

Network Guide

Anybus[®] CompactCom 40 PROFIBUS DP-V0/DP-V1

Doc.Id. HMSI-27-210
Rev. 1.30

Important User Information

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

The reader of this document is expected to be familiar with high level software design, and communication systems in general. The use of advanced PROFIBUS-specific functionality may require in-depth knowledge in PROFIBUS networking internal and/or information from the official PROFIBUS specifications. In such cases, the people 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.

Liability

Every care has been taken in the preparation of this manual. 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 US and other countries.

Trademark Acknowledgements

Anybus® is a registered trademark of HMS Industrial Networks AB. All other trademarks are the property of their respective holders.

Warning: This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

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

Anybus CompactCom 40 PROFIBUS DP-V0/DP-V1 Network Guide

Rev 1.30

Copyright© HMS Industrial Networks AB

Nov 2015 Doc Id HMSI-27-210

Table of Contents

Preface	About This Document	
	Related Documents	6
	Document History	6
	Conventions & Terminology	7
	Abbreviations	7
	Support	7
 Chapter 1	 About the Anybus CompactCom 40 PROFIBUS	
	General	8
	Features	8
 Chapter 2	 Fieldbus Conformance and Certification	
	Fieldbus Conformance Notes	9
	Certification	9
 Chapter 3	 Basic Operation	
	General Information	10
	<i>Software Requirements</i>	10
	<i>Support for DP-V1 and DP-V0</i>	10
	<i>Electronic Data Sheet (GSD)</i>	10
	Communication Settings	11
	Device Identity	11
	Data Exchange	12
	<i>Application Data Instances (ADIs)</i>	12
	<i>Process Data</i>	13
	Parameterization Data Handling	14
	<i>General Information</i>	14
	<i>Validation</i>	14
	Configuration Data Handling	15
	<i>General Information</i>	15
	<i>Validation</i>	15
	Set Slave Address	17
	Parameter Read/Write with Call	18
	<i>General Information</i>	18
	<i>Parameter Read with Call Handling</i>	18
	<i>Parameter Write with Call Handling</i>	19
	Identification & Maintenance (I&M)	20
	<i>General Information</i>	20
	<i>I&M Data Structures</i>	21

Chapter 4	Diagnostics	
	Standard Diagnostics	22
	Additional Diagnostics Information.....	22
	<i>Extended Diagnostics</i>	22
Chapter 5	Anybus Module Objects	
	General Information	23
	Anybus Object (01h).....	24
	Diagnostic Object (02h)	25
	<i>Severity</i>	26
	Network Object (03h).....	27
	Network Configuration Object (04h).....	29
	PROFIBUS DP-V0 Diagnostic Object (10h).....	31
	<i>Diagnostic Data Layout</i>	31
	<i>Object Specific Error Codes</i>	32
Chapter 6	Host Application Objects	
	General Information	33
	Modular Device Object (ECh).....	34
	PROFIBUS DP-V1 Object (FDh).....	35
Appendix A	Categorization of Functionality	
	Basic.....	41
	Extended.....	41
Appendix B	Implementation Details	
	SUP-Bit Definition.....	42
	Anybus State Machine	42
	Watchdog Behavior (Application Stopped).....	42
Appendix C	Error Handling	

Appendix D GSD File Customization

General.....	44
Device Identification.....	45
Supported Hardware Features.....	46
Supported DP Features.....	46
Supported Baud Rates.....	47
Maximum Responder Time for Supported Baud Rates.....	48
Maximum Polling Frequency.....	48
I/O-related Keywords.....	49
Definition of Modules.....	50
Parameterization-related Keywords.....	55
Diagnostic-related Keywords.....	56
Identification & Maintenance-related Keywords.....	57
Status Diagnostic Messages.....	58
DP-V1 related Keywords.....	59
Alarm-related Keywords.....	60

Appendix E Timing & Performance

General Information.....	61
Internal Timing.....	61
<i>Startup Delay</i>	61
<i>NW_INIT Handling</i>	61
<i>Event Based WrMsg Busy Time</i>	62
<i>Event Based Process Data Delay</i>	62

Appendix F Technical Specification

Front View.....	63
Protective Earth (PE) Requirements.....	64
Power Supply.....	64
Environmental Specification.....	64
EMC Compliance.....	64

P. About This Document

For more information, documentation etc., please visit the HMS website, ‘www.anybus.com’.

