE | Error free Class 1 connection active (RUN-bit set in the 32-bit Run/Idle header of an Exclusive-Owner connection) | Only valid for consuming connections. |
| IDLE | Class 1 connection idle. | |
| EXCEPTION | Unexpected error, e.g. watchdog timeout etc. | MS LED turns red (to indicate a major fault)<br>NS LED is off |

## B.3        Application Watchdog Timeout Handling

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

# C  Secure HICP (Secure Host IP Configuration Protocol)

## C.1  General

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

## C.2  Operation

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

To alter the network settings of a module, click on its entry in the list. The settings for the module will appear to the right.
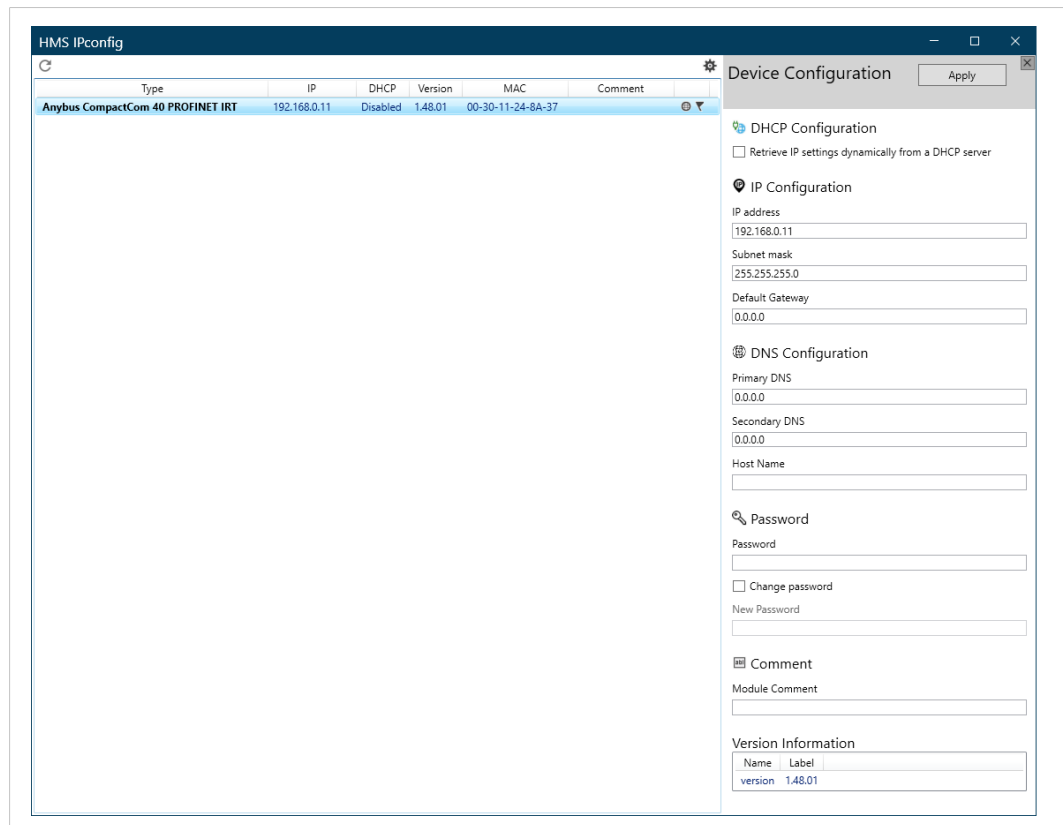


**Fig. 3**

Click **Apply** to send and apply the settings. Settings are saved in non-volatile memory in the device. Optionally, the configuration can be protected from unauthorized access by a password.

# D      Technical Specification

## D.1      Front View

### D.1.1      Front View (Ethernet Connectors)

| # | Item | Connector |
|---|------|-----------|
| 1 | Network Status LED | Ethernet, RJ45 |
| 2 | Module Status LED | |
| 3 | Link/Activity LED (port 1) | |
| 4 | Link/Activity LED (port 2) | |



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

### D.1.2      Network Status LED

| LED State | Description |
|-----------|-------------|
| Off | No power or no IP address |
| Green | Online, one or more connections established (CIP Class 1 or 3) |
| Green, flashing | Online, no connections established |
| Red | Duplicate IP address, FATAL error |
| Red, flashing | One or more connections timed out (CIP Class 1 or 3) |

### D.1.3      Module Status LED

| LED State | Description |
|-----------|-------------|
| Off | No power |
| Green | Controlled by a Scanner in Run state and, if CIP Sync is enabled, time is synchronized to a Grandmaster clock |
| Green, flashing | Not configured, Scanner in Idle state, or, if CIP Sync is enabled, time is synchronized Grandmaster clock |
| Red | Major fault (EXCEPTION-state, FATAL error etc.) |
| Red, flashing | Recoverable fault(s). Module is configured, but stored parameters differ from currently used parameters. |
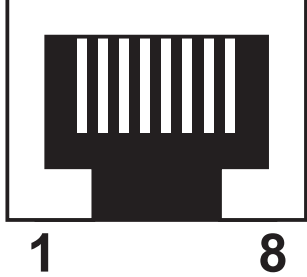
### D.1.4      LINK/Activity LED 3/4

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

### D.1.5      Ethernet Interface

**Ethernet Interface (RJ45 connectors)**

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

| Pin no | Description |
|--------|-------------|
| 4,5,7,8 | Connected to chassis ground over serial RC circuit |
| 6 | RD- |
| 3 | RD+ |
| 2 | TD- |
| 1 | TD+ |
| Housing | Cable Shield |

## D.2 Functional Earth (FE) Requirements

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

## D.3 Power Supply

### D.3.1 Supply Voltage

The Anybus CompactCom 40 EtherNet/IP requires a regulated 3.3 V power source as specified in the general *Anybus CompactCom 40 Hardware Design Guide*.

### D.3.2 Power Consumption

TheAnybus CompactCom 40 EtherNet/IP is designed to fulfil the requirements of a Class B module. The current hardware design consumes up to 360 mA

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

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

## D.4 Environmental Specification

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

## D.5 EMC Compliance

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

# E        Timing & Performance

## E.1        General Information

This chapter specifies timing and performance parameters that are verified and documented for the Anybus CompactCom 40 EtherNet/IP.

| Category | Parameters | Page |
|---|---|---|
| Startup Delay | T1, T2 | *122* |
| NW_INIT Handling | T100 | *122* |
| Event Based WrMsg Busy Time | T103 | *122* |
| Event Based Process Data Delay | T101, T102 | *123* |

For further information, please consult the Anybus CompactCom 40 Software Design Guide.

## E.2        Internal Timing

### E.2.1        Startup Delay

The following parameters are defined as the time measured from the point where /RESET is released to the point where the specified event occurs.

| Parameter | Description | Max. | Unit. |
|---|---|---|---|
| T1 | The Anybus CompactCom 40 EtherNet/IP module generates the first application interrupt (parallel mode) | 64 | ms |
| T2 | The Anybus CompactCom 40 EtherNet/IP module is able to receive and handle the first application telegram (serial mode) | 64 | ms |

### E.2.2        NW_INIT Handling

This test measures the time required by the Anybus CompactCom 40 EtherNet/IP module to perform the necessary actions in the NW_INIT-state.

| Parameter | Conditions |
|---|---|
| No. of network specific commands | Max. |
| No. of ADIs (single UINT8) mapped to Process Data in each direction. (If the network specific maximum is less than the value given here, the network specific value will be used.) | 32 |
| Event based application message response time | > 1 ms |
| Ping-pong application response time | > 10 ms |
| No. of simultaneously outstanding Anybus commands that the application can handle | 1 |

| Parameter | Description | Communication | Max. | Unit. |
|---|---|---|---|---|
| T100 | NW_INIT handling | Event based modes | 58 | ms |

### E.2.3        Event Based WrMsg Busy Time

The Event based WrMsg busy time is defined as the time it takes for the module to return the H_WRMSG area to the application after the application has posted a message.

| Parameter | Description | Min. | Max. | Unit. |
|---|---|---|---|---|
| T103 | H_WRMSG area busy time | 6 | 9 | µs |

### E.2.4 Event Based Process Data Delay

"Read process data delay" is defined as the time from when the last bit of the network frame has been received by the network interface, to when the RDPDI interrupt is asserted to the application.

"Write process data delay" is defined as the time from when the application exchanges write process data buffers, to when the first bit of the new process data frame is sent out on the network.

The tests were run in 16-bit parallel event mode, with interrupts triggered only for new process data events. Eight different IO sizes (2, 16, 32, 64, 128, 256, 512 and 1024 bytes) were used in the tests, all giving the same test results.

