



Anybus[®] CompactCom[™] 40

PROFIBUS DP-V0/DP-V1

NETWORK GUIDE

HMSI-27-210 2.2 ENGLISH

Important User Information

Liability

Every care has been taken in the preparation of this document. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the USA and other countries.

Table of Contents

Page

1	Preface	5
1.1	About this document	5
1.2	Related Documents	5
1.3	Document History	5
1.4	Document Conventions	5
1.5	Document Specific Conventions	6
1.6	Abbreviations	6
1.7	Trademark Information	6
2	About the Anybus CompactCom 40 PROFIBUS DPV1/DPV0	8
2.1	General	8
2.2	Features	8
3	Fieldbus Conformance and Certification	9
3.1	Fieldbus Conformance Notes	9
3.2	Certification	9
4	Basic Operation	10
4.1	General Information	10
4.2	Communication Settings	11
4.3	Device Identity	12
4.4	Data Exchange	13
4.5	Parameterization Data Handling	14
4.6	Configuration Data Handling	15
4.7	Set Slave Address	18
4.8	Parameter Read/Write with Call	18
4.9	Identification & Maintenance (I&M)	21
5	Diagnostics	23
5.1	Standard Diagnostics	23
5.2	Additional Diagnostics Information	23
6	Anybus Module Objects	24
6.1	General Information	24
6.2	Anybus Object (01h)	25
6.3	Diagnostic Object (02h)	26
6.4	Network Object (03h)	28
6.5	Network Configuration Object (04h)	30
6.6	PROFIBUS DP-V0 Diagnostic Object (10h)	32

7	Host Application Objects	34
7.1	General Information	34
7.2	Modular Device Object (ECh).....	35
7.3	PROFIBUS DP-V1 Object (FDh)	36
A	Categorization of Functionality	41
A.1	Basic.....	41
A.2	Extended.....	41
B	Implementation Details.....	42
B.1	SUP-Bit Definition	42
B.2	Anybus State Machine	42
B.3	Watchdog Behavior (Application Stopped)	42
C	Error Handling.....	43
C.1	Translation of Anybus Error Codes	43
C.2	Other Errors	43
D	GSD File Customization	44
D.1	General.....	44
D.2	Device Identification	44
D.3	Supported Hardware Features	45
D.4	Supported DP Features	46
D.5	Supported Baud Rates	46
D.6	Maximum Responder Time for Supported Baud Rates	47
D.7	Maximum Polling Frequency	47
D.8	I/O Related Keywords	47
D.9	Definition of Modules.....	48
D.10	Parameterization Related Keywords.....	53
D.11	Diagnostic Related Keywords	54
D.12	Identification & Maintenance Related Keywords	54
D.13	Status Diagnostics Messages	55
D.14	DP-V1 Related Keywords	56
D.15	Alarm Related Keywords	56
E	Technical Specification	57
E.1	Front View	57
E.2	Functional Earth (FE) Requirements.....	58
E.3	Power Supply	58
E.4	Environmental Specification.....	59
E.5	EMC Compliance	59

F	Timing & Performance	60
F.1	General Information	60
F.2	Internal Timing	60
G	Backward Compatibility	62
G.1	Initial Considerations	62
G.2	Hardware Compatibility	62
G.3	General Software	67
G.4	Network Specific — PROFIBUS	69

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 PROFIBUS DPV1/DPV0. The document describes the features that are specific to Anybus CompactCom 40 PROFIBUS DPV1/DPV0. 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 PROFIBUS specific functionality is to be used, in-depth knowledge of PROFIBUS networking internals and/or information from the official PROFIBUS specifications may be required. In such cases, the persons responsible for the implementation of this product should either obtain the PROFIBUS 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
PROFIBUS Profile Guidelines Part 1: Identification & Maintenance Functions	PNO	
Specification for PROFIBUS Device Description and Device Integration Volume 1: GSD (order. no. 2.122)	PNO	

1.3 Document History

Version	Date	Description
1.00	2014-03-25	First official revision
1.10	2014-04-28	Added information on DP-V0
1.20	2014-08-11	Major updates
1.30	2015-11-13	Minor updates
2.0	2017-01-04	Moved from FM to DOX, M12 connectors added, misc. updates
2.1	2017-02-24	Minor corrections
2.2	2017-07-10	Added appendix on backward compatibility Minor correction

1.4 Document Conventions

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

1. First do this
2. Then do this

Unordered (bulleted) lists are used for:

- Itemized information

- Instructions that can be carried out in any order

...and for action-result type instructions:

- ▶ This action...
 - ➡ leads to this result

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

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

This is a cross-reference within this document: [Document Conventions, p. 5](#)

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



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



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



Caution

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



WARNING

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

1.5 Document Specific Conventions

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

1.6 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 Trademark Information

Anybus® is a registered trademark of HMS Industrial Networks AB.

All other trademarks are the property of their respective holders.

2 About the Anybus CompactCom 40 PROFIBUS DPV1/DPV0

2.1 General

The Anybus CompactCom 40 PROFIBUS DPV1/DPV0 communication module provides instant PROFIBUS 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.

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

- Supports PROFIBUS DP-V1 and DP-V0
- PROFIBUS connector (9-pin female D-Sub) or M12 connectors
- Automatic baud rate detection
- Max. read process data: 244 bytes
- Max. write process data: 244 bytes
- Max. process data (read + write, in bytes): 488 bytes
- Generic and PROFIBUS specific diagnostic support
- User Parameterization Data support
- Set Slave Address support
- ADI access via DP-V1 read/write services
- Device identity customization
- Generic GSD file provided
- Support for Modular Device Mode

3 Fieldbus Conformance and Certification

3.1 Fieldbus Conformance Notes

Using the GSD file supplied by HMS Industrial Networks AB, the module is precertified for network compliance. However, since parameter changes which require deviations from the standard GSD file are necessary, a recertification is advised, though not mandatory.

For further information, please contact HMS Industrial Networks AB.

3.2 Certification

The following items are necessary to perform to obtain a certification:

Change PNO Ident Number:

The PNO Ident Number can be requested from PNO (PROFIBUS Nutzerorganisation e.V.). Replace the default PNO Ident Number with this. This is done by implementing the PROFIBUS DP-V1 object (FDh), instance #1, attribute #1, and returning the PNO Ident Number when receiving a Get_Attribute request.

Add Node Address Information:

If the host application does not set a valid node address by messaging the Network Configuration Object (04h), instance 1 ("Node Address"), the PROFIBUS Set Slave Address (SSA) service is enabled.

If SSA functionality is enabled, it is mandatory to provide a mechanism for resetting the node address to its default value (126). This is because it is possible to lock the value from the network side. See [Set Slave Address, p. 18](#) for more information.

Change Manufacturer Id, Order Id, serial number and revision information:

This is done by implementing the PROFIBUS DP-V1 object (FDh), instance #1, attributes #8 - #12, and returning the corresponding attributes when receiving a Get_Attribute request.

The Manufacturer Id can be requested from PNO (PROFIBUS Nutzerorganisation e.V.).

Modify the GSD file:

Modify the PROFIBUS DP-V1 GSD file so that it corresponds to the changes made above.

In addition, all modules used in the application must be defined in the GSD file. For more information, see [Configuration Data Handling, p. 15](#).

4 Basic Operation

4.1 General Information

4.1.1 Software Requirements

No additional network support code needs to be written in order to support the Anybus CompactCom 40 PROFIBUS DPV1/DPV0, however certain restrictions must be taken into account:

- Due to the nature of the PROFIBUS networking system, at least one ADI must be mapped to Process Data.
- Only ADIs with instance numbers less than 65026 can be accessed acyclically from the network.
- The host application must be able to provide a response to an ADI request within the time period specified by the GSD file ([DP-V1 Related Keywords, p. 56](#), “C1_Response_Time-out”), or the master will terminate the connection and reparameterize the slave. The default value for this parameter (i.e. the time specified by the generic GSD file supplied by HMS Industrial Networks AB) is 1 (one) second.
- The order in which ADIs are mapped to Process Data is significant and must be replicated in the PROFIBUS master when setting up the network communication (i.e. the I/O modules must be set up in the same order, and with the same size and direction, as the mapped ADIs). If not taken into account, the network connection establishment will fail and no communication will take place.
- The use of advanced PROFIBUS specific functionality may require in-depth knowledge in PROFIBUS networking internals and/or information from the official PROFIBUS specification (IEC 61158). In such cases, the ones responsible for the implementation of this product should either obtain the PROFIBUS specification to gain sufficient knowledge or limit their implementation in such a way that this is not necessary.

4.1.2 Support for DP-V1 and DP-V0

The Anybus CompactCom PROFIBUS module supports both DP-V1 and DP-V0. At delivery the default settings give full DP-V1 functionality. However the PROFIBUS network master can choose to limit the functionality to DP-V0 via the parametrization telegrams during startup.

4.1.3 Electronic Data Sheet (GSD)

On PROFIBUS, the characteristics of a device is stored in an ASCII data file with the suffix GSD. This file is used by the PROFIBUS configuration tool when setting up the network.

HMS Industrial Networks AB provides an example GSD 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 a way that invalidates the generic GSD file.

The example GSD file supports full DP-V1 functionality, another GSD file is needed if the module is to run only DP-V0 functionality.

See also...

- [Fieldbus Conformance Notes, p. 9](#)
- [GSD File Customization, p. 44](#)

4.2 Communication Settings

Network related communication settings are grouped in the Network Configuration Object (04h).

Node Address:

See [Network Configuration Object \(04h\), p. 30](#) for more information

Baud Rate:

The baud rate is detected automatically by the module. The following baud rates are supported:

- 9.6 kbps
- 19.2 kbps
- 45.45 kbps
- 93.75 kbps
- 187.5 kbps
- 500 kbps
- 1.5 Mbps
- 3 Mbps
- 6 Mbps
- 12 Mbps

4.3 Device Identity

By default, the Anybus CompactCom module appears as a generic HMS Industrial Networks AB device with the following network identity:

Vendor Name	"HMS Industrial Networks"
Model Name	"Anybus CompactCom 40 DP-V1"
Ident Number	1815h

It is possible to customize the network identity information so that the Anybus module appears as a vendor specific implementation rather than a generic HMS Industrial Networks AB product.