P.1 Related Documents

Anybus CompactCom 40 Software Design Guide	HMS
Anybus CompactCom M40 Hardware Design Guide	HMS
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

P.2 Document History

Summary of Recent Changes (1.20... 1.30)

Change	Page(s)
Moved Front View to Technical Specification	63
Removed distinction between advanced and extended	
Attribute #19 in Anybus object is not used	24
Recertification advised, though not mandatory	9
Updated description of instance attribute #11 in Modular Device Object (ECh)	34
Corrected and clarified section D9 (GSD file customization, definition of modules)	50
Updated attribute #10 in the PROFINET DP-VI object (FDh)	36

Revision List

Revision	Date	Author	Chapter(s)	Description
1.00	2014-03-25	KeL	All	First official revision
1.10	2014-04-28	KeL	1, 3	Added information on DP-V0
1.20	2014-08-11	KeL, KaD	1, 3, D, E	Major updates
1.30	2015-11-13	KeL	2, 5, 6, D, F	Minor updates

P.3 Conventions & Terminology

The following conventions are used throughout this manual:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The terms ‘Anybus’ or ‘module’ refers to the Anybus CompactCom 40 module.
- The terms ‘host’ or ‘host application’ refers to the device that hosts the Anybus module.
- Hexadecimal values are either written in the format NNNNh or the format 0xNNNN, where NNNN is the hexadecimal value.

P.4 Abbreviations

Abbreviation	Explanation
ACK	Acknowledge
IM, I&M	Identification and Maintenance
LED	Light Emitting Diode
LSB	Least Significant Byte
MS0	Communication between a DP class 1 or class 2 master and all related slaves
MS1	Communication between a DP-V1 class 1 master and a slave
MS2	Communication between a DP-V1 class 2 master and a slave
MSB	Most Significant Byte
N/A	Not Applicable
NAK	Negative Acknowledge
PDU	Protocol Data Unit
PI	PROFIBUS International
PNO	PROFIBUS Nutzerorganization e.V.
SSA	Set Slave Address

P.5 Support

For general contact information and where to find support, please refer to the contact and support pages at www.anybus.com.

1. About the Anybus CompactCom 40 PROFIBUS

1.1 General

The Anybus CompactCom 40 PROFIBUS communication module provides instant PROFIBUS 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 active modules defined in the Anybus CompactCom 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 take advantage of advanced network specific functionality, a certain degree of dedicated software support may be necessary.

1.2 Features

- Supports PROFIBUS DP-V1 and DP-V0
- PROFIBUS connector (9-pin female D-Sub)
- Automatic baud rate detection
- Generic and PROFIBUS specific diagnostic support
- User Parameterization Data support
- Set Slave Address support
- ADI access via DP-V1 read/write services
- Up to 488 bytes of I/O data (244 bytes in each direction)
- Device identity customization
- Generic GSD file provided
- Support for Modular Device Mode

2. Fieldbus Conformance and Certification

2.1 Fieldbus Conformance Notes

- Using the GSD file supplied by HMS, 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.

2.2 Certification

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

1. 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.

2. 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” on page 17 for more information.

3. 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.).

4. 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 about that, see “Configuration Data Handling” on page 15.

3. Basic Operation

3.1 General Information

3.1.1 Software Requirements

No additional network support code needs to be written in order to support the Anybus CompactCom 40 PROFIBUS, 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” on page 59, ‘C1_Response_Timeout’), 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) 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.

3.1.2 Support for DP-V1 and DP-V0

The Anybus CompactCom 40 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.

3.1.3 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 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” on page 9
- “GSD File Customization” on page 44

3.2 Communication Settings

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

In this case, this includes...

- **Node Address**

See also...

- “Network Configuration Object (04h)” on page 29

- **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

3.3 Device Identity

By default, the Anybus module appears as a generic HMS 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 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 ‘Get_IM_Record’- and ‘Set_IM_Record’-commands (PROFIBUS DP-V1 Object (FDh)) to support all I&M records for all slots.