The delay added by the PHY circuit has not been included, as this delay is insignificant compared to the total process data delay.

| Parameter | Description | Delay (min.) | Delay (typ.) | Delay (max.) | Unit |
|-----------|-------------|--------------|--------------|--------------|------|
| T101 | Read process data delay Measured at an IO size of 32 bytes | - | - | 84 | µs |
| T102 | Write process data delay Measured at an IO size of 32 bytes | - | - | 106 | µs |
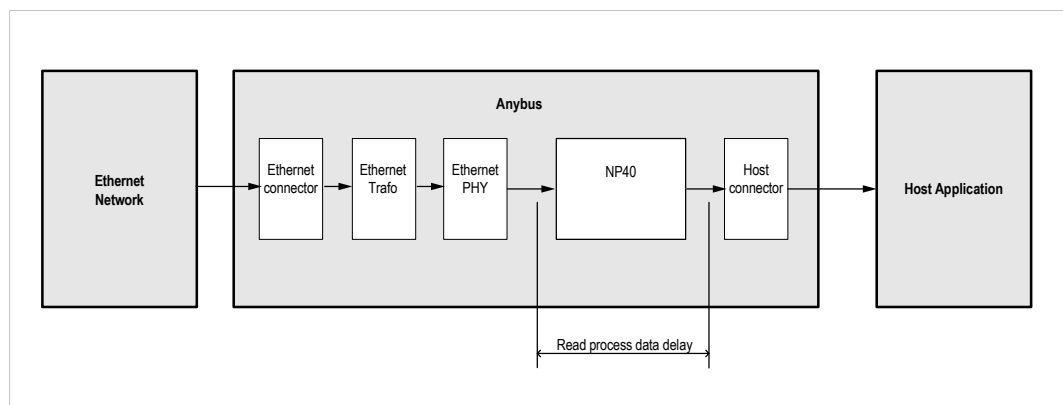


**Fig. 4**
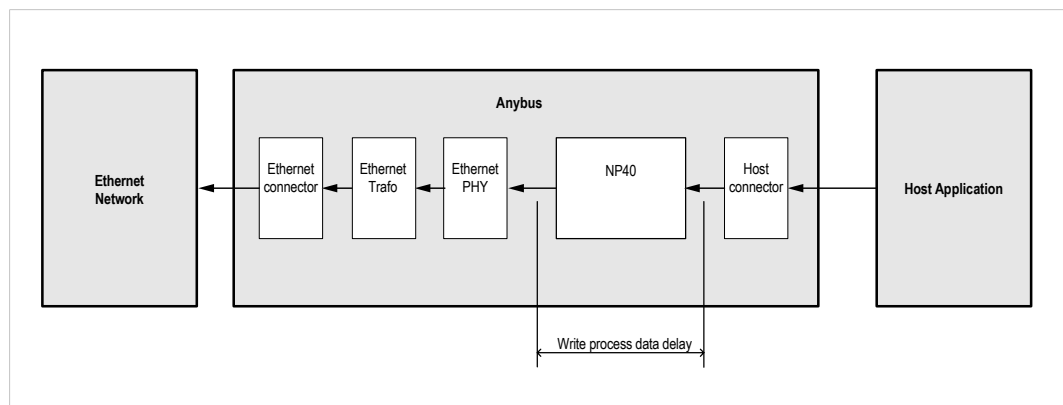


**Fig. 5**

# F Conformance Test Guide

## F.1 General

When using the default settings of all parameters, the Anybus CompactCom 40 EtherNet/IP is precertified for network compliance. This precertification is done to ensure that the end product *can* be certified.

To be allowed to use EtherNet/IP in a product the vendor is required to be a licensed EtherNet/IP vendor, with a vendor ID of its own. Please contact www.odva.org to obtain a vendor ID.

Changes in the parameters in the example EDS file, supplied by HMS Networks, will require a certification. This chapter provides a guide for successfully conformance testing your product, containing the Anybus CompactCom 40 EtherNet/IP, to comply with the demands for network certification set by the ODVA.

The actions described in this appendix have to be accounted for in the certification process, e.g. the identity of the product needs to be changed to match your company and device.

> ❗ This appendix provides guidelines and examples of what is needed for conformance testing and certification. Depending on the functionality of your application, there may be additional steps to take.

All screenshots within this document are taken from the ODVA Conformance Test Software Tool for EtherNet/IP CT14, © ODVA Inc. This software is available for order through the ODVA website. It is required to perform pre-testing with this software prior to submitting the product for conformance testing.

Also, a Statement of Conformance file (STC file), describing the EtherNet/IP application, has to be prepared prior to submitting the product for conformance testing.

## F.2 Suggested Test Tools

### F.2.1 Wireshark

This free, open source tool is the de facto standard for network capture and analysis. It is heavily used by ODVA TSPs, HMS Networks, and the greater EtherNet/IP user base. Wireshark ( www.wireshark.org) captures Ethernet traffic using your computers network interface card, and displays the contents in an intuitive fashion that allows for detailed analysis of the packets. Developers from HMS Networks have contributed to the EtherNet/IP dissectors (the analysis engine), and it is possible for users to create their own dissectors for their application data. The use of Wireshark is well documented, but there are a few good tips for EtherNet/IP testing that will help users get to the crucial information.

- Use viewing filters "CIP" to see only EtherNet/IP traffic.

- It is possible to filter by the HMS MAC ID. This will only show Ethernet messages with HMS devices as the source or destination "eth.addr[0:3] == 00:30:11".

- There are many other useful filters available on the Wireshark webpage.

### F.2.2 NMAP

NMAP is a free, open source tool for network discovery and security testing. NMAP will discover which TCP and UDP ports are open or responding. It will also determine which layer 3 services are supported by your device. ODVA has strict guidelines for open ports, and mandatory layer 3 services. For the NMAP procedure used by TSPs please see the Sample Test Report that comes with Conformance Test Software from ODVA.

### F.2.3 ODVA Conformance Test Software

This automated test software is designed to query, provoke, and detect software flaws in your device. ODVA sells yearly subscriptions of this software to vendors so that they can prepare for conformance testing. This software is also the best way to modify or create the Statement of Conformance (STC) file. Pressing CTRL+D will bring up a GUI for the Data section of the STC file.

**Getting Started**

After completing the install, a webpage is brought up in the default browser. This page gives an overview of the test software and lists the relevant documentation with a brief summary. The setup for testing is covered in the Conformance Test Software User Manual.

| Chapter | Contents |
| --- | --- |
| 1 | System requirements and installation |
| 2 | How to select a device and how to modify the Statement of Conformance file |
| 3 | How to set up the network to prepare for testing |
| 4 | How to run the test software |

**The User Manual - Critical Points**

Users are strongly encouraged to read through the Conformance Test Software User Manual to fully understand the testing software. The following points are meant to recapture the critical sections of this document.

• The Network Interface that will be used for testing needs to be selected from the available network interface cards in the Setup menu.
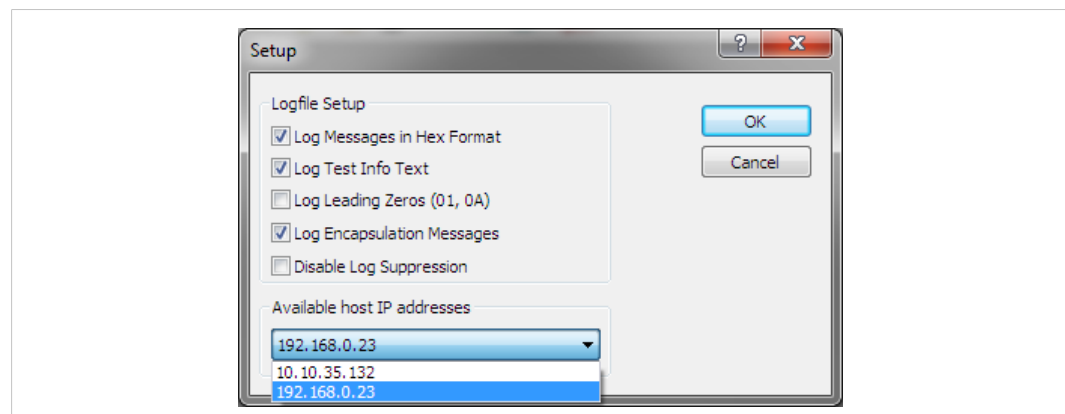


**Fig. 6**

• Most devices will comply with the default timeout settings, but some require more relaxed standards for responses. This can be set in the Set Message Wait Timers menu.

• The latest version of the CT Test software requires users to allocate a second IP address for their network interface card.

• Enabling the Encapsulation Logging feature of the CT test will allow users to efficiently work with Wireshark captures and Conformance Test logs.

### F.2.4 EZ-EDS

EZ-EDS is a free utility made available by ODVA. This tool is very helpful for editing and testing Electronic Data Sheets. Electronic Data Sheets are ASCII formatted files that describe data organization, configuration, and performance capabilities. They are commonly called EDS files, and have the extension .eds. EDS files can be built and modified using a text editor, but EZ-EDS provides a graphical user interface that brings attention to major fields. EZ-EDS also tests EDS

files for correct formatting. Much of the possible content of EDS files is optional, and ODVA tests stress correct formatting and not content.

### F.2.5        Anybus EDS Generator

The Anybus EDS Generator tool automatically generates an EDS file by scanning a device using the Anybus CompactCom 40 EtherNet/IP. This tool is easy to use and will provide a correctly configured EDS file that matches your product. It is still required to validate the EDS file via EZ-EDS.

The tool is available from the HMS Networks web site.

### F.2.6        Sample Test Reports

The subscription to the conformance test software includes the EtherNet/IP Sample Test Report document. This document outlines the manual procedure that testers will perform in addition to running the automated test software.

**Manual Test Procedure**

Some features of EtherNet/IP cannot be properly verified by automated test software or the development of a fully automated test may be impractical. For these features, a manual test procedure, as well as passing criteria, is listed in the Sample Test Report. This is the exact procedure and criteria used by the Test Service Providers.