The PROFIBUS I&M-functionality (Identification & Maintenance) provides a standard way of gathering information about an I/O device. The I&M information is accessed by the master by means of the Call State Machine using DP-V1 read/write services.

By default, Anybus module supports I&M records 0... 4 for slot #0 (which is the device itself). Optionally, the host application can implement the commands Get_IM_Record and Set_IM_Record'-commands ([PROFIBUS DP-V1 Object \(FDh\)](#), [p. 36](#)) to support all I&M records for all slots.

See also...

- PROFIBUS Profile Guidelines Part 1: Identification & Maintenance Functions
- [Network Configuration Object \(04h\)](#), [p. 30](#)
- [PROFIBUS DP-V1 Object \(FDh\)](#), [p. 36](#)
 - Attribute #1, PNO Ident Number
 - Command details for Get_IM_Record
 - Command details for Set_IM_Record
- [Device Identification](#), [p. 44](#)
- [Identification & Maintenance Related Keywords](#), [p. 54](#)

4.4 Data Exchange

4.4.1 Application Data Instances (ADIs)

ADIs can be accessed acyclically from the network using DP-V1 read/write services. The module translates these services into object requests towards the Application Data Object. If the host application responds with an error to such a request, the error code in the response will be translated to DP-V1 standard.

If the Modular Device Object is implemented in the application, the addressing of ADIs is carried out in accordance with the Modular Device Object. For more information, see [Modular Device Object \(ECh\)](#), p. 35. If the Modular Device Object is not implemented, ADIs are mapped to slots and indexes as follows:

Correlation:

$$\text{ADI} = \text{slot} \cdot 255 + \text{index} + 1$$

$$\text{slot} = (\text{ADI} - 1) / 255$$

$$\text{index} = (\text{ADI} - 1) \text{ MOD } 255$$

Examples:

ADI	Slot	Index
256	1	0
510	1	0
65025	254	254

The length parameter in the DP-V1 request specifies the number of bytes to read/write.

- When reading more data than the actual size of the ADI, the response will only contain the actual ADI data, i.e. no padding on the data is performed by the module.
- When reading less data than the actual size of the ADI, only the requested amount of data is returned by the module.
- The maximum ADI data size that can be accessed is 240 bytes for acyclic DP-V1 read/writes and 234 bytes for acyclic read/writes using the call service.
- When writing to an ADI, the length parameter is not checked by the module, i.e. the host application must respond with an error if the length differs from the actual size of the requested ADI.



Due to technical reasons, it is generally not recommended to use ADI numbers 1... 255 since this may cause trouble with certain PROFIBUS configuration tools.

See also..

- [GSD File Customization](#), p. 44
- [Implementation Details](#), p. 42

4.4.2 Process Data

Mapping an ADI to Write Process Data results in PROFIBUS input data, and mapping an ADI to Read Process Data results in PROFIBUS output data. If the host application tries to map more data than PROFIBUS permits, the module will go into the EXCEPTION state (exception code 06h) after "Setup Complete".

See also..

- [GSD File Customization, p. 44](#)
- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)

4.5 Parameterization Data Handling

4.5.1 General Information

The master identifies itself to the slaves by sending Parameterization Data, specifying how the slave shall operate (i.e. Master address, PNO-ID, Sync/Freeze capabilities etc.).

	DP Standard Parameters	DP-V1 Status Bytes	User Parameterization Data
Size	7 bytes	3 bytes	Dynamic
Defined by	IEC	IEC	Host application
Evaluated by	Module	Module	Host application
Supported in the Generic HMS GSD file	Yes	Yes	No

User Parameterization Data is not supported by default. Optionally, User Parameterization Data can be supported by implementing the attribute Parameterization Data (#2) in the PROFIBUS DP-V1 Object (FDh). In such case, the generic GSD file supplied by HMS must be modified.

The maximum amount of User Parameterization Data that can be handled by the module is 234 bytes.

See also...

- [GSD File Customization, p. 44](#)
- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)(Attribute #2, Parameterization Data)
- [Parameterization Related Keywords, p. 53](#)

4.5.2 Validation

The User Parameterization Data must be evaluated by the host application. This is handled through the attribute Parameterization Data (#2) in the [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)..

Attribute Parameterization Data not implemented

If the Parameterization Data contains any User Parameterization Data, the module will reject the Parameterization Data.

Attribute Parameterization Data implemented

The application must evaluate the contents of the attribute and provide a suitable response.

- To accept the Parameterization Data, respond with no error code.
- To reject the Parameterization Data, respond with one of the following error codes:
 - NOT_ENOUGH_DATA
 - TOO_MUCH_DATA
 - OUT_OF_RANGE

4.6 Configuration Data Handling

4.6.1 General Information

The Anybus module determines its Expected Configuration Data based on the ADI mapping process. Alternatively, it can be specified by the host application by implementing the Get service of the attribute configuration Data (#3) in the [PROFIBUS DP-V1 Object \(FDh\), p. 36](#).

Implementing this attribute in the PROFIBUS DP-V1 Object (FDh) in the host application is optional.

The maximum amount of configuration data that can be handled by the module is 244 bytes.

See also...

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#) (Attribute #3, Configuration Data)
- [I/O Related Keywords, p. 47](#)
- [Definition of Modules, p. 48](#)

4.6.2 Validation

Using the Chk_Cfg service, the PROFIBUS master will send the Actual Configuration Data needed for the application to the module. The module will compare the Actual Configuration Data with the Expected Configuration Data. In case of a mismatch, the module will send the Actual Configuration Data to the host application for further evaluation.

Attribute Configuration Data not implemented

The module will calculate the Expected Configuration Data. The way this is done will depend on whether the application has implemented the Modular Device Object or not. For more detailed information...

- [Definition of Modules, p. 48](#) (Modular Device Object Implemented)
- [Parameterization Related Keywords, p. 53](#)

In case of a mismatch between the Actual Configuration Data and the Expected Configuration Data, the module will send a remap command to the application based on the Actual Configuration Data.

If the application approves the new configuration, the module will accept it and go online.

If the application discards the remap command, the module will signal configuration fault on the network.

For backward compatibility with the Anybus CompactCom 30 series, it is also possible for the module to approve the configuration if the following criteria are met:

- Actual Configuration Data is not in special format
- The lengths of the Actual Configuration Data and the Expected Configuration Data are equal
- Bit 0 (ChkCfg mode) in byte 2 of the DP-V1 status is set to 1

Attribute Configuration Data implemented

Expected Configuration Data is provided by the application. The format of the configuration data is not specified or known by the module. The module will however calculate the total input and output lengths based on the parts of the configuration that it can interpret. If the provided configuration data length does not match the length of all mapped ADIs, the module will enter EXCEPTION state.

If the Actual Configuration Data matches the Expected Configuration Data, the module will approve the configuration.

If the Actual Configuration Data does not match the Expected Configuration Data, and if bit 0 (ChkCfg mode) in byte 2 of the DP-V1 status is set to 1, the host application must evaluate the contents of the Configuration Data attribute.

- To accept the Configuration Data, respond with no error code.



If the new configuration affects the Process Data mapping, it is important that the host application updates the Process Data before responding. Failure to observe this may cause erroneous data to be sent to the bus on the next state shift. Preferably, choose to reject the Actual Configuration Data and adapt to it by restarting the Anybus module and then revise the Process Data map and/or the Expected Configuration Data. Also note that the new configuration must exist in the GSD file of the product.

Please note that if the application accepts the Configuration Data, but the length of the Expected Configuration Data is less than the length of the Actual Configuration Data, the configuration will be rejected by the module.

- To reject the Configuration Data, respond with one of the following error codes:
 - NOT_ENOUGH_DATA
 - TOO_MUCH_DATA
 - OUT_OF_RANGE
 - INVALID_STATE
 - NO_RESOURCES

4.7 Set Slave Address

The module supports the Set Slave Address service, which enables a master or configuration tool to set the node address from the network.

This service features a flag which specifies whether or not it is allowed to change the device address from the network again at a later stage. If the service is accepted, the module saves the value of this flag in nonvolatile memory; the only way to restore it again is by performing a Factory Default-reset on the Network Configuration Object (consult the general Anybus CompactCom 40 Software Design Guide for more information). This behavior is mandatory for the application to pass PROFIBUS network certification.

The module will accept new settings received via this service under the following conditions:

- The Device Address attribute (Network Configuration Object (04h)) is set to a value higher than 125.

The SSA Enabled attribute (PROFIBUS DP-V1 Object (FDh)) is set to TRUE (or not implemented).

The module is not in Data Exchange.

The module is addressed with the correct Ident Number.

No previous Set Slave Address request prevents the module from accepting the new settings.

See also...

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#) (Attribute #4, SSA Enabled)
- [Supported DP Features, p. 46](#)



It is possible to disable support for this service by implementing the SSA Enabled attribute in the PROFIBUS DP-V1 Object (FDh). In such a case, a new GSD file must be created, and fieldbus re-certification is necessary.

4.8 Parameter Read/Write with Call

4.8.1 General information

Parameter Read/Write with Call enables addressing of ADIs based on instance numbers rather than Slot and Index. This is useful if the ADI implementation is primarily designed for a linear addressing scheme as used on most other networks. It may also prove useful when using masters with limited addressing capabilities for slot 0, since such masters may otherwise have trouble accessing ADI instances 1... 255.

Unlike the standard DP-V1 Read/Write service, Parameter Read/Write with Call uses the Call application service. On the PROFIBUS telegram level, the Parameter Read with Call service request consists of a standard DP-V1 header, a Call header, and the ADI number. When received by the module, this is translated into a standard object request towards the application data object.

