

Application Note 134/10 v03

Title: Flexible E-Mode Channels

Status:

Draft Proposal

Date:

2013.08.02

Transitional period: Immediate effect after Final Voting.

Date: 2013.08.02

Subject: Resources, Procedures and Profiles for Flexible E-Mode Channels.

Documents

Modified

- [01] Chapter 3/5/1 "Resources"
- [02] Chapter 3/5/2 "Management Procedures"
- [03] Chapter 3/7/3 "Standard Identifier Tables"
- [04] Volume 6 "Profiles"

Referred

- [05] Chapter 3/3/4 "Transport Layer"
- [06] Chapter 3/3/7 "Application Layer"
- [07] Chapter 3/5/3 "Configuration Procedures"
- [08] Chapter 3/7/1 "Interworking Model"
- [09] Chapter 3/7/2 "Datapoint Types"
- [10] Supplement 12 "E-Mode Channel Codes, integrated in
 - Chapter 7/1/11 "Common E-Mode Channels"
 - Chapter 7/1/12 "Generic E-Mode Channels"
 - Chapter 7/20/11 "Lighting E-Mode Channels"
 - Chapter 7/50/11 "Shutters and Blinds E-Mode Channels"
- [11] Volume 7 "Application Descriptions"
- [12] Chapter 8/3/4 "Transport Layer Tests" v1.0 AS 2002.02.05.
- [13] Chapter 8/3/7 "Application (Interface) Layer Testing –
Network Management Server/Client Testing" v1.0 AS
2002.02.05
- [14] AN122 "Realisation of OpenTherm on KNX RF" v11 AS,
2009.06.24
- [15] AN124 "Interface Object Index Discovery" v02 DV
- [16] AN127 "Master Reset" v02 DV
- [17] AN128 "WGI accepted DPTs 09.01" v01 DP
- [18] AN132 "A_NetworkParameter_InfoReport" v01 DP
- [19] AN133 "A_DomainAddressSelective_Read" v01 DP
- [20] KSG436-04 "Configuration Signature.docx"
- [21] KSG502-09 "ETS RF system aspects"
- [22] KSG511-03 "Extended Interface Object Addressing"

Document updates

Version	Date	Description
KSG537-01	2013.05.06	<ul style="list-style-type: none"> Replacement of terminology and add new error code in PID_OBJECT_LINK function property. Localisation Mode may also be activated in a device via point-to-point connectionless communication mode. Use the discovery mechanism newly defined for ETS-RF. Other minor or editorial corrections
KSG537-02	2013.06.13	<ul style="list-style-type: none"> Update after first review in KSG (2013.06.12)
KSG537-03	2013.07.08	<ul style="list-style-type: none"> Update after KSG FEC dedicated meeting (2013.06.28)
KSG537-04	2013.07.16	<ul style="list-style-type: none"> Add the possibility to read the number of free associations (§2.3.4.13.2)
AN134 v03	2013.08.02	<ul style="list-style-type: none"> Preparation of the Draft Proposal.

This document is an update of the already approved Application Note 134 "Flexible E-Mode Channels". The scope of this voting is limited to the modifications brought in comparison to the approved version. These are marked in blue formatted font and highlighted.

Contents

1	Purpose, motivation and scope	5
1.1	Introduction and overview	5
1.2	Solutions	5
1.2.1	More flexible E-Mode Channels	5
1.2.2	More than four E-Mode Channel types per device	6
1.2.3	More than 32 E-Mode Channels per device	6
1.2.4	More than one Subnetwork per installation	6
1.2.5	Explicit description of Basic E-Mode Channels	6
1.2.6	Improved Localisation	7
1.2.7	Implementation independent Configuration Procedures	7
1.2.8	More Connection Codes	7
1.3	Results	8
1.3.1	New generation E-Mode device	8
1.3.2	What is FEC?	8
1.3.3	Single Profile E-Mode device - informative	9
2	Specification	12
2.1	Terms and definitions	12
2.2	Stack and communication	13
2.3	Resource definition or used Resources	14
2.3.1	Device Descriptor Type 2	14
2.3.2	Device Object (Object Type 0)	17
2.3.3	Application Program Object (Object Type 3)	19
2.3.4	E-Mode Channel Object (Object Type 14)	19
2.3.5	Adjusted E-Mode Channel Object (Object Type 15)	43
2.3.6	Text Catalogue Object (Object Type 16)	45
2.3.7	E-Mode Device Object (Object Type 18)	50
2.4	Management Procedures	55
2.4.1	Procedure: DMP_Connect_RCI	55
2.5	Configuration Procedures	57
2.5.1	Flexible E-Mode Channels	57
2.5.2	Ctrl-Mode for Flexible E-Mode Channels	59
2.5.3	PB-Mode for Flexible E-Mode Channels	68
2.5.4	ETS access to FEC devices	69
2.6	Interworking	70
2.6.1	Guidelines for the design of Adjustable E-Mode Channels	70
2.7	Property Datatypes	70
2.7.1	Overview	70
2.7.2	Detailed specification	70
2.8	Usage and context	72
2.9	Profile definition	72
2.9.1	Communication	73
2.9.2	Device Management	74
2.9.3	Device Identification	76
2.9.4	Device Individualisation	76

2.9.5	Programming Mode control	79
2.9.6	Device Linking	80
2.9.7	Application handling	81
2.10	Profile definition - Interface Objects and Properties.....	82
2.10.1	Interface Objects	82
2.10.2	Device Object	82
2.10.3	Applicationprogram Object	83
2.10.4	E-Mode Channel Object	83
2.10.5	Adjusted E-Mode Channel Object	85
2.10.6	Text Catalogue Object.....	85
2.10.7	E-Mode Device Object	86
2.10.8	Identifiers and discovery.....	86
3	Impact and dependencies	86
3.1	System specification ("Handbook") dependencies.....	86
3.2	Configuration interworking	86
3.3	Run-time Interworking	87
3.3.1	Concerning Basic E-Mode Channels.....	87
3.3.2	Concerning Extended E-Mode Channels	87
3.4	Integration and common tool impact.....	88
3.5	Risks and compatibility issues	88
3.5.1	Definition and use of two octet Connection Codes.....	88
Annex A	(informative) Parameter Types.....	89
A.1	Parameter Type coding	89

1 Purpose, motivation and scope

1.1 Introduction and overview

KNX E-Mode is divided into several submodes (see [07]).

- LTE-Mode
- Ctrl-Mode
- PB-Mode

This document affects only Ctrl-Mode and PB-Mode, which both base on the concept of E-Mode Channels.

The following requirements have been discovered:

- more flexible channels,
- more than 4 channel types per device,
- more than 32 channels per device,
- more than one Subnetwork per installation,
- explicit description of E-Mode Channels,
- improved Localisation
- implementation independent Configuration Procedures.
- more Connection Codes
- Master reset

The aim of this document is to introduce solutions for these requirements.

1.2 Solutions

 *This clause is only for introduction and does not contain normative requirements. This clause is not intended for integration in the KNX Specifications.*

1.2.1 More flexible E-Mode Channels

1.2.1.1 Current situation

The E-Mode Channel concept today is very inflexible. Every distributed *application* has to be mapped to the defined set of E-Mode Channel Codes. In many cases manufacturers are not satisfied with the available possibilities. E-Mode Channels do not cover all applications. Manufacturers miss the possibility to differentiate their products.

The existing E-Mode Channel definitions (see [10]) are simple and according an easy philosophy. They cover the application areas Lightening and Shutters and Blinds. The higher numbered E-Mode Channels are also for HVAC and some of them are quite complex. The reusability is very limited: the basic functionality is commonly wanted, but functionality other than the core functionality can rarely be reused. To define more and more E-Mode Channels will not be a pragmatic solution. Any time a new E-Mode Channel is added, the E-Mode Controller has to be updated.

1.2.1.2 Resolution

To solve this problem, in the future, Basic E-Mode Channels are introduced and specified.

The flexibility is introduced by defining Extended E-Mode Channels.

See 2.1 for the definition and 3.3 for the allowed use.

1.2.2 More than four E-Mode Channel types per device

1.2.2.1 Current situation

DD2 can report only 4 E-Mode Channel types. Each E-Mode Channel type can have up to 8 instances in a device. If localization E-Mode Channels are used, only 3 E-Mode Channel types are possible.

1.2.2.2 Resolution

If a device requires more than 4 E-Mode Channel types, the extended style introduced in this document (see below) shall be implemented.

1.2.3 More than 32 E-Mode Channels per device

This shall be solved as in 1.2.2, this is, can be solved by the introduction and use of one or more Extended E-Mode Channels.

1.2.4 More than one Subnetwork per installation

PB-Mode - and Ctrl-Mode installations with more than one Subnetwork are basically possible but not specified so far in the KNX Specifications. This document does not handle this topic but any extension shall not be limited to one Subnetwork.

1.2.5 Explicit description of Basic E-Mode Channels

1.2.5.1 Current situation

Existing Management Clients shall parse the DD2 to get the features of a device. The knowledge of any E-Mode Channel layout must be stored in the E-Mode Management Client. Besides the limitation in size of DD2, there is no possibility to report E-Mode Channel extensions. The introduction of any new E-Mode Channel type necessarily requires an update of the E-Mode Management Client.

1.2.5.2 Resolution

The mechanism of DD2 shall be kept for backward compatibility and for implementation of simple, low-cost devices. It is defined as the Basic style within the KNX Ctrl-Mode.

For devices using Extended E-Mode functionality a new style (Extended) will be introduced. These devices shall indicate in DD2 through the value 1000b for the field *Management Profile*, that the E-Mode description is accessible via Interface Objects.

The complete description of an Extended E-Mode Channel shall be available via Properties of Interface Objects. The Properties shall provide information about each E-Mode Channel and about all implemented Datapoints (Group Objects and Parameters). To give an E-Mode Management Client the possibility to learn unknown E-Mode Channels, the complete set of Datapoints shall be available; this is not only the implemented optional extensions but also the Datapoints of the Basic E-Mode Channel.

For each E-Mode Channel a separate Interface Object shall be implemented. Besides the device description each E-Mode Channel Object shall contain a Property for the parameter values. So in Extended Style the parameters are no longer located in the limited range of PID 101 to 132 in the Device Object.

The link procedure of the devices is changed. The services A_Link_Read and A_Link_Write, - using numbering space for the GOs common to the entire device – is replaced by a Function Property based linking with PID_OBJECTLINK in which the GOs are numbered separately for each E-Mode Channel.

For the technical specification of this resolution, and as an introduction to the below, please refer to 2.5 at first.

1.2.6 Improved Localisation

1.2.6.1 Current situation

Each current localisation method has one or more of the following drawbacks.

- It is limited in the possible number of E-Mode Channels in a device (8-, 16- or 24).
- An E-Mode Channel Type is occupied (“lost”) for the Localisation Channel.
- Group Addresses need to be assigned temporarily and group communication is misused for configuration purposes.

1.2.6.2 Resolution

The new solution proposed in this document bases on dedicated Properties in the newly defined E-Mode Device Object. This does not have the above drawbacks; the number of E-Mode Channels is only limited by the addressing possibilities of Interface Objects.

1.2.7 Implementation independent Configuration Procedures

1.2.7.1 Current situation

The current E-Mode implementations are all combinations of an E-Mode Profile together with an S-Mode Profile. This allows ETS to access these devices but it requires an ETS product database entry for the devices.

1.2.7.2 Resolution

The FEC extensions allow for a full, Property based and implementation independent discovery (identification) and reading out of the configuration (links, Parameters, etc.) of the E-Mode device without the combined presence of an S-Mode Profile in the device.

1.2.8 More Connection Codes

1.2.8.1 Current situation

The range of definable Connection Codes is limited to 255 entries of which today (Feb. 2010) 133 have been defined. The continued use of E-Mode in new, more specialised E-Mode Channels with often larger numbers of DPs occupies increasingly fast the remaining free Connection Codes.

1.2.8.2 Resolution

The Connection Codes will be encoded as two octet values; this allows for up to 65 535 Connection Codes.

1.3 Results

This Application Note introduces a KNX device model for E-Mode devices with the following major differences to existing KNX (E-Mode) devices.

1.3.1 New generation E-Mode device

The Flexible E-Mode device (FEC device) not only differs from existing device Profiles in the Resources and Configuration Procedures. It shall also implement the following new or recent KNX system features.

- Master Reset (see [16])
- Interface Object Index Discovery (see [15])
- PDT_REFERENCE (see 2.7.2 + see [22])
- Two octet sized Connection Codes (see 2.3.4.11.1)
- Configuration Signature (Download Counter) (see [20])
- Group Objects numbered per E-Mode Channel, not for the entire device.

There is thus good reason to talk about a new generation of E-Mode devices.

These are general extensions to the KNX system that will at first be used in FEC devices, but are as well usable to implementations of other KNX Profile as well.

1.3.2 What is FEC?

FEC denotes the possibility to specify and implement optional functionality (Group Objects, Parameters) in E-Mode Channels. It opposes in this to the existing (March 2010) E-Mode Profiles, that only allow for E-Mode Channels with fixed functionality. Next, also other functionality, as listed above, is added.

FEC has during his specification phase evolved from an optional and compatible extension to the existing PB-Mode and Ctrl-Mode towards an own standing Profile for the next generation E-Mode devices.

1. Ctrl FEC
2. PB FEC, which shall include Ctrl FEC

This Profile is not specified in full in this document. This will be discussed in a stand-alone KSG discussion topic in a separate paper. This paper only lists the first conclusions on the Profile PB FEC.

It is the intention that ETS can discover in the entire product description in a Property based way. This self description is typical for Ctrl-Mode devices and is now inherited in PB-Mode devices as well. Therefore, the Profile PB FEC will contain the Profile Ctrl FEC in full.

NOTE 1 Ctrl SEC denotes Controller Mode with Static E-Mode Channels. This is used to refer to both existing Profiles named "Ctrl-Mode Fixed DMA" and "Ctrl-Mode reloc. DMA" in [04].

PB SEC denotes Push Button Mode with Static E-Mode Channels. This is used to refer to the existing Profile named "PB-Mode" in [04].

1.3.3 Single Profile E-Mode device - informative

This clause is only informative. The terms used in the below are however to be standardised and normative. Therefore, please refer to 2.1 firstly for the new and changed definitions of terms.

1.3.3.1 Basic plan

Existing E-Mode Profiles and – Devices (state Aug 13) are all combinations of an E-Mode device Profile and an S-Mode device Profile. **The Profiles specified in this document – this is Ctrl FEC and PB FEC - are intended to be implemented as sole Profile in a device.**

This means that none of the Resources of any legacy Profiles are inherited and that also the ETS access to such Flexible E-Mode device is specified specifically for that Profile and are not inherited from any existing S-Mode device Profile.

During an intermediate transition period however, waiting for ETS to support FEC devices, it will be necessary that the FEC Profile be combined with the Profile and Resources Types of existing S-Mode Profiles.

Figure 1 sketches the basic time plan of the major Profiles for now and the future.

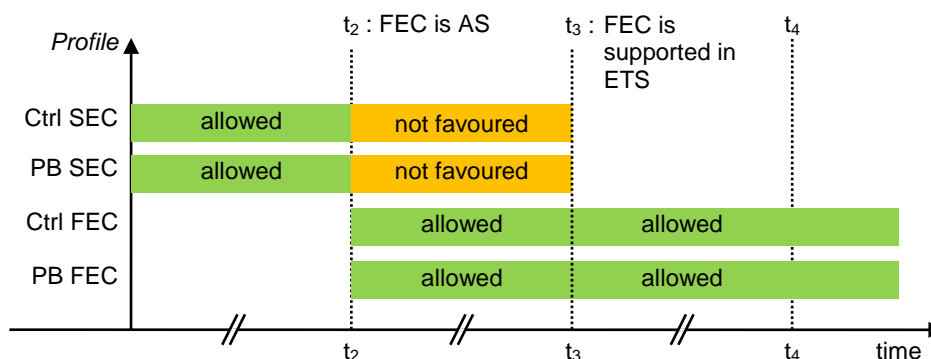


Figure 1 – Main Profiles transition

This time plan is not part of the normative contents of this paper. This clause is informative. The real definition of t₄ will be discussed in KNX KTB and the ETS management.

1.3.3.2 Full combinatorics and time plan

Whereas the above clause only gives the basic plan, other combinations can be considered that may or may not, immediately, or for an intermediate period, be allowed or not allowed. Table 1 lists these currently existing E-Mode Profiles, the envisaged next generation E-Mode Profiles and possible intermediate implementations.

The rows with **green background** indicate the intended next generation Profiles:

- Ctrl FEC
- PB FEC

Summary

The following is envisaged.

- Controller Mode devices and Push Button devices will have FEC functionality. These are further on noted as “Ctrl FEC” and “PB FEC”. Seeing 1.3.1, FEC means more than only using the flexibility of optional items. Thus, even if not using the flexibility, the device will be flexible.
- It is the intention that Push Button Mode devices will not be implemented with only the “Push Button Mode” style, but only in combination with the Controller Mode style. So, it shall be noted that PB FEC will always include Ctrl FEC.
- It is the intention that the next generation E-Mode devices will be Ctrl FEC **or** PB FEC and that there are new implementations of the classic Controller Mode (Ctrl SEC) and the classic Push Button Mode (PB SEC).
- During a transition period, Ctrl FEC and PB FEC can be combined with an S-Mode Profile, to allow access by ETS.

Table 1 – Possible and meaningful Profiles with FEC

Profile				Timeline			
Any S-Mode Profile	Ctrl SEC	PB SEC	+ FEC extension	t ₁ : until FEC is AS	t ₂ : FEC is AS	t ₃ : ETS 4 supports FEC	t ₄ : t ₃ + Δt
General footnotes:				none	g	none	none
NO	NO	NO	NO	n/a	n/a	n/a	n/a
NO	NO	NO	YES	n/a a	n/a a	n/a a	n/a a
NO	NO	YES	NO	allowed, no devices	allowed, not favoured	not allowed b	not allowed b
NO	NO	YES	YES	n/a	X c	X c	X c
NO	YES	NO	NO	allowed, no devices	allowed, not favoured	not allowed b	not allowed b
NO	YES	NO	YES	n/a	allowed unrealistic d	allowed	mandatory f
NO	YES	YES	NO	This combination of features, namely classic Ctrl SEC with Flexible E-Mode extension is the new Profile named "Ctrl FEC" .			
NO	YES	YES	YES	not foreseen no devices	not allowed	not allowed	not allowed
NO	YES	YES	YES	n/a	allowed unrealistic d	allowed	mandatory f
YES	NO	NO	NO	This combination of features, namely classic PB SEC with Flexible E-Mode extension including Ctrl FEC functionality is the new Profile named "PB FEC" .			
YES	NO	NO	YES	classic S-Mode device	classic S-Mode device	classic S-Mode device	classic S-Mode device
YES	NO	YES	NO	n/a a	n/a a	n/a a	n/a a
YES	NO	YES	YES	allowed, no devices	allowed, not favoured	not allowed b	not allowed b
YES	NO	YES	YES	n/a	not allowed	not allowed	not allowed
YES	YES	NO	NO	classic Ctrl-Mode device (= Ctrl SEC)	allowed not favoured	not allowed b	not allowed b
YES	YES	NO	YES	n/a	allowed, a necessity	allowed	allowed e
YES	YES	YES	NO	allowed (CIAT)	allowed, not favoured	not allowed b	not allowed b
YES	YES	YES	YES	n/a	allowed, a necessity	allowed	allowed e

n/a: not applicable

- a Please remind that FEC is an extension to Ctrl SEC and PB SEC; it is not an own standing Profile. A device with "only FEC" is not possible.


- b It is of course the intention that new stack implementations of the Profiles Ctrl SEC and PB SEC will no longer be allowed in the future, as the requirement that new E-Mode Channels may need to be possible with these stacks will cause again the trouble (inflexibility) for which FEC has been the solution. A good point to no longer allow old style E-Mode implementations may be this point t_3 when ETS 4 starts supporting FEC. Possibly, this should be earlier.
- c It is the intention that Push Button Mode devices should not simply be extended with FEC functionality.
PB FEC \neq PB SEC + FEC features
Instead, PB FEC devices should support the features of Ctrl FEC!
PB FEC = PB SEC + Ctrl FEC
This only draws the idea. The clear requirements for PB FEC are laid down in the Profiles part of this document.
- d It would of course be possible and allowed to implement a device with FEC as soon as the FEC specification is approved, without the combined implementation of an S-Mode Profile. However, as such device would not be configurable with ETS, it is assumed that this will be unrealistic.
- e It is of course the intention that later on, devices with FEC can live without the combination of an S-Mode Profile. The combination will no longer be a necessity, but an optional possibility.
- f "Mandatory" mean that from this point in time onwards, FEC is mandatory for Ctrl-Mode respectively PB-Mode devices.
- g The point t_2 denotes the point in time where FEC is fully specified and has become Approved Standard. This point t_2 will differ for PB FEC and Ctrl FEC.


2 Specification

2.1 Terms and definitions

Basic E-Mode Channel	Specification of the minimal, mandatory Functional Blocks and Datapoints thereof that compose an E-Mode Channel. NOTE 1 This complies 1-to-1 with the existing E-Mode Channel definition. NOTE 2 The Basic E-Mode Channel gives the Channel Code to the Extended E-Mode Channel.
Extended E-Mode Channel:	Specification of a Basic E-Mode Channel extended with possible (allowed) additional, optional extensions that are implemented. NOTE 3 An Extended E-Mode Channel does not have an own unique Channel Code. It has the Channel Code of its Basic E-Mode Channel. Two or more implementations of the same Extended E-Mode Channels, differing in their implemented set of optional features will still have the same E-Mode Channel Code.
Fixed E-Mode Channel	This opposed to an Adjusted E-Mode Channel. The definition of the available Group Objects and Parameters is fixed.
Adjusted E-Mode Channel	This opposes to a Fixed E-Mode Channel. The definition of the available Group Objects and Parameters depends on the value of the <i>Adjustable Parameter</i> .

NOTE 2 "Flexible E-Mode Channels" and "Adjusted E-Mode Channels" should not be confused. A Flexible E-Mode Channels adds the possibility to add optional but fixed defined Group Objects and Parameters. In an Adjusted E-Mode Channel for a given value of the Adjusted Parameter, the number and definition of Group Objects and Parameters is fixed.

Ctrl SEC	The name of the Profile for the Controller Mode with Static E-Mode Channels.  This term replaces the indication "Ctrl-Mode".
Ctrl FEC	The name of the Profile for the Controller Mode with Flexible E-Mode Channels.

