System Specifications

3

Standardised Interfaces

6

External Message Interface

3

Summary

This document gives the general description and specification of the Internal - and External Message Interface (IMI and EMI) of the KNX system.

Version 01.03.03 is a KNX Approved Standard.

This document is part of the KNX Specifications v2.1.

Document updates

Version	Date	Modifications
0.1	2000.11.13	SDB: Document creation based on EIBA document.
0.2	2001.03.22	SDB: System Broadcast removed: shifted to Supplement N^r. 16
0.3	2001.06.29	Inclusion of comments from RfV (TM_Timer.ind Style 2). Inclusion of T_Data_Indvidual.con for EMI1.
1.0	2002.03.04	Preparation of the Approved Standard. Inclusion of comments from Final Voting.
1.1	2003.10.23	• **S22 "KNX Radio Fequency"** integrated
1.1	2007.08.07	• **S16 "System Broadcast Services**" integrated
1.1	2007.10.30	• **AN033 "cEMI specification**" integrated
1.1	2007.11.26	• Streamlined naming of PC_Set_Value and PC_Get_Value
1.1	2008.03.25	• **AN064 "cEMI Function Services**" integrated
1.1	2008.04.28	• **AN066 "cEMI extension**" integrated
1.1	2008.08.24	• **AN078 "cEMI extensions for RF and BiBat**" integrated
1.1	2008.10.30	• 4.2.2.4.2 updated according AN066.
1.1	2009.06.24	• Preparation for inclusion in the KNX Specifications v2.0.
1.2.00	2009.11.06	• **AN118 "cEMI Transport Layer"** integrated. • Correction of AT-field encoding in EMI1 and EMI2.
1.2.00	2009.11.20	• [KSG00107]: inclusion of cEMI Frame error handling • Editorial corrections.
01.03.00	2013.07.19	• **AN151 "cEMI AddInfo for KNX RF Multi and new Properties"** integrated.
01.03.02	2013.10.28	Editorial updates for the publication of KNX Specifications 2.1.
01.03.03	2013.11.29	Editorial updates.

References

[01]	Chapter 3/2/5	"Radio Frequency"
[02]	Chapter 3/3/2	"Data Link Layer general"
[03]	Chapter 3/3/3	"Network Layer"
[04]	Chapter 3/3/4	"Transport Layer"
[05]	Chapter 3/3/4	"Application Layer"
[06]	Chapter 3/4/1	"Application Interface Layer"
[07]	Chapter 3/5/1	"Resources"
[08]	Chapter 3/6/2	"Physical External Interface"
[09]	Chapter 3/7/2	"Datapoint Types"
[10]	Chapter 3/7/3	"Standard Identifier Tables"
[11]	Part 3/8	"KNXnet/IP"
[12]	Chapter 3/8/3	"KNXnet/IP Device Management"
[13]	Volume 6	"Profiles"
[14]	Volume 9	"Basic - and System Components and Devices, minimum requirements, standardised solutions, tests"
[15]	Part 9/4	"Couplers"
[16]	Part 9/4	"BCUs and BIMs"
[17]	Part 10/1	"Logical Tag Extended"

Filename:	03_06_03 EMI_IMI v01.03.03 AS.docx
Version:	01.03.03
Status:	Approved Standard
Savedate:	2013.11.29
Number of pages:	115

Contents

1 Overview

The KNX device manufacturer has the choice either to develop a user application to be downloaded to the BAU (“internal user application”) or to implement an external user application. In case of an external user application the communication with the local BAU is via external messages. All the external messages together build the "External Message Interface" (EMI) for which certain subsets are defined.

Communication between the Application Module and the BAU is based on message exchange via the serial PEI, see [08].

All the following explanations of this document are valid for PEI-types 10, 12, 14 and 16 only.

All messages exchanged between the modules inside a device build the “Internal Message Interface” (IMI). The contents of the message code fields of the IMI usually differ from the coding of the EMI. The PEI performs a message translation from the IMI to the EMI and vice versa.

The subsequent clauses explain

- the general message format (clause 2 "Message Format") without protocol overhead and the types of message interfaces,
- how to let an external user application access a certain KNX protocol layer (clause 3.1 “Layer access management”) and
- the message flow in a device,

and the subsets of the External - and Internal Message Interface that are called

- Busmonitor EMI/IMI,
- Data Link Layer EMI/IMI,
- Transport Layer EMI/IMI and
- User Layer EMI/IMI.

Please refer to clause 3.3 “Messages at the KNX protocol layer EMI”. In the corresponding subclauses the message formats available at each EMI subset are summarized.

This Chapter contains a collection of services implemented in various BAU types. In the BAU-descriptions in [16] you will find which services are supported for which BAU-type.

2 Message Format

Two EMI/IMI versions are defined: EMI/IMI 1.x and EMI/IMI 2.0.

The EMI-versions differ in:

- layer access management
- available services
  (where EMI1.x is a subset of EMI2.0)
- service encoding

The message format depends on the serial PEI-protocol (see [08]).

Default Destination

The default destination is the layer where the system shall direct the message to if no redirection is set. Implementation specific exceptions are dealt with in [14].

DLL	Data Link Layer
NL	Network Layer
TL	Transport Layer
TLG	Transport Layer Group Oriented (not in EMI1/IMI1)
TLC	Transport Layer Connection Oriented (not in EMI1/IMI1)
TLL	Transport Layer Local (not in EMI1/IMI1)
AL	Application Layer
ALG	Group oriented part of the AL
MAN	Management part of the AL
PEI	Physical External Interface
USR	Application running in the BAU. If the User is not running, the messages are directed to the PEI.

Message Code Field

The message code field contains the code of the message (also called “service primitive”) which depends on the EMI/IMI release. Service Codes in *italics* for IMI1 and EMI1 denote services of the extended IMI1 / EMI1 (= IMI1+ / EMI1+).

Table 1 – Overview EMI message codes and default destination

	Message code → default destination			
Service primitive	**IMI1**	**EMI1**	**EMI2 / IMI2**	**cEMI**
Ph_Data.req [b]	-	-	01h	-
Ph_Data.con [b]	-	-	1Eh	-
Ph_Data.ind [b]	-	-	19h	-
L_Busmon.ind	29h	49h	2Bh	2Bh
→ destination	→ PEI	→ PEI	→ PEI	→ PEI
L_Data.req	11h	11h	11h	11h
→ destination	→ DLL	→ DLL	→ DLL	→ cEMI Server’s DLL
L_Data.con	2Eh	4Eh	2Eh	2Eh
→ destination	→ TL	→ TL	→ NL	→ CEMI Client’s NL
L_Data.ind	29h	49h	29h	29h
→ destination	→ TL	→ TL	→ NL	→ CEMI Client’s NL
L_SystemBroadcast.req	15h	15h	17h	-
L_SystemBroadcast.con	2Ch	4Ch	26h	-
L_SystemBroadcast.ind	2Dh	4Dh	28h	-
L_Plain_Data.req	-	-	10h	-
→ destination			→ DLL	
L_Raw.req	-	-	d	10h
→ destination				→ DLL
L_Raw.ind	-	-	-	2Dh
→ destination				→ NL
L_Raw.con	-	-	-	2Fh
→ destination				→ NL
L_Poll_Data.req	-	-	13h	13h
→ destination			→ DLL	→ cEMI Server’s DLL
L_Poll_Data.con	-	-	25h	25h
→ destination			→ NL	→ cEMI Client’s NL
L_Meter.ind [b]	-	-	24h	-
→ destination				
N_Data_Individual.req	-	-	21h	-
→ destination			→ NL	
N_Data_Individual.con	-	-	4Eh	-
→ destination			→ TLC	
N_Data_Individual.ind	-	-	49h	-
→ destination			→ TLC	

Table 1 – Overview EMI message codes and default destination

Service primitive	Message code → default destination			
	IMI1	EMI1	EMI2 / IMI2	cEMI
N_Data_Group.req	-	-	22h	-
→ destination			→ NL	
N_Data_Group.con	-	-	3Eh	-
→ destination			→ TLG	
N_Data_Group.ind	-	-	3Ah	-
→ destination			→ TLG	
N_Data_Broadcast.req	-	-	2Ch	-
→ destination			→ NL	
N_Data_Broadcast.con	-	-	4Fh	-
→ destination			→ TLC	
N_Data_Broadcast.ind	-	-	4Dh	-
→ destination			→ TLC	
N_Poll_Data.req	-	-	23h	-
→ destination			→ NL	
N_Poll_Data.con	-	-	35h	-
→ destination			→ TLG	
T_Connect.req	23h	23h	43h	-
→ destination	→ TL	→ TL	→ TLC	
T_Connect.con	-	-	86h	-
→ destination			→ MAN	
T_Connect.ind	33h	43h	85h	-
→ destination	→ MAN	→ MAN	→ MAN	
T_Disconnect.req	24h	24h	44h	-
→ destination	→ TL	→ TL	→ TLC	
T_Disconnect.con	-	-	88h	-
→ destination			→ MAN	
T_Disconnect.ind	34h	44h	87h	-
→ destination	→ MAN	→ MAN	→ MAN	
T_Data_Connected.req	21h	21h	41h	41h
→ destination	→ TL	→ TL	→ TLC	→ cEMI Server
T_Data_Connected.con	-	-	8Eh	-
→ destination			→ MAN	
T_Data_Connected.ind	39h	49h	89h	89h
→ destination	→ MAN	→ MAN	→ MAN	→ cEMI Client
T_Data_Group.req	22h	22h	32h	-
→ destination	→ TL	→ TL	→ TLG	
T_Data_Group.con	3Eh	4Eh	7Eh	-
→ destination	→ ALG	→ ALG	→ ALG	
T_Data_Group.ind	3Ah	4Ah	7Ah	-
→ destination	→ ALG	→ ALG	→ ALG	

Table 1 – Overview EMI message codes and default destination

	Message code → default destination			
Service primitive	**IMI1**	**EMI1**	**EMI2 / IMI2**	**cEMI**
T_Data_Broadcast.req	2Bh	2Bh	4Ch	-
→ destination	→ TLC	→ TLC	→ TLC	
T_Data_Broadcast.con	-	-	8Fh	-
→ destination			→ MAN	
T_Data_Broadcast.ind	38h	48h	8Dh	-
→ destination	→ MAN	→ MAN	→ MAN	
T_Data_SystemBroadcast.req	25h	25h	-	-
→ destination				
T_Data_SystemBroadcast.con	3Ch	4Ch	-	-
→ destination				
T_Data_SystemBroadcast.ind	3Dh	4Dh	-	-
→ destination				
T_Data_Individual.req	2Ah	2Ah	4Ah	4Ah
→ destination	→ TL	→ TL	→ TLC	→ cEMI Server
T_Data_Individual.con	3Fh	4Fh	9Ch	-
→ destination	→ MAN	→ MAN	→ MAN	
T_Data_Individual.ind	32h	42h	94h	94h
→ destination	→ MAN	→ MAN	→ MAN	→ cEMI Client
T_Poll_Data.req	-	-	33h	-
→ destination			→ TLG	
T_Poll_Data.con	-	-	75h	-
→ destination			→ ALG	
M_Connect.req	-	-	-	-
→ destination				
M_Connect.con	-	-	-	-
→ destination				
M_Connect.ind	-	-	D5h	-
→ destination			→ User → PEI if User is not running.	
M_Disconnect.req	-	-	-	-
→ destination				
M_Disconnect.con	-	-	-	-
→ destination				
M_Disconnect.ind	-	-	D7h	-
→ destination			→ User → PEI if User is not running.	
M_User_Data_Connected.req	31h	31h	82h	-
→ destination	→ MAN	→ MAN	→ MAN	
M_User_Data_Connected.con	-	-	D1h	-
→ destination			→ User → PEI if User is not running.	

Table 1 – Overview EMI message codes and default destination

Service primitive	Message code → default destination			
	IMI1	EMI1	EMI2 / IMI2	cEMI
M_User_Data_Connected.ind	59h	49h	D2h	-
→ destination	→ User	→ User	→ User → PEI if User is not running.	
A_Data_Group.req	-	-	72h	-
→ destination			→ ALG	
A_Data_Group.con	-	-	EEh	-
→ destination			→ User → PEI if User is not running.	
A_Data_Group.ind	-	-	EAh	-
→ destination			→ User → PEI if User is not running.	
M_User_Data_Individual.req	-	-	81h	-
→ destination			→ MAN	
M_User_Data_Individual.con	-	-	DEh	-
→ destination			→ User → PEI if User is not running.	
M_User_Data_Individual.ind	-	-	D9h	-
→ destination			→ User → PEI if User is not running.	
A_Poll_Data.req	-	-	73h	-
→ destination			→ ALG	
A_Poll_Data.con	-	-	E5h	-
→ destination			→ User → PEI if User is not running.	
M_InterfaceObj_Data.req	-	-	9Ah	-
→ destination			→ MAN	
M_InterfaceObj_Data.con	-	-	DCh	-
→ destination			→ User → PEI if User is not running.	
M_InterfaceObj_Data.ind	-	-	D4h	-
→ destination			→ User → PEI if User is not running.	
U_Value_Read.req	35h	35h	74h	-
→ destination	→ ALG	→ ALG	→ ALG	
U_Value_Read.con	55h	45h	E4h	-
→ destination	→ User	→ User	→ User	
U_Flags_Read.req	37h	37h	7Ch	-
→ destination	→ ALG	→ ALG	→ ALG	
U_Flags_Read.con	57h	47h	ECh	-
→ destination	→ User	→ User	→ User	

Table 1 – Overview EMI message codes and default destination

	Message code → default destination			
Service primitive	**IMI1**	**EMI1**	**EMI2 / IMI2**	**cEMI**
U_Event.ind	5Dh	4Dh	E7h	-
→ destination	→ User	→ User	→ User	
U_Value_Write.req	36h	36h	71h	-
→ destination	→ ALG	→ ALG	→ ALG	
U_User_Data	-		>D0h	-
→ destination			→ User	
PC_Set_Value.req c	46h	46h	A6h	-
→ destination	→ MAN	→ MAN	→ MAN	
PC_Get_Value.req c	4Ch	4Ch	ACh	-
→ destination	→ MAN	→ MAN	→ MAN	
PC_Get_Value.con	4Bh	4Bh	ABh	-
→ destination	→ PEI	→ PEI	→ PEI	
PEI_Identify.req	-	-	A7h	-
→ destination				
PEI_Identify.con	-	-	A8h	-
→ destination			→ PEI	
PEI_Switch.req	-	-	A9h	-
→ destination				
TM_Timer.ind	-	-	C1h	-
→ destination			→ User	
M_PropRead.req	-	-	-	FCh
→ destination				→ cEMI Server
M_PropRead.con	-	-	-	FBh
→ destination				→ cEMI Client
M_PropWrite.req	-	-	-	F6h
→ destination				→ cEMI Server
M_PropWrite.con	-	-	-	F5h
→ destination				→ cEMI Client
M_PropInfo.ind	-	-	-	F7h
→ destination				→ cEMI Client
M_FuncPropCommand.req	-	-	-	F8h
→ destination				→ cEMI Server
M_FuncPropStateRead.req	-	-	-	F9h
→ destination				→ cEMI Server
M_FuncPropCommand.con M_FuncPropStateread.con	-	-	-	FAh
→ destination				→ cEMI Client
M_Reset.req	-	-	-	F1h
→ destination				→ cEMI Server

Table 1 – Overview EMI message codes and default destination

	Message code → default destination			
Service primitive	**IMI1**	**EMI1**	**EMI2 / IMI2**	**cEMI**
M_Reset.ind	-	-	-	F0h
→ destination				→ cEMI Client
a The default destination is the layer where the system shall direct the message to; please refer to clause 2 for more details.				
b Shall only be used in RF.				
c In BCU 2 not posted to MAN but handled in PEI module				
d EMI 2 = L_Plain_Data.req				

3 EMI1 and EMI2

3.1 Layer access management

3.1.1 Introduction

Layer access management shall allow giving an application direct access to a KNX OSI communication layer.

In EMI1 it shall be possible to switch on and off any KNX protocol layer, so that each protocol layer shall be directly accessible. But only a few switch combinations make real sense. Layers shall be switched by writing to memory location “system status”. From the PEI a PC_Set_Value.req message shall be used to this purpose.

In EMI2 layers shall not be switched, but it shall be possible to redirect messages from their default destination to another destination. From the PEI an LM_Switch.req-message shall be used.

EXAMPLE Messages can be redirected to the PEI in order to make them accessible to an external user.

The following clause summarizes useful combinations and the consequences for them.

3.1.2 Useful Layer interfaces and their consequences for the application design

- In an EMI1 system, the internal user application shall only run if all the KNX communication layers are switched on. This shall be true for the default "system status".
- In an EMI1 system, parallel I/O PEI (see [08]) shall also require the default system status.
- The internal user application shall only run if the PEI-type expected by the internal user application corresponds to the PEI-type measured at the A/D converter of the BAU.
- An external user application (i.e. one that is not situated at the BAU processor) shall need serial PEI communication (via PEI-types 10, 12, 14 or 16) for communication with the local BAU. The local BAU may offer to the external user application an access to one of the following KNX communication layers:
    - to the KNX Data Link Layer in Busmonitor mode. The corresponding external interface is called Busmonitor EMI. The external user application must be a Busmonitor application.
    - to the KNX Data Link Layer in normal mode. The corresponding external interface is called Data Link Layer EMI.
    - to the KNX Network Layer in normal mode. The corresponding external interface is called Network Layer EMI.
    - to the KNX Transport Layer. The corresponding external interface is called Transport Layer EMI. The Transport Layer EMI shall exist in 3 versions:
        1. full Transport Layer EMI
        2. connection-oriented Transport Layer EMI
        3. connectionless Transport Layer EMI
    - to the KNX Application Layer. The corresponding external interface is called Application Layer EMI. The Application Layer EMI shall consist of two parts:
        1. the group-oriented Application Layer EMI
        2. the Management EMI
    - to the KNX Application Interface Layer and to the internal user application. The corresponding external interface is called Application Interface layer EMI, and shall be available in the default system status.

- Selecting access to a KNX communication layer:

EMI1.x	(PEI-type 12, 16 or 20):	use the	PC_Set_Value.req-message
EMI2.0	(PEI-type 10):		LM_Switch.req message

The description and use of these services is BAU implementation dependent, and are therefore not described in clause 3.3 “Messages at the KNX protocol layer EMI”. Please refer to the EMI description of the BAUs in [16].

Consequences of the chosen access to a KNX communication layer for the application design

- In an EMI1 system with Busmonitor EMI, Data Link Layer EMI and full Transport Layer EMI selected the internal user application and the object servers shall not run. Instead the external user layer shall implement all the KNX communication layers above the chosen EMI.
- A broadcast communication relationship to the local device shall be possible in case it is brought to programming mode, independently of the chosen EMI. The external user application never has to implement the broadcast server functionality for the Individual Address Object.
- With connection-oriented Transport Layer EMI selected, multicast communication relationships shall be possible between the internal user application and a remote partner device, i.e. Group Object-oriented communication by the internal user application shall be possible. At the external user application a connection-oriented one-to-one communication relationship to a remote partner device shall be possible. Except for the Group Object- and the Individual Address Object no object server shall run at the local BAU.
- With connectionless Transport Layer EMI selected, one-to-one connection-oriented communication relationships shall be possible between the internal user application and a remote partner device. At the external user application connectionless one-to-one and multicast communication relationships shall be possible. The internal Group Object server shall not run.
- With default user layer EMI multicast communication relationships and either one-to-one connectionless or one-to-one connection-oriented communication relationships between the internal user application and a remote partner device shall be possible. The external user application can only communicate connection-oriented or connectionless with the local BAU respectively the internal user application. Speciality: Even if the internal user application does not run, because the PEI-type expected is not identical to the PEI-type measured at the A/D converter, the object server functionality shall be active.

3.1.3 Layer access in EMI1

In EMI1, layers shall be accessed by writing to the memory location “system status”. Locally this shall be done by the PC_Set_Value.req service, remotely by the service A_Memory_Read. “System status” shall have the coding as specified in Figure 1.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	Bit 0
PARITY	DM	UE	SE	ALE	TLE	LLM	PROG

Field	Function
PROG	not used, shall be 0
LLM	enable Busmonitor mode (1=disabled)
TLE	enable Transport Layer (1=enabled)
ALE	enable Application Layer (1=enabled)
SE	enable PEI (1=enabled)
UE	enable user program (1=enabled)
DM	not used, shall be 0
PARITY	even parity for the “system status” octet

Figure 1 - System status in EMI1

EXAMPLES

Mode	Value
Busmonitor	90h
Data Link Layer	12h
Transport Layer	96h
Application Layer	1Eh

3.1.4 Layer access in EMI2

The PEI_Switch.req service shall set the static redirection table and the contents of the memory location “system status”.

