



# Basic profile 10

V2.0

03-27-2012

07

Profile: Basic  
Profile number: 10  
Version: V2.0  
Version counter: 07  
Date: 27. March 2012  
Publisher: INTERBUS administrative department  
Phone: +49 5235 3-42100  
E-mail: [info@interbusclub.com](mailto:info@interbusclub.com)  
[www.interbus.com](http://www.interbus.com)

Copyright by INTERBUS administrative department

All figures and descriptions have been created and checked to the best of our knowledge. However, users should also carry out their own checks and tests. We reserve the right to make any changes, particularly those that serve the purpose of technical progress. The administrative department of the former INTERBUS Club Deutschland e.V. assumes no liability for erroneous handling or damage resulting from a failure to observe the information contained in this profile. This profile, including all figures contained herein, is copyright protected. Use of this profile by any third party in contravention of copyright is forbidden.  
Subject to alterations

## List of revisions

Index	Date	Modifications, additions or explanations
1.00	10/14/2002	First version
1.01	08/12/2004	<ul style="list-style-type: none"> <li>• Addition of channel number in diagnostic object 0018 and onwards</li> <li>• Addition of priorities 81, 82, 83 "Error removed" in diagnostic object 0018 and onwards</li> <li>• Revision/addition of error codes</li> <li>• Extension of handling description for diagnostic messages</li> <li>• Object 000F DeviceProfile Mandatory</li> <li>• Introduction and appendix added</li> <li>• Editorial revision</li> <li>• Introduction of objects               <ul style="list-style-type: none"> <li>- 002E – CheckSum</li> <li>- 002F - PDOUT_Subst</li> <li>- 0030 - PF_Code</li> <li>- 0031 - PDIN_Subst</li> <li>- 0032 - IBS_ID</li> </ul> </li> <li>• Definition of values for objects               <ul style="list-style-type: none"> <li>- 0024 – IBSResetCode</li> <li>- 0020 – PDTimeoutCode</li> </ul> </li> </ul>
1.02	01/28/2005	<ul style="list-style-type: none"> <li>• Error resolution (Re/Gr/formats/etc.)</li> <li>• Object 0019 "ResetDiag" name and meaning adapted</li> <li>• Object 000F "DeviceProfile" type modified</li> <li>• Error codes extended: 2344; 2345; 341X; 342X; 5230</li> </ul>
1.10	06/20/2006	<ul style="list-style-type: none"> <li>• Object 0018 "DiagState.Kanal" changed to "DiagState.Channel/Group", meaning adapted</li> <li>• Object 0018 "DiagState. MoreFollows" meaning adapted</li> <li>• Object 001A "GetErrorRepMethod", name and description extended</li> <li>• Object 0x001D "Password" adapted</li> <li>• Object 0x0025 "PDIN" additional explanation</li> <li>• Object 0x0026 "PDOUT" additional explanation</li> <li>• Object 002E "CheckSum", type changed from UINT16 =&gt; UNIT32</li> <li>• Object 0033 "DiagStateChannelNumber" added</li> <li>• Object 0034 "DiagStateAddValue" added</li> <li>• Object 0035 "NoOfModules" added</li> <li>• Object 0036 "DeviceStructure" added</li> <li>• Object 0037 "DeviceType" added</li> <li>• Object 0038 "ObjDescrReq" added</li> <li>• Object 0039 "ObjDescr" added</li> <li>• Object 003A "VersionCount" added</li> <li>• Object E805 "ObjDescrLong" added</li> <li>• Object E800 "DiagStateLong" described in part as a link and duplicate descriptions to be avoided</li> <li>• Modular devices described. The Invoke ID, now "Module number", is used for module addressing</li> <li>• List of permitted communication error codes revised and extended</li> <li>• "Bit string" data type added</li> <li>• "PCP objects" section revised, restructured, and extended</li> <li>• "Object description" section added</li> <li>• Addition of profile-specific error class 8 error code 1</li> </ul>

1.11	10/06/2009	<ul style="list-style-type: none"> <li>• Additional error code revised</li> <li>• Handling object 0x0038 "ObjDescrReq" object 0x0039 "ObjDescr" extended</li> <li>• Error class "8" other, error code "1" profile-specific consistently corrected</li> <li>• Visible string data type (always scheduled "0x00"!)</li> <li>• Object 0037 "DeviceType" slightly revised</li> <li>• Object 0008 "ProductID" re-named to "SerialNo" to avoid confusion</li> <li>• Error codes 6800 and 6810 introduced. A000 et seq. extended and explained in more detail</li> <li>• Explanation of the object area for modular devices</li> <li>• Object 003B "PDIN_Descr" added</li> <li>• Object 003C "PDOOUT_Descr" added</li> <li>• Object 0019 "ResetDiag" extended and made more specific</li> </ul>
1.12	11/19/2010	<ul style="list-style-type: none"> <li>• Device range supplemented by digital/analog IN/OUT</li> <li>• Data type list completed</li> <li>• LanguageCode corrected</li> <li>• Length for visible string corrected</li> <li>• Object 003B "PDIN_Descr" revised, new types added</li> <li>• Object 003C "PDOOUT_Descr" revised, new types added</li> <li>• Object 000C "FirmwareVersion" and 000B "PChVersion" supplemented by entries if no FW/PChVersion is available</li> <li>• Handling of priorities in diagnostics object 0x0018 DiagState explained</li> <li>• Object 0019 "ResetDiag" extended and made more specific</li> <li>• Object 0018.5 "ResetDiag.MoreFollows" made more specific</li> <li>• Additional error codes added</li> <li>• Object 0037 "DeviceType" error rectified R/W =&gt; R, could at no time have been R/W</li> <li>• Subindices "Access-Rights", "DisplayFormat", "Resolution", and "Offset" inserted in object 0x0039 "ObjDescr", description added</li> <li>• Value index for "force" and "el. resistance" added</li> <li>• Object 0020, 0024, 0030 UINT16 =&gt; array of UINT16 modified and commented</li> </ul>
2.0	12/07/2011	<ul style="list-style-type: none"> <li>• Basic profile "neutralized", i.e., relationship with INTERBUS and PCP dissolved. However, this is still an</li> <li>• Comments on the version counters added</li> <li>• Object 003D "WakeUpTime" added</li> <li>• ResetCode explained in more detail</li> <li>• Object 002D ResetParam code "02" added</li> <li>• <b>!!! Object 0x0039 "ObjDescr" incompatible, revised (minimum and maximum moved)</b></li> <li>• Translation table for the object names added to appendix</li> <li>• Description of download services (download write, download read), extended and made more specific</li> <li>• Object 0011 error in the string length (10+1) corrected</li> <li>• Object 0x0005 "Capabilities" added</li> <li>• Object 0038 "ObjDescrReq" length corrected</li> </ul>
		<ul style="list-style-type: none"> <li>•</li> </ul>

## Contents

1	Foreword.....	7
2	Application and device properties.....	8
3	General.....	9
4	Overview.....	9
4.1	Index ranges of the PCP objects.....	9
4.2	List of standard objects.....	10
4.3	List of permitted communication error codes.....	12
4.3.1	Error class and error code.....	12
4.3.2	Additional code:.....	13
5	PCP objects.....	15
5.1	Data types.....	16
5.2	Data objects.....	16
5.2.1	Domain variable object.....	16
5.2.1.1	Domain variable - formal description.....	16
5.2.2	Simple variable object.....	17
5.2.2.1	Simple variable - formal description.....	17
5.2.3	Array object.....	17
5.2.3.1	Array variable - formal description.....	17
5.2.4	Record object.....	18
5.2.4.1	Record variable - formal description.....	18
5.2.5	Variable list object.....	18
5.2.5.1	Variable list - formal description.....	18
5.2.5.2	Static variable list.....	19
5.2.5.3	Dynamic variable list.....	19
5.2.5.4	Transmission format for variable lists.....	19
5.2.6	String variable object.....	20
5.2.6.1	String variable - formal description.....	20
6	Services.....	21
6.1	Download function with write service ("Download-Write").....	21
6.2	Upload function with read service (Upload-Read).....	24
6.3	Example for variable list download.....	27
7	Standard objects.....	28
7.1	Identification.....	29
7.1.1	DeviceFamily.....	33
7.1.2	CommunicationProfile.....	34
7.1.3	DeviceProfile.....	34
7.2	Device diagnostics.....	35
7.2.1	Objects.....	35
7.2.1.1	Notification by reading the DiagState object.....	40
7.2.1.2	Notification via information report from the DiagState object.....	41
7.2.1.3	Classification of messages.....	41
7.2.1.4	Error codes.....	43
7.2.2	Trace data.....	49
7.3	User data management.....	50
7.3.1	Process data management.....	50
7.3.2	Parameter channel management.....	56
7.4	Device management.....	57
7.4.1	Block parameterization.....	58
7.4.1.1	Write control.....	58
7.4.1.2	Conflict dictionary.....	59
7.4.2	Parameter record identification.....	59
7.4.3	Password protection.....	60
7.5	Multilingual capacity.....	62

---

7.6	Modular devices .....	64
7.7	Object description.....	65
8	Appendix A.....	69
8.1	Definition of terms .....	69
8.2	Symbols and abbreviations .....	70
9	Appendix B.....	72
9.1	Translation table for the object names.....	72

## 1 Foreword

More powerful and more flexible systems for industrial sensors and actuators are increasingly required for factory automation. Intelligent field devices can meet these requirements. However, they must support open and standardized connectivity in order to be fully integrated in complex production procedures.

The basic idea of an open system is to enable the exchange of information between application functions which are implemented in devices from different manufacturers. This involves specified application functions, a uniform user interface for communication, and a uniform transmission medium.

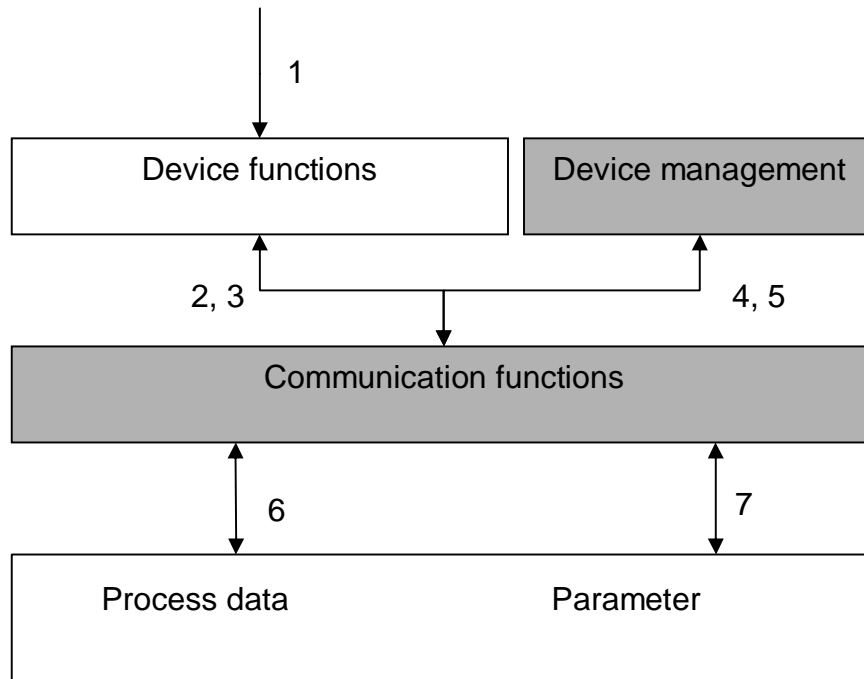
INTERBUS Club Deutschland e.V. has set itself the task of standardizing the most important field device functions and bringing them together in this profile. To enable field device functions to be defined regardless of the communication medium, an internationally recognized and standardized IEC 61158 user interface was used for communication. This provided continuity with MMS. No specific fieldbus system has been selected as a transmission medium. All the transmission medium has to do is meet the requirements for field communication in terms of realtime behavior and a standardized user interface.

The basic profile is aimed at all users and manufacturers of field devices that are designed for operation on a fieldbus. This profile definition provides the user with a logical means of enhancing standardized communication and it ensures universal agreement on data content and device behavior. These function definitions standardize the key field device parameters. This ensures that devices from different manufacturers behave in the same way in terms of the communication medium.

An independent specialist body will be set up for conformance testing and certification of products with the basic profile. As standardization work continues, further additions can be expected.

## 2 Application and device properties

This section describes the entire application in terms of communication. The application is divided into the following function blocks:



### Communication function

The communication function executes all communication-specific functions.

### Device function

The device function executes all device-specific functions.

### Device management

The device information manages information about the device in a non-volatile memory.

### Interaction between function blocks

1 Process variables

2 Process data from the higher-level control system to the device functions

3 Process data from the device functions to the higher-level control system

4 Storing device information for management, diagnostics, and identification

5 Reading device information for management, diagnostics, and identification

6 Mapping to the process data channel

7 Mapping to the parameter channel



### 3 General

In addition to information about the current state of the device, modern field device management and diagnostics also encompass information about the device itself, historical status information, error images, etc.

Previously only a small amount of the information available was provided as standard. The aim of this document is to define a basic profile which will enable uniform management of devices that are equipped with a parameter channel.

This basic profile enables device manufacturers to provide end users (as well as tools) with considerably more information about their devices and associated features.

This "multitude" of information simplifies device startup, in particular, due to this standardization. As for service work, there is also now the option to access certain diagnostics data in a uniform and thus faster manner.

Understanding the profile:

"Information and sequences that are similar in terms of content are also displayed in a standardized form, which aids manufacturers and users instead of restricting them."

In order to meet this requirement, the relevant information is stored consistently in standard objects. Mechanisms are specified that control the access to these objects and, in the event of errors, uniform error codes are established.

### 4 Overview

#### 4.1 Index ranges of the PCP objects

The PCP objects in the basic profile are divided into the following index ranges:

Index	Name	Object type
0	Object dictionary	OD
0001 – 003D	General standard objects These objects are further described in this document.	Variable
003E – 007F	Reserved	
0080 – 5FFF	Manufacturer-specific application objects Manufacturers specify their device-specific variable objects in this area.	Variable
6000 – CFFF	Profile-specific application objects Additional specified application objects can be found in the relevant profiles (e.g., DRIVECOM, sensor/actuator, etc.).	Variable
D000 – DFFF	Reserved	
E000 – E3FF	Static VariablenListen	VariableList
E400 – E7FE	Dynamic variable list	VariableList
E7FF	Creating and reading variable list definitions	VariableListStructure
E800 – E805	General domain variables These objects are further described in this document.	DomainVariable
E805 – E820	Reserved domain variable	DomainVariable
E820 – E8FF	Device-specific for domain variable Manufacturers specify their device-specific variable objects in this area.	DomainVariable
F000 – FFFF	Reserved	