PB SEC	The name of the Profile for the Push Button Mode with Static E-Mode Channels.  <i>This term replaces the indication PB-Mode.</i>
PB FEC	The name of the Profile for the Push Button Mode with Flexible E-Mode Channels.
<i>device localisation action</i>	This shall be an action performed by the device and that should be observable by a human and is further implementation dependent. It may be <ul style="list-style-type: none"> - a single short action, or EXAMPLE 1 A relay may close and open again. - a continuous state, or EXAMPLE 2 An LED may flash. - a repeated action, or EXAMPLE 3 A signal tone may be given periodically. - any implementation specific action.
<i>user localisation action:</i>	This shall be an action performed by a human on the device. the specific realisation is implementation dependent. EXAMPLE 4 a press on a rocker of a push button

Abbreviations

FEC	Flexible E-Mode Channel
SEC	Static E-Mode Channel

2.2 Stack and communication


 *This clause shall not be integrated in the KNX Specifications.*

This Application Note does not introduce neither modify any communication stack service or mechanism. However, the Flexible E-Mode Channel concept makes use of mechanisms that are specified in different papers that do modify the communication stack.

- “Master reset” as specified in [16] modifies the A_Restart-service.
- “Interface Object Index Discovery” as specified in [15] modifies the A_NetworkParameter_Read-service.
- In 2.5.2.7, the A_NetworkParameter_InfoReport-service is used. This service has been introduced in AN055 in the year 2003 – 2004, but has not been brought to an Approved Standard, with the outphasing of A-Mode. This service is used here and is re-introduced in the document [18].
- For the discovery of a free RF DoA, in the TF FEC meeting of 2010.03.03, it was concluded to revise the A_DomainAddressSelective_Read-service. In order not to overload this document, this topic is handled separately as KSG discussion topic.

2.3 Resource definition or used Resources

2.3.1 Device Descriptor Type 2

 This clause shall replace clause 4.1.3 in [01].
(Black text = inherited from existing version; green = new text; red = modifications.)

2.3.1.1 Device Descriptor Type 2 format

The Device Descriptor Type 2 shall have the format as specified in Figure 2.

Octet 0 (MSB)	Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7	Octet 8	Octet 9	Octet 10	Octet 11	Octet 12	Octet 13 (LSB)
Application Manufacturer		Application Identification		Appli- cation Version	Mgmt Prof. + reserved	Channel Info 1		Channel Info 2		Channel Info 3		Channel Info 4	

Figure 2 – Device Descriptor Type 2 format

All fields shall always be provided, even if unused.

- Application Manufacturer
 - position: octets 0 (MSB) and 1
 - description: This shall be the manufacturer code of the manufacturer of the E-Mode device.
 - format, encoding: U_{16} . The value shall be binary encoded.
- Application Identification ¹⁾
 - position: octets 2 and 3
 - description: This field shall encode the manufacturer specific Application Identification.
 - format, encoding: 16 bit value. The coding is manufacturer specific.
- Application Version
 - position: octet 4
 - description: This field shall contain the version of the manufacturer specific version of the application.
 - format, encoding: U_8 . The value shall be binary encoded.

¹⁾ This field may be named "Device Type" in other publications.

- Management Profile
 - position: octet 5 bits 7 (msb) to 4
 - description: This field shall contain the Management Profile of the E-Mode device.
 - format, encoding: N_4

Table 2 – Management Profile and reserved field - overview

Management Profile				reserved				Link services shall be supported	Flexible E-Mode Channels possible	Master Reset	Requirements
7	6	5	4	3	2	1	0				
0	0	0	0	0	0	0	0	No	No	No	These are the Profiles: <ul style="list-style-type: none"> – Easy Ctrl fixed DMA, and – Easy Ctrl reloc DMA as specified in [04]. This is not allowed for future use.
0	0	1	1	1	1	1	1	No	No	No	Reserved Management Profile.
0	1	0	0	0	0	0	0	Yes	No	No	These are the Profiles: <ul style="list-style-type: none"> – Easy Ctrl fixed DMA, and – Easy Ctrl reloc DMA as specified in [04] with the support of Link management Services This is not allowed for future use.
1	0	0	0	0	0	0	0	No	Yes	Yes	This denotes a Ctrl FEC or PB FEC device. For this Management Profile, the fields Channel Info 1 to Channel Info 4 shall be 0000h.

Other values of Management Profile are reserved and shall not be used by the E-Mode device.

The Management Client shall verify that the reserved bits are 0 or 1 as specified in Table 2. If these bits do not have the specified value then the Management Client should not change the device.

NOTE 3 The value 1111b for the reserved bits shall not be used for any future Management Profile.

 *This is the resolution to the comment INSTA 1 from RfV.*

NOTE 4 A FEC Profile implementation may for a transition period be combined with a S-Mode Profile to allow for ETS access to the device. This is described in 2.5.4. As it is not the intention that any such combination be a new Profile, there are no provisions for such combinations in the above encoding of the Management Profile.

- Channel Info 1

position: octet 6 and 7
description: Number and type of channels, first type
format, encoding: U_3U_{13}

bit nr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
name:	number			Channel Code												

- number

position: bit 15 to bit 13
description: The field *number* shall contain the number of available E-Mode Channels of the following E-Mode Channel Code. The possible number of E-Mode Channels for a given E-Mode Channel Code implemented in a device varies from 1 to 8. Value 0 shall mean 1 such E-Mode Channel.
format, encoding: U_3
The real available number of E-Mode Channels of this E-Mode Channel Code is number + 1.
If the field *Management Profile* denotes a FEC device (value 1000b) then this field *Number* shall be 0.

- Channel Code

position: bit 12 to bit 0
description: E-Mode Channel Code
format, encoding:

This field Channel Code shall identify the type of the E-Mode Channel. The E-Mode Channels and the corresponding E-Mode Channels are specified for every application in the appropriate Chapters in [11]. The overview of the E-Mode Channels Codes is located in [03].

The E-Mode Channels Codes shall be arranged as follows.

For Management Profiles 0000b and 0100b

If a channel does not exist the Channel Code shall be 0000h. For unknown channels the Channel Code shall be set to 0001h.

Some Channel Code values are reserved as specified in Table 3.

Table 3 – Reserved Channel Code values

Channel Code	Use
0000h	unused (then no more valid data afterwards)
0001h	descriptor not provided through this service This value is reserved by KNX Association for future extensions. It shall not be used.
1FF0h	Localisation Channel for 1 to 8 Channels
1FF1h	Localisation Channel for 1 to 16 Channels
1FF2h	Localisation Channel for 1 to 24 Channels
1FF3h	This Channel Code shall not be used.
1FF4h	This channel code shall be used by LTE-Mode devices
1FF5h to 1FFFh	system reserved

For Management Profile 1000b

If the field *Management Profile* denotes a FEC device (value 1000b) then this field *Channel Code* shall be 0.

- Channel Info 2
 - position: octet 8 and 9
 - description: Number and type of channels, second type. See Channel Info 1.
 - format, encoding: U₃U₁₃
- Channel Info 3
 - position: octet 10 and 11
 - description: Number and type of channels, third type. See Channel Info 1.
 - format, encoding: U₃U₁₃
- Channel Info 4
 - position: octet 12 and 13 (LSB)
 - description: Number and type of channels, fourth type. See Channel Info 1.
 - format, encoding: U₃U₁₃

2.3.2 Device Object (Object Type 0)

2.3.2.1 Device Object – PID_DESCRIPTION (PID = 21)

2.3.2.1.1 Interface Object Type independent specification

 *This clause shall replace clause 4.2.21 in [01].*

- Property name: Description
- Property Datatype:
 - PDT_UNSIGNED_CHAR[]
 - PDT_REFERENCE
- Datapoint Type: Every character: DPT_UTF_8 (DPT_ID: 28.001)
- Access: Data Property: 3/1

This Property shall include additional descriptive information to the functionality represented by the Interface Object in which this Property is located.

EXAMPLE 5 In the Device Object, this Property shall be interpreted as additional descriptive information to the device.

The descriptive information shall be encoded in ASCII / ISO 8859-1 as an array of characters (unsigned char[]; the maximum number of elements shall be fixed in the device). The access rights of this Property depend on the implementation.

2.3.2.1.2 Resource definition

For the Interface Object Type independent specification of PID_DESCRIPTION, see above.

In a FEC device, this Property

- is mandatory in the Device Object, and
- shall have a minimal size of 10 characters; no maximal size is specified.

2.3.2.1.3 Usage by the Management Client

The Management Client may use PID_DESCRIPTION to store human readable textual information, entered by the installer, to help localising, identifying and describing the device.

This shall help for reverse engineering E-Mode installations.

2.3.2.2 Device Object – PID_IO_LIST (Property Identifier 71)

PID_IO_LIST is specified here, as it has been the original intention to use this Property for discovery of the E-Mode Channel Object and the Adjusted E-Mode Channel Object. This approach has been discontinued in favour of a solution with NM_InterfaceObject_Index_Read ([15]). However, as the below is an improved specification of PID_IO_LIST than what is available in [01], this clause 2.3.2.2 is kept.

- Property name: Interface Object List
- Property Datatype: PDT_UNSIGNED_INT[]
- Datapoint Type: DPT_Value_2_Ucount (7.001)

2.3.2.2.1 Resource definition

PID_IO_LIST shall be a read-only, array structured Property with as much elements as Interface objects are located (or “active”) in the device. The array element with index 0 shall contain the number of array elements (this is, Interface Objects) implemented in the device. The Property shall contain the Interface Object Types in ascending order of the object indexes, starting with the array field with index 1. For Interface Objects with more than 1 instance, the 1st instance shall be at the position with the lower/lowest index, the Interface Object Type of the 2nd instance shall be placed at the next index and so on.

2.3.2.2.2 Usage by the Management Client

2.3.2.2.2.1 Remote access

The Management Client shall access this Property through

- NM_InterfaceObject_Index_Read, as specified in [15].
- DMP_InterfaceObject_Read_RCI, or
- DMP_InterfaceObject_Read_RCo if the Management Server supports connection-oriented communication mode and a Transport Layer connection to the Management Server exists.

The below procedure reads PID_IO_LIST from the Device Object. Please note that the Management Procedure DMP_InterfaceObject_Read_R foresees multiple calls of A_PropertyValue_Read in case the Property Value cannot be transmitted in a single frame.



```
/* Read PID_IO_LIST of the Device Object */  
DMP_InterfaceObject_Read_RCI( /* [IN] */ object_index = 0; /* [IN] */ PID = PID_IO_LIST,  
/* [OUT] */ PID_IO_LIST = data)
```

2.3.2.2.2.2 Local access

AN033 “cEMI” specifies PID_IO_LIST but does not specify any requirements towards a Management Client on how to access it (even if this is trivial): there exist no Management Procedures for the cEMI Management Service (M_PropRead, M_PropWrite, etc.). It is outside the scope of this paper to define these (as the local access is not used here).

2.3.3 Application Program Object (Object Type 3)

2.3.3.1 Application Program Object – PID_RUN_STATE_CONTROL (PID = 6)

-  *This clause shall not be integrated in the KNX Specification.
This clause 2.3 is named "Resource definition or used Resources" and as the already defined PID_RUN_STATE_CONTROL is used, it is referred here.*
-  *PID_RUN_STATE_CONTROL will in Ctrl FEC be used by the Controller to hold the Application Program prior to changing the device configuration.*

For the specification of PID_RUN_STATE_CONTROL please refer to [01].

2.3.4 E-Mode Channel Object (Object Type 14)

2.3.4.1 General

2.3.4.1.1 Use and general requirements of the E-Mode Channel Object

This Interface Object shall hold information about the E-Mode Channel itself, about all in the E-Mode channel implemented Group Objects and Parameters.

Information in the various Properties on the same Property element index shall relate to the same Group Object or Parameter.

Please refer to 2.10 for the mandatory and optional Properties in the E-Mode Channel Object; Table 4 only gives the overview and is informative.

One instance of this Interface Object shall be implemented for each implemented E-Mode Channel.

There are no requirements towards the Object Indexes of the E-Mode Channel Objects: they do not need to be in consecutive order or be sorted in any way. There is no mandatory relation between the Object Indexes and the E-Mode Channel Number; this shall only be given through the Property *Channel Number*.

2.3.4.1.2 Group Objects

The GOs in an E-Mode Channel Object instance shall be numbered independently of the GOs of other E-Mode Channel Objects. The GOs of each separate E-Mode Channel shall be numbered starting from 1.

The GOs in the E-Mode Channel are numbered consecutively starting from 1 and have consecutive values without gaps. In the Properties that describe the GOs of one E-Mode Channel, data on the same Property Index shall refer to the same GO.

2.3.4.1.3 Property overview

Table 4 – E-Mode Channel Object (informative)

PID Value	PID Name	Property Name	PDT/DPT	Access
1	PID_OBJECT_TYPE	Interface Object Type	DPT_PropDataType (DPT_ID = 7.010)	3/1
2	PID_OBJECT_NAME	Interface Object Name	PDT_UNSIGNED_CHAR[] (DPT_UTF_8 (DPT_ID: 28.001)) - PDT_REFERENCE	3/1
21	PID_DESCRIPTION	Description	PDT_UNSIGNED_CHAR[]	3/3

PID Value	PID Name	Property Name	PDT/DPT	Access
25	PID_VERSION	Version	PDT_VERSION (DPT_Version, DPT_ID = 217.001)	3/1
51	PID_CHAN_NUMBER	Channel Number	PDT_UNSIGNED_INT	3/1
52	PID_CHAN_CODE	Channel Code	PDT_UNSIGNED_INT	3/1
53	PID_CHAN_FLAGS	Channel Flags	PDT_GENERIC_02	3/1
54	PID_CHAN_FB_LIST	Functional Block List	PDT_UNSIGNED_INT[] (DPT_Value_2_Ucount, DPT_ID = 7.001)	3/1
55	PID_CHAN_ADJ_LISTS	Adjustable Channel Object List	PDT_UNSIGNED_CHAR[]	3/1
61	PID_GO_CCOCES_LIST	Connection Codes	PDT_GENERIC_08[]	3/1
62	PID_GO_CFLAGS_LIST	Connection Flags	PDT_BITSET16[]	3/1
63	PID_OBJECTLINK	Object Link	PDT_FUNCTION	3/3
64	PID_GO_SUBUNIT	Subunit Number	PDT_UNSIGNED_CHAR[]	3/1
65	PID_GO_NAME_LIST	Group Object Names	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
70	PID_PARAM_TYPES	Parameter Types	PDT_GENERIC_10[]	3/1
71	PID_PARAM_FLAGS	Parameter Flags	PDT_BITSET16[]	3/1
72	PID_PARAM_NAMES	Parameter Names	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
73	PID_PARAM_UNITS	Parameter Units	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
79	PID_PARAM_VALUES	Parameter Values	- PDT_GENERIC_01[], or - PDT_GENERIC_02[], or - PDT_GENERIC_04[], or - PDT_GENERIC_10[]	3/3

2.3.4.2 E-Mode Channel Object – PID_OBJECT_TYPE (PID = 1)

Please refer to [01] for the common specification of PID_OBJECT_TYPE.

The Object Type for the E-Mode Channel Object shall have the value 14.

2.3.4.3 E-Mode Channel Object – PID_OBJECT_NAME (PID = 2)

■ Chapter 3/5/1 “Resources” ([01]) so far does not specify the Datapoint Type of this Property. Therefore, this definition is taken over here and extended with that information. This information shall complete the specification in [01].

■ The alternative PDT for PID_OBJECT_NAME has to be introduced in [03] as well.

- Property name: Interface Object Name
- Property Datatype:
 - PDT_UNSIGNED_CHAR[]
 - PDT_REFERENCE
- Datapoint Type: DPT_UTF_8 (DPT_ID: 28.001)
- Access: Data Property: 3/1

This Property shall contain the name of the E-Mode Channel. In this case, the PDT shall be PDT_UNSIGNED_CHAR[]. Alternatively, this Property may also only be a reference to the name of the E-Mode Channel. In that case the PDT shall be PDT_REFERENCE and the Property Value shall be the reference to the Property Name.

EXAMPLE 6 “Blinds Ch. 1”

The name shall in both cases be encoded as a string of variable length. Every textual character shall be encoded as DPT_UTF_8 (DPT_ID: 28.001).

The value of this Property is given by the product manufacturer and can typically not be changed by the planer or installer. If it is wanted that the product installer or – planner assigns an own textual description to this E-Mode Channel, then this can be done and stored in the E-Mode Management Client instead, and not in the E-Mode device or the below Property PID_DESCRIPTION can be used.

2.3.4.4 E-Mode Channel Object – PID_DESCRIPTION (PID = 21)

2.3.4.4.1 Resource definition - FEC

For the Interface Object Type independent specification of PID_DESCRIPTION, please refer to [01].

In a FEC device, this Property

- is mandatory in the E-Mode Channel Object, and
- shall have a minimal size of 10 characters; no maximal size is specified.

2.3.4.4.2 Usage by the Management Client

The Management Client may use PID_DESCRIPTION to store human readable textual information, entered by the installer, to help localising, identifying and describing the E-Mode Channel.

This shall help for reverse engineering E-Mode installations.

EXAMPLE 7 “Light Room 1”

2.3.4.5 E-Mode Channel Object – PID_VERSION (PID = 25)

- Property name: Channel Version
- Property Datatype: PDT_Version
- Datapoint Type: DPT_Version (DPT_ID = 217.001)
- Access: Data Property: 3/1

The Property Version (PID_VERSION) shall contain the manufacturer specific version of the E-Mode Channel.

2.3.4.5.1 Usage by the Management Server (device)

The manufacturer of the E-Mode Channel shall take care that the version indication in this Property is properly handled according the definition of DPT_Version in [09].

2.3.4.5.2 Usage by the Management Client

The Management Client *may* interpret this Property Version (PID_VERSION) in combination with the Properties Channel Code (PID_CHAN_CODE) and the manufacturer code of the device as retrieved through DD2. The E-Mode Management Client may store the manufacturer code, channel version and channel description of each E-Mode Channel of which it reads out the description. If the same values are returned later by another E-Mode Channel, the E-Mode Management Client may assume that these cached values are correct and up-to-date and may skip the further discovery of this E-Mode Channel. This speeds up the discovery procedure. This behaviour by the E-Mode Management Client is optional.

2.3.4.6 E-Mode Channel Object – PID_CHAN_NUMBER (PID = 51)

- Property name: Channel Number
- Property Datatype: PDT_UNSIGNED_INT
- Datapoint Type: DPT_Value_2_Ucount
- Access: Data Property: 3/1

2.3.4.6.1 Usage by the Management Server (device)

This Property shall contain the Channel Number of the E-Mode Channel.

Any Channel Number shall be within the range 1 to 255. The value 0 is reserved and shall not be used.

The Channel Number shall be a unique identifier for the E-Mode Channel Object within the FEC device. The implementer shall take care that no two E-Mode Channel Objects have the same value for the Property *Channel Number*.

This value shall amongst other be used in the Localisation Procedures with PID_LOCALISATION_REPORT (see 2.3.7.5) and PID_LOCALISATION_COMMAND (see 2.3.7.6).

2.3.4.6.2 Usage by the Management Client

The Management Client shall read out this Property during the discovery of the E-Mode Channel to link E-Mode Channel number with the Object Indexes of the E-Mode Channel Objects.

2.3.4.7 E-Mode Channel Object – PID_CHAN_CODE (PID = 52)

- Property name: Channel Code
- Property Datatype: PDT_UNSIGNED_INT
- Datapoint Type: None.

This Property shall contain the Channel Code of the Basic E-Mode Channel that is described by this *E-Mode Channel* Object. If this E-Mode Channel is not an Adjustable E-Mode Channel, then this is the only Channel Code indication; otherwise, alternative E-Mode Channel Codes will be present in one or more *Channel Code* Properties in related instances of the *Adjustable E-Mode Channel* Object.

The E-Mode Channel Code shall be a 13 bit identifier. It shall be positioned on the least significant positions of the PID_CHAN_CODE-value; the three most significant bits of PID_CHAN_CODE shall be 0.

2.3.4.8 E-Mode Channel Object – PID_CHAN_FLAGS (PID = 53)

- Property name: Channel Flags
- Property Datatype: PDT_GENERIC_02
- Datapoint Type: None.
- Access: Data Property: 3/1

Table 5 – PID_CHAN_FLAGS

Bit Nr.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
field symbol	r	r	r	r	r	r	r	r	r	r	LR	LC	r	O	I	A
	0	0	0	0	0	0	0	0	0	0						

- bits 0 (lsb):
 - name (abbreviation): Adjustable (A)
 - description: This specifies whether the E-Mode Channel is fully specified through this instance of the E-Mode Channel Object, or whether additional GO and Parameter information is available via one or more *Adjustable E-Mode Channel* Interface Objects.
 - encoding:
 - 0b: This E-Mode Channel is not an adjustable E-Mode Channel.
The GOs and Parameters of this Flexible E-Mode Channel are fully specified in this instance of the *E-Mode Channel* Object; there are no instances of the *Adjusted E-Mode Channel Object* Type for this Flexible E-Mode Channel.
 - 1b: This E-Mode Channel is an adjustable E-Mode Channel.
The GOs and Parameters of this Flexible E-Mode Channel may vary in number and function and are described in one or more instances of the *Adjusted E-Mode Channel Object* Type.
If this bit is set, the Property PID_CHAN_ADJ_LISTS shall be available and shall contain at least one object index referring to an Object of type Adjusted E-Mode Channel.

- bits 1:

name (abbreviation):	Input (I)
description:	This specifies whether the E-Mode Channel is an input E-Mode Channel or not.
	NOTE 5 This information does not affect the Configuration of this E-Mode Channel. It is merely considered to allow the E-Mode Controller to position the E-Mode Channel in a graphical representation of the link with another E-Mode Channel either on the left (input) or on the right (output).
encoding:	0b: This E-Mode Channel is not an Input E-Mode Channel. 1b: This E-Mode Channel is an Input E-Mode Channel.
 - bit 2:

name (abbreviation):	Output (O)
description:	This specifies whether the E-Mode Channel is an output E-Mode Channel or not..
encoding:	0b: This E-Mode Channel is not an Output E-Mode Channel. 1b: This E-Mode Channel is an Output E-Mode Channel.
- An E-Mode Channel that is neither an Input Channel nor an output Channel (I = 0 and O = 0) shall be regarded as a Functional Module.
- It is possible that an E-Mode Channel has both an Input Channel functionality (I = 1) as well as output Channel functionality (O = 1).
- bit 3:

name (abbreviation):	reserved (r)
description:	This bit is reserved.
encoding:	This bit shall always be 0.
 - bit 4

name (abbreviation):	Localisation Command (LC)
description:	This field shall signal the support of a Localisation Command by this E-Mode Channel.
encoding:	0b: This E-Mode Channel <u>does not support</u> Localisation Commands. If a Localisation action is requested for this E-Mode Channel (see 2.3.7.6.3), this device shall not react. 1b: This E-Mode Channel supports Localisation Commands.
 - bit 5

name (abbreviation):	Localisation Report (LR)
description:	This field shall signal the support of Localisation Reports by this E-Mode Channel.
encoding:	0b: This E-Mode Channel <u>does not support</u> Localisation Reports. While Localisation Mode is active in the E-Mode device, local operations on this E-Mode Channel's user interface will not cause the device to report such localisation action. 1b: This E-Mode Channel supports Localisation Reports.