Message format

<table>
<tr><th>octet 1</th><th>octet 2</th><th colspan="2">octet 3</th><th colspan="2">octet 4</th><th colspan="2">octet 5</th><th colspan="2">octet 6</th><th colspan="2">octet 7</th></tr>
<tr><td rowspan="2">m_code</td><td rowspan="2">System Status</td><td colspan="10">new redirection for modules</td></tr>
<tr><td>LL</td><td>NL</td><td>TLG</td><td>TLC</td><td>TLL</td><td>AL</td><td>MAN</td><td>PEI</td><td>USR</td><td>res</td></tr>
<tr><td>7 6 5 4 3 2 1 0</td><td>7 6 5 4 3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

System Status:	00h:	do not change memory location “system status”
	!= 00h:	new contents of memory location “system status”

The “system status” shall be used to set the LL to Busmonitor mode (set to 90h).

The redirections shall contain module IDs to which messages addressed to the specified modules shall be redirected. For example an “8h” at position “NL” in the message shall have the effect, that all messages addressed to NL will be redirected to the PEI (module ID 8) instead. Module ID “0” shall mean that messages shall be discarded.

List of module IDs

Destination Layer	ID
LL	1
NL	2
TLG	3
TLC	4
TLL	5
AL	6
MAN	7
PEI	8
USR	9

Examples

	system status											
m_code	SS	LL	NL	TLG	TLC	TLL	AL	MAN	PEI	USR	res.	
1 octet	*1 octet*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	*4 bit*	
	00h	1	2	3	4	5	6	7	8	8	A	Application Layer
	00h	1	2	3	4	4	8	8	8	0	A	Transport Layer Remote
	00h	1	8	3	4	5	6	7	8	0	A	Data Link Layer
	90h	1	8	3	4	5	6	7	8	0	A	Data Link Layer (Busmonitor mode)
	00h	1	2	3	4	5	6	7	8	9	A	Normal Mode

Figure 2 - Setting System Status in EMI2 using the LM_Switch-service

The above examples illustrate typical cases. In normal mode messages shall be directed to their default destination (see Figure 2).

NOTE No service has the TLL as its default destination and therefore no message will be directed to TLL in normal mode.

Switching to LL means that all messages addressed to the NL (this includes all which originate from LL) shall be redirected to the PEI (8). Messages addressed to the USR shall be discarded since the User has no functionality in this case. Busmonitor mode shall be a special case of switching to LL and shall be initiated by writing a value of 90h to “system status”.

A switch to the remote TL shall redirect all messages from TLL to TLC to enable a TL-connection from the external user to a remote user. All incoming messages addressed to the AL or MG (originating from TLG and TLC) shall be redirected to the PEI in order to reach the external user. Messages for the USR shall be discarded.

Switching to the AL shall mean that all messages addressed to the user shall be redirected to the PEI and shall thus be obtained by the external user.

3.2 Message flow

3.2.1 Normal Mode

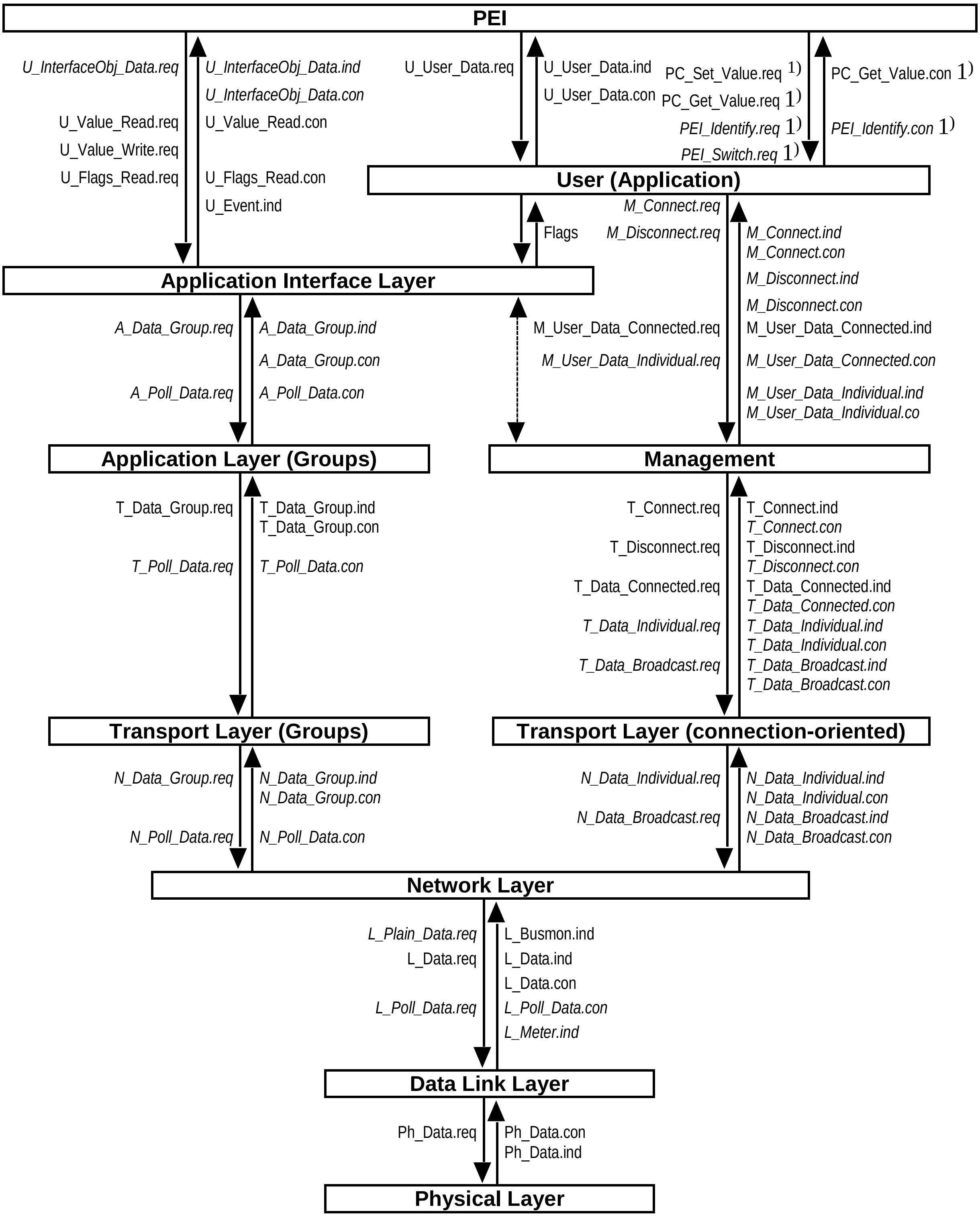


Figure 3 - Message Flow in a BAU

[1] These services may be processed in the PEI-module or in device management alternatively.

This figure indicates the message flow in a BAU. The messages available to the PEI are those in “normal mode”, i.e. when no switch to a specific layer has been performed (see 3.1 “Layer access management”). Services printed in *italics* indicate services that shall be available in EMI2 only. In general EMI2 shall include all EMI1 services, but with a different service coding.

TM_Timer.ind has been omitted since the destination layer can be chosen when a timer is initialised by the API function call (usually User or PEI).

The dotted arrow indicates Property access from the Management module to the Interface Objects that shall be are located in the AIL.

3.2.2 Local Transport Layer

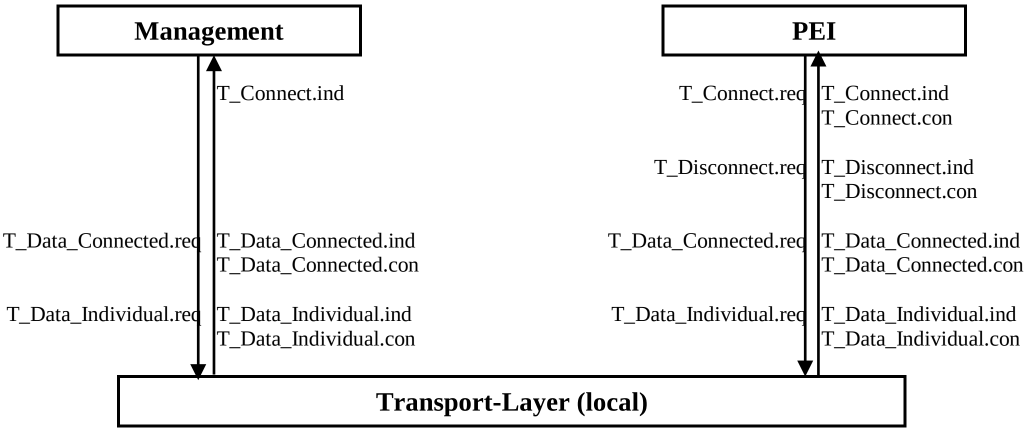


Figure 4 - Local Transport Layer

If the PEI is switched to the local TL (TLL), all services of the TLC shall be available to the external user in order to access the management of the local PEI.

3.3 Messages at the KNX protocol layer EMI

3.3.1 Notation conventions

The following convention is used in the messages coding: lower case letters denote single bits; upper case double letters (e.g. XX) denote one octet. Those data in a message that are named “unused” shall be set to zero when sent to the server (i.e. in .req messages), whereas “unused” data received from the server (i.e. in .ind and .con messages) shall not be interpreted by the host, regardless of their value.

3.3.3 Busmonitor EMI

3.3.3.1 Available messages

In Busmonitor mode exactly the L_Busmon.ind message, the L_Plain_Data.req message, and the LM_Reset.ind message shall be available.

3.3.3.2 L_Busmon.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5
m_code	Status	Time Stamp		Control
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

octet 6	…	octet n-1	octet n
LPDU			Checksum (FCS)
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

- **octete 1: m_code:**
  definition: This shall be the message code for L_Busmon.ind.
  encoding: 1 octet
  value: See Table 1.
- **octet 2: status:**
  encoding: 1 octet: 8 bits

bit nr:	7	6	5	4	3	2	1	0
name:	frame error	bit error	parity error	overflow	lost	sequence number (mod 8)		

  - **frame error**
    definition: A frame error was detected in one or several of the frame bits.
    format: 1 bit
  - **bit error**
    definition: An invalid bit is detected in one or several of the frame characters.
  - **parity error**
    definition: An invalid parity bit was detected in one or several of the frame bits.
  - **overflow:**
    definition: The overflow flag is set.

- **lost:**
  definition: The Lost flag shall be set if at least one frame or frame piece is lost by the Busmonitor.
    - NOTE The difference between the sequence number of the previous BUSMON.ind without lost flag set and the sequence number of the BUSMON.ind with lost flag set may not reflect exactly the number of lost frames or frame pieces.

  encoding: 1 bit
  0: no frame has been lost
  1: one or more frames have been lost
- **sequence number:**
  definition: Each received frame shall let the Data Link Layer increment the modulo 8 value of the sequence number. The least significant bit of octet 2 shall also be the least significant bit of the sequence number.
  encoding: 3 bit value

- **octets 3 and 4: Time stamp:**
  definition: Time stamp shall be a 16 bit value and shall refer to the relative time taken exactly at the time when the frame's control field is completely received at the Data Link Layer.
  The time shall be the value of the free-running counter of the BAU. The time unit ("tick") shall depend on the clock rate of the BAU microcontroller.
  encoding: 16 bit unsigned integer value
- **Control field (octet 5) and LPDU (octet 6 to octet n-1):**
  definition: These fields shall contain the received Data Link Layer PDU.
  n shall be less than or equal to 27.
- **Checksum (FCS):**
  definition: The value of the FCS octet shall be the FCS value received from the KNX medium. It shall be the task of the external Busmonitor application to check its correctness.

3.3.3.3 L_Plain_Data.req (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7 … octet n
m_code	(unused)	Time				Data
	00h	octet 3	octet 2	octet 1	octet 0	

It shall be possible to send up to 28 octets of plain data by this service.

In “time” optionally a time delay before sending the message on the bus can be specified. If “time”=00000000h the frame shall be sent immediately. Otherwise the frame shall be sent if the free running system counter of the sending device equals the value given in “time”.

3.3.3.4 LM_Reset.ind message (EMI1/EMI2)

This message is protocol dependent. It is described in [08].

3.3.4 Data Link Layer EMI

3.3.4.1 Definition

In normal operation mode of the Data Link Layer, exactly the L_Data.req, the L_Data.ind, the L_Data.con, the L_Poll_Data.req and the L_Poll_Data.con messages shall be available.

3.3.4.2 L_Data.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7	octet 8 … octet n
m_code	Control	Unused		Destination Address		AT, NPCI, LG	NPDU
				high	low		
		00h	00h			[2]	

- **m_code:**
  definition: This shall be the message code for L_Data.req.
  encoding: 1 octet
  value: See Table 1.
- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	r	r	r	r	p		a	r
value	0	0	0	0				0

  - **priority (p)**
    definition: This shall be the value of the priority that shall be used for the transmission of the frame requested with this L_Data.req
    encoding: Value binary encoded.
  - **ack_request (a)**
    definition: This flag shall indicate whether a Data Link Layer acknowledge shall be requested when the frame is transmitted on KNX or whether this is don't care.
    encoding: 0 = Don't care [2]
    1 = no L2-acknowledge requested
- **AT, NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	AT	NPCI			octet count (LG)			
value								

  - **Address Type (AT):**
    encoding: 0: The Destination Address shall be an Individual Address.
    1: The Destination Address shall be a Group Address.

[2] "Don't care" means that no explicit L2-acknowledge is requested by the upper layer(s); this means the default behaviour of the Data Link Layer concerning L2- acknowledge requesting applies, as laid down in the specifications of the communication medium.

3.3.4.3 L_Data.con message (EMI1 / EMI2)

The L_Data.con message shall exist in a positive and a negative version. Both versions shall differ in the value of the Confirm flag © contained in the Control Field: the positive L_Data.con message shall have the Confirm flag cleared, the negative L_Data.con shall have the Confirm flag set.

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7	octet 8 … octet n
m_code	Control	Unused		Destination Address		AT, NPCI, LG	NPDU
				high	low		
		X	X			[2]	

- **m_code:**
  definition: This shall be the message code for the L_Data.con.
  encoding: 1 octet
  value: See Table 1.
- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	x	x	r	x	p		x	c
value								

  - **repeat (r)**
  - **priority (p)**
    definition: This shall be the value of the priority that is used for the transmission of the L_Data.req frame that is confirmed by this L_Data.con frame.
    encoding: Value binary encoded.
  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this L_Data.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.
- **AT, NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	AT	NPCI			octet count			
value								

  - **Address Type (AT):**
    encoding: 0: The Destination Address shall be an Individual Address.
    1: The Destination Address shall be a Group Address.

3.3.4.4 L_Data.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7	octet 8 … octet n
m_code	Control	Source Address		Destination Address		AT, NPCI, LG	NPDU
		high	low	high	low	[2]	

- **m_code:**
  definition: This shall be the message code for the L_Data.ind.
  encoding: 1 octet
  value: See Table 1.
- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	x	x	x	x	p		x	x
value								

  - **priority (p)**
    definition: This shall be the value of the priority of the telegram that has been received and that is transferred by this L_Data.ind
    encoding: Value binary encoded.
- **AT, NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	AT	NPCI			octet count			
value								

  - **Address Type (AT):**
    encoding: 0: The Destination Address of the received message shall be interpreted as an Individual Address.
    1: The Destination Address of the received message shall be interpreted as a Group Address.

3.3.4.5 L_Poll_Data.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7	
m_code	Control	unused		Polling Group		reser-ved	nr of slots
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	7 6 5 4	3 2 1 0
	1 1 1 1 0 0 0 0					0 0 0 0	

3.3.4.6 L_Poll_Data.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7	
m_code	Control	unused		Polling Group			nr of slots
				high	low		
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	7 6 5 4	3 2 1 0
		X	X			x x x x	

octet 8				octet n
poll data 0	…	poll data	…	poll data n -1
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:							x	c
value	1	1	1	1	0	0		

  - **confirm (c)**

    definition: This Confirm flag shall indicate whether this L_Poll_Data.con is a positive confirmation or a negative confirmation.

    encoding: 0: This shall be a negative confirmation.
    1: This shall be a positive confirmation.

3.3.4.7 L_SystemBroadcast.req message (EMI/IMI 1 / EMI/IMI 2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		AT, NPCI, LG
				high	low	
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	7 6 5 4 3 2 1 0
		00h	00h			

octet 8				octet n
…		NPDU		…
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

- **m_code:**
  definition: This shall be the message code for the L_SystemBroadcast.req.
  encoding: 1 octet
  value: See Table 1.
- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	r	r	r	r	p		a	r
value	1	0	0	0				0

  - **priority (p)**
    definition: This shall be the value of the priority that shall be used for the transmission of the frame requested with this L_SystemBroadcast.req.
    encoding: Value binary encoded.
  - **ack_request (a)**
    definition: This flag shall indicate whether a Data Link Layer acknowledge shall be requested when the frame is transmitted on KNX or whether this is don’t care.
    encoding: 0 = Don't care [3]
    1 = no L2-acknowledge requested
- **AT, NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	AT	NPCI			octet count (LG)			
value								

  - **Address Type (AT):**
    encoding: 0: The Destination Address shall be an Individual Address.
    1: The Destination Address shall be a Group Address.

3.3.4.8 L_SystemBroadcast.con message (EMI/IMI 1 / EMI/IMI 2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		AT, NPCI, LG
				high	low	
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	7 6 5 4 3 2 1 0
		XX	XX			

octet 8				octet n
…		NPDU		…
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

[3] "Don’t care" means that no explicit L2-acknowledge is requested by the upper layer(s); this means the default behaviour of the Data Link Layer concerning L2- acknowledge requesting applies, as laid down in the specifications of the communication medium.

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused		r	unused	priority		unused	c
value	x	x		x			x	

  - **repeat (r)**
  - **priority (p)**
    definition: This shall be the value of the priority that is used for the transmission of the L_SystemBroadcast.req frame that is confirmed by this L_SystemBroadcast.con frame.
    encoding: Value binary encoded.
  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this L_SystemBroadcast.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

The L_SystemBroadcast.con message exists in a positive and a negative version. Both versions differ in the value of the error flag contained in the Control field octet: the positive L_SystemBroadcast.con message has the error flag reset, the negative one set.

3.3.4.9 L_SystemBroadcast.ind message (EMI/IMI 1 / EMI/IMI 2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	Source Address		Destination Address		NPCI, LG
		high	low	high	low	
76543210	76543210	76543210	76543210	76543210	76543210	76543210

octet 8				octet n
…		NPDU		…
76543210		76543210		76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	x	x	x	x			x	x

  - **priority (p)**
    definition: This shall be the value of the priority that is used for the transmission of the L_SystemBroadcast.ind frame.
    encoding: Value binary encoded.
- **NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	x							

3.3.5 Network Layer EMI

3.3.5.1 Definition

In Network Layer mode exactly the N_Data_Individual.req, N_Data_Individual.con, N_Data_Individual.ind, N_Data_Group.req, N_Data_Group.con, N_Data_Group.ind, N_Data_Broadcast.req, N_Data_Broadcast.con, N_Data_Broadcast.ind, N_Poll_Data.req and N_Poll_Data.con messages are available. All NL services belong to EMI/IMI2 only.

3.3.5.2 N_Data_Individual.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		LG
				high	low	
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	7 6 5 4 3 2 1 0
		00h	00h			

octet 8						octet n
		…	TPDU	…		
7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0

3.3.5.3 N_Data_Individual.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		LG
				high	low	
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	7 6 5 4 3 2 1 0
		XX	XX			

octet 8						octet n
		…	TPDU	…		
7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused							c
value	x	x	x	x	x	x	x	

  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this N_Data_Individual.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7: octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				octet count (LG)			
value	x	x	x	x				

3.3.5.4 N_Data_Individual.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	Source Address		Destination Address		NPCI, LG
		high	low	high	low	
76543210	76543210	76543210	76543210	76543210	76543210	76543210

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	x	x	x	x			x	x

- **NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	x							

3.3.5.5 N_Data_Group.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		NPCI, LG
				high	low	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h			

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	0	0	0	0			0	0

- **NPCI, LG:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	0							

3.3.5.6 N_Data_Group.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		LG
				high	low	
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	7 6 5 4 3 2 1 0
		XX	XX			

octet 8					octet n
…		TPDU			…
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused							c
value	x	x	x	x	x	x	x	

  - **confirm (c)**

    definition: This Confirm flag shall indicate whether this N_Data_Group.con is a positive confirmation or a negative confirmation.