4.8.2 Parameter Read with Call Handling

The Parameter Read with Call service request looks as follows:

Byte #	Contents	Field Name	Value	Notes
1	DP-V1 Header	Function no.	5Fh	Indicates a DP-V1 Write service
2		Slot	00h	(must not be set to FFh)
3		Index	FFh	(fixed)
4		Length	06h	Size of telegram (Call Header + ADI number)
5	Call Header	Ext. Function no.	08h	Call service
6		(reserved)	00h	(reserved, set to zero)
7		Subindex (high)	00h	Subindex 0002h, used when reading
8		Subindex (low)	02h	
9	ADI number	ADI (high)	0000h... FFFFh	Number of the ADI which shall be read
10		ADI (low)		

Upon reception, the module translates this into a Get_Attribute request towards the Application Data Object. In the same manner, the response from the host application will be transformed into an appropriate response telegram on PROFIBUS as follows:

Parameter Read with Call response:

Byte #	Contents	Field Name	Value	Notes
1	DP-V1 Header	Function no.	5Eh	Indicates a DP-V1 Write service
2		Slot	00h	(must not be set to FFh)
3		Index	FFh	(fixed)
4		Length	06... F0h	Size of telegram (Call Header + ADI number + Data)
5	Call Header	Ext. Function no.	08h	Call service
6		(reserved)	00h	(reserved, set to zero)
7		Subindex (high)	00h	Subindex 0002h, used when reading
8		Subindex (low)	02h	
9	ADI number	ADI (high)	0000... FFFFh	Number of the ADI which shall be read
10		ADI (low)		
11...n	Data	(actual data)	-	Data returned from the host application The max value of n = 244

4.8.3 Parameter Write with Call Handling

The Parameter Write with Call service request looks as follows:

Byte #	Contents	Field Name	Value	Notes
1	DP-V1 Header	Function no.	5Fh	Indicates a DP-V1 Write service
2		Slot	00h	(must not be set to FFh)
3		Index	FFh	(fixed)
4		Length	06... F0h	Size of telegram (Call Header + ADI number + Data)
5	Call Header	Ext. Function no.	08h	Call service
6		(reserved)	00h	(reserved, set to zero)
7		Subindex (high)	00h	Subindex 0001h, used when writing
8		Subindex (low)	01h	
9	ADI number	ADI (high)	0000h... FFFFh	Number of the ADI which shall be read
10		ADI (low)		
11...n	Data	(actual data)	-	Data which will be sent to the host application The max value of n = 244

Upon reception, the module translates this into a Set_Attribute request towards the Application Data Object. In the same manner, the response from the host application will be transformed into an appropriate response telegram on PROFIBUS as follows:

Parameter Write with Call response:

Byte #	Contents	Field Name	Value	Notes
1	DP-V1 Header	Function no.	5Eh	Indicates a DP-V1 Read service
2		Slot	00h	(must not be set to FFh)
3		Index	FFh	(fixed)
4		Length	06... F0h	Size of telegram (Call Header + ADI number)
5	Call Header	Ext. Function no.	08h	Call service
6		(reserved)	00h	(reserved, set to zero)
7		Subindex (high)	00h	Subindex 0001h, used when reading
8		Subindex (low)	01h	
9	ADI number	ADI (high)	0000... FFFFh	Number of the ADI which shall be read
10		ADI (low)		

4.9 Identification & Maintenance (I&M)

4.9.1 General Information

Identification & Maintenance (I&M) provides a standard way of gathering information about an I/O device. The I&M information can be accessed by the PROFIBUS master by means of acyclic Record Data Read/Write services.

Default I&M0 Information:

IM Manufacturer ID	010Ch (HMS Industrial Networks)
IM Order ID	"ABCC 40-DPV1"
IM Serial Number	(unique serial number, set during manufacturing)
IM Hardware Revision	(Anybus hardware revision ID, set during manufacturing)
IM Software Revision	(Anybus software revision, set during manufacturing)
IM Revision Counter	(Revision counter)
IM Profile ID	F600h (Generic Device)
IM Profile Specific Type	0004h (Communication Module)
IM Version	0101h
IM Supported	001Eh (IM0..4 supported)

Optionally, the host application can customize the information for these I&M entries, or implement the support for the Get_IM_Record and Set_IM_Record commands to support all I&M record for all slots.

See also ...

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)
 - Command Details: Get_IM_Record
 - Command Details: Set_IM_Record
- [Network Configuration Object \(04h\), p. 30](#)

4.9.2 I&M Data Structures

The I&M records uses the following data structures.

Re- cord	Content	Size	Description/Comment
I&M0	Manufacturer Id	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #8 ('Vendor ID/I&M Vendor ID')
	Order Id	20 bytes	PROFIBUS DP-V1 Object (FDh), attribute #9 ('I&M Order ID')
	Serial number	16 bytes	PROFIBUS DP-V1 Object (FDh), attribute #10 ('I&M Serial number')
	Hardware revision	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #11 ('I&M Hardware revision')
	Software revision	4 bytes	PROFIBUS DP-V1 Object (FDh), attribute #12 ('I&M Software revision')
	Revision counter	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #13 ('I&M Revision counter')
	Profile Id	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #14 ('I&M Profile ID')
	Profile specific type	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #15 ('I&M Profile specific type')
I&M1	Tag Function	32 bytes	(Data of this field can only be accessed from the network by the IO Controller/Supervisor.)
	Tag Location	22 bytes	(Data of this field can only be accessed from the network by the IO Controller/Supervisor.)

Re-cord	Content	Size	Description/Comment
I&M2	Installation date	16 bytes	(Data of this field can only be accessed from the network by the IO Controller/Supervisor.)
I&M3	Descriptor	54 bytes	(Data of this field can only be accessed from the network by the IO Controller/Supervisor.)
I&M4	Signature	54 bytes	(Data of this field can only be accessed from the network by the IO Controller/Supervisor.)

See also...

- [*PROFIBUS DP-V1 Object \(FDh\), p. 36*](#)

5 Diagnostics

PROFIBUS diagnostics contains two parts: standard diagnostics and extended diagnostics (optional). The standard diagnostics are contained in the first 6 bytes of the diagnostics data telegram and, optionally, extended diagnostics can follow after that.

The application defines whether to use transparent diagnostics using the DP-V0 diagnostic object (10h), or to use the Anybus diagnostic object (02h). It is not possible to change type of diagnostics at runtime, thus it has to be defined in the indesign phase.

Extended diagnosis are produced by the Anybus module if any of the following events occur:

- Configuration mismatch: if the configuration is calculated by the module, identifier related and module status diagnosis will be sent to clarify in which of the slots the configuration is faulty.
- Application creates diagnostics using the DP-V0 diagnostic object (10h).
- Application creates diagnostics using the diagnostic object (02h).

See also ...

- [PROFIBUS DP-V0 Diagnostic Object \(10h\), p. 32](#)
- [Diagnostic Object \(02h\), p. 26](#)

5.1 Standard Diagnostics

The Standard Diagnostics are handled automatically, with the exception of the following flags:

Ext Diag Overflow

This flag can be controlled by the host application via the attribute Ext Diag Overflow (#12) in the PROFIBUS DP-V0 Diagnostic Object (10h). If the flag is set, it indicates that there is more diagnostic data than the diagnostic telegram can hold (244 bytes).

Static Diag Flag

This flag can be controlled by the host application via the attribute Static Diag (#13) in the PROFIBUS DP-V0 Diagnostic Object (10h). This flag is set if the application reports an error (only applicable for event based communication). For more information, see Anybus CompactCom 40 Software Design Guide, "Application Status Register".

5.2 Additional Diagnostics Information

5.2.1 Extended Diagnostics

The extended diagnostics can be divided into four groups.

Identifier Related	The diagnosis information is related to modules of a slave.
Channel Related	The diagnosis information informs about errors of channels within modules.
Device Related - Status Model	The diagnosis telegram contains special slot based information that will be transferred to the master.
Device Related - Alarm Model	The diagnosis telegram contains device related information defined by alarms.

6 Anybus Module Objects

6.1 General Information

This chapter specifies the Anybus Module Object implementation and how the objects correspond to the functionality in the Anybus CompactCom 40 PROFIBUS DPV1/DPV0.

Standard Objects:

- [*Anybus Object \(01h\), p. 25*](#)
- [*Diagnostic Object \(02h\), p. 26*](#)
- [*Network Object \(03h\), p. 28*](#)
- [*Network Configuration Object \(04h\), p. 30*](#)

Network Specific Objects:

- [*PROFIBUS DP-V0 Diagnostic Object \(10h\), p. 32*](#)

6.2 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*.

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	-	-	-	See the general Anybus CompactCom 40 Software Design Guide for further information.
12	LED colors	Get	struct of: UINT8 (LED1A) UINT8 (LED1B) UINT8 (LED2A) UINT8 (LED2B)	<u>Value:</u> <u>Color:</u> 01h Green 02h Red 01h Green 02h Red
13...1-6	LED status	Get	UINT8	See the general Anybus CompactCom 40 Software Design Guide for further information.

Extended

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

6.3 Diagnostic Object (02h)

Category

Extended

Object Description

This object provides a standardised way of handling host application events, alarms, and diagnostics, and is thoroughly described in the general Anybus CompactCom 40 Software Design Guide. In the case of PROFIBUS, each instance created in this object adds one Status PDU to the Extended Diagnostics.

Please note that this object cannot be used at the same time as the transparent diagnostics (DP-V0 diagnostic object (10h)).

Supported Commands

Object:	Get_Attribute
	Create
	Delete
Instance:	Get_Attribute

Object Attributes (Instance #0)

#	Name	Access	Type	Value	Description
1... 4	-	-	-		See the general Anybus CompactCom 40 Software Design Guide for further information.
11	Max no. of instances	Get	UINT16	6	One instance of type major unrecoverable and five instances of other severity types can be created.
12	Supported Functionality	Get	BITS32	1	Bit 0 = 1: Latching events are supported.

Instance Attributes (Instance #1...n)

Extended

The use of attribute #3 is optional; if not implemented, the module will use slot no. 0 (zero), and the 'Severity' and 'Event Code' attributes will be reported as application specific data. If implemented, a custom GSD file may be required.