The majority of functionality that needs to be tested manually is provided by the systems of the Anybus CompactCom, and has no interaction with the host application. Therefore, developers using the Anybus CompactCom may omit this lengthy procedure, but they must check the following:

- HMS recommends everyone to complete the Physical layer and EDS test sections of the sample test report. This ensures that produce labeling of LED's is correct and that the EDS file is verified prior to submitting the product to the TSP for conformance test

- If DLR is enabled in the product, it is required to be able to configure the speed and the duplex of all Ethernet ports in some way. The host application may elect to disable the standard means of configuring the speed and duplex in the Anybus CompactCom by:

  – Disabling set access to the Ethernet Link object by setting instance attribute #9 (Enable Communication Settings from NET) in the Host EtherNet/IP object to False.

  – Disabling the web server. On the standard web pages of the Anybus CompactCom it is possible to configure speed and duplex of the ethernet ports. For applications using transparent ethernet functionality the web server is always disabled.

  If none of the these ways of configuring the speed and duplex is possible, the host application must provide some other way to configure them. For example the application can have a keypad interface which can be used for configuration.

- If the host application includes hardware switches (for example DIP switches or rotary switches) for configuring the IP address or has disabled either ACD or DLR, HMS Networks recommends to perform the manual test cases in TCP/IP Interface Object Tests (section 4), Ethernet Link Object Tests (section 5), and Address Conflict Detection (ACD) Tests (section 10) in the sample test report.

## F.3        Statement of Conformance (STC)

> **!**  This document is not a comprehensive guide. Following the steps below will not absolutely guarantee that a device will complete conformance testing.

The goal of this section is to explain the relation the Anybus Objects to the Conformance Test and the Statement of Conformance (STC). The objects listed below exist in the host application, the Anybus CompactCom, and not in the EtherNet/IP interface. The objects are described in the Anybus CompactCom 40 Software Design Guide and in the Anybus CompactCom 40 EtherNet/IP Network Guide.

It is recommended to read the CIP Protocol Test Specification and the EtherNet/IP Test Specification prior to testing. In these documents the expected response and/or the acceptable behavior are stated, which is useful to be able to avoid a lot of initial errors. Modifications can be made to the Statement of Conformance and to the host application at an early stage, reducing time and effort.

## F.3.1        Implementation of Host Objects

The implementations of the host objects may have to be adapted, to make sure that the end product will pass a conformance test. Using the CT Software, follow the instructions below. Only the host objects relevant to EtherNet/IP will be discussed.

**EtherNet/IP Host Object (F8h)**

The implementation of the EtherNet/IP Host Object (F8h) has impact on the following objects: Identity Object (01h, CIP object), Assembly Object (04h, CIP object), Port Object (F4h, CIP object), and CIP Port Configuration Object (host object, 0Dh). It also has impact on how the STC is configured. The instance attributes, listed below, need to be considered.

**EtherNet/IP Host Object (F8h) - Attribute #1 - Vendor ID**

The Vendor ID must match the Vendor name in the CT software and the STC.
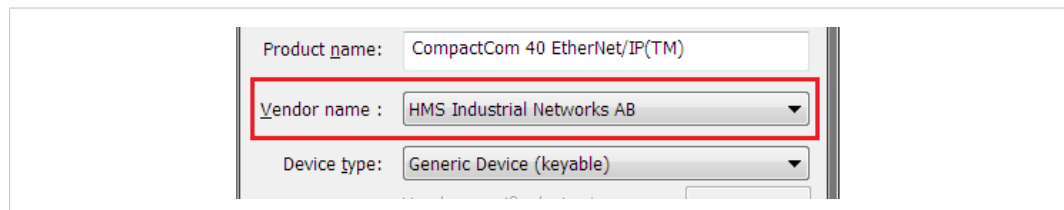


**Fig. 7**

First time EtherNet/IP vendors may not find their name available from the drop down menu, as it's not certain that the test software has been updated. It is possible to pre-test with any Vendor ID in the list, to reduce the number of errors reported due to Vendor ID mismatch, as long as the Vendor ID is changed in both the device and in the STC before actual conformance testing.

Alternatively, vendors can add the vendor information to the VID.dat file.

**EtherNet/IP Host Object (F8h) - Attribute #2 - Device Type**

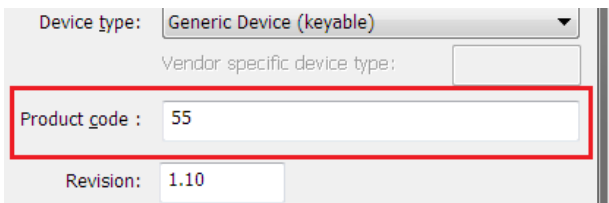The Device Type must match the Device Type in the drop down list:



**Fig. 8**

**EtherNet/IP Host Object (F8h) - Attribute #3 - Product Code**

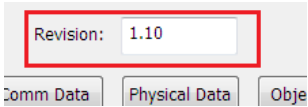The Product Code must match the Product Code in the drop down list:



**Fig. 9**

**EtherNet/IP Host Object (F8h) - Attribute #4 - Revision**

The Revision must match the revision field <major>. <minor>.



**Fig. 10**

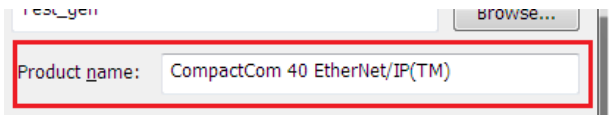**EtherNet/IP Host Object (F8h) - Attribute #5 - Serial Number**

The current version of CT test does not check serial number.

> *According to the CIP specification, the combination of Vendor ID and serial number must be unique. It is not permitted to use a custom serial number in combination with the HMS Vendor ID (005Ah).*

**EtherNet/IP Host Object (F8h) - Attribute #6 - Product Name**

The Product Name must match the Product Name field.



**Fig. 11**

### EtherNet/IP Host Object (F8h) - Attribute #7 - Producing Instance No.

The Producing Instance(s) will impact the assembly object, and will need to be listed. For most applications the producing instance(s) will be Static Inputs.
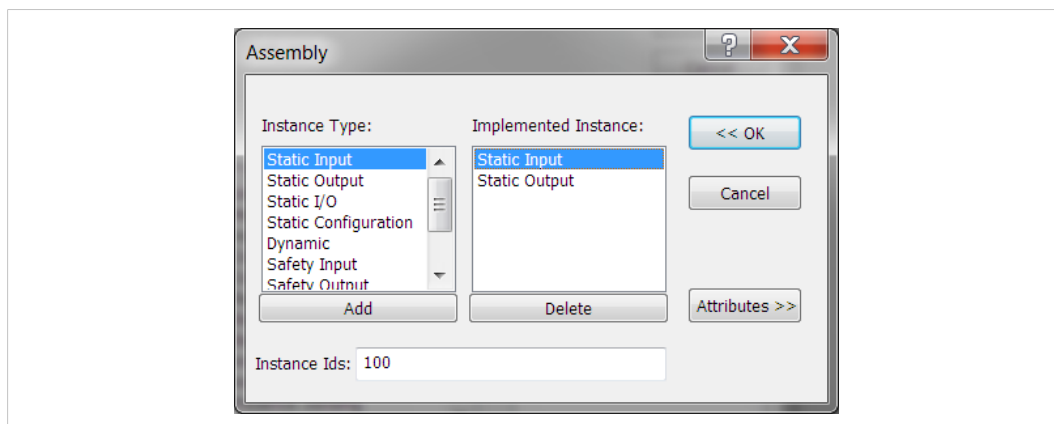


**Fig. 12**

Producing Instances will also impact the Connections of the Connection Manager object. Each of the connections must have the connection path modified to point to the correct instance(s). The example below lists 0x64 as the producing instance. See *Volume 1: Common Industrial Protocol Specification appendix C* for the encoding of the Connection Path.
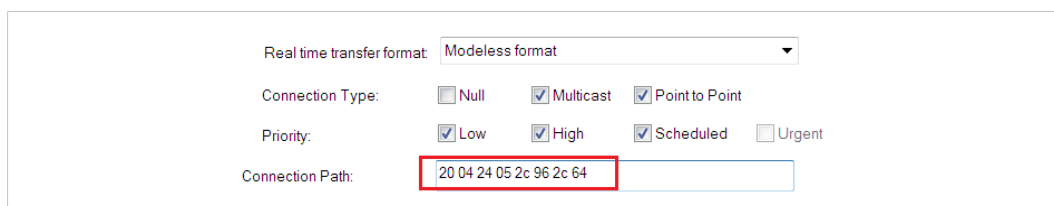


**Fig. 13**

> (i) *This attribute is an array if the host application implements the Assembly Mapping Host object, see details of this object below.*

### EtherNet/IP Host Object (F8h) - Attribute #8 - Consuming Instance No.

The response field will impact the assembly object. For most applications the producing instance (s) will be Static Outputs.