The fields Localisation Command (LC) and Localisation Report (LR) are independent of each other. The field Localisation Action is typically set for an actuator E-Mode Channel but can also be set for a sensor E-Mode Channel and vice versa.

EXAMPLE 8 A sensor E-Mode Channel may have a small LED (e.g. a status LED) that can be used for localisation.

EXAMPLE 9 A binary output E-Mode Channel may have a hardware switch for local operation of the relay that can also be used for localisation.

Please refer to 2.10.4.1 for the requirements concerning the support of these fields.

- bits 15 to 6:

These bits are reserved and shall be cleared in the device.

The E-Mode Management Client shall ignore the value of these bits.

2.3.4.9 E-Mode Channel Object – PID_CHAN_FB_LIST (PID = 54)

- Property name: Functional Block List
- Property Datatype: PDT_UNSIGNED_INT[]
- Datapoint Type: DPT_Value_2_Ucount
- Access: Data Property: 3/1

The Property Functional Block List shall be an array containing **all** Interface Object Types defined for the Functional Blocks realised by this E-Mode Channel.

If a FB is used more than once in the E-Mode Channel, then it shall be reported as many times in this list.

This list may but does not have to be sorted.

2.3.4.10 E-Mode Channel Object – PID_CHAN_ADJ_LISTS (PID = 55)

- Property name: Adjustable Channel Object List
- Property Datatype: PDT_UNSIGNED_CHAR[]
- Datapoint Type: None.
- Access: Data Property: 3/1

The Property *Adjustable Channels Object List* shall be an array of the indexes of the *Adjusted E-Mode Channel Object* instances for this E-Mode Channel, if there are any.

The Property *Adjustable Channels Object List* shall have as many array elements as possible values that it supports for the parameter AJS. Every array element n ($n \neq 0$) shall specify the Object Index within the device that specifies the Adjustable Channel (Group Objects and Parameters) for the value n of AJS.

Adjustable Parameter Value	Property Value Array Index	Property Value
-	0	(current nr. of array elements)
1	1	Object Index of the Adjusted E-Mode Channel Object nr. 1
2	2	Object Index of the Adjusted E-Mode Channel Object nr. 2
3	3	Object Index of the Adjusted E-Mode Channel Object nr. 3
...

Table 6 – PID_CHAN_ADJ_LISTS format

If this Property is not implemented, then the bit *Adjustable* in PID_CHAN_FLAGS of the same E-Mode Channel Interface Object shall be cleared. If this Property is implemented, it shall contain at least one object index and the bit *Adjustable* in PID_CHAN_FLAGS shall be set.

2.3.4.11 E-Mode Channel Object – PID_GO_CCODES_LIST (PID = 61)

- Property name: Connection Codes
- Property Datatype: PDT_GENERIC_08[]
- Datapoint Type: None.
- Access: Data Property: 3/1

This Property shall contain the Main Connection Codes and all Additional Connection Codes for all Group Objects of this E-Mode Channel.

2.3.4.11.1 Format

This Property Value shall have an array structure. The Property Value array index used in this Property shall refer to the same Group Object as referred to in the array element with the same index in the Properties PID_GO_CFLAGS_LIST, PID_GO_SUBUNIT and PID_GO_NAME_LIST.

EXAMPLE 10 The GO is listed on array index 2 in PID_GO_LIST shall be the same GO as the GO listed on array index 2 in this Property PID_GO_CCODES_LIST.

The array elements of the Property Value shall be values formatted according PDT_GENERIC_08. The most significant double octet shall be the Main Connection Code. The following double octets may be used for subsequent Additional Connection Codes, filling the Property Value from the left (most significant positions) to the right (least significant positions). Positions that are not used by Additional Connection Codes shall have the value 0000h.

Property Index	Property Value			
0	current_nr_of_elem = 3			
1	Main CC for GO 1	0000h	0000h	0000h
2	Main CC for GO 2	1 st Add CC for GO 2	2 nd Add CC for GO 2	0000h
3	Main CC for GO 3	1 st Add CC for GO 3	0000h	0000h

Figure 3 - PID_GO_CCODES_LIST (example)

2.3.4.11.2 Usage by the Management Client and – Server

The Connection Codes shall be used by the Management Client and the Management Server for Ctrl FEC and PB FEC as laid down in [07]. If this does not allow for the transmission of two octets Connection Codes AND if the higher octet of the Connection Code equals 00h then these Configuration Modes shall use single octet Connection Codes with only the least significant octet.

See also 3.5.1.

2.3.4.12 E-Mode Channel Object – PID_GO_CFLAGS_LIST (PID = 62)

- Property name: Connection Flags
- Property Datatype: PDT_BITSET16[]
- Datapoint Type: None.
- Access: Data Property: 3/1

2.3.4.12.1 Use

This Property shall contain all Connection Flags for all Group Objects of this E-Mode Channel.

2.3.4.12.2 Format

This Property shall have an array structure. The Property Value array index used in this Property shall refer to the same Group Object as referred to in the Property Value array element with the same index in the Properties PID_GO_CCODES_LIST, PID_GO_SUBUNIT and PID_GO_NAME_LIST.

EXAMPLE 11 If GO 8 is listed on array index 2 in PID_GO_LIST, then array index 2 in this Property PID_GO_CFLAGS_LIST shall be interpreted as the Connection Flags for GO 8.

The Property array elements shall be values formatted according the format as specified in Table 7.

Bit Nr.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
field symbol	r	r	r	r	I	I	I	I	r	r	r	r	r	I	O	X
	0	0	0	0	0	0	0	0	0	0	0	0	0			

Table 7 – PID_GO_CFLAGS_LIST (one Property Value array element)

For the definition and use of Visualisation DPs, please refer to [08] clause “Describe the Channel”.

- bit 0:
 - name (abbreviation): X (X)
 - description: The X-field shall indicate whether this GO may be connected more than once or not. This flag is only meaningful for an input DP.
 - EXAMPLE 12 In the definition of the channel CH_logic_AndOr (id 0200h), each of the four input Datapoints shall be connected only once, in order to guarantee the coherency of the logical result.
 - encoding:
 - 0b: This GO may be connected more than once.
 - 1b: This GO shall not be connected more than once.
- bit 1:
 - name (abbreviation): O (O)
 - description: The O-field shall indicate whether this GO is an Output DP or not.
 - encoding:
 - 0b: This GO is not an Output DP.
 - 1b: This GO is an Output DP.
- bit 2:
 - name (abbreviation): I (I)
 - description: The I-field shall indicate whether this GO is an Input DP or not.
 - encoding:
 - 0b: This GO is not an Input DP.
 - 1b: This GO is an Input DP.

• bits 3 to 15

These bits are reserved and shall be cleared in the device.

The E-Mode Management Client shall ignore the value of these bits.

2.3.4.13 E-Mode Channel Object – PID_OBJECTLINK (PID = 63)

- Property name: Object Link
- Property Datatype: PDT_FUNCTION
- Datapoint Type: None.

This clause shall replace the clause 4.3.14 in [01], because the current specification there is much targeted on the use in RF circumstances. This new definition should be more generic.

Used by: Ctrl FEC
PB FEC

2.3.4.13.1 Function *Write Object Link*

2.3.4.13.1.1 Format

octet 10	octet 11	octet 12	octet 13	octet 14	octet 15	octet 16	octet 17	octet 18	octet 19	octet 20	octet 21
Flags	00h	DoA (high)	DoA	DoA	DoA	DoA	DoA (low)	GA (high)	GA (low)	group object index (hi)	group object index (lo)
		SN (high)	SN	SN	SN	SN	SN (low)				
		Extended Group Address									

**Figure 4 – Function *Write Object Link*
A_FunctionPropertyCommand-PDU (example)**

2.3.4.13.1.2 Usage by the Management Client

The Management Client shall use the Function *Write Object Link* to establish and to break a single link between a Group Object and a Group Address or an Extended Group Address.

1. Flags

Description: This field shall indicate whether a link to the referred Group Object shall be established or shall be broken. In case the link shall be added, it shall also indicate whether the link shall be a sending Group Address or not.

Encoding:

bit nr	7	6	5	4	3	2	1	0
name:	r	r	r	r	r	act	d	s

- **Sending (s)**

position:

description:

bit 0

In case the contained (Extended) GA is added, then this field *Sending (s)* shall indicate whether it shall be the sending GA for the referred GO or not.

The flag s shall only be interpreted in case the flag d equals 0; in case the flag d equals 1, the value of the flag s shall be don't care.

format, encoding:

0: not sending: The contained GA shall not be the sending GA for the referred GO.

1: sending: The contained GA shall be the sending GA for the referred GO.

Error and exception handling

- If a link is Added using PID_OBJECTLINK to a GO to which the contained GA is already linked, but with a different value of the field S, then this new value of the flag s shall be used.
- A GA set as "sending" shall take precedence on other previous sending GA for that GO.

- **Add/Delete (d)**
 - position: bit 1
 - description: The field *Add/Delete* (d) shall indicate whether the contained (Extended) GA shall be added to or removed from the list of GA assigned to the referred GO.
 - format, encoding:

0: add:	Add the contained GA to the list of GAs assigned to the referred GO.
1: delete:	Remove the contained GA from the list of GAs assigned to the referred GO.
- **Address extension type (aet)**
 - position: bit 2
 - description: This field (aet) shall indicate whether the DoA or the KNX SN is used in the link.
Only value 1 shall be supported on RF media.
 - format, encoding:

0: SN
1: DoA
- **reserved (r)**
 - position: bits 7 to 2
 - description: These bits are reserved.
 - format, encoding: The E-Mode Management Client shall clear (0) these bits. The E-Mode Management Server (device) shall ignore the value of these bits.

2. KNX Serial Number (SN) or RF Domain Address (DoA)

Description: If the field AET equals 0, then this field shall contain the KNX Serial Number of the extended GA. If the field AET equals 1, then this field shall be 000000000000.
If this Function request does not contain an Extended GA but only a basic GA, then this field shall be 000000000000h.

Error and exception handling

- If a link is Added using PID_OBJECTLINK with the field AET = 1 and the field DoA different from 0, then the links shall be refused with an explicit error code.

Encoding: DPT_SerNum (DPT_ID = 221.001) or 000000000000

3. Group Address (GA)

Description: This field shall be the basic GA or the GA part of the Extended GA, to which the Link shall be established or from which the Link shall be broken.

Encoding: U₁₆ value

4. Group Object Index

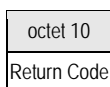
Description: This shall be the Group Object number. This shall refer to the same value as used to refer Group Object in the Property Value arrays PID_GO_CCODES_LIST, PID_GO_SUBUNIT and PID_GO_NAME_LIST.

This shall be the GO number of the GO within the E-Mode Channel. This shall start with 1.

Encoding: U₁₆ value

2.3.4.13.1.3 Usage by the Management Server (device)

The Management Server shall execute the Function *Write Object Link*. It shall respond to the Management Client with the response as given in Figure 5 that shall contain the *Return Code* of the Function.



**Figure 5 – Response to the Function *Write Object Link*
A_FunctionPropertyState_Response-PDU (example)**

1. Return Code

Description: The Return Code shall give information on the success or possible errors that occur in the execution of the Function *Write Object Link*.

Encoding: 00h: SUCCESS: Link established or broken or already established or broken before.
FFh: ERR_TABLEFULL: The device does not have memory space anymore for establishing the requested Link.
FEh: ERR_GRP OBJECT: Group Object does not exist or is out of range (0 for example).
FDh: ERR_DOA: The value of field DoA (different from 0) is not correct with the field AET (equal to 1).
FCh: ERR_GENERIC: Any other error than previously defined.

Error and exception handling

- If the Extended GA contains the own KNX Serial Number of the Management Server (device), then this Function shall be interpreted as the request to establish or break an internal Link.

2.3.4.13.2 Function *Read Object Link*

2.3.4.13.2.1 Format

octet 10	octet 11	octet 12	octet 13
Code	Group object index (hi)	Group object index (lo)	Iterator

Figure 6 – Function *Read Object Link*
A_FunctionPropertyState_Read-PDU (example)

Octet 10: Code

The octet 10 shall indicate whether the read command is used to go through the list of links or to know the number of free associations.

Code: 00h: Read the links through an iterator and a group object index.
01h: Read free number of associations: in this case, the response contains a return code together with two bytes for the number of free associations to existing GAs and two bytes for the number of associations with new GAs. The octets 11 to 13 are not used in this case and shall be set to 0.

2.3.4.13.2.2 Read through iterator (Code = 00h)

2.3.4.13.2.2.1 Usage by the Management Client

The Management Client shall use the Function *Read Object Link* to read a single Group Addresses or Extended Group Address linked to a single Group Object in a Management Server (device). Through subsequent calls of the Function *Read Object Link* with values of the *Iterator* from 0 incremented each call by 1, it can read through all links of the Group Object in the Management Server.

1. Group object index

Description: This shall be the Group Object number. This shall refer to the same value as used to refer Group Object in the Property Value arrays PID_GO_CCODES_LIST, PID_GO_SUBUNIT and PID_GO_NAME_LIST.

This shall be the GO number of the GO within the E-Mode Channel. This shall start with 1.

Encoding: U₁₆ value

2. Iterator

Description: This Parameter shall serve as an iterator: it shall be an index in the list with the Group Object links to the Group Object identified by the **Group Object Index** in the Management Server. (This list is an implementation specific storage that is not further standardised.) The Management Client can read the linked Group Addresses or Extended Group Addresses one by one, starting with Iterator value 0 and incremented by one for each next link to be read, until the Management Server responds that there are no more links for this Group Object for the requested Iterator value and above.

Encoding: U₈ value

2.3.4.13.2.2 Usage by the Management Server (device)

The Management Server shall execute the Function *Read Object Link*. It shall respond to the Management Client with the response as given in Figure 7.

octet 10	octet 11	octet 12	octet 13	octet 14	octet 15	octet 16	octet 17	octet 18	octet 19	octet 20	octet 21
Return Code	Flags	DoA (high)	DoA	DoA	DoA	DoA	DoA (low)	GA (high)	GA (low)	GO index (hi)	GO index (lo)
		SN (high)	SN	SN	SN	SN	SN (low)				

Figure 7 – Response to the Function *Read Object Link* (case Code = 00h)
A_FunctionPropertyState_Response-PDU (example)

NOTE 6 The Iterator from the request in the A_FunctionPropertyState_Read-PDU is not repeated in this response in the A_FunctionPropertyState_Response-PDU. The Management Client can only link this response to the iterator used in the preceding request. Therefore, the Management Client should not issue a new request before the response to a previous request has been received.

1. Return Code

Description: The Return Code shall give information on the success or possible errors that occur in the execution of the Function *Read Object Link* (code = 00h).

Encoding: 00h: This shall signal that for the requested value of the Iterator, the contained link information is valid

FFh: This shall signal that for the requested value of the Iterator the Function Read Object Link does not find a link.
In this case, the A_FunctionPropertyState_Response-PDU shall only contain the Return Code and shall not contain the fields Flags, Serial Number, GA or **GO index**.

2. Flags

Description: This field shall indicate whether the contained GA is the sending GA for the referred GO or not.

Encoding:

bit nr	7	6	5	4	3	2	1	0
name:	r	r	r	r	r	r	act	s

• Sending (s)

position: bit 0

description: This field *Sending* (s) shall indicate whether the contained (extended) GA is the sending GA for the referred GO or not.

format, encoding: 0: not sending: The contained GA is not be the sending GA for the referred GO.
1: sending: The contained GA is the sending GA for the referred GO.

- **Address extension type (aet)**

position: bit 1

description: This field (aet) shall indicate whether the link contains the DoA or the KNX SN.
Only value 1 shall be supported.

format, encoding: 0 : SN
1: DoA

- **reserved (r)**

position: bits 7 to 2

description: These bits are reserved.

format, encoding: The E-Mode Management Client shall clear (0) these bits.
The E-Mode Management Client shall ignore the value of these bits.

3. KNX Serial Number (SN) or RF Domain Address (DoA)

Description: If the field AET equals 0, then this field shall contain the KNX Serial Number of the extended GA. If the field AET equals 1, then this field shall be 000000000000.

If the Group Address is not an Extended Group Address, then the field KNX Serial Number shall be 0.

Encoding: DPT_SerNum (DPT_ID = 221.001) or 000000000000

4. Group Address (GA)

Description: This field shall be the basic Group Address or the Group Address part of the Extended Group Address, for which the contained Link information is responded.

Encoding: U₁₆ value

5. GO index

Description: This shall be the GO index that shall identify the Group Object within the Channel (same value as used to refer Group Object in the Property Value arrays
PID_GO_CCODES_LIST, PID_GO_SUBUNIT and PID_GO_NAME_LIST)

This shall be the GO number of the GO within the E-Mode Channel. This shall start with 1.

Encoding: U₁₆ value

2.3.4.13.2.3 Read number of free associations (Code = 01h)

2.3.4.13.2.3.1 Usage by the Management Client

The Management Client shall use the Function *Read Object Link* to read the number of free associations (with existing GAs or with new GAs) in a Management Server (device).

Use Case 1: a MaC may want to know, before adding several links that must be added together, whether he can add or not all the associations. Moreover, he already knows the GA that are already been used by the MaS. So if the MaS sends him the information on the number of free associations with existing and with new Gas, then he is able to decide to add or not all associations.

Use Case 2: for diagnostic purpose, an installer may want to know whether a device (MaS) is close to its limits in term of associations, in order to decide on an extension of the installation. If the number of free associations is below a certain level, then the MaC may warn the user (installer) that the present device may not support this extension and the installer should use a new device or modify its installation.

2.3.4.13.2.3.2 Usage by the Management Server (device)

The Management Server shall execute the Function *Read Object Link*. It shall respond to the Management Client with the response as given in Figure 7.

octet 10	octet 11	octet 12	octet 13	octet 14
Return Code	NFAEGA (high)	NFAEGA (low)	NFANGA (high)	NFANGA (low)

**Figure 8 – Response to the Function *Read Object Link* (case code = 01h)
A_FunctionPropertyState_Response-PDU (example)**

1. Return Code

Description: The Return Code shall give information on the success or possible errors that occur in the execution of the Function *Read Object Link* (code = 01h).

Encoding: 00h: This shall signal a successful answer.
FFh: This shall signal an error.

2. Number of Free Associations to Existing Group Addresses (NFAEGA)

Description: The octets 11 and 12 contain the high and low octets of the number of free associations to existing Group Addresses (GA).

Encoding: U₁₆ value

3. Number of Free Associations with New Group Addresses (NFANGA)

Description: The octets 13 and 14 contain the high and low octets of the number of free associations with new Group Addresses (GA).

Encoding: U₁₆ value

2.3.4.14 E-Mode Channel Object – PID_GO_SUBUNIT (PID = 64)

- Property name: Subunit Number
- Property Datatype: PDT_UNSIGNED_CHAR[]
- Datapoint Type: DPT_Value_1_Ucount (5.010)

The Property Subunit Number shall be an array Property with number of elements equal to the number of Group Objects for this E-Mode Channel.

It shall specify for each Group Object n in this E-Mode Channel in Property Value Array element n the subunit number in the E-Mode Channel to which this Group Object belongs.

GO nr	Property Value Property Index	Property Value
-	0	(current nr. of array elements)
1	1	Subunit of GO 1
2	2	Subunit of GO 2
3	3	Subunit of GO 3
...

Table 8 – PID_GO_SUBUNIT format

PID_GO_SUBUNIT is always mandatory. If there are no Subunits, then the value of PID_GO_SUBUNIT shall be 0. Also the value of PID_GO_SUBUNIT of GOs that do not belong to a Subunit shall be 0. If there is only one Subunit, then the value for PID_GO_SUBUNIT for its DP's shall be 1.

Subunits shall be numbered consecutively, starting with 1.

2.3.4.15 E-Mode Channel Object – PID_GO_NAME_LIST (PID = 65)

- Property name: Group Object Names
- Property Datatype: - PDT_GENERIC_10[]
PDT_REFERENCE[]
- Datapoint Type: DPT_UTF_8 (DPT_ID: 28.001)

The Property *Group Object Names* shall be an array Property with number of elements equal to the number of *Group Objects* for this E-Mode Channel.

It shall specify for each Group Object the human readable description and identification of this Group Object.

The Property Value array index used in this Property shall refer to the same Group Object as referred to in the array element with the same index in the Properties PID_GO_CCOCES_LIST, PID_GO_CFLAGS_LIST and PID_GO_SUBUNIT.

PID_GO_NAME_LIST		
GO nr	Property Value Array Index.	Property Value = GO names
	0	(current nr. of array elements)
1	1	"A Switch"
2	2	"A Dimming"
3	3	"A State"

In this example, it can be read that "A State" is the name of the 4th Group Object in this device.

2.3.4.16 E-Mode Channel Object – PID_PARAM_TYPES (PID = 70)

- Property name: Parameter Types
- Property Datatype: PDT_GENERIC_10[]
- Datapoint Type: None.

2.3.4.16.1 Format

The Property *Parameter Types* shall be an array Property with number of elements equal to the number of Parameters for this E-Mode Channel.

It shall specify for each Parameter n in this E-Mode Channel the Parameter Type Identifier, as defined in Table 9.

Table 9 – PID_PARAM_TYPES format

Parameter nr	Property Value Array Index	Property Value	
-	0	(current nr. of array elements)	2 octets
1	1	Parameter Type of Parameter 1	10 octets
2	2	Parameter Type of Parameter 2	10 octets
3	3	Parameter Type of Parameter 3	10 octets
...	

The encoding of the Parameter Type is flexible and shall depend on a main type indication in the most significant octet as specified in Table 10.