## 4.2 List of standard objects

Index	Object name	R/W	Length	Data type	M/O/D
0001	VendorName	R	58 bytes, maximum	Visible string	<b>M</b>
0002	VendorID	R	6+1 bytes	Visible string	O
0003	VendorText	R	58 bytes, maximum	Visible string	O
0004	DeviceFamily	R	58 bytes, maximum	Visible string	<b>M</b>
0005600F	Capabilities	R	N x 8 bytes	Array of visible string	O
0006	ProductFamily	R	58 bytes, maximum	Visible string	O
0007	ProductName	R	58 bytes, maximum	Visible string	<b>M</b>
0008	SerialNo	R	58 bytes, maximum	Visible string	O
0009	ProductText	R	58 bytes, maximum	Visible string	O
000A	OrderNumber	R	58 bytes, maximum	Visible string	<b>M</b>
000B	HardwareVersion	R	2 entries	Record	<b>M</b>
000C	FirmwareVersion	R	2 entries	Record	<b>M</b>
000D	PChVersion	R	2 entries	Record	<b>M</b>
000E	CommProfile	R	3+1 bytes	Visible string	<b>M</b>
000F	DeviceProfile	R	4+1 bytes, maximum	Visible string	<b>M</b>
0010	Reserved				
0011	ProfileVersion	R	2 entries	Record	<b>M</b>
0012	VendorURL	R	58 bytes, maximum	Visible string	O
0013	DeviceDescFile	R	58 bytes, maximum	Visible string	O
0014	Location	R/W	58 bytes, maximum	Visible string	O
0015	EquipmentIdent	R/W	58 bytes, maximum	Visible string	O
0016	AppDeviceAddr	R/W	2 bytes	UINT16	O
0017	Language	R/W	2 entries	Record	<b>M</b>
0018	DiagState	R	6 entries	Record	<b>M</b>
0019	ResetDiag	W	1 byte	UINT8	O
001A	GetErrorRepMethod	R/W	1 byte	UINT8	O
001B	TestMode	R/W	2 bytes	UINT16	O
001C	ControlTrace	R/W	1 byte	UINT8	O
001D	Password	W	40 bytes, maximum	Octet string	O
001E	SetPassword	W	3 entries	Record	O
001F	PDTimeout	R/W	2 bytes	UINT16	O
0020	PDTimeoutCode	R/W	N x 2 bytes	Array of UINT16	O
0021	PChTimeout	R/W	2 bytes	UINT16	O
0022	PChTimeoutCode	R/W	2 bytes	UINT16	O
0023	AbortCode	R/W	2 bytes	UINT16	O
0024	ResetCode	R/W	N x 2 bytes	Array of UINT16	O
0025	PDIN	R	PD length	Octet string	<b>M</b>
0026	PDOOUT	R/W	PD length	Octet string	<b>M</b>
0027	GetExRight	R/W	1 byte	UINT8	O
0028	ChangePDSet	R/W	2 bytes	UINT16	O
0029	ParamSetWriteControl	R/W	1 byte	UINT8	O
002A	ConflictDictionary	R	2*N entries	Record	O
002B	ParamSet	R/W	2 bytes	UINT16	O
002C	ParameterMoment	R/W	2 entries	Record	O
002D	ResetParam	W	1 byte	UINT8	O
002E	Checksum	R	4 bytes	UINT32	O
002F	PDOOUT_Subst	R/W	PD length	Octet string	D
0030	PF_Code	R/W	N x 2 bytes	Array of UINT16	O
0031	PDIN_Subst	R/W	PD length	Octet string	D
0032	FieldBus_ID	R	2 entries	Record	<b>M</b>
0033	DiagStateChannelNo	R	2 entries	Record	D
0034	DiagStateAddValue	R	2 entries	Record	D
0035	NoOfModules	R	1 byte	UINT8	D

0036	DeviceStructure	R/W	4 entries	Record	D
0037	DeviceType	R	8 bytes	Octet string	O
0038	ObjDescrReq	R/W	2 entries	Record	O
0039	ObjDescr	R	16 entries	Record	O
003A	VersionCount	R	4 entries	Array	<b>M</b>
003B	PDIN_Descr	R	4 entries	Array	O
003C	PDOOUT_Descr	R	4 entries	Array	O
003D	WakeUpTime	R	2 bytes	UINT16	O
E800	DiagStateLong	R		Domain variable record	O
E801	DiagHistory	R		Domain variable record	O
E802	DiagHistoryLong	R		Domain variable record	O
E803	TraceBuffer	R		Domain variable	O
E804	LanguageAvailable	R		Domain variable record	O
E805	ObjDescrLong	R		Domain variable record	O

R = Read only, this object is read only  
W = Write only, this object can only be written  
R/W = Read/Write, this object can be read and written

M = Mandatory (must be implemented)  
O = Optional (does not have to be implemented)  
D = Dependent (depends on other objects, must be implemented if the dependency is specified)

### 4.3 List of permitted communication error codes

#### Error messages that occur during communication via parameter channel:

The "Error type" service parameter consists of the following parameters:

- Error class
- Error code
- Additional code

#### 4.3.1 Error class and error code

Error class	Error code	Meaning
2 Application reference		This error class refers to the communication relationship which is used to process the service.
	1 Application unreachable	Access to the object is not possible because the application is not implemented/available. This error message usually appears when an attempt is made to access a module that does not exist on a modular device via the module number. Module not available or module number incorrect.
	0 Other	This error code is reported if the error cannot be assigned to any of the error codes previously specified.
5 Service		This error class is reported in the event of a faulty service.
	1 Object state conflict	This error code is reported if the current state of the object prevents the service from being executed.
	2 Service PDU size	This error code is reported in the event of a problem with the PDU size.
	3 Object constrain conflict	This error code is reported if the service cannot be executed at present.
	4 Parameter inconsistent	This error code is reported if the service contains inconsistent parameters.
	5 Illegal parameter	This error code is reported if a parameter accepted an invalid value.
	0 Other	This error code is reported if the error cannot be assigned to any of the error codes previously specified.
6 Access		This error class is reported in the event of faulty access.
	1 Object invalidated	This error code is reported if access relates to a defined object, which has an undefined reference attribute. This represents a permanent error for access to this object.
	2 Hardware fault	Access to the object failed due to a hardware fault. More detailed information about the cause can be found in the additional code (global = 8 and 9).
	3 Object access denied	Client has insufficient access rights.
	4 Invalid address	Access to an invalid internal address.
	5 Object attribute inconsistent	A service parameter accepted an invalid value. More detailed information about the cause can be found in the additional code (global = 1).
	6 Object access unsupported	The object is not a variable access object.

	7 Object non existent	No object exists for this index.
	8 Type conflict	The data does not correspond to the object data type.
	0 Other	This error code is reported if the error cannot be assigned to any of the error codes previously specified.
8 Other (application)	0 Other	The service was not executed. The cause is manufacturer-specific. More detailed information about the cause can be found in the additional code.
	1 Profile-specific	The service was not executed. The cause is profile-specific. More detailed information about the cause can be found in the additional code. All additional codes listed in this profile are profile-specific.

### 4.3.2 Additional code:

The additional code consists of a global, specific, and device-specific part. Specifying a global or a specific code is optional. This means that if the application cannot provide more detailed information about the error cause, additional code = 0000hex is output. The specific code contains a detailed description of the error cause defined in the global code. If the error cause does not logically correspond to a specific code, then specific code = 0 must be specified. The device-specific code (bits 8 to 15) is specified by the device manufacturer.

2 octets with the following structure:

Bit 15	8	7	4	3	0
	Device-specific		Global code		Specific code

The values for bits 8 to 15 are reserved and are currently set to 0.

All additional codes listed in this profile are profile-specific.

Global code [hex]	Specif. code [hex]	Meaning
0	0	No detailed information about the error cause
1	0	Service parameter with invalid value
1	1	Subindex not available
1	2	Object access is not a request
1	3	Reserved service code
1	4	SubSlot not supported
1	5	Object access type not supported on this object
1	6	Object access request index for this access type must be 0x0000
1	7	Object access request length for this access type must be 0
1	8	Object length for this object does not match
1	9	Object is read only and may not be overwritten.
2	0	Service cannot be executed at present
2	1	Service cannot be executed at present due to local control
2	2	Service cannot be executed in current device state (device control)
2	3	Service cannot be executed at present as no object dictionary is available
3	0	Parameter value out of range (the server will not provide the value)
3	1	Parameter value too large
3	2	Parameter value too small
4	0	Collision with other values
4	1	PCP object cannot be mapped to the process data
4	2	Process data length exceeded
8	0	Hardware fault
8	1	Application failed
A	0	Invalid segment number E.g, upload without initiation with subindex == 0xFF
A	1	Resource not available No more resources (memory) are available for downloading
A	2	Incorrect CRC
A	3	Error opening the file (if file system is available)
A	4	Error writing the file (if file system is available)
A	5	Error closing the file (if file system is available)
A	6	Segment missing Fewer data blocks were received than specified in the last segment
A	7	Too many segments More data blocks were received than specified in the last segment
A	8	Error reading the file (if file system is available)
A	9	Error reading the file (if file system is available)
B	1	The password cannot be replaced (deleted).
B	2	The password cannot be added (too many passwords).
B	3	The password cannot be assigned for the desired type of access.

Codes that are not listed are reserved.

If a manufacturer cannot assign an error message to any of the above, the INTERBUS Club should be notified.

## 5 PCP objects

Unlike standard parameter channel implementation, the PCP objects are not described in an object dictionary, but must instead be recognized implicitly by the user, e.g., from a description of the objects in the user manual for the relevant device.

## 5.1 Data types

Index of type (dec.)	Symbol description	Number of bytes
1	Boolean	1
2	INT8	1
3	INT16	2
4	INT32	4
5	UINT8	1
6	UINT16	2
7	UINT32	4
8	Floating point	4
9	Visible string (always scheduled "0x00"!)	1,2,3 ...
10	Octet string	1,2,3 ...
11	Date	7
12	Time of day	6
13	Time difference	6
14	Bit string	1,2,3 ...

## 5.2 Data objects

### 5.2.1 Domain variable object

The "Domain variable" object is used to transmit user-specific structured data of a moderately large to very large size (greater than (PDU size - 6) bytes). The object description must be recognized implicitly by the application program.



The "Domain variable" object can only be addressed fully and only using the "Download-Write" and "Upload-Read" macro services.

The total size of the domain variable must not exceed the maximum user data volume of  $[\text{PDU size} - 8] * \text{FFF0}_{\text{hex}}$  (approximately 3.5 MB).

#### 5.2.1.1 Domain variable - formal description

Object: domain variable  
 Object code: 0x02  
 Key attribute (m): index  
 Attribute (m): length (maximum PDU size - 8) \* FFF0<sub>hex</sub> bytes  
 Attribute (o): password

Services:

- (o) Download-Write
- (o) Upload-Read



## 5.2.2 Simple variable object

The "Simple variable" object represents a single, basic variable. The object description must be recognized implicitly by the application program. The size of a simple variable must not exceed the maximum user data volume of [PDU size – 6] bytes.

### 5.2.2.1 Simple variable - formal description

Object: simple variable  
Object code: 0x07  
Key attribute (m): index  
Attribute (m): length (maximum (PDU size - 6) bytes)  
Attribute (o): password

Services:

- (o) Read
- (o) Write
- (o) Information report

## 5.2.3 Array object

The "Array" object consists of a string of simple variables of the same data type. The object description must be recognized implicitly by the application program.

If an element of the object is addressed, a subindex must also be specified in the service next to the index. Subindex 1 accesses the 1st element of the object. Subindex 0 addresses the object as a whole.

It is not mandatory for each element of an array to be addressed individually.

The total size of the array must not exceed the maximum user data volume of (PDU size - 6) bytes.

### 5.2.3.1 Array variable - formal description

Object: array variable  
Object code: 0x08  
Key attribute (m): index  
Attribute (m): length (maximum (PDU size - 6) bytes)  
Attribute (m): number of elements  
Attribute (o): password

Services:

- (o) Read
- (o) Write
- (o) Information report

## 5.2.4 Record object

The "Record" object consists of a string of simple variables and, in certain cases, a string of variables at the end with different data types. The object description must be recognized implicitly by the application program. The record object may be addressed as a whole or element by element. If the object is to be addressed as a whole, the associated index is specified in the service.

If an element of the object is addressed, a subindex must also be specified in the service next to the index. Subindex 1 accesses the 1st element of the object.

It is not mandatory for each element of a record to be addressed individually.

The total size of the record must not exceed the maximum user data volume of (PDU size - 6) bytes.

### 5.2.4.1 Record variable - formal description

Object: record variable  
 Object code: 0x09  
 Key attribute (m): index  
 Attribute (m): length (maximum (PDU size - 6) bytes)  
 Attribute (m): list of elements  
 Attribute (o): password

Services:

- (o) Read
- (o) Write
- (o) Information report

## 5.2.5 Variable list object

A variable list contains a collection of individual variables. The structure must be recognized implicitly by the application program.



As the values belonging to the variable list can quickly exceed the maximum size of a telegram, objects in the variable list data type are accessed using the "Download-Write" and "Upload-Read" macro services.

Compiling several individual variables into a variable list is only useful for variables with small data types (e.g., UINT8, Integer32, etc.). In theory, the length of an individual variable in a variable list can be a maximum of (PDU size - 8) bytes. However, this sort of variable can be accessed more effectively using a normal Read service.

### 5.2.5.1 Variable list - formal description

Object: variable list  
 Object code: 0x0A  
 Key attribute (m): index  
 Attribute (m): number of elements  
 Attribute (m): list of elements index  
 → Attribute (m): → index  
 Attribute (o): password

Services:

- (o) Download-Write
- (o) Upload-Read

#### 5.2.5.2 Static variable list

The device manufacturer may have created static variable lists. These variable lists are found in the "static variable lists" index range, see index areas for PCP objects.

The structure of the static variable list can be read using the "VariableListRecord" variable (index E7FF). First of all, the index of the variable list to be read is written to object E7FF. Download-Write should be used here, as object E7FF is a domain object. The "VariableListRecord" can then be read using Upload-Read access to object E7FF.

The "VariableListRecord" is transmitted according to the following structure:

Index (2 bytes)	Meaning
1st value	Index of the 1st variable in the list
2nd value	Index of the 2nd variable in the list
...	
...	
...	
nth value	Index of the nth variable in the list

#### 5.2.5.3 Dynamic variable list

Dynamic variable lists are created using the "VariableListRecord" variable (index E7FF). These variable lists are found in the "dynamic variable lists" index range (see index ranges for PCP objects). The "VariableListRecord" is created using Download-Write access to object E7FF.

The "VariableListRecord" is transmitted according to the following structure:

Index (2 bytes)	Meaning
1st value	Index under which the "VariableListRecord" is to be created
2nd value	Index of the 1st variable in the list
3rd value	Index of the 2nd variable in the list
...	
...	
...	
nth value	Index of the nth variable in the list

#### 5.2.5.4 Transmission format for variable lists

To read and write a variable list, all the values of the individual variables are entered one after the other in the "Data" parameter of the Upload-Read or Download-Write.

	Meaning
1st value	Content of the 1st variable in the list
2nd value	Content of the 2nd variable in the list
...	
...	
nth value	Content of the nth variable in the list

## 5.2.6 String variable object

The "String variable" object represents a single, basic variable that is characterized by a specific data type (octet string, visible string or bit string).

The object description of the "variable access object" string variable is stored statically in the object dictionary (S-OD). A string variable is mapped to a real string variable that actually exists in the user system by means of the object description for the "String variable" object.

In the description of the "String variable" object, only one data type (octet string, visible string or bit string) is permitted.

A string variable has the same structure as a simple variable.

The difference is the variable length of the data types. The maximum length is configured in the object dictionary.

### 5.2.6.1 String variable - formal description

Object: array variable  
Object code: 0x0B  
Key attribute (m): index  
Attribute (m): maximum length (maximum (PDU size - 6) bytes)  
Attribute (o): password

Services:

- (o) Read
- (o) Write
- (o) Information report

## 6 Services

All basic variables, arrays, and records are accessed using the standard services, for example:

- Read
- Write
- Information report

As these services are functions of parameter channel implementation, they are not described here. For additional information, please refer to the relevant parameter channel documentation)\*.

The domain variable and variable list objects, however, are accessed using the Download-Write and Upload-Read macro services. The use of these services is described below.

)\* E.g., IBS PCP RE HB (5052b.pdf) and IBS PCP RR HB (5054b.pdf)

### 6.1 Download function with write service ("Download-Write")

Unlike standard parameter channel implementation (which contains download services), large volumes of data (e.g., application program, firmware update, variable lists, etc.) are transmitted using the write service. A download protocol is defined in the user data of the write service, which enables segmented transmission of larger volumes of data. Downloading can only be performed on domain variables and variable lists.

Parameters	Req/Ind	Rsp/Cnf
Communication reference	M	M
Module number	M	M
Index	M	
Subindex	O	
Array	M	
Segment number	M	
Data	M	
Result(+)		S
Result(-)		S
Error type		M

#### Communication reference

The communication reference addresses the desired communication partner. ("Compact" implemented devices can only communicate with the master (not peer-to-peer), which means that the communication reference is always 2.)

## Module number (Invoke ID)

The Invoke ID, which was previously not used and only available for reasons of compatibility, is used to address modules on a modular device. For this reason, the name of this parameter has also been changed to "Module number".

However, the handling of this parameter has not changed. The module number of the indication is still entered in the response.

For more detailed information, please refer to the "Modular devices" section.