    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7: octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				octet count (LG)			
value	x							

3.3.5.7 N_Data_Group.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Destination Address		NPCI, LG
				high	low	
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	7 6 5 4 3 2 1 0
		XX	XX			

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	x	x	x	x			x	x

- **NPCI, octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	x							

3.3.5.8 N_Data_Broadcast.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		unused		NPCI, LG
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	7 6 5 4 3 2 1 0
		00h	00h	00h	00h	

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	0	0	0	0			0	0

- **NPCI, octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	0							

3.3.5.9 N_Data_Broadcast.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		unused		LG
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		XX	XX	00h	00h	

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				unused		unused	c
value	x	x	x	x	p	p	x	

  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this N_Data_Broadcast.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7: octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				octet count (LG)			
value	X							

3.3.5.10 N_Data_Broadcast.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	Source Address		unused		NPCI, LG
		high	low			
76543210	76543210	76543210	76543210	76543210	76543210	76543210
				XX	XX	

octet 8						octet n
…			TPDU			…
76543210			76543210			76543210

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				priority		unused	
value	x	x	x	x			x	x

- **NPCI, octet count (LG):**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count (LG)			
value	x							

3.3.5.11 N_Poll_Data.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Polling Group		NPCI
				high	low	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
	11110000	00h	00h			

- **NPCI:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				nr of slots			
value	x	x	x	x				

3.3.5.12 N_Poll_Data.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		Polling Group		NPCI
				high	low	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
	11110000	XX	XX			

octet 8						octet n
poll data 0			…			poll data m - 1
76543210			76543210			76543210

- **NPCI:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				nr of slots			
value	x	x	x	x	m			

3.3.6 Transport Layer EMI

3.3.6.1 Definition

The full Transport Layer EMI shall consist of the connection-oriented Transport Layer EMI and the connectionless Transport Layer EMI.

The connection-oriented Transport Layer EMI shall consist of exactly the messages T_Connect.req, T_Connect.ind, T_Connect.con, T_Disconnect.req, T_Disconnect.ind, T_Disconnect.con, T_Data_Connected.req, T_Data_Connected.ind and T_Data_Connected.con.

The connectionless Transport Layer EMI shall consist of exactly the messages T_Data_Individual.req, T_Data_Individual.ind, T_Data_Individual.con, T_Data_Broadcast.req, T_Data_Broadcast.ind, T_Data_Broadcast.con, T_Data_Group.req, T_Data_Group.ind and T_Data_Group.con [4].

The T_Data_Connected.con and T_Data_Group.con messages shall exist in a positive and a negative version. Both versions differ in the value of the error flag (c) contained in the Control field: the positive message shall have the error flag cleared; the negative shall have the error flag set.

The T_Data_Connected.req, T_Data_Connected.ind, T_Data_Connected.con, T_Data_Individual.req, T_Data_Individual.ind, T_Data_Individual.con, T_Data_Broadcast.req, T_Data_Broadcast.ind, T_Data_Broadcast.con, T_Data_Group.req, T_Data_Group.ind and T_Data_Group.con messages shall allow transferring APDUs. See [04] and [05] for the APDU formats allowed to be transferred with the respective service.

3.3.6.2 T_Connect.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused		Destination Address	
				high	low
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
	0 0 0 0 0 0 0 0	00h	00h		

3.3.6.3 T_Connect.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control (unused)	Destination Address			
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
	x x x x x x x x	XX	XX	XX	00h

[4] The messages corresponding to T_System_Broadcast_xx service primitives come in addition to this specification on open media. (service specification under editing) Except for the message code, format is identical to T_Data_Broadcast messages

3.3.6.4 T_Connect.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	
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
	x x x x x x x x	XX	XX	XX	00h

3.3.6.5 T_Disconnect.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	unused
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
	0 0 0 0 0 0 0 0	00h	00h	00h	00h

3.3.6.6 T_Disconnect.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused		unused	
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
	x x x x x x x x	XX	XX	XX	00h

3.3.6.7 T_Disconnect.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused		unused	
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
	x x x x x x x x	XX	XX	XX	00h

3.3.6.8 T_Data_Connected.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		unused		
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h	00h	00h	

octet 8						octet n
unused	…	APDU				…
765432			76543210			76543210
000000						

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:					p			
value	0	0	0	0			0	0

  - **priority (p)**
    definition: This shall be the value of the priority of the telegram that shall be transmitted to execute this T_Data_Connected.req.
    encoding: Value binary encoded.
- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count			
value	0							

3.3.6.9 T_Data_Connected.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused		unused	TSAP	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		XX	XX	XX	00h	

octet 8						octet n
unused	…	APDU				…
765432			76543210			76543210
xxxxxx						

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused							c
value	x	x	x	x	x	x	x	

  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this T_Data_Connected.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused				octet count			
value	x	x	x	x				

3.3.6.10 T_Data_Connected.ind (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	Source Address		unused		
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	7 6 5 4 3 2 1 0
				XX	00h	

octet 8						octet n
unused (bits 7–2)	…		APDU			…
7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0			7 6 5 4 3 2 1 0
x x x x x x						

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value								

  - **priority (p)**
    definition: This shall be the value of the priority that is used for the transmission of the T_Data_Connected.ind frame.
    encoding: Value binary encoded.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			octet count			
value								

3.3.6.11 T_Data_Group.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused			cr_id	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h	00h		

octet 8				octet n
unused	…	APDU		…
76543210		76543210		76543210
000000				

- **Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	0	0	0	0			0	0

  - **priority (p)**
    definition: This shall be the value of the priority of the telegram that shall be transmitted to execute this T_Data_Group.req.
    encoding: Value binary encoded.
- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value	0							

In EMI1 the T_Data_Group.req service fulfils additionally the purpose of the T_Data_Broadcast.req in EMI2. In this case the cr_id octet is unused (= 0).

3.3.6.12 T_Data_Group.con message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused			cr_id	
76543210	76543210	76543210	76543210	76543210	76543210	76543210
		XX	XX	00h		

octet 8		octet 9			octet n
unused		…		data	…
765432	10	76543210			76543210
	00				

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	C
value	x	x	x	x			x	

  - **confirm (c)**
    definition: This Confirm flag shall indicate whether this T_Data_Group.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value								

3.3.6.13 T_Data_Group.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused		unused	cr_id
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
		XX	XX	00h	

octet 7	octet 8	octet 9		octet n
	unused	…	APDU	…
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
	x x x x x x			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	x	x	x	x			x	x

In EMI1 the T_Data_Group.ind service fulfils additionally the purpose of the T_Data_Broadcast.ind of EMI2. In this case the cr_id octet is unused (=00h).

3.3.6.14 T_Data_Broadcast.req message (EMI2/IMI1+)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	unused
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
		00h	00h	00h	00h

octet 7	octet 8	octet 9		octet n
	unused …	APDU		…
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
	0 0 0 0 0 0			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	0	0	0	0			0	0

3.3.6.15 T_Data_Broadcast.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	unused
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
		XX	XX	00h	00h

octet 7	octet 8	octet 9		octet n
	unused …	APDU		…
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
	0 0 0 0 0 0			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	Confirm
value	x	x	x	x	x	x	x	

  - **Confirm (c)**

    definition: This Confirm flag shall indicate whether this T_Data_Broadcast.con is a positive confirmation or a negative confirmation.

    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

3.3.6.16 T_Data_Broadcast.ind message (EMI2/ IMI1+/ EMI1+)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	unused
		high	low		
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
		XX	XX	00h	00h

octet 7	octet 8		octet 9		octet n
	unused	…	APDU		…
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
	x x x x x x				

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	x	x	x	x			x	x

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value	x							

3.3.6.17 T_Data_SystemBroadcast.req message (IMI1/EMI1)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	unused
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
		00h	00h	00h	00h

octet 7	octet 8		octet 9		octet n
	unused	…	APDU		…
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
	0 0 0 0 0 0				

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	0	0	0	0			0	0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value								

3.3.6.18 T_Data_SystemBroadcast.con message (IMI1/EMI1)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	unused
		high	low		
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
				00h	00h

octet 7	octet 8	octet 9		octet n
	unused	…	APDU	…
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
	x x x x x x			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	Error
value:	x	x	x	x			x	

- **Priority (P):**
  definition: This shall be the value of the priority that is used for the transmission of the T_Data_SystemBroadcast.req frame that is confirmed by this T_Data_SystemBroadcast.con frame.
  encoding: Value binary encoded.
- **Error:**
  definition: This Error flag shall indicate whether this T_Data_SystemBroadcast.con is a positive confirmation or a negative confirmation.
  encoding: 0: This shall be a positive confirmation.
  1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value	x							

The T_Data_SystemBroadcast.con message shall exist in a positive version and a negative version. These versions shall only differ from each other in the value of the error flag contained in the Control Field: the positive T_Data_SystemBroadcast.con message shall have the error flag cleared, the negative T_Data_SystemBroadcast.con message shall have the error flag set.

3.3.6.19 T_Data_SystemBroadcast.ind message (IMI1/EMI1)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	unused
		high	low		
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
				00h	00h

octet 7	octet 8	octet 9		octet n
		…	APDU	…
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
	x x x x x x			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value	x	x	x	x			x	

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	x							

3.3.6.20 T_Data_Individual.req message (EMI2/ IMI1+)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	Destination Address	
				high	low
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
		00h	00h		

octet 7	octet 8	octet 9		octet n
	unused	…	APDU	…
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
	0 0 0 0 0 0			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	priority		unused	unused
value	0	0	0	0			0	0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value	0							

3.3.6.21 T_Data_Individual.con message (EMI2/IMI1+/EMI1+)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	Destination Address	
				high	low
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
		XX	XX		

octet 7	octet 8	octet 9		octet n
	unused …	APDU		…
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
	x x x x x x			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	Confirm
value	x	x	x	x	x	x	x	

  - **Confirm (C)**

    definition: This Confirm flag shall indicate whether this T_Data_Individual.con is a positive confirmation or a negative confirmation.

    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value:	x							

3.3.6.22 T_Data_Individual.ind message (EMI2/ IMI1+/ EMI1+)

Message format

octet 1	octet 2	oct²et 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		Destination Address	
		high	low	high	low
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

octet 7	octet 8	octet 9		octet n
	unused …	APDU		…
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
	x x x x x x			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value	x	x	x	x			x	x

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	x							

3.3.6.23 T_Poll_Data.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	Polling Group	
				high	low
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
	1 1 1 1 0 0 0 0	00h	00h		

octet 7
7 6 5 4 3 2 1 0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	nr of slots			
value:	0	0	0	0				

3.3.6.24 T_Poll_Data.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		Polling Group	
				high	low
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
	1 1 1 1 0 0 0 0	XX	XX		

octet 7	octet 8		octet n+7
	Poll data 0	…	Poll data n-1
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	nr of slots			
value:	0	0	0	0				

3.3.7 Application Layer EMI

3.3.7.1 Definition

The full Application Layer EMI consists of the group-oriented Application Layer and the management parts.

The group-oriented Application Layer part consists of exactly the A_Data_Group.req, A_Data_Group.ind, A_Data_Group.con, A_Poll_Data.req and A_Poll_Data.con services.

The management part consists of exactly the M_Connect.ind, M_Disconnect.ind, M_User_Data_Connected.req, M_User_Data_Connected.ind, M_User_Data_Connected.con, M_User_Data_Individual.req, M_User_Data_Individual.ind and M_User_Data_Individual.con services.

3.3.7.2 M_Connect.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	unused	Source Address		unused	must be 00h
		high	low		
76543210	76543210	76543210	76543210	76543210	76543210
	XX			XX	00h

3.3.7.3 M_Disconnect.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	unused	unused	unused	unused	must be 00h
76543210	76543210	76543210	76543210	76543210	76543210
	XX	XX	XX	XX	00h

3.3.7.4 M_User_Data_Connected.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	unused
76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h	00h	00h

octet 7	octet 8	octet 9		octet n
	unused	…	APDU	…
76543210	76543210	76543210		76543210
	00000010			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value	0	0	0	0			0	0

  - **Priority (P)**
    definition: This shall be the value of the priority of the telegram that shall be transmitted to execute this M_User_Data_Connected.req.
    encoding: Value binary encoded.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	0							

3.3.7.5 M_User_Data_Connected.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	unused
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
		XX	XX	00h	00h

octet 7	octet 8	octet 9		octet n
	unused	…	APDU	…
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
	x x x x x x 1 0			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	Confirm
value	x	x	x	x	x	x	x	

  - **Confirm (C)**
    definition: This Confirm flag shall indicate whether this M_User_Data_-Connected.con is a positive confirmation or a negative confirmation.
    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value:								

3.3.7.6 M_User_Data_Connected.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		unused	unused
76543210	76543210	76543210	76543210	76543210	76543210
		XX	XX	00h	00h

octet 7	octet 8	octet 9		octet n
	unused	… APDU		…
76543210	76543210	76543210		76543210
	xxxxxx10			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value	x	x	x	x			x	x

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	x							

3.3.7.7 A_Data_Group.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	unused	SAP-Nr.
76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h	00h	

octet 7	octet 8	octet 9		octet n
	APCI	… data		…
76543210	76543210	76543210		76543210
	000000			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value:	0	0	0	0			0	0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	0							

3.3.7.8 A_Data_Group.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused		SAP-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	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
		XX	XX	XX	

octet 7	octet 8		octet 9		octet n
		APCI	…	data	…
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
	x x x x x x				

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	Confirm
value	x	x	x	x	x	x	x	

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value:								

3.3.7.9 A_Data_Group.ind message

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused		SAP-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	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
		XX	XX	XX	

octet 7	octet 8		octet 9		octet n
		APCI	…	data	…
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
	x x x x x x				

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	Confirm
value	x	x	x	x			x	

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	x							

3.3.7.10 M_User_Data_Individual.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	Destination Address	
				high	low
76543210	76543210	76543210	76543210	76543210	76543210
		00h	00h		

octet 7	octet 8	octet 9		octet n
	unused APCI	…	data	…
76543210	76543210	76543210		76543210
	00000010	11		

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:					Priority			
value:	0	0	0	0			0	0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	0							

3.3.7.11 M_User_Data_Individual.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	unused	unused	Destination Address	
				high	low
76543210	76543210	76543210	76543210	76543210	76543210
		XX	XX		

octet 7	octet 8	octet 9		octet n
	unused APCI	…	data	…
76543210	76543210	76543210		76543210
	xxxxxx10	11		

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	unused	unused	unused	Confirm
value	x	x	x	x	x	x	x	

  - **Confirm (C)**

    definition: This Confirm flag shall indicate whether this M_User_Data_Individual.con is a positive confirmation or a negative confirmation.

    encoding: 0: This shall be a positive confirmation.
    1: This shall be a negative confirmation.

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	information length			
value:								

3.3.7.12 M_User_Data_Individual.ind message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Source Address		Destination Address	
		high	low	high	low
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

octet 7	octet 8	octet 9		octet n
	unused APCI	… data		…
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
	0 0 0 0 0 0 1 0	1 1		

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	Priority		unused	unused
value:	x	x	x	x			x	x

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	x							

3.3.7.13 A_Poll_Data.req message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	unused	unused	Polling Group		
				high	low	
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	7 6 5 4 3 2 1 0
	1 1 1 1 0 0 0 0	00h	00h			

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	nr of slots			
value:	0	0	0	0				

3.3.7.14 A_Poll_Data.con message (EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6	octet 7
m_code	Control	Source Address		Polling Group		
				high	low	
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	7 6 5 4 3 2 1 0
	1 1 1 1 0 0 0 0	XX	XX			

octet 8				octet n+7
Poll Data 0		…		Poll Data n-1
7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	unused	unused	unused	nr of slots			
value:	x	x	x	x				

3.3.8 User Layer (i.e. Default) EMI

3.3.8.1 Definition

The user layer EMI consists of exactly the U_Value_Read.req, U_Value_Read.con, U_Value_Write.req, U_Flags_Read.req, U_Flags_Read.con, U_Event.ind, U_User_Data.req, U_User_Data.ind message, and M_InterfaceObj_Data services

3.3.8.2 U_Value_Read.req message (EMI1 / EMI2)

Message format

octet 1	octet 2
m_code	Group Object Number
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

3.3.8.3 U_Value_Read.con message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4		octet n
m_code	Group Object Number	RAM flags		value	
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

- **octet 3: RAM flags:**

bit nr:	7	6	5	4	3	2	1	0
name:								
value	x	x	x	x				

  - Bit 3: update
  - Bit 2: data request
  - Bit 1 to 0: transmission status

3.3.8.4 U_Flags_Read.req message (EMI1 / EMI2)

Message format

octet 1	octet 2
m_code	Group Object Number
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

3.3.8.5 U_Flags_Read.con message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5
m_code	Group Object Number	RAM flags	EEPROM flags	value type
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

- **octet 3: RAM flags:**

bit nr:	7	6	5	4	3	2	1	0
name:								
value	x	x	x	x				

  - Bit 3: update
  - Bit 2: data request
  - Bit 1 to 0: transmission status

- **octet 4: EEPROM flags:**

bit nr:	7	6	5	4	3	2	1	0
name:								
value	1							

  - Bit 6: transmit enable
  - Bit 5: memory type
  - Bit 4: write enable
  - Bit 3: read enable
  - Bit 2: communication enable
  - Bit 1 to 0: transmission priority

3.3.8.6 U_Event.ind message (EMI1 / EMI2)

Message format

octet 1	octet 2
m_code	cr_id
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

3.3.8.7 U_Value_Write.req message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4		octet n
m_code	Group Object Number	write mask RAM flags		value	
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

- **octet 3: RAM flags:**

bit nr:	7	6	5	4	3	2	1	0
name:	write mask				RAM flags			
value								

- Bit 7: value write enable
- Bit 6: update flag write enable
- Bit 5: data request flag write enable
- Bit 4: transmission status write enable
- Bit 3: update
- Bit 2: data request
- Bit 1 to 0: transmission status

3.3.8.8 U_User_Data.req / .con / .ind message (EMI1 / EMI2)

U_User_Data-services are used for the communication between an external and an internal user. A set of service codes is reserved for U_User_Data services (e.g. BCU 1 F0h to FFh, see [16]). The data contents can be defined by the application programmer.

Message format

octet 1	octet 2		octet n
m_code		data	
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0

3.3.8.9 M_InterfaceObj_Data.req / .con / .ind messages (EMI2)

The M_InterfaceObj_Data services are used in EMI2 to access user defined Interface Objects located in an external user. The services must be included in the application callback function, which detects if an Interface Object is located externally. If a Property-APCI is received (T_Data_Individual.ind or T_Data_Connected.ind) for an object located externally, an M_InterfaceObj_Data.ind must be generated and sent to the PEI by the callback function. The external user can send an M_InterfaceObj_Data.req, which is sent to the Management by the PEI.

M_InterfaceObj_Data.req

Message format

octet 1	octet 2	octet 3	octet 4	octet 5	octet 6
m_code	Control	Destination Address		CC	unused
		high	low		
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
					00h

octet 7	octet 8	octet 9		octet n
	unused APCI	…	data	…
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
	0 0 0 0 0 0			

- **octet 2: Control:**

bit nr:	7	6	5	4	3	2	1	0
name:					Priority			
value	0	0	0	0			0	0

- **octet 5: Communication Type:**
  encoding: = 00h: connection-oriented; do not use Destination Address
  ≠ 00h: connectionless; use Destination Address
- **octet 7:**

bit nr:	7	6	5	4	3	2	1	0
name:	unused	hop_count_type			information length			
value:	0							

3.3.9 Other default EMI services

3.3.9.1 Definition

These services are responsible for access to the internal memory by an external user (PC_ services), for PEI identification and for layer management.

3.3.9.2 PC_Set_Value.req message (EMI1 / EMI2)

Write data to memory locations in the bus device. The maximum amount of data is 15 octets.

Message format

octet 1	octet 2	octet 3	octet 4	octet 5		octet n
m_code	length	Address		data		
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

3.3.9.3 PC_Get_Value.req message (EMI1 / EMI2)

Read data from memory locations in the bus device. The maximum amount of data is 15 octets.