Table 10 – Parameter Type definitions

Format	Octet									
	0 (MSB) Type	1	2	3	4	5	6	7	8	9 (LSB)
Reserved	0	00h	00h	00h	00h	00h	00h	00h	00h	00h
Standard	1	Standard Parameter Type		00h	00h	00h	00h	00h	00h	00h
Bool	2	Type	00h	00h	00h	00h	00h	00h	00h	00h
Enum	3	Count	Reference Type	Object Type		Object Instance	Object Index	PID	0 (4 bit)	Start Index (12 bit)
INT8	4	Flags	Steps	00h	Minimum	00h	Maximum	00h	00h	00h
UINT8	5	Flags	Steps	00h	Minimum	00h	Maximum	00h	00h	00h
INT16	6	Flags	Steps	Minimum		Maximum		00h	00h	00h
UINT16	7	Flags	Steps	Minimum		Maximum		00h	00h	00h
FLOAT16	8	Flags	Steps	Minimum		Maximum		00h	00h	00h
Enum extended	9	Count	Reference Type	Object Type		Object Instance (12 bits)		PID (12 bits)		Start Index

NOTE 7 A possible future additional Parameter Type may be String.

◆ **Format 0: “Reserved”**

- Use: This Parameter Type format is reserved and shall not be used.

◆ **Format 1: “Standard”**

- Use: This Parameter Type format shall be used to refer to a Standard Parameter Type encoding.
If the Parameter is encoded according a standard Parameter Type then this format shall be used and none of the below formats.
- Encoding: The Parameter Type format shall be identified by the field *Standard Parameter Type* with value as given in Table 24.

◆ **Format 2: “Bool”**

- Use: This Parameter Type format shall be used to identify a Boolean Parameter.
- Encoding: **Type**
For the Boolean Parameter Type, standard labels are defined.
This avoids that the Management Server needs to define this
PID_PARAM_UNITS as enum labels in a Text Catalogue Object, as is done for
the type “Enum” below.
The value of Type shall equal the Subtype value of the 1 bit DPTs in [09].

Table 11 – Parameter Type

Type	Parameter Value	Label for Value
1	0	Off
	1	On
2	0	False
	1	True
3	0	Disable
	1	Enable
...	...	

EXAMPLE 13 If the Ctrl FEC Controller discovers a Parameter Format 2 “Bool” with Type 3, then it knows it can show on its display the texts “Disable” and “Enable”, which will stand for the values 0 and 1 respectively, without further reading a Text Catalogue Object.

NOTE 8 If none of the defined 1 bit DPTs can be meaningfully referred, a Boolean Parameter Format can also be described as an enumerated Parameter Format limited to two values only. See below.

◆ **Format 3: “Enum”**

- Use: This Parameter Type format shall be used to identify an enumeration Parameter. This matches a *real value*, which is written by the Management Client in the Management Server, with a *displayed value*, which is read by the Management Client from the Management Server and is displayed to the user.
- Encoding: **Count**
The real values of enumerated Parameter Type shall be encoded as unsigned 8 bit integers; the minimal real value shall always be 0 and the next real values shall be consecutive without gaps; the field *Count* shall contain the number of elements in the enumeration and shall thus equal the maximal value + 1. *Count* shall at maximum have the value 191; the values 192 to 255 for *Count* are reserved and shall not be used.

NOTE 9 The real values set by the Management Client will be as requested by the installer within the consecutive range 0 to Count - 1. By implementation specific means, it is however possible to map these device internal to other values.

EXAMPLE 14 (Count = 3)

displayed value = User's choice	real value = Set by Management Client in Management Server	Management Server's internal mapping.
Frost protection	0	10 °C
Economy mode	1	18 °C
Comfort mode	2	21 °C

ObjIdx and PID

The displayed values for the enumerated Parameter Type format shall be stored in an instance of the Text Catalogue Object (see 2.3.6). This shall be referred by the fields Reference Type, Object Type, Object Instance, Object Index, PID and Start Index, which shall all be encoded and used as specified for PDT_REFERENCE **OLD** in 2.7.2. The values of these fields shall in this way build a reference to the first displayed value of the enumeration; the next displayed values shall be read as is indicated in the field subfield Subtype of Reference Type (see 2.7.2).

◆ Format 4: “INT8”

- Use: This Parameter Type format shall be used to identify a Parameter value encoded as an 8 bit signed integer value.
- Encoding: **Flags**

This field shall give common information for numerical Parameter Values.

Octet	1							
Bit	7	6	5	4	3	2	1	0
Name	r	r	r	r	r	r	Max	Min
Value	0	0	0	0	0	0		

- Min

- 0: The Parameter encoded according this Parameter Type does not have a minimal value. The smallest encodable minimal value shall be the minimal value for the Parameter Value.
- 1: The Parameter encoded according this Parameter Type does have a minimal value. This minimal value shall be read from the field “min” in this Parameter Type definition.

- Max

- 0: The Parameter encoded according this Parameter Type does not have a maximal value. The largest encodable maximal value shall be the maximal value for the Parameter Value.
- 1: The Parameter encoded according this Parameter Type does have a maximal value. This maximal value shall be read from the field “max” in this Parameter Type definition.

Steps

If not all possible values between the minimal value and the maximal value are supported or allowed, but if only a limited number of equally spread intermediate values is possible, the field *Steps* shall be used to indicate this possible number of intermediate values.

Steps shall be encoded as an 8 bit unsigned integer value.

Octet	2							
Bit	7	6	5	4	3	2	1	0
Name	Steps							
Value								

NOTE 10 This avoids the introduction of an Enumeration Parameter Type (see below).

EXAMPLE 15 If the minimal value is 10 and the maximal value is 20 and Steps equals 21 then the possible values shall be 10.0, 10.5, 11.0 etc. to 19.5 and 20.

The encoding of *Steps* shall be as follows.

- 0: infinite
NOTE 11 This can be used for Parameter values that can be entered in an HMI as edit box input or keyboard input.
- 1 to 254: number of possible steps (“combo box selection”)
- 255: reserved. Shall not be used.

Minimum

The field *Minimum* shall contain the minimal value for the Parameter Value encoded according this Parameter Type.

This field shall only be interpreted if the field *Min* in the field *Flags* is set.

This field shall be encoded as the value for this Parameter itself, in this case thus as an 8 bit signed integer value.

Maximum

The field *Maximum* shall contain the maximal value for the Parameter Value encoded according this Parameter Type.

This field shall only be interpreted if the field *Max* in the field *Flags* is set.

This field shall be encoded as the value for this Parameter itself, in this case thus as an 8 bit signed integer value.

◆ Format 5: "UINT8"

- Use: This Parameter Type format shall be used to identify a Parameter value encoded as an 8 bit unsigned integer value.
- Encoding: The encoding of the fields *Flags* and *Steps* shall be identical as for the Parameter Type format INT8 specified above.
The fields *Minimum* and *Maximum* shall be encoded as an 8 bit unsigned integer value.

◆ Format 6: "INT16"

- Use: This Parameter Type format shall be used to identify a Parameter value encoded as a 16 bit signed integer value.
- Encoding: The encoding of the fields *Flags* and *Steps* shall be identical as for the Parameter Type format INT8 specified above.
The fields *Minimum* and *Maximum* shall be encoded as 16 bit signed integer values.

◆ Format 7: "UINT16"

- Use: This Parameter Type format shall be used to identify a Parameter value encoded as a 16 bit unsigned integer value.
- Encoding: The encoding of the fields *Flags* and *Steps* shall be identical as for the Parameter Type format INT8 specified above.
The fields *Minimum* and *Maximum* shall be encoded as 16 bit unsigned integer values.

◆ Format 8: "FLOAT16"

- Use: This Parameter Type format shall be used to identify a Parameter value encoded as a "2-Octet Float Value" as specified in [09] (F_{16} ; DPTs in the range 9.nnn).
- Encoding: The encoding of the fields *Flags* and *Steps* shall be identical as for the Parameter Type format INT8 specified here above.
The fields *Minimum* and *Maximum* shall be encoded as 16 bit float values (F_{16} ; see also [09]).

◆ Format 9: "Enum extended"

- Use: Same as for Format 3.
- Encoding: **Count**
Same as for Format 3.

Object Type, Object Instance and PID

The displayed values for the enumerated Parameter Type format shall be stored in an instance of the Text Catalogue Object (see 2.3.6). This shall be referred by the fields Reference Type, Object Type, Object Instance, PID and Start Index, which shall all be encoded and used as specified for PDT_REFERENCE in document [22]. The values of these fields shall in this way build a reference to the first displayed value of the enumeration; the next displayed values shall be read as is indicated in the field subfield Subtype of Reference Type (see 2.7.2). The MaS shall only reply with this reference type only if the MaS effectively also supports Extended Property Services.

2.3.4.17 E-Mode Channel Object – PID_PARAM_FLAGS (PID = 71)

- Property name: Parameter Flags
- Property Datatype: PDT_BITSET16[]
- Datapoint Type: None.

The Property *Parameter Flags* shall be an array Property with number of elements equal to the number of Parameters for this E-Mode Channel.

It shall specify for each Parameter n in this E-Mode Channel in Property Value array element n its Parameter Flags.

Parameter nr	Property Value Array Index	Property Value
-	0	(current nr. of array elements)
1	1	Parameter Flags of Parameter 1
2	2	Parameter Flags of Parameter 2
3	3	Parameter Flags of Parameter 3
...

Table 12 – PID_PARAM_FLAGS format

The parameter flags of one Parameter shall be encoded as specified in Table 13.

Table 13 – Coding of Parameter Flags

Bit Nr.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
field symbol	r	r	r	r	r	r	r	r	r	r	r	r	r	R	X	L
	0	0	0	0	0	0	0	0	0	0	0	0	0			

- bit 0 (lsb):
 - name (abbreviation): Local (L)
 - description: This flag shall indicate whether the Parameter is set locally or not.
 - encoding:
 - 0: This Parameter is not set locally: this means that if it is attempted to set this Parameter Value from remote, that this will be accepted.
 - 1: This Parameter is set locally. An attempt to set this parameter remotely will be refused by the device; this Parameter is currently “read-only”.
- bit 1:
 - name (abbreviation): Void (X)
 - description: This flag shall indicate whether this Parameter is void or not. The value of a void Parameter shall have no effect on the behaviour of the application.
This flag shall be independent of the value of the L-flag.
 - NOTE 12 A Parameter may for instance become void due to the value of another Parameter (such as AJS): the Parameter is still accessible but has no effect.
 - encoding:
 - 0: This Parameter is not void.
 - 1: This Parameter is void.

- bit 2 (lsb):
 - name (abbreviation): Restart (R)
 - description: This flag shall indicate whether the changing of this Parameter will have immediate effect, or will only become effective after a restart of the device.
 - encoding:
 - 0: If this Parameter is changed, it will have immediate effect.
 - 1: If this Parameter is changed, it will have effect only after the next restart of the device.

Ctrl FEC

If the Controller modifies one or more Parameters for which the R flag is set, it shall at the end of the Configuration Procedure restart the device ²⁾.

PB FEC

Any device that hosts the Parameter shall keep track if during the PB-Mode linking one or more Parameters are modified of which the flag R is set. If this is the case, the device shall after the linking has completed autonomously execute a restart.

- bits 15 to 3:
 - These bits are reserved and shall be cleared in the device.
 - The E-Mode Management Client shall ignore the value of these bits.

2.3.4.18 E-Mode Channel Object – PID_PARAM_NAMES (PID = 72)

- Property name: Parameter Names
- Property Datatype:
 - PDT_GENERIC_10[] (DPT_UTF_8 (DPT_ID: 28.001))
 - PDT_REFERENCE[]
- Datapoint Type: Every character to be encoded as DPT_UTF_8 (DPT_ID: 28.001)

The Property *Parameter Names* shall be an array Property with number of elements equal to the number of *Parameters* for this E-Mode Channel.

It shall specify for each Parameter n in this E-Mode Channel in Property Value array element n its parameter name, which shall be a human readable description and identification of the parameter.

Parameter nr	Property Value Array Index	Property Value
-	0	(current nr. of array elements)
1	1	Parameter Name of Parameter 1
2	2	Parameter Name of Parameter 2
3	3	Parameter Name of Parameter 3
...

Table 14 – PID_PARAM_NAMES format

²⁾ This allows that the Controller changes two more Parameters one after the other and that these only become active or evaluated together in a consistent way after a restart.

Every parameter name shall be a 10 octet value, of which every octet shall encode one textual character according DPT_UTF_8 (DPT_ID: 28.001), starting with the most significant octet; unused octets shall be filled with 00h.

2.3.4.19 E-Mode Channel Object – PID_PARAM_UNITS (PID = 73)

- Property name: Parameter Units
- Property Datatype:
 - PDT_GENERIC_10[] (DPT_UTF_8 (DPT_ID: 28.001))
 - PDT_REFERENCE[]
- Datapoint Type: None.

The Property *Parameter Units* shall be an array Property with number of elements equal to the number of *Parameters* for this E-Mode Channel.

It shall specify for each Parameter n in this E-Mode Channel in Property Value array element n its parameter unit as shown in Table 15.

2.3.4.19.1 Format

PID_PARAM_UNITS shall be an array Property.

Every array element shall be a string of textual Characters each encoded according DPT_UTF_8 (DPT_ID: 28.001), as shown in Table 15.

Table 15 – PID_PARAM_UNITS format

Parameter nr	Property Value Array Index	Property Value
-	0	(current nr. of array elements)
1	1	Unit of Parameter 1
2	2	Unit of Parameter 2
3	3	Unit of Parameter 3
4	4	Unit of Parameter 4
...

The characters shall be left aligned on most significant positions (left) of the Property Value array element; unused Characters shall have the value 00h and shall be positioned on the least significant positions (right) of the Property Value array element.

Parameter nr	Property Value Array Index	Property Value								
-	0	4								
1	1	4Dh	69h	6Eh	75h	74h	65h	73h	00h	Minutes
2	2	B0h	43hh	00h	00h	00h	00h	00h	00h	°C
3	3	25h	00h	00h	00h	00h	00h	00h	00h	%
4	4	4Ch	75h	78h	00h	00h	00h	00h	00h	Lux
...	...									

Figure 9 – Example PID_PARAM_UNITS

Alternatively, the PDT may be PDT_REFERENCE, in case of which all array elements shall be references to strings in a Catalogue Object.

2.3.4.19.2 Usage by the Management Server (device)

PID_PARAM_UNITS shall hold in each array element *n* human readable textual information about the physical unit of the Parameter Value *n* or a reference to it.

2.3.4.19.3 Usage by the Management Client

The Management Client shall read out PID_PARAM_UNITS during the discovery of the device in order to be able to give human understandable representations of the Parameter Values in its user interface.

2.3.4.20 E-Mode Channel Object – PID_PARAM_VALUES (PID = 79)

- Property name: Parameter Values
- Property Datatype:
 - PDT_GENERIC_01[], or
 - PDT_GENERIC_02[], or
 - PDT_GENERIC_04[], or
 - PDT_GENERIC_10[]
- Datapoint Type: None.

The Property *Parameter Values* shall be an array Property with number of elements equal to the number of *Parameters* for this E-Mode Channel.

It shall specify for each Parameter *n* in this E-Mode Channel in Property Value array element *n* its parameter value as shown in Table 16.

Table 16 – PID_PARAM_VALUES format

Parameter nr	Property Value Array Index	Property Value
-	0	(current nr. of array elements)
1	1	Parameter Value of Parameter 1
2	2	Parameter Value of Parameter 2
3	3	Parameter Value of Parameter 3
4	4	Parameter Value of Parameter 4
...

The Property Datatype of this Property may vary according the size of the Parameter Values to be stored. The PDT shall be any of the type PDT_GENERIC_01, PDT_GENERIC_02, PDT_GENERIC_04 or PDT_GENERIC_10.

Each parameter value shall be left adjusted in the Property Value array element value: the msb of the parameter value shall be the msb of the Property Value array element value. If the Parameter Size is smaller than the PDT size, then the unused Property Value array element bits and octets shall be at the lower positions and shall be 0.

EXAMPLE 16 Table 17 shows an example of PID_PARAM_VALUES with PDT_GENERIC_02. (Cells for unused bits are left empty; in a device, these shall be filled with 0).

Table 17 – PID_PARAM_VALUES Example

Parameter Type	Parameter Size	Parameter Number	Property Value Array Index	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			0									4							
PART_Boolean	1 bit	1	1	1															
PART_UpDown_Switch_Action	2 bit	2	2	0	1														
PART_COV_Lux	1 octet	3	3	0	0	0	0	1	0	1	0								
PART_Light_Value	2 octets	4	4	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0

2.3.4.20.1 Usage by the Management Client

If the Management Client writes the Property Value then it shall clear the unused bits.

Every *Parameter Value* *n* shall be encoded as specified by the *Parameter Type* in the array with the same element number *n* in the Property *Parameter Types* (PID_PARAM_TYPES) as specified in 2.3.4.16.

If the FEC Channel is an Adjustable Channel then the Adjustable Parameter shall be the first parameter in the Property Parameter Values.

The Management Client should hold the run state machine of the Application Program prior to changing any Parameter Value. Otherwise, it is possible that the Management Server (device) gives a negative A_PropertyValue_Write.res if a Parameter value is changed that it does not allow changing during runtime.

2.3.4.20.2 Usage by the Management Server

The Management Server may negatively confirm the write access to Parameter values (negative A_PropertyValue_Write.res) if it is requested to write a Parameter value that can not be changed without halting the Application Program.

The Management Server may use the unused fields in PID_PARAM_VALUES for internal purposes. If the Management Client however reads the value of PID_PARAM_VALUES, the Management Server shall guarantee that the response contains the value 0 for the unused fields.

2.3.5 Adjusted E-Mode Channel Object (Object Type 15)

2.3.5.1 General

Any instance of the Adjusted E-Mode Channel Object shall represent the adjustable data (description and configuration of Group Objects and Parameters) of one E-Mode Channel Object for one value of its Parameter AJS.

Please refer to 2.10.5 for the mandatory and optional Properties in the Adjusted E-Mode Channel Object.

NOTE 13 There are no requirements concerning the Object Indexes of the Adjusted E-Mode Channel Objects. These Object Indexes may or may not follow the Object Index of the parent E-Mode Channel Object and may or may not be consecutive.

Table 18 – Adjusted E-Mode Channel Object (informative)

PID Value	PID Name	Property Name	PDT/DPT	Access
1	PID_OBJECT_TYPE	Interface Object Type	DPT_PropDataType (DPT_ID = 7.010)	3/1
2	PID_OBJECT_NAME	Interface Object Name	- PDT_UNSIGNED_CHAR[] (DPT_UTF_8 (DPT_ID: 28.001)) - PDT_REFERENCE	3/1
61	PID_GO_CCODES_LIST	Connection Codes	PDT_GENERIC_08[]	3/1
62	PID_GO_CFLAGS_LIST	Connection Flags	PDT_BITSET16[]	3/1
63	PID_OBJECTLINK	Object Link	PDT_FUNCTION	3/3
65	PID_GO_NAME_LIST	Group Object Names	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
70	PID_PARAM_TYPES	Parameter Types	PDT_GENERIC_10[]	3/1
71	PID_PARAM_FLAGS	Parameter Flags	PDT_BITSET16[]	3/1
72	PID_PARAM_NAMES	Parameter Names	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
73	PID_PARAM_UNITS	Parameter Units	- PDT_GENERIC_10[] or - PDT_REFERENCE[]	3/1
79	PID_PARAM_VALUES	Parameter Values	- PDT_GENERIC_01[], or - PDT_GENERIC_02[], or - PDT_GENERIC_04[], or - PDT_GENERIC_10[]	3/3

2.3.5.2 Adjusted E-Mode Channel Object – PID_OBJECT_TYPE (PID = 1)

Please refer to [01] for the common specification of PID_OBJECT_TYPE.

The Object Type for the Adjusted E-Mode Channel Object shall have the value 15.

2.3.5.3 Adjusted E-Mode Channel Object – PID_OBJECT_NAME (PID = 2)

Please refer to the definition of Property *Object Name* in [01].

This Property *Object Name* shall contain or refer to the name of the Adjusted E-Mode Channel Object.

EXAMPLE 17 The first instance of the Object Adjusted E-Mode Channel, for AJS = 1, may have the name "PB Switch". The second instance of this Object, for AJS = 2, may have the name "PB Toggle".

2.3.5.4 Adjusted E-Mode Channel Object – PID_GO_CCODES_LIST (PID = 61)

Please refer to the specification of PID_GO_CCODES_LIST in 2.3.4.11.

In any IO *Adjusted E-Mode Channel*, this Property shall specify the Connection Codes that apply for the GOs used in this IO *Adjusted E-Mode Channel*. This list may differ in number and value compared to the related IO *E-Mode Channel Object* and its possible further IOs *Adjusted E-Mode Channel*.

2.3.5.5 Adjusted E-Mode Channel Object – PID_GO_CFLAGS_LIST (PID = 62)

Please refer to the specification of PID_GO_CFLAGS_LIST in 2.3.4.12.

2.3.5.6 Adjusted E-Mode Channel Object – PID_OBJECTLINK (PID = 63)

Please refer to the specification of PID_OBJECTLINK in 2.3.4.13.

2.3.5.7 Adjusted E-Mode Channel Object – PID_GO_NAME_LIST (PID = 65)

Please refer to the specification of PID_GO_NAME_LIST in 2.3.4.15.

2.3.5.8 Adjusted E-Mode Channel Object – PID_PARAM_TYPES (PID = 70)

Please refer to the specification of PID_PARAM_TYPES in 2.3.4.16.

2.3.5.9 Adjusted E-Mode Channel Object – PID_PARAM_FLAGS (PID = 71)

Please refer to the specification of PID_PARAM_FLAGS in 2.3.4.17.

2.3.5.10 Adjusted E-Mode Channel Object – PID_PARAM_NAMES (PID = 72)

Please refer to the specification of PID_PARAM_NAMES in 2.3.4.18.

2.3.5.11 Adjusted E-Mode Channel Object – PID_PARAM_UNITS (PID = 73)

Please refer to the specification of PID_PARAM_NAMES in 2.3.4.19.

2.3.5.12 Adjusted E-Mode Channel Object – PID_PARAM_VALUES (PID = 79)

Please refer to the specification of PID_PARAM_VALUES in 2.3.4.20.