## Index

The parameter index addresses the object to be written.

## Subindex

The value of the subindex for the download function is "0".

## Array

The "Array" parameter contains the download protocol.

Array	
Segment number 2-byte download control	User data 1 .. (PDU size – 8) bytes download data blocks E.g. 1 .. 56 bytes for "Compact" implementation

## Segment number

This parameter contains the number of the data block to be written, whereby the following definition applies:

Segment number = 1	Initiation of a download sequence with the first data block.
Segment number = 2..0xFFFF0hex	Data block number n
Segment number = 0xFFFFDhex	The last data block only contains <ul style="list-style-type: none"> <li>the number of transmitted data blocks (without this end data block) (2 bytes) and</li> <li>the CRC16 residual polynomial<sup>1)</sup> (2 bytes)</li> </ul>
Segment number = 0xFFFFEhex	The last data block only contains <ul style="list-style-type: none"> <li>the number of transmitted data blocks (without this end data block) (2 bytes) and</li> <li>the CRC32 residual polynomial<sup>1)</sup> (4 bytes)</li> </ul>
Segment number = 0xFFFFFhex	The last data block only contains <ul style="list-style-type: none"> <li>the number of the transmitted data blocks (without this end data block) (2 bytes) and no CRC residual</li> </ul>

<sup>1)</sup> The CRC is always generated using all the pure user data. The segment numbers are not part of the user data.

The download server must check that the segment number sequence is in ascending order without any gaps and, in the event of an error, send a negative response to the client (usually the fieldbus master). In the case of a negative response, the client can transmit the correct segment or terminate transmission with an end segment.

Only an end data block can terminate the download service.

**Data**

This parameter contains the data blocks. The length of these data blocks depends on the maximum PDU size. For a PDU size of 64 bytes, the maximum data block length is 56 bytes. The maximum data block length does not have to be used.

**Result(+)**

The "Result(+)" parameter indicates a positive result. The segment number may only be incremented if the result is positive.

**Result(-)**

The "Result(-)" parameter indicates a negative result.

**Error type**

The "Error type" parameter contains the error cause.

The content of the "Error type" parameter corresponds to that of the write service, with the following definition for the specific download sequence error:

Error class "8" - other  
Error code "1" – profile-specific

Additional code (hex)	Meaning
0x00A0 Invalid segment number - segment missing	Invalid segment number (segment missing)
0x00A1 Resource unavailable	No more resources (memory) are available for downloading
0x00A2 Invalid CRC	Incorrect CRC
0x00A3 File open error	Error opening the file (if file system is available)
0x00A4 File write error	Error writing the file (if file system is available)
0x00A5 File close error	Error closing the file (if file system is available)
0x00A6 Segment missing	Fewer data blocks were received than specified in the last segment
0x00A7 Segment overrun	More data blocks were received than specified in the last segment
0x00A9 Invalid segment number – double segment	Invalid segment number (segment duplicated, segment ignored)

## 6.2 Upload function with read service (Upload-Read)

Unlike standard parameter channel implementation (which contains upload services), large volumes of data (e.g., application program, backups, variable lists, etc.) are read using the read service. An upload protocol is defined in the subindex of the read service, which enables segmented transmission of fairly large volumes of data. Uploading can only be performed on domain variables and variable lists.

Parameters	Req/Ind	Rsp/Cnf
Communication reference	M	M
Module number	M	M
Index	M	
Subindex	M	
Result(+)		S
Array		M
Segment number		M
Data		M
Result(-)		S
Error type		M

### Communication reference

Like download function with write service ("Download-Write")

### Module number

Like download function with write service ("Download-Write")

### Index

Like download function with write service ("Download-Write")

### Subindex

An upload sequence is initiated using a special subindex, whereby the following definition applies:

Subindex = 0xFF<sub>hex</sub> Initiation of an upload sequence with request for the first data block

Subindex = 0x00<sub>hex</sub> Request for the next data block

Subindex = 0x01<sub>hex</sub> Request for the relevant previous data block (e.g., after an error)

### Result(+)

The "Result(+)" parameter indicates a positive result. The segment number may only be incremented if the result is positive.



## Array

The "Array" parameter contains the upload protocol.

Array	
Segment number 2-byte upload controller	User data 1 .. (PDU size – 8) bytes download data blocks 1 .. 56 bytes for "Compact" implementation

### Segment number

This parameter contains the number of the data block to be read, whereby the following definition applies:

Segment number = 1	Initiation of an upload sequence with the first data block.
Segment number = 2..0xFFF0 <sub>hex</sub>	Data block number n
Segment number = 0xFFFD <sub>hex</sub>	The last data block only contains <ul style="list-style-type: none"> <li>• the number of transmitted data blocks (without this end data block) (2 bytes) and</li> <li>• the CRC16 residual polynomial<sup>1)</sup> (2 bytes)</li> </ul>
Segment number = 0xFFFE <sub>hex</sub>	The last data block only contains <ul style="list-style-type: none"> <li>• the number of transmitted data blocks (without this end data block) (2 bytes) and</li> <li>• the CRC32 residual polynomial<sup>1)</sup> (4 bytes)</li> </ul>
Segment number = 0xFFFF <sub>hex</sub>	The last data block only contains <ul style="list-style-type: none"> <li>• the number of the transmitted data blocks (without this end data block) (2 bytes) and no CRC residual</li> </ul>

<sup>1)</sup> The CRC is always generated using all the pure user data. The segment numbers are not part of the user data.

The upload client must check that the segment number sequence is in ascending order without any gaps. In the event of an error, the client can request the last segment again with subindex 0xFE or restart the upload with subindex 0xFF.

### Data

This parameter contains the data blocks. The length of these data blocks depends on the maximum PDU size. For a PDU size of 64 bytes, the maximum data block length is 56 bytes. The maximum data block length does not have to be used.

**Result(-)**

The "Result(-)" parameter indicates a negative result.

**Error type**

The "Error type" parameter contains the error cause.

The content of the "Error type" parameter corresponds to that of the read service, with the following definition for the specific upload sequence errors:

Error class "8" - other

Error code "1" - profile-specific

<b>Additional code (hex)</b>	<b>Meaning</b>
0x00A0 Invalid segment number - segment missing	Upload without initiation with subindex == 0xFF
0x00A3 File open error	Error opening the file (if file system is available)
0x00A5 File close error	Error closing the file (if file system is available)
0x00A8 File read error	Error reading the file (if file system is available)

### 6.3 Example for variable list download

In this example, a "Compact" device uses the following objects, which are grouped into a variable list.

Index	Subindex	Data type	Length	Example value (hex)
0080	0	UINT8	1	FF
0081	7	UINT16	2	12 34
0090	3	UINT8	1	00
0091	0	Octet string	16	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0100	0	Octet string	16	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
0101	0	Octet string	16	20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
0105	0	Octet string	16	30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F

#### Download-Write

##### 1. Download-Write request/indication

Parameters	Content (hex)
Com. ref.	02
Module number	00
Index	E0 00
Subindex	00
Segment number	00 01
Data	FF 12 34 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F

##### 2. Download-Write request/indication

Parameters	Content (hex)
Com. ref.	02
Module number	00
Index	E0 00
Subindex	00
Segment number	00 02
Data	30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F

##### 3. Download-Write request/indication

Parameters	Content (hex)
Com. ref.	02
Module number	00
Index	E0 00
Subindex	00
Segment number	FF FF
Data	00 02

## 7 Standard objects

The following objects are described as follows:

Index (hex)	Unique ID for the object The services use this index to access the object
Object name	Meaningful name for the object
R/W	Access type R = Read only W = Write only R/W = Read and write access
Length	Length of object in bytes
Data type	Data type of the object, see above
Meaning	Explanation of the object contents
M/O	Implementation instruction M = Mandatory This object must be implemented. O = Optional This object can be implemented. D = Dependent This object must be implemented, if a specific optional function is implemented.

To ensure standardized access using tools, the objects, if they exist, must usually be implemented in the precise form defined below. However, the device manufacturer can adapt the contents to meet requirements. For example, a hardware state that cannot be determined by the device firmware for technical reasons could be represented as follows:

000B	HardwareVersion	R	2 entries	Record	Hardware version (device or communication module)	<b>M</b>
.1	• BuildDate	R	10+1 bytes	Visible string	0000-00-00	<b>M</b>
.2	• Version	R	Maximum 40 bytes	Visible string	<i>Not available</i>	<b>M</b>

## 7.1 Identification

The information specified in the following objects describes the device itself, the manufacturer, and the device application. On dispatch, the data entered must match up with what is printed on the device.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
0001	VendorName	R	Maximum 58 bytes	Visible string	Manufacturer name e.g., Phoenix Contact GmbH & Co KG	<b>M</b>
0002	VendorID	R	6+1 bytes	Visible string	Organizationally unique identifier Organizationally Unique Identifiers (OUI) according to: <a href="http://standards.ieee.org/regauth/oui/index.shtml">http://standards.ieee.org/regauth/oui/index.shtml</a> e.g., "00A045"	O
0003	VendorText	R	Maximum 58 bytes	Visible string	Comments on the manufacturer e.g., address, sector, etc.	O
0012	VendorURL	R	Maximum 58 bytes	Visible string	Manufacturer URL e.g., <a href="http://www.phoenixcontact.com">http://www.phoenixcontact.com</a>	O
0007	ProductName	R	Maximum 58 bytes	Visible string	Manufacturer-specific, unique product designation e.g., IB IL 24 DO 8	<b>M</b>
0008	SerialNo (ProductId)	R	Maximum 58 bytes	Visible string	Serial number Manufacturer-specific, unique device code e.g., 123456789	O
0009	ProductText	R	Maximum 58 bytes	Visible string	Manufacturer-specific product text e.g., digital output module	O
000A	OrderNumber	R	Maximum 58 bytes	Visible string	Manufacturer-specific, unique product type ID, e.g., Order No. e.g., 2726269	<b>M</b>
0037	DeviceType	R	8 bytes	Octet string	Manufacturer-specific device/module identification not equal to "0" The manufacturer-specific device/module identification can be used to replace and operate devices of the same type within a configuration. For example, a 16-channel output module with screw-on coupling may be replaced by one with spring-cage coupling even though it does not have the same order number. Different functionalities also require a different device type. If this object is used in a modular device, it can also indicate a dummy. "0" if module is (still) not plugged in. Nevertheless, in this case, data can be mapped to the process data.	O
0032	FieldBus_ID	R	2 entries	Record	Fieldbus identification information	<b>M</b>
.1	• ID-Code	R	1 byte	UINT8	Fieldbus-specific ID code (8 bit, usually represented as a decimal number)	<b>M</b>
.2	• PDLength	R	2 bytes	UINT16	Number of process data bits (unit bit)	<b>M</b>
000B	HardwareVersion	R	2 entries	Record	Hardware version (device or communication module) <i>On dispatch, the data entered must match up with what is printed on the device.</i>	<b>M</b>

.1	• BuildDate	R	10+1 bytes	Visible string	Manufacturing date Version date Format YYYY-MM-DD According to ISO 8601 e.g., 2002-11-29	M
.2	• Version	R	Maximum 40 bytes	Visible string	Version ID e.g., 4.01 Beta customer	M
000C	FirmwareVersion	R	2 entries	Record	Firmware version (device or communication module) <i>On dispatch, the data entered must match up with what is printed on the device.</i>	M
.1	• BuildDate	R	10+1 bytes	Visible string	BuildDate Version date Format YYYY-MM-DD According to ISO 8601 e.g., 2002-05-03 If no FW is available, "0000-00-00" is entered	M
.2	• Version	R	Maximum 40 bytes	Visible string	Version ID e.g., 1.03 customer If no FW is available, "--" is entered	M
000D	PChVersion	R	2 entries	Record	Parameter channel version Parameter channel implementation version	M
.1	• BuildDate	R	10+1 bytes	Visible string	BuildDate Version date Format YYYY-MM-DD According to ISO 8601 e.g., 2002-05-03 If no parameter channel is available, "0000-00-00" is entered	M
.2	• Version	R	Maximum 40 bytes	Visible string	Version ID e.g., "PCP Compact V1.00" If no parameter channel is available, "--" is entered	M
0005	Capabilities	R	Array of 8 bytes	Visible string	Properties/functionalities of the device in addition to the basic functions. All current functionalities are listed below: <ul style="list-style-type: none"> <li>• "Nothing" - This entry does not define any additional functions. The entry can also occur several times as a dummy.</li> <li>• "Syncl_0" - The slave supports synchronization of the inputs.</li> <li>• "SyncO_0" - The slave supports synchronization of the outputs.</li> </ul> <p>The type is exactly 8 characters long. Characters not used to be completed with 0x00. If the device does not have any properties/functionalities other than the basic functions, this object does not have to be implemented. If the device has properties/functionalities other than the basic functions, this object must be implemented.</p> <p>If the slave supports the "Capabilities" object but no other additional functions, the "Nothing" entry is included at least once.</p>	D
0006	ProductFamily	R	Maximum 58 bytes	Visible string	Manufacturer-specific product range e.g., Inline	O

003D	WakeUpTime	R	2 bytes	UINT16	Wake-up time The period of time that elapses between the point at which the supply voltage is switched on (if there are several supply voltages, the time when the last relevant one is switched on) and the time when the system is ready to operate. It is used to support the various energy saving profiles. The parameter only needs to be created if the wake-up time > 500 ms. If this parameter is not available, a default value of < 500 ms is assumed.	D
0004	DeviceFamily	R	Maximum 58 bytes	Visible string	Device range (e.g., I/O, drive) according to specifications, see below, e.g., motion control/frequency inverters	M
000E	CommProfile	R	Maximum 4+1 bytes	Visible string	Communication profile System-specific profile ID, see below. e.g., 634	M
000F	DeviceProfile	R	Maximum 4+1 bytes	Visible string	Device profile, designation of the application profile (see below) e.g., "10" for the basic profile or "22" for the DRIVECOM profile	M
0011	ProfileVersion	R	2 entries	Record	Version designation of this profile	M
.1	• BuildDate	R	10+1 bytes	Visible string	"2012-03-27"	M
.2	• Version	R	Maximum 40 bytes	Visible string	"Basic profile V2.0"	M
0013	DeviceDescFile	R	Maximum 58 bytes	Visible string	File name of the FDCML device description file, e.g., XYZ.xml	O
003A	VersionCount	R	4 entries	Array	Version counter Unique, consecutive numbering for the version of the corresponding components with the following indices: • 0x0011 • 0x000D • 0x000B • 0x000C If a given component is changed, the corresponding VersionCount must be increased by 1.	M
.1	• ProfileVersion	R	2 bytes	UINT16	07 For this profile.	M
.2	• PChVersion	R	2 bytes	UINT16	e.g., 01 for "Compact" version implementation	M
.3	• HardwareVersion	R	2 bytes	UINT16	e.g., 02 for the target hardware	M
.4	• FirmwareVersion	R	2 bytes	UINT16	e.g., 05 for the target firmware	M
0014	Location	R/W	Maximum 58 bytes	Visible string	Installation location Text that was stored in this parameter by the device user. This text indicates the installation location of the device and is stored in a non-volatile memory. E.g., machine 1, back-left	O
0015	EquipmentIdent	R/W	Maximum 58 bytes	Visible string	Equipment identifier Text that was stored in this parameter by the device user. The device description is stored in a non-volatile memory. Here, for example, the device user can store a description about device operation in the system. E.g., M1H747h.l.	O

0016	ApplDeviceAddr	R/W	2 bytes	UINT16	Application device address (user-specified device number) Any identification for this device in this specific application. This code number need not be unique. Management is left entirely to the end user. E.g., 123	○
------	----------------	-----	------------	--------	--	---



### 7.1.1 DeviceFamily

A unique designation for the device range.

en	de
Actuator	Aktor
Bus coupler	Buskoppler
Closed loop controller	Regler
Dosing device	Dosiergerät
Drive	Antrieb
Drive - frequency inverter	Antrieb - Frequenzumrichter
Drive – motor starter	Antrieb - Motorschalter
Drive – servo amplifier	Antrieb - Servoverstärker
Drive – stepper motor controller	Antrieb - Schrittmotor-Steuerungcontroller
Encoder	Encoder
Gateway	Gateway
General	Allgemeines
HMI	HMI
HMI display	HMI-Anzeige
HMI operator panel	HMI-Bediengerät
Hydraulic device	Hydraulik-Gerät
I/O	E/A
I/O analog IN/OUT	E/A analog IN/OUT
I/O analog IN	E/A analog IN
I/O analog OUT	E/A analog OUT
I/O digital IN/OUT	E/A digital IN/OUT
I/O digital IN	E/A digital IN
I/O digital OUT	E/A digital OUT
I/O function module	E/A-Funktionsmodul
Identification system	Identifikationssystem
Media converter active	Medienkonverter aktiv
Media converter passive	Medienkonverter passiv
NC	NC
NC/RC	NC/RC
PC	PC
PC board	PC-Karte
PLC	PLC
PLC board	SPS-Karte
Pneumatic device	Pneumatik-Gerät
Positioning controller	Positionier-Steuerung
Power supply	Stromversorgung
Robot control	Roboter
Sensor	Sensor
Switching device	Schaltgerät
Technology controller	Technologie-Steuerung
Valve	Ventil
Weighing or batching system	Wiege- oder Dosiersystem
Welding controller	Schweißsteuerung
Wrenching controller	Schraubersteuerung

If a manufacturer cannot assign a device to any of the above ranges, the INTERBUS Club should be notified.