Message format

octet 1	octet 2	octet 3	octet 4
m_code	length	Address	
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

3.3.9.4 PC_Get_Value.con message (EMI1 / EMI2)

Message format

octet 1	octet 2	octet 3	octet 4	octet 5		octet n
m_code	length	Address		data		
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

3.3.9.5 PEI_Identify.req message (EMI2)

Message format

octet 1
m_code
7 6 5 4 3 2 1 0

3.3.9.6 PEI_Identify.con message (EMI2)

Message format

octet 1	octet 2	octet 3
m_code	Individual Address	
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

octet 4	octet 5	octet 6	octet 7	octet 8	octet 9
KNX Serial Number					
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

3.3.9.7 PEI_Switch.req message (EMI2)

The PEI_Switch.req service sets the static redirection table and the contents of the memory location "system status".

Message format

<table>
<tr><th>octet 1</th><th>octet 2</th><th colspan="2">octet 3</th><th colspan="2">octet 4</th><th colspan="2">octet 5</th><th colspan="2">octet 6</th><th colspan="2">octet 7</th></tr>
<tr><td rowspan="2">m_code</td><td rowspan="2">System Status</td><td colspan="10">new redirection for modules</td></tr>
<tr><td>LL</td><td>NL</td><td>TLG</td><td>TLC</td><td>TLL</td><td>AL</td><td>MAN</td><td>PEI</td><td>USR</td><td>res</td></tr>
<tr><td>7 6 5 4 3 2 1 0</td><td>7 6 5 4 3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td><td>7 6 5 4</td><td>3 2 1 0</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

For a detailed description of this service see 3.1.4 "Layer access in EMI2" of this document.

3.3.9.8 TM_Timer.ind message (EMI2)

A TM_Timer.ind message is generated when a user timer expires, which was started in the mode "message generation". The destination module ID is handed over when the timer is started by an API function.

3.3.9.8.1 TM_Timer.ind Style 1

Message format

octet 1	octet 2
m_code	timer number
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

3.3.9.8.2 TM_Timer.ind Style 2

Message format

octet 1	octet 2	octet 3
m_code	timer parameter	timer number
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

octet 4	octet 5	octet 6	octet 7	octet 8
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
00h	00h	00h	00h	00h

4 cEMI

4.1 cEMI: message format and services

4.1.1 Introduction

The cEMI message format is a generic structure for medium independent KNX messages, which can be added with information like a timestamp or other.

The cEMI message format claims to be independent from the frame structures of the different KNX media. Respectively, it claims to make possible transportation of all information of all the different KNX (medium dependent) frame formats.

4.1.2 Message flow – overview

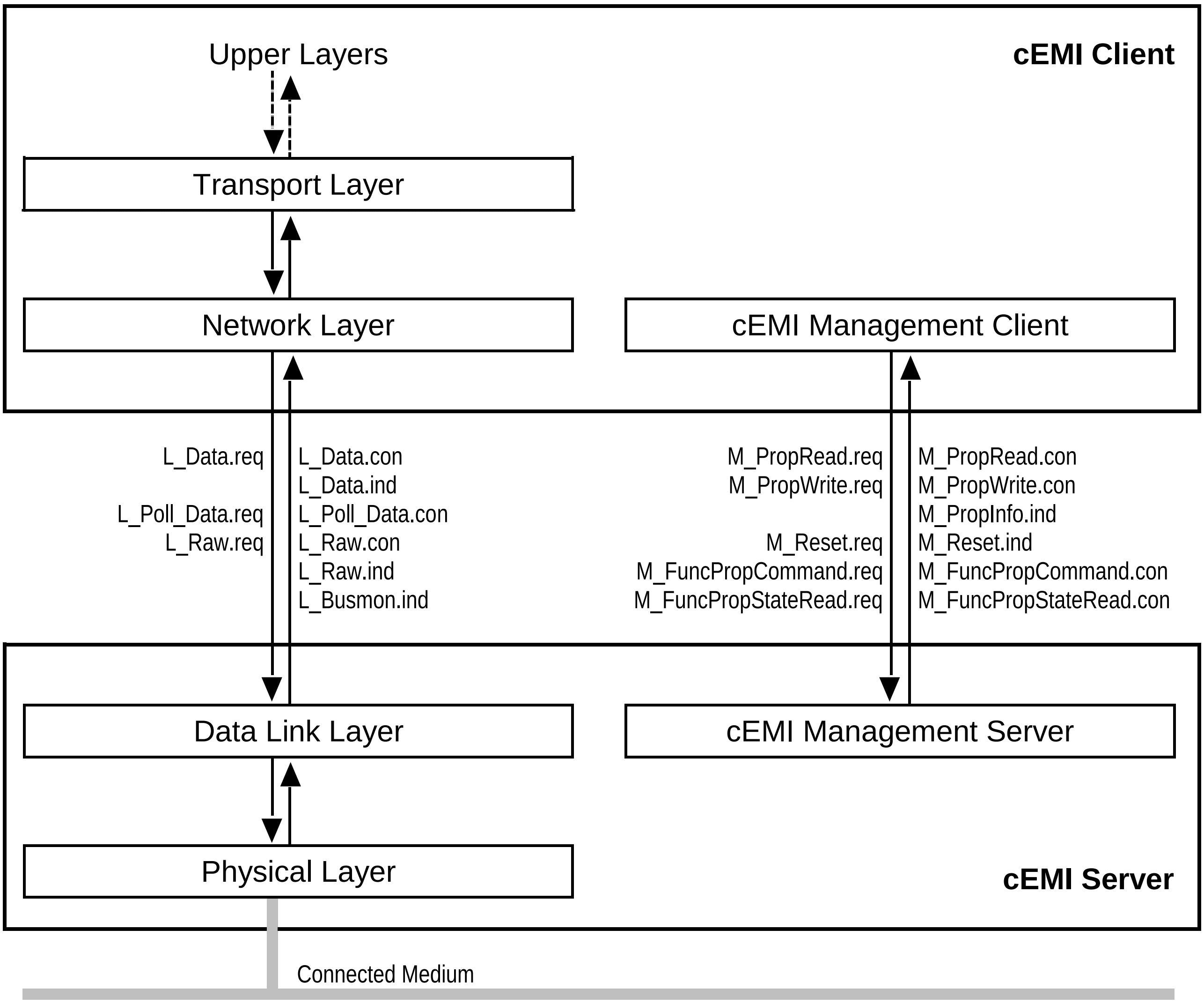


Figure 5 – Message flow between cEMI client and cEMI Server

4.1.3 Message Code and Message Code Set

4.1.3.1 Definitions

Each cEMI message shall start with the *Message Code* octet.

The cEMI *Message Code Set* covers all different message types as for Busmonitor communication mode, link & Transport Layer communication and for local device management communication.

4.1.3.2 Overview of the cEMI Message Codes

The codes for the Data Link Layer messages shall be the same as used for EMI 2. Please refer to Table 1 for the overview of the EMI message codes and the message code set supported by cEMI.

NOTE Codes for cEMI Transport Layer services are not yet listed in Table 1. If needed later (if the cEMI specification is extended with the corresponding definitions in clause 4.1.6, “Transport Layer messages”), the same codes as defined for EMI 2 shall be used for the Transport Layer services.

4.1.3.3 Exception handling: unknown messages

General behaviour after reception of an unknown or unsupported message (this is, the cEMI Server does not know the received cEMI message code): the received message shall be ignored, this is, the cEMI Server shall generate no confirmation message.

4.1.4 Basic message structure

4.1.4.1 Generic message structure

A cEMI message shall have the following generic structure:

Message Code	Additional Info Length	Additional Information	Service Information
MC	**AddIL**	**…**	**…**
1 octet	1 octet	var. length	var. length

MC: message code
AddIL: length of additional information

Services for management of the local device do not need any additional information. For this reason, the Additional Information field and the Additional Info Length (AddIL) field are not present in local device management services. For more details please refer to clause 4.1.7 “Services for Local device management” and clause 4.2 “Common EMI: Local Device Management”.

A cEMI management message shall have the following generic structure:

Message Code	Service Information
MC	**…**
1 octet	var. length

4.1.4.2 cEMI length information

As for EMI 1 and EMI 2 within the PEI protocols of a BCU 1 or BCU 2, a cEMI message shall be encapsulated in a message or frame structure of a host protocol. Currently known cEMI host protocols (KNX on USB, KNXnet/IP) represent the number of octets of the cEMI frame by a length information field within a header data structure (of the host protocol). The cEMI frame itself is typically the data field (or “body”), or a part of the data field within the host protocols frame structure. For details please refer to documents [11] and [15].

When further cEMI host protocols will be defined, these protocols shall also represent the cEMI message length by a length information field within their message or frame header’s data structure.

4.1.4.3 Additional information

4.1.4.3.1 Overview

The *Additional Information* field is intended for:

- medium dependent information, and
- other information: e.g. (relative) timestamp & error flags Busmonitor function.

The *Additional Information* field shall contain tagged information, identified by the *Type ID*:

Additional Information Type	Type ID	Length of Information	Information	Data Direction
	00h		reserved	
PL medium information	01h	2 octets	Domain Address used by PL medium	Client ↔ Server
RF medium information	02h	8 octets	RF-Info byte (formerly named RF-Ctrl) and KNX Serial Number/DoA and Data Link Layer Frame Number (LFN)	Client ↔ Server
Busmonitor – Status Info	03h	1 octet	Busmonitor Error Flags; see §4.1.5.7.6	Client ← Server
Timestamp relative	04h	2 octets	Relative timestamp; e.g. for L_Raw.ind	Client ← Server
Time delay until sending	05h	4 octets	Time delay (L_Raw.req, see §4.1.5.7.3)	Client → Server
Extended relative timestamp	06h	4 octets	Device independent time stamp, e.g. for L_Raw.ind or L_Busmon.ind, see 4.1.4.3.3	Client ← Server
BiBat information	07h	2 octets	Contains b7-b4 of the RF KNX-Ctrl field and BiBat Block-number	Client ↔ Server
RF Multi information	08h	4 octets	RF Multi frequency, call channel and Fast Ack number.	Client ↔ Server
Preamble and postamble	09h	3 octets	Preamble and postamble length	Client ↔ Server
RF Fast Ack information	0Ah	Variable (N*2 octets)	Status and information about each expected number of Fast Ack (N)	Client ↔ Server
Manufacturer specific data	FEh	Variable (N + 3 octets)	Manufacturer specific data, including manufacturer ID (2 octets) and Subfunction ID (1 octet).	Client ↔ Server
reserved	FFh	…	for future system extension (ESC Code)	

The structure of the *Additional Information* field is based on the approach to be open for future extensions; e.g. new types of Additional Information.

Combinations (concatenation) of Additional Information Types can be used in the additional information field.

Each Additional Information Type shall be accompanied by a length information giving the data length (number of octets) of the information type itself.

The *Additional Information Length* field shall be the number of octets of all additional information including the *Type IDs*, which shall be 1 octet long, and the type data length field(s), which shall be each 1 octet long, too.

If no *Additional Information* field is included in a cEMI message, then the value representing the *Additional Information Length* shall be set to zero. Value 255 is reserved for future extension.

Message structure including Additional Information:

Message Code	Additional Info Length	Additional Information								Service Information
MC	**AddIL**	**Type ID**	**Len**	**information**	**Type ID**	**Len**	**information**	**…**	**…**	**….**
1 octet	1 octet	1 octet	1 octet	dep. on inf. type	1 octet	1 octet	dep. on inf. type	…	…	var. length

|◄——————————————————►|

AddIL (= number of octets of *Additional Information*)

Concatenated *Additional Information* fields shall be sorted in ascending order of their *Type IDs*.

A receiver of a cEMI message including additional information but not interested on all or part of the information can ignore the whole or part of the additional information.

The different length information fields are intended to simplify evaluating the frame, e.g. to easily find the start of the *Service Information* field or to faster find the beginning of next *Additional Information* field.

4.1.4.3.2 AddInfoType 02h: RF medium information

'RF medium information' is mandatory for RF frames. It shall contain additional Data Link Layer information that is not present in the L_Data structure of TP and PL media.

The corresponding fields in the RF frame shall be encoded in the 'Info' (see note), 'SN' fields in the 1st RF Block and L/NPCI. LFN in the 2nd RF Block

Additional Information									
Type ID	**Len**	**RF–'Info'**	**RF-'SN' KNX Serial number/DoA**						**RF-'LFN'**
1 octet	1 octet	1 octet	6 octets						1 octet
02h	08h								

NOTE The 'Info' field in the 1st Block of the RF frame was formerly named RF-'Ctrl'.

Field	**Bit**	**Description**	**Coding**
RF-'Info'	bit 7 (msb)	reserved, shall be set to 0 in cEMI as well as by the sender of the RF message	
	bit 6	- This bit is reserved as 'confirm bit' in EMI only. - This bit is not used (void) in cEMI because in cEMI the 'C' flag in Ctrl1 field is used for confirmation. It shall be set to 0 in cEMI as well as by the sender of the RF message.	
	bits 5 … 4	- Shall be used in cEMI in L_Data.ind to indicate the received signal strength at the RF interface of the cEMI Server. - The value shall be void (00b) in cEMI L_Data.req and L_Data.con. The cEMI Server shall set the corresponding value to 00b in the transmitted RF message	00b: void (no measurement) 01b: weak 10b: medium 11b: strong

Field	Bit	Description	Coding
	bits 3 … 2	The corresponding field in the RF message shall contain the received signal strength filled in by the retransmitter with the lowest received signal strength. - L_Data.req: the cEMI Server shall insert the value in the RF message to be sent. Mandatory usage: - 00b for RF senders in normal operation mode - values ≠ 00b to be used only for test purpose to simulate the behaviour of a retransmitter - L_Data.con: dummy value in (the same value shall be used as in L_Data.req). The cEMI Client shall ignore the value. - L_Data.ind, shall contain the received value from RF message.	00b: void (no measurement) 01b: weak 10b: medium 11b: strong
	bit 1	Battery state of the RF sender - L_Data.req: the cEMI Server shall insert the value in the RF message to be sent. - Dummy value in L_Data.con (the same value shall be used as in L_Data.req). The value shall be ignored by the cEMI Client. - L_Data.ind, shall contains the received value from RF message.	0: battery is weak 1: battery is ok
	bit 0 (lsb)	Unidir flag to indicate if the RF message is sent by a bidirectional or transmit only device. - L_Data.req: the cEMI Server shall insert the value in the RF message to be sent. - L_Data.con: the same value shall be used as in L_Data.req. The value shall be ignored by the cEMI Client. - L_Data.ind, shall contain the received value from RF message.	0: frame sent by bidirectional device 1: frame sent by unidirectional device
RF-'SN'		KNX Serial Number or DoA: shall be interpreted as KNX Serial Number or DoA according to 'SB' flag in cEMI Control Field 1. - L_Data.req: - if SN ≠ 000000000000h the cEMI Server shall insert the value of the 'SN' field in the RF message to be sent - if SN = 000000000000h (void) the cEMI Server shall insert its own KNX Serial Number or DoA in the RF message to be sent - L_Data.con: shall contain the effective value of SN in the sent RF message. - L_Data.ind: contains the received value from RF message	000000000000h: void (no valid KNX Serial number or DoA)
RF-'LFN'		Data Link Layer frame number that shall be mapped to the corresponding field in L/NPCI.LFN (bit 3..1) of the RF frame - L_Data.req: - if LFN = 0…7: the cEMI Server shall insert the value of the 'LFN' field in the RF message to be sent. - if LFN = 8…254: L_Data.req shall be discarded. - if LFN = 255 (void): The cEMI Server shall insert its own local Data Link Layer frame number in the RF message to be sent. - L_Data.con: shall contain the effective value of LFN in the sent RF message. The value is normally ignored by the cEMI Client. - L_Data.ind: shall contain the received value from RF message.	0…7: valid LFN number 8…254: reserved, must not be used in cEMI 255: void

4.1.4.3.3 AddInfo-Type 06h: Extended relative timestamp

A cEMI Server can use the "Timestamp relative" (Additional Information Type ID: 04h) to deliver a timestamp in cEMI messages, typically used for bus monitoring (L_Busmon.ind / L_Raw.ind). However, this implementation of timestamp information has some drawbacks:

a. The timestamp information is given in "ticks" within 2 octets, providing a range of values from 1 to 65 535. Typically, the internal clock rate of modern cEMI devices is about 1 MHz to 10 MHz, and therefore a "tick" is usually about a few µs, sometimes even lower. This causes a counter overflow after a very short time (in the ms range).

   On the tool side (e.g. Busmonitor software on PC), this counter overflow has to be taken care of, resulting in complicated timing-procedures to synchronise the device-counter and the tool's clock, especially under non-real-time systems (e.g. Windows^{TM}).

   EXAMPLE cEMI Server with a clock rate of 1 MHz

   ⇒ 1 tick = 1 µs

   ⇒ Every 65 ms the internal counter has an overflow, causing the need for a fast and accurate timing on the tool side.

b. The timestamp information only delivers "ticks", which means the tool has to know the internal clock rate of the device to be able to calculate the relative time. This dependency makes it nearly impossible to develop tools being compatible with different cEMI device-types, as this information (clock rate) cannot be read out of the device.

For these reasons the Additional Information Type called "Extended relative timestamp" is defined:

Type ID (1 octet)	**Len (1 octet)**	**Timestamp (4 octets)**			
06h	04h				

The timestamp shall contain the 4 octet value of the free running counter of the cEMI Server at the time of frame reception. The value shall be measured always at the same position in the frame (e.g. beginning of first start bit) in order to allow the client the precise calculation of the time difference between successive frames.

A cEMI Server implementing the *Extended relative time stamp* shall provide the Property PID_TIME_BASE (PID = 55, PDT_UNSIGNED_INT) in the cEMI Server Interface Object containing the used time base. The time base shall be measured in nanoseconds per tick of the free running counter. The cEMI Client can read out this time base and display the relative time between frames in a device independent way (e.g. in µs).

EXAMPLE 1: cEMI Server with a clock rate of 1 MHz
1 tick = 1 µs ⇒ time base = 1000 (dec)
Overflow after 2^32 µs = 71,582 minutes, uncritical

EXAMPLE 2: cEMI Server with a clock rate of 8 MHz
1 tick = 1/8 µs ⇒ time base = 125 (dec)
Overflow after 8,94 minutes, uncritical

Please refer to the specification of PID_TIME_BASE in [07].

4.1.4.3.4 AddInfo-Type 07h: BiBat information

‘BiBat information’ shall be an addition to the ‘RF medium information’ and mandatory for RF BiBat frames. It shall contain additional Data Link Layer information, which is not present in the L_Data structure for asynchronous RF frames.

Additional Information			
Type ID	**Len**	**‘BiBat-Ctrl’**	**‘BiBat-Block’**
1 octet	1 octet	1 octet	1 octet
07h	02h		

Field	**Bit**	**Description**	**Coding**
‘BiBat-Ctrl’	bits 7 … 4	In the BiBat system these bits shall be used to encode BiBat specific Data Link Layer frames, see [01]. The corresponding field in the RF message shall be the upper 4 bits (b7-b4) of the KNX-Ctrl field in the 2nd RF Block. NOTE EFF information in the lower 4 bits (b3-b0) of the KNX-Ctrl field in the 2nd RF Block shall be mapped to Ctrl2-field (b3…b0); the handling shall be the same as for asynchronous RF Frames. In case of asynchronous RF frames b7…b4 in the KNX-Ctrl field shall have the fixed value 0000b, see 4.1.5.4. - L_Data.req: The cEMI Server shall copy the value of b7…b4 of ‘BiBat-Ctrl’ to b7…b4 of the KNX-Ctrl field of the RF message that shall be sent. - L_Data.con: shall contain the same value as in L_Data.req. - L_Data.ind: shall contain the received value of b7…b4 of the KNX-Ctrl field from BiBat RF message.	0000b: not allowed in ‘BiBat-Ctrl’ ⇒ asynchr. RF frame 0001b: Fast_ACK 0010b: reserved 0011b: reserved 0100b: synchronous L_Data frames 0101b: Sync frame 0110b: Help Call 0111b: Help Call Response 1xxxb: reserved
	bits 3 … 0	Reserved: the lower 4 bits of the ‘BiBat-Ctrl’ field shall be fixed (0000b). - L_Data.req: values ≠ 0000b shall be discarded by the cEMI Server - L_Data.con, L_Data.ind: values ≠ 0000b shall be discarded by the cEMI Client.	
‘BiBat-Block’		‘BiBat-Block’ shall not be contained in the RF frame but it shall be used to indicate the Block Number in the synchronous BiBat system (synchronisation of cEMI Server and client) - L_Data.req: shall contain the number of the next Block in which the synchronous frame shall be sent. - L_Data.con: shall contain the same value as in L_Data.req. - L_Data.ind: shall contain either the Block Number in which the synchronous frame is received (if known by the cEMI Server) or ‘void’ if unknown For some values of ‘BiBat-Ctrl’ the field ‘BiBat-Block’ is not meaningful and the value of ‘BiBat-Block’ shall be void.	0…127: valid Block number 128…254: reserved, must not be used in cEMI 255: void Block number

4.1.4.3.5 AddInfoType 08h: RF Multi information

4.1.4.3.5.1 Usage for KNX RF Ready and KNX RF Multi messages, message format

The “KNX RF Multi information” is optional for KNX RF Ready messages. If the cEMI message does not include the “KNX RF Multi information”, then the cEMI Server shall transmit the RF message with the parameters of a KNX RF Ready message; this is as follows.