**Fig. 14**

Producing Instances will also impact the Connections of the Connection Manager object. Each of the connections must have the connection path modified to point to the correct instance(s). The

example below lists 0x96 as the consuming instance. See *Volume 1: Common Industrial Protocol Specification appendix C* for the encoding of the Connection Path.
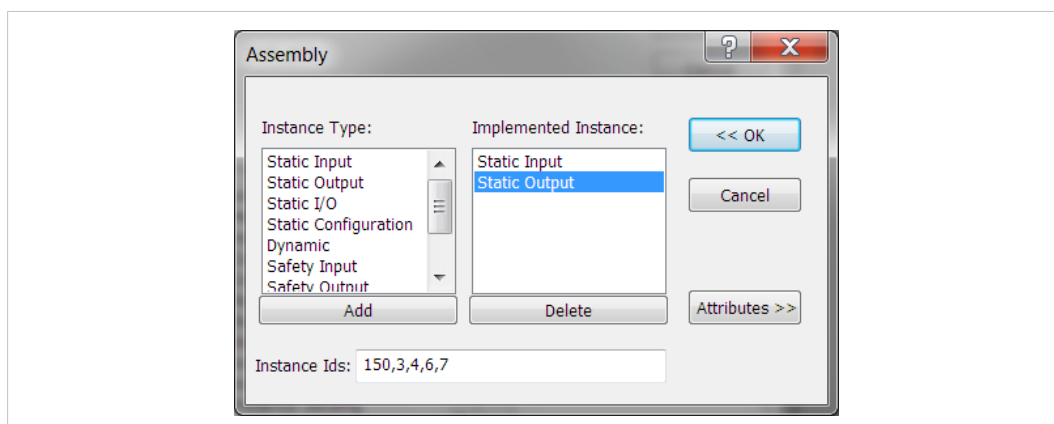


| Real time transfer format: | Modeless format | ▼ |
| Connection Type: | ☐ Null   ☑ Multicast   ☑ Point to Point | |
| Priority: | ☑ Low   ☑ High   ☑ Scheduled   ☐ Urgent | |
| Connection Path: | 20 04 24 05 2c 96 2c 64 | |

**Fig. 15**

---

ℹ️  *This attribute is an array if the host application implements the Assembly Mapping Host object, see details of this object below.*

---

### EtherNet/IP Host Object (F8h) - Attribute #9 - Enable Communication Settings from Net

This attribute sets the ability for other devices on the network to adjust the communication settings using access to the CIP TCP/IP object and the CIP Ethernet Link Object. Check the box in the Physical Data section if this method is supported.



☐ DNS Client          ☐ DHCP-D
☑ Configuration Settable
☐ Hardware Configurable
☐ Interface Configuration Change F
☑ ACD Capable

**Fig. 16**

The STC file must be configured to support the ability to set the affected attributes in the TCP/IP Interface object and in the Ethernet Link object according to the table below.

| Enable communication settings from net | CIP TCP/IP Object | CIP Ethernet Link Object |
|---|---|---|
| True | Set service enabled on attributes 3, 5, 6, 8, 9, 10, 11 and 12. | Attribute 6 enabled.<br>Set service enabled on object level. |
| False | Set service enabled on attributes 3, 8, 9, 10, 11 and 12.<br>Set service disabled on attributes 5 and 6. | Attribute 6 disabled.<br>Set service disabled on object level (does not apply if the Admin state attribute is implemented).<br>Attribute 3 shall be settable in all modes except when hardware switches are used. |

Please note that using the DLR functionality requires that Ethernet ports are able to have forced settings e.g. 10 Mbps Half-duplex. For DLR devices with Communication Setting from the network disabled, the application must provide the ability to force Ethernet settings. For example, the web server can provide the user the ability to force Ethernet settings.

### EtherNet/IP Host Object (F8h) - Attribute #11 - Enable CIP Forwarding

Enabling CIP Forwarding allows the host application to respond to all requests to both CIP objects and instances of the Assembly Object not implemented by the Anybus CompactCom. The Conformance Test software will check to see if those requests are handled properly by the application. All Objects and Assembly Object Instances listed in the section about CIP objects in the corresponding EtherNet/IP network appendices, are handled by the Anybus CompactCom. This means that all requests to CIP objects and instances not listed in the CIP objects section need to be handled by responses to the Process_CIP_Request command if CIP Forwarding is enabled.

Enabling CIP Forwarding can be necessary when users support device profiles defined by Volume 1 of the CIP Network Libraries. Additionally, vendors may define and support objects and

Assembly Object Instances that are not specified in the CIP network specification as long as those objects and Assembly Object Instances are in the vendor specific range.

CIP defined device profiles define which object(s) instance(s) attributes(s), and Instances of the Assembly Object need be supported by a device. Additionally, mandatory services and behaviors are defined. Chapter 6 of Volume 1 CIP Network Libraries details device profiles. The default profile supported by the Anybus CompactCom is Generic Device, Keyable. This device profile, and some other profiles do not require any additional objects or assembly instances to be supported making it not necessary to enable CIP forwarding.

Responding properly can mean different things for different requests at different times. The following list gives advice how to reduce the complexity.

- Decide which Object/Instance/attributes combinations will be implemented. Consult specifications to ensure that mandatory/optional/vendor specific combinations are correct.

- Decide which services are supported for the implemented combinations. Consult specification to ensure that mandatory/optional/vendor specific services are implemented properly.

- Verify that proper application behavior is provided for the correct interaction of the implemented services and paths.

- Provide the correct error response for all paths not supported by application.

  CIP status code 0x05 (path destination unknown) will be reported, when the application returns any of the following Anybus CompactCom error codes: Unsupported Object (3), Unsupported Instance (4). Consult the CIP network libraries Vol1 appendix B for status codes, and the section on CIP Objects in the Anybus CompactCom EtherNet/IP Network Guide for a translation of CIP error codes to Anybus CompactCom error codes.

- Provide the correct error response for all unsupported commands.

  CIP status code 0x08 service not supported will be reported when the application returns the Anybus CompactCom error code Unsupported Command (5). Consult the CIP network libraries Vol1 appendix B for status codes, and the CIP objects chapter in the Anybus CompactCom EtherNet/IP Network Guide for a translation of CIP error codes to Anybus CompactCom error codes.

- Provide error checking for all commands that modify variables, and respond with the correct CIP defined error code. Consult the CIP network libraries Vol1 appendix B for status codes, and the CIP Objects chapter in the Anybus CompactCom EtherNet/IP Network Guide for a translation of CIP error codes to Anybus CompactCom error codes.

Please consult the profile requirements, services, and behaviors as well as the object definitions specified in the CIP Network Libraries.

### EtherNet/IP Host Object (F8h) - Attribute #12 - Enable Parameter Object

The purpose of the CIP Parameter Object is to provide a uniform interface for device configuration. EtherNet/IP requires one instance of the parameter object per configurable parameter. A request to the CIP parameter object is converted into a request to the Host Application Data object. It is possible to disable access to the Parameter Object by responding FALSE to this request. The required and optional instance attributes are listed in table 5A-14.7 of The CIP Network Libraries Volume 1. If the object is disabled the parameter object must be removed from the list of supported objects in the STC file.

### EtherNet/IP Host Object (F8h) - Attribute #13 - Input-Only Heartbeat Instance Number

By default this instance number is 3. Changing this value from the default will require users to modify the Input only connection listed in the Connection Manager portion of the .stc file. The following figure shows that for the input only connection 03 shows up as the 0->T connection

point for the connection path. For an explanation of the configuration path please see The CIP
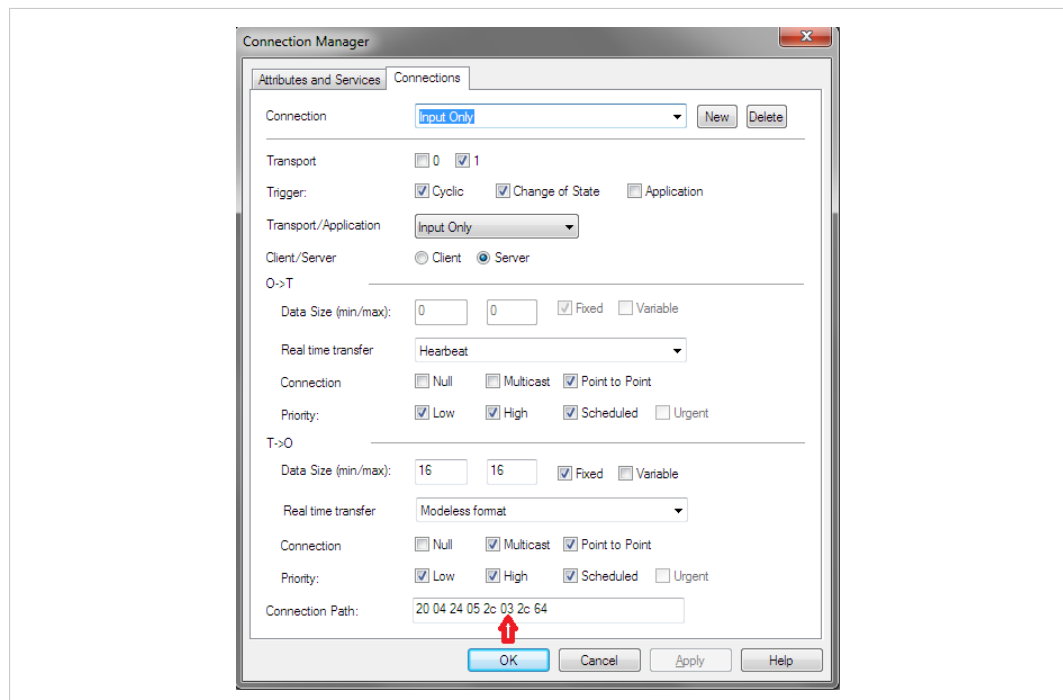Networks Library Volume 1 Appendix C.