2.3.6 Text Catalogue Object (Object Type 16)

Table 19 – Text Catalogue Object

	PID	Property Name	PDT and DPT	Description
1	PID_OBJECT_TYPE	Interface Object Type	DPT_PropDataType (DPT_ID = 7.010)	Type of the Interface Object
25	PID_VERSION	Version	PDT_VERSION (alt. PDT_GENERIC_02) DPT_Version (217.001)	Version of this Text Catalogue Object
51	PID_LOCALE_LIST	Locale List	PDT_GENERIC_04[] DPT_Locale_ASCII (231.001)	List of the supported languages and regions
52	PID_LOCALE_SELECTION	Locale Selection	PDT_GENERIC_04 DPT_Locale_ASCII (231.001)	To change the active language and region
53	PID_ACTIVE_LOCALE	Active Locale	PDT_GENERIC_04 DPT_Locale_ASCII (231.001)	To read the active language and region
60	PID_STRING_001	String 001	PDT_UTF_8[] DPT_UTF-8 (28.001)	Text string 001
...
nn	PID_STRING_mm	String mm	PDT_UTF_8[] DPT_UTF-8 (28.001)	Text string mm
...
200	PID_STRING_141	String 141	PDT_UTF_8[] DPT_UTF-8 (28.001)	Text string 141

2.3.6.1 General requirements

The Text Catalogue Object shall be used to hold in its Object Type dependent Properties amongst other Properties for storing and managing textual data.

There may be more than one instance of the Text Catalogue Object in a device. No standard functionality is foreseen for references or dependencies between Catalogue Object Instances.

EXAMPLE 18 If the language of one Catalogue Object is changed, the language of any other Catalogue Object is not affected. The Property Client thus has to observe for itself that when it accesses another Catalogue Object, that it should verify that IO's language.

2.3.6.2 PID_OBJECT_TYPE (PID = 1)

Please refer to 4.2.1 in [01] for the general requirements for PID_OBJECT_TYPE.

This Property shall contain Object Type of the Interface Object. The Text Catalogue Object shall have Object Type 16.

2.3.6.3 PID_VERSION (PID = 25)

Please refer to 4.2.25 in [01] for the general requirements for PID_OBJECT_TYPE.

This Property shall contain the version of the Text Catalogue Object.

2.3.6.4 PID_LOCALE_LIST (PID = 51)

- Property name: *Locale List*
- Property Datatype: PDT_GENERIC_04[]
- Datapoint Type: DPT_Locale_ASCII (see [17])


2.3.6.4.1 Abstract Resource definition

The Property *Locale List* shall be used to list the available language flavours and regional options of the textual data accessible in the device.

This Property shall allow discovery of the languages and regional dependencies for which textual information may be available in the device.

This list should be correct and complete. The following is the responsibility of the product manufacturer.

- This list should be complete. That is, it should contain combinations of language and region to which the textual data can be switched.
- This list should not contain languages or regions or combinations thereof that are not supported.

 *At the time of preparation of this version of the specification it has been accepted by KSG that if for a certain language no regional differences are supported, that then the Region field of DPT_Locale_ASCII can be set to ZZ. This use of DPT_Locale_ASCII is not part of this specification and will be discussed in KNX WGI.*

2.3.6.4.2 Encoding

This Property shall be an array of elements each encoded according DPT_Locale_ASCII.

There is no requirement concerning the sorting of this list.

Property Value Array Index	Value	Contents	Comment
0	4	nr. of elements	
1	6465h 4445h	"de DE"	German Germany
2	656Eh 4742h	"en GB"	English Great-Britain
3	6E6Ch 4245h	"nl BE"	Dutch Belgium
4	656Eh 5553h	"en US"	English United States

Figure 10 – PID_LOCALE_LIST

2.3.6.4.3 Usage by the Management Server (device)

The Management Server shall list in the *Locale List* all locale options (languages and regions) that it provides for the accessible textual information in it.

2.3.6.4.4 Usage by the Management Client

The Management Client shall read this Property to discover all possible regional options for which textual information can be accessed in the device.

If this Property is not available, the Management Client may assume that the language- and regional options of the textual data in the device cannot be selected.

2.3.6.5 PID_LOCALE_SELECTION (PID = 52)

- Property name: *Locale Selection*
- Property Datatype: PDT_GENERIC_04
- Datapoint Type: DPT_Locale_ASCII (see [17])

2.3.6.5.1 Abstract Resource definition

The Property Locale Selection shall allow the Management Client to set one language and region for the Text Catalogue Object.

2.3.6.5.2 Encoding

This Property shall be encoded according DPT_Locale_ASCII.

Property Value Array Index	Value	Contents	Comment
0	1	nr. of elements	
1	656Eh 4742h	"en GB"	English Great-Britain is requested

2.3.6.5.3 Usage by the Management Server (device)

The Management Server shall use this Property Value to learn the language and regional options selected by the Management Client.

This Property shall always be writeable.

If the language and regional options of the textual Properties can be switched, then the Property Server shall switch the contents of the textual Properties in this Text Catalogue Object according the value set by the Management Client. This means that the Management Server shall on a change of PID_LOCALE_SELECTION to a supported language and region, accordingly change the Values of the Properties PID_STRING_001 to PID_STRING_141.

The Management Server may activate the selected language and region either as soon as possible or only after a restart.

When the selected language and region become active, the Management Server shall set the value of PID_ACTIVE_LOCALE accordingly.

Error handling

If an A_PropertyValue_Write-PDU is received with a value for PID_LOCALE_SELECTION for a language and region that are not supported by the Management Server, then the Management Server shall not switch the language or regional options and shall respond with a negative A_PropertyValue_Response-PDU.

2.3.6.5.4 Usage by the Management Client

The Management Client shall use PID_LOCALE_SELECTION to select the active language and regional options in the hosting Text Catalogue Object. After changing PID_LOCALE_SELECTION, the Management Client should read PID_ACTIVE_LOCALE. If the value set for PID_LOCALE_SELECTION and the read value from PID_ACTIVE_LOCALE do not equal, and if the Management Client did not get a negative response when setting PID_LOCALE_SELECTION, then the Management Client may assume that the Management Server requires a restart before the selected locale becomes effective.

2.3.6.6 PID_ACTIVE_LOCALE (PID = 53)

- Property name: *Active Locale*
- Property Datatype: PDT_GENERIC_04
- Datapoint Type: DPT_Locale_ASCII (see [17])

2.3.6.6.1 Abstract Resource definition

In this Property, the Management Server shall at any time hold the currently active language and regional options.

2.3.6.6.2 Encoding

This Property shall be encoded according DPT_Locale_ASCII.

Property Value Array Index	Value	Contents	Comment
0	1	nr. of elements	
1	656Eh 4742h	"en GB"	English Great-Britain is active

2.3.6.6.3 Usage by the Management Server (device)

The Management Server shall in this Property at any time hold the currently active language and regional options.

This Property Value shall be any of the elements contained in PID_LOCALE_LIST (see 2.3.6.4), if available.

2.3.6.6.4 Usage by the Management Client

The Management Client shall read PID_ACTIVE_LOCALE to learn about the active language and regional options in the *Text Catalogue Object*.

In particular, when the Management Client has requested a change of the locale options through PID_LOCALE_SELECTION, it shall read PID_ACTIVE_LOCALE to check whether the selected language and regional options become effective immediately or only after a restart.

2.3.6.7 PID_STRING_001 (PID = 60) to PID_STRING_141 (PID = 200)

- Property name: String 001 to String 141
- Property Datatype: PDT_UTF_8 (A[n], DPT_UTF-8)
- Datapoint Type: See above.

2.3.6.7.1 Abstract Resource definition

These Properties shall be used to hold the textual data of the *Text Catalogue Object*.

The Property Datatypes shall be PDT_UTF_8 (A[n]).

These Properties are intended to be referred to by textual Properties in other Interface Objects in the Management Server, that shall then be of type PDT_REFERENCE (see 2.7.2). The semantic meaning of the Properties PID_STRING_001 to PID_STRING_141 is thus only given by the referring Properties. It should be noted that PDT_REFERENCE allows referring to a start_index within an array Property. This allows that one Property holds more than one string.

The Text Catalogue Object shall at least contain the Property PID_STRING_001; additional Properties in the range PID_STRING_002 to PID_STRING_141 shall have subsequent Property Identifiers.

2.3.6.7.2 Usage by the Management Server (device)

The Management Server shall make available in these Properties textual data that can be read out in one or more languages selectable by the Management Client.

These Properties may be referred by textual Properties in other Interface Objects by using PDT_REFERENCE as alternative Property Datatype.

If the Management Server accepts a change of language of the *Text Catalogue Object* through PID_LOCALE_SELECTION by the Management Client, then the Management Server shall make accessible through these Properties the textual data according the selected language and regional options. This may require a restart of the Management Server.

The correctness and the completeness of these language values are the responsibility and choice of the device manufacturer. If for a certain language, not all textual information is translated, then it may be that the textual information is provided when read out in another language than the activated language.

EXAMPLE 19 If "Dutch" is selected, but for "Ausgang A" the manufacturer has not given the Dutch translation, then still "Ausgang A" may be provided when the textual Property Value be read out.

2.3.6.7.3 Usage by the Management Client

The Management Client shall read the Values of the Properties PID_STRING_001 to PID_STRING_141 – as far as implemented in the Management Server – to read textual information.

2.3.7 E-Mode Device Object (Object Type 18)

2.3.7.1 Overview

Table 20 – E-Mode Device Object

PID	Property Name	PDT and DPT	Description
1 PID_OBJECT_TYPE	Interface Object Type	DPT_PropDataType (DPT_ID = 7.010)	Type of the Interface Object
60 PID_LOCALISATION_MODE	Localisation Mode	DPT_State (DPT_ID = 1.011) PDT_BINARY_INFORMATION (alt.: PDT_UNSIGNED_CHAR)	To activate and inactivate the Localisation Mode.
61 PID_LOCALISATION_REPORT	Localisation Report	PDT_UNSIGNED_INT U ₁₆	To report on human operations performed for localisation.
62 PID_LOCALISATION_COMMAND	Localisation Command	PDT_GENERIC_03 U ₈ U ₁₆	To receive localisation commands.

2.3.7.2 General requirements

The E-Mode Device Object shall be used to hold in its Object Type dependent Properties amongst other Properties for the Property based Management of E-Mode FEC devices for:

- localisation,
- E-Mode Channel independent Parameters
- etc.

2.3.7.3 E-Mode Device Object – PID_OBJECT_TYPE (PID = 1)

Please refer to [01] for the common specification of PID_OBJECT_TYPE.

The Object Type for the E-Mode Device Object shall have the value 18.

2.3.7.4 E-Mode Device Object – PID_LOCALISATION_MODE (PID = 60)

- Property name: Localisation Mode
- Property Datatype: PDT_BINARY_INFORMATION (alt.: PDT_UNSIGNED_CHAR)
- Datapoint Type: DPT_State (DPT_ID = 1.011)

2.3.7.4.1 Usage by the Management Server (device)

This Property shall control whether the *Localisation Mode* is inactive or active.

NOTE 14 Please refer to [07] for the definition of *Localisation Mode*.

If the Management Client sets PID_LOCALISATION_MODE = Active, then the Management Server shall immediately activate *Localisation Mode*. It shall also temporarily store the Source Address of the Management Client as long as the *Localisation Mode* is active. If the Management Client sets PID_LOCALISATION_MODE = Inactive, then the Management Server shall immediately inactivate *Localisation Mode* and may clear the Individual Address information of the Management Client.

The default value shall be 0 = Inactive.

Once *Localisation Mode* has become Active, the Management Server shall start a time-out timer of 3:30 minutes. This time-out timer shall be restarted with each activation or repeated activation of the Localisation Mode through PID_LOCALISATION_MODE by the Management Client. If this time-out timer expires, the Management Server shall autonomously inactivate *Localisation Mode*.

NOTE 15 It is not foreseen that the Management Server informs the Management Client in any way if it autonomously inactivates its *Localisation Mode*.

2.3.7.4.2 Usage by the Management Client

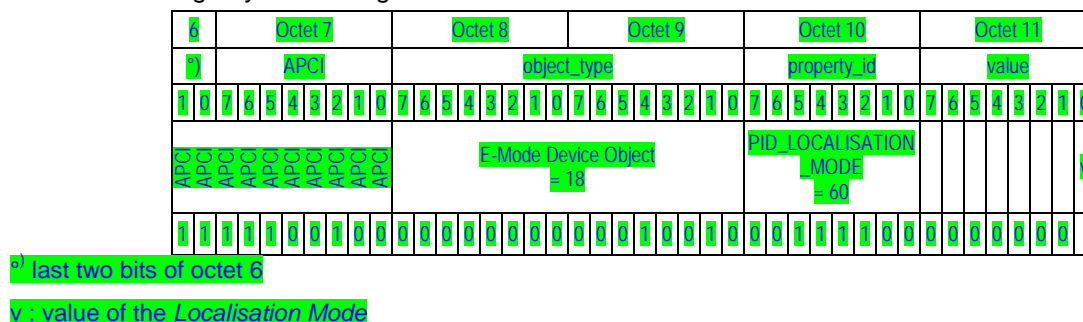


Figure 11 – A_NetworkParameter_Write-PDU with Localisation Mode (example)

The Management Client shall activate the *Localisation Mode* in the Management Server by setting PID_LOCALISATION_MODE to 1 via broadcast communication **mode in all devices supporting this feature in the installation or via point-to-point connectionless communication mode in one single device**. As the *Localisation Mode* automatically times out in the Management Server after 3:30 minutes, the Management Client shall resend the localisation request every 1 minute as long as it is required by the installer via the Controller user interface.

```
/* Activate the Localisation Mode in all FEC devices supporting this feature in the installation */
NM_NetworkParameter_Write_R(ASAP = void; comm_mode = broadcast; hop_count_type_request = Network Layer
Parameter; object_type = E-Mode Device Object, PID = PID_LOCALISATION_MODE, priority = system,
value = 1)

OR

/* Activate the Localisation Mode in one single device */
NM_NetworkParameter_Write_R(ASAP = Device.IA; comm_mode = point-to-point connectionless;
hop_count_type_request = "Network Layer Parameter"; object_type = E-Mode Device Object,
PID = PID_LOCALISATION_MODE, priority = system, value = 1)

/* The FEC devices shall store the Source Address of this Procedure for later use. */
```

If the Controller wants to quit the *Localisation Mode*, it shall explicitly do this (this is; it shall not wait until the *Localisation Mode* automatically becomes inactive in the devices).

```
/* Inactivate the Localisation Mode in all FEC devices in the installation.*/
NM_NetworkParameter_Write_R(ASAP = void; comm_mode = broadcast; hop_count_type_request = "Network Layer
Parameter"; object_type = E-Mode Device Object, PID = PID_LOCALISATION_MODE, priority = system,
value = 0)
```

2.3.7.5 E-Mode Device Object – PID_LOCALISATION_REPORT (PID = 61)

- Property name: Localisation Report
- Property Datatype: PDT_UNSIGNED_INT
- Datapoint Type: None.

2.3.7.5.1 Format

The Property *Localisation Report* shall have the format as specified in Figure 12.

octet	octet 0								octet 1							
bit nr	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
field	Channel Number															
value																

Figure 12 – Localisation Report format

- **octet 0 and 1: Channel Number**

definition: This shall contain the number of the device Channel on which a user localisation action is performed. This shall be the value of the Property PID_CHAN_NUMBER of the E-Mode Channel for which the localisation action is reported.

encoding: 2 octets: U_{16}
E-Mode Channel numbers shall be counted starting from 1.
The Channel Number 0000h shall be reserved to denote “the entire device” or all E-Mode Channels.

2.3.7.5.2 Usage by the Management Server (device)

If *Localisation Mode* is active in a FEC device and if a user localisation action is performed, then the FEC device shall send an A_NetworkParameter_InfoReport-PDU in point-to-point communication mode to the IA of the Management Client that has activated the *Localisation Mode* as follows.

6	Octet 7								Octet 8								Octet 9								Octet 10								Octet 11								Octet 12								Octet 13																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
°)	APCI								object_type																property_id								test_info								test_result																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	E-Mode Device Object = 18									PID_LOCALISATION _REPORT = 61								reserved 00h								Channel Number																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
1	1	1	1	1	0	1	1	0		1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

°) last two bits of octet 6

Figure 13 – A_NetworkParameter_InfoReport-PDU with Localisation Report (example)

```
/* Report on a localisation action on E-Mode Channel n. */
DMP_InterfaceObjectInfoReport_RCI(mpp_ASAP = Controller.IA, mpp_Obj.Type = E-Mode Device Object,
  mpp_Prop.ID = PID_LOCALISATION_REPORT, mpp_Test_Info = 00h, mpp_Test_Result = Channel Number)
```

NOTE 16 The field test_info shall be 00h. This field can be used for later extensions.

The *Localisation Report* is typical for but not limited to sensors.

2.3.7.5.3 Usage by the Management Client

The Management Client shall accept the A_NetworkParameter_InfoReport-PDU from the Management Server only under the following conditions:

- 1 it has before activated the *Localisation Mode* in that Management Server, and
- 2 this *Localisation Mode* is not yet inactive, this is, the Management Client has itself not yet inactivated the *Localisation Mode* in that Management Server.

In all other cases, the Management Client shall ignore the information report on PID_LOCALISATION_REPORT.

If the Management Client accepts the Localisation report then it shall interpret it appropriately in its internal data and in its user interface.

2.3.7.6 E-Mode Device Object – PID_LOCALISATION_COMMAND (PID = 62)

- Property name: Localisation Command
- Property Datatype: PDT_GENERIC_03
- Datapoint Type: None.

NOTE 17 This clause uses the term “device localisation action”. For the definition please refer to 2.1.

2.3.7.6.1 Format

octet	octet 0								octet 1								octet 2							
bit nr	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
field	r	r	r	r	r	r	P	S	Channel Number															
value	0	0	0	0	0	0																		

Figure 14 – Localisation Command format

- octet 0 - bit 7 to 2:
 - name (abbreviation): reserved (r)
 - description: These bits are reserved.
 - encoding: These bits shall be cleared by the E-Mode Management Client. The Management Server (device) shall ignore the value of these bits **and shall not react**.

- octet 0 - bit 1:

name (abbreviation):	Physical (P)
description:	It is possible that an E-Mode Channel exhibits two different localisation actions. This bit Physical shall indicate whether the device localisation action shall be physical action or not.
encoding:	0: Physical action 1: Not a physical action It is optional to support "Not a physical action" for a Management Server. If this is not supported and a Localisation command is received with this value then the Management Server shall not react.
- octet 0 - bit 0:

name (abbreviation):	Start (S)
description:	This bit shall indicate whether the device localisation action shall be started or stopped.
encoding:	0: Stop 1: Start
- octet 1 and 2:

name (abbreviation):	Channel Number
description:	This shall be the Channel Number for which it is requested that the device localisation action be started or stopped. This shall be the value of the Property PID_CHAN_NUMBER of the E-Mode Channel for which the localisation action is requested
encoding:	The value shall be binary encoded. 0000h: The device localisation action shall be started or stopped for all E-Mode Channels in this device or for the entire device. 0001h to FFFFh: The device localisation action shall be started or stopped for the E-Mode Channel given by this value.

2.3.7.6.2 Usage by the Management Server (device)

The *Localisation Command* is typical for but not limited to actuators.

The Management Server shall interpret write accesses to PID_LOCALISATION_COMMAND exclusively if *Localisation Mode* is active.

If *Localisation Command* is written, then the Management Server shall perform its device localisation action for the contained E-Mode Channel as requested.

The device localisation action shall be started or stopped as indicated by the field *Start*. The device localisation action shall at the latest stop if *Localisation Mode* becomes inactive in the Management Server.

Error and exception handling

The Management Server (device) shall ignore the Localisation Command in the following cases.

- The Localisation Command contains an E-Mode Channel number that is not supported or for which no localisation command is supported.
- The Localisation Command contains a Physical Action that is not supported for the contained E-Mode Channel(s).
- The Localisation Command is sent with a different IA (so from a different Management Client) than the one who activates the Localisation Mode.

2.3.7.6.3 Usage by the Management Client

The installer can request through the HMI of the Management Client that the actuator performs this device localisation action on a user selected E-Mode Channel.

[illegible]

o) last two bits of octet 6

Figure 15 – A_NetworkParameter_Write-PDU with Localisation Command (example)

```

/* The Controller request the actuator to start a physical device localisation action on a Channel. */
/* This procedure is called by the Controller. It sends a message from the Controller to the device. */
NM_NetworkParameter_Write_R(ASAP = Device.IA; comm_mode = point-to-point connectionless;
hop_count_type_request = "Network Layer Parameter"; object_type = E-Mode Device Object,
PID = PID_LOCALISATION_COMMAND, priority = system, value = {01h, Channel Number})


```

The Management Client may call the A_NetworkParameter_Write-service in point-to-point connectionless communication mode as well as in point-to-all-points (broadcast) communication mode.

NOTE 18 The latter is interesting if the human has called multiple localisation effects in multiple E-Mode Channels and wants to stop all effect.

2.4 Management Procedures

2.4.1 Procedure: DMP Connect RCI

 The current specification of DMP_Connect_RCI only foresees the reading of DD0. This has to be extended to allow reading and receiving also DD2.
This clause shall replace the current specification of DMP_Connect_RCI in [02].

This Management Procedure shall be used to read the Device Descriptor of one Management Server (device). It shall allow differentiating between requesting DD0 and DD2 and is capable of handling both DD-types expected and unexpected.

This Management Procedure shall use the point-to-point connectionless communication mode.

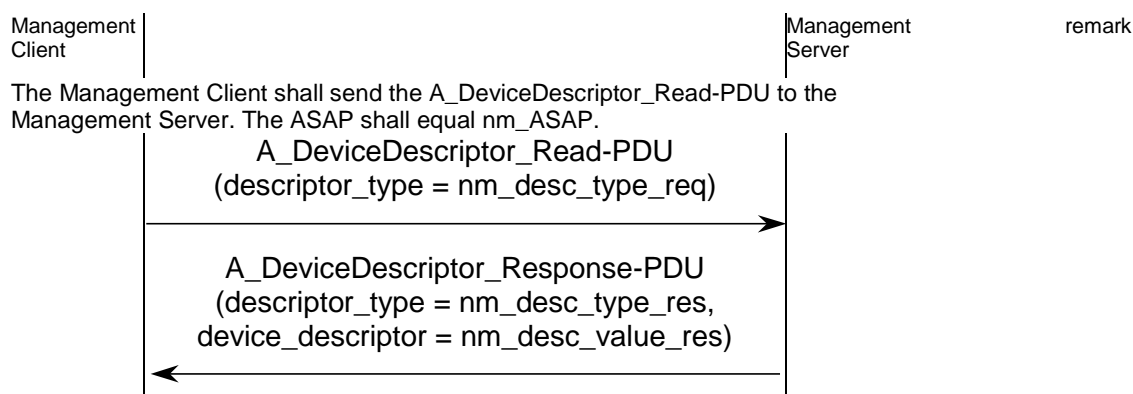
DMP_Connect_RCI (/* [in] */ nm_ASAP, /* [in] */ nm_desc_type_req,
/* [out] */ nm_desc_type_res, /* [out] */ nm_desc_value_res)

nm_ASAP:	The parameter shall contain the IA of the communication partner of which the Device Descriptor is to be read.
nm_desc_type_req:	This parameter shall contain the requested Device Descriptor Type; it can be DD0 or DD2.
nm_desc_type_res:	This result shall contain the DD-type with which the Management Server has responded. This may be different from nm_desc_type_req. Please refer to the exception handling.
nm_desc_value_res:	This result shall contain the Device Descriptor value responded by the Management Server.