See also...

- PROFIBUS Profile Guidelines Part 1: Identification & Maintenance Functions
- “Network Configuration Object (04h)” on page 29
- “PROFIBUS DP-V1 Object (FDh)” on page 35, (Attribute #1, ‘PNO Ident Number’)
- “Command Details: Get_IM_Record” on page 39
- “Command Details: Set_IM_Record” on page 40
- “Device Identification” on page 45
- “Identification & Maintenance-related Keywords” on page 57

3.4 Data Exchange

3.4.1 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)” on page 34. If the Modular Device Object is not implemented, ADIs are mapped to slots and indexes as follows:

Correlation:

$$\begin{aligned} \text{ADI} &= \text{slot} \cdot 255 + \text{index} + 1 \\ \text{slot} &= (\text{ADI} - 1) / 255 \\ \text{index} &= (\text{ADI} - 1) \text{ MOD } 255 \end{aligned}$$

Examples:

ADI	Slot	Index
256	1	0
510	1	254
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.

Note: 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” on page 44
- “Error Handling” on page 43

3.4.2 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" on page 44
- "PROFIBUS DP-V1 Object (FDh)" on page 35

3.5 Parameterization Data Handling

3.5.1 General Information

The master identifies itself with 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 'Parameterization Data' attribute 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" on page 44
- "PROFIBUS DP-V1 Object (FDh)" on page 35 (Attribute #2, 'Parameterization Data')
- "Parameterization-related Keywords" on page 55

3.5.2 Validation

The User Parameterization Data must be evaluated by the host application. This is handled through the 'Parameterization Data' attribute in the PROFIBUS DP-V1 Object (FDh).

- **'Parameterization Data' attribute not implemented**

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

- **'Parameterization Data' attribute implemented**

The application must evaluate the contents of the 'Parameterization Data' 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

3.6 Configuration Data Handling

3.6.1 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 'Configuration Data' attribute in the PROFIBUS DP-V1 Object (FDh).

Implementing the 'Configuration Data' 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)" on page 35 (Attribute #3, 'Configuration Data')
- "I/O-related Keywords" on page 49
- "Definition of Modules" on page 50

3.6.2 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.

- **'Configuration Data' attribute 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...

"Modular Device Object Implemented" on page 52

"Parameterization-related Keywords" on page 55

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.

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

- **‘Configuration Data’ attribute 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 ADI's, 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 a no error code.

Important: 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.

Note: 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

3.7 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)" on page 35 (Attribute #4, 'SSA Enabled')
- "Supported DP Features" on page 46

Note: 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.

3.8 Parameter Read/Write with Call

3.8.1 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.

3.8.2 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 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 + 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

3.8.3 Parameter Write with Call Handling

The 'Parameter Write with Call' telegram 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)	0000... FFFFh	Number of the ADI which shall be written
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	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 0001h, used when writing
8		Subindex (low)	01h	
9	ADI number	ADI (high)	0000... FFFFh	Number of the ADI which shall be read
10		ADI (low)		

3.9 Identification & Maintenance (I&M)

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)" on page 35
- "Command Details: Get_IM_Record" on page 39
- "Command Details: Set_IM_Record" on page 40
- "Network Configuration Object (04h)" on page 29

I&M Data Structures

The I&M records uses the following data structures.

Record	Content	Size	Description
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')
	IM version	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #16 ('I&M Version')
	IM supported	2 bytes	PROFIBUS DP-V1 Object (FDh), attribute #17 ('I&M supported')
I&M1 ^a	Tag Function	32 bytes	-
	Tag Location	22 bytes	-
I&M2 ^a	Installation date	16 bytes	-
I&M3 ^a	Descriptor	54 bytes	-
I&M4 ^a	Signature	54 bytes	-

a. Data of this field can only be accessed from the network by the IO Controller/Supervisor.

See also...

- “PROFIBUS DP-V1 Object (FDh)” on page 35

4. 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)” on page 31
- “Diagnostic Object (02h)” on page 25

4.1 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 ‘Ext Diag Overflow’-attribute 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 ‘Static Diag’-attribute 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”.