#	Name	Access	Type	Value
1	Severity	Get	UINT8	See “Severity” below
2	Event Code	Get	UINT8	See the general Anybus CompactCom 40 Software Design Guide for further information.
3	NW specific extension	Get	Array of UINT8 <u>Element</u> 0 1 2...57	<u>Color</u> (reserved) (reserved) Application Specific Field
4	Slot	Get	UINT16	See the general Anybus CompactCom 40 Software Design Guide for further information.
5	ADI	Get	UINT16	
6	Element	Get	UINT8	
7	Bit	Get	UINT8	

See also...

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)
- [Status Diagnostics Messages, p. 55](#)

Severity

This parameter indicates the severity level of the event, and describes the kind of PROFIBUS diagnostics generated for the different levels. It also indicates whether extended diagnostics are used by the instance.

For more information, see the Anybus CompactCom 40 Software Design Guide.

Severity Levels

#	Severity	Modular Device Object not Implemented (and slot number = 0)	Modular Device Object Implemented (and slot number > 0)
00h	Minor, recoverable	Status message.	Identifier related + channel related For more information, see Additional Diagnostics Information, p. 23
10h	Minor, unrecoverable	Status message. Unrecoverable events cannot be deleted.	
20h	Major, recoverable	Status message.	
30h	Major, unrecoverable	Causes a state-shift to EXCEPTION.	
50h	Minor, latching	Diagnosis alarm.	See the Anybus CompactCom 40 Software Design Guide for more information.
60h	Major, latching		
(other)	-	(reserved for future use)	

Network Specific Error Codes (Latching Events)

In case of an unsuccessful create command for a latching event, the following network specific error codes are returned.

Error Code	Description
8	Alarm type disabled
9	Too many active alarms

6.4 Network Object (03h)

Category

Basic

Object Description

This object contains network specific data for the module. It also controls the mapping of ADIs to the process data part of the telegrams. For more information, consult the general Anybus CompactCom 40 Software Design Guide.



The order in which ADIs are mapped to Process Data is significant and must be replicated in the PROFIBUS master when setting up the network communication.

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	0005h
2	Network type string	Get	Array of CHAR	"PROFIBUS DP-V1"
3	Data format	Get	ENUM	0001h (MSB first)
4	Parameter Data support	Get	BOOL	True (This attribute indicates if the network supports acyclic data services and must not be confused with PROFIBUS Parameterization Data.)
5	Write process data size	Get	UINT16	Current write process data size (in bytes) Updated on every successful Map_ADI_Write_Area Consult the general Anybus CompactCom 40 Software Design Guide for further information.
6	Read process data size	Get	UINT16	Current read process data size (in bytes) Updated on every successful Map_ADI_Read_Area Consult the general Anybus CompactCom 40 Software Design Guide for further information.
7	Exception Information	Get	UINT8	Additional PROFIBUS specific exception information is presented here in case the Anybus module has shifted to the EXCEPTION state.

Exception Information

This attribute holds additional information when the Anybus module shifts to the EXCEPTION state.

#	Value
00h	No information.
01h	The ADI mapping resulted in too much configuration data.
02h	The configuration data attribute in the PROFIBUS DP-V1 object contains too much data.
03h	Configuration error. The 'Configuration Data' attribute does not match the actual Process Data map.
04h	Reserved.
05h	Reserved.
06h	Implementation error. Support for the Set Slave Address (SSA) telegram has been disabled ('SSA Enable' attribute, PROFIBUS DP-V1 Object (FDh) , p. 36), but no valid device address has been supplied by the application.
07h	Reserved.
08h	Reserved.
09h	Reserved.
0Ah	Reserved.
0Bh	Reserved.
0Ch	The Modular Device Object has mapped process data on slot 0.
0Dh	The Modular Device Object has mapped data on an empty slot.

6.5 Network Configuration Object (04h)

Category

Basic

Object Description

This object contains network specific configuration parameters that may be configured by the end user.

A Reset command towards this object will cause the module to revert all instance values to their factory default values.

Supported Commands

Object:	Get_Attribute
	Reset
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

The module must be assigned a unique node address (a.k.a. device address) in order to be able to communicate on the PROFIBUS network. Valid settings range from 0... 125. The node address is set either from the network (SSA service) or from the application.

Address 126 is reserved for SSA functionality, see [Set Slave Address, p. 18](#). This feature allows the device address to be set from the PROFIBUS master. Default after factory reset is 126.

#	Name	Access	Data Type	Description						
1	Name	Get	Array of CHAR	“Node address” (Multilingual, see page 31)						
2	Data type	Get	UINT8	04h (= UINT8)						
3	Number of elements	Get	UINT8	01h (one element)						
4	Descriptor	Get	UINT8	07h (read/write/shared access)						
5	Value	Get/Set	Array of UINT8	<table><tr><td><u>Value</u></td><td><u>Meaning</u></td></tr><tr><td>0... 125</td><td>Node address</td></tr><tr><td>(other)</td><td>Get node address using SSA (default)</td></tr></table> (Note that support for SSA can be globally disabled by implementing the PROFIBUS object, see PROFIBUS DP-V1 Object (FDh) , p. 36)	<u>Value</u>	<u>Meaning</u>	0... 125	Node address	(other)	Get node address using SSA (default)
<u>Value</u>	<u>Meaning</u>									
0... 125	Node address									
(other)	Get node address using SSA (default)									
6	Configured Value	Get	Array of UINT8	Holds the configured value, which will be written to attribute #5 after the module has been reset. If SSA is enabled (attribute 5 set to 126) the saved SSA address is returned.						

Multilingual String

The instance name in this object is multilingual, and is translated based on the current language settings as follows:

Instance	English	German	Spanish	Italian	French
1	IP address	IP-Adresse	Dirección IP	Indirizzo IP	Adresse IP

6.6 PROFIBUS DP-V0 Diagnostic Object (10h)

Object Description

Implementing this object makes it possible to create transparent diagnostic data, instead of using the standard diagnostic data.

For the diagnostic functionality to work correctly during the connection phase with the master, the application must write zero bytes to attribute #1 in instance #1 before setting setup complete.



It is not possible to switch between the use of transparent diagnostics and the use of standard diagnostics at runtime. The PROFIBUS DP-V0 Diagnostic Object (10h) object cannot be used at the same time as the Diagnostic Object (02h).

Supported Commands

Object:	Get_Attribute
	Set_Attribute
Instance:	Get_Attribute
	Set_Attribute

Object Attributes (Instance #0)

#	Name	Access	Type	Value
1	Name	Get	Array of CHAR	"PROFIBUS DP-V0 Diagnostic"
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h
12	Ext Diag Overflow	Get/Set	BOOL	False (Default): Ext Diag Overflow is cleared True: Ext Diag Overflow is set
13	Static Diag	Get/Set	BOOL	False (Default): Static Diag is cleared True: Static Diag is set

Instance Attributes (Instance #1)

Extended

#	Name	Access	Type	Value
1	Diagnostic Data	Get	Array of UINT8	Array of bytes, containing the diagnostic data. For more information, see "Diagnostic Data Layout" below.

Diagnostic Data Layout

Data Byte	Contents	Description
0	Extended diagnostic data flag	0 - No extended diagnostic data 1 - Extended diagnostic data is available
1 - 238	Diagnostic data	The user specific part of the diagnostic buffer

Object Specific Error Codes

Number	Name	In Response to Command
01h	Diagnostic Object (02h) in use	Set_Attribute
02h	Invalid extended diagnostic data flag value	Set_Attribute

7 Host Application Objects

7.1 General Information

The objects listed in this chapter may optionally be implemented within the host application firmware to expand the PROFIBUS implementation.

Standard Objects:

- Application Object (see Anybus CompactCom 40 Software Design Guide)
- Application Data Object (see Anybus CompactCom 40 Software Design Guide)
- [Modular Device Object \(ECh\), p. 35](#)

Network Specific Object:

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)

7.2 Modular Device Object (ECh)

Category

Extended

Object Description

This object is used to describe a modular device. 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.

Each instance of this object describes a slot in the modular device. Instance 1 corresponds to the coupler.

When mapping ADIs to process data, the application shall map the process data of each module in slot order. If the application maps the process data in any other order, the Anybus CompactCom module will enter EXCEPTION state.

For more information, see Anybus CompactCom 40 Software Design Guide, “Modular Device Object (ECh)”

Supported Commands

Object: Get_Attribute
 Get_List

Instance: -

Object Attributes (Instance #0)

#	Name	Access	Type	Value
1	Name	Get	Array of CHAR	“Modular device”
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	Number of connected modules. For limitations, see Number of slots (attribute #11)
4	Highest instance no.	Get	UINT16	Highest connected module
11	Number of slots	Get	UINT16	The maximum value according to PROFIBUS is 255 (0-254). However, due to the definition of the PROFIBUS configuration data, the maximum number of slots which can be guaranteed to work is 40. Each slot can contain between 2 and 16 bytes of configuration data. The total amount of configuration data must not exceed 244 bytes. (If not all slots have process data, the number of slots can be increased since empty slots only needs 1 byte configuration data)
12	Number of ADIs per slot	Get	UINT16	The maximum value for this attribute is 255 for PROFIBUS, since Index only may have values in the range 0-254. However, for the application to be able to send ADI specific diagnostics for the whole ADI range, this value must not exceed 64.

7.3 PROFIBUS DP-V1 Object (FDh)

Category

Basic, extended

Object Description

This object implements PROFIBUS specific settings in the host application.

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

During operation, the host application must always be able to respond to requests from the module. Respond either with the requested data or an adequate error message. Never leave a request from the module unattended.

Altering the default settings within this object may require a new GSD file, which in turn requires fieldbus recertification.

See also...