### 7.1.2 CommunicationProfile

This parameter contains a system-specific communication profile code.  
 The following communication profiles apply for "Compact" implementation:

63	Implementation	Channel type	Suitability
632	Compact	Fieldbus management, Compact parameter channel	No cyclic process data, no downloading/uploading of variable lists, programs, etc.
633	Compact Process data	Fieldbus management, cyclic process data, Compact parameter channel	Parameterizable devices, only parameters, no downloading/uploading of variable lists and programs, etc.
634	Compact Process data Upload/Download protocol	Fieldbus management, cyclic process data, Compact parameter channel, Download/upload	Complex devices, downloading/uploading of variable lists
635	Compact Upload/Download protocol	Fieldbus management, Compact parameter channel, Download/upload	Complex devices, downloading/uploading of variable lists, no cyclic process data

### 7.1.3 DeviceProfile

This parameter contains information on the implemented device profile of the device.  
 Structure of the parameter:

B15 .....b12	b11 .....b8	b7 .....b4	b3 .....b0
Profile group			Version

DeviceProfile	Meaning
0000	No profile
0010	Basic profile (basis for all other profiles)
0012	Sensor/actuator
0020	DRIVECOM only process data
0021	DRIVECOM frequency inverter
0022	DRIVECOM servo
0030	Reserved
0040	Controller boards
0050	Reserved
0060	Reserved
0070	Encoder
0080	Process controllers
0090	Robot controllers
00A0	Wrenching controllers
00B0	ISO valves
00C0	Welding controllers
00D0	Operating/display units
00E0	Hydraulic devices
FFFF	More than one application profile

If a device supports more than one device profile, "FFFF<sub>hex</sub>" is entered into the DeviceProfile parameter.

## 7.2 Device diagnostics

These objects are used to send diagnostic information about the state of the device and any connected I/O devices to the application. The current diagnostic information for the device is stored in the "DiagState" object. In addition, this information can be stored in long form in the domain variable (index E800).

A history of the diagnostic information can be found in the "DiagHistory" domain variable or the "DiagHistoryLong" domain variable for the long form.

### 7.2.1 Objects

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
0018	DiagState	R	6 entries	<b>Record</b>	Current diagnostic status of the device	<b>M</b>
.1	• Seq.no.	R	2 bytes	UINT16	Unique, consecutive error number since the last power up reset or history reset	<b>M</b>
.2	• Priority	R	1 byte	UINT8	Priority of the message "1" is the highest priority	<b>M</b>
.3	• Channel/Group/Module	R	1 byte	UINT8	Channel, group or module on which the error occurred. Channel/Group/Module "FF" refers to the entire device.  For additional information, see "MoreFollows"	<b>M</b>
.4	• Code	R	2 bytes	Octet string	Error code (see below)	<b>M</b>

.5	<ul style="list-style-type: none"> <li>MoreFollows</li> </ul>	R	1 byte	Bit string 8	<p>= "00<sub>hex</sub>" For subindex 3 "Channel/Group/Module", this refers to a channel number.</p> <p>= "01<sub>hex</sub>" There is additional information about this error, which can be read via the "DiagStateLong" object.</p> <p>= "02<sub>hex</sub>" There is an additional value for diagnostics. This value can be read via the "DiagStateAddValue" object.</p> <p>= "04<sub>hex</sub>" For subindex 3 "Channel/Group/Module", this refers to a group number. So that a distinction can be made, the channel can be read in the "DiagStateChannelNumber" object as an option.</p> <p>= "08<sub>hex</sub>" This is a modular device. Subindex 3 "Channel/Group/Module" indicates the module concerned. By accessing this object (0x0018) again with the corresponding module number, the group and, if necessary, the channel can be read.</p>	<b>M</b>
.6	<ul style="list-style-type: none"> <li>Text</li> </ul>	R	Maximum 51 bytes	Visible string	The first 51 characters, maximum, of the message sent. Default: "Status OK"	<b>M</b>
E800	DiagStateLong	R		<b>Domain variable Record</b>	Current diagnostic information for the device in long form	O
Seg.1	<ul style="list-style-type: none"> <li>Seq.no.</li> </ul>				- 0018.1	D
-a						
-b	<ul style="list-style-type: none"> <li>Priority</li> </ul>				- 0018.2	D
.c	<ul style="list-style-type: none"> <li>Channel/Group</li> </ul>				- 0018.3	<b>D</b>
-d	<ul style="list-style-type: none"> <li>Code</li> </ul>				- 0018.4	D
-e	<ul style="list-style-type: none"> <li>MoreFollows</li> </ul>				- 0018.5	D
-f	<ul style="list-style-type: none"> <li>Date</li> </ul>	R	10+1 bytes	Visible string	Date on which the error occurred. YYYY-MM-DD 0000/00/00 = No date available	D
-g	<ul style="list-style-type: none"> <li>Time</li> </ul>	R	8+1 bytes	Visible string	Time at which the error occurred. hh:mm:ss 00:00:00 = No time	D

-h	• TimeofOperation	R	4 bytes	UINT32	Absolute operating hours counter reading at which the error occurred. 0 .. 4294967295 seconds About 136 years = 0 no operating hours counter available	D
-i	• ParamSet	R	2 bytes	UINT16	Current valid parameter record number at which the error occurred. 0000 = No specific parameter record number available	D
-j	• TraceData	R	2 bytes	UINT16	Start index, used to indicate relevant trace data, if necessary. = 0000 no trace data available	D
Seg.2 -a	• RecipientLength	R	2 bytes	UINT16	Length of the subsequent recipient of the message text in characters (bytes) = "00 <sub>hex</sub> " no information	D
-b	• Recipient	R	Any	Visible string	Recipient of the message	D
Seg.N -a	• Senderlength	R	2 bytes	UINT16	Length of the subsequent sender of the message text in characters (bytes) = "00 <sub>hex</sub> " no information	D
-b	• Sender	R	Any	Visible string	Sender of the message	D
Seg.M -a	• Textlength	R	2 bytes	UINT16	Length of the subsequent message text in characters (bytes)	D
-b	• Text – 0018.6	R	Any	Visible string	The message sent. Default: "Status OK"	D
E801	DiagHistory	R		<b>Domain variable Record</b>	Diagnostic information for the device in short form with history (the oldest information/lowest error number is transmitted first). The structure is the same as that of "DiagState".	O
	• DiagStateM				Oldest diagnostic status in short form	D
	• DiagStateM-1					D
	...				...	D
	• DiagStateN+1					D
	• DiagStateN				Latest diagnostic status in short form	D
E802	DiagHistoryLong	R		<b>Domain variable Record</b>	Diagnostic information for the device in long form with history (the oldest information/lowest error number is transmitted first). The structure is the same as that of "DiagStateLong".	O
	• DiagStateLongM				Oldest diagnostic status in long form	D
	• DiagStateLongM-1					D
	...					D
	• DiagStateLongN+1					D
	• DiagStateLongN				Latest diagnostic status in long form	D

0019	ResetDiag	W	1 byte	UINT8	<p>Deletes the corresponding diagnostic memory of the device and acknowledges the diagnostic message(s)</p> <p>= "00<sub>hex</sub>" no limitation for the diagnostic messages</p> <p>= "01<sub>hex</sub>" deletes the diagnostic history. This value is only supported if the "DiagHistory(Long)" object exists.</p> <p>= "02<sub>hex</sub>" deletes (and acknowledges) all unretrieved and pending DiagStates (errors).</p> <p>= "03<sub>hex</sub>" deletes (and acknowledges) the entire diagnostics</p> <p>= "04<sub>hex</sub>" deletes (and acknowledges) all DiagStates already reported via an information report</p> <p>= "05<sub>hex</sub>" deletes (and acknowledges) the most recently reported DiagState</p> <p>= "06<sub>hex</sub>" deletes (and acknowledges) the entire diagnostics (such as "03<sub>hex</sub>") and does not permit any new diagnostic messages.</p> <p>= "80<sub>hex</sub>" acknowledges without deleting</p> <p>Otherwise: reserved</p>	O
------	-----------	---	-----------	-------	--	---

001A	GetErrorRepMethod	R/W	1 byte	Bit string 8	<p>Specifies the method with which an error is to be reported to the higher-level system.</p> <p>= "01<sub>hex</sub>" switches the generation of an information report using the contents of the "DiagState" object in the event of an error on (= "1")/off (= "0").</p> <p>= "02<sub>hex</sub>" switches the generation of an I/O fault status message using the contents of the "DiagState" object in the event of an error on (= "1")/off (= "0").</p> <p>= "04<sub>hex</sub>" switches the generation of a process data status message (process data bit freely selected by the manufacturer) using the contents of the "DiagState" object in the event of an error on (= "1")/off (= "0").</p> <p>= "09<sub>hex</sub>" switches the generation of an information report using the contents of the "DiagState" object in the event of an error on (= "1")/off (= "0") and deletes the corresponding DiagState without explicit acknowledgement.</p> <p>If several report methods are selected simultaneously, the diagnostic message (and the subsequent diagnostic messages) must be stored in object 0018 until they have been read, even if one or more other diagnostic messages have already been sent via the report mechanism.</p>	O
0033	DiagStateChannelNo	R	6 entries	<b>Record</b>	Current diagnostic status of the device	D
.1	• Seq.no.	R	2 bytes	UINT16	Unique, consecutive error number since the last power up reset or history reset	D

.2	• ChannelNo	R	1 byte	UINT8	<p>If the device has grouped channels, the affected channel of the corresponding group can be entered here.</p> <p>If "DiagState" is read, this object must be kept consistent with "DiagState" until at least the end of the next read access.</p> <p>If "DiagState" is reported as an information report and if this object is relevant for this purpose, this object is also reported via an information report (chronologically after the DiagState).</p>	D
0034	DiagStateAddValue	R	6 entries	<b>Record</b>	Current diagnostic status of the device	D
.1	• Seq.no.	R	2 bytes	UINT16	Unique, consecutive error number since the last power up reset or history reset	D
.2	• AddValue	R	4 bytes	UINT32	<p>Here an accompanying value for channel diagnostics such as "Current temperature in the event limit value is exceeded" can be entered. Details: see PROFINET I&amp;M profile</p> <p>If "DiagState" is polled, this object must be kept consistent with "DiagState" until at least the end of the next read access.</p> <p>If "DiagState" is reported as an information report and if this object is relevant for this purpose, this object is also reported via an information report (chronologically after the DiagState).</p>	D
001B	TestMode	R/W	2 bytes	UINT16	<p>A manufacturer-specific code is used to switch to a test mode. Other test parameters are thereby activated. The value "0" is used to switch back to the normal operating mode. = "0000<sub>hex</sub>" normal operating mode (Default)</p>	O

The depth of the diagnostic history, i.e., the number of entries, can be freely selected by the device developer according to requirements and resources.

#### 7.2.1.1 Notification by reading the DiagState object

Available diagnostic information is reported (for example, via status bit [e.g., bit 7 in byte 0 of the process data channel] in the process data channel or via the /StatErr module error input of the fieldbus slave protocol chip) according to its priority. Several items of diagnostic information may be available at the same time. The master or a software tool can retrieve the relevant information via the parameter channel.



The current diagnostic information in DiagState(Long) remains until:

- This information has been read at least once  
and
- The cause of this diagnostic information does not exist anymore.

Only then is the next item of information (e.g., "Error removed" or "Error no. 2") made available.

The status bit(s) in the process data channel or module error input remain(s) until:

- Each item of diagnostic information has been read at least once  
and
- No more diagnostic information is available.

Only then is the current information, e.g., "Status OK", made available.

#### 7.2.1.2 Notification via information report from the DiagState object

Alternatively, errors can be reported to the master using the "Information Report" service (only once using the "DiagState" object). This function is enabled or disabled using the "GetErrorRepMethod" object. When the "GetErrorRepMethod" function is activated, each new item of diagnostic information (contents: object 0018) is reported once automatically to the master (without a request from the master).

In this case, the evaluation of the information report must be present on the fieldbus master side, e.g., as a PLC function block or HLL program.

This does not affect normal diagnostic handling of object 0018 "DiagState" (see above) (see description for "DiagState" object). This means that the first item (and subsequent items) of diagnostic information remain pending here even if they were reported via the "Information Report".

Here, the diagnostic information in "DiagState(Long)" always refers to the first error that remains pending.

The status bit(s) in the process data channel or module error input remain(s) until:

- No more diagnostic information is available.

Only then is the current information, e.g., "Status OK", reported.

#### 7.2.1.3 Classification of messages

Prio 3 (green) message, information, notification

Example: general operating message indicating that 10,000 operating hours have elapsed

Prio 2 (yellow) warning

Meaning: risk of error

Example: limit value not reached or exceeded

A warning does not require action to be taken in the device

Prio 1 (red) error (alarm)

Meaning: an error has occurred and requires a response.

An error requires action to be taken in the drive, but does not necessarily require the system to be stopped with immediate effect.

Prio 81, 82, 83 removed

Meaning: the error reported with the same number has been removed.

Messages and information are defined by the user. Warnings and errors can be predefined or defined by the user.

An outgoing message (DiagnoseStatus) overwrites the corresponding incoming message (DiagnoseStatus), if this has not yet been retrieved by the master. This reduces the volume of communication traffic without losing any information.

This results in the following sequence of priorities:

Encoding	Priority	Meaning
0x81 – highest priority	Prio 1 (red) outgoing	Error (alarm) removed
0x01	Prio 1 (red) incoming	Error (alarm)
0x82	Prio 2 (yellow) outgoing	Warning removed
0x02	Prio 2 (yellow) incoming	Warning
0x83	Prio 3 (green) outgoing	Information, message removed
0x03 – lowest priority	Prio 3 (green) incoming	Information, message

#### 7.2.1.4 Error codes

The aim of standardizing the error codes is to provide the user with a quick and easy guide for remedying problems without requiring detailed knowledge of the device.

The error code is represented as an octet string with a length of 2 bytes. It is encoded hierarchically, beginning with a rough distinction that is gradually refined.

Bit	Grouping
15 ... 12	Main groups
11 ... 8	Subgroups
7 ... 0	Details

If the device is in the "Error" state, the parameter contains a value that is not equal to 0.  
If the device is not in the "Error" state, the parameter contains the value "0".