Used Application Layer services for Management

- A_DeviceDescriptor_Read

Sequence



Exception handling

There are several error situations when building up a connectionless communication.

- If no A_DeviceDescriptor_Response-PDU is received, the Management Server with the Individual Address does not exist or the network is not configured correctly.
- If more than one A_DeviceDescriptor_Response-PDU is received, there is more than one device with the target address.

If the above indicates the network may be configured incorrectly, than the network topology shall be checked. In all other cases, the DM_Connect_RCI may be repeated several times. If this procedure is not successful, the depending procedures shall be aborted.

2.5 Configuration Procedures

2.5.1 Flexible E-Mode Channels

2.5.1.1 Basic model – Fixed E-Mode Channel Object

 Please refer to 2.1 for the definition of the Fixed E-Mode Channel/

Every E-Mode Channel shall be represented as an Interface Object of the type E-Mode Channel (Object Type 14) as specified in 2.3.4. This E-Mode Channel Object shall contain all data for description and configuration of:

- the E-Mode Channel itself, and
- its Group Objects, and
- its Parameters.

Table 21 – E-Mode Channel Object (example)

NOTE 19 This is a worked out example of the existing E-Mode Channel CH_PB_Timed_Info.

PID Value	PID Name	Property Name	Value (remark)
1	PID_OBJECT_TYPE	Interface Object Type	14
2	PID_OBJECT_NAME	Interface Object Name	[52h, 6Fh, 63h, 6Bh, 65h, 72h, 20h, 54h, 6Fh, 70h] ("Rocker Top")
25	PID_VERSION	Version	1003h (version 2.0.3)
51	PID_CHAN_NUMBER	Channel Number	1
52	PID_CHAN_CODE	Channel Code	0004h (CH_PB_Timed_Info)
53	PID_CHAN_FLAGS	Channel Flags	000000000100010b (non-adjustable input E-Mode Channel; supporting Localisation Reports)
54	PID_CHAN_FB_LIST	Functional Block List	[0196h, 01A5h] (406 = FB Timed Sensor, 421 = FB Switching Sensor Basic)
55	PID_CHAN_ADJ_LISTS	Adjustable Channel Object List	Not implemented: this is not an Adjustable E-Mode Channel.
61	PID_GO_CC_CODES_LIST	Connection Codes	[0002000000000000h, 0003000000000000h] (GO : CC_Switched_OnOff_Status +0 + 0 + 0 GO : CC_Timed + 0 + 0 + 0)
62	PID_GO_CFLAGS_LIST	Connection Flags	[0004h, 0202] (GO : {I}, GO : {O, L})
63	PID_OBJECTLINK	Object Link	PDT_FUNCTION
64	PID_GO_SUBUNIT	Subunit Number	[00h, 00h] (GO : does not belong to a subunit, GO : does not belong to a subunit)
65	PID_GO_NAME_LIST	Group Object Names	[53746174757320686E2031h, 54696D64204F75742031h] (GO : "Status In 1", GO : "Timd Out 1")
70	PID_PARAM_TYPES	Parameter Types	(This E-Mode Channel does not have Parameters.)
71	PID_PARAM_FLAGS	Parameter Flags	(This E-Mode Channel does not have Parameters.)
72	PID_PARAM_NAMES	Parameter Names	(This E-Mode Channel does not have Parameters.)
73	PID_PARAM_UNITS	Parameter Units	(This E-Mode Channel does not have Parameters.)
79	PID_PARAM_VALUES	Parameter Values	(This E-Mode Channel does not have Parameters.)

2.5.1.2 Adjusted E-Mode Channel Object

2.5.1.2.1 Specification

The concept of Adjustable the E-Mode Channel shall allow for alternative description and configuration of

- its Group Objects, and
- its Parameters.

This is controlled through the Adjustable Parameter (AJS), which shall always be the first parameter of the E-Mode Channel Object.

If the AJS has value 0, then the description and configuration of Group Objects and Parameters of the E-Mode Channel shall apply.

Every higher value of AJS (1, 2, 3...) shall specify alternative descriptions and configuration of Group Objects and Parameters of the same E-Mode Channel, which shall however be accessed through 1 or more instances of the Adjusted E-Mode Channel Object as specified in 2.3.5.

In the E-Mode Channel Object, the Property *Adjustable Channel Object List* (see 2.3.4.10) shall specify the object indexes of the Adjusted E-Mode Channel Objects for every possible value of AJS.

2.5.1.2.2 Example

Table 22 gives an example of an Adjusted E-Mode Channel Object for AJS = 1. Please compare with Table 23 to see the difference in active GOs, names, Connection Codes, etc.

Table 22 – Adjusted E-Mode Channel Object for AJS = 5: PB Sunblind (example)

PID Value	PID Name	Property Name	Value (remark)
1	PID_OBJECT_TYPE	Interface Object Type	15 (Adjusted E-Mode Channel)
2	PID_OBJECT_NAME	Interface Object Name	50422053756E426C696E64h ("PB SunBlind")
61	PID_GO_CC_CODES_LIST	Connection Codes	[0001001300000000h, 000B001300000000h] (GO 1: CC_Switch_OnOff, + CC_Logical+ 0 + 0, GO 2: CC_Move_UpDown+CC_Logical+0+0)
62	PID_GO_CFLAGS_LIST	Connection Flags	[0002h, 0002h] (GO 1: {O}, GO 2: {O})
63	PID_OBJECTLINK	Object Link	PDT_FUNCTION
65	PID_GO_NAME_LIST	Group Object Names	[5374657053746F700000h, 5570446F776E00000000h] (GO 1: "StepStop", GO 2: "UpDown")
70	PID_PARAM_TYPES	Parameter Types	[0103h] (Param. 1: PART_UpDownAction
71	PID_PARAM_FLAGS	Parameter Flags	[0000h] (Param. 1: is not set locally, is not void and does not require a restart)
72	PID_PARAM_NAMES	Parameter Names	[504220616374696F6E00h] (Param. 1: "PB action")
73	PID_PARAM_UNITS	Parameter Units	[00000000000000000000h] (There are no Parameter Units.)
79	PID_PARAM_VALUES	Parameter Values	[1] (Param. 1: The user has changed the default value 0 = Up to 1 = Down.)

Table 23 – Adjusted E-Mode Channel Object for AJS = 8: Scene Number (example)

PID Value	PID Name	Property Name	Value (remark)
1	PID_OBJECT_TYPE	Interface Object Type	15 (Adjusted E-Mode Channel)
2	PID_OBJECT_NAME	Interface Object Name	5042205363656E65204E756D626572h ("PB Scene Number")
61	PID_GO_CCODES_LIST	Connection Codes	[0008000000000000h] (GO 8: CC_Scene_Number+0+0+0) (Only one GO is active. It has CC_Scene_Number, which is 0008h and no additional CCs.)
62	PID_GO_CFLAGS_LIST	Connection Flags	[0002h] (GO 8: 0)
63	PID_OBJECTLINK	Object Link	PDT_FUNCTION
65	PID_GO_NAME_LIST	Group Object Names	{5363656E654E724F7574h} (GO 8: "SceneNrOut")
70	PID_PARAM_TYPES	Parameter Types	[1101h] (Param. 1: PART_Scene_Number)
71	PID_PARAM_FLAGS	Parameter Flags	[0000h] (Param. 1: Parameter 1 is not set locally, is not void and does not require a restart.)
72	PID_PARAM_NAMES	Parameter Names	5363656E65204E720000h (Param. 1: "Scene Nr")
73	PID_PARAM_UNITS	Parameter Units	00000000000000000000h (There are no Parameter Units.)
79	PID_PARAM_VALUES	Parameter Values	[04] (Param. 1: The user has changed the default scene number value from 0 to 4; the PDT of this PID_PARAM_VALUES is PDT_GENERIC_01.)

2.5.2 Ctrl-Mode for Flexible E-Mode Channels

2.5.2.1 Introduction

2.5.2.1.1 Overview

The below specifications give the typical Configuration Procedures for the FEC Client to configure FEC devices. Deviations and optimizations are possible. In the normal case, the configuration of a FEC-device consists of the subsequent execution of the procedures described in the following clauses:

- Domain Address Assignment
- Individual Address Assignment
- Device Identification
- FEC Channel reading
- Localisation for Ctrl-Mode FEC devices
- Read the current links and Parameter Values
- Link FEC Channels
- Parameter setting

2.5.2.1.2 General remarks

In [07] it is specified for DMP_InterfaceObject_Read_RCI that the Property description shall be read for “unknown Properties”. The Property description shall in any case be read if it has not been read before for a given Property instance, if the Property specification allows for multiple Property Datatypes. In particular in this document, the Property description shall be read if the PDT can be PDT_REFERENCE.

This shall make sure that the Property value is properly interpreted and that a good error handling is possible.

EXAMPLE 21 This shall make sure that an 8 octet Property Value is not interpreted as a string if it is in fact a reference.

2.5.2.2 Domain Address Assignment

2.5.2.2.1 Finding a free RF Domain Address

The RF Domain Address needs to be unique. It can however not be checked with sufficient certainty that any RF Domain Address value would not be used in neighbouring RF Domains. Therefore, the unique KNX Serial Number of one of the KNX RF devices in the installation is taken as RF DoA. This device could in Ctrl-Mode be the Controller itself; as it should however be foreseen that this Controller can also be used to configure other installations, the Controller's KNX Serial Number should not be taken as DoA. Therefore, it is proposed the KNX Serial Number of the first programmed KNX RF device in the installation shall be used. THIS IS A PROPOSAL.

In order to use a unique value for the RF DoA, it is recommended that the KNX Serial Number of the device that is linked first in an installation be taken as DoA for that installation. Other solutions for choosing an RF DoA with sufficient guarantee of uniqueness are possible.

```
/* Read the DoA and the KNX Serial Number of one device */  
/* in which the Programming Mode is active. */  
NM_DomainAddress_Read(Device.SN = serial_number; Device.IA = individual_address,  
    Device.DoA = domain_address)
```

2.5.2.3 Domain Address assignment via Programming Mode

This procedure sets the RF DoA and the device's Individual Address; the device is identified by the user by activating the Programming Mode in the device.

```
/* The Domain Address "Device.DoA" is assigned to the device. In the same go, the Individual */  
/* Address Device.IAnew is assigned to the device. */  
NM_DomainAndIndividualAddress_Write2(NmpDoANew = Device.DoA, NmpIANew = Device.IAnew,  
    Device.IAold = NmpIACurrent)
```

2.5.2.4 Individual Address Assignment

The Individual Address can be assigned according the following procedures.

- Using Programming Mode
- Using the KNX Serial Number

The Network Configuration Procedures for FEC are identical to the existing Configuration Procedures for Ctrl-Mode as specified in clause 4.2.2.1 “Programming the Individual Address” in [07].

2.5.2.5 Device Identification

2.5.2.5.1 Overview

- ▢ *The existing Device Identification for Ctrl-Mode (see [07]) only consists of reading DD2. The interpretation of "Identification" is here broadened to be a general discovery.*

The Management Client shall now from the preceding Individual Address Assignment procedure use the Individual Addresses of the devices that it configures.

Optimisation

- The preceding Procedures for Individual Address Assignment already contain reading out the Individual Address of the device as a means to check the success of the IA assignment. This procedure of clause 2.5.2.5.2 can in that case be skipped and the discovery can directly start with the sequences of clause 2.5.2.6.

2.5.2.5.2 Read the Device Descriptor

```
/* Read the Device Descriptor Type 2 */  
DMP_Connect_RCI(nm_ASAP = Device.IA, nm_desc_type_req = DD2, nm_desc_type_res = DDresp,  
nm_desc_value_res = Device.DD)
```

Now, the Device Descriptor Type 2 shall be evaluated. The procedure only continues if the device is a FEC device.

```
/* If the responded Device Descriptor Type is not DD2 */  
/* then the device is a pure S-Mode device or unknown Profile. */  
if DDresp != DD2 then continue with the next device  
/* If the device is an E-Mode device, but no FEC device */  
/* then the further discovery can be skipped as well. */  
if DDres.ManagementProfile != 1000b then continue with the next device
```

An optimisation is possible here if another instance of the same E-Mode device has been read out before: if DD2 holds the same Manufacturer Code, Application Identification and Application Version as has been read out from a different device before, then the Management Client may stop the reading out of this device, use the data read out from the previously read device instance and continue with the next device.

The Management Client now knows that the device is a Flexible E-Mode device. It shall store the information of the Individual Address and the DD2-value.

If the device's KNX Serial Number is not yet known from the procedures in 2.5.2.4 "Individual Address Assignment" above, then it should be read.

```
/* Read PID_SERIAL_NUMBER of the Device Object */  
DMP_InterfaceObject_Read_RCI( /* [IN] */ object_index = 0; /* [IN] */ PID = PID_SERIAL_NUMBER,  
/* [OUT] */ Device.SerialNumber = data)
```

The obtained KNX Serial Number may be needed in the assignment of the Group Addresses (see 2.5.2.9) when the communication partner needs Extended Group Addresses.

2.5.2.6 FEC Channel reading

For every discovered FEC device FEC[n], its E-Mode Channel description shall be read in full.

◆ Discovery of the Object Index of E-Mode Device Object

To start with, the Object Index of the E-Mode Device Object shall be read. This Object Index is needed for the later localisation of the FEC-device.

 *The used NM_ObjectIndex_Read is specified in [15].*

```
FOR every FEC device FEC[n] discovered in the installation
    /* This reads the Object Index of the E-Mode Device Object in the device FEC[n]. */
    NM_ObjectIndex_Read(ASAP = FEC[n].IA,
        comm_mode_req = point-to-point connectionless, object_type = E-Mode Device Object,
        PID = PID_OBJECT_INDEX, test_info = 0101h, test_result = FEC[n].EDevObjectIndex,
        comm_mode_res = point-to-point connectionless)
NEXT FEC device
```

◆ Interface Object Index discovery

To start with, the Object index of the E-Mode Channel Object shall be read.

```
FOR every FEC device FEC[n] discovered in the installation
    /* This reads the IO-index of all Properties "E-Mode Channel Object" 3) 4) in the device FEC[n]. */
    /* It returns the array of E-Mode Channel Objects in the device: EObjectIndex[m]. */
    NM_ObjectIndex_Read(ASAP = FEC[n].IA, comm_mode_req = point-to-point connectionless,
        object_type = E-Mode Channel Object, PID = PID_OBJECT_INDEX,
        test_info = 0100h 5), test_result = FEC[n].EObjectIndex[m],
        comm_mode_res = point-to-point connectionless)
    /* The discovered Interface Object Index is used to immediately check whether the discovered */
    /* E-Mode Channel is an Adjusted E-Mode Channel or not. */
    FOR every E-Mode Channel Object EObject m discovered in the device n
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
            PID = PID_CHAN_FLAGS)
        IF PID_CHAN_FLAGS.A = 1 THEN
            /* One or more instances of the Adjusted E-Mode Channels are present in the device. */
            /* The IO-index of these Adjusted E-Mode Channel Objects is read 6). */
            DMP_InterfaceObject_Read_R(ASAP = FEC[n].IA,
                object_index = FEC[n].EObjectIndex[m], PID = PID_CHAN_ADJ_LISTS)
        END IF
    NEXT E-Mode Channel Object EObject m
NEXT FEC device
```

³⁾ The Interface Object Type for the E-Mode Channel has value 14.

⁴⁾ The specification of NM_InterfaceObject_Index_Read allows that the used A_NetworkParameter_Read be possibly repeated if more FEC Channels would be implemented than can be discovered in one A_NetworkParameter_Response-PDU.

⁵⁾ The number_of_instance equals 0: this is the "lazy client" approach. The device responds with as many instances as it can answer in one telegram.

⁶⁾ It can be noted that these Adjusted E-Mode Channel Objects are not read via NM_InterfaceObject_Index_Read, which would also return the Adjusted E-Mode Channel Objects of other E-Mode Channels in the device. Instead, PID_CHAN_ADJ_LISTS is read.

◆ Reading the descriptions of the Flexible E-Mode Channels

This procedure reads the description of the E-Mode Channel Object and for each also the possible Adjusted E-Mode Channel Objects. This description consists of the values of the Properties

- PID_CHAN_CODE
- PID_VERSION
- PID_CHAN_NUMBER
- PID_CHAN_FLAGS
- PID_CHAN_FB_LIST
- PID_GO_CCODICES_LIST
- PID_GO_CFLAGS_LIST
- PID_GO_SUBUNIT
(only in the E-Mode Channel Object, not in the Adjusted E-Mode Channel Object)
- PID_GO_NAME_LIST
- PID_PARAM_TYPES
- PID_PARAM_FLAGS
- PID_PARAM_NAMES
- PID_PARAM_UNITS

This procedure can be merged with the preceding discovery of the Interface Object Indexes.

```
FOR every FEC device FEC[n] discovered in the installation
  FOR every E-Mode Channel Object EObject m discovered in the device n
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_CHAN_CODE)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_VERSION)
    /* An optimisation is possible here 7). */
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_OBJECT_NAME)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_CHAN_NUMBER)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_CHAN_FLAGS)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_CHAN_FB_LIST)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_PARAM_TYPES)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_PARAM_FLAGS)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_PARAM_NAMES)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
      PID = PID_PARAM_UNITS)
```

⁷⁾ An optimisation is possible here: if the same Channel Code and Channel Version from the same manufacturer is read out as has already been read out from a (different) device before, then the Management Client may interrupt the FEC Channel Reading of this FEC Channel, use previously stored data instead and continue with the next E-Mode Channel. It may be noted however that the number and indexes of the Group Objects may differ with the value of the Adjustable Parameter, so these may be assumed identical only if also the value of the Adjustable Parameters equals the value read in another instance of the FEC Channel before.

```

/* If the E-Mode Channel is NOT an Adjusted E-Mode Channel, then all data related to GO is read. */
IF PID_CHAN_FLAGS.A = 0 THEN
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
        PID = PID_GO_CCODES_LIST)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
        PID = PID_GO_CFLAGS_LIST)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
        PID = PID_GO_SUBUNIT)
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
        PID = PID_GO_NAME_LIST)
/* If the E-Mode Channel is an Adjusted E-Mode Channel, then also every related Adjusted E-Mode */
/* Channel Objects l is read. */
ELSE THEN
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[l],
        PID = PID_CHAN_ADJ_LISTS)
    FOR every Adjusted E-Mode Channel Object l related to E-Mode Channel Object m
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_OBJECT_NAME)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_GO_CCODES_LIST)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_GO_CFLAGS_LIST)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_GO_NAME_LIST)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_PARAM_TYPES)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_PARAM_FLAGS)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_PARAM_NAMES)
        DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEOObjectIndex[l],
            PID = PID_PARAM_UNITS)
    NEXT Adjusted E-Mode Channel Object
END IF
NEXT E-Mode Channel Object
NEXT FEC device

```

The Controller now has for every FEC device the complete description of its E-Mode Channels and Adjusted E-Mode Channels.

2.5.2.7 Localisation for Ctrl-Mode FEC devices

2.5.2.7.1 Compatibility with existing Ctrl SEC

 *This clause is not intended for integration in the KNX Specifications. It should be clear from the Profile specifications that Ctrl FEC devices shall support these methods.*

Ctrl FEC devices shall not support Localisation via Localisation Channels.

Ctrl FEC devices may optionally support Localisation through the Localisation Flags L and LA.

Every Ctrl FEC device shall support at least one of the following localisation methods.

a. *Localisation via Programming Mode*

This is localisation at the level of the device, not at the level of the individual E-Mode Channel. Programming Mode is always mandatory, possibly not only for localisation, but also for discovering devices that do not have the medium dependent default IA.

b. *Localisation via the localisation flags L and LA in PID_GO_CFLAGS_LIST per FEC Channel.*

Both methods a. and b. shall be implemented in the Ctrl FEC device. The Ctrl FEC Controller shall support both methods as well.

Please refer to 2.9.4.1.

2.5.2.7.2 Ctrl FEC specific: Property based Localisation

The Ctrl FEC device shall additionally support Localisation through PID_LOCALISATION_REPORT or PID_LOCALISATION_COMMAND. At least one of these Properties shall be implemented.

NOTE 20 In a typical device, both Properties may be present.

It is possible that a FEC Channel supports both methods.

EXAMPLE 22 A relay output channel can be localised via PID_LOCALISATION_COMMAND; however, it may exhibit a local operation of the hardware switch on the device itself, which, when operated while *Localisation Mode* is active in the device, initiates a Localisation Report with PID_LOCALISATION_REPORT.

◆ Localisation with PID_LOCALISATION_COMMAND

This shall be initiated by the Controller.

1. The Controller shall firstly activate the *Localisation Mode* in all FEC-devices in the installation as specified in 2.3.7.4.2.
2. Then the controller may request the device localisation action through PID_LOCALISATION_COMMAND as specified in 2.3.7.6.3; or receive Localisation reports issued by the FEC Channels as specified in 2.3.7.5.2.
3. The Controller shall explicitly inactivate the *Localisation Mode* in all FEC-devices in the installation as specified in 2.3.7.4.2.