- [Technical Specification, p. 57](#)
- [Support for DP-V1 and DP-V0, p. 10](#)
- [GSD File Customization, p. 44](#)
- Guideline Information & Maintenance functions
- Anybus CompactCom Software Design Guide, "Error Codes"

Supported Commands

Object:	Get_Attribute
	Get_IM_Record
	Set_IM_Record
Instance:	Get_Attribute
	Set_Attribute

Object Attributes (Instance #0)

#	Name	Access	Type	Value
1	Name	Get	Array of CHAR	"PROFIBUS DP-V1"
2	Revision	Get	UINT8	04h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	PNO Ident Number	Get	UINT16	PNO Ident Number (default = 1815h)

Extended

#	Name	Access	Type	Comment
2	Parameterization data	Set	Array of UINT8	The module attempts to forward the Parameterization Data to this attribute during network startup. Max number of elements in array: 244
3	Expected configuration	Get/Set	Array of UINT8	This attribute is used both to specify the Expected Configuration Data, and for evaluation of the Actual Configuration Data. Max number of elements in array: 244 If this attribute doesn't match the Process Data map, or if it doesn't fit into the Configuration Data Buffer, the module will enter the EXCEPTION state (exception code 06h)
4	SSA enabled	Get	BOOL	This attribute enables/disables support for Set Slave Address <div> <div>Value:</div> <div>Meaning:</div> </div> <div> <div>True</div> <div>Enabled (default)</div> </div> <div> <div>False</div> <div>Disabled</div> </div>
5 - 7	(not used)	-	-	(Used in Anybus CompactCom 30 series)
8	Manufacturer ID	Get	UINT16	Device Manufacturer ID (default = 010Ch)
9	Order ID	Get	Array of CHAR	Device Order ID, up to 20 characters (default = "ABCC 40 -DPV1") Maximum number of elements in array: 20
10	Serial Number	Get	Array of CHAR	Serial number, up to 16 characters. If not implemented, the serial number from the Application Object (FFh), attribute #3 will be used. If this attribute is also missing, the Anybus CompactCom defaults to its production assigned serial number.
11	Hardware Revision	Get	UINT16	Hardware revision of the device. <div> <div>Value:</div> <div>Meaning:</div> </div> <div> <div>0... FFEh</div> <div>Hardware revision</div> </div> <div> <div>FFFFh</div> <div>Indicates profile specific information</div> </div> If not implemented, the module defaults to its hardware functionality ID.
12	Software Revision	Get	Struct of: CHAR (Type) UINT8 (Major) UINT8 (Minor) UINT8 (Build)	Software revision of the device. If not implemented, the module defaults to the Anybus firmware revision.
13	Revision Counter	Get	UINT16	Revision counter; a changed value of this counter marks a change of the hardware or of its parameters. If not implemented, the module defaults to 0 (zero).
14	Profile ID	Get	UINT16	Specifies the Profile ID. Default: F600h (Generic Device) The attribute does not affect the behavior of the module, since the module does not handle profiles.
15	Profile specific type	Get	UINT16	Specifies the Profile specific type. Default: 0004h (Communication Module) The attribute does not affect the behavior of the module, since the module does not handle profiles.

#	Name	Access	Type	Comment
18	IM header	Get	Array of UINT8	This header will be sent to the master together with I&M0... 4 for slot 0. The master is then required to supply this information back with each write request to these records. If not implemented, the header will be filled with zeroes. The maximum number of elements in the array is 10.
19	(not used)			

Software Revision Structure

Member	Contents		Comments
CHAR	<u>Letter</u>	<u>Meaning</u>	-
	“V”	Official release	
	“R”	Revision	
	“P”	Prototype	
	“U”	Under test (field test)	
	“T”	Test device	
UINT8	Major version Range: 0,,, 255		Functional enhancement
UINT8	Minor version Range: 0,,, 255		Bug fix
UINT8	Internal change Range: 0,,, 255		No impact on function

Command Details: Get_IM_Record

Category

Extended

Details

Command Code: 10h

Valid for: Object

Description

This command is sent to the host application when the master (Class 1 or Class 2) asks for an I&M Record other than I&M0... 4 for slot 0, and for all I&M Records for slots other than 0. If the command is rejected, the original I&M-request from the PROFIBUS master will be rejected as well.

See also

- Guideline Information & Maintenance functions.
- Command Details

Field	Comments
CmdExt[0]	I&M Record index (0... 199)
CmdExt[1]	(reserved, ignore)
Msg_Data[0]	Slot number (0... 254)
Msg_Data[1... 3]	(reserved, ignore)

- Response Details (Success)

Field	Comments
Msg_Data[0... 9]	I&M user specific header
Msg_Data[10... 63]	I&M Record; I&M parameters associated with the request

Command Details: Set_IM_Record

Category

Extended

Details

Command Code: 11h
Valid for: Object

Description

This command is sent to the host application when the master (Class 1 or Class 2) requests to update an I&M Record other than I&M0... 4 for slot 0, and for all I&M Records for slots other than 0. If the command is rejected, the original I&M-request from the PROFIBUS master will be rejected as well.

See also

- Guideline Information & Maintenance functions.
- Command Details

Field	Comments
CmdExt[0]	I&M Record index (0... 199)
CmdExt[1]	(reserved, ignore)
Msg_Data[0]	Slot number (0... 254)
Msg_Data[1... 3]	(reserved, ignore)
Msg_Data[4... 13]	I&M user specific header.
Msg_Data[14... 67]	I&M Record; I&M parameters associated with the request

- Response Details (Success)
(no data)

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. For PROFIBUS, this bit is set when any of the following conditions are fulfilled.

- Parameterization and Configuration Data has been accepted (i.e. MS0 connection established)
- An MS2 connection is open

B.2 Anybus State Machine

The table below describes how the Anybus state machine relates to the PROFIBUS network

Anybus State	Description
WAIT_PROCESS	No MS0 connection DP state = Power-On/WaitPrm/WaitCfg MS2 connection may be open
ERROR	Not used
PROCESS_ACTIVE	Master Mode = Operate DP State = DataExchange MS0 connection established MS2 connection may be open
IDLE	Master Mode = Clear DP State = DataExchange MS0 connection established MS2 connection may be open
EXCEPTION	MS0, MS1 and MS2 connections will be closed. The module will enter this state in the following cases: <ul style="list-style-type: none"> • Invalid Device Address and "SSA Enabled" = FALSE • Size of 'Configuration Data' attribute is larger than the size of the Configuration Data Buffer • Major Unrecoverable event created in Diagnostic Object • Configuration Data does not match the mapped Process Data

B.3 Watchdog Behavior (Application Stopped)

If the application watchdog expires, the module will enter the EXCEPTION state, terminate all open PROFIBUS connections (MS0, MS1 and MS2) and leave the network.

C Error Handling

C.1 Translation of Anybus Error Codes

When a DP-V1 request is received from the network, the module translates this request into an object request to the application data object. When such requests are rejected by the host application, the error code in the response is translated to DP-V1 standard as described in the table below. Note that error code 2 will always be 0 when handling a CALL request.

Anybus Error Code	Resulting DP-V1 Error Codes			
	Error Decode	Error Code 1		Error Code 2
		Error Class	Error Code	
'Unsupported Object'	DP-V1 (128)	Access	Invalid area	Same as the Anybus error code
'Unsupported Instance'		Access	Invalid index	
'Unsupported Command'		Access	Access denied	
'Invalid CmdExt0'		Access	Invalid parameter	
'Invalid CmdExt1'		Access	Invalid parameter	
'Attribute not Setable'		Access	Access denied	
'Attribute not Getable'		Access	Access denied	
'Too Much Data'		Access	Write error length	
'Not Enough Data'		Access	Write error length	
'Out of range'		Access	Invalid range	
'Value too high'		Access	Invalid range	
'Value too low'		Access	Invalid range	
'Invalid State'		Access	State conflict	
'Out of Resources'		Resource	Resource busy	
'Object Specific Error'		Application	User specific (10)	
(All other Anybus error codes)		Application	User specific (11)	

C.2 Other Errors

Error	Resulting DP-V1 Error Codes			
	Error Decode	Error Code 1		Error Code 2
		Error Class	Error Code	
ADI truncated	DP-V1 (128)	Application	User specific (12)	0
No free source IDs		Resource	Resource busy	

D GSD File Customization

D.1 General

The GSD file specifies the characteristics of the device, and is used by the PROFIBUS configuration tool when setting up the network.

HMS provides an example GSD 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 a way that invalidates the generic GSD file. In such case, a custom GSD file must be created, and fieldbus recertification is necessary.

This chapter is intended to provide a brief overview of the GSD entries that may need alteration, and how they correspond to settings within the Anybus module. Some of the entries should not be changed, and the others are divided in the same way as the objects and object attributes, into the groups Basic and Extended.

For further information, consult the Specification for PROFIBUS Device Description and Device Integration Volume 1: GSD (order. no. 2.122).



The user is expected to have sufficient knowledge in the PROFIBUS networking system to understand the concepts involved when performing the changes specified in this chapter. In case of uncertainties, send the customized GSD file to HMS for verification.

D.2 Device Identification

D.2.1 General

By default, the module will appear as a generic Anybus implementation ("Anybus CompactCom 40 DPV1") from HMS Industrial Networks (PROFIBUS ident no. 1815h).

However, the identity of the module can be customized to appear as a vendor specific implementation by creating a custom GSD file and implementing the attribute PNO Ident Number (#1) in the [PROFIBUS DP-V1 Object \(FDh\)](#), p. 36.

Contact PNO to obtain a unique Ident Number.

D.2.2 GSD File Entries

```
; Device identification
Vendor_Name      = "<vendor>"
Model_Name       = "<product>"
Revision         = "<prod_rev>"
Ident_Number     = <ident_no>
Protocol_Ident   = 0                ; DP protocol
Station_Type     = 0                ; Slave device
FMS_supp         = 0                ; FMS not supported
Slave_Family     = 0                ; General device
Hardware_Release = "Version <hw_rev>"
Software_Release = "Version <sw_rev>"
```

Basic

Setting	Description
<vendor>	Vendor name as text (e.g. "HMS Industrial Networks")
<product>	Product name as text (e.g. "Anybus CompactCom DPV1")
<prod_rev>	Product revision (major.minor) (e.g. "1.01")
<ident_no>	PNO Ident Number in HEX. Written as 0xNNNN, where NNNN is the hexadecimal value.
<hw_rev>	Hardware revision (major.minor) (e.g. "Version 1.00")
<sw_rev>	Software revision (major.minor) (e.g. "Version 1.00")

D.2.3 Related Information

- [Device Identity, p. 12](#)
- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#), attribute #1

D.3 Supported Hardware Features

D.3.1 General

Do not change the standard settings.