4.2 Additional Diagnostics Information

4.2.1 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.

5. Anybus Module Objects

5.1 General Information

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

Standard Objects:

- “Anybus Object (01h)” on page 24
- “Diagnostic Object (02h)” on page 25
- “Network Object (03h)” on page 27
- “Network Configuration Object (04h)” on page 29

Network Specific Objects:

- “PROFIBUS DP-V0 Diagnostic Object (10h)” on page 31

5.2 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	-	-	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
12	LED colors	Get	struct of: UINT8 (LED1A) UINT8 (LED1B) UINT8 (LED2A) UINT8 (LED2B)	<u>Value:</u> <u>Color:</u> 01h Green 02h Red 01h Green 02h Red
13... 16	-	-	-	Consult the general Anybus CompactCom 40 Software Design Guide for further information.

Extended

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

5.3 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	Data Type	Value	Description
1... 4	-	-	-		Consult 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

#	Name	Access	Type	Value
1	Severity	Get	UINT8	See "Severity" on page 26.
2	Event Code	Get	UINT8	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
3	NW specific extension ^a	Get	Array of UINT8: <u>Element:</u> <u>Contents:</u> 0 (reserved) 1 (reserved) 2...57 Application Specific Field	
4	Slot	Get	UINT16	Consult the general Anybus CompactCom 40 Software Design Guide for further information.
5	ADI	Get	UINT16	
6	Element	Get	UINT8	
7	Bit	Get	UINT8	

a. The use of this attribute 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.

See also...

- "“PROFIBUS DP-V1 Object (FDh)” on page 35” on page 13
- "“Status Diagnostic Messages” on page 58

5.3.1 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 ^a .
10h	Minor, unrecoverable	Status message. Unrecoverable events cannot be deleted.	Identifier related + channel related ^a .
20h	Major, recoverable	Status message.	Identifier related + channel related ^a .
30h	Major, unrecoverable	Causes a state-shift to EXCEPTION.	
50h	Minor, latching	Diagnosis alarm.	
60h	Major, latching	See the Anybus CompactCom 40 Software Design Guide for more information.	
(other)	-	(reserved for future use)	

a. For more information, see "Additional Diagnostics Information" on page 22.

5.4 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.

Note: 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 ^a	Get	BOOL	True
5	Write process data size	Get	UINT16	Current write process data size (in bytes) Updated on every successful Map_ADI_Write_Area ^b
6	Read process data size	Get	UINT16	Current read process data size (in bytes) Updated on every successful Map_ADI_Read_Area ^b
7	Exception Information	Get	UINT8	Additional PROFIBUS specific exception information is presented here in case the Anybus module has shifted to the EXCEPTION state.

- a. This attribute indicates if the network supports acyclic data services and must not be confused with PROFIBUS Parameterization Data.
- b. Consult the general Anybus CompactCom 40 Software Design Guide for further information.

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)), 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.

5.5 Network Configuration Object (04h)

Category

Basic

Object Description

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

Note 1: 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, ‘Node Address’)

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” on page 17. This feature allows the device address to be set from the PROFIBUS master. Default after factory reset is 126.

#	Name	Access	Type	Description
1	Name ^a	Get	Array of CHAR	‘Node address’
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (get/set/shared access)
5	Value	Get/Set	UINT8	Value:Meaning: 0... 125 Node address (other) Get node address using SSA (default) (Note that support for SSA can be globally disabled by implementing the PROFIBUS object, see “PROFIBUS DP-V1 Object (FDh)” on page 35))

a. Multilingual, see “Multilingual Strings” on page 30.

Multilingual Strings

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

Instance	English	German	Spanish	Italian	French
1	Node address	Geräteadresse	Direcc nodo	Indirizzo	Adresse
3	Function tag	Funktion	Función	Descr. Funz.	Fonction
4	Location tag	Position	Locación	Descr. Locaz.	Placement
5	Install. date	Install. Tag	Fecha Inst.	Data install.	Date Installé
6	Description	Beschreibung	Descripción	Descrizione	Description

5.6 PROFIBUS DP-V0 Diagnostic Object (10h)

Using this object, it is possible to create transparent diagnostic data.