2.5.2.8 Read the current links and Parameter Values

NOTE 21 This reads all GAs and all Parameter values of the FEC device. The further procedures focus on a single FEC Channel only.

```
FOR every FEC device FEC[n] discovered in the installation
/* Read the GAs already assigned to that GO. */
count = 0; error = 0;
WHILE error <> FFh
    DMP_FunctionPropertyState_Read(dmp_iterator = count;
    KNX_SerialNumber[count] = dmp_SerialNumber; GA[count] = dmp_GroupAddress;
    GOindex[count] = dmp_GOindex);
    count++;
END WHILE
/* This procedure returns all the GAs of all the GOs. */
/* By the preceding device discovery, the Controller knows which GOs and thus which GAs */
/* are related to which E-Mode Channel in the device. */
;
```

```
/* Now, the the current Parameter Values are read. */
FOR every E-Mode Channel Object EObject m discovered in the device n
    /* The size of PID_PARAM_VALUES may vary. Therefore, in the following */
    /* DMP_InterfaceObjectRead_RCI, the included A_PropertyDescription_Read shall */
    /* effectively be called. */
    DMP_InterfaceObjectRead_RCI(object_index = FEC[n].EObjectIndex[m],
        PID = PID_PARAM_VALUES)
/* If the E-Mode Channel is an Adjusted E-Mode Channel, then also the Parameters
/* of every related Adjusted E-Mode Channel Objects are read. */
    IF PID_CHAN_FLAGS.A = 1 THEN
        FOR every Adjusted E-Mode Channel Object l related to E-Mode Channel Object m
            /* Here as well, the A_PropertyDescription_Read shall be called. */
            DMP_InterfaceObjectRead_RCI(object_index = FEC[n].AEObjectIndex[l],
                PID = PID_PARAM_VALUES)
        NEXT Adjusted E-Mode Channel Object
    END IF
```

2.5.2.9 Link FEC Channels

The installer procedures for the use of CTRL FEC devices can be identical to those of Ctrl SEC devices.

EXAMPLE 23 The presentation and the selection of the FEC Channels to link can be done by the Controller in the exact same way as for Ctrl SEC devices.

The Controller procedures can be identical to those of the Ctrl SEC. Specifically the Connection Rules are identical as in Ctrl SEC.

EXAMPLE 24 The Controller can use the same Management Procedures to find a free Group Address.

From the indications given by the installer about what FEC Channels shall be linked in this device, the Controller calculates the necessary links between GOs and chooses GAs to use for this. The Controller downloads the GAs in the FEC device. This is done GO per GO. For each GO the GAs are added one by one. (The used procedure and AL-services support linking only one GA at a time.)

NOTE 22 The below Configuration procedure links all GOs related to **one** FEC Channel. The procedures can be repeated and combined for further FEC Channels in the device. It is however not required and should not be assumed that all FEC Channels in a device will be linked consecutively. Oppositely, the Controller should neither skip any GO to be linked in a FEC Channel; it should only do "complete" links.

If the Controller concludes that the newly assigned GA shall be the sending GA for this GO, it shall set the sending flag accordingly in the below procedure.

To avoid any unwanted effects, the Controller shall hold the application prior to changing any links or Parameters.

```
/* Hold the Application Program. */
/* DMP_RunStateMachineWrite_R_IO always uses PID_RUN_STATE_CONTROL. */
/* If the Object Index of the Application Program Object is unknown, then it has to be discovered. */
/* This can be done by e.g. NM_ObjectIndex_Read as specified in [15]. */
DMP_RunStateMachineWrite_R_IO(object_index = ApplicationProgram.ObjectIndex,
    data = 020000000000000000000000h)
```

The Controller can now add the GAs for the FEC Channel.

PID_OBJECTLINK is actually inherited from the linking of RF bidirectional devices and extended. Note however that in the specifications in Chapter 3/5/3 "Configuration Procedures" ([07]) in clause 2.5.5 standard Property services are used (A_PropertyValue_Write in DMP_InterfaceObject_Write_R). Here, the Function Services (A_FunctionPropertyCommand in DM_FunctionProperty_Write_R) are used.

```
/* This procedure adds the GA(s) for one FEC Channel. */
For every GO to link in the FEC Channel
/* Now, the GAs can be assigned. The Group Object Index shall be the GO number within the E-Mode
Channel. */
/* The flag s ("Sending") has to be set only if the GA is the sending GA for the GO. */
For every GA to be assigned to the GO
    DM_FunctionProperty_Write_R(dmp_OI = FEC[n]. EObjectIndex[m], dmp_PID = PID-OBJECTLINK,
        dmp_command = "Add" + "Sending" + "KNX_Serial_Number" + GA + GO.number,
        Return Code = dmp_error)
next GA
next GO
```

Optimisation

If the Controller knows from 2.5.2.8 the GAs that are already assigned to this device, then it may skip in the above procedure the GAs that do not need to be changed and only handle the GAs that need to be added or removed. The Controller may for instance evaluate the device's Configuration Signature to conclude whether its stored information on the device's configuration still matches the device's real situation.

2.5.2.10 Parameter setting

It is recommended that firstly the Adjustable Parameter is set. This Parameter shall be of type PART_ADJUSTABLE_SELECTION and should be the first Parameter in the E-Mode Channel Object.

```
/* Set the Adjustable Parameter. */
DMP_InterfaceObjectWrite_RCI(object_index = FEC[n]. EObjectIndex[m],
    PID = PID_PARAM_VALUES, start_index = 1;
    element_count = 1; data = "value of Adjustable Parameter");
```

Based on the parameter description read in 2.5.2.6, the controller can download every separate parameter of a linked FEC Channel by simple writing the Property value in the E-Mode Channel Object. If the FEC Channel has been adjusted, the appropriate Property of the Adjusted E-Mode Channel Object is written instead.

```
Restart = False
IF FEC Channel is not adjusted
    FOR every parameter i of the FEC Channel
```

```
DMP_InterfaceObjectWrite_RCI(object_index = FEC[n].EObjectIndex[m],
    PID = PID_PARAM_VALUES, start_index = i;
    element_count = 1; data = parameter_value);
/* Keep track if a Parameter is changed that requires a restart. */
IF FEC[n].EObjectIndex[m].PID_PARAM_FLAGS.R = 1 THEN Restart = True
NEXT parameter
ELSE
/* The FEC Channel is adjusted. */
/* The Adjusted E-Mode Channel Interface Object is accessed now. */
FOR every parameter i of the Adjusted FEC Channel
    DMP_InterfaceObjectWrite_RCI(object_index = FEC[n].AEObjectIndex[i],
        PID = PID_PARAM_VALUES, start_index = i;
        element_count = 1; data = parameter_value);
/* Keep track if a Parameter is changed that requires a restart. */
IF FEC[n].AEObjectIndex[m].PID_PARAM_FLAGS.R = 1 THEN Restart = True
NEXT parameter
ENDIF
IF Restart = True THEN DM_Restart_RCI
```

If the Application Program is stopped prior to the linking or Parameter setting, then it shall be restarted again. This will also mark the point in time from which onward the application shall start using the new Parameter values.

```
/* Start the Application Program. */
DMP_RunStateMachineWrite_R_IO(object_index = ApplicationProgram.ObjectIndex,
    data = 010000000000000000000000h)
```

Optimisation

If the Controller knows the Parameter Values, then it may skip in the above procedure the Parameters that do not need to be changed and only handle the Parameters that do need to be changed. The Controller may for instance evaluate the device's Configuration Signature to conclude whether its stored information on the device's configuration still matches the device's real situation.

2.5.3 PB-Mode for Flexible E-Mode Channels

2.5.3.1 Introduction

 *The specification below follows the structure of the current PB-Mode as given in [07].*

Please observe the following differences between PB SEC and PB FEC.

- Unload of the Individual Address is not possible for PB SEC devices. For PB FEC devices the Unload of the Individual Address is mandatorily possible through the mandatory support of the Master Reset.
- For PB FEC devices a manual Master Reset is mandatory.

2.5.3.2 Domain Address Assignment

 *This clause is not considered for integration in the KNX Specifications.*

This document does not introduce any new procedures for this neither does it change any existing procedures. This is discussed in KSG (discussion topics [KSG00136] “DoA assignment on RF PB-Mode” and [KSG00142] “PB FEC”).

2.5.3.2.1 Assignment of the Individual Address through self acquisition by the device

 *This clause is not considered for integration in the KNX Specifications.*

There is no difference concerning the IA-assignment of PB SEC devices and PB FEC devices, for none of the KNX media.

2.5.3.2.2 Unload of the Individual Address in PB-Mode

 *In [07] clause 2.4.2 “Unload of Individual Address in PB-Mode” shall be replaced by the following.*

PB SEC devices have no requirements concerning the unloading of the IA by any means.

PB FEC devices are required to support the unloading of the IA through the mandatory support of the Master Reset.

2.5.3.2.3 Assignment of Group Addresses

 *This clause is not considered for integration in the KNX Specifications.*

There is no difference concerning the generation and testing of GAs between PB SEC devices and PB FEC devices, for none of the KNX media.

2.5.3.3 Link Procedure

The Configuration Procedures for PB FEC devices shall be identical to those of PB SEC devices. The presence of additional DPs will solely lead to the linking of more DPs than what would be the case with linking PB SEC devices.

In PID_CONFIG_LINK(Start_Link), the number of GOs to link is now no longer related to the Channel Code. Two E-Mode Channels with the same Channel Code may have a different number of GOs to link.

As specified in 2.3.4.17 for the specification of the field *Restart* in the *Property Parameter Flags*, as PB FEC device shall trace whether any Parameter is changed during the configuration, and if needed, autonomously conclude on an own device restart when the linking has completed.

2.5.3.4 Manual Master Reset

PB FEC devices shall foresee the possibility for a manual “reset to ex-factory state”. The way how this is done is implementation specific.

2.5.4 ETS access to FEC devices

 *This clause is not considered for integration in the KNX Specifications.*

The FEC requirements have been designed to allow for an implementation free of any legacy to existing S-Mode Profiles.

ETS shall manage a FEC device as Ctrl FEC Management Client as specified in 2.5.2. As the FEC Profiles require that a PB FEC device also implements the Ctrl FEC Profile, this shall allow ETS to also manage without change Ctrl FEC devices.

This clause is extended according the resolution to the comment "INSTA 1" from RfV.

Yet, **for a transition period**, starting when FEC obtains the status of Approved Standard until ETS supports FEC, it is foreseen that the FEC Profiles be combined with any S-Mode Profile: see 1.3.3.2. ETS shall in that case access these devices purely as S-Mode devices: ETS shall read DD0 and the device shall respond with a DD0-value of the chosen S-Mode Profile implementation, possibly with an ETS plug-in. This shall not lead to new S-Mode Profiles.

There shall be no dedicated standard - or non-standard Profile for the combination of FEC with any S-Mode Profile.

Later, when ETS supports FEC, ETS may still read DD0 but the device shall respond only with DD2 and ETS shall manage the device as a FEC device as indicated above, this is, according the Ctrl FEC procedures. There shall be no differences in the Management Client handling of a FEC device between S-Mode (ETS) and Ctrl FEC, this is, ETS shall act as a Ctrl FEC Management Client.

2.6 Interworking

The contents of this clause shall be integrated in the Chapter 3/7/1 "Interworking Model".

2.6.1 Guidelines for the design of Adjustable E-Mode Channels

1. AJS shall be the first parameter.

The other Parameters are accessed independently of the value of AJS (on same Properties or on same memory locations). This means that if AJS is changed, the other Parameters have to be reset to their corresponding default value (corresponding to the AJS value), by the Management Client (in case of Ctrl-Mode) or by the Management Server (device itself) in PB-Mode.

NOTE 23 This may happen if the functionality of the device is selected locally on the device. A Management Client (Controller) may prevent from "half links" by clearing firstly all links before re-using an E-Mode Channel.

2. The Connection Code should not be changed; only the visibility.

This gave problems in the past when links were removed.

2.7 Property Datatypes

The contents of this clause shall be integrated in [03].

2.7.1 Overview

*The row shall **NOT** be added to Table 6 "Overview Property Datatypes".*

Property Datatype	Property Datatype	Format	Type-Length [octets]	Level
25h	PDT_REFERENCE	Variable format, typed.	8	3

2.7.2 Detailed specification

*This definition shall **NOT** be inserted appropriately in clause 4.2 of [03]. <This clause is maintained to keep the document understandable. It is only informative, since the final and correct specification of PDT_REFERENCE is done in [22].*

25h PDT_REFERENCE

This PDT shall be used if the related Property Value does not contain any actual Property data, but is instead only a reference to a location where the real data shall be read.

The Property of the type PDT_REFERENCE is below named the “referring Property”.

Format

The format of the Property Value according PDT_REFERENCE shall be variable. It shall be typed in the field *Type*. The length shall be 8 octets.

The data in this Property doesn't indicate which Application Layer services are supported by the MaS for accessing Properties.

Reference Type 0

This reference shall be a reference to a Property Value addressed by Object Type and Object Instance and Object Index and PID and Start Index.

The fields Object Type, Object Instance and Object Index, PID and Start Index shall respectively point to the Object Type, Object Instance, Object Index, Property and Start Index of the Property at which the Value of the referring Property shall be read.

It shall be possible that a PDT_REFERENCE value refers to more than one variable.

EXAMPLE 25 This allows for the easy referencing of multiple strings of equal length, or to values of enumerated Parameter Types, etc.

The field Subtype shall indicate if and where next values can be accessed.

Subtype (4 bit)	Description
0h	No further elements follow.
1h	The next values, if available, shall be accessed <ul style="list-style-type: none"> - in the same Property - in the same array element as is referred by this reference. The values are separated by a NULL-character (00h).
2h	The next values, if available, shall be accessed <ul style="list-style-type: none"> - in the same Property - in the same array element as is referred by this reference. The values have a fixed length.
3h	The next values, if available, shall be accessed <ul style="list-style-type: none"> - in the same Property - in the next array elements as is referred by this reference.
4h	The next values, if available, shall be accessed in the Property Values with next subsequent PID-values as referred by this reference.
5h to Fh	These subtypes are reserved for future use. They shall not be used by a Management Server. A Management Client shall use such value of PDT_REFERENCE solely to find the first value.

Reference Type 1

This reference shall be a reference to a memory location expressed as a four octet memory address.

The field Memory Address shall point to the absolute memory address at which the value shall be read.

For this Reference Type 1, the Subtype shall always be 0h.

Other Reference Types

Other types of PDT_REFERENCE are reserved for future definition by KNX Association and shall not be used.

Alternative type (for PDT_REFERENCE)

It is the intention that the Property Client (Management Client) detects through this PDT that the data read as Property Value is not the real data but only a reference. Therefore, for this PDT there cannot be an alternative PDT.

None.

2.8 Usage and context

This clause is not intended for integration in the KNX Specifications.

The usage and context are clear from the introduction (see clause 1) and the Configuration Procedures (see 2.5).

2.9 Profile definition

This clause shall be integrated in Volume 6 "Profiles" in clause 5" as new Profile. Blue text is copied from the existing Profiles specifications.

This document specifies the following two Profiles

1. Ctrl FEC, and
2. PB FEC.

The indication "FEC General" is not a Device Profile. It is a help for reading this document, specifying the minimal requirements for any FEC implementation, but is as such not sufficient to build a KNX device upon and not allowed as reference for an implementation.

- The existing Ctrl-Mode Profiles - Ctrl-Mode fixed DMA and Ctrl-Mode reloc DMA (collectively referred to as Ctrl SEC) and
- the existing PB-Mode Profiles (referred to as PB SEC)

are only given in this document as a reference, to enable the reader to see the difference between these existing E-Mode flavours and E-Mode devices with FEC. It is not the intention to modify these Profiles.

Further on, for easy reading of this document, the detailed requirements of the features have been removed as far as they are identical to the Feature definitions in [04].

In the below, please mind the indication that the existing Profiles

- **Ctrl-Mode fixed DMA**
- **Ctrl-Mode reloc DMA and**
- **PB-Mode**

are only given in this document as a reference, to enable the reader to see the difference between these existing E-Mode flavours and E-Mode devices with FEC. It is not the intention to modify these Profiles. These indications for these Profiles are not part of the KTB voting of this document.

If these Profiles were to be phased out, this will be subject to a different document!

2.9.1 Communication

Feature	Ctrl-Mode fixed DMA	Ctrl-Mode reloc DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1 TL - broadcast	M	M	M	M	M	M
2 TL - connection oriented	M	M	-	O	O	O
3 TL - connectionless	-	-	M	M	M	M
4 TL - Extended	-	-	-	O	O	O

2.9.2 Device Management

In this clause all general requirements on a device concerning the mechanisms used for access by the management client are described.

- ▢ *Memory mapped Resources are no longer required for FEC devices.*
- ▢ *In the current KNX Specifications "Profiles", Load- and Run State Machines are one feature. These are here separated, so that it is possible to specify that a device shall have Run State Machines without having Load State Machines.*
- This Part of this document DOES have the intention to modify the existing Feature list and definitions in [04].*

Feature	Ctrl-Mode Fixed DMA	Ctrl-Mode reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1. Direct Memory Access	M	M	-	O	O	O
2. Verify Mode	-	-	-	O	O	O
3. Interface Object Handling ⁸⁾	-	C*	-	M	M	M
4. Reduced Interface Objects	-	C*	-	X	X	X
5. Load and Run State Machines						
5.1 Load State Machines	-	M	-	O	O	O
5.2 Run State Machines	-	M	-	M	M	M
6. Hardware Specific Parameters	-	-	-	O	O	O
7. RAM (cleared)	-	-	-	O	O	O
8. User EEPROM	M	M	-	O	O	O
9. Restart						
a. connectionless	O	O	O	M	M	M
b. connection-oriented	M	M	-	O	O	O
d. Master Reset	n/a ⁹⁾	n/a	n/a	M	M	M
e. Manual reset to ex-factory	O	O	O	O	O	M
10. Authorization	-	-	-	O	O	O
nr of access levels	-	-	-	4	4	4

* conditional, one choice per column mandatory.

⁸⁾ Please refer to Annex A for the specification of mandatory and optional Interface Objects, Properties and Property fields.

⁹⁾ "n/a as the feature" "Master Reset" was not yet defined at the time this Profile was defined.

- ☞ *FEC does not require "Memory Mapped Resources". However, in KNX in the Profiles, we have the approach that features that are not "Mandatory" are rather "Optional" than "Not allowed". Therefore, in the above, "Memory Mapped Resources" are "Optional" instead of "Not allowed". If it would be wanted that they are really forbidden, then this should be changed. Obviously, during the E-Mode configuration no memory addressing should happen.*
- During ETS access to the device to a "necessary intermediate", memory mapped access can be possible; the FEC support of ETS SHALL NOT use any memory mapped access.*

2.9.2.1 Run State Machine

a) Realisation Type 1 - Property based		
Specification	Test	
[01] - records		
[02] - "DMP_RunStateMachineWrite_R_IO" - §3.31.3 "DMP_RunStateMachineVerify_R_IO" - §3.32.3 "DMP_RunStateMachineRead_R_IO"	[13]	- §2 "Network Management Tests" corresponding tests

2.9.2.2 Master Reset

- ☞ *This clause shall be added to [04].*

Specification (Volume 3)	Test (Volume 8)
[01] - clause 4.8 "Group Address Table" – "default state" - clause 4.9 "Group Object Association Table" – "default state" - application reset to default application - application parameters reset to their default value	To be completed. To be completed. To be completed. To be completed.
[02] - clause 3.7.2 "DM_Restart_RCI" – master reset - clause 3.7.3 "DM_Restart_RCo" – master reset	To be completed. To be completed.

2.9.2.3 Manual reset to ex-factory

Specification	Test
[01] - The device shall exhibit an implementation specific means in its HMI to cause an ex-factory reset.	None.

2.9.3 Device Identification

Feature	Ctrl-Mode Fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1 Device Descriptor Service – connection oriented	M	M	O	O ¹⁰⁾	O ¹⁰⁾	O ¹⁰⁾
2 Device Descriptor Service – connectionless	O	O	M	M	M	M
3 Device Descriptor Type 0	M	M	O	O ¹¹⁾	O ¹¹⁾	O ¹¹⁾
4 Device Descriptor Type 2	M	M	M	M	M	M
5 Device Descriptor InfoReport	M ¹²⁾	M ¹²⁾	M ¹²⁾	C ¹³⁾	C ¹³⁾	C ¹³⁾

2.9.4 Device Individualisation

2.9.4.1 Overview

In this clause all requirements on a device for device individualisation and assignment of the Individual Address are described.

	Ctrl-Mode Fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl- FEC	PB- FEC
Programming Mode control						
1 Realisation Type 1 (Property)	O	O	O	M	M	M
2 Realisation Type 2 (memory mapped)	C ^{h)}	C ^{h)}	C	O	O	O
DoA assignment						
3 DoA assignment via Programming Mode – Type 0	C ^{d)}	C ^{d)}	C ^{d)}	M	M	M
4 DoA assignment via Programming Mode – Type 1	O	O	O	M	M	M

¹⁰⁾ The A_DeviceDescriptor_Read-service shall be supported in point-to-point connectionless communication mode. If however the optional connection oriented Transport Layer is also supported, then the A_DeviceDescriptor_Read-service shall also be supported on point-to-point connection-oriented communication mode. See as well footnote in [04].

¹¹⁾ This Profile does not require a DD0.
The combination of this Profile with another Profile should be identified by a proper DD-value (DD0 and/or DD2).
Devices that implement this Profile and another Profile, may have a DDO.

¹²⁾ Mandatory for RF implementations.

¹³⁾ Device Descriptor InfoReport is mandatory for unidirectional RF devices.

¹³⁾ Device Descriptor InfoReport is mandatory for unidirectional RF devices.