### 7.2.1.4.1 Main groups and subgroups

#### Code (hex) Meaning

<b>0000</b>	<b>No error</b>
<b>1000</b>	<b>General error</b>
<b>2000</b>	<b>Current</b>
2100	Current on the device input side
2200	Current inside the device
2300	Current on the device output side
<b>3000</b>	<b>Voltage</b>
3100	Mains voltage
3200	Voltage inside the device
3300	Output voltage
3400	I/O supply voltage
<b>4000</b>	<b>Temperature</b>
4100	Ambient temperature
4200	Device temperature
4300	External temperature (e.g., drive)
4400	Supply temperature
<b>5000</b>	<b>Device hardware (only inside the device housing)</b>
5100	Supply, internally and through the device
5200	Controller (device application)
5300	Operating and display unit
5400	Power section
5500	Communication with add-on module
<b>6000</b>	<b>Device software</b>
6100	Internal software (firmware)
6200	Application software
6300	Data record not OK
<b>7000</b>	<b>Add-on module(s) (fixed to the device) faulty</b>
7100	Power
7200	Measuring circuit
7300	Sensor (on device)
7400	Computer circuit
7500	Communication
7600	Data memory
7700	Open circuit/Cable error
<b>8000</b>	<b>Monitoring</b>
8100	Communication
8200	Closed-loop control
8300	Torque controller
8400	Speed controller
8500	Position controller
8600	Positioning controller
8700	Synchro controller
8800	Winding controller
8900	Sensors (own device) measurement error
8A00	Actuators
8B00	Preventive maintenance required (condition monitoring)
<b>9000</b>	<b>External error</b>
<b>A000</b>	<b>Modular devices</b>
<b>F000</b>	<b>Additional functions</b>

### 7.2.1.4.2 Main groups with subgroups and details

#### Code<sub>(hex)</sub> Meaning

<b>0000</b>	<b>No error</b>
<b>1000</b>	<b>&gt;General error</b>
1800	General error (manufacturer-specific)
...	General error (manufacturer-specific)
1FFF	General error (manufacturer-specific)
<b>2000</b>	<b>Current</b>

<b>2100</b>	<b>Current on the device input side</b>
2110	Short circuit/ground fault
2120	Ground fault
2121	Ground fault, phase L1
2122	Ground fault, phase L2
2123	Ground fault, phase L3
2130	Short circuit
2131	Short circuit, phases L1 - L2
2132	Short circuit, phases L2 - L3
2133	Short circuit, phases L3 - L1
2136	Short circuit to VCC
2137	Short circuit to GND
<b>2200</b>	<b>Current inside the device</b>
2211	Current inside the device, no. 1
2212	Current inside the device, no. 2
2213	Overcurrent on startup
2214	Overcurrent during operation
2220	Continuous overcurrent
2221	Continuous overcurrent, no. 1
2222	Continuous overcurrent, no. 2
2230	Short circuit/ground fault
2240	Ground fault
2250	Short circuit
<b>2300</b>	<b>Current on the device output side</b>
2310	Continuous overcurrent
2311	Continuous overcurrent, no. 1
2312	Continuous overcurrent, no. 2
2320	Short circuit/ground fault
2330	Ground fault
2331	Ground fault, phase U
2332	Ground fault, phase V
2333	Ground fault, phase W
2340	Short circuit
2141	Short circuit, phases U - V
2142	Short circuit, phases V - W
2343	Short circuit, phases W - U
2344	Output overload
2345	Overload of initiator supply
2346	Short circuit to VCC
2347	Short circuit to GND
2350	Open cables
2360	Cable interrupt
<b>3000</b>	<b>Voltage</b>
<b>3100</b>	<b>Mains voltage</b>
3110	Mains surge voltage
3111	Mains surge voltage, phase L1
3112	Mains surge voltage, phase L2
3113	Mains surge voltage, phase L3
3120	Mains undervoltage
3121	Mains undervoltage, phase L1
3122	Mains undervoltage, phase L2
3123	Mains undervoltage, phase L3
3130	Phase failure
3131	Phase failure L1
3132	Phase failure L2
3133	Phase failure L3
3134	Phase sequence
3140	Mains frequency
3141	Mains frequency too high
3142	Mains frequency too low
<b>3200</b>	<b>Voltage inside the device</b>
3210	Surge voltage inside the device
3211	Surge voltage, no. 1
3212	Surge voltage, no. 2
3220	Undervoltage inside the device
3221	Undervoltage, no. 1
3222	Undervoltage, no. 2
3230	Charging error
<b>3300</b>	<b>Output voltage</b>

3310	Output surge voltage
3311	Output surge voltage, phase U
3312	Output surge voltage, phase V
3313	Output surge voltage, phase W
3320	Output undervoltage
3320	Armature circuit
3321	Armature circuit interrupted
3330	Field circuit
3331	Field circuit interrupted
<b>3400</b>	<b>I/O supply voltage</b>
3410	Initiator supply
3411	Initiator supply undervoltage
3412	Initiator supply not available
3413	Initiator supply surge voltage
3420	Actuator supply
3421	Actuator supply undervoltage
3422	Actuator supply not present
3423	Actuator supply surge voltage
<b>4000</b>	<b>Temperature</b>
<b>4100</b>	<b>Ambient temperature</b>
4110	Ambient overtemperature
4120	Ambient undertemperature
4130	Intake air temperature
4140	Exhaust air temperature
<b>4200</b>	<b>Device temperature</b>
4210	Device overtemperature
4220	Device undertemperature
<b>4300</b>	<b>External temperature (e.g., drive)</b>
4310	Drive overtemperature
4320	Drive undertemperature
<b>4400</b>	<b>Supply temperature</b>
4410	Supply overtemperature
4420	Supply undertemperature
<b>5000</b>	<b>Device hardware (only inside the device housing)</b>
5010	Component error
<b>5100</b>	<b>Supply, internally and through the device</b>
5110	Supply low voltage, general
5111	Supply +/- 15 V
5112	Supply + 24 V
5113	Supply + 5 V
5114	Supply + 3.3 V
5115	Supply + 2.5 V
5116	Supply + 1.2 V
5118	U8 = Manufacturer-specific
....	..... for U8 to U15 manufacturer-specific
511F	U15 = Manufacturer-specific
5120	Supply for air
5130	Supply for paint
5140	Supply for intermediate circuit
5150	Supply for initiator
5151	Short circuit
5160	Supply of I/O devices supplied by the device
<b>5200</b>	<b>Controller (device application)</b>
5210	Measuring circuits
5220	Computer circuits
5230	Communication (inside the device)
<b>5300</b>	<b>Operating and display unit</b>
<b>5400</b>	<b>Power section</b>
5410	Output stages
5420	Choppers
5430	Input stages
5440	Contactors
5441	Channel 1
...	...
5448	Channel 8
5450	Fuses
5451	S1 = L1
5452	S2 = L2
5453	S3 = L3

5454	S4 = Manufacturer-specific
....	.... for S5, S6, S7, S8
5459	S9 = Manufacturer-specific
<b>5500</b>	<b>Communication with add-on module</b>
5510	Interface no. 1
5520	Interface no. 2
<b>6000</b>	<b>Device software</b>
6010	Software reset (watchdog)
<b>6100</b>	<b>Internal software (firmware)</b>
<b>6200</b>	<b>Application software</b>
6210	PD index not available
6211	Variable no. not available
<b>6300</b>	<b>Data record not OK</b>
6301	Data record no. 1
..	And so on for 2 to 14
630F	Data record no. 15
6310	Parameter missing
6320	Parameter errors
6330	Parameter not yet initialized
6800	Configuration
6810	Data configuration double mapped
<b>7000</b>	<b>Add-on module(s) (fixed to the device) faulty</b>
<b>7100</b>	<b>Power</b>
7110	Brake chopper
7111	Brake chopper failure
7112	Brake chopper overcurrent
7113	Brake chopper wiring
7120	Motor
7121	Motor blocked
7122	Motor missing or commutation error
7123	Motor tilted
<b>7200</b>	<b>Measuring circuit</b>
<b>7300</b>	<b>Sensor (on device)</b>
7301	Sensor 1 faulty
730F	Sensor 15 faulty
7301	Tachometer faulty
7302	Tachometer polarity reversal
7303	Resolver 1 faulty
7304	Resolver 2 faulty
7305	Incremental encoder 1 faulty
7306	Incremental encoder 2 faulty
7307	Incremental encoder 3 faulty
7310	Speed
7320	Position
<b>7400</b>	<b>Computer circuit</b>
<b>7500</b>	<b>Communication</b>
7510	Serial interface no 1
7520	Serial interface no 2
<b>7600</b>	<b>Data memory</b>
7610	RAM
7620	EPROM
7630	EEPROM
<b>7700</b>	<b>Open circuit/Cable error</b>
7701	Cable 1 faulty
770F	Cable 15 faulty
7710	Sensor cable open circuit
<b>8000</b>	<b>Monitoring</b>
<b>8100</b>	<b>Communication</b>
8110	Process data monitoring
8120	Host monitoring
8121	iPD channel handshake timeout
<b>8200</b>	<b>Closed-loop control</b>
8210	System deviation, desired > actual. Deviation present longer than a specified period of time (manufacturer-specific)
8211	Maximum manipulated variable reached
8220	System deviation, desired < actual. Deviation present longer than a specified period of time (manufacturer-specific)
8221	Minimum manipulated variable reached
...	Reserved for profile-specific system errors

827f	Reserved for profile-specific system errors
8280	Manufacturer-specific system errors
...	Manufacturer-specific system errors
82FF	Manufacturer-specific system errors
<b>8300</b>	<b>Torque controller</b>
8311	Excess torque
8312	High-inertia starting
8313	Static torque
8321	Insufficient torque
8331	Torque break
<b>8400</b>	<b>Speed controller</b>
<b>8500</b>	<b>Position controller</b>
<b>8600</b>	<b>Positioning controller</b>
8611	Following error
8612	Reference limit
<b>8700</b>	<b>Synchro controller</b>
<b>8800</b>	<b>Winding controller</b>
<b>8900</b>	<b>Sensors (own device) measurement error</b>
8910	Measuring range overshoot
8920	Measuring range undershoot
<b>8A00</b>	<b>Actuators</b>
<b>8B00</b>	<b>Preventive maintenance required (condition monitoring)</b>
<b>9000</b>	<b>External error</b>
<b>A000</b>	<b>Modular devices</b>
A001	No module present
A002	Incorrect module present
A003	Replaced (the expected module was replaced by a compatible module)
A004	Sub bus more devices than expected
A010	Device error
A012	Application on module not ready
A013	Sub bus device reset
A020	Sub bus communication error
A021	Sub bus error - timeout
A022	Multiple transmission errors in sub bus
A023	Sub bus I/O data communication error
A024	Sub bus management data communication error
A030	Sub bus configuration error
A031	I/O data configuration double mapped
A040	Common errors
A041	Hardware error
A042	Firmware error
A043	Sub bus asynchronous to higher-level system
B0XX	Reserved
<b>F000</b>	<b>Additional functions</b>
F001	Delay
F002	Subsynchronous operation
F003	Lifting system
F004	Controller

Codes that are not listed are reserved.

NOTE: additional error codes are defined in the profiles.

If a manufacturer cannot assign an error to any of the above codes, the INTERBUS Club should be notified.



## 7.2.2 Trace data

Traces are consecutive recordings of specific device data. They can be used, for example, to recapitulate sequences in order to detect errors or problems, or to archive data.

Basically, there are many different types of traces. Contents, structure, scope, and handling are specified by the device manufacturer.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
001C	ControlTrace	R/W	1 byte	UINT8	Controls a recording process. "00 <sub>hex</sub> " Resets the recording and the corresponding parameters "01 <sub>hex</sub> " Starts a recording "02 <sub>hex</sub> " Stops a recording Additional details are specified by the manufacturer	O
E803	TraceBuffer	R		<b>Domain variable</b>	Contains the recorded trace data. The structure is specified by the manufacturer	O

Trace data is usually transmitted using the upload mechanism. When uploading trace data, the header should contain the basic information.

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).

## 7.3 User data management

### 7.3.1 Process data management

This section defines the behavior of process data. Behavior in the event of a timeout and fieldbus reset is defined as well as that for reading and writing process data via the parameter channel.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
001F	PDTimeout	R/W	2 bytes	UINT16	Process data monitoring time Maximum permissible time in ms (e.g., 100 ms) by which the new data must have been transmitted via the process data channel. The action that is then triggered is specified in object 0x0020 "PDTimeoutCode". FFFFh = Disabled	O
0020	PDTimeoutCode	R/W	N x 2 bytes	Array of UINT16	If the process data monitoring time elapses, the function assigned to the "process data timeout code" is carried out by the device.  This type of function can be, for example, the defined shutdown of a drive or the continuous transmission of error telegrams via a serial interface.  0000h "0" output to all output bits 0001h "1" output to all output bits 0002h Keep last valid value 0003h Accept substitute value from object 002F "PDOOUT_Subst"  0004h to 7FFFh Reserved 8000h to FFFFh Manufacturer-specific  Since different behavior may be required for each channel/each group, this parameter is set up as an array. The number of array elements is defined by the number of elements of the corresponding PDOOUT (0x0026) (PDIN 0x0025) object. N = Number of elements If only one entry is supported (written) even though the device has several elements (channels/groups), the parameter is valid for the device as a whole.	O

0024	ResetCode	R/W	N x 2 bytes	Array of UINT16	<p>Fieldbus reset code In the event of a bus reset, the function assigned to the "bus reset code" is carried out by the device.</p> <p>This type of function can be, for example, the defined shutdown of a drive or the continuous transmission of error telegrams via a serial interface.</p> <p>0000h "0" output to all output bits 0001h "1" output to all output bits 0002h Keep last valid value 0003h Acceptance of the substitute value from object 002F "PDOOUT_Subst"</p> <p>0004h to 7FFFh Reserved 8000h to FFFFh Manufacturer-specific</p> <p>Since different behavior may be required for each channel/each group, this parameter is set up as an array. The number of array elements is defined by the number of elements of the corresponding PDOOUT (0x0026) (PDIN 0x0025) object. N = Number of elements If only one entry is supported (written) even though the device has several elements (channels/groups), the parameter is valid for the device as a whole.</p>	O
0030	PF_Code	R/W	N x 2 bytes	Array of UINT16	<p>Peripheral fail code In the event of an error on the I/O of the fieldbus protocol chip, which prevents the processing of IN process data, the function assigned to the "Peripheral fail code" is carried out by the device.</p> <p>This type of function can be displayed, for example, via the /StatErr input or microprocessor WD on the protocol chips.</p> <p>0000h "0" output to all input bits 0001h "1" output to all input bits 0002h Keep last valid value 0003h Acceptance of the substitute value from object 0031 "PDIN_Subst"</p> <p>0004h to 7FFFh Reserved 8000h to FFFFh Manufacturer-specific</p> <p>Since different behavior may be required for each channel/each group, this parameter is set up as an array. The number of array elements is defined by the number of elements of the corresponding PDOOUT (0x0026) (PDIN 0x0025) object. N = Number of elements If only one entry is supported (written) even though the device has several elements (channels/groups), the parameter is valid for the device as a whole.</p>	O

0025	PDIN	R	PD length	Record	<p>IN process data The IN process data (from the device to the master) is mapped to an object. The PD bytes, like the bytes in the octet string, are counted in ascending order starting at the (top) left with 0.</p> <p>If the process data is structured (e.g., several channels), this object should also be structured and individual structure elements are accessed via the subindex.</p>	<b>M</b>
.1	• Part 1	R		Dependent	First section of the process date	<b>O</b>
.2	• Part 2	R		Dependent	Second section of the process date	<b>O</b>
...	•					
.N	• Part N	R		Dependent	Nth section of the process date	<b>O</b>
0026	PDOOUT	R/W <sub>D</sub>	PD length	Record	<p>OUT process data The OUT process data (from the master to the device) is mapped to an object. The PD bytes, like the bytes in the octet string, are counted in ascending order starting at the (top) left with 0.</p> <p>If the process data is structured (e.g., several channels), this object should also be structured and individual structure elements are accessed via the subindex.</p> <p>If the "GetExRight" object is not implemented, the status of the PDOOUT object is "Read-only". If the "GetExRight" object is implemented, the status of the PDOOUT object is "Read and write". If the value for the "GetExRight" object = 0, write access to the PDOOUT object is rejected with Error class "8" - other Error code "1" - profile-specific Additional code 0022<sub>hex</sub> "Service cannot be executed in current device state".</p>	<b>M</b>
.1	• Part 1	R		Dependent	First section of the process date	<b>O</b>
.2	• Part 2	R		Dependent	Second section of the process date	<b>O</b>
...	•					
.N	• Part N	R		Dependent	Nth section of the process date	<b>O</b>
002F	PDOOUT_Subst	R/W	PD length	Octet string	<p>OUT process data substitute Substitute value for the OUT process data (from the master to the device) in the event of an error. The PD bytes, like the bytes in the octet string, are counted in ascending order starting at the (top) left with 0. Must be implemented if the value 0003h is permitted for object 0020 "PTimeoutCode" or object 0024 "ResetCode". See relevant section.</p>	<b>D</b>