D.3.2 GSD File Entries

```
; Supported hardware features
Redundancy = 0 ; not supported
Repeater_Ctrl_Sig = 2 ; TTL
24V_Pins = 0 ; not connected
Implementation_Type = "NP40"
```

D.4 Supported DP Features

D.4.1 GSD File Entries

```
; Supported DP features
Freeze_Mode_supp = 1
Sync_Mode_supp = 1
Auto_Baud_supp = 1
Set_Slave_Add_supp = <SSA>
Fail_Safe = 1
```

Extended

Setting	Description
<SSA>	This value must be set to match the 'SSA enable'-attribute in the PROFIBUS DP-V1 Object (FDh) , p. 36: 0: 'SSA enabled'-attribute set to FALSE 1: 'SSA enabled'-attribute set to TRUE (or not implemented)

D.4.2 Related Information

- [Set Slave Address, p. 18](#)
- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)

D.5 Supported Baud Rates

D.5.1 General

Do not change the standard settings.

D.5.2 GSD File Entries

```
; Supported baud rates
9.6_supp = 1
19.2_supp = 1
45.45_supp = 1
93.75_supp = 1
187.5_supp = 1
500_supp = 1
1.5M_supp = 1
3M_supp = 1
6M_supp = 1
12M_supp = 1
```


D.6 Maximum Responder Time for Supported Baud Rates

D.6.1 General

Do not change the standard settings.

D.6.2 GSD File Entries

```
; Maximum responder time for supported baud rates
MaxTsdr_9.6 = 15
MaxTsdr_19.2 = 15
MaxTsdr_45.45 = 15
MaxTsdr_93.75 = 15
MaxTsdr_187.5 = 15
MaxTsdr_500 = 15
MaxTsdr_1.5M = 25
MaxTsdr_3M = 50
MaxTsdr_6M = 100
MaxTsdr_12M = 200
```

D.7 Maximum Polling Frequency

D.7.1 General

Use the GSD file default value (1)

D.7.2 GSD File Entries

```
; Maximum polling frequency
Min_Slave_Intervall = 1 ; 100 µs
```

D.8 I/O Related Keywords

D.8.1 GSD File Entries

```
; I/O related keywords
Modular_Station = 1
Max_Module = <module>
Max_Input_Len = <input>
Max_Output_Len = <output>
Max_Data_Len = <total>
Modul_Offset = 1
```

Basic

Setting	Unit	Description
<module>	bytes	
<input>	bytes	
<output>	bytes	
<total>	bytes	

D.9 Definition of Modules

D.9.1 General

These parameters need to be altered if customizing module names, if the Configuration Data attribute

(*PROFIBUS DP-V1 Object (FDh)*, p. 36) has been implemented.

D.9.2 GSD File Entries

```
; Definition of modules
Module = "<name>" <identifier>
<module_id>
EndModule
```

Basic

Setting	Description
<name>	Name of module
<identifier>	Configuration Identifier; hexadecimal value (written as NNh where NN is the hexadecimal value) specifying the properties of the module (see below).
<module_id>	Decimal number, must be unique for each module. Max. 16 bits when converted to hexadecimal.

D.9.3 Identifier Explanation

The identifiers will differ slightly, depending on whether the Modular Device Object has been implemented or not. Both possibilities are exemplified below. Big endian format is used throughout.

Modular Device Object Not Implemented

If the modular device object is NOT implemented in the application, every slot in the configuration is seen as an ADI. The module will calculate the expected configuration as follows:

First Configuration Identifier for Each Slot

Header	
Bit 0 - 3:	Length of vendor specific data 0000 = No data 0001 - 1110 = 1 - 14 bytes 1111 = No data
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O data specification follows 01 = One byte I/O data specification for input follows 10 = One byte I/O data specification for output follows 11 = Two bytes I/O data specification follows (not valid)

I/O Data Specification

Bit 0 - 5:	Data length (max 64 elements) 000000 = 1 byte/word 111111 = 64 bytes/words
Bit 6:	Size 0 = Byte 1 = Word
Bit 7:	Data consistency 0 = Byte/word 1 = Whole length (The module will set this bit to 1 if the module only contains one ADI or only one ADI input and output)

Vendor Specific Data

Byte 0:	More cfg follows
Bit 7:	
Byte 1 - 2:	ADI number

Following Configuration Identifiers (If More than 64 Elements in One ADI)

Note: the module will start a new configuration identifier when: size changes, total data length exceeds 64 elements or when the ADI index is full.

Header

Bit 0 - 3:	Length of vendor specific data 0000 = No data 0001 - 1110 = 1 - 14 bytes 1111 = No data
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O data specification follows 01 = One byte I/O data specification for input follows 10 = One byte I/O data specification for output follows 11 = Two bytes I/O data specification follows (not valid)

I/O Data Specification

Bit 0 - 5:	Data length (max 64 elements) 000000 = 1 byte/word 111111 = 64 bytes/words
Bit 6:	Size 0 = Byte 1 = Word
Bit 7:	Data consistency 0 = Byte/word 1 = Whole length (The module will set this bit to 1 if the module only contains one ADI or only one ADI input and output)

Vendor Specific Data

Byte 0:	More cfg follows
Bit 7:	

Example

The application maps the following ADI's to process data:

ADI Number	Direction	Data Type	Number of Elements
300	Input	UINT8	2
600	Output	UINT8	66

The expected configuration will then look as follows:

```
[0x43 0x81 0x00 0x2C 0x01] (Slot 1)
[0x83 0xBF 0x80 0x58 0x02] [0x81 0x81 0x00] (Slot 2)
```

GSD Entries

```
Modular_Station = 1
Modul_Offset    = 1
DPV1_Data_Types = 0
Chk_Cfg_Mode    = 0
Max_Module      = 48          (244/5=48 min conf data to describe one ADI)
; Definition of modules
Module = "ADI 300" 0x43,0x81,0x00,0x2C,0x01;
1
EndModule
;
Module = "ADI 600" 0x83,0xBF,0x80,0x58,0x02,0x81,0x81,0x00;
2
EndModule
```

Modular Device Object Implemented

If the modular device object is implemented in the application, every slot in the configuration is seen as a module containing one or several ADI's. Note that it is not allowed to have any ADIs mapped on the coupler (slot 0).

The module will calculate the expected configuration as follows:

First Configuration Identifier for a Specific Module with I/O Data

Header	
Bit 0 - 3:	Length of vendor specific data 0000 = No data 0001 - 1110 = 1 - 14 bytes 1111 = No data
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O data specification follows 01 = One byte I/O data specification for input follows 10 = One byte I/O data specification for output follows 11 = Two bytes I/O data specification follows

I/O Data Specification	
Bit 0 - 5:	Data length (max 64 elements) 000000 = 1 byte/word 111111 = 64 bytes/words
Bit 6:	Size 0 = Byte 1 = Word
Bit 7:	Data consistency 0 = Byte/word 1 = Whole length (The module will set this bit to 1 if the module only contains one ADI or only one ADI input and output)
Vendor Specific Data	
Byte 0: Bit 0 -3: Bit 7:	No. of output ADI index follows More config for this slot follows
Byte 1 - 2:	Module ID
Byte 3 - 13:	ADI index in the module, i.e. which ADIs within a slot that are mapped to this module (1 byte for each ADI, outputs first)

Following Configuration Identifiers

The module will start a new configuration identifier when: size changes, total data length exceeds 64 elements or when the ADI index is full.

Header	
Bit 0 - 3:	Length of vendor specific data 0000 = No data 0001 - 1110 = 1 - 14 bytes 1111 = No data
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O data specification follows 01 = One byte I/O data specification for input follows 10 = One byte I/O data specification for output follows 11 = Two bytes I/O data specification follows
I/O Data Specification	
Bit 0 - 5:	Data length (max 64 elements) 000000 = 1 byte/word 111111 = 64 bytes/words
Bit 6:	Size 0 = Byte 1 = Word
Bit 7:	Data consistency 0 = Byte/word 1 = Whole length (The module will set this bit to 1 if the module only contains one ADI or only one ADI input and output)
Vendor Specific Data	
Byte 0: Bit 0 -3: Bit 7:	No. of output ADI index follows More config for this slot follows
Byte 1 - 13:	ADI index in the module, i.e. which ADIs within a slot that are mapped to this module (1 byte for each ADI, outputs first)

Configuration Identifier for a Specific Module Without I/O Data

Header	
Bit 0 - 3:	Length of vendor specific data 0003 = 3 bytes
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O
Vendor Specific Data	
Byte 0:	Reserved
Byte 1 - 2:	Module ID

Configuration Identifier for an Empty Slot

Header	
Bit 0 - 3:	Length of vendor specific data 0000 = No data
Bit 4 - 5:	Special format (00)
Bit 6 - 7:	Input/output 00 = No I/O

Example

The application maps the following ADI's to process data:

Slot	Module	Module ID	ADI Range
0	Coupler	0x00000001	1 - 50
1	No 1 (module with input data)	0x00000002	51 - 100
2	No 2 (module without I/O)	0x00000003	101 - 150
3	Empty Slot	0x00000000	151 - 200
4	No 4 (Module with both input and output data)	0x00000004	201 - 250

The application maps the following ADI's to process data:

ADI Number	Direction	Data Type	Number of Elements
51	Input	UINT8	2
210	Output	UINT16	1
211	Output	UINT16	1
220	Input	UINT16	1

The expected configuration will then look as follows:

```
[0x44 0x81 0x01 0x02 0x00 0x00] (Slot 1 - ADI 51)
[0x03 0x00 0x03 0x00] (Slot 2 - Module without I/O data)
[0x00] (Slot 3 - empty slot)
[0xC6 0x41 0xC0 0x02 0x04 0x00 0x09 0x0A 0x13] (Slot 4 - ADI 210, 211, 220)
```

GSD Entries

```
Modular_Station = 1
Modul_Offset    = 1
DPV1_Data_Types = 0
Chk_Cfg_Mode    = 0
Max_Module      = 40      Number of slots in Modular object
; Definition of modules
```

```

Module = "No 1" 0x44,0x81,0x01,0x02,0x00,0x00;
1
EndModule
;
Module = "No 2" 0x03,0x00,0x03,0x00;
2
EndModule
;
Module = "Empty slot" 0x00;
3
EndModule
;
Module = "No 4" 0xC6,0x41,0xC0,0x02,0x04,0x00,0x09,0x0A,0x13;
4
EndModule
;

```

D.10 Parameterization Related Keywords

D.10.1 General

These parameters generally only need to be altered in advanced implementations which requires the use of User Parameterization Data. The details about such implementations are beyond the scope of this document and requires in-depth knowledge in the PROFIBUS networking system.