	Ctrl-Mode Fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl- FEC	PB- FEC
5 DoA assignment via KNX Serial Number IA assignment	O	O	O	M	M	M
6 SNA	O	O	O	O	O	O
7 IA assignment via Programming Mode	C c ^{a)}	C c ^{a)}	C a ^{a)}	M	M	M
8 IA assignment via KNX Serial Number	C c ^{a)}	C c ^{a)}	O	M	M	M
9 IA Local Assignment	O	O	O	X	X	X
10 Default IA	O	O	O	M	M	M
11 Distributed Address Assignment (DAA)	O	O	C b ^{a)}	O	O	M
Localisation						
12 Localisation via Programming Mode	C e ^{a)}	C e ^{a)}	C e ^{a)}	M	M	M
13 Localisation via localisation channel	C e ^{a)}	C e ^{a)}	C e ^{a)}	X	X	X
14 Localisation via localisation flag L	C e ^{a)}	C e ^{a)}	C e ^{a)}	O	O	O
15 Localisation via Localisation Flag LA	O	O	O	O	O	O
16 Localisation via Localisation Properties	O	O	O	M	M	M
Device scan						
17 Serial Number read by Programming Mode	O	O	O	M	M	M
18 Serial Number read by Ex Factory State	O	O	O	M	M	M
19 Serial Number read by Power Reset	O	O	O	M	M	M
<p>a Mandatory for bidirectional RF devices.</p> <p>b Not mandatory for bidirectional RF devices.</p> <p>c It is mandatory to implement at least one of the features</p> <ul style="list-style-type: none"> - "Programming Mode", or - "KNX Serial Number" or - "Local Assignment". <p>If the Property "KNX Serial Number" (PID_SERIAL_NUMBER = 11) in the Device Object is implemented then the AL-services A_IndividualAddressSerialNumber_Read (in full, this is A_IndividualAddressSerialNumber_Read-PDU as well as A_IndividualAddressSerialNumber_Response-PDU) and A_IndividualAddressSerialNumber_Write shall be supported too.</p> <p>d Mandatory on open media.</p> <p>e It is mandatory to support at least one of the features</p> <ul style="list-style-type: none"> - "Localisation via Programming Mode", or - "Localisation via localisation Channel" or - "Localisation via Localisation Flag L". <p>f At least one of these three methods shall be applied.</p> <p>g At least one of these two methods shall be applied.</p> <p>h If Programming Mode is required for the assignment of DoA or IA for localisation, then the Programming Mode Control (activation, deactivation) shall be realised in this way.</p>						

	Ctrl-Mode Fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl- FEC	PB- FEC
<p>i Next to the mandatory support of Localisation via Programming Mode, it is mandatory to support at least one of the following</p> <ul style="list-style-type: none"> - "Localisation via localisation Channel" or - "Localisation via Localisation Flag L" and "Localisation via Localisation Flag LA" 						

2.9.4.2 DoA assignment

2.9.4.2.1 DoA assignment via Programming Mode – Type 0

Specification	Test
[02] - §2.7 "NM_DomainAddress_Read" - §2.9 "NM_DomainAndIndividualAddress_Write" - §2.12 "NM_DomainAddress_Scan"	[13]

ⓘ About the above used NM_DomainAddress_Scan, please note that in [19] it will be specified that that procedure shall only use the A_DomainAddressSelective_Read with the field Type set to 00h.

2.9.4.2.2 DoA assignment via Programming Mode – Type 1

ⓘ This new Feature "DoA assignment via Programming Mode – Type 1" needed to be introduced to indicate that only FEC devices shall support the redefined A_DomainAddressSelective_Read with the Type 1.

Specification	Test
[02] - §2.7 "NM_DomainAddress_Read" - 2.9 "NM_DomainAndIndividualAddress_Write" - §2.12 "NM_DomainAddress_Scan" [19] - §2.3.4 "NM_DomainAddress_Scan2"	[13]

2.9.4.2.3 DoA assignment via KNX Serial Number

Specification	Test
[02] - §2.11 "NM_DomainAnd-IndividualAddress_Write3"	[13]

2.9.4.3 Localisation via Localisation Properties

 This clause shall be added to [04].

Specification (Volume 3)	Test (Volume 8)
E-Mode Device Object <ul style="list-style-type: none"> - PID_LOCALISATION_MODE - PID_LOCALISATION_REPORT - PID_LOCALISATION_COMMA ND 	To be completed.

2.9.4.4 Device Scan

Specification	Test
[21] §2.4.2.4 "NM_Read_SerialNumber_By_Programming Mode" §2.4.2.5 "NM_Read_SerialNumber_By_ExFactoryStat e" §2.4.2.6 "NM_Read_SerialNumber_By_PowerReset	To be completed.

2.9.5 Programming Mode control

2.9.5.1 Realisation Type 1 (Property)

Specification	Test
[01] - §4.3.5 "PID_PROGMODE"	

2.9.5.2 Realisation Type 2 (memory mapped)

Specification	Test
[01] - §4.1.7.3 "Programming Mode – Realisation Type 2"	


2.9.6 Device Linking

In this clause all requirements on a device for linking of group objects are described. This includes the configuration of address and association tables. Note that in this part of Volume 6 only the Management Server part is described.

		Ctrl-Mode Fixed DMA	Easy Ctrl reloc DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1	Address Table fixed	M	X	-	O	O	O
2	Address table relocatable	X	M	-	O	O	O
3	Association Table	M	X	-	O	O	O
4	Assoc table relocatable	X	M	-	O	O	O
5	Group Object Table Relocatable	X	C ^a		O	O	O
6	Routing Table	-	-	-	-	-	-
7	Link management services	O	O	-	X	X	X
8	GA Check	M	M	-	O	O	M
9	Direct Link	-	-	M	O	O	M
10	Address table Property based	-	-	-	O	O	O
11	Association table Property based	-	-	-	O	O	O
12	Logical Tag extended linking	-	-	-	O	O	O
13	Linking via KNX Serial Number				O	O	O
14	Linking via Domain Address				O	O	O
15	Linking via LTE logical tags local assignment				O	O	O
16	Linking via LTE logical tags remote assignment				O	O	O
17	Property based KNX Serial NumberTable				O	O	O
18	Distribution of KNX Serial NumberTable				O	O	O
a Mandatory for implementations based on mask 0701h.							

2.9.7 Application handling

	Ctrl-Mode Fixed DMA	Easy Ctrl reloc DMA	PB-Mode	RF unidirectional sender	FEC General	Ctrl FEC	PB FEC
1 Group Object Table	O	O	O		O	O	O
2 Application Program and Parameters	O	O	O		O	O	O
3 Application Specific Parameters	O	O	O		O	O	O
4 Application Programming Interface (API)	O	O	O		O	O	O
5 Functional Parameters general ^{b)}	M	M	M		O	O	O
6 Functional Parameters set locally ^{b)}	O ^{a)}	O ^{a)}	O		O	O	O
7 Functional Parameters fixed DMA ^{b)}	M ^{a)}	O	O		O	O	O
8 Functional Parameters reloc DMA ^{b)}	O	M ^{a)}	O		O	O	O
9 Functional Parameters red. IO ^{b)}	O ^{a)}	O ^{a)}	O		O	O	O
10 Functional Parameters via direct link ^{b)}	O	O	M ^{a) c)}		O	O	O
11 Functional Parameters via Properties	O	O	O	O	O	O	O
12 Functional Parameters in the E-Mode Channel Objects	X	X	X	X	C ^{d)}	C ^{d)}	C ^{d)}
12 Property based Group Object Table	O	O	O		O	O	O
^{a)} If set locally, there shall be read access to these parameters. A write access may be rejected. ^{b)} Features 5 to 10 are only mandatory, if a device implements E-Mode Channels with functional parameters defined. ^{c)} Unidirectional devices compliant with the PB-Mode <i>may</i> send the current state of parameters within the link procedure as described in §5.4.3.9 "Link Procedure for unidirectional devices" in [07]. ^{d)} Functional Parameters shall be implemented as Properties in the E-Mode Channel Object if a device implements at least one E-Mode Channel with functional parameters defined. For most implemented devices, this feature will thus be mandatory.							

 The feature "Functional Parameters via Properties" does not mean the Property based implementation of E-Mode Channel Parameters in an array in the E-Mode Channel Object. Instead, it refers to the System 300 way of having application specific Parameters in application specific Interface Objects (Functional Blocks).

Features 5-10 are only mandatory, if a device implements channels with functional parameters defined.
 local setting only.

+ When locally set, the parameters must be readable. A write to the parameters can be rejected.

2.10 Profile definition - Interface Objects and Properties

 This clause shall extend Annex A in [04].

2.10.1 Interface Objects

 E-Mode Profiles → Interface Objects

Interface Object	Easy Ctrl. fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
0 Device Object	O	M	M	M	M	M
1 Addressable Object	O	M	O	O	O	O
2 Associationtable Object	O	M	O	O	O	O
3 Applicationprogram Object	O	M	O	M	M	M
14 E-Mode Channel Object	O	O	O	M	M	M
15 Adjusted E-Mode Channel	O	O	O	C ^a	C ^a	C ^a
16 Text Catalogue Object	O	O	O	C ^b	C ^b	C ^b
18 E-Mode Device Object	O	O	O	M	M	M
^a One instance of the Adjusted E-Mode Channel shall be implemented for each possible value of the Adjustable Parameter (AJS) in each instance of the E-Mode Channel Object. ^b This Text Catalogue Object is mandatory if the device allows selection of the language of textual information.						

2.10.2 Device Object

 E-Mode Profiles → Device Object

Property		Easy Ctrl. fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/x	3/x	3/x	3/x	3/x	3/x
11 PID_SERIAL_NUMBER	Data	(3/x)	(3/x)	(3/x)	3/x	3/x	3/x
12 PID_MANUFACTURER_ID	Data	(3/x)	(3/x)	(3/x)	3/x	3/x	3/x
15 PID_ORDER_INFO	Data	(3/x)	(3/x)	(3/x)	3/1	3/1	3/1
21 PID_DESCRIPTION	Data	(3/3)	(3/3)	(3/3)	3/3	3/3	3/3
30 PID_DOWNLOAD_COUNTER	Data	(3/1)	(3/1)	(3/1)	3/1	3/1	3/1
54 PID_PROGMode	Data	(3/3)	(3/3)	(3/3)	3/3	3/3	3/3
56 PID_MAX_APDU_LENGTH	Data	3/x	3/x	3/x	3/x	3/x	3/x
63 PID_OBJECTLINK	Function				X	X	X
101 PID_CHANNEL_01_PARAM	Data	(3/3)	(3/3)	(3/3)	X	X	X
... ..	Data	(3/3)	(3/3)	(3/3)	X	X	X

Property		Easy Ctrl. fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
132 PID_CHANNEL_32_PARAM	Data	(3/3)	(3/3)	(3/3)	X	X	X

2.10.3 Applicationprogram Object

[E-Mode Profiles → Applicationprogram Object](#)

Property		Easy Ctrl. fixed DMA	Easy Ctrl. reloc. DMA	PB-Mode	FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/x ¹⁴	3/x	(3/x) ¹⁴	3/X	3/X	3/X
5 PID_LOAD_STATE_CONTROL	Data	(3/3)	3/3	(3/3)	(3/3)	(3/3)	(3/3)
6 PID_RUN_STATE_CONTROL	Data	(3/3)	3/3	(3/3)	3/3	3/3	3/3
7 PID_TABLE_REFERENCE	Data	(3/3)	3/3	(3/3)	(3/3)	(3/3)	(3/3)
51 PID_PARAM_REFERENCE	Data	(3/3)	C ¹⁴	(3/3)	(3/3)	(3/3)	(3/3)

2.10.4 E-Mode Channel Object

[E-Mode Profiles → E-Mode Channel Object](#)

Property		FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/1	3/1	3/1
2 PID_OBJECT_NAME[]	Data	(3/1)	(3/1)	(3/1)
21 PID_DESCRIPTION[]	Data	3/3	3/3	3/3
25 PID_VERSION	Data	3/1	3/1	3/1
51 PID_CHAN_NUMBER	Data	3/1	3/1	3/1
52 PID_CHAN_CODE	Data	3/1	3/1	3/1
53 PID_CHAN_FLAGS	Data	3/1	3/1	3/1
54 PID_CHAN_FB_LIST[]	Data	(3/1)	(3/1)	(3/1)

¹⁴⁾ Only mandatory for system 7 Easy Controller implementations

Property		FEC General	Ctrl FEC	PB FEC
55 PID_CHAN_ADJ_LISTS	Data	(3/1) ¹⁵⁾	(3/1) ¹⁵⁾	(3/1) ¹⁵⁾
61 PID_GO_CCODES_LIST	Data	3/1	3/1	3/1
62 PID_GO_CFLAGS_LIST	Data	3/1	3/1	3/1
63 PID_OBJECTLINK	Function	M	M	M
64 PID_GO_SUBUNIT	Data	(3/1)	(3/1)	(3/1)
65 PID_GO_NAME_LIST	Data	(3/1)	(3/1)	(3/1)
70 PID_PARAM_TYPES	Data	3/1	3/1	3/1
71 PID_PARAM_FLAGS	Data	3/1	3/1	3/1
72 PID_PARAM_NAMES	Data	(3/1)	(3/1)	(3/1)
73 PID_PARAM_UNITS	Data	(3/1)	(3/1)	(3/1)
79 PID_PARAM_VALUES	Data	3/1	3/1	3/1

2.10.4.1 PID_CHAN_FLAGS (PID = 53)

Field	Ctrl FEC	PB FEC
A Adjustable	M	M
I Input	M	M
O Output	M	M
LC Localisation Command	C a	C a
LR Localisation report	C a	C a
^a At least one of the fields <i>Localisation Command</i> or <i>Localisation Report</i> shall be supported for each E-Mode Channel. The field <i>Localisation Command</i> is typical for an actuator channel; the field <i>Localisation Report</i> is typical for a sensor channel.		

¹⁵⁾ PID_CHAN_ADJ_LISTS shall be implemented if the E-Mode Channel is adjustable.

2.10.5 Adjusted E-Mode Channel Object

[E-Mode Profiles](#) → [Adjusted E-Mode Channel Object](#)

Property		FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/1	3/1	3/1
2 PID_OBJECT_NAME[]	Data	(3/1)	(3/1)	(3/1)
61 PID_GO_CCODES_LIST	Data	3/1	3/1	3/1
62 PID_GO_CFLAGS_LIST	Data	3/1	3/1	3/1
63 PID_OBJECTLINK	Function	M	M	M
65 PID_GO_NAME_LIST	Data	(3/1)	(3/1)	(3/1)
70 PID_PARAM_TYPES	Data	3/1	3/1	3/1
71 PID_PARAM_FLAGS	Data	3/1	3/1	3/1
72 PID_PARAM_NAMES	Data	(3/1)	(3/1)	(3/1)
73 PID_PARAM_UNITS	Data	(3/1)	(3/1)	(3/1)
79 PID_PARAM_VALUES	Data	3/1	3/1	3/1

2.10.6 Text Catalogue Object

[E-Mode Profiles](#) → [Text Catalogue Object](#)

Property		FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/1	3/1	3/1
25 PID_VERSION	Data	3/1	3/1	3/1
50 PID_LOCALE_LIST	Data	(3/1)	(3/1)	(3/1)
51 PID_LOCALE_SELECTION	Data	(3/3)	(3/3)	(3/3)
52 PID_ACTIVE_LOCALE	Data	3/x	3/x	3/x
61 PID_STRING_001	Data	3/1	3/1	3/1
...			
141 PID_STRING_141	Data	3/1	3/1	3/1

^a If the Text Catalogue Object is implemented, then there shall at least be the first string Property PID_STRING_001; further text shall be stored in subsequent Properties PID_STRING_002 up to PID_STRING_141, as far as needed and without gaps.

2.10.7 E-Mode Device Object

 *E-Mode Profiles → E-Mode Device Object*

Property		FEC General	Ctrl FEC	PB FEC
1 PID_OBJECT_TYPE	Data	3/1	3/1	3/1
60 PID_LOCALISATION_MODE	Data	x	x	x
	NwPar	W	W	W
61 PID_LOCALISATION_REPORT	Data	x	x	x
	NwPar	I	I	I
62 PID_LOCALISATION_COMMAND	Data	x	x	x
	NwPar	W	W	W

 *This new legend symbol has to be added to [04] clause A.1.2.*

I: It is mandatory to support this Property via A_NetworkParameter_InfoReport.

2.10.8 Identifiers and discovery

 *This clause is not intended for integration in the KNX Specifications.*

Any E-Mode device shall exclusively be identified as FEC device by the value 1000b of the field *Management Profile* in the DD2 of the device, either as Ctrl FEC- or as PB FEC device.

For Ctrl-Mode, the further discovery shall be based on the reading of the Properties of the E-Mode Device Object and the various E-Mode Channel Objects and possible Adjusted E-Mode Channel Objects as specified in 2.5.2.6.

As the Profile PB FEC embeds the Profile Ctrl FEC in full, the discovery of PB FEC devices can be the same as for Ctrl FEC devices.

3 Impact and dependencies

3.1 System specification (“Handbook”) dependencies

 *This clause is not intended for integration in the KNX Specifications.*

The various clauses of this document have to be integrated in the KNX Specifications as marked above in the clauses themselves.

3.2 Configuration interworking

By the compatible reformulation of DD2 and by the requirement that an Extended E-Mode Channel shall base on a Basic E-Mode Channel, compatibility of Ctrl FEC devices with existing Controllers and existing Ctrl SEC devices is given. Obviously, a new Ctrl FEC device may base on a new Basic E-Mode Channel that is unknown to a Ctrl SEC Controller. This is however a versioning issue of the Ctrl SEC Controller and is not specific for new Basic E-Mode Channels implemented in FEC devices. The problem of handling unknown Basic E-mode Channels can however better be solved by the FEC solution, amongst other through the extended description of Parameters.

The FEC specifications in this document are not assumed to put requirements on interfaces and Couplers.

For PB FEC, compatibility with PB SEC is given, as the PB FEC device will firstly have to check whether or not its communication partner supports FEC.

3.3 Run-time Interworking

 *This clause shall be integrated in Chapter 371 "Interworking Model" ([08]).*

3.3.1 Concerning Basic E-Mode Channels

3.3.1.1 Design directives

New Basic E-Mode Channels definitions should only contain a limited set of "mainstream" Datapoints ¹⁶⁾ to implement a basic functionality. A new Basic E-Mode Channel definition shall only be accepted if no existing Basic E-Mode Channel with any of the flexibility of its Extended E-Mode Channels can suit the needs.

An "empty" Basic E-Mode Channel, of which the implementation would thus only exist of implementation of optional features, is not foreseen.

The DPs of the Basic E-Mode Channel should be the ones that guarantee and realize the functionality of the distributed application. For instance, the DPs that are mandatory in the composing FBs should be mandatory in the basic Channel.

3.3.1.2 Standardisation

Any Basic E-Mode Channel and any Extended E-Mode Channel shall be approved by WGI.

3.3.2 Concerning Extended E-Mode Channels

The combination of optional features will not be laid down explicitly in the KNX Specifications.

3.3.2.1 Design directives

The Extended E-Mode Channel shall contain all features and DPs of the Basic E-Mode Channel on which it bases.

The extensions of the Basic E-Mode Channels shall be based on FBs. A FB can be added to an E-Mode Channel if it is part of the same Application Domain and if it is a normal communication partner of a FB that is already part of a FB in the E-Mode Channel (Basic - or already existing Extended E-Mode Channel). This is, the Extended E-Mode Channel shall again be a clear cut within the application model. If this is not the case, WGI shall be contacted. The set of Datapoints (Group Objects, Parameters...) to be used from the added Functional Block is manufacturer specific. All Connection Codes and all Parameter Types used within an E-Mode Channel have to be unique within the E-Mode Channel.

3.3.2.2 Implementation

For the implementation of any Extended E-Mode Channel, the mandatory Datapoints (Group Objects, Parameters...) shall be implemented as given in the Extended E-Mode Channel definition; the implementation of the optional features in the Extended E-Mode Channel may be subject to conditions that may be specified in the Extended E-Mode Channel definition.

NOTE 24 At the time of publication of this document, no Extended E-Mode Channel definitions have been drafted.

¹⁶⁾ A Datapoint can be an Input, an Output, a Parameter or Diagnostic Data. Please refer to [08] for the definition and further details.

3.4 Integration and common tool impact

The integration in ETS is described in clause 2.5.4.

For the integration in ETS and for the description of FEC functionality in XML schema, KSG shall be consulted.

The FEC self descriptive features exhibit interesting functionality that may be interesting also for S-Mode devices. This is assumed to relate to the ETS discussion topic of online discovery of the devices, as for instance discussed in TF Device Management.

3.5 Risks and compatibility issues

3.5.1 Definition and use of two octet Connection Codes

The definition of Connection Codes is co-ordinated by KNX Association's Working Group Interworking. So far (February 2010) no Connection Codes have been defined with numerical value higher than 255.

The current specification of PID_PB_CONFIG (see [01]) does not allow for the exchange of two octet Connection Codes. This adaptation is not foreseen in this Application Note.

Annex A

(informative)

Overview existing Parameter Types and Identification

A.1 Parameter Type coding

Table 24 – Standard Parameter Types

PART_Name	Parameter Size	Parameters Format	Standard Parameter Type	DPT_ID	DPT_Name
PART_Switch_Value	1 bit	B ₁	0101h	1.001	DPT_Switch
PART_Boolean	1 bit	B ₁	0102h	1.002	DPT_Bool
PART_UpDown_Action	1 bit	B ₁	0103h	1.008	DPT_UpDown
PART_Invert	1 bit	B ₁	0104h	1.012	DPT_Invert
PART_Logical	1 bit	B ₁	0105h	1.021	DPT_LogicalFunction
PART_Scene_Value	1 bit	B ₁	0106h	1.022	DPT_Scene_AB
PART_Blind_Mode	1 bit	B ₁	0107h	1.023	DPT_ShutterBlinds_Mode
PART_Scene_Number	6 bit	U ₆	1101h	17.001	DPT_SceneNumber
PART_Date_Time	8 octets	U ₈ [r ₄ U ₄][r ₃ U ₅][U ₃ U ₅][r ₂ U ₆][r ₂ U ₆]B ₁₆	1301h	<u>19.001</u>	DPT_DateTime
PART_Cycle_Time	1 octet	N ₈	1401h	20.013	DPT_Time_Delay
PART_Time_Delay	1 octet	N ₈	1402h	<u>20.013</u>	DPT_Time_Delay
PART_Prewarning_Delay	1 octet	N ₈	1403h	20.013	DPT_Time_Delay
PART_Adaptive_Selection	1 octet	U ₅ N ₃	E401h	228.1000	DPT_Adaptive_Selection
PART_OnOff_Action	2 bit	N ₂	1701h	<u>23.001</u>	DPT_OnOffAction
PART_Alarm_Reaction	2 bit	N ₂	1702h	<u>23.002</u>	DPT_Alarm_Reaction
PART_UpDown_Switch_Action	2 bit	N ₂	1703h	23.003	DPT_UpDown_Action
PART_PB_Action_HVAC	2 bit	N ₂	1704h	23.102	DPT_HVAC_PB_Action
PART_Byte_Value	1 octet	U ₈	0501h	(none)	(none)
PART_Dimming_Value	8 bit	U ₈	0502h	5.001	DPT_Scaling
PART_Adjustable_Selection	1 octet	U ₈	0503h	<u>5.010</u>	DPT_Value_1_Ucount
PART_Render_Value	2 octets	U ₁₆	0701h	7.001	DPT_Value_2_Ucount

PART_Name	Parameter Size	Parameters Format	Standard Parameter Type	DPT_ID	DPT_Name
PART_Light_Value	2 octets	U ₁₆	0702h	<u>7.013</u>	DPT_Brightness
PART_COV_Lux	2 octets	F ₁₆	0901h	9.004	DPT_Value_Lux
PART_Input_Connected	4 bit	none		none	none
PART_PB_Action_HVAC_Extended	3 bit	t.b.d.	t.b.d.	t.b.d.	t.b.d.