- Default preamble of a KNX RF Ready Frame.
- Default KNX CTRL field.
- Default postamble of a KNX RF Ready Frame.
- Transmission on the F1 frequency.
- No Fast Ack management.

The “KNX RF Multi’ information” is mandatory for KNX RF Multi messages. The “KNX RF Multi information” can also be used to transmit a KNX RF Ready message by setting the frequency to the value 1: F1 (RF1.R).

Table 2 – KNX RF Multi information AddInfo

cEMI Additional Information					
Type ID	**Len**	**KNX RF Multi Transmission Frequency**	**KNX RF Multi Call Channel**	**KNX RF Multi Fast Ack**	**KNX RF Multi Reception Frequency**
1 octet	1 octet	1 octet	1 octet	1 octet	1 octet
08h	04h				

4.1.4.3.5.2 Transmission Frequency

If this AddInfo is not sent to the cEMI Server, the Transmission Frequency defined in the Property PID_TRANSMISSION_MODE shall be used. By default this shall have the value RF1.R.

If this AddInfo is used, after the Frame transmission, the Property PID_TRANSMISSION_MODE shall contain the real frequency used during transmission.

Table 3 – Transmission Frequency description

Field	Bit	Description	Coding
Transmission Frequency and transmission type		This octet shall be set in cEMI L_Data.req to indicate the Transmission Frequency (together with the preamble length and the KNX Ctrl Octet) on which the RF message shall be sent. - L_Data.req: - Previous: the cEMI Server uses the frequency defined in PID_TRANSMISSION_MODE Property. - F1 (RF1.R): the cEMI Server shall transmit the RF message in Ready mode. - Fx (RF1.M): the cEMI Server shall use one of the fast frequencies according to the RF media occupancy. The cEMI Server shall transmit the RF message in Multi-mode. - F1/F2/F3 (RF1.M): the cEMI Server shall transmit the RF message on the specified frequency. The cEMI Server shall transmit the RF	0: Use previous frequency 1: F1 Ready (RF1.R) 2: F1 (RF1.M) 3: F2 (RF1.M) 4: F3 (RF1.M) 5: S1 (RF1.M) 6: S2 (RF1.M) 7: Fx (RF1.M) 8: Sx (RF1.M) 9: Reserved for future use of the KNX system 10: F1 Ready (RF2.R) 11: F1 (RF2.M) 12: F2 (RF2.M) 13: F3 (RF2.M) 14: S1 (RF2.M) 15: S2 (RF2.M) 16: Fx (RF2.M) 17: Sx (RF2.M)

Field	Bit	Description	Coding
		message in Multi-mode. - Sx (RF1.M): the cEMI Server shall use one of the slow frequencies according to the RF media occupancy. The cEMI Server shall transmit the RF message in Multi-mode. - S1/S2 (RF1.M): the cEMI Server shall transmit the RF message on the specified frequency. The cEMI Server shall transmit the RF message in Multi-mode. - L_Data.con: depending on the capacity of the Physical Layer of the cEMI Server, it shall contain the same value as the L_Data.req or shall contain the real frequency used for transmission (in this case, value “Previous frequency”, Sx and Fx are not used). - L_Data.ind: shall contain the frequency value contained in the PID_TRANSMISSION_MODE Property.	18: reserved 19: F1 (RF5.M) 20: F2 (RF5.M) 21: F3 (RF5.M) 22: S1 (RF5.M) 23: S2 (RF5.M) 24: Fx (RF5.M) 25: Sx (RF5.M) 26: reserved 27 to 255: Reserved for future use of the KNX system

4.1.4.3.5.3 Fast and Slow Call Channel

The second octet of the cEMI Additional Information shall be used to set the RF Call Channel for the fast - and the slow frequencies.

If the first octet defines a frequency different than Sx and Fx, then the second octet shall define the RF call channel for next transmission request using Fx or Sx frequency.

If the first octet defines a frequency Sx or Fx, then the second octet shall define the RF call channel for current transmission request.

If this AddInfo is not sent to the cEMI Server, the RF Call Channel defined in the Property PID_RF_MULTI_CALL_CHANNEL shall be used. By default this shall have the value F1 and S1 (RF1.M).

If this AddInfo is used, the Property PID_RF_MULTI_CALL_CHANNEL shall be updated with the Call Channel specified in this AddInfo.

Table 4 – Call channels description

Field	Bit	Description	Coding
Fast Call Channel	bits 7 … 4	These bits shall be set in cEMI L_Data.req to indicate the Fast Call Channel that shall be used by the cEMI sever to transmit the RF message. - L_Data.req: - 0: the cEMI Server shall use the current Fast Call Channel, defined in the Property PID_RF_MULTI_CALL_-CHANNEL. - 1/2/3: the cEMI Server sets the current Fast Call Channel to F1/F2/F3. - L_Data.con: shall contain the actual value of the Fast Call Channel (except the "Current Fast Call Channel"). - L_Data.ind: shall contain the actual value of the Fast Call Channel (except the "Current Fast Call Channel").	0: set F1 as the current Fast Call channel 1: set F2 as the current Fast Call channel 2: set F3 as the current Fast Call channel 3 to 14: Reserved for future use of the KNX system 15: use current Fast Call Channel
Slow Call Channel	bits 3 … 0	These bits shall be set in cEMI L_Data.req to indicate the Slow Call Channel that shall be used by the cEMI sever to transmit the RF message. - L_Data.req: - 0: the cEMI Server shall use the current Slow Call Channel, defined in the Property PID_RF_MULTI_CALL_CHANNEL. - 1/2: the cEMI Server sets the current Slow Call Channel to S1/S2. - L_Data.con: shall contain the actual value of the Slow Call Channel (except the "Current Slow Call Channel"). - L_Data.ind: shall contain the actual value of the Slow Call Channel (except the "Current Slow Call Channel").	0: set S1 as the current Slow Call channel 1: set S2 as the current Slow Call channel 2 to 14: Reserved for future use of the KNX system 15: use current Slow Call Channel

4.1.4.3.5.4 Physical Acknowledge

If the cEMI Client requires that the cEMI Server transmits a KNX RF Multi message with the management of the physical acknowledge, then cEMI shall specify the number of expected Fast Acknowledges in the "KNX RF Multi information".

The third octet shall be meaningful only if the first octet of the current AddInfo defines the RF1.M frequency.

If this AddInfo is not sent to the cEMI Server, no Fast Ack management shall be used.

Table 5 – Physical Acknowledge description

Field	Bit	Description	Coding
Number of expected Fast Ack	bits 7 to 0	This octet shall indicate the number of expected Fast Ack for the RF message to transmit. The cEMI Server shall receive and process the expected Fast Ack. - L_Data.req: - 0: no Fast Ack management. - 1 to 64: number of expected Fast Ack. - 65 to 255: shall not be used. - L_Data.con: the same value shall be used as in L_Data.req. - L_Data.ind, L_Busmon.ind: shall contain the same value as indicated in the “Expected Number Fast Ack” field of the received Frame.	0: no Fast Ack management 1 to 64: number of expected Fast Ack 65 to 255: Reserved for future use of the KNX system

4.1.4.3.5.5 Reception frequency

After the cEMI Server has sent the Frame, it shall use the reception frequency octet to initialise the new reception frequency of the cEMI Server.

If this AddInfo is not sent to the cEMI Server, the reception frequency defined in the Property PID_RECEPTION_MODE shall be used after the Frame transmission. By default this shall have the value RF1.R.

If this AddInfo is used, after the Frame transmission, the Property PID_RECEPTION_MODE shall contain the reception frequency defined in the AddInfo.

Table 6 – Reception frequency description

Field	Bit	Description	Coding
Reception frequency		This octet shall be set in cEMI L_Data.req to indicate the reception frequency to set in the cEMI Server after the transmission of the Frame. - L_Data.req: - Previous: the cEMI Server uses the frequency defined in PID_RECEPTION_-MODE Property. - F1 (RF1.R): the cEMI Server shall scan the RF message in Ready mode. - F1/F2/F3 (RF1.M): the cEMI Server shall scan the RF message on the specified frequency. - S1/S2 (RF1.M): the cEMI Server shall scan the RF message on the specified frequency. - Fx (RF1.M): the cEMI Server shall scan the fast frequencies. - Sx (RF1.M): the cEMI Server shall scan the slow frequencies. - Fx and Sx (RF1.M): the cEMI Server shall scan the fast and the slow frequencies. - L_Data.con: shall contain the frequency value contained in the PID_RECEPTION_MODE Property. - L_Data.ind, shall contain the frequency value based on the reception frequency (“Previous frequency”, Fx and Sx not used, shall be the real RF channel) and the KNX CTRL field from RF message.	0: Use previous frequency 1: F1 Ready (RF1.R) 2: F1 (RF1.M) 3: F2 (RF1.M) 4: F3 (RF1.M) 5: S1 (RF1.M) 6: S2 (RF1.M) 7: Fx (RF1.M) 8: Sx (RF1.M) 9: Fx and Sx (RF1.M) 10: F1 Ready (RF2.R) 11: F1 (RF2.M) 12: F2 (RF2.M) 13: F3 (RF2.M) 14: S1 (RF2.M) 15: S2 (RF2.M) 16: Fx (RF2.M) 17: Sx (RF2.M) 18: Fx and Sx (RF2.M) 19: F1 (RF5.M) 20: F2 (RF5.M) 21: F3 (RF5.M) 22: S1 (RF5.M) 23: S2 (RF5.M) 24: Fx (RF5.M) 25: Sx (RF5.M) 26: Fx and Sx (RF5.M) 27 to 255: Reserved for future use of the KNX system

4.1.4.3.6 AddInfoType 09h: Preamble and postamble

This mandatory cEMI Additional Information shall be used for debugging and manufacturer specific purposes. The possible value range and the possibility to modify the preamble and postamble length are closely linked to the Physical Layer implementation in the USB RF interface.

Table 7 – Preamble and postamble AddInfo description

<table>
<tr><td colspan="5">cEMI Additional Information</td></tr>
<tr><th>Type ID</th><th>Len</th><th colspan="2">KNX RF Ready/Multi Preamble length</th><th>KNX RF Ready/Multi Postamble length</th></tr>
<tr><td>1 octet</td><td>1 octet</td><td colspan="2">2 octets</td><td>1 octet</td></tr>
<tr><td>09h</td><td>03h</td><td>High</td><td>low</td><td></td></tr>
</table>

Table 8 – Preamble and postamble description

Field	Bit	Description	Coding
Preamble Length	2 octets	These 2 octets shall indicate the number of “01” chip sequences in the preamble that the cEMI Server shall transmit for the RF Telegram. The most significant octet shall be transmitted firstly. - L_Data.req: - The value 0000h shall indicate that the cEMI Server shall use the default preamble length according to the type of the RF message (Ready, Multi fast or slow). - The possible value range depends on the capacity of the Physical Layer of the cEMI Server and the RF chip used for the USB interface. - L_Data.con: the same value shall be used as in L_Data.req. - L_Data.ind, L_Busmon.ind: shall contain the effective value of the number of “01” chip sequences detected in the RF message. The possible value range depends on the capacity of the Physical Layer of the cEMI Server and the RF chip used for the USB interface. If the preamble cannot be detected by the Physical Layer, the value 0000h shall be set.	0000h: default value xxxxh: number of “01” chip sequences in the preamble
Postamble Length	1 octet	This octet shall indicate the number of “01” chip sequences in the postamble that the cEMI Server shall transmit for the RF message. - L_Data.req: - The value 00h shall indicate that the cEMI Server shall use the default postamble length according to the type of the RF message (Ready, Multi fast or slow). - The possible value range depends on the capacity of the Physical Layer of the cEMI Server and the RF chip used for the USB interface. - L_Data.con: the same value shall be used as in L_Data.req. - L_Data.ind, L_Busmon.ind: shall contain the effective value of the number of “01” chip sequences postamble detected in the RF message. - The possible value range depends on the capacity of the Physical Layer of the cEMI Server and the RF chip used for the USB interface. If the postamble cannot be detected by the Physical Layer, the value 00h shall be set.	00h: default value xxh: number of “01” chip sequences in the postamble

4.1.4.3.7 AddInfoType 0Ah: RF Fast ack information

If an RF message is transmitted by the cEMI Server that requires the management of the Fast Ack with a specific number of expected Fast Ack, the Physical Layer may receive all the expected Fast Ack from the distant devices related to the transmitted RF message.

Symmetrically, if an RF message is received by the cEMI Server that requires the management of the Fast Ack with a specific number of expected Fast Ack, the Physical Layer may also receive the Fast Ack from the distant devices related to the received RF message.

In both cases, the Fast Ack shall be transferred by the cEMI Server to the cEMI Client for display with the “Fast Ack information”.

This cEMI Additional Information is mandatory to enable the cEMI Client to display the Fast Ack received and processed by the cEMI Server.

Table 9 – Fast Ack information AddInfo description

cEMI Additional Information						
Type ID	**Len**	**Fast Ack** **Fast Ack n°0**			**Fast Ack** **Fast Ack n°(N-1)**	
1 octet	1 octet	2 octets		….	2 octets	
0Ah	N*02h	Status	Info		Status	Info

The length of the ‘Fast Ack information’ shall depend on the number of expected Fast Ack as specified in the RF message. If the expected number of Fast Ack is N, then the length of the cEMI Additional Information shall be N*2 octets. The maximal number of Fast Ack that can be processed in a RF message is 64, thus the total length for the cEMI Additional Information is 128 octets.

Each Fast Ack field shall be composed of three octets.

Table 10 – Status and Info description

Field	Bit	Description	Coding
Status	bits 7 to 3	These bits shall be reserved and shall be set to 0b. - L_Data.req: not used. - L_Data.con: shall contain the value 00000b (case of emitted Frame). - L_Data.ind: shall contain the value 00000b (case of received Frame).	00000h
	bits 2	This bit shall indicate if the Fast Ack is received with a CRC error or not. - L_Data.req: not used. - L_Data.con: shall contain the value 0b or 1b (case of emitted Frame). - L_Data.ind: shall contain the value 0b or 1b (case of received Frame)	0b: Fast Ack received with no CRC error 1b: Fast Ack received with a CRC error
	bits 1	This bit shall indicate if the Fast Ack is received with a Manchester error or not. - L_Data.req: not used. - L_Data.con: shall contain the value 0b or 1b (case of emitted Frame). - L_Data.ind: shall contain the value 0b or 1b (case of received Frame).	0b: Fast Ack received with no Manchester error 1b: Fast Ack received with a Manchester error

Field	Bit	Description	Coding
	bits 0	This bit indicates if the Fast Ack has been received or not by the cEMI Server (e.g. the synchro pattern of the Fast Ack has been received or not). - L_Data.req: not used. - L_Data.con: shall contain the value 0b or 1b (case of emitted Frame). - L_Data.ind: shall contain the value 0b or 1b (case of received Frame).	0b: Fast Ack not received 1b: Fast Ack received
Info	1 octet	This octet shall contain the Fast Ack Info byte of the received Fast Ack. - L_Data.req: not used. - L_Data.con: shall contain the Fast Ack Info byte as contained in the received Fast Ack (case of emitted Frame). - L_Data.ind: shall contain the Fast Ack Info byte as contained in the received Fast Ack (case of received Frame).	0 to FFh: Fast Ack Info Octet

4.1.4.3.8 AddInfoType FEh: Manufacturer specific data

For debugging - and test purposes that are specific to manufacturer, a cEMI Additional Information ID is defined.

This ID shall be the following: FEh.

The content of data in the cEMI Additional Information shall only be interpreted together with the manufacturer ID.

The length of the data is variable according to the following format.

Table 11 – Manufacturer specific data AddInfo description

cEMI Additional Information					
Type ID	**Len**	**KNX Manufacturer ID**		**Manufacturer specific data (variable length)**	
1 octet	1 octet	2 octets		1 octet	N octets (N ≥ 0)
FEh	N+3h	High	Low	Subfunction	Data is interpreted by manufacturer and by subfunction

4.1.4.3.9 System broadcast and differentiation of SN / DoA information on RF

In cEMI frames, the AddrExtensionType information for the RF medium (SN or DoA) shall be mapped to the system broadcast bit (SB, bit 4 within the Ctrl1 octet); as specified in clause 4.1.5.3.2 “Basic frame structure for L_Data messages”. For more information please refer to [01].

The SB bit shall also be used for differentiation of system broadcast and broadcast within a domain on Powerline media.

4.1.4.3.10 Frame examples with indication of additional information

- *L_Data.req* message without any additional information:

Message Code	Additional Info Length	Service Information
MC	**AddIL**	**….**
11h	0	…

- *L_Data.con* message, from a Powerline medium frame, with Domain Address:

Message Code	Additional Info Length	Additional Information				Service Information
MC	**AddIL**	**Type ID**	**Len**	**information**		**….**
2Eh	4	01h	2	Domain	Address	…

- *L_Data.ind* message, RF frame with RF Control Information and KNX Serial Number (SN; SN_6 = MSB):

Message Code	Additional Info Length	Additional Information									Service Information
MC	**AddIL**	**Type ID**	**Len**	**information**							**….**
29h	9	02h	7	RF-Ctrl	SN_6	SN_5	SN_4	SN_3	SN_2	SN_1	…

- *L_Data.ind* message, RF frame with RF Control Information and RF Domain Address (DoA; DoA_6 = MSB):

Message Code	Additional Info Length	Additional Information									Service Information
MC	**AddIL**	**Type ID**	**Len**	**information**							**….**
29h	9	02h	7	RF-Ctrl	DoA_6	…	…	…	…	DoA_1	…

4.1.4.4 Service Information

Use of the *Service Information* is shown in the following clauses of this document.

4.1.5 Data Link Layer messages

4.1.5.1 Flow Control

cEMI Client

To keep the flow control for Data Link Layer services as simple as possible (this allows a simple flow control state machine in the cEMI client), it is recommended that:

- a cEMI client sends a new Data Link Layer request only when the confirmation of the preceding request is received, or
- a request-to-confirmation timeout is recognised; the recommended time-out for the cEMI client is 3 seconds.

A cEMI client shall at any time be able to accept an indication message from the cEMI Server.

Behaviour of the cEMI Server

A cEMI Server device shall have a receive buffer for one or more cEMI frames. The cEMI Server shall accept new frames from the cEMI client only if the receive buffer is not full. The request frames in the receive buffer shall be treated sequentially after the FIFO rule (first in, first out).

NOTE For cEMI Servers with a receive buffer for more than one frame, the order of received frames can be any concerning the type of received messages (Data Link Layer or Management).

During treatment of a request that is not yet confirmed to the cEMI client, the cEMI Server shall accept a new request from the cEMI client. This is used e.g. for management requests (see clause 4.1.7) during an L_Data.req/L_Data.con cycle.

4.1.5.2 General exception handling

4.1.5.2.1 Collisions

If during sending on the bus (on media with CSMA/CA method) the cEMI Server device detects a collision, it shall stop instantly its transmission.

The cEMI client shall not be informed about the detected collisions [5].

In case of collision the cEMI Server device shall (instead of complete transmission of its own send request) receive the message from the „winning” device from the bus. This shall lead to a corresponding Data Link Layer indication message from the cEMI Server to the cEMI client.

4.1.5.2.2 Physical Medium Error

In case of a physical medium error (transmission medium permanently disturbed, transmission not possible), the cEMI Server device is not able to send the request message on the bus. On level of the Data Link Layer message flow, the cEMI client shall recognise such an error by a confirmation timeout, i.e. no Data Link Layer confirmation message shall be sent from the cEMI Server to the cEMI client.