D.10.2 GSD File Entries

```

; Parameterization related keywords
Max_User_Prm_Data_Len = <up_len>
Ext_User_Prm_Data_Const(0) = <up_data>

```

Extended

Setting	Unit	Description
<up_len>	bytes	Size of User Parameterization Data
<up_data>	-	Actual User Parameterization Data as hexadecimal values. The first three bytes are fixed and should be set to C0h, 00h, 00h.

D.10.3 Related Information

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)

D.11 Diagnostic Related Keywords

D.11.1 GSD File Entries

```
; Diagnostic related keywords
Max_Diag_Data_Len = <diag_len>
```

Basic

Setting	Unit	Description
<diag_len>	bytes	

D.11.2 Related Information

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)
- [Diagnostic Object \(02h\), p. 26](#)
- [Diagnostics, p. 23](#)

D.12 Identification & Maintenance Related Keywords

D.12.1 GSD File Entries

```
; Identification & Maintenance related keywords
Ident_Maintenance_supp = <1>
```

Basic

Setting	Description
<supp>	1: I&M supported

D.12.2 Related Information

- [PROFIBUS DP-V1 Object \(FDh\), p. 36](#)
- [Device Identity, p. 12](#)
- [Network Configuration Object \(04h\), p. 30](#)

D.13 Status Diagnostics Messages

D.13.1 General

These settings may need alteration when the “NW specific extension” of the Diagnostic Object is used. It is generally recommended to remove diagnostic codes which are not used by the implementation (e.g. remove “Value (48) = ‘Voltage’” if this code is not applicable for the end product).

D.13.2 GSD File Entries

```
;Status diagnostic messages
Unit_Diag_Area=16-17
Value(0) = "Status not changed"
Value(1) = "Status appears"
Value(2) = "Status disappears"
Unit_Diag_Area_End

Unit_Diag_Area = <start>-<end>
Value(<val>) = "<text>"
Value(<val>) = "<text>"
Value(<val>) = "<text>"
...
Value(<val>) = "<text>"
Unit_Diag_Area_End

Unit_Diag_Area = <start>-<end>
Value(<val>) = "<text>"
Value(<val>) = "<text>"
Value(<val>) = "<text>"
...
Value(<val>) = "<text>"
Unit_Diag_Area_End

Unit_Diag_Area = <start>-<end>
Value(<val>) = "<text>"
Value(<val>) = "<text>"
Value(<val>) = "<text>"
...
Value(<val>) = "<text>"
Unit_Diag_Area_End
```

Extended

Setting	Description
<start>	These settings specify the bit range to associated with <val> and <text>.
<end>	
<val>	Value
<text>	String associated with the value of <val>

D.13.3 Related Information

- [Diagnostic Object \(02h\), p. 26](#)
- [Diagnostics, p. 23](#)

D.14 DP-V1 Related Keywords

D.14.1 GSD File Entries

```
; DPV1 related keywords
DPV1_Slave = 1
Check_Cfg_Mode = 1
C1_Read_Write_Supp = 1
C1_Max_Data_Len = <C1_L>
C1_Response_Timeout = <C1_T>
C2_Read_Write_Supp = 1
C2_Max_Data_Len = <C2_L>
C2_Response_Timeout = <C1_T>
C2_Max_Count_Channels = 1
Max_Initiate_PDU_Length = 52
```

Basic

Setting	Unit	Description
<C1_L>	bytes	
<C1_T>	10 ms	
<C2_L>	bytes	1: I&M supported

D.15 Alarm Related Keywords

D.15.1 General

Do not change the standard settings.

The application cannot influence these settings. Changing the keywords may result in errors when making a Conformance test.

D.15.2 GSD File Entries

```
; Alarm
Extra_Alarm_SAP_supp           = 1    ; Do not change
Alarm_Sequence_Mode_Count     = 32    ; Max. allowed value
Alarm_Type_Mode_supp          = 1

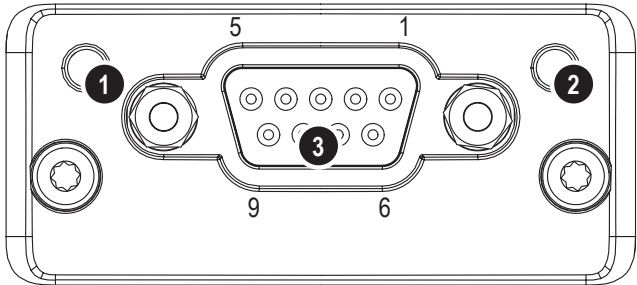
Diagnostic_Alarm_supp          = 1
Process_Alarm_supp            = 0
Pull_Plug_Alarm_supp          = 0
Status_Alarm_supp             = 0
Update_Alarm_supp             = 0
Manufacturer_Specific_Alarm_supp = 0

Diagnostic_Alarm_required      = 0
Process_Alarm_required        = 0
Pull_Plug_Alarm_required      = 0
Status_Alarm_required         = 0
Update_Alarm_required         = 0
Manufacturer_Specific_Alarm_required = 0
```

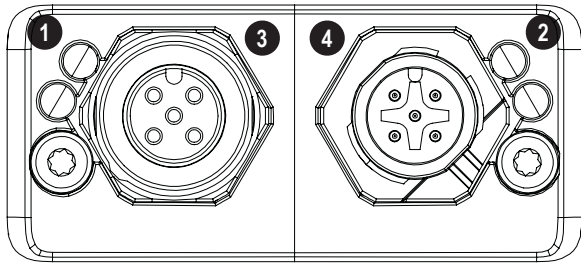
E Technical Specification

E.1 Front View

E.1.1 Front View, PROFIBUS Connector

#	Item	
1	Operation Mode	
2	Status	
3	PROFIBUS Connector, 9-pin female D-Sub	

E.1.2 Front View, M12 Connectors

#	Item	
1	Network Status LED	
2	Module Status LED	
3	M12 Female Connector	
4	M12 Male Connector	

E.1.3 Operation Mode

LED State	Indication	Comments
Off	Not online / No power	-
Green	Online, data exchange	-
Flashing Green	Online, clear	-
Flashing Red (1 flash)	Parameterization error	See Parameterization Data Handling, p. 14
Flashing Red (2 flashes)	PROFIBUS Configuration error	See Configuration Data Handling, p. 15

E.1.4 Status

LED State	Indication	Comments
Off	Not initialized	Anybus state = SETUP or NW_INIT
Green	Initialized	Anybus module has left the NW_INIT state
Flashing Green	Initialized, diagnostic event (s) present	Extended diagnostic bit is set
Red	Exception error	Anybus state = EXCEPTION

E.1.5 PROFIBUS Connector (DB9F)

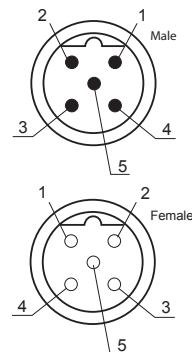
The current drawn from this pin will affect the total power consumption. To simplify development, the output supplies up to 60 mA when operated in room temperature (20 - 22 °C), which is sufficient to power e.g. master simulators etc. During normal operating conditions (or higher temperatures), i.e. in an industrial environment, the specified max. current for this output is 10 mA. See also [Power Consumption, p. 59](#).

Pin	Signal	Description
1	-	-
2	-	-
3	B Line	Positive RxD/TxD, RS485 level
4	RTS	Request to send
5	GND Bus	Ground (isolated)
6	+5 V Bus Output	+5 V termination power (isolated, short-circuit protected)
7	-	-
8	A Line	Negative RxD/TxD, RS485 level
9	-	-
Housing	Cable Shield	Internally connected to the Anybus protective earth via cable shield filters according to the PROFIBUS standard.

E.1.6 M12 Connectors, Code B

The female M12 connector is used when modules are used in a daisy-chain topology.

Pin	Name (Male/Female)	Male	Female
1	NC / P5V	Not connected	Power Supply, 5V DC
2	V+	Receive/Transmit negative	Receive/Transmit negative
3	NC / DGND	Not Connected	Power ground, 0 V
4	RxD/TxD-P (B) / RxD/TxD-P (B)	Receive/Transmit positive	Receive/Transmit positive
5 (Thread)	Shield	Shield	Shield



E.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 general Anybus CompactCom M40 Hardware Design Guide.

HMS Industrial Networks AB does not guarantee proper EMC behaviour unless these FE requirements are fulfilled.

E.3 Power Supply

E.3.1 Supply Voltage

The module requires a regulated 3.3V power source as specified in the general Anybus CompactCom M40 Hardware Design Guide.

E.3.2 Power Consumption

The Anybus CompactCom 40 PROFIBUS DPV1/DPV0 is designed to fulfil the requirements of a Class A module. For more information about the power consumption classification used on the Anybus CompactCom platform, consult the general Anybus CompactCom Hardware Design Guide.