To be able to use the PROFIBUS DP-V0 Diagnostic Object, the application must write zero bytes to attribute #1 in instance #1 before setup is complete. Otherwise the module will detect a configuration mismatch and no diagnostics will be created. Thus, to use the object, and create transparent diagnostic data, a restart of the module is necessary.

Please note that this object cannot be used at the same time as the Diagnostic Object (02h), see page 25.

Supported Commands

Object: Get_Attribute
 Set_Attribute

Instance: Get_Attribute
 Set_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data 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	Comment
1	Diagnostic data	Get/Set	Array of UINT8	Array of bytes, containing the diagnostic data. For more information, see "Diagnostic Data Layout" on page 31.

5.6.1 Diagnostic Data Layout

Data Byte	Contents	Description
0	Extended diagnostic data flag	0 - No extended diagnostic data 1 - Extended diagnostic data is available
1 - 235	Diagnostic data	The user specific part of the diagnostic buffer

5.6.2 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

6. Host Application Objects

6.1 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)” on page 34

Network Specific Objects:

- “PROFIBUS DP-V1 Object (FDh)” on page 35

6.2 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.

- Anybus CompactCom 40 Software Design Guide, “Modular Device Object (ECh)”

Supported Commands

Object: Get_Attribute
 Get_List

Object Attributes (Instance #0)

#	Name	Access	Data 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.

6.3 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 start-up; 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]”).

Note 1: 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.

Note 2: Altering the default settings within this object may require a new GSD file, which in turn requires fieldbus recertification.

See also...

- “Front View” on page 63
- “Support for DP-V1 and DP-V0” on page 10
- “GSD File Customization” on page 44
- Guideline Information & Maintenance functions
- Anybus CompactCom Software Design Guide, “Error Codes”

Supported Commands

Object: Get_Attribute
 Get_IM_Record (See “Command Details: Get_IM_Record” on page 39)
 Set_IM_Record (See “Command Details: Set_IM_Record” on page 40)

Instance: Get_Attribute
 Set_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data 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	Comment
1	PNO Ident Number	Get	UINT16	PNO Ident Number (default = 1815h) See also... - "Device Identity" on page 11 - "Device Identification" on page 45
6	(not used)			Buffer mode in 30 series -

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. See also... - "PROFIBUS DP-V1 Object (FDh)" on page 35" on page 13 - "Diagnostic-related Keywords" on page 56
3	Configuration Data	Get/Set	Array of UINT8	This attribute is used both to specify the Expected Configuration Data, and for evaluation of the Actual Configuration Data. See also... - "Configuration Data Handling" on page 15 - "Definition of Modules" on page 50 Note: 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 'Set Slave Address'-support. <u>Value:</u> <u>Meaning:</u> True Enabled (default) False Disabled See also... - "Set Slave Address" on page 17 - "Supported DP Features" on page 46
5	(not used)			
7	(not used)			
8	Manufacturer ID	Get	UINT16	Device Manufacturer ID (default = 010Ch) See also... - "Device Identity" on page 11 - Guideline Information & Maintenance functions
9	Order ID	Get	Array of CHAR	Device Order ID, up to 20 characters (default = 'ABCC 40 - DPV1') See also... - "Device Identity" on page 11 - Guideline Information & Maintenance functions
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.

#	Name	Access	Type	Comment
11	Hardware Revision	Get	UINT16	Hardware revision of the device. <u>Value:</u> <u>Meaning:</u> 0... FFFEh Hardware revision FFFFh Indicates profile specific information See also... - Guideline Information & Maintenance functions 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. See also... - "Software Revision Structure" on page 38 - Guideline Information & Maintenance functions
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). See also... - Guideline Information & Maintenance functions
14	Profile ID ^a	Get	UINT16	Specifies the Profile ID. Default:F600h (Generic Device) See also... - Guideline Information & Maintenance functions
15	Profile specific type ^a	Get	UINT16	Specifies the Profile specific type. Default:0004h (Communication Module) See also... - Guideline Information & Maintenance functions
16	IM version	Get	Struct of: UINT8 (Major) UINT8 (Minor)	Should only be used if the application shall support another I&M version than 1.1. In such case, it is mandatory to implement Get_IM_Record and Set_IM_Record. If not implemented, the module defaults to v1.1. See also... - Guideline Information & Maintenance functions
17	IM supported	Get	UINT16	Bit field specifying which I&M records that are supported for slot 0. See also... - "IM Supported Structure" on page 38
18	IM header	Get	Array of UINT8	This header will be sent to the master together with I&M... 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)			

a. These attributes do not affect the behavior of the module, since it does not handle profiles.