4.1.5.2.3 Use of Frame Type flag (FT) and Extende Frame Format (EFF) field

As already specified in clause 2.2.4 in [02], any receiver shall be tolerant towards the use of the Frame Format. It shall accept the use of an Extended Frame even if the size of the payload would allow a Standard Frame. Therefore, it a cEMI Server receives a cEMI Frame with FT = 0 and EFF = 0, thus denoting a Standard Frame for an APDU > 15 octets, yet with a payload shorter then 15 octets, then the cEMI Server may:

- either correct the situation and transmit an L_Data_Standard Frame, or
- transmit an L_Data_Extended Frame with payload smaller than 15 octets.

4.1.5.3 L_Data services

4.1.5.3.1 Implementation and usage

The L_Data services within the Data Link Layer Interface are mandatory for a cEMI Server.

4.1.5.3.2 Basic frame structure for L_Data messages

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length

- MC: message code
- AddIL: length of additional information

[5] Implementations with TP-UART: TP-UART does not support such a feature.

- Ctrl1: Control field 1

1 octet							
Ctrl1							
7	6	5	4	3	2	1	0
FT	0	R	SB	P		A	C

  - Frame Type flag (FT) (msb):

    description: This shall specify the Frame Type that shall be used for transmission or reception of the frame. See [02].
    encoding: 0: extended frame
    1: standard frame

  - Repeat flag (R) (bit 5):

    description: Repeat, not valid for all media.
    encoding: 0: repeat frame on medium if error
    1: do not repeat

  - System Broadcast flag (SB) (bit 4):

    description: This shall specify whether the frame is transmitted using system broadcast communication mode or broadcast communication mode (applicable only on open media); see 4.1.4.3.5.
    encoding: 0: system broadcast
    1: broadcast

  - Priority (P) (bit 3 and bit 2):

    description: This shall specify that Priority that shall be used for transmission or reception of the frame.
    encoding: Please refer to “Usage of priority” in [02].

  - Acknowledge request flag (A) (bit 1):

    description: This shall specify whether a L2-acknowledge shall be requested for the L_Data.req frame or not. This is not valid for all media.
    encoding: 0: no acknowledge is requested
    1: acknowledge requested

  - Confirm flag (C) (lsb)

    description: In L_Data.con this shall indicate whether there has been any error in the transmitted frame.
    encoding: 0: no error
    1: error

- Ctrl2: Control field 2

1 octet							
Ctrl2							
7	6	5	4	3	2	1	0
AT	HC			EFF			

  - Destination Address Type(AT) (msb):

    encoding: 0: individual
    1: group

  - Hop Count (HC) (bit 6 to bit 4):

    encoding: value binary encoded

  - Extended Frame Format (EFF) (bit 3 to bit 0 (lsb)):

    encoding: 0000b: for standard frame (long frames, APDU > 15 octet)
    01xxb: for LTE frames

- SAH: Source Address High (Source Subnetwork Address)
- SAL: Source Address Low (Source Device Address)
- DAH: Destination Address High (Destination Subnetwork Address)
- DAL: Destination Address Low (Destination Device Address)
- L: Information-Length (max. value is 255); number of NPDU octets, TPCI octet not included!
  → L = number of octets (without FCS), counting starts with the octet after the TPCI octet
  (0 = no octet after the TPCI)

NOTE 1 The *Info-Length* (L) is the value used in the length field (NPDU-Length, excl. TPCI octet) in the frame structures on KNX twisted pair (TP 0, TP 1) and Powerline media (PL110). This length is the parameter used in the upper layers (NL, TL) of the KNX communication stack model as the octet_count (in NL and TL specification, see [03] and [04]).
A KNX RF frame does not contain any NPDU length as an information field in the RF frame structure. Instead, a RF frames contain a length information field quite at the beginning of the RF frame. Its value includes more than the NPDU-Length. For more details please refer to [01].
If a cEMI frame is mapped to a RF frame (or vice versa), the L field is 'void'. The value shall be fixed to '00h'. Mapping of cEMI to/from RF media frame format is indicated by presence of *RF Control* field in the additional information part of cEMI frame.

NOTE 2 Source Address: it is imaginable for the implementation of any future tool/interface device combinations, that the interface device is fully transparent on Data Link Layer level. That means, e.g. a PC tool could hold the KNX Individual Address itself. In such a case, the interface device doesn't absolutely need an Individual Address itself. For this reason, the cEMI L_Data messages contain a field for the Source Address in all L_Data Services. Use of the Source Address field in L_Data.req frame: see 4.1.5.3.3.

NOTE 3 A combination as mentioned above probably is not applicable on a medium with a time critical Data Link Layer acknowledge mechanism, as it is used e.g. on the KNX TP medium.

4.1.5.3.3 L_Data.req

Message Code	Additional Information Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length
11h	…	…	x0rxppa0	…	…		…		…	…

If a cEMI Server receives the L_Data.req message with *Source Address* set to 0000h, then the cEMI Server shall fill in the Source Address field before sending the message onto the KNX network. Typically, this is the cEMI Server device's own Individual Address.

If the field *Source Address* is not set to 0000h, the cEMI Server shall send the frame onto the KNX network with the Source Address received from the cEMI client with L_Data.ind message.
On KNX media with Data Link Layer acknowledge (like TP1), this feature is only possible if the cEMI Server device supports more than one Individual Address for sending L2-acknowledges onto the bus in case of received frames (confirmed services, responses in point-to-point connectionless or connection-oriented communication mode to requests in point-to-point connectionless or connection-oriented communication mode). If the cEMI Server device does not support multiple IAs, the cEMI Server shall always insert it's own Individual Address in the Source Address field. This specification is limited to cEMI Server implementations in devices supporting only a single Individual Address.

A cEMI Server in full transparent mode, this is a cEMI Server without an own Individual Address, shall send back a negative confirmation (*Confirm* Flag set to 1 in *L_Data.con*, see clause 4.1.5.3.4 "L_Data.con") if a *L_Data.req* message is received from cEMI client with *Source Address* set to 0000h.

Use of flags in Ctrl1-field

- Frame Type (FT)

description: This field shall specify whether the frame is a standard frame or an extended frame.

encoding:

Value of FT-Flag	Meaning / behaviour on bus media		
	TP1	PL110	RF [a]
0	Extended	Extended	Extended
1	Standard	Standard	Extended

[a] Bit value is "Don't care" if cEMI Server is interface to RF, since RF uses only extended frames.

- Repetition (r)

description: This shall specify whether repetitions shall be sent on the medium. This flag is relevant only on media with possibility of Data Link Layer controlled frame repetitions (TP1, PL110);

NOTE: "Don't care" means that (LL-) repetitions shall be sent (if error; e.g. on TP1: if no ACK-, NACK- or BUSY-frame) according the default media's repetition behaviour, i.e. according the value of the Property PID_MAX_RETRY_COUNT.

encoding: 0: do not repeat if error

1: Don't care

Value of r-Flag	Meaning / behaviour on bus media		
	TP1 [a]	PL110	RF
0: no repetitions	No repetitions	No repetitions	No repetitions
1: Don't care	Repetitions are allowed	Repetitions are allowed	No repetitions

[a] Existing implementations, particularly implementations based on BCU 1 / 2 technology, may ignore the r-Flag.

- System Broadcast (SB)

description: This flag shall only be applicable on open media.
It shall be don't care on "closed" media (e.g. TP1), i.e. a cEMI Server to a closed medum shall ignore the SB-flag.

encoding: 0: system broadcast
1: broadcast

- Acknowledge request (A)

description: This shall specify whether a L2-acknowledge shall be requested for the L_Data.req frame or not. This is not valid for all media.
"Don't care" means that no explicit L2-acknowledge is requested by the upper layer(s); this means the default behaviour of the Data Link Layer concerning L2-acknowledge requesting applies, as laid down in the specifications of the communication medium.

encoding: 0: no acknowledge is requested
1: acknowledge requested

Value of a-Flag	Meaning / behaviour on bus media		
	TP1	PL110	RF
0: Don't care	Requested	Requested	Not applicable
1: Ack requested	Requested	Requested	Not applicable

- C

  description: The C-flag shall not be used in the L_data.req. It shall be “don’t care”, i.e. a cEMI Server shall ignore the C-Flag in L_Data.req.

4.1.5.3.4 L_Data.con

Message Code	Additional Information Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length
2Eh	…	…	x0rxppxC	…	…		…		…	…

C: Confirm Flag: 1 = Error; 0 = No Error

The *L_Data.con* shall be a “local” primitive generated by the cEMI Server’s Data Link Layer for its own cEMI client to indicate that it is satisfied with the transmission (error flag C cleared) or not (error flag C set).

If a message is sent onto a medium with immediate acknowledge (L2-acknowledge) the confirmation message is normally generated after receiving this immediate acknowledge.

The *Source Address* field shall be used to indicate the Source Address of the requested message. The *Destination Address* field shall be used to indicate the Destination Address of the requested message; the *Destination Address Type* (AT-bit, Individual or Group) shall be the Destination Address Type of the requested message.

Use of flags in Ctrl1-field

- Frame Type (FT)

  description: The cEMI Server shall set the flag to the value according the frame type that is sent onto the bus.

  encoding: FT = 0: extended frame
  FT = 1: standard frame

- Repeat flag (R) (bit 5):

  description: Don’t care: the flag can have the same value as in the original L_Data.req, but it shall not be interpreted by the cEMI Client.

- SB

  description: Don’t care: the flag can have the same value as in the original L_Data.req, but it shall not be interpreted by the cEMI Client.

- a

  description: Don’t care: the flag can have the same value as in the original L_Data.req, but it shall not be interpreted by the cEMI Client.

  encoding:

- Confirm (C)

  encoding: 0 = No Error
  1 = Error

4.1.5.3.5 L_Data.ind

Message Code	Additional Information Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length
29h	…	…	x0rxppxx	…	…		…		…	…

The *Source Address* field shall contain the Source Address that is received by the cEMI Server from the bus in the originating medium specific frame.

Use of flags in Ctrl1-field

- Frame Type (FT)

  description: The cEMI Server shall set the flag to the value according the frame type that is received from the bus.

  encoding: FT = 0: extended frame
  FT = 1: standard frame

- Repeat flag (R) (bit 5):

  description: If the cEMI Server receives a repeated frame from the bus and if it also receives the originating frame (and acknowledged it with the LL-IAck) then the cEMI Server shall not indicate the repeated frame to the cEMI client.

  encoding: 0: repeated L_Data frame on media
  1: not repeated frame on media

- System Broadcast (SB)

  description: This field shall be applicable only on open media; it shall be don’t care on (“closed”) media (e.g. TP1) , i.e. a cEMI client shall ignore the SB-flag if received from a cEMI Server as interface from a closed media.

  encoding: SB = 0: system broadcast
  SB = 1: broadcast

- a

  description:
  – TP1, PL110, RF:
  The a-flag shall not be used; it shall be “don’t care”; this is, the cEMI client shall ignore the a-flag.
  – TP0, PL132:
  The cEMI Server shall set the a-flag to the value received from the bus.

  NOTE The specification of L_Data.ind in [02], which is referred to by both the TP0 as well as the PL132 specifications, does foresee the ack_request service parameter (though without functionality in the upper layers).

  encoding:

- Confirm (C)

  description: The C-flag shall in the L_Data.ind be “don’t care”; the C-flag does not exist; i.e. a cEMI client shall ignore the C-flag in L_Data.ind

  encoding:

4.1.5.4 L_Data services for KNX RF asynchronous frames

4.1.5.4.1 L_Data.req

cEMI L_Data.req shall be mapped by the cEMI Server to the corresponding RF Data Link Layer frame as follows:

RF Data Link Layer frame			Mapping in cEMI L_Data.req
Block	**Field**	**Bit**	
1st Block	Length		Shall not be contained in the cEMI message because the RF frame format does not contain a NPDU length. The corresponding ‘L’ field in the cEMI L_Data structure shall be void and the cEMI Client shall insert the value 00h. The ‘L’ field shall be ignored by the cEMI Server
	C-field		Shall not be contained in the cEMI message because the ‘C-field’ shall have a fixed value (44h) on KNX RF. The cEMI Server shall generate the value of the ‘C-field’ locally.
	ESC		Shall not be contained in the cEMI message because the ‘ESC’ field shall have a fixed value (FFh) on KNX RF. The cEMI Server shall generate the value of the ‘ESC’ field locally.
	Info (RF-‘Ctrl’)		Shall be mapped to the Additional Info Type ‘RF medium information’; for further details see 4.1.4.3.2.
	SN		Shall be mapped to the Additional Info Type ‘RF medium information’; for further details see 4.1.4.3.2
	CRC		Shall not be contained in the cEMI message. The cEMI Server shall generate the value of the ‘CRC’ locally.
2nd Block	KNX-Ctrl	EFF (bits 3 … 0)	Shall be mapped to the Extended Frame Format in Control field Ctrl2.b3-b0.
		reserved 0000b (bits 7 … 4)	Shall not be contained in the cEMI message because for asynchronous RF frames the upper 4 bits of the KNX-Ctrl field shall be fixed (0000b). The cEMI Server shall insert the fixed value 0000b locally in the RF message to be sent.
	SRC		Shall be mapped to the Source Individual Address SA field in the cEMI message. Same special handling of SA = 0000h as for other media.
	Dest		Shall be mapped to the Destination Address DA field in the cEMI message. Same handling of DA as for other media.
	L/NPCI	address type (bit 7)	Shall be mapped to the Address Type AT flag in Control field Ctrl2.b7. Same handling as for other media.
		routing counter (bits 6 … 4)	Shall be mapped to the Hop Count in Control field Ctrl2.b6-4 Same handling as for other media.
		LFN (bits 3 … 1)	Shall be mapped to the Additional Info Type ‘RF medium information’; for further details see 4.1.4.3.2.
		AddrExtensionType (bit 0)	Shall be mapped to the System Broadcast ‘SB’ flag in cEMI Control Field 1 Ctrl1.b4. According to ‘SB’ the value of the ‘SN’ field in the Additional Info Type ‘RF medium information’ shall be interpreted as either the KNX Serial Number or the Domain Address
	TPCI/APCI		TPCI/APCI
	Data		Data
	CRC		Same handling as for CRC in 1st Block

RF Data Link Layer frame			Mapping in cEMI L_Data.req
Block	**Field**	**Bit**	
3rd Block	Data		Data
	CRC		Same handling as for CRC in 1st Block

The following fields in the Ctrl1 field of cEMI L_Data.req frame shall be void for RF messages:

FT: frame type: (standard/extended) Shall be ignored by the cEMI Server. In RF frames the EFF field shall always be present.
⇒ Extended Frame Format on RF.

r repeat flag: Shall be ignored by the cEMI Server.
⇒ No repetitions on RF.

p priority: Shall be ignored by the cEMI Server.
⇒ Specific medium access mechanisms on RF, no CSMA/CA.
⇒ RF frame does not contain priority information.

a acknowledge request Shall be ignored by the cEMI Server.
⇒ Special acknowledge solution for RF is introduced in the BiBat system with specific encoding in cEMI.

C Confirm: Shall be void in L_Data.req, shall be ignored by the cEMI Server.

Incorrect L_Data frames shall be discarded by the cEMI Server.

4.1.5.4.2 L_Data.con

In L_Data.con the Additional Info Type ‘RF medium information’ shall be included as specified in 4.1.4.3.2. Moreover the mapping rules as specified for L_Data.req in 4.1.5.4.1 apply.

SA (source Address) and DA (Destination Address) shall be used as specified in 4.1.5.3.4. Serial Number / Domain Address shall be used as specified in 4.1.5.4.1.

The fields in the Ctrl1 field of cEMI L_Data.con shall be used as follows.

FT: Frame Type = 0 (extended)

r Repeat Flag: This field shall be don’t’ care; it is recommended to use the same value as in the original L_Data.req. The cEMI Client shall ignore the value.

SB System Broadcast: The same value as in L_Data.req shall be used.

p Priority: Shall be don’t’ care; it is recommended to use the same value as in the original L_Data.req. The cEMI Client shall ignore the value.

a Acknowledge request: Shall be don’t’ care; it is recommended to use the same value as in the original L_Data.req. The cEMI Client shall ignore the value.

C Confirm: 0: No Error
1: Error

L_Data.con with C = 0 (No Error) shall be generated after transmission of the corresponding RF frame.

In practice L_Data.con with C = 1 (Error) will never be generated due to transmission errors on the RF medium because this can not be detected by the sender on RF.

L_Data.con with C = 1 shall only generated in case of

- the sending frame buffer in cEMI Server is exceeded, or
- other error cases (to be defined product specific).

4.1.5.4.3 L_Data.ind

A received RF Data Link Layer frame shall be checked by the cEMI Server and be mapped to the corresponding L_Data.ind.

The cEMI Server shall discard incorrect received RF frames.

In L_Data.ind the Additional Info Type 'RF medium information' shall be included as specified in 4.1.4.3.2.

RF Data Link Layer Frame			**Mapping in cEMI L_Data.ind**
Block	**Field**	**Bit**	
1st Block	Length		Shall not be contained in the cEMI message. The corresponding 'L' field in the cEMI L_Data structure shall be void and the cEMI Server shall insert the value 00h. The 'L' field shall be ignored by the cEMI Client.
	C-field		Shall not be contained in the cEMI message, see 4.1.5.4.1.
	ESC		Shall not be contained in the cEMI message, see 4.1.5.4.1.
	Info (RF-'Ctrl')		Shall be mapped to the Additional Info Type 'RF medium information'; for further details see 4.1.4.3.2.
	SN		Shall be mapped to the Additional Info Type 'RF medium information'; for further details see 4.1.4.3.2.
	CRC		Shall not be contained in the cEMI message. The cEMI Server shall check the value of the 'CRC' locally and discard the RF frame in case of CRC error.
2nd Block	KNX-Ctrl	EFF (bits 3 … 0)	Extended Frame Format in Control field Ctrl2.b3-b0.
		reserved 0000b (bits 7 … 4)	Shall not be contained in the cEMI message because for asynchronous RF frames the upper 4 bits of the KNX-Ctrl field are fixed (0000b).
	SRC		Shall be copied to the Source Individual Address SA field in the cEMI message.
	Dest		Shall be copied to the Destination Address DA field in the cEMI message.
	L/NPCI	address type (bit 7)	Address Type AT flag in Control field Ctrl2.b7 Same handling as for other media.
		routing counter (bits 6 … 4)	Hop Count in Control field Ctrl2.b6-4 Same handling as for other media.
		LFN (bits 3 … 1)	Shall be mapped to the Additional Info Type 'RF medium information'; for further details see 4.1.4.3.2.
		AddrExtensionType (bit 0)	Shall be mapped to System Broadcast 'SB' flag in cEMI Control Field 1 Ctrl1.b4., see also 4.1.5.4.1.
	TPCI/APCI		TPCI/APCI fields.
	Data		Data fields.
	CRC		Same handling as for CRC in 1st Block.
3rd Block	Data		Data.
	CRC		Same handling as for CRC in 1st Block.

The fields in the Ctrl1 field of cEMI L_Data.ind shall be used as follows:

FT:	Frame Type	= 0 (extended)
R	Repeat flag	Shall be don't' care (1). The cEMI Client shall ignore the value.
SB	System Broadcast	Shall be set according to AddrExtensionType in RF frame, see above.
P	Priority	Shall be don't' care (11b). The cEMI Client shall ignore the value.
A	Acknowledge request	Shall be don't' care (0). The cEMI Client shall ignore the value.
C		Confirm: don't' care (0). The cEMI Client shall ignore the value.

4.1.5.5 Mapping of L_Data Services for BiBat RF frames

4.1.5.5.1 Introduction

In cEMI, the L_Data service shall be used to encode synchronous RF Data Frames as well as BiBat specific Data Link Layer frames such as:

- Sync frame
- Help Call / Help Call Response
- Fast_ACK frame

Synchronous RF data frames as well as BiBat specific Data Link Layer Frames shall be ignored by the cEMI Server if it works in asynchronous mode.

4.1.5.5.2 L_Data.req for BiBat synchronous data frames

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in L_Data.req.

If the cEMI Server supports BiBat Master functionality, it shall handle L_Data.req for synchronous data frames as specified below. If it does not support BiBat Master functionality, L_Data.req shall be discarded for synchronous data frames.

Example:

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SA L**	**DA H**	**DA L**	**L**	TPCI/APCI & data
11h	14d	…	…	…	…	…		…		0	…

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA, L, TPCI/APCI and data shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames.