At 12 Mbit the current hardware design consumes up to 200 mA. This value is valid under the condition that no current is being drawn from bus connector pin 6 (+5 V termination power; see [PROFIBUS Connector \(DB9F\)](#), p. 58



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.

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. Note however that in any case, the Anybus CompactCom 40 PROFIBUS DPV1/DPV0 will remain as a Class A module.

E.4 Environmental Specification

Consult the Anybus CompactCom Hardware M40 Design Guide for further information.

E.5 EMC Compliance

Consult the Anybus CompactCom Hardware M40 Design Guide for further information.

F Timing & Performance

F.1 General Information

This chapter specifies timing and performance parameters that are verified and documented for the Anybus CompactCom 40 PROFIBUS DPV1/DPV0.

The following timing aspects are measured:

Category	Parameters	Page
Startup Delay	T1, T2	60
NW_INIT Handling	T100	60
Event Based WrMsg Busy Time	T103	60
Event Based Process Data Delay	T101, T102	61

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

F.2 Internal Timing

F.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 PROFIBUS DPV1/DPV0 module generates the first application interrupt (parallel mode)	7.1	ms
T2	The Anybus CompactCom 40 PROFIBUS DPV1/DPV0 module is able to receive and handle the first application telegram (serial mode)	7.1	ms

F.2.2 NW_INIT Handling

This test measures the time required by the Anybus CompactCom 40 PROFIBUS DPV1/DPV0 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 CompactCom commands that the application can handle	1

Parameter	Description	Communication	Max.	Unit.
T100	NW_INIT handling	Event based modes	14.6	ms

F.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	2.8	7.2	µs

F.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 in two different ways, depending on network type.

- For software stack based cyclic/pollled networks, it is defined as the time from when the module exchanges write process data buffers, to when the first bit of the new process data frame is sent out on the net work.
- For COS (Change Of State) networks, it 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.

Parameter	Description	Measured Time	Unit	Comment
T101	Read process data delay	380	ns	Time from last input bit received until nIRQ goes active on the chip
T102	Write process data delay	1.1	µs	Time from when the Anybus Compact-Com module exchanges write process data buffers, to when the first bit of the new process data frame is sent out on the network interface towards the PHY.

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 PROFIBUS DPV1/DPV0, 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.



This documentation only deals with differences between the 30-series and the 40-series. For a description of new and enhanced functionality in the Anybus CompactCom 40-series, please consult our support pages, where you can find all documentation.

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. 1 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 Industrial Networks AB has developed a separate brick version in the 40-series, that can be used for migration. This product, B40-2, shares dimensions etc. with the B30. Please contact HMS Industrial Networks AB for more information on the Anybus CompactCom B40-2.

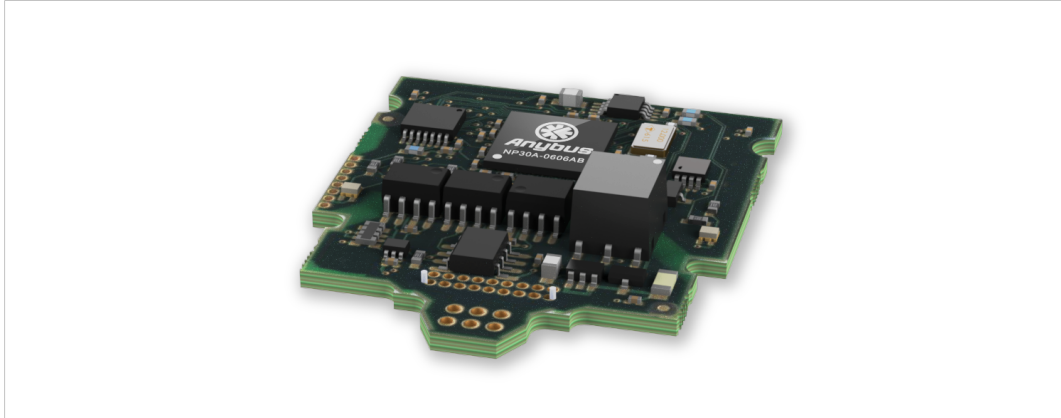


Fig. 2 Anybus CompactCom B30

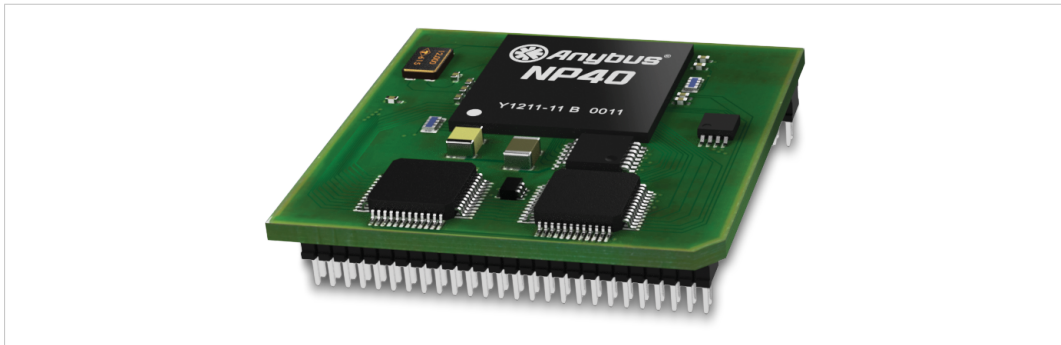


Fig. 3 Anybus CompactCom B40-1 (not for migration)

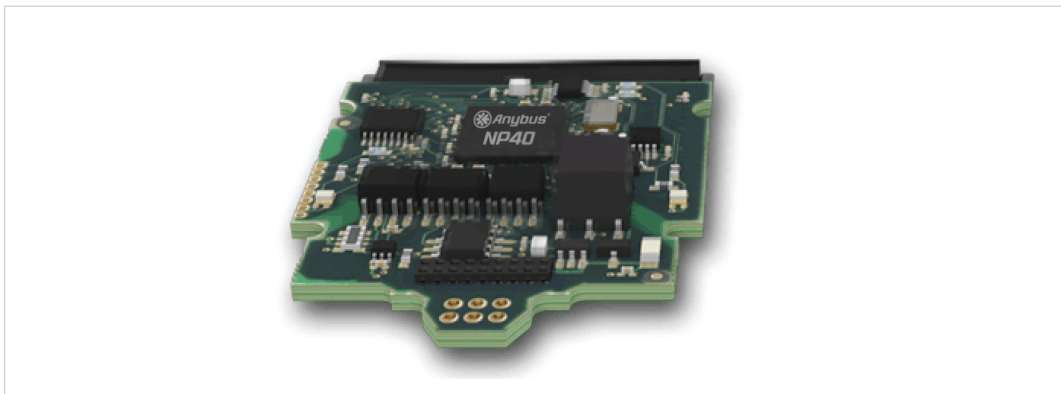


Fig. 4 Anybus CompactCom B40-2

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 $V_{IH}=V_{DD}+0,2\text{ V}$, and for the 40-series as $V_{IH}=3.45\text{ V}$. Make sure that you do not exceed 3.45V for a logic high level.

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.



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 “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 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 .. <ul style="list-style-type: none"> GIP[0..1]/LED3[A..B], p. 66 GOP[0..1]/LED4[A..B], p. 66

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

G.4 Network Specific — PROFIBUS

G.4.1 Additional Diagnostic Object (05h)

Object removed in the 40-series. To create diagnostics, use Diagnostic Object (02h).

Another option is to use the PROFIBUS DP-V0 Diagnostic Object (10h) where diagnostics can be sent transparently from the host application to the network.

If you use the Additional Diagnostic Object you need to update your software implementation.

G.4.2 Network PROFIBUS DP-V1 Object (0Bh)

Object removed in the 40-series, i.e. commands Map_ADI_Specified_Write_Area and Map_ADI_Specified_Read_Area are not supported.

G.4.3 PROFIBUS DP-V1 Object (FDh)

Attribute	30-series	40-series	Change/Action/Comment
#1, PNO Ident Number	1811h	1815h	If the attribute is implemented in the host application, it overrides the default value and there is no difference between the 30-series and the 40-series. If the attribute is not implemented, the default value is used.
#2, Parameterization Data	See Comment		In the first 10 bytes, only the Parameter Struct bit (bit 3) is copied to this attribute in the 40-series. All other bits are set to 0. In the 30-series all information in the first 10 bytes were copied.
#5, Size of Identifier Related Diagnostics	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. Nothing needs to be changed.
#6, Buffer Mode	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. No buffer modes needed in the 40-series since maximum sizes for all buffers are supported. Nothing needs to be changed.
#7, Alarm Settings	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. Only Diagnostic Alarms supported.
#16, I&M Version	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. The host application cannot influence the I&M version implemented by the Anybus CompactCom.
#17, I&M Supported	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. The host application cannot influence the I&M version supported by the Anybus CompactCom.
#19, Check Config Behavior	Available	Removed	Attribute removed in the 40-series. The Anybus CompactCom will never request this attribute. The 40-series depends on CheckCfgMode (often configurable in the PROFIBUS master) in default mode. Either the expected and actual configuration must match exactly or can differ as long as the expected input and output sizes are equal or larger than the actual sizes.

G.4.4 Network Configuration Object (04h)

The following attributes are removed in the 40-series. The Anybus CompactCom will never request these attributes. It is only possible to set these values via the network (I&M1-4) – end user configuration.

- Instance #3, Function Tag
- Instance #4, Location Tag
- Instance #5, Installation Date
- Instance #6, Description

G.4.5 GSD file (PROFIBUS configuration file used by engineering tool)

Implementation Type	If the keyword "Implementation Type" is present in the GSD-file (optional keyword), the value for the 30-series shall be "NP30" and the value for the 40-series shall be "NP40".
Length Related Keywords	<p>The following keywords are possible to set to maximum values if needed in the 40-series. In the 30-series the maximum lengths were dependent of the buffer mode setting.</p> <ul style="list-style-type: none">• Max_Input_Len• Max_Output_Len• Max_Data_Len• Max_User_Prm_Data_Len• Max_Diag_Data_Len

This page intentionally left blank