**Fig. 17**

**EtherNet/IP Host Object (F8h) - Attribute #14 - Listen-Only Heartbeat Instance Number**

This attribute will set the Assembly Instance for the Heartbeat connection point (Originator to Target). This instance should be listed as a connection point in the connection manager object for the input only connection. The default instance number is 4.
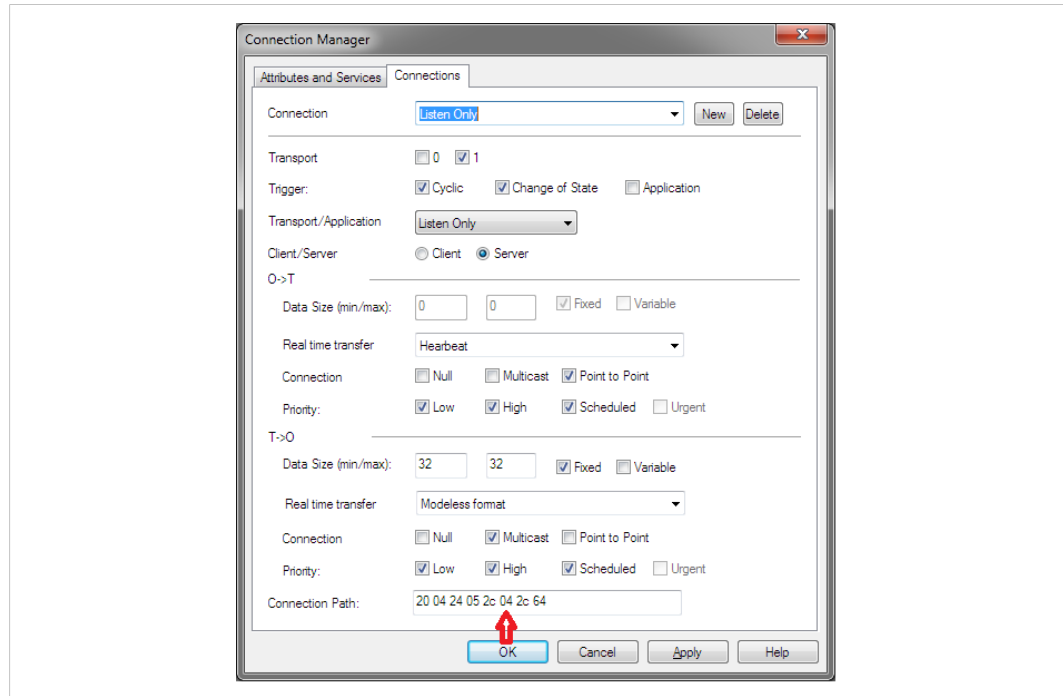


**Fig. 18**

**EtherNet/IP Host Object (F8h) - Attribute 15 - Assembly Object Configuration Instance Number**

Device configuration parameters can be grouped together in an assembly instance. By default this Instance is 5. Support for the Configuration instance is provided by the functions Get_ Configuration_Instance and Set_Configuration_Instance of the EtherNet/IP host object. If this instance is used to pass configuration data, this assembly should be listed in the Assembly object as a Static Configuration, and should be listed in the connection paths in the Connection Manager.

**EtherNet/IP Host Object (F8h) - Attribute 16 - Disable Strict IO Match**

Device configuration parameters can be grouped together in an assembly instance. By default this Instance is 5. Support for the Configuration instance is provided by the functions Get_ Configuration_Instance and Set_Configuration_Instance of the EtherNet/IP host object. If this instance is used to pass configuration data, this assembly should be listed in the Assembly object as a Static Configuration, and should be listed in the connection paths in the Connection Manager.
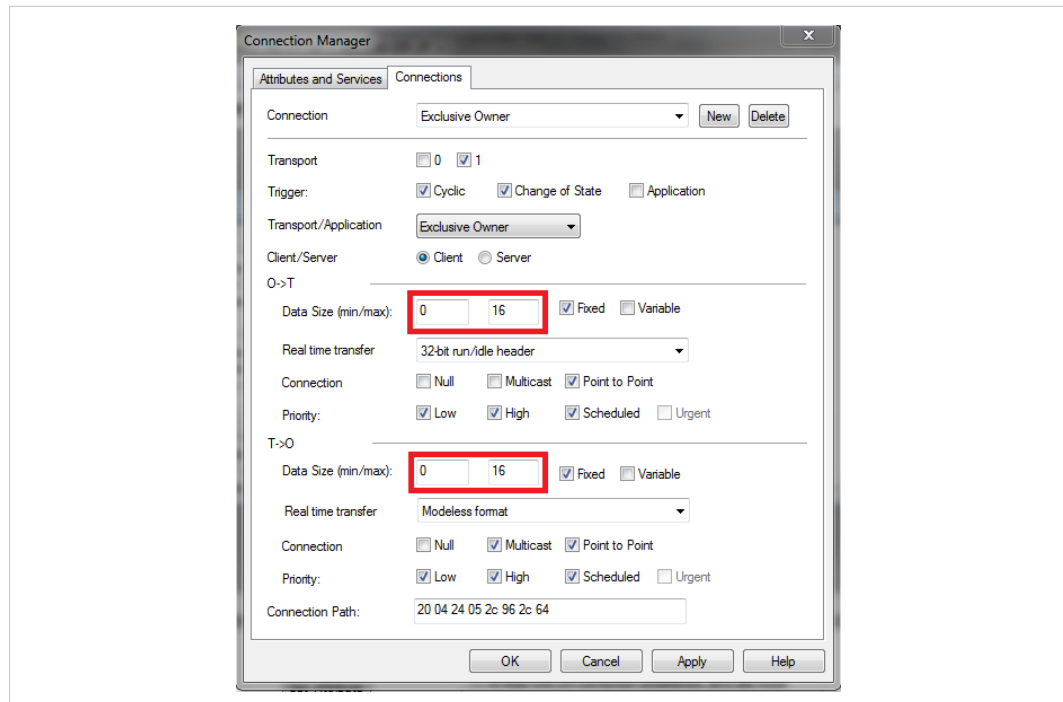


**Fig. 19**

**EtherNet/IP Host Object (F8h) - Attribute 17 - Enable Unconnected Routing**

Enabling this attribute will allow unconnected routing, and allow access to the CIP Port Object (F4h). It is possible for originators to use CIP routing to link to other subnets or backplanes through the Anybus CompactCom. For EtherNet/IP, multiple Port Object Instances can share the single or dual Physical ports. For each CIP routable port, one instance of the CIP Port Object should exists. Enabling this attribute also requires that applications support for Hosts CIP Port Configuration Object (0Dh). The Statement of Conformance file can be configured to reflect the ability for port forwarding, by selecting the check box in the Communication data section, and the port object can be added to the list of implemented objects.
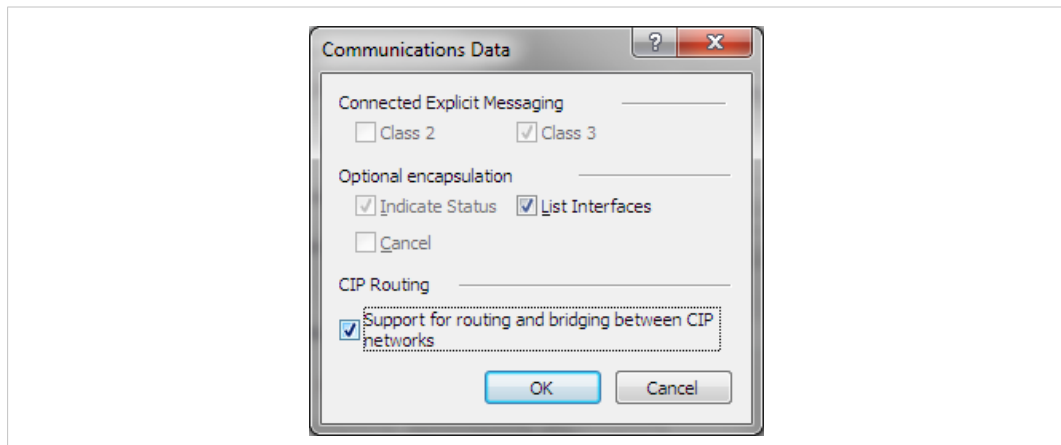


**Fig. 20**

**EtherNet/IP Host Object (F8h) - Attribute 18 - Input-Only Extended Heartbeat Instance Number**

The extended version of the Input-only Heartbeat connection is functionally the same with one exception. The state of the connection does not affect the state of the module, i.e., a timeout of this connection will not force the module into the Exception state. This Instance can be used for a connection point and that connection should be annotated in the Connection Manager. The instance number should appear in the connection path.

**EtherNet/IP Host Object (F8h) - Attribute 19 - Listen-Only Extended Heartbeat Instance Number**

The extended version of the Listen-only Heartbeat connection is functionally the same with one exception. The state of the connection does not affect the state of the module, i.e., a timeout of this connection will not force the module into the Exception state. This Instance can be used for a connection point and that connection should be annotated in the Connection Manager. The instance number should appear in the connection path.

**EtherNet/IP Host Object (F8h) - Attribute 20 - Interface label port 1**

This label is not checked by the CT test software, however if changed please ensure that the EDS file is updated with the equivalent string.

**EtherNet/IP Host Object (F8h) - Attribute 21 - Interface label port 2**

This label is not checked by the CT test software, however if changed please ensure that the EDS file is updated with the equivalent string.

**EtherNet/IP Host Object (F8h) - Attribute 22 - Interface label internal port**

This label is not checked by the CT test software, however if changed please ensure that the EDS file is updated with the equivalent string.

**EtherNet/IP Host Object (F8h) - Attribute 26 - Enable EtherNet/IP QuickConnect**

Enabling QuickConnect will make the duration from power-on to available on the network as short as possible. QuickConnect will require a change to the default EDS file, and also require that, for two port modules, ports 1 and 2 labeled externally on the device.