Software Revision Structure

Member	Contents	Comments
CHAR	<u>Letter: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

IM Supported Structure

This is a bit field where each bit corresponds to an I&M parameter that shall be supported for slot 0. A set bit means that the corresponding I&M parameter is supported.

Bit	I&M Record	Default
0	Profile Specific IM	Disabled
1	IM1	Enabled
2	IM2	Enabled
3	IM3	Enabled
4	IM4	Enabled
other	-	(reserved)

Note: I&M0 is mandatory and cannot be disabled.

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 will be rejected as well.

Note: Implementation of this command is optional unless attribute #16 ('IM version') has been implemented.

- **Command**

Field	Contents
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 (Success)**

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

See also...

- “Command Details: Set_IM_Record” on page 40
- “Instance Attributes (Instance #1)” on page 36 (Attribute #16, 'IM version')
- Guideline Information & Maintenance functions

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 will be rejected as well.

Note: Implementation of this command is optional unless attribute #16 ('IM version') has been implemented.

- **Command**

Field	Contents
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 (Success)**

-

See also...

- “Command Details: Get_IM_Record” on page 39
- “Instance Attributes (Instance #1)” on page 36 (Attribute #16, 'IM version')
- Guideline Information & Maintenance functions

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.

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

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)	

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	0

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).

Note: 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

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 'PNO Ident Number'-attribute in the PROFIBUS DP-V1 Object (FDh).

Contact PNO to obtain a unique Ident Number.

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")

Related Information

Information	Page(s)
Device Identity	11
PROFIBUS DP-V1 Object (FDh) (Attribute #1)	35

D.3 Supported Hardware Features

General

Do not change the standard settings.

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

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): 0: 'SSA enabled'-attribute set to FALSE 1: 'SSA enabled'-attribute set to TRUE (or not implemented)

Related Information

Information	Page(s)
Set Slave Address	17
PROFIBUS DP-V1 Object (FDh) (Attribute #4)	35

D.5 Supported Baud Rates

General

Do not change the standard settings.

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

General

Do not change the standard settings.

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

General

Use the GSD file default value (6).

GSD File Entries

```
; Maximum polling frequency
Min_Slave_Intervall= 6      ; 0.6 ms
```


D.8 I/O-related Keywords

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	

Related Information

Information	Page(s)
"GSD File Customization" on page 44	10

D.9 Definition of Modules

General

These parameters need to be altered if customizing module names, if the Configuration Data attribute (“PROFIBUS DP-V1 Object (FDh)” on page 35) has been implemented.

GSD File Entries

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

Extended

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.

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

I/O Data Specification	
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 7:	More cfg follows
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:	
Bit 7:	More cfg follows

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:	No. of output ADI index follows
Bit 7:	More config for this slot follows
Byte 1 - 2:	Module ID

Vendor Specific Data	
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

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

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:	No. of output ADI index follows
Bit 7:	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 has the following modules:

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

Using the information in the two tables above, the expected configuration will 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 Example:

```
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

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.

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.

Related Information

Information	Page(s)
"GSD File Customization" on page 44	10
"PROFIBUS DP-V1 Object (FDh)" on page 35	13
PROFIBUS DP-V1 Object (FDh) (Attribute #2)	35

D.11 Diagnostic-related Keywords

GSD File Entries

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

Basic

Setting	Unit	Description
<diag_len>	bytes	

Related Information

Information	Page(s)
"GSD File Customization" on page 44	10
"PROFIBUS DP-V1 Object (FDh)" on page 35	13
Diagnostic Object (02h)	25
PROFIBUS DP-V1 Object (FDh) (Attribute #5)	35

D.12 Identification & Maintenance-related Keywords

GSD File Entries

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

Basic

Setting	Description
<supp>	1: I&M supported

Related Information

Information	Page(s)
Device Identity	11
Network Configuration Object (04h)	29
PROFIBUS DP-V1 Object (FDh)	35

D.13 Status Diagnostic Messages

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).

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>

Related Information

Information	Page(s)
"PROFIBUS DP-V1 Object (FDh)" on page 35	13
Diagnostic Object (02h)	25

D.14 DP-V1 related Keywords

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>	10ms	The host application must be able to provide a response to an ADI request within this time period, or the master may terminate the connection (default = 100)
<C2_L>	bytes	

Related Information

Information	Page(s)
"GSD File Customization" on page 44	10

D.15 Alarm-related Keywords

General

These keywords are only applicable if latching diagnostic instances are created.

GSD File Entries

```

; Alarm related keywords
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

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

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

```

Related Information

Information	Page(s)
"GSD File Customization" on page 44	10

E. Timing & Performance

E.1 General Information

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

The following timing aspects are measured:

Category	Parameters	Page
Startup Delay	T1, T2	61
NW_INIT Handling	T100	61
Event Based WrMsg Busy Time	T103	62
Event Based Process Data Delay	T101, T102	62

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

E.2 Internal Timing

E.2.1 Startup Delay

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

Parameter	Description	Max.	Unit.
T1	The Anybus CompactCom 40 PROFIBUS module generates the first application interrupt (parallel mode)	7.1	ms
T2	The Anybus CompactCom 40 PROFIBUS module is able to receive and handle the first application telegram (serial mode)	7.1	ms

E.2.2 NW_INIT Handling

This test measures the time required by the Anybus CompactCom 40 PROFIBUS 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	32 ^a
Event based application message response time	> 1 ms
Ping-pong application response time	> 10 ms
No. of simultaneously outstanding Anybus commands that the application can handle	1

a. Or maximum amount in case the network specific maximum is less.

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

E.2.3 Event Based WrMsg Busy Time

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

Parameter	Description	Min.	Max.	Unit.
T103	H_WRMSG area busy time	2.8	7.2	µs

E.2.4 Event Based Process Data Delay

“Read process data delay” is defined as the time from when the last bit of the network frame enters the module, to when the RDPDI interrupt is asserted to the application.

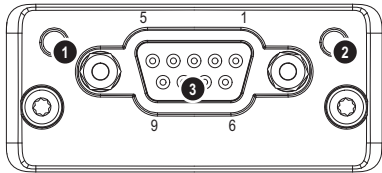
“Write process data delay” is defined in two different ways, depending on network type.

- 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	Comment	Unit
T101	Read process data delay	380 ns	Time from last input bit received until nIRQ goes active on the chip	ns
T102	Write process data delay	1.1 µs	Time from when the CompactCom 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.	µs

F. Technical Specification

F.1 Front View

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

Operation Mode

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" on page 14
Flashing Red (2 flashes)	PROFIBUS Configuration error	See "Configuration Data Handling" on page 15

Status

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'

PROFIBUS Connector (DB9F)

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 ^a	+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.

a. 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 degrees Celsius), 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" on page 64.

F.2 Protective Earth (PE) Requirements

In order to ensure proper EMC behavior, the module must be properly connected to protective earth via the PE pad / PE mechanism described in the general Anybus CompactCom 40 Hardware Design Guide.

HMS Industrial Networks does not guarantee proper EMC behavior unless these PE requirements are fulfilled.

F.3 Power Supply

Supply Voltage

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

Power Consumption

The Anybus CompactCom 40 PROFIBUS DP-V1 is designed to fulfil the requirements of a Class A module. For more information about the power consumption classification used on the Anybus CompactCom 40 platform, consult the general Anybus CompactCom 40 Hardware Design Guide.

At 12 Mbit the current hardware design consumes up to 200 mA¹².

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

F.4 Environmental Specification

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

F.5 EMC Compliance

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

-
1. Note that 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 PROFIBUS DP-V1 will remain a Class A module.
 2. 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)” on page 63).