0031	PDIN_Subst	R/W	PD length	Octet string	IN process data substitute Substitute value for the IN process data (from the device to the master) in the event of an error in the connected I/O devices. The PD bytes, like the bytes in the octet string, are counted in ascending order starting at the (top) left with 0. Must be implemented if the value 0003h is permitted for object 0030 "PF_Code". See relevant section.	D						
0027	GetExRight	R/W	1 byte	UINT8	Get exclusive process data write rights This parameter can be used to request exclusive write access to the process data outputs via the parameter channel. Following positive confirmation, the data is no longer updated via the process data channel. Any modifications to the process data outputs are made via the "PDOOUT" object, which can now be read from and <u>written</u> to. The exclusive rights are reset each time a connection is aborted or the bus is reset.  <b>Note:</b> this action may have serious consequences for the connected process. This is why this object should be password protected.  = "00 <sub>hex</sub> " Output data via the PD channel = "01 <sub>hex</sub> " Output data via the parameter channel Otherwise: reserved	O						
0028	ChangePDSet	R/W	2 bytes	UINT16	Change process data settings Selects one of the following possible process data assignments defined by the manufacturer and activates it:  <table style="margin-left: 40px;"> <tr> <td>0</td> <td>Default</td> </tr> <tr> <td>0001h to 7FFFh</td> <td>Reserved</td> </tr> <tr> <td>8000h to FFFFh</td> <td>Manufacturer-specific</td> </tr> </table> aus und aktiviert diese. If the ID code and/or length code needs adapting in a new process data assignment, the changes only come into effect after the next bus reset. If possible, this configuration should be stored in the non-volatile memory and the device should be started with this configuration.	0	Default	0001h to 7FFFh	Reserved	8000h to FFFFh	Manufacturer-specific	O
0	Default											
0001h to 7FFFh	Reserved											
8000h to FFFFh	Manufacturer-specific											
003B	PDIN_Descr	R	N x 3 entries	Array of records	Process data description Description of the process data structure The description must be continuous and arranged in ascending order, starting on the left.  The number of array elements is defined by the number of elements of the PDIN (0x0025) object. N = Number of elements PDIN (0x0025)	O						

.1	• Type	R	8 bytes	Visible string	<p>Type of I/O data. The following are currently defined:</p> <ul style="list-style-type: none"> <li>• "ACK" - Response to command</li> <li>• "STATUS" - Status information</li> <li>• "DIAG" - Diagnostic information</li> <li>• "DI" - Digital IN</li> <li>• "AI" - Analog IN</li> <li>• "CNT" - Counter</li> <li>• "CmpP" - Complex protocol</li> <li>• "SubMa" - SubBus master</li> <li>• "SubSl" - SubBus slave</li> <li>• "SI" - Safety input</li> <li>• "PLC" - PLC (intelligent module data)</li> <li>• "MotSt" - Motor starter</li> <li>• "PWM" - PWM data</li> <li>• "POS" - Positioning data</li> <li>• "NO" - Number</li> <li>• "NC" - Not connected, not in use</li> <li>• "DO_F" - DO feedback</li> <li>• "AO_F" - AO feedback</li> <li>• "SO_F" - SO feedback</li> </ul> <p>The type is exactly 8 characters long. Characters not used to be completed with 0x00.</p>	D
.2	• ChNo	R	2 bytes	UINT16	<p>Channel number Number of channels of this type</p> <p><u>Note:</u> A channel always refers to a number of bits (even when there is only 1) which collectively have a meaning. Example: 4 analog inputs represent 4 channels or 16 digital inputs represent 16 channels</p>	D
.3	• ChLength	R	2 bytes	UINT16	<p>Channel length Length of the channel</p>	D
003C	PDOOUT_Descr	R	N x 3 entries	Array of records	<p>Process data description Description of the process data structure The description must be arranged in ascending order without any gaps, starting on the left.</p> <p>The number of array elements is defined by the number of elements of the PDOOUT (0x0026) object. N = Number of elements PDOOUT (0x0026)</p>	O

.1	• Type	R	8 bytes	Visible string	Type of I/O data. The following are currently defined: <ul style="list-style-type: none"> <li>• "CMD" - Commands</li> <li>• "CTRL" - Control/Configuration</li> <li>• "DO" - Digital OUT</li> <li>• "AO" - Analog OUT</li> <li>• "CNT" - Counter</li> <li>• "CmpP" - Complex protocol</li> <li>• "SubMa" - SubBus master</li> <li>• "SubSl" - SubBus slave</li> <li>• "SO" - Safety output</li> <li>• "PLC" - PLC (intelligent module data)</li> <li>• "MotSt" - Motor starter</li> <li>• "PWM" - PWM data</li> <li>• "POS" - Positioning data</li> <li>• "NO" - Number</li> <li>• "NC" - Not connected, not in use</li> </ul> <p>The type is exactly 8 characters long. Characters not used to be completed with 0x00.</p>	D
.2	• ChNo	R	2 bytes	UINT16	Channel number Number of channels of this type	D
.3	• ChLength	R	2 bytes	UINT16	Channel length Length of the channel	D

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).

**Note about output behavior following power-on:**

Following power-on, the outputs must adopt a safe (defined) state within the meaning of the Machinery Directive (inactive, type "0"). This state must be maintained until a check has been performed to see if any parameter settings that may have been stored in the module for the substitute value behavior are suitable for the system configuration (installation location) or whether new valid values have been transmitted.

### 7.3.2 Parameter channel management

This section defines the behavior of the parameter channel in the case of a timeout and a communication abort.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O						
0021	PChTimeout	R/W	2 bytes	UINT16	Communication monitoring time Maximum permissible time in ms (e.g., 1000 ms) by which the new data must have been transmitted via the parameter channel. The action that is then triggered is specified in object 0x0022 "PChTimeoutCode". FFFFh = Disabled	O						
0022	PChTimeoutCode	R/W	2 bytes	UINT16	If the communication monitoring time elapses, the function assigned to the PCP timeout code is carried out.  This type of function can be, for example, the defined shutdown of a drive or the continuous transmission of error telegrams via a serial interface.  <table style="margin-left: 40px; border: none;"> <tr> <td style="padding-right: 20px;">0</td> <td>No action</td> </tr> <tr> <td>0001h to 7FFFh</td> <td>Reserved</td> </tr> <tr> <td>8000h to FFFFh</td> <td>Manufacturer-specific</td> </tr> </table>	0	No action	0001h to 7FFFh	Reserved	8000h to FFFFh	Manufacturer-specific	O
0	No action											
0001h to 7FFFh	Reserved											
8000h to FFFFh	Manufacturer-specific											
0023	AbortCode	R/W	2 bytes	UINT16	If a connection is aborted, the function that is assigned to the connection abort code is carried out.  This type of function can be, for example, the defined shutdown of a drive or the continuous transmission of error telegrams via a serial interface.  <table style="margin-left: 40px; border: none;"> <tr> <td style="padding-right: 20px;">0</td> <td>No action</td> </tr> <tr> <td>0001h to 7FFFh</td> <td>Reserved</td> </tr> <tr> <td>8000h to FFFFh</td> <td>Manufacturer-specific</td> </tr> </table>	0	No action	0001h to 7FFFh	Reserved	8000h to FFFFh	Manufacturer-specific	O
0	No action											
0001h to 7FFFh	Reserved											
8000h to FFFFh	Manufacturer-specific											

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).



## 7.4 Device management

Field devices use increasingly complex functions, which must be suitably parameterized by the application or software tools. The following establishes a basic structure that enables the usual principal handling procedures to be standardized.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
0029	ParamSetWriteControl	R/W	1 byte	UINT8	Block parameterization control (see below)	O
002A	ConflictDictionary	R	2*N entries	<b>Record</b>	Block parameterization result (see below)	O
.1	• Index1	R	2 bytes	UINT16	Index of the 1st dependent parameter	D
.2	• Add. Code1	R	2 bytes	UINT16	Additional info on the 1st dependent parameter	D
.3	• Index2	R	2 bytes	UINT16	Index of the 2nd dependent parameter	D
.4	• Add. Code2	R	2 bytes	UINT16	Additional info on the 2nd dependent parameter	D
	• ...					
.2*N-1	• IndexN (maximum 14)	R	2 bytes	UINT16	Index of the Nth dependent parameter	D
.2*N	• Add. CodeN (maximum 14)	R	2 bytes	UINT16	Additional info on the Nth dependent parameter	D
002B	ParamSet	R/W	2 bytes	UINT16	Parameter record identification If the device has parameter records, the active one is set or displayed here. 0000h Default 0001h – FFFEh The parameter record of the device was initialized via the bus. For identification purposes, it has been assigned the freely selectable number 0001-FFFE. FFFFh The device was switched to "local mode" or manual mode and there is no guarantee that the previously loaded parameter record remains unmodified.	O
002C	ParameterMoment	R/W	2 entries	<b>Record</b>	Time of the last modification to the parameterization	O
.1	• Date	R/W	10+1 bytes	Visible string	Date YYYY/MM/DD	D
.2	• Time	R/W	8+1 bytes	Visible string	Time hh:mm:ss	D
002D	ResetParam	W	1 byte	UINT8	"01 <sub>hex</sub> " = Reset parameterization This command is used to undo all settings and replace them with default values. This also applies to passwords and other user-defined settings.  "02 <sub>hex</sub> " = Reset application parameterization This command is used to undo just the application parameters and replace them with default values.	O

002E	Checksum	R	4 bytes	UINT32	<p>Unique, device-specific generated checksum (e.g., CRC16), which ensures the integrity of the parameter data. The security mechanism used involves all the device parameter data. A "modification counter" is also possible.</p> <p>The checksum is regenerated each time a parameter is modified and enables the user to check whether all parameter settings are still unchanged. To do this, the current checksum must be compared with the checksum provided during parameterization.</p>	O
------	----------	---	---------	--------	---	---

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).

#### 7.4.1 Block parameterization

Block parameterization serves to enable the joint transfer of interdependent parameters. Block parameterization is initiated via the "Write control = 01" parameter and terminated with the "Write control = 00" parameter. Any conflicts that occur are stored in the "Conflict dictionary" parameter.

If an attempt is made to parameterize dependent parameters individually, this may result in the following error:

"Dependency of other parameter not taken into consideration".

The dependent parameter(s) can be found in the conflict dictionary. The parameter which was the subject of the recent write attempt is located in the first position.

#### Result(-) when an attempt is made to write to objects that may only be modified in conjunction with other objects due to dependency

Error class "8" other, error code "1" profile-specific

Additional code	Meaning
0x0040 DependencyIgnored	Collision with other values Dependency ignored

##### 7.4.1.1 Write control

The transition between individual parameterization and block parameterization is initiated by this parameter.

Write control = "00<sub>hex</sub>": individual parameterization

Write control = "01<sub>hex</sub>": block parameterization

The following actions are carried out when the parameter content is modified:

**Write control = "00<sub>hex</sub>" -> write control = "01<sub>hex</sub>":**

- Block parameterization is initiated
- No individual parameterization
- "Conflict dictionary" parameter is reset

**Write control = "01<sub>hex</sub>" -> write control = "00<sub>hex</sub>":**

- Block parameterization is terminated
- Individual parameterization again from now on

- Compatibility is checked:
  - If compatible:
    - The parameter contents are accepted
    - Write access to the "Write control" parameter is acknowledged positively.
  - If incompatible:
    - The old contents of all the parameters required for block parameterization remain in effect
    - "Conflict dictionary" parameter is set
    - Write access to the "Write control" parameter is acknowledged negatively as follows.

Error class "8" other, error code "1" profile-specific

Additional code	Meaning
0x0040    DependencyIgnored	Collision with other values, dependency ignored As a rule, dependent values were not taken into consideration.

#### 7.4.1.2 Conflict dictionary

The parameter contains the indices and error messages (additional code) for the parameters involved in the conflict. The "Conflict dictionary" parameter must be updated in the case of:

- Negative acknowledgment of "Block write control" and
- Individual parameterization of dependent parameters, if the dependency of the other parameter is not taken into consideration.
- However, it can also be updated after each write access inside and/or outside the "Block parameterization" state.

#### 7.4.2 Parameter record identification

This parameter is used to identify the device parameter record that currently applies. For example, this can be a welding program or a recipe.

Parameter records can be predefined by the manufacturer or loaded by the user depending on the device using individual or block parameterization or "Download-Write" services.

If the device cannot save the received device parameter values so that they are not lost in the event of a power failure, the device automatically sets the parameter record identification to "0000" when the mains voltage is switched on. The device user can evaluate this information and reinitialize accordingly.

0000h	The parameter record of the device has not yet been initialized via the bus. The default parameter record is valid.
0001h - FFFEh	The parameter record of the device was initialized via the bus. It received a freely selectable ID number (1 - FFFEh) for identification purposes.
FFFFh	The device was or is switched to "local mode" and there is no guarantee that the previously loaded parameter record remains unmodified.

The device user can enter any value between 0 and FFFFh.

### 7.4.3 Password protection

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
001D	Password	W	Maximum 40 bytes	Octet string	This command is used to transmit a password (e.g., "Superuser") which is valid for subsequent actions. Where access is attempted without a password or with an incorrect password, this must be rejected with reference to the absent access rights. The password is disabled if: <ul style="list-style-type: none"> <li>• Communication is aborted</li> <li>• The bus is stopped (bus reset)</li> <li>• No password is written (string length is "0")</li> <li>• An incorrect password is written</li> </ul> No access protection can be provided for this object.	O
001E	SetPassword	W	3 entries	<b>Record</b>	One or more passwords are set for a specific index using this object. The password can only be deleted using the "ResetParam" command. The manufacturer may have defined a default password for any object. To prevent passwords from being modified, this object can itself be protected. For all intents and purposes, it is the "Superuser" which protects the "SetPassword" object. Passwords may have been preset by the manufacturer. These are not necessarily deletable.	O
	• Index	W	2 bytes	UINT16	"FFFF" as an index sets the password for all objects.	D
	• Add/Replace	W	1 byte	Bit string 8	0Xh = All existing passwords are replaced 1Xh = Add to existing passwords X1h = Applies for read access X2h = Applies for write access X3h = Applies for read and write access	D
	• Password	W	Maximum 40 bytes	Visible string	Password for the "Index" object "Replace" and no password overrides password protection.	D

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).

Additional code (hex)	Meaning
00B1	The password cannot be replaced (deleted).
00B2	The password cannot be added (too many passwords).
00B3	The password cannot be assigned for the desired type of access.

## 7.5 Multilingual capacity

The "Language" object can be used to read the current language selected and select the language.

The "LanguageAvailable" domain variable indicates which languages are available. The first entry contains the default language.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O	Example
0017	Language	R/W	2 entries	<b>Record</b>	Object for the device language selection The currently valid language may be accessed and modified here.	<b>M</b>	
.1	• LanguageCode	R/W	5+1 bytes	Visible string	Language code according to ISO 639-1 and country code according to ISO 3166-1-Alpha-2 code, separated by a "-" if desired e.g., en[-us] If a country code is not being used, 0x00 will be entered. The language code of the language to be selected is entered here. Following a positive response, all text is output in the selected language.	<b>M</b>	en
.2	• NameLanguage	R	Maximum 50 bytes	Visible string	The text string for the language that is currently valid can be read here. The name changes as soon as a new number is entered.	<b>M</b>	English
E804	LanguageAvailable	R		<b>Domain variable Record</b>	Object for device language selection – displays all available languages Language name, already in the relevant language	O	
Seg.1 -a	• LanguageCode1 (default)	R	5+1	Visible string	Language code according to ISO 639-2 (default)	D	En
-b	• NameLanguage1 (default)	R	Maximum 50	Visible string	Text string for the 1st language available (default)	D	English
Seg.2 -a	• LanguageCode2	R	5+1	Visible string	Language code according to ISO 639-2 of the 2nd language available	O	De
-b	• NameLanguage2	R	Maximum 50	Visible string	Text string for the 2nd language available	O	Deutsch
Seg.3 -a	• LanguageCode3	R	5+1	Visible string	Language code according to ISO 639-2 of the 3rd language available	O	Fr
-b	• NameLanguage3	R	Maximum 50	Visible string	Text string for the 3rd language available	O	Français
Seg.4 -a	• LanguageCode4	R	5+1	Visible string	Language code according to ISO 639-2 of the 4th language available	O	Es
-b	• NameLanguage4	R	Maximum 50	Visible string	Text string for the 4th language available	O	Español

Seg.N	• LanguageCodeN	R	5+1	Visible string	Language code according to ISO 639-2 of the Nth language available	O	It
-a							
-b	• NameLanguageN	R	Maximum 50	Visible string	Text string for the Nth language available	O	Italiano

If an error occurs when writing/reading objects, e.g., invalid values are transmitted, access must be rejected using the appropriate error codes (see "List of permitted error codes" section).