If QuickConnect is enabled, attribute #12 of TCP/IP Interface object needs to be set to Set and Get access in the STC file. If QuickConnect is disabled Get and Set access must be unchecked.
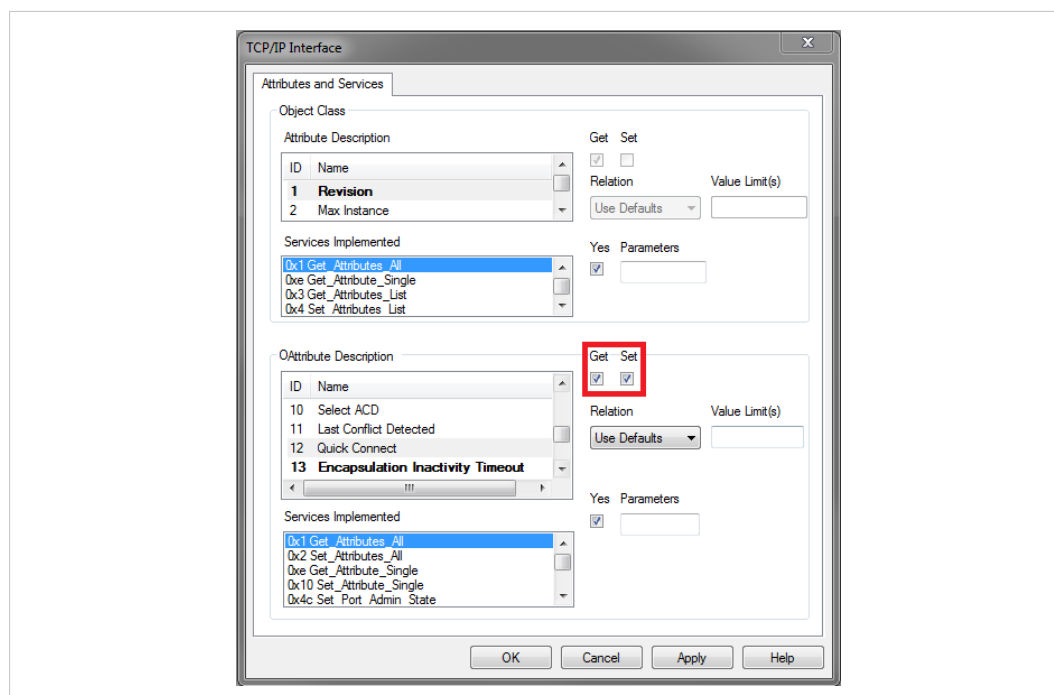


**Fig. 21**

**EtherNet/IP Host Object (F8h) - Attribute 29 - Ignore Sequence Count Check**

Enabling this functionality violates the CIP Network Libraries specifications.

**EtherNet/IP Host Object (F8h) - Attribute 30 - ABCC ADI Object Number**

The default behavior of the Anybus CompactCom EtherNet/IP family of modules provide access to Instances of the Application Data Objects through the CIP ADI Object (A2h). It is possible to change this Object Class number or disable access altogether. It is important to note that A2h is in the vendor specific range where Vendors are free to implement their own objects. Choosing an object class number outside the vendor specific range should only be done when the device provides the functionality specified by the object, and adheres to the organization of attributes and services set in the CIP Networks Library. The vendor specific range is 64h – C7h and 300h – 4FFh.

**EtherNet/IP Host Object (F8h) - Attribute 31 - Enable DLR**

The default behavior of the Anybus CompactCom EtherNet/IP is to have DLR enabled, if for some reason the DLR is disabled the DLR object must be removed from the list of supported objects in the STC file.

### Ethernet Host Object (F9h)

Many of the attributes for this object are outside ODVA's specification and have no bearing on the conformance test, and will not be listed in this document.

### Ethernet Host Object (F9h) - Attribute #1 - MAC Address (Also Attribute #8 and #9)

The MAC address should be listed in the Statement of Conformance. This attribute can be accessed in the Physical Data section.
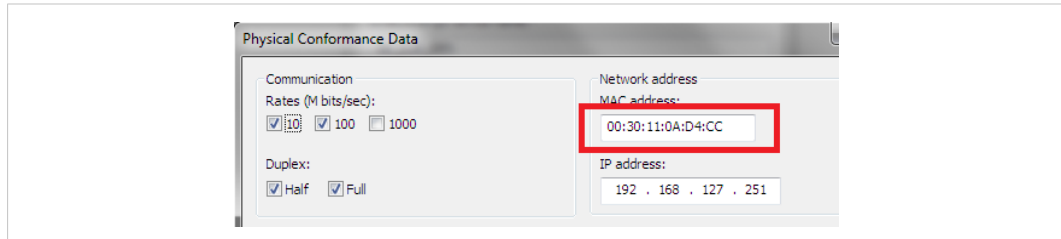


**Fig. 22**

### Ethernet Host Object (F9h) - Attribute #11 - Enable ACD (Automatic Collision Detection)

The MAC address should be listed in the Statement of Conformance. This attribute can be accessed in the Physical Data section.

Auto Collision Detection is a feature of EtherNet/IP that will detect and mitigate the errors due to multiple devices having the same IP address. This attribute can be accessed in the Physical Data section. Also, there is a section in ODVA' Conformance Test Details form that indicates if the device is ACD capable.
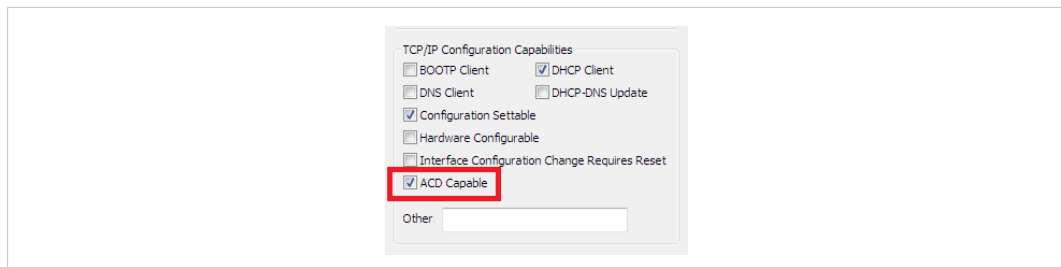


**Fig. 23**

### Ethernet Host Object (F9h) - Attribute 13 - Port 2 State

For Anybus CompactCom B40 and C40 applications port 2 may not be mounted if the application only has space for one ethernet port or for some other reason only need one ethernet port. If this attribute is set to inactive, the DLR object will automatically be disabled, hence the object must be removed from the list of supported objects in the STC file.

**Ethernet Host Object (F9h) - Attribute #24 - Enable DHCP Client**

If the host application for any reason does not want to support DHCP, this attribute shall be set to False. The DHCP client support in Physical data section of the stc file must be unchecked.
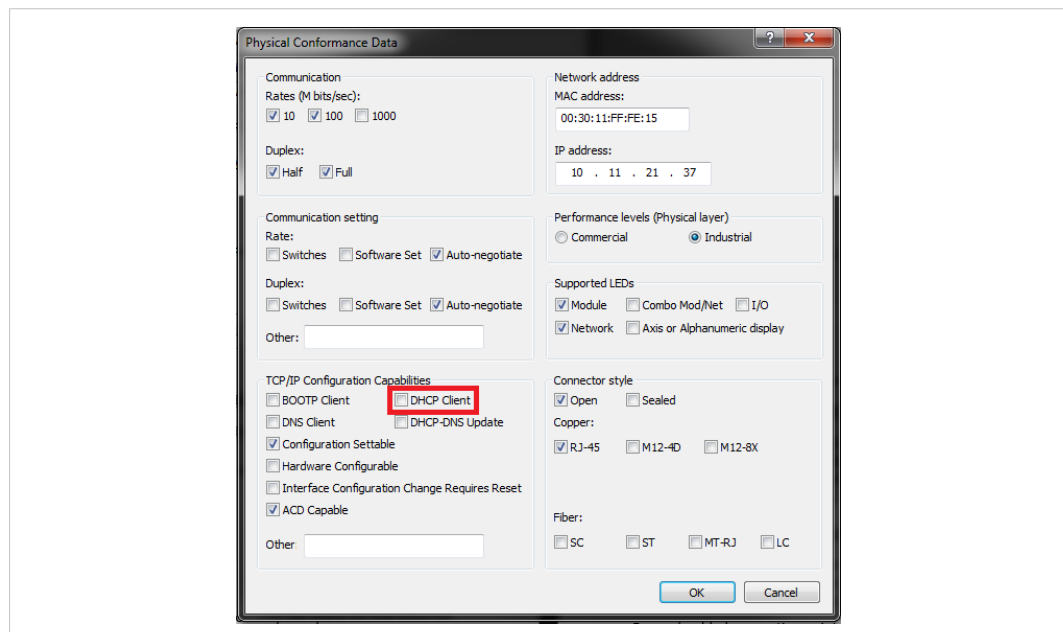


**Fig. 24**

**CIP Identity Host Object (EDh)**

This object allows devices to support additional instances of the Identity Object (CIP object, 01h) beyond the 1st instance which is supported by default. The support for additional instances of the Identity Object must be reflected in the Statement of Conformance by changing Identity Class attributes Max Instance, and Number of Instances to the proper values.

**Assembly Mapping Object (EBh)**

The assembly mapping host object can be used to create up to 6 producing and 6 consuming CIP assembly instances. These additional assemblies will also create connection points in the connection manager which will be equivalent to the standard exclusive owner connection in the Anybus CompactCom40 EtherNet/IP.