'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

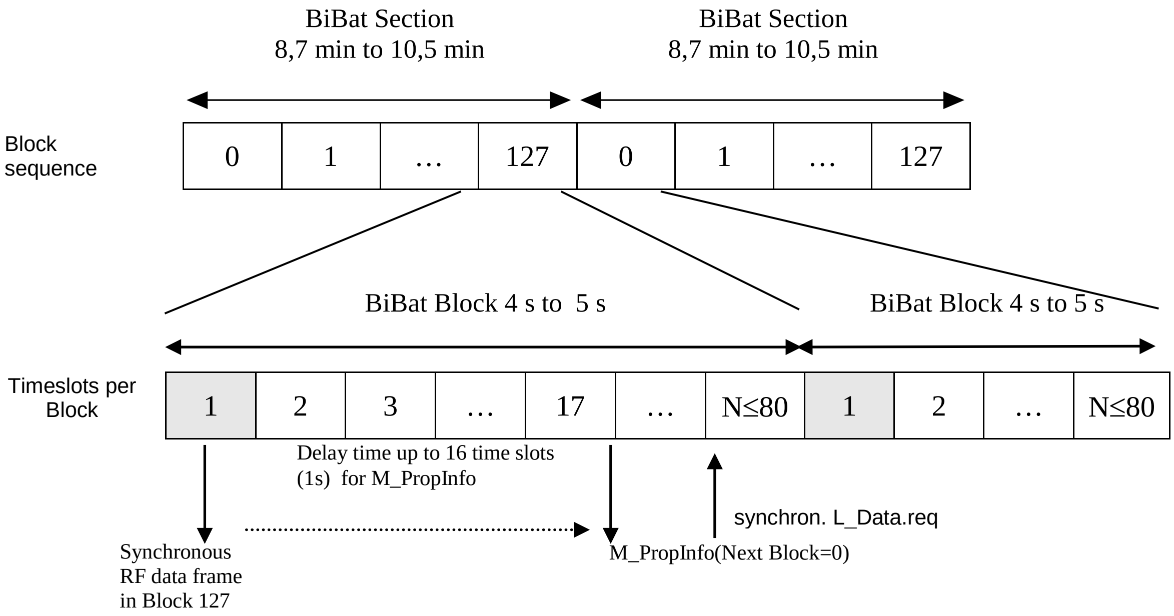
- 'BiBat-Ctrl' shall have the value 01000000b (synchronous data frame).
  In the 'KNX-Ctrl' field of the corresponding RF frame
  - bits b_7 (msb) to b_4 shall have the value 0100b
  - bits b_3 to b_0 (lsb) shall be set according the Frame Type.
- 'BiBat-Block' shall contain a valid Block number 0 … 127 that indicates in which Block in the time-slotted BiBat scheme the data frame shall be sent.

Normally the cEMI Server/BiBat Master will only be able to handle L_Data.req for the next Block (no ahead buffering of synchronous frames). The 'BiBat-Block' number shall be used to check block number synchronisation of cEMI Server and client.

If the cEMI Server is not able to handle the requested block number, the L_Data.req shall be discarded and L_Data.con with the field C = 1 shall be generated as negative confirmation.

The cEMI Server shall inform the cEMI Client about the next block number at least 3 s in advance using the M_PropInfo-service. See also PID_BIBAT_NEXTBLOCK in [07].

EXAMPLE



Duration of one BiBat Block: 4 s to 5 s (pseudo random sequence).

Each Block consists of 64 to 80 timeslots. Duration of one timeslot: 62,5 ms

The synchronous data frame shall be sent in the first timeslot of each block.

After completion of the synchronous transmission, the cEMI Server shall generate an M_PropInfo message containing the number of the next block, see also PID_BIBAT_NEXTBLOCK in [07]. M_ProfInfo shall be generated before timeslot 18 in order to guarantee ~3 s reaction time for the cEMI Client to prepare the synchronous L_Data.req for the next block.

NOTE L_Data.req for asynchronous frames my occur in parallel at any time, independent of BiBat communication.

Concatenated L_Data frames in one Block:

The BiBat system shall support transmission of up to three subsequent data frames in one Block (subsequent telegrams within a communication block start directly after the preceding telegram with the preamble of the following frame).

Therefore the cEMI Server shall accept up to three subsequent synchronous L_Data.req with the same Block number and transmit them in the first timeslot of the next Block.

If the cEMI client sends more than three subsequent synchronous L_Data.req only the first three shall be accepted and the following shall be discarded with a negative confirmation in L_Data.con (C = 1).

If the cEMI client sends multiple requests with different Block number, the cEMI Server will discard all requests and generate a negative confirmation in L_Data.con (C = 1).

NOTE There is no mechanism to cancel or overwrite separate pending L_Data frames in the cEMI Server.

4.1.5.5.3 L_Data.con for BiBat synchronous data frames

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in L_Data.con. The same frame structure shall be used as specified in 4.1.5.5.2, with message code = 2Eh.

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA, L, TPCI/APCI and data shall be encoded as specified in 4.1.5.4.2 for asynchronous data frames.

'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- 'BiBat-Ctrl' shall have the value 01000000b (synchronous data frame)

'KNX-Ctrl' in the corresponding RF frame shall have the value 0100xxxxb (xxxx according to EFF).

- 'BiBat-Block' shall have the same Block number as in L_Data.req

L_Data.con shall only be generated after completed transmission of the synchronous frame in the next Block. Therefore there is a time delay in the range of up to 5 s between synchronous L_Data.req and L_Data.con.

Concatenated L_Data frames in one Block

In case of multiple (up to three) subsequent synchronous data frames in one Block, for each L_Data.req the corresponding L_Data.con shall be generated.

Therefore the cEMI Client shall be able to handle up to three subsequent synchronous L_Data.con with the same Block number.

If the cEMI Server sends more than three subsequent BiBat L_Data.con or confirmation with different or mismatching Block number, the cEMI Client shall discard all confirmations.

L_Data.con with C = 1 shall only be generated if

- the sending frame buffer in cEMI Client is exceeded, or
- there is a mismatching Block number in L_Data.req, see above, or
- there are more than three concatenated L_Data.req, see above, or
- the requested synchronous transmission is not possible due to a pending transmission of a frame with higher priority (e.g. Help Call Response, "stolen" Block for Fast_Ack mechanism, see [01]).

4.1.5.5.4 L_Data.ind for BiBat synchronous data frames

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in L_Data.ind.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	TPCI/APCI & data
29h	14d	…	…	…	…	…		…		0	…

As a receiver of synchronous frames the cEMI Server shall work in BiBat Slave mode. The cEMI Server shall support BiBat synchronisation mechanisms with the BiBat Master.

Handling of BiBat Block number

- If the cEMI Server works as a BiBat Slave that *is not* yet synchronized to the Master, then it shall have no knowledge about the current BiBat timeslot structure. In this case the field 'BiBat-Block' shall be void (255).
- If the cEMI Server works as BiBat Slave that *is* synchronized to the Master, then the receiver in the cEMI Server shall always be activated (for asynchronous frames).
  Received synchronous data frames that fit to the timeslot scheme of the BiBat Master, shall be identified by the corresponding BiBat Block number (0 … 127).
  If synchronous frames are received outside of the timeslot structure given by the BiBat Master, an L_Data.ind shall anyway be generated with void field 'BiBat-Block' (255).

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA, L, TPCI/APCI and data shall be encoded as specified in 4.1.5.4.3 for asynchronous data frames.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4.

Concatenated L_Data frames in one Block

The BiBat system shall support the transmission of up to three subsequent data frames in one Block.

Therefore the cEMI Server shall be able to handle up to three subsequent synchronous RF data frames in the same Block and generate the corresponding independent L_Data.ind containing the same Block number.

4.1.5.5.5 L_Data.req for BiBat Sync frame

In the synchronous BiBat system a Sync frame can be sent instead of a synchronous data frame.

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in L_Data.req.

If the cEMI Server supports BiBat Master functionality, it shall handle L_Data.req for synchronous data frames as specified below. If it does not support BiBat Master functionality, L_Data.req shall be discarded for Sync frames.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Random pause pointer**
11h	14d	…	…	…	…	…		…		0	…

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA, L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames. There shall be no TPCI/APCI. Instead the 'Random Pause Pointer' shall be included and encoded as follows:

- Values 0 to 12: This shall be interpreted as a valid random pause pointer value generated by the cEMI client; these mainly serve for test purposes.
- Value 255: The Random Pause pointer in the sent RF frame shall be generated autonomously by the cEMI Server/BiBat Master.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- ‘BiBat-Ctrl’ shall have the value 01010000b (Sync frame).
  The field ’KNX-Ctrl’ in the corresponding RF frame shall have the value 01010000b.
- ‘BiBat-Block’ shall contain a valid Block number 0 … 127 that shall indicate in which Block of the time-slotted BiBat scheme the Sync frame shall be sent.
  Sync frame may be sent in Block 0 and 64 only in normal BiBat timeslot procedure. But for test purpose also other Blocks may be used.

  Normally the cEMI Server/BiBat Master shall only be able to handle L_Data.req for the next Block (no ahead buffering of synchronous frames). The ‘BiBat-Block’ number shall be used to check block number synchronisation of cEMI Server and client.

  If the cEMI Server is not able to handle the requested block number, L_Data.req shall be discarded.

The cEMI Server shall inform cEMI Client about the next block number at least 3 s in advance using the M_PropInfo service. See 4.1.5.5.2.

4.1.5.5.6 L_Data.con for BiBat Sync frame

In the message format, additionally to the ‘RF medium information’ also ‘BiBat information’ shall be included in L_Data.con. The same frame structure as specified in 4.1.5.5.5 shall be used, with Message code = 2Eh.

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA, L shall be encoded as specified in 4.1.5.4.2 for asynchronous data frames. There shall be no TPCI/APCI. Instead the ‘Random Pause Pointer’ shall be included that shall contain the same value as in the L_Data.req.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following applies.

- ‘BiBat-Ctrl’ shall have the same value as in L_Data.req.
- ‘BiBat-Block’ shall have the same Block number as in L_Data.req.

The L_Data.con shall only be generated after completed transmission of the Sync frame in the next Block. Therefore there is a time delay in the range of up to 5 s between synchronous L_Data.req and L_Data.con.

4.1.5.5.7 L_Data.ind for BiBat Sync frames

In the message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included in L_Data.ind.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Random pause pointer**
29h	14d	…	…	…	…	…		…		0	…

The cEMI Server shall work in BiBat Slave mode as a receiver of Sync frames: the same conditions shall apply as for synchronous data frames as specified in see 4.1.5.5.4.

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA, L shall be encoded as specified in 4.1.5.4.3 for asynchronous data frames. There shall be no TPCI/APCI field. Instead the received ‘Random Pause Pointer’ shall be included.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4.

4.1.5.5.8 L_Data.req for BiBat Help Call frame

NOTE The support of the Help Call frame by the cEMI Server is mainly intended for testing purposes to emulate a BiBat Slave and to test a BiBat Master.

If a cEMI Server working as a BiBat Slave looses synchronisation it shall send out a help call on RF autonomously.

In the BiBat system a Help Call frame can be sent at any time (asynchronously) by a BiBat Slave. The cEMI Server shall work in BiBat Slave mode as a sender of Help Call frames.

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in L_Data.req.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Retransmitter number**
11h	14d	…	…	…	…	…		…		0	…

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA, L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames. There shall be no TPCI/APCI. Instead the field 'Retransmitter number' shall be included.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- The field 'BiBat-Ctrl' shall have the value 01100000b (Help Call).
  The field 'KNX-Ctrl' in the corresponding RF frame shall have the value 01100000b.
- The field 'BiBat-Block' shall be void and have the value 255.

For normal use, the field 'Retransmitter number' shall have the value 0 (original help call sent by a BiBat Slave). Only for test purposes, in order to simulate the behaviour of a retransmitter the 'Retransmitter number' may also contain values 1 … 3.

4.1.5.5.9 L_Data.con for Help Call frame

NOTE The support of the Help Call frame by the cEMI Server is mainly intended for testing purposes to test a BiBat Master. See note in 4.1.5.5.8.

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included. The same frame structure as specified in 4.1.5.5.8 shall be used, with message code = 2Eh.

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.2 for asynchronous data frames. There shall be no TPCI/APCI. Instead the field 'Retransmitter number' shall be included, which shall contain the same value as in the L_Data.req.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- The field 'BiBat-Ctrl' shall have the same value as in L_Data.req.
- The field 'BiBat-Block' shall be void and shall have the value 255.

4.1.5.5.10 L_Data.ind for Help Call frame

In the message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included in L_Data.ind.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Retransmitter number**
29h	14d	…	…	…	…	…		…		0	…

The cEMI Server shall work in BiBat Master mode as a receiver of Help Call frames.

> After reception of a Help Call an L_Data.ind shall be sent to the cEMI Client but additionally the cEMI Server shall locally generate a corresponding Help Call Response according to the ‘Retransmitter number’ (allowed range 0 … 3) and its local timeslot scheme.

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.3 for asynchronous data frames. There shall be no TPCI/APCI field. Instead the received ‘Retransmitter number’ shall be included.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- The field ‘BiBat-Block’ shall be void and shall have the value 255.

4.1.5.5.11 L_Data.req for BiBat Help Call Response frame

In the BiBat system a Help Call Response frame shall be sent by the BiBat Master with a fixed delay time after a Help Call and without checking the medium; see [01].

NOTE L_Data.req for a Help Call Response is normally not generated by the cEMI Client because the critical timing procedure after a Help Call can only be handled by the cEMI Server locally.
Generation of L_Data.req for a Help Call Response by the cEMI Client may be useful for test purposes only.

In the message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included in the L_Data.req.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Data**
11h	14d	…	…	…	…	…		…		0	…

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames. There shall be no TPCI/APCI. Instead a ‘Data’ field containing ‘ticks_until_start_of_next_block’ (3 octet) , next_block_nr (1 octet) and ‘Random Pause Pointer’ (1 octet) shall be included.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following shall apply.

- The field ‘BiBat-Ctrl’ shall have the value 01110000b (Help Call Response).
  The field ’KNX-Ctrl’ in the corresponding RF frame shall have the value 01110000b.
- The field ‘BiBat-Block’ shall be void and have the value 255.

4.1.5.5.12 L_Data.con for Help Call Response frame

In the message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included. The same frame structure as specified in 4.1.5.5.11 shall be used, with message code = 2Eh.

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.2 for asynchronous data frames. There shall be no TPCI/APCI. Instead the ‘Data’ field shall be included, which shall contain the same value as in the L_Data.req.

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following shall apply.

- The field ‘BiBat-Ctrl’ shall have the same value as in L_Data.req.
- The field ‘BiBat-Block’ shall be void and shall have the value 255.

4.1.5.5.13 L_Data.ind for Help Call Response frame

In the message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included in L_Data.ind.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**	**Data**
29h	14d	…	…	…	…	…		…		0	…

The fields ‘RF medium information’, Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames. There shall be no TPCI/APCI. Instead a ‘Data’ field containing ‘ticks_until_start_of_next_block’ (3 octets) , next_block_nr (1 octet) and ‘Random Pause Pointer’ (1 octet) shall be included as specified in [01].

The field ‘BiBat information’ shall be used as specified in 4.1.4.3.4; additionally, the following shall apply.

- The field ‘BiBat-Block’ shall be void and shall have the value 255.

The cEMI Server shall work in BiBat Slave mode as a receiver of Help Call Response frames.

After reception of a Help Call Response an L_Data.ind shall be sent to the cEMI Client but additionally the cEMI Server shall synchronize its local BiBat Slave timeslot scheme.

4.1.5.5.14 L_Data.req for BiBat Fast_ACK frame

A Fast_ACK shall be generated by the BiBat Master after reception of an asynchronous data frame (within 200 ms reaction time) whenever either the KNX Serial Number or the Individual Address in the received message matches with an entry in a table that is held by the BiBat Master; see [01].

The timing to generate Fast_ACK is not very critical. Therefore the tables containing KNX Serial Numbers and Individual Addresses of devices that require Fast_ACK response shall be contained in the cEMI Client. The cEMI Client shall handle the Fast_ACK mechanism locally to reduce overhead (configuration and storage of tables) in the cEMI Server.

The cEMI Client shall check the KNX Serial Number or Individual Address of each L_Data.ind (asynchronous data frames) and generate an L_Data.req for the corresponding Fast_ACK in due time, so that the cEMI Server is able to send the Fast_ACK RF frame within 200 ms.

In the cEMI Message format, in addition to the ‘RF medium information’ also ‘BiBat information’ shall be included in L_Data.req.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SA H**	**SAL**	**DA H**	**DA L**	**L**
11h	14d	…	…	…	…	…		…		0

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames. There shall be no TPCI/APCI and there shall be no Data field.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply.

- The field 'BiBat-Ctrl' shall have the value 00010000b (Fast_ACK).

  In the 'KNX-Ctrl' field of the corresponding RF frame
  - bits b7 (msb) to b4 shall have the value 0001b, and
  - bits b3 to b0 (lsb) shall be set according the Frame Type.
- The field 'BiBat-Block' shall be void and shall have the value 255.

4.1.5.5.15 L_Data.con for BiBat Fast ACK frame

In the message format, in addition to the 'RF medium information' also the field 'BiBat information' shall be included. The same frame structure as specified in 4.1.5.5.14 shall be used, with message code = 2Eh.

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.2 for asynchronous data frames.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply.

- The field 'BiBat-Ctrl' shall have the same value as in L_Data.req.
- The field 'BiBat-Block' shall be void and shall have the value 255.

4.1.5.5.16 L_Data.ind for BiBat Fast ACK frame

The cEMI Server shall work in asynchronous or BiBat Slave mode to handle received Fast_ACK frames.

In the message format, in addition to the 'RF medium information' also 'BiBat information' shall be included in the L_Data.ind.

EXAMPLE

Message Code	Additional Info Length	RF medium information	BiBat information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU
MC	**AddIL**			**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**
29h	14d	…	…	…	…	…		…		0

The fields 'RF medium information', Ctrl1, Ctrl2, SA, DA and L shall be encoded as specified in 4.1.5.4.1 for asynchronous data frames.

The field 'BiBat information' shall be used as specified in 4.1.4.3.4; additionally, the following shall apply:

- The field 'BiBat-Block' shall be void and shall have the value 255.

4.1.5.6 L_Poll_Data service

4.1.5.6.1 Implementation and usage

L_Poll_Data are applicable only with TP physical medium devices. L_Poll_Data is optional for a cEMI Server.

4.1.5.6.2 L_Poll_Data.req

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	number of slots
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**NoS**
13h	…	…	x0r0ppa0	…	…	…	Polling	Group	0000ssss

4.1.5.6.3 L_Poll_Data.con

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	number of slots	Poll Data 0 … (ssss – 1)
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**NoS**	**Poll Data**
25h	…	…	x0r0ppxC	…	XX	XX	Polling	Group	0000ssss	…

C: Confirm Flag: 1 = ok; 0 = not ok

4.1.5.7 L_Raw service

4.1.5.7.1 Implementation and usage

The L_Raw service is optional for a cEMI Server. It is intended for special use, this is for use in testing or diagnostic tools. It shall not be used for “normal operation”.

4.1.5.7.2 Basic message structure for L_Raw messages

Message Code	Additional Info Length	Additional Information	Raw Frame on Medium [6]
MC	**AddIL**	**…**	**Data**
1 octet	1 octet	var. length	

The L_Raw service, as an important feature, shall also contain the frame checksum information as part of the field *Data* (*Raw Frame on medium*).

4.1.5.7.3 L_Raw.req [7], [8]

Message Code	Additional Info Length	Additional Information	Raw Frame on Medium
MC	**AddIL**	**…**	**Data**
10h	…	…	…

[6] Raw Frame on media: starting with the Ctrl octet & ending with the CRC octet(s).
Depending on the used medium: including e.g. also the PL Domain Address on PL media, or the RF Ctrl octet & Serial Number/DoA on RF media → the cEMI Client must know, which medium is used to send raw data.

[7] EMI 2: the equivalent service is specified as *L_Plain_Data.req*

[8] If the cEMI Server supports L_Raw service and multiple media types, it’s not allowed to indicate more than one medium as “supported” in PID_Medium_Type in the cEMI Server object until it’s fully specified (e.g. by a channel concept) how the cEMI client knows the frame format to be used in L_Raw.req

The L_Raw.req service primitive shall be used for sending a frame in raw format onto any of the given KNX media [9]. This message can be used e.g. in a test environment (by a test tool like the EITT) to generate faulty frames and to send them onto the KNX network.