The "LanguageCode" (subindex 1) subobject contains the language code according to ISO 639-1 and country code according to ISO 3166-1-Alpha-2 code, separated by a "-" if desired. If a country code is not being used, 0x00 is entered.

Writing this subobject selects the language. This language then appears in the "NameLanguage" subobject.

For write access to the entire "Language" object, the entry for the "NameLanguage" subobject is ignored, as the "NameLanguage" subobject has "Read-only" status.

Each segment in the "LanguageAvailable" domain variable contains both the "LanguageCode" and the "NameLanguage" subobjects of the available language. The individual read services of the "Upload-Read" macro service, therefore, (almost) always have different lengths, but each language entry is always transmitted in its entirety in one segment. This results in easy handling.

## 7.6 Modular devices

In addition to compact devices, there are also numerous modular devices. Their handling should also be taken into account in this profile. A modular device is defined as follows:

- It has exactly one communication access module (header).
- It has N modules, whereby  $0 \leq N \leq 253$ .
- A module does not necessarily have to be active/present.
- Each module has its own independent set of objects.  
This object area does not need to be the object area described in this basic profile, but is determined instead by the subsystem.

The module structure of a modular device is described in the object 0x0036 "DeviceStructure".

The Invoke ID, which was previously not used and only available for reasons of compatibility, is used to address modules on a modular device. For this reason, the name of this parameter has also been changed to "Module number".

However, the handling of this parameter has not changed. The module number of the indication is still entered in the response.

The behavior for compact devices has not changed either. Any module number may be used here as usual. However, for reasons of consistency, we recommend selecting module number "00".

A modular device as a whole can always be addressed using the module number 0xFF. The individual modules can be addressed via the corresponding module number.

The module number should begin with 1. The head of a modular device does not necessarily have to have the module number "1".

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
0035	NoOfModules	R	1 byte	UINT8	Number of the last module that is currently present. (Number of modules including dummy.) The "Module number" parameter is of no significance for this object. <ul style="list-style-type: none"> <li>• If this object does not exist, the device is a compact device and not a modular device.</li> <li>• If "0" is returned, the device is a modular device in which no modules are installed (inserted) at present.</li> </ul>	<b>D</b>
0036	DeviceStructure	R/W	4 entries	Record	The structure of the modular device and the mapping of the module process data to the device process data is specified here. The start and length of process data mapping can be called for each module using its module number.  This object is also suitable for describing compact devices. Mandatory object for modular devices.	<b>D</b>
.1	<ul style="list-style-type: none"> <li>• PDINOffset</li> </ul>	R/W	2 bytes	UINT16	Bit offset in the IN process data channel	<b>D</b>
.2	<ul style="list-style-type: none"> <li>• PDINLength</li> </ul>	R/W	2 bytes	UINT16	Bit length in bits in the IN process data channel	<b>D</b>



.3	• PDOUOffset	R/W	2 bytes	UINT16	Bit offset in the OUT process data channel	<b>D</b>
.4	• PDOULength	R/W	2 bytes	UINT16	Bit length in bits in the OUT process data channel	<b>D</b>

## 7.7 Object description

During startup and servicing, not only is it important to have the requisite knowledge of the desired parameterization; the actual parameterization of the device must also be known. This requires knowledge of the existing user objects.

These objects and their meaning can be read using the two profile objects below.

Index (hex)	Object name	R/W	Length	Data type	Meaning	M/O
0038	ObjDescrReq	R/W	2 entries	<b>Record</b>	Object whose description is requested.	O
	• Index	R/W	2 bytes	UINT16		D
	• SubIndex	R/W	1 byte	UINT8		D
0039	ObjDescr	R	16 entries	<b>Record</b>	Description of the object whose index was requested. Automatically incremented to next valid index/subindex on next access attempt. Empty objects and indices that are not present are skipped.	O
.1	• Index	R	2 bytes	UINT16	Object number e.g., 0x6051	D
.2	• SubIndex	R	1 byte	UINT8	Substructure number of an object e.g., 0x00	D
.3	• ObjectCode	R	1 byte	UINT8	As specified under "Data objects" e.g., 0x07 - simple variable	D
.4	• IndexOfType	R	1 byte	UINT8	As specified under "Data types" e.g., 0x0A – OctetString	D
.5	• Length	R	1 byte	UINT8	Length of the (sub)object in bytes e.g., 0x04	D
.6	• UnitText	R	5+1 bytes	Visible string	Value unit – 0x00 scheduled e.g., rpm	D
.7	• UnitCode	R	1 byte	UINT8	Unit code of the value according to the "Indices of physical values" table  e.g., 0x11	D
.8	• UnitCodeExp	R	1 byte	UINT8	Unit exponent code of the value according to "Indices of physical values" table The unit exponent code specifies the decimal power of the unit value. The unit exponent code has the value range 127 to 128. Example <ul style="list-style-type: none"> <li>• Unit index 0 for 10<sup>0</sup></li> <li>• Unit index 3 for 10<sup>3</sup></li> <li>• Unit index -3 for 10<sup>-3</sup></li> </ul> etc. The listed unit exponent codes for SI-compatible units (unit exponent code < 64) serve as examples only. The unit indices for other SI compatible prefixes (Pico, etc.) are generated in a similar way. Unit exponent codes greater than +64 have a special significance which must be determined from the table.	D

					These units can, for example, take the form of the units day, hour or minute or be non-SI compatible units, such as Fahrenheit.  E.g., 0x73	
.9	• Offset	R	2 bytes	INT16	Offset of the corresponding value This must be added to the process data to calculate the measured value, taking the resolution into account.  Example: the process data value is 12345, the offset is -15000, this results, taking the above resolution into account, in a measured value of -1.77 V	D
.10	• RDR	R	2 bytes	UINT16	<b>ResolutionDimensionRange</b>  Resolution = $\frac{RDR}{RNR}$  Dimension range of the resolution related to the unit in which the size is displayed and within which the process data can fluctuate. Is used for calculating the resolution. Example: 10 (V) Resolution = 10 V/3000 = 3.333 mV	D
.11	• RNR	R	2 bytes	INT16	<b>ResolutionNumberRange</b>  Resolution = $\frac{RDR}{RNR}$  Number range of the resolution related to the dimension range in which the process data can fluctuate. Is used for calculating the resolution. Example: 3000 Resolution = 10 V/3000 = 3.333 mV	D
.12	• Access Rights	R	1 byte	Bit string 8	Bit 0 = WriteProtected (1=active) Bit 1 = ReadProtected (1=active) Bit 2 = Write_OnlyWithPassword (1=active) Bit 3 = Read_OnlyWithPassword (1=active) Bit 4 - 7 Reserved	D
.13	• DisplayFormat	R	1 byte	UINT8	0x00 – undefined 0x01 - binary 0x02 - unsigned decimal 0x03 - signed decimal 0x04 - hexadecimal 0x05 - text 0x06 - float 0x07 - time 0x08 – 0xFF reserved	D
.14	• Min.	R	“Length” byte	According to data type	Lower permitted limit of a value. The length is based on ".5 length" of the object. For all string variables, there is no minimum value. In this case, the length is 0. E.g., -500	D
.15	• Max	R	“Length” byte	According to data type	Upper permitted limit of a value. The length is based on ".5 length" of the object. For all string variables, there is no minimum value. In this case, the length is 0. E.g., 500	D
.16	• Symbol	R	M byte	Visible string	Name of the object. The length can be a maximum of 58 bytes. maxM=58-2xLength-22 E.g., set speed – 0x00 scheduled	D

E805	ObjDescrLong	R	N segments	<b>Domain variable</b>	Description of all existing objects (index/subindex) One description is transmitted per segment. Strings are completed with 0x00.	O
Seg.1	ObjDescr1	R	58 bytes	Record	See above	D
...	...	...	...	...	...	
Seg.N	ObjDescr1	R	58 bytes	Record	See above	D

If an object has subindices, these are arranged after the index.

If the object 0x0038 "ObjDescrReq" is used to request an index/subindex that is not available, an error message is issued as follows: error class 8 (application), error code 1 (profile-specific), additional code 0x0030 (value range of parameter has been exited). The description of the next available index/subindex is then provided in object 0x0039 "ObjDescr".

After the last index/subindex, this process starts again from the beginning.

The auto-increment function of object 0x0039 can easily be used by an application to read all available indices/subindices and thereby obtain a complete overview of all available objects in a device. (After an initial read access, the description of the next available index/subindex is provided for the next read access).

No description has to be saved for basic profile objects, because this description (including subindices) is already known across the entire system as a result of the basic profile. Object 0x0039 is assigned the index only. The remaining entries are not available (user data length: 2 bytes). This means that the availability of optional objects can be indicated. A basic profile object must therefore not be ignored.

Process data objects 0x0025 and 0x0026 are an exception, since their availability and function are process-specific, but their structures are manufacturer-specific. As they can only be determined in this way, they must be described in full.

If object 0x0039 is not available, the process data objects 0x0025 and 0x0026 are specified by default as bit strings of corresponding length.

**"Indices of physical values" table**

Physical_value	Value index	Unit	Unit index (incomplete)
	0	Without dimension	0
Length	1	Meter	0
Area	2	Square meter	0
Volume	3	Cubic meter	0
Time	4	Second	0
		Minute	70
		Hour	74
		Day	77
		Half-waves	78
		Millisecond	-3
		Microsecond	-6
Force	6	Newton	0
Real power	9	Watt	0
Apparent power	10	Volt-ampere	0
Rotational speed	11	1/second	0
		1/minute	73
		1/hour	74
Angle	12	Radiant	0
		Second	75
		Minute	76
		(Old) degree	77
		New degree	78
Speed	13	Meter/second	0
		Millimeter/second	-3
		Millimeter/min.	79
		Meter/min.	80
		Kilometer/min.	81
		Millimeter/hour	82
		Meter/hour	83
		Kilometer/hour	84
Torque	16	Newton meter	0
Temperature	17	Kelvin	0
		Degrees_Celsius	94
		Degrees_Fahrenheit	95
Electric_voltage	21	Volt	0
Electric_current	22	Amps	0
Electric_resistance	23	Ohm	0
Relation	24	Percent	0
Frequency	28	Hertz	0
Steps	32	Steps	0
Encoder resolution	33	Increments/rotation	0
Flow rate	34	Cubic meter/second	0
		Liter/second	-3
		Milliliter/second	-6
		Cubic meter/minute	85
		Liter/minute	86
		Milliliter/minute	87
Acceleration	35	1/second <sup>2</sup>	0

The unit index corresponds to the exponent (base 10) as standard

E.g.,

UnitCodeExp -6 for  $10^3$  => kilo

UnitCodeExp -6 for  $10^0$  => without

UnitCodeExp -6 for  $10^{-3}$  => m

UnitCodeExp -6 for  $10^{-6}$  =>  $\mu$

UnitCodeExp -9 for  $10^{-9}$  => n

## 8 Appendix A

### 8.1 Definition of terms

#### Substitute values

If the optional PCP objects are not implemented, the device behaves according to the substitute value defined for this parameter.

#### Error message

An error message is returned if a service cannot be executed.

#### Function units, definition of

The device function is described by a function unit. The function is controlled and parameterized via the inputs. In addition, internal signals or internal parameters can affect the function. The function output can be linked to the inputs of other functions or made accessible via the bus.

Function unit

Input:

Internal signals

Actions

External signals

Parameter (services, process data)

PCP objects

Function output:

Internal variable

PCP objects

#### Device parameters

Default values for all device parameters are defined in this profile.

#### Device profile

The device profile specifies the application functions that are to be made visible via communication. The application functions are mapped to the communication process by defining the following:

- The communication profile
- The interactions between application functions which are executed via the communication system
- The PCP services used and the PCP objects that are manipulated using them

Mapping makes the behavior of the application transparent. The application profile definitions ensure interoperability in an area of application, provided that the device properties used support this. In addition, the profile defines device properties which are important for the user. A distinction is made between mandatory, optional, and manufacturer-specific device functions and parameters. If the user uses only mandatory functions or parameters, devices can be replaced, provided this is supported by the device properties and settings used. In terms of communication, the devices can always be replaced by others with the same parameters, regardless of their function.

#### Index, subindex

The index is used to address a parameter (PCP object). The subindex is used within a parameter, which is created as a structure, to address a subparameter (element of a PCP object).

#### Communication interface

The communication interface consists of a process data channel and a parameter channel.

#### Communication profile

The communication profile restricts or classifies the degree of freedom included in the specification of the transmission medium, according to the application or device group. The communication profile defines PCP services and parameters, which are marked as optional in the specification. All optional functions and parameters that are not specified in the communication profile remain optional. Mandatory services and parameters are binding even if they are not specified in the profile. Value ranges for attributes and parameters are also specified in the profile.

### Communication reference

Every communication relationship between two devices is configured regardless of when it is used. The configuration is stored in every device in a communication relationship list (CRL). An application process identifies the communication relationship via a local communication reference. This is then used to address the communication partner.

### Parameter channel

The parameter channel can be used to access all PCP objects. Parameter channel services provide acknowledged access to device parameters, i.e., access to a device parameter is confirmed by the device.

### Mandatory range

The mandatory range is the value range in which a parameter can always be set, if it is implemented.

### Process data description

If the device transmits or receives several items of process data in parallel, the structure and meaning of this process data must be described by the manufacturer.

### Process data channel

The process data channel enables fast transmission of process data. Data is transmitted in an unacknowledged way and at regular intervals (equidistant) via the process data channel. Process data can be read and written.

The direction of the process data is regarded as from the bus; i.e.,

- OUT process data is data which is transmitted from the control system to the device. The device reads this data from the process data channel and transfers it to the process, depending on the function.
- IN process data is data which is transmitted from the device to the control system. The device writes this data to the process data channel and thereby transmits it to the control system.

### VFD object

The virtual field device (VFD) is an abstract model for describing the data and the behavior of an automation device from the point of view of its communication partner. The VFD model is based on the VFD object. The VFD object contains all objects and object descriptions that can be used by a communication partner through a service. The object descriptions are stored in an object dictionary. Each VFD has one object dictionary.

### State machine

In this profile, some functions are described using a state machine. A state refers to a specific internal and external response. It can only be exited by defined events. These events are assigned corresponding state transitions. Actions can be executed in a transition. This modifies the state response. When the transition is complete, the current state changes to the next state.