For each added connection point the Assembly object and Connection manager object in the STC file must be updated to describe these new assembly instances.
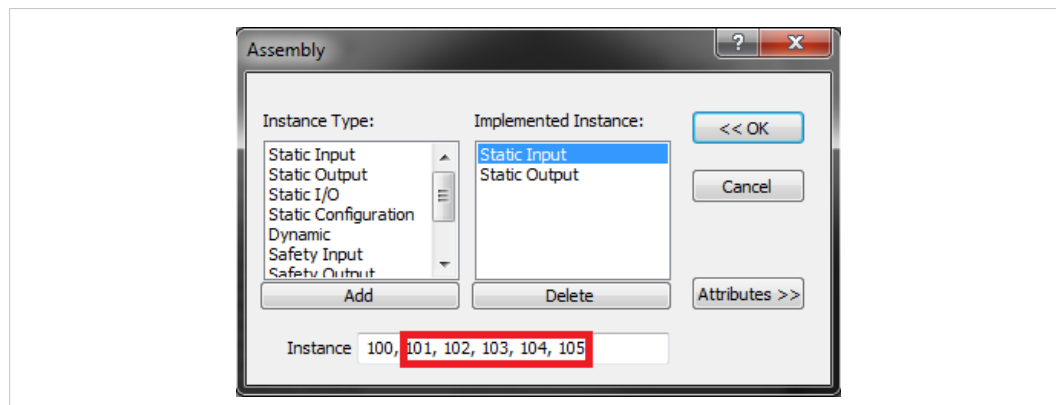


**Fig. 25**

In this example five additional producing assemblies has been added.
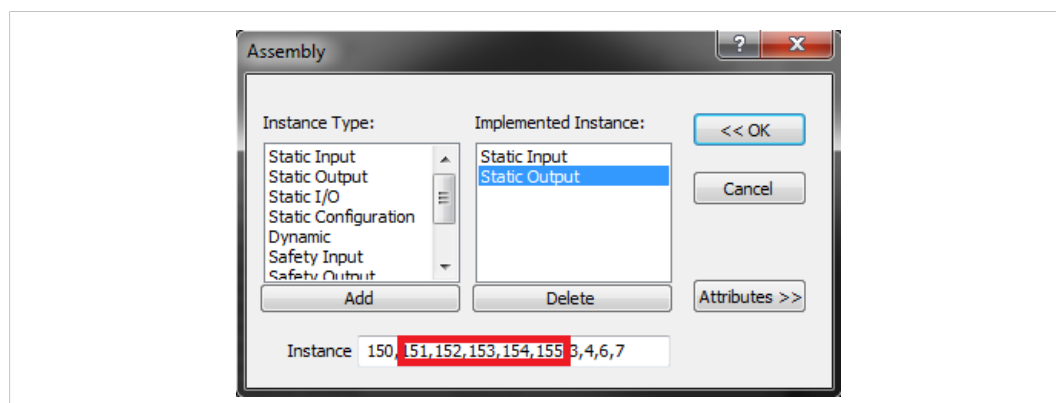


**Fig. 26**

In this example five additional consuming assemblies has been added.

It is not required to list all possible combinations of connections between the producing and consuming connection points in the Connection Manager section of the STC file. It is up to the vendor do decide which connection combinations that should be available for the customer in the EDS file, it is however required to list all connections that exist in the EDS file in the Connection Manager section of the EDS file.
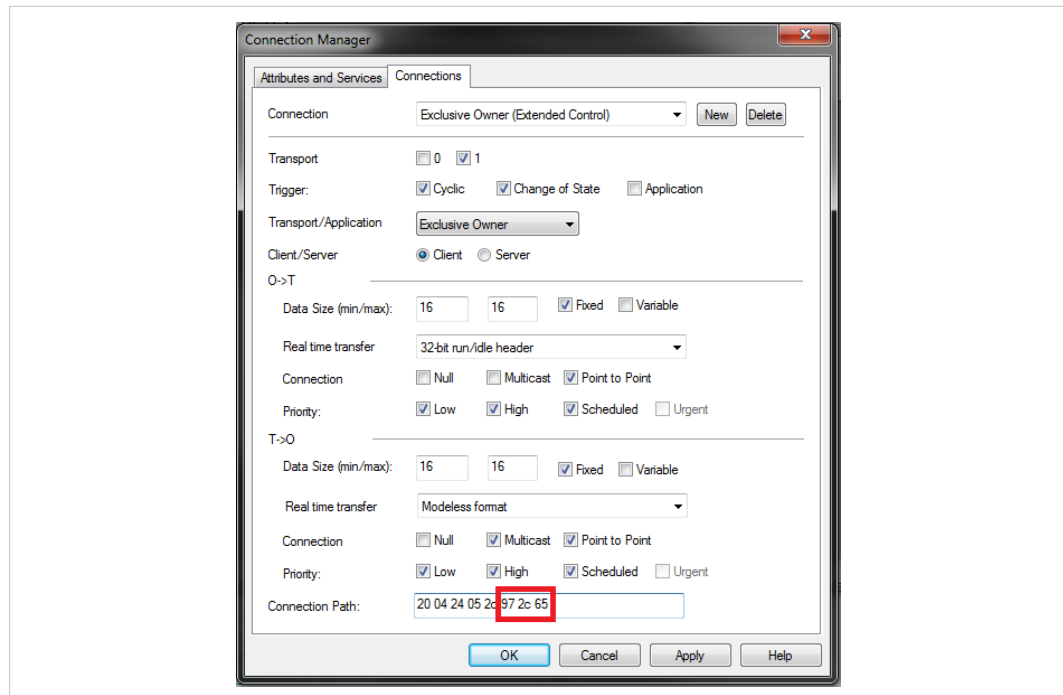
**Fig. 27**

This is an example of an additional exclusive owner connection connecting to connection points 101/151.

> ⓘ  *Please note that it is required to implement the Write_Assembly_Data and Read_Assembly_Data services of the Assembly mapping host object in the application to pass Conformance testing.*

### F.3.2        Implementation of Anybus Module Objects

Only the Anybus module objects relevant to EtherNet/IP will be discussed.

**Network Object (03h)**

**Network Object (03h) - Attribute #5 - Write Process Data Size**

The Write Process Data Size represents the current amount of data mapped for the T-> O connection. In most cases, but not always, this value will correspond to the T->O connection size in the connection manager. Please note that it is possible to support multiple assemblies for connection points by supporting process data remapping.

**Network Object (03h) - Attribute #6 - Read Process Data Size**

The Read Process Data Size represents the current amount of data mapped for the O->T connection. In most cases, but not always, this value will correspond with the O->T connection size in the connection manager. Please note that it is possible to support multiple assemblies for connection points by supporting process data remapping.

**Network Object (03h) - CIP Port Configuration Object (0Dh)**

CIP routing can be enabled by instance attribute #17 of the Host EtherNet/IP object. Each instance of this object corresponds to an instance of the CIP Port object (F4h).

Please note that a CIP port does not necessarily correspond to a Physical port. The two network connectors on the two-port Anybus CompactCom correspond to a single CIP routable port. Devices with a single CIP port are not required to support the Port Object, but the Communication Adapter device profile does require support for this object.

# G        Backward Compatibility

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

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

## G.1        Initial Considerations

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

- Add support with as little work as possible i.e. reuse as much as possible of the current design.

    – This is the fastest and easiest solution but with the drawback that many of the new features available in the 40-series will not be enabled (e.g. enhanced and faster communication interfaces, larger memory areas, and faster communication protocols).

    – You have to check the hardware and software differences below to make sure the host application is compatible with the 40-series modules. Small modifications to your current design may be needed.

- Make a redesign and take advantage of all new features presented in the 40-series.

    – A new driver and host application example code are available at www.anybus.com/starterkit40 to support the new communication protocol. This driver supports both 30-series and 40-series modules.

    – You have to check the hardware differences below and make sure the host application is compatible with the 40-series modules.

---

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

---

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

## G.2 Hardware Compatibility

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

### G.2.1 Module

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



**Fig. 28      Anybus CompactCom M30/M40**

### G.2.2 Chip

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

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

### G.2.3 Brick

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



**Fig. 29** **Anybus CompactCom B30**



**Fig. 30** **Anybus CompactCom B40–1 (not for migration)**



**Fig. 31** **Anybus CompactCom B40–2**

## G.2.4    Host Application Interface



**Fig. 32**

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

### Tx/OM3

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

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

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

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

### Module Identification (MI[0..1])

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

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

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

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

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

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

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

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

- tied these pins to GND

- pulled up the pins

- pulled down the pins

- left the pins unconnected

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

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

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

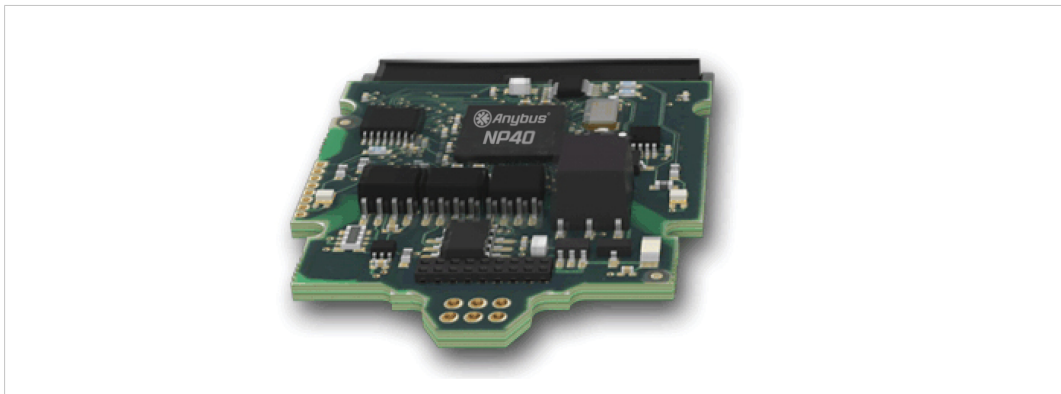Related Information: *Anybus CompactCom M40 Hardware Design Guide (HMSI-216-126)*, Section "LED Interface/D8-D15 (Data Bus)".

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

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

This change should not affect your product.

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

**Address Pins A[11..13]**

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

**Max Input Signal Level ($V_{IH}$)**

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

**RMII Compatibility**

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

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

## G.3          General Software

### G.3.1        Extended Memory Areas

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

> ℹ️ *To use the extended memory areas you need to implement a new communication protocol which is not part of this document.*
>
> *Memory areas not supported by the specific network cannot be used. Make sure you do not access these areas, e.g. for doing read/write memory tests.*

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

### G.3.2        Faster Ping-Pong Protocol

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

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

### G.3.3        Requests from Anybus CompactCom to Host Application During Startup

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

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

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

### G.3.4        Anybus Object (01h)

| Attribute | 30-series | 40-series | Change/Action/Comment |
|---|---|---|---|
| #1, Module Type | 0401h | 0403h | Make sure the host application accepts the new module type value for the 40-series. |
| #15, Auxiliary Bit | Available | Removed | It is not possible to turn off the "Changed Data Indication" in the 40-series. Also see "Control Register CTRL_AUX-bit" and "Status Register STAT_AUX-bit" below. |
| #16, GPIO Configuration | Default: General input and output pins | Default: LED3 and LED4 outputs | See also ..<br>• *GIP[0..1]/LED3[A..B], p. 146*<br>• *GOP[0..1]/LED4[A..B], p. 146* |

### G.3.5        Control Register CTRL_AUX-bit

| | |
|---|---|
| **30-series** | The CTRL_AUX bit in the control register indicates to the Anybus CompactCom if the process data in the current telegram has changed compared to the previous one. |
| **40-series** | The value of the CTRL_AUX bit is always ignored. Process data is always accepted. |

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

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

### G.3.6        Status Register STAT_AUX-bit

| | |
|---|---|
| **30-series** | The STAT_AUX bit in the status register indicates if the output process data in the current telegram has changed compared to the previous one. This functionality must be enabled in the Anybus object (01h), Attribute #15. By default, the STAT_AUX bit functionality is disabled. |
| **40-series** | The STAT_AUX bit indicates updated output process data (not necessarily changed data) from the network compared to the previous telegram. The functionality is always enabled. |

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

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

### G.3.7        Control Register CTRL_R-bit

| | |
|---|---|
| **30-series** | The application may change this bit at any time. |
| **40-series** | For the 8-bit parallel operating mode, the bit is only allowed to transition from 1 to 0 when the STAT_M-bit is set in the status register. When using the serial operating modes, it is also allowed to transition from 1 to 0 in the telegram immediately after the finalizing empty fragment. |

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

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

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

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

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

## G.4 Network Specific — EtherNet/IP

### G.4.1 Network Object (03h)

**Attribute #1, Network Type** The 30-series module is available in two network type versions, either with "Beacon based DLR" (Highest performance) or with "Announce based DLR" which both are Ethernet redundancy protocols. The 40-series is only available with "Beacon based DLR". The network type value differs between the versions.

| Value | Network Type | Anybus CompactCom Product |
|---|---|---|
| 0085h | EtherNet/IP, No DLR | 30-series 1-port |
| 009Ch | EtherNet/IP, Announce Based DLR | 30-series 2-port |
| 009Bh | EtherNet/IP, Beacon Based DLR | 30-series and 40-series |
| 00ABh | EtherNet/IP, Beacon Based DLR + IIoT | 40–series |

### G.4.2 EtherNet/IP Host Object (F8h)

| Attribute | Default | Anybus CompactCom Product | Comment |
|---|---|---|---|
| #2, Device Type | 0000h | 30-series, EtherNet/IP, No DLR | If the attribute is implemented in the host application, it overrides the default value and there is no difference between the 30-series and the 40-series.<br>If the attribute is not implemented, the default value is used. |
| | 0000h | 30-series, EtherNet/IP, Announce Based DLR | |
| | 002Bh | 30-series, EtherNet/IP, Beacon Based DLR | |
| | 002Bh | 40-series, EtherNet/IP, Beacon Based DLR | |
| #3, Product Code | 0063h | 30-series, EtherNet/IP, No DLR | If the attribute is implemented in the host application, it overrides the default value and there is no difference between the 30-series and the 40-series.<br>If the attribute is not implemented, the default value is used. |
| | 002Eh | 30-series, EtherNet/IP, Announce Based DLR | |
| | 0036h | 30-series, EtherNet/IP, Beacon Based DLR | |
| | 0037h | 40-series, EtherNet/IP, Beacon Based DLR | |
| #6, Product Name | Anybus-CC EtherNet/IP | 30-series, EtherNet/IP, No DLR | If the attribute is implemented in the host application, it overrides the default value and there is no difference between the 30-series and the 40-series.<br>If the attribute is not implemented, the default value is used. |
| | CompactCom EtherNet/IP (TM) 2P | 30-series, EtherNet/IP, Announce Based DLR | |
| | Anybus-CC EIP (2-Port) BB DLR | 30-series, EtherNet/IP, Beacon Based DLR | |
| | Anybus CompactCom 40 EtherNet/IP(TM) | 40-series, EtherNet/IP, Beacon Based DLR | |
| Attribute #27, Producing Instance Map | See comment | | Attribute removed in the 40-series (only available in the 30-series EtherNet/IP Beacon Based DLR). The CompactCom will never request this attribute. Replaced by the functionality in the Assembly Mapping Object (EBh).<br>If this attribute is used, the Assembly Mapping object must be implemented instead. |
| Attribute #28, Consuming Instance Map | See comment | | Attribute removed in the 40-series (only available in the 30-series EtherNet/IP Beacon Based DLR). The CompactCom will never request this attribute. Replaced by the functionality in the Assembly Mapping Object (EBh).<br>If this attribute is used, the Assembly Mapping object must be implemented instead. |

**EtherNet/IP functionality**

| | |
|---|---|
| **Max Message Connections** | The maximum number of simultaneous Class 3 connections are 16 in the 30-series and 6 in the 40-series. |
| | No change is needed in the host application. |
| **EtherNet/IP Encapsulation Sessions** | The maximum number of simultaneous encapsulation sessions are 48 in the 30-series and 15 in the 40-series. |
| | No change is needed in the host application. |

## G.4.3 EDS file (Electronic Datasheet file used by configuration tool)

### EDS file Generator Tool

An EDS-generator for automatic EDS-file generation up to date with the differences below. The EDS-generator only works with the 40-series, version 1.30 and later.

The generator can be downloaded from www.anybus.com/starterkit40: .

### Keywords

The following keywords differs between the 30-series and the 40-series. The EDS generator reflects this change.

| Keyword | Comments |
|---|---|
| Capacity->MaxCIPConnections | Removed in 40-series – replaced by: MaxMsgConnections and MaxIOConnections (see below) |
| Capacity->MaxMsgConnections | New keyword in the 40-series, Value: 6 |
| Capacity->MaxIOConnections | New keyword in the 40-series, Value: 4 |

# H License Information

lwIP is licenced under the BSD licence:

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

All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.


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

--------------------------------------------------------------------------------

Print formatting routines

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

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.


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

--------------------------------------------------------------------------------

Copyright (c) 2002 Florian Schulze.
All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the authors nor the names of the contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

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

--------------------------------------------------------------------------------

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

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

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

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

--------------------------------------------------------------------------------

Copyright (c) 2016 The MINIX 3 Project.
All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT

SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Author: David van Moolenbroek <david@minix3.org>

--------------------------------------------------------------------------------

MD5 routines

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

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

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

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

L. Peter Deutsch

ghost@aladdin.com

--------------------------------------------------------------------------------

Copyright 2013 jQuery Foundation and other contributors

http://jquery.com/

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

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

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

--------------------------------------------------------------------------------

rsvp.js

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

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

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

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

--------------------------------------------------------------------------------

libb (big.js)
The MIT Expat Licence.
Copyright (c) 2012 Michael Mclaughlin

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

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

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

--------------------------------------------------------------------------------

The "inih" library is distributed under the New BSD license:
Copyright (c) 2009, Ben Hoyt
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
* Neither the name of Ben Hoyt nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

open62541 is licensed under the Mozilla Public License v2.0

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/.

To obtain customized changes please contact foss@anybus.com.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

musl as a whole is licensed under the following standard MIT license:

----------------------------------------------------------------------

Copyright © 2005-2014 Rich Felker, et al.

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

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

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

----------------------------------------------------------------------

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

PCG Random Number Generation for C.

Copyright 2014 Melissa O'Neill <oneill@pcg-random.org>

Licensed under the Apache License, Version 2.0 ("the License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

For additional information about the PCG random number generation scheme, including its license and other licensing options, visit

http://www.pcg-random.org

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

queue.h

Copyright (c) 1991, 1993

The Regents of the University of California. All rights reserved.

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

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

@(#)queue.h 8.5 (Berkeley) 8/20/94

--------------------------------------------------------------------------------

Format - lightweight string formatting library.
Copyright (C) 2010-2013, Neil Johnson
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
* Neither the name of nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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