## 8.2 Symbols and abbreviations

### Network-specific abbreviations

ALI	Application layer interface
CRL	Communication relationship list

---

CRL header	Header of the communication relationship list
CR	Communication reference
m	Mandatory
MAP	Manufacturing automation protocol
o	Optional
OD	Object dictionary
PCP	Peripherals communication protocol
PCh	Parameter channel
PD	Process data
PMS	Peripherals message specification
S-OD	Static object dictionary
ST-OD	Static type dictionary
VFD	Virtual field device
.con	Confirmation primitive
.ind	Indication primitive
.req	Request primitive
.res	Response primitive

## 9 Appendix B

### 9.1 Translation table for the object names

Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
0001		VendorName	Herstellername	Vendor name	Nombre del fabricante	Nom du fabricant	Nome del produttore	厂商名称
0002		VendorID	Herstellerkennung	Vendor ID	Identificación del fabricante	Identificateur du fabricant	Identificativo produttore	厂商标识
0003		VendorText	Herstellertext	Vendor text	Texto del fabricante	Texte du fabricant	Testo produttore	厂商文本
0004		DeviceFamily	Gerätefamilie	Device family	Familia de dispositivos	Famille d'appareils	Famiglia di dispositivi	设备系列
0005		Capabilities	Eigenschaften	Capabilities				
0006		ProductFamily	Produktfamilie	Product family	Familia de productos	Famille de produits	Famiglia di prodotti	产品系列
0007		ProductName	Produktname	Product name	Nombre del producto	Nom du produit	Nome del prodotto	产品名称
0008		SerialNo	Seriennummer	Serial number	Número de serie	Numéro de série	Numero di serie	序列号
0009		ProductText	Produkttext	Product text	Texto de producto	Texte du produit	Testo prodotto	产品文本
000A		OrderNumber	Bestellnummer	Order number	Número de pedido	Numéro d'ordre	Numero d'ordinazione	订货号
000B		HardwareVersion	Hardware-Version	Hardware version	Versión del hardware	Version du matériel	Versione hardware	硬件版本
	.1	BuildDate	Hardware-Version - Herstelldatum	Hardware version - build date	Versión del hardware - fecha de fabricación	Version du matériel - date de fabrication	Versione hardware - data di produzione	硬件版本 - 创建日期
	.2	Version	Hardware-Version - Versionskennung	Hardware version - version	Versión del hardware - identificación de versión	Version du matériel - identificateur de version	Versione hardware/ID versione	硬件版本 - 版本标识
000C		FirmwareVersion	Firmware-Version	Firmware version	Versión del firmware	Version du microprogramme	Versione firmware	固件版本
	.1	BuildDate	Firmware-Version - Datum	Firmware version - build date	Versión del firmware - fecha	Version du microprogramme - date	Versione firmware - data	固件版本 - 日期
	.2	Version	Firmware-Version - Versionskennung	Firmware version - version	Versión del firmware - identificación de	Version du microprogramme - identificateur de	Versione firmware/ID versione	固件版本 - 版本标识



Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
					versión	version		
000D		PChVersion	Parameterkanal-Version	Parameter channel version	Versión del canal de parámetros	Version du canal de paramètres	Versione PCh	参数通道版本
	.1	BuildDate	Parameterkanal-Version - Datum	PCP version - build date	Versión de canal de parámetros - fecha	Version du canal de paramètres - date	Versione PCP - data	参数通道版本 - 日期
	.2	Version	Parameterkanal-Version - Versionskennung	PCP version - version	Versión de canal de parámetros - identificación de versión	Version du canal de paramètres - identificateur de version	Versione PCP - ID versione	参数通道版本 - 版本标识
000E		CommProfile	Kommunikationsprofil	Communication profile	Perfil de comunicación	Profil de communication	Profilo di comunicazione	通信概要
000F		DeviceProfile	Geräteprofil	Device profile	Perfil del dispositivo	Profil	Profilo dispositivo	设备概要
0010		Reserved	Reserviert	Reserved	Reservado	Réservé	Riservato	保留
0011		ProfileVersion	Profil-Version	Profile version	Versión del perfil	Version du profil	Versione profilo	概要版本
	.1	BuildDate	Profil-Version - Datum	Profile version - build date	Versión del perfil - fecha	Version du profil - date	Versione profilo - data	概要版本 - 日期
	.2	Version	Profil-Version - Versionskennung	Profile version - version	Versión del perfil - identificación de versión	Version du profil - identificateur de version	Versione profilo - ID versione	概要版本 - 版本标识
0012		VendorURL	Hersteller URL	Vendor URL	URL del fabricante	URL du fabricant	URL venditore	厂商URL
0013		DeviceDescFile	Gerätebeschreibungsdatei	Device description file	Archivo de descripción del dispositivo	Fichier de description d'appareil	File descrizione dispositivo	设备描述文件
0014		Location	Einbauort	Location	Lugar de montaje	Emplacement de montage	Posizione di montaggio	安装地点
0015		EquipmentIdent	Betriebsmittel-Kennzeichen	Equipment identifier	Identificación del equipo	Identificateur de matériel	Identificatore dispositivo (BMK)	设备标识符
0016		AppDeviceAddr	applikationsspezifische Geräteadresse	Application device address	Dirección del dispositivo específica de la aplicación	Adresse d'appareil spécifique à l'application	Indirizzo dispositivo per la specifica applicazione	应用特定的设备地址
0017		Language	Sprache	Language	Idioma	Langue	Lingua	语言
	.1	LanguageCode	Sprache - Ländercode	Language - language code	Idioma - código del país	Langue - code pays	Lingua - codice paese	语言 - 国家代码
	.2	NameLanguage	Sprache - Name	Language - name	Idioma - nombre	Langue - nom	Lingua - nome	语言 - 名称

Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
0018		DiagState	Diagnosestatus	Diagnostic state	Estado de diagnóstico	État du diagnostic	Stato diagnostico	诊断状态
	.1	Seq.no.	Diagnosestatus - Lfd. Nr.	Diagnostic state - cons. no.	Estado de diagnóstico - nº de orden	État du diagnostic - nº de série	Stato diagnostico - numero d'ordine	诊断状态 - 连续编号
	.2	Priority	Diagnosestatus - Priorität	Diagnostic state - priority	Estado de diagnóstico - prioridad	État du diagnostic - priorité	Stato diagnostico - priorità	诊断状态 - 优先级
	.3	Channel/Group/Module	Diagnosestatus - Kanal/Gruppe/Modul	Diagnostic state - channel/group/module	Estado de diagnóstico - canal/grupo/módulo	État du diagnostic - canal/groupe/module	Stato diagnostico - canale/gruppo/modulo	诊断状态 - 通道/组/模块
	.4	Code	Diagnosestatus - Code	Diagnostic state - code	Estado de diagnóstico - código	État du diagnostic - code	Stato diagnostico - codice	诊断状态 - 代码
	.5	MoreFollows	Diagnosestatus - Zusatzinformationen	Diagnostic state - more follows	Estado de diagnóstico - información adicional	État du diagnostic - informations supplémentaires	Stato diagnostico - informazioni aggiuntive	诊断状态 - 附加信息
	.6	Text	Diagnosestatus - Text	Diagnostic state - text	Estado de diagnóstico - texto	État du diagnostic - texte	Stato diagnostico - testo	诊断状态 - 文本
0019		ResetDiag	Diagnosemeldungen quittieren	Reset diagnostic messages	Confirmar mensajes de diagnóstico	Acquitter les messages de diagnostic	Consenso messaggi diagnostici	应答诊断信息
001A		GetErrorRepMethod	Meldemethode für Störung	Get error report method	Método de avisar un fallo	Méthode de signalisation des perturbations	Metodo avvertimento guasti	故障提示方法
001B		TestMode	Testbetrieb	Test mode	Servicio de prueba	Mode d'essai	Modalità di prova	试运行
001C		ControlTrace	Steuerung der Protokollierung	Control trace	Mando del registro	Commande de la journalisation	Controllo protocollo	记录控制
001D		Password	Passwort	Password	Contraseña	Mot de passe	Password	密码
001E		SetPassword	Passwort setzen	Set password	Establecer contraseña	Définir le mot de passe	Impostare password	设置密码
001F		PDTimeout	Prozessdaten-Überwachungszeit	Process data timeout	Tiempo de monitorización de los datos de proceso	Temps de surveillance des données de process	Tempo di monitoraggio dei dati di processo	过程数据监视时间
0020		PDTimeoutCode	Prozessdaten-Überwachungs-Code	Process data timeout code	Código de monitorización de los datos de proceso	Code de surveillance des données de process	Codice di monitoraggio dei dati di processo	过程数据监视代码

Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
0021		PCPTimeout	Parameterkanal-Überwachungszeit	PCP timeout	Tiempo de monitorización del canal de parámetros	Temps de surveillance du canal de paramètres	Tempo di monitoraggio dei dati di processo	参数通道监视时间
0022		PCPTimeoutCode	Parameterkanal-Überwachungs-Code	PCP timeout code	Código de monitorización del canal de parámetros	Code de surveillance du canal de paramètres	Codice di monitoraggio PCP	参数通道监视代码
0023		AbortCode	Verbindungsabbruch-Code	Connection abort code	Código de interrupción de comunicación	Code d'interruption de la connexion	Codice per interruzione connessione	连接中断代码
0024		IBSResetCode	Reset-Code	INTERBUS reset code	Código de restablecimiento	Code de réinitialisation	Codice di reset	复位代码
0025		PDIN	Prozesseingangsdaten	Process input data	Datos de proceso de entrada	Données de process entrantes	Dati ingresso processo	过程输入数据
0026		PDOOUT	Prozessausgangsdaten	Process output data	Datos de proceso de salida	Données de process sortantes	Dati uscita processo	过程输出数据
0027		GetExRight	Exklusive Schreibrechte anfordern	Get exclusive process data write rights	Solicitar derechos de escritura exclusivos	Demander les autorisations d'accès en écriture exclusives	Richiedere diritto di scrittura esclusivo	请求独享写权限
0028		ChangePDSet	Prozessdatenzuordnungen einstellen	Change process data settings	Ajustar las asignaciones de datos de proceso	Régler les affectations des données de process	Impostare assegnazione dati di processo	设置过程数据分配
0029		ParamSetWriteControl	Parametersatz-Schreibsteuerung	Parameter set write control	Control de escritura de juego de parámetros	Commande d'écriture du jeu de paramètres	Controllo scrittura set parametri	参数组写控制
002A		ConflictDictionary	Konflikt-Verzeichnis	Conflict dictionary	Directorio de conflicto	Répertoire des conflits	Elenco conflitti	冲突目录
002B		ParamSet	Parametersatzkennung	Parameter set identification	Identificación de juego de parámetros	Identificateur de jeu de paramètres	Identificatore set parametri	参数组标识
002C		ParameterMoment	Zeitpunkt der letzten Parametrierungsänderung	Moment of last parameterization modification	Hora y fecha del cambio de parametrización más reciente	Moment de la dernière modification du paramétrage	Ultima modifica parametro	最后一次参数化更改的时间点
002D		ResetParam	Parametrierung zurücksetzen	Reset parameterization	Restablecer parametrización	Réinitialiser le paramétrage	Reset parametrizzazione	复位参数化
002E		CheckSum	Prüfsumme	Checksum	Suma de comprobación	Somme de contrôle	Somma di controllo	检查和
002F		PDOOUT_Subst	Ersatzwert Prozessausgangsdaten	Process output data	Valor de reserva	Valeur de	Valore sostitutivo per	过程输出数据替换值

Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
				substitute	para datos de proceso de salida	remplacement des données de process sortantes	dati di processo in uscita	
0030		PF_Code	Peripherie-Fehler-Auswahlcode	Peripheral fail code	Código de selección de fallo en periferia	Code de sélection de l'erreur périphérique	Codice selezione errore periferia	外设错误选择代码
0031		PDIN_Subst	Ersatzwert Prozesseingangsdaten	Process input data substitute	Valor de reserva para datos de proceso de entrada	Valeur de remplacement des données de process entrantes	Valore sostitutivo per dati di processo in ingresso	过程输入数据替换值
0032		IBS_ID	INTERBUS Identifikation/ reserviert	INTERBUS identification/ reserved	Identificación de INTERBUS/ reservado	Identification INTERBUS/ réservé	Identificazione INTERBUS/ riservato	INTERBUS识别/ 保留
	.1	ID-Code	INTERBUS Identifikation - ID-Code	INTERBUS identification - ID code	Identificación de INTERBUS - código ID	Identification INTERBUS - code d'identification	Identificazione INTERBUS - codice ID	INTERBUS识别- ID代码
	.2	PDLength	INTERBUS Identifikation - Prozessdatenlänge	INTERBUS identification - process data length	Identificación de INTERBUS - longitud de datos de proceso	Identification INTERBUS - longueur des données de process	Identificazione INTERBUS - lunghezza dati di processo	INTERBUS识别 - 过程数据长度
0033		DiagStateChannelNo	Diagnosestatus-Kanal-Nr.	Diagnostic state channel no.	N° de canal de diagnóstico	N° de canal d'état du diagnostic	N. canale stato diagnostico	诊断状态通道号
0034		DiagStateAddValue	Diagnosestatus-Zusatzwert	Diagnostic state additional value	Valor adicional del estado de diagnóstico	Valeur supplémentaire d'état du diagnostic	Valore aggiuntivo stato diagnostico	诊断状态附加值
0035		NoOfModules	Anzahl der Submodule	Number of modules	Cantidad de submódulos	Nombre de sous-modules	Numero di moduli subordinati	子模块数
0036		DeviceStructure	Gerätestruktur	Device structure	Estructura del dispositivo	Structure d'appareil	Struttura dispositivo	设备结构
0037		DeviceType	Gerätetyp	Device type	Tipo de dispositivo	Type d'appareil	Tipo di dispositivo	设备类型
0038		ObjDescrReq	Anfrage Objektbeschreibung	Object description request	Solicitud de descripción del objeto	Requête de description d'objet	Richiesta descrizione oggetto	对象描述询问
0039		ObjDescr	Objektbeschreibung	Object description	Descripción del objeto	Description d'objet	Descrizione oggetto	对象描述
003A		VersionCount	Versionszähler	Version counter	Contador de versiones	Compteur de version	Conteggio versione	版本计数器
	.1	ProfileVersion	Versionszähler - Profil-Version	Version counter - profile version	Contador de versiones - versión del perfil	Compteur de version - version du profil	Contatore versioni - versione profilo	版本计数器 - 概要版本

Index	Subindex	Object name (basic profile)	DE	EN-US	ES	FR	IT	ZH-CN
	.2	PCPVersion	Versionszähler - Parameterkanal-Version	Version counter - PCP version	Contador de versiones - versión del canal de parámetros	Compteur de version - version du canal de paramètres	Contatore versioni - versione canale parametri	版本计数器 - 参数通道版本
	.3	HardwareVersion	Versionszähler - Hardware-Version	Version counter - hardware version	Contador de versiones - versión del hardware	Compteur de version - version du matériel	Contatore versioni - versione hardware	版本计数器 - 硬件版本
	.4	FirmwareVersion	Versionszähler - Firmware-Version	Version counter - firmware version	Contador de versiones - versión del firmware	Compteur de version - version du microprogramme	Contatore versioni - versione firmware	版本计数器 - 固件版本
003B		PDIN_Descr	Prozesseingangsdatenbeschreibung	Process input data description	Descripción de datos de proceso de entrada	Description des données de process entrantes	Descrizione dati in ingresso processo	过程输入数据描述
003C		PDOOUT_Descr	Prozessausgangsdatenbeschreibung	Process output data description	Descripción de datos de proceso de salida	Description des données de process sortantes	Descrizione dati in uscita processo	过程输出数据描述
E800		DiagStateLong	Diagnosestatus (Langform)	Diagnostic state (long form)	Estado de diagnóstico (de forma detallada)	État du diagnostic (forme longue)	Stato diagnostico (forma estesa)	诊断状态(长式)
E801		DiagHistory	Diagnoseinformationen	Diagnostic history	Datos de diagnóstico	Informations de diagnostic	Informazioni diagnostiche	诊断信息
E802		DiagHistoryLong	Diagnoseinformationen (Langform)	Diagnostic history (long form)	Datos de diagnóstico (de forma detallada)	Informations de diagnostic (forme longue)	Informazioni diagnostiche (forma estesa)	诊断信息(长式)
E803		TraceBuffer	Puffer für Fehlerprotokollierung	Trace buffer	Búfer para registro de errores	Tampon pour le journal des erreurs	Buffer per protocollo errori	错误记录缓冲区
E804		LanguageAvailable	Landessprache	Language	Idioma nacional	Langue	Lingua nazionale	语言
E805		ObjDescrLong	Objektbeschreibung (Langform)	Object description (long form)	Descripción del objeto (de forma detallada)	Description d'objet (forme longue)	Descrizione oggetto (forma estesa)	对象描述(长式)

- 
- 
- 
- 

**- The End -**