A time delay until sending can be used. It shall be available as a field within *Additional Information*.

If no time delay is present as additional information field, the frame shall be sent onto the KNX medium by the cEMI Server device with the regular interframe time.

NOTE The following mentioned *Delay Time* is specified for use with existing implementations, in particular for mapping a cEMI's *L_Raw.req* message to an EMI 2's *L_Plain_Data.req* message. Future implementations shall use implementation independent delay times, e.g. a delay time in ms or sec. These may be also positive and negative values, compared to "regular interframe time", could be used. Such delay times are not yet specified. If needed, they shall be specified as another new *Additional Information Type* [10].

If the *Delay Time* is present as additional information with Type ID = 05h (see 4.1.4.3 "Additional information"), then the *Delay Time* shall be a 4 octet long (unsigned) counter value.

If *Delay Time* = 00000000h the frame shall be sent immediately. Otherwise the frame shall be sent when the free running system counter of the sending device is equal to the value given in *Delay Time* [11].

4.1.5.7.4 L_Raw.con

Message Code	Additional Info Length	Additional Information	Raw Frame on Medium
MC	**AddIL**	**…**	**Data**
2Fh	…	…	…

The *L_Raw.con* shall be a local primitive generated by the cEMI Server's Data Link Layer for its own cEMI client to indicate whether it is satisfied with the transmission or not.

A positive L_Raw.con message shall contain the same data in the message as it is received with L_Raw.req. A positive L_Raw.con message shall be generated as soon as the raw frame is sent completely.

Negative L_Raw.con message: to be defined [12].

4.1.5.7.5 L_Raw.ind

Message Code	Additional Info Length	Additional Information	Raw Frame on Medium
MC	**AddIL**	**…**	**Data**
2Dh	…	…	…

The *L_Raw.ind* service shall be used for receiving a frame in raw format from any of the given KNX media. This message can be used e.g. in a test environment (by a test tool like the EITT) to receive faulty frames from the KNX network and to display them for diagnostic purposes.

L_Raw.ind shall be the Data Link Layer service primitive that transfers received frames from the local Layer-2 completely (including FCS) to the local Layer-2 user, including all received octets in raw format.

As a difference to the *L_Busmon.ind* service (as specified in clause 4.1.5.7.6 "L_Busmon.ind "), Data Link Layer acknowledges shall not be transferred with the *L_Raw.ind* service primitive.

[9] For implementations with TP-UART, please refer to the corresponding datasheet for restrictions using the "L_Raw feature".

[10] Interested applicants shall contact the KNX System Department for extension of the specification.

[11] Identical to specification of "time"-field within L_Plain_Data.req in EMI 2 specification, see 3.3.3.3.

[12] Currently (May 2002), no implementation of L_Raw.req or L_raw.con on cEMI is planned or known; to be defined if needed for an implementation.

4.1.5.7.6 L_Busmon.ind [13]

Message Code	Additional Info Length	Additional Information	Raw Frame on Medium
MC	**AddIL**	**…**	**Data**
2Bh	…	…	…

The *L_Busmon.ind* message is the equivalent to the *L_Busmon.ind* message in EMI 1 and EMI 2. This Data Link Layer service primitive shall be used to transfer every received message from the local Layer-2 completely (inclusive the FCS) to the local Layer-2 user, including all received octets in raw format. Data Link Layer acknowledges shall also be transferred.

The value of the last octet within the *L_Busmon.ind* message is usually the FCS octet that is received from the bus by the cEMI Server device. It is the task of the external Busmonitor application (cEMI Client device/tool) to check its correctness.

The following example shows the *L_Busmon.ind* message frame as 1:1 equivalent to the L_Busmon.ind message of EMI 1 and EMI 2; see 3.3.3.2.

Message Code	Additional Info Length	Additional Information							Raw Frame on Medium
		Busmonitor Error Flags			Timestamp				
MC	**AddIL**	**Type ID**	**Len**	**Status**	**Type ID**	**Len**	**Timestamp Relative**		**Data**
2Bh	7	03h	1	FBPxLsss	04h	2	XX	XX	…

The field *Status* shall be 1 octet long and shall be contained in the field *Busmonitor Error Flags*. The field *Status* shall be mapped to an own information type within the *Additional Information*; see 4.1.4.3.

The field *Status* shall be used as specified for EMI 1 and EMI 2.

The field *Status* shall be encoded as follows (originating from TP Busmonitor applications):

- bit 7 (msb):

  name (abbreviation): Frame error flag (F)
  description: This bit shall be set only if a frame error is detected in one or several of the frame bits in the message.
  encoding: 0 = No frame error is detected in the message
  1 = One or more frame errors are detected in the message.

- bit 6:

  name (abbreviation): Bit error flag (B)
  description: This bit shall be set only if an invalid bit is detected in one or several of the frame characters.
  encoding: 0 = None of the frame characters contains an invalid bit.
  1 = One or more frame characters contains an invalid bit.

- bit 5:

  name (abbreviation): Parity error flag (P)
  description: This bit shall be set only if an invalid parity bit is detected in one or several of the frame bits.
  encoding: 0 = None of the frame bits contains a parity error.
  1 = One or more frame bits contains a parity error.

[13] Please refer to the datasheets of used components (e.g. TP-UART) and certified BAUs for restrictions in implementation of the “Busmonitor / L_Raw Data Link Layer services” feature.

- bit 4:

  name (abbreviation): don’t care
  description: Overflow flag. The overflow flag is set.
  encoding: Shall be 0.

- bit 3:

  name (abbreviation): Lost flag - L
  description: This bit shall only be set if at least one frame or frame piece was lost by the Data Link Layer in Busmonitor mode.
  NOTE The difference between the sequence number of the previous L_Raw.ind without lost flag set and the sequence number of the L_Raw.ind with lost flag set may not reflect exactly the number of lost frames or frame pieces
  encoding: 0 = No frame or frame part is lost.
  1 = At least one frame or frame part is lost.

- bit 2 to bit 0 (lsb):

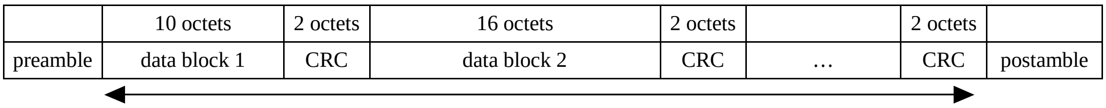
  name (abbreviation): Sequence Number - sss
  description: The Data Link Layer in Busmonitor mode shall maintain a sequence counter. This sequence counter shall be incremented with every received frame. The value of this sequence counter modulo 8 shall be contained in this Sequence Number field. The least significant bit of octet 2 is also the least significant bit of the sequence number.
  encoding: Value binary encoded.

An *L_Busmon.ind* service primitive using the *Status* information field (Busmonitor Error Flags) is not obliged to support all the flags and the sequence number within the status octet. Unused bits shall not be used for other purpose(s); they shall be fixed to 0.

The field *Timestamp Relative* shall be a 16 bit value and shall refer to the relative time taken exactly at the time when the frame's control field is completely received at the Data Link Layer. The time shall be the value of the free running counter of the BAU. The time unit ("tick") depends on the clock rate of the BAU micro controller. The field *Timestamp Relative* shall be mapped to an own information type within the *Additional Information*; see 4.1.4.3.

4.1.5.7.7 L_Raw Services on RF

NOTE Data Link Layer frame on the RF Medium:



	10 octets	2 octets	16 octets	2 octets		2 octets	
preamble	data block 1	CRC	data block 2	CRC	…	CRC	postamble

Figure 6 – L_Raw data for the RF medium

L_Raw messages shall not contain the preamble and postamble of the RF frame. Therefore using L_Raw service it is not possible to generate complete RF frames on the basis of RF chip sequences.

When handling an L_Raw.ind from the cEMI Client to transmit a frame on RF, the cEMI Server shall generate the correct preamble and postamble locally.
When receiving a frame from the cEMI Server shall trigger an L_Raw.ind and shall copy the received frame starting from data block1 up to and including the CRC of the last block to the data field of the L_Raw.ind.

4.1.6 Transport Layer messages

4.1.6.1 Basic frame structure for cEMI Transport Layer messages

The cEMI Transport Layer interface shall provide two services.

1. T_Data_Individual
2. T_Data_Connected

The basic Frame Structure for both services shall be as specified in Figure 7 below.

Message Code	Additional Info Length	Additional Information	Reserved	TPDU			
MC	**AddIL**	**…**		**L**	**reserved**	**APCI /Data**	**Data**
1 octet	1 octet	var. length	6 octets	1 octet	6 bits	10 bits	var. length
			000000000000h		000000b		

Figure 7 – Basic cEMI Transport Layer frame structure

- MC: message code
- AddIL: length of additional information
- Additional Information: This shall contain one or more Additional Information fields according the value of the field *Additional Information Length* as specified in 4.1.4.3.
- Reserved: This field is reserved and shall be filled with 000000000000h.
  NOTE The goal of this empty field is to align the position of the fields L and further in the message buffer with the position of these fields in the cEMI L_Data-frame.
- L: This field shall contain the Information Length. The maximal value shall be 255.The value of this field shall be the size in octets of the field *TPDU*, not including the field *L*, not including the field *reserved* and starting with 1 with the field *APCI/Data*. The minimal value of this field shall thus be 1.
- reserved: This field is reserved and shall be filled with 000000b.
  NOTE 1 The goal of this empty field is to align the position of the fields APCI and further in the message buffer with the position of these fields in the cEMI L_Data-frame.
- APCI/Data
  name (abbreviation): Application Layer Protocol Control Information (APCI) or Data
  description: This field shall contain the APCI of the transported AL service, or the data.
  encoding: See [05]

4.1.6.2 Flow Control

Opposite to the cEMI Data Link Layer, the cEMI Transport Layer does not foresee confirmations of the cEMI frames. The cEMI Transport Layer has no provisions for flow control.

4.1.6.3 General exception handling

There is no general exception handling for cEMI Transport Layer frames.

4.1.6.4 T_Data_Individual service

4.1.6.4.1 Implementation and usage

The implementation of the T_Data_Individual service depends on the cEMI Profile as specified in [13].

4.1.6.4.2 Basic frame structure for T_Data_Individual messages

The basic frames structure shall be as specified in 4.1.6.1.

4.1.6.4.3 Message flow

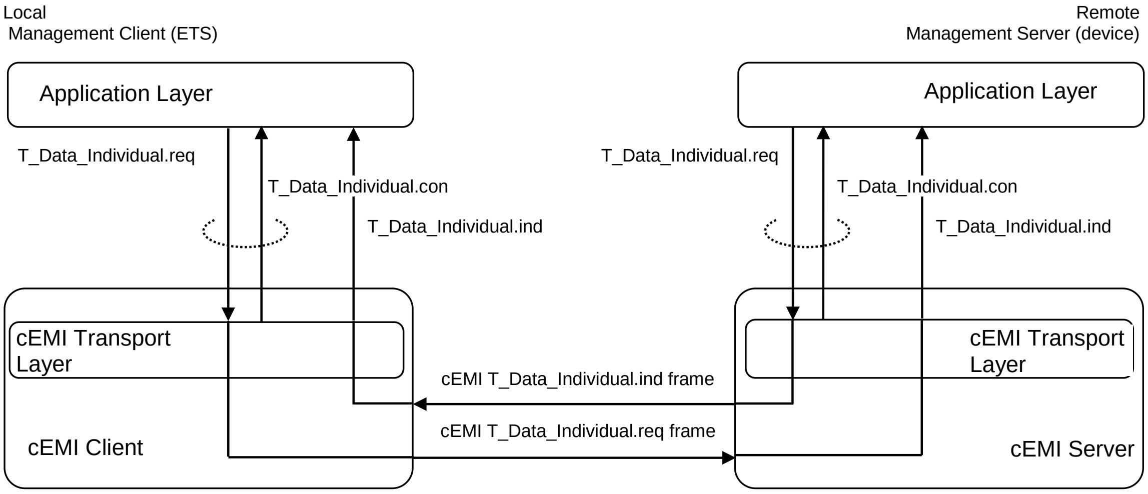


Figure 8 – Message flow for T_Data_Individual

4.1.6.4.4 cEMI T_Data_Individual.req frame

If the local user of the Transport Layer in the cEMI Client device applies the T_Data_Individual.req service primitive then the cEMI Client shall send the cEMI T_Data_Individual.req frame to the cEMI Server.

If the cEMI Client is satisfied with the transmission, it shall apply the T_Data_Individual.con service primitive to the local cEMI Transport Layer user.

If the cEMI Server receives a cEMI T_Data_Individual.req frame, it shall pass the contained APDU to the remote Application Layer by a T_Data_Individual.ind service primitive.

Message Code	Additional Info Length	Additional Information	Reserved	TPDU			
MC	**AddIL**	…		**L**	**reserved**	**APCI /Data**	**Data**
1 octet	1 octet	var. length	6 octets	1 octet	6 bits	10 bits	var. length
4Ah			000000000000h		000000b		

Figure 9 – cEMI T_Data_Individual.req frame

4.1.6.4.5 cEMI T_Data_Individual.ind frame

If the remote user of the Transport Layer in the cEMI Server device applies the T_Data_Individual.req service primitive then the cEMI Server shall send the cEMI T_Data_Individual.ind frame to the cEMI Client.

If the cEMI Server is satisfied with the transmission, it shall apply the T_Data_Individual.con service primitive to the remote cEMI Transport Layer user.

If the cEMI Client receives a cEMI T_Data_Individual.ind frame, it shall pass the contained APDU to the local Application Layer by a T_Data_Individual.ind service primitive.

Message Code	Additional Info Length	Additional Information	Reserved	TPDU			
MC	**AddIL**	**…**		**L**	**reserved**	**APCI /Data**	**Data**
1 octet	1 octet	var. length	6 octets	1 octet	6 bits	10 bits	var. length
94h			000000000000h		000000b		

Figure 10 – cEMI T_Data_Individual.ind frame

4.1.6.5 T_Data_Connected service

4.1.6.5.1 Implementation and usage

The implementation of the T_Data_Connected service depends on the cEMI Profile as specified in [13].

4.1.6.5.2 Basic frame structure for T_Data_Connected messages

The basic frames structure shall be as specified in 4.1.6.1.

4.1.6.5.3 Message flow

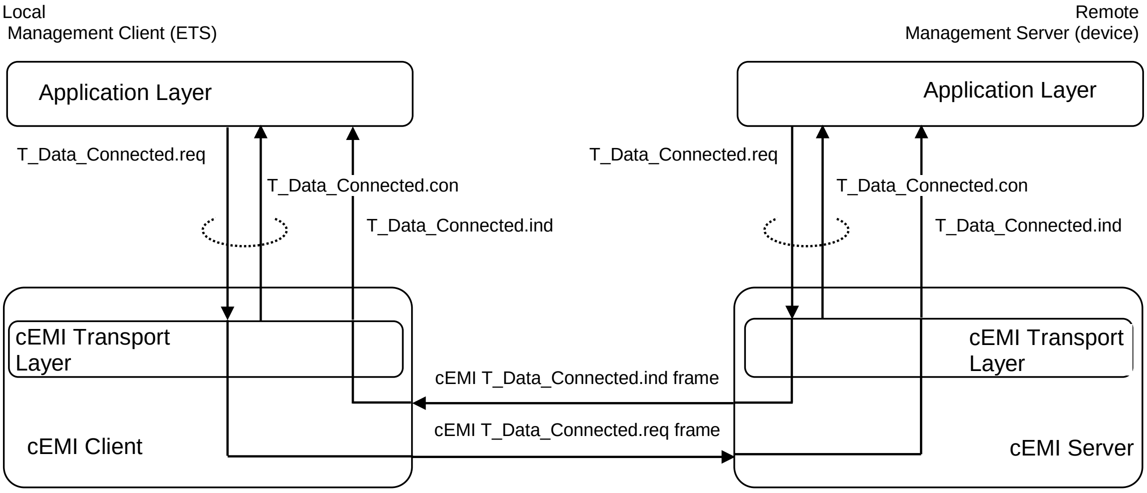


Figure 11 – Message flow for T_Data_Connected

4.1.6.5.4 cEMI T_Data_Connected.req frame

If the local user of the Transport Layer in the cEMI Client device applies the T_Data_Connected.req service primitive then the cEMI Client shall send the cEMI T_Data_Connected.req frame to the cEMI Server.

If the cEMI Client is satisfied with the transmission, it shall apply the T_Data_Connected.con service primitive to the local cEMI Transport Layer user.

If the cEMI Server receives a cEMI T_Data_Connected.req frame, it shall pass the contained APDU to the remote Application Layer by a T_Data_Connected.ind service primitive.

Message Code	Additional Info Length	Additional Information	Reserved	TPDU			
MC	**AddIL**	**…**		**L**	**reserved**	**APCI /Data**	**Data**
1 octet	1 octet	var. length	6 octets	1 octet	6 bits	10 bits	var. length
41h			000000000000h		000000b		

Figure 12 – cEMI T_Data_Connected.req frame

4.1.6.5.5 cEMI T_Data_Connected.ind frame

If the remote user of the Transport Layer in the cEMI Server device applies the T_Data_Connected.req service primitive then the cEMI Server shall send the cEMI T_Data_Connected.ind frame to the cEMI Client.

If the cEMI Server is satisfied with the transmission, it shall apply the T_Data_Connected.con service primitive to the remote cEMI Transport Layer user.

If the cEMI Client receives a cEMI T_Data_Connected.ind frame, it shall pass the contained APDU to the local Application Layer by a T_Data_Connected.ind service primitive.

Message Code	Additional Info Length	Additional Information	Reserved	TPDU			
MC	**AddIL**	**…**		**L**	**reserved**	**APCI /Data**	**Data**
1 octet	1 octet	var. length	6 octets	1 octet	6 bits	10 bits	var. length
89h			000000000000h		000000b		

Figure 13 – cEMI T_Data_Connected.ind frame

4.1.7 Services for Local device management

4.1.7.1 Introduction

This clause specifies the communication relevant parts (services) for local device management. A more detailed specification of information and mechanisms used for local device management follows in clause 4.2 “Common EMI: Local Device Management”.

The local management services shall be confirmed services. The *M_PropRead.req* as well as the *M_PropWrite.req* service primitives shall always be followed by the corresponding *M_PropRead.con or M_PropWrite.con* service primitives containing information about success of the request. The *M_FuncPropCommand.req* as well as the *M_FuncPropStateRead.req* service primitives shall always be followed by the corresponding *M_FuncPropCommand.con* servie service primitive containing information about success of the request.

Messages generated by the cEMI Server, e.g. for event notification, shall use the unconfirmed *M_PropInfo.ind* service.

4.1.7.2 Management services flow control

cEMI Client

To keep the flow control for management services as simple as possible (this allows for a simple flow control state machine in the cEMI client), it is recommended that:

- a cEMI client sends a new request only when the confirmation of the preceding request is received, or
- a request-to-confirmation timeout is recognised; the recommended value for this time-out for the cEMI client is 1 second.

A cEMI client shall be able to accept an indication message from the cEMI Server at any time.

Behaviour of the cEMI Server

A cEMI Server device shall have a receive buffer for one or more cEMI messages. The cEMI Server shall accept new frames from the cEMI client only if the receive buffer is not full. The request frames in the receive buffer shall be treated sequentially according to a FIFO queue (first in, first out).

NOTE For cEMI Servers with a receive buffer for more than one message, the order of received frames can be any concerning the type of received messages (Data Link Layer or Management).

During treatment of a request that is not yet confirmed to the cEMI client, the cEMI Server shall accept a new request from the cEMI client. This is used e.g. for L_Data.req service primitives (see 4.1.5.3) during an M_PropRead.req, M_PropRead.con, M_PropWrite.req M_PropWrite.con cycle.

4.1.7.3 Data Properties

4.1.7.3.1 Basic message structure for Data Properties

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Data**
1 octet	2 octets		1 octet	1 octet	4 bit	12 bits	var. length

MC: message code

IOTH: Interface Object Type, high octet

IOTL: Interface Object Type, low octet

OI: Object Instance; 0 = reserved (not used) [14]; 1 = 1st instance; 2 = 2nd instance; 3 = 3rd instance, …

PID: Property Identifier

NoE: Number of elements for an array structured Property; if the Property is not an array: NoE = 1

SIx: Start Index within an array-structured Property, the 1st element shall be placed at index 1; the array element ´0´ shall contain the current number of valid array elements (unsigned 16 bit value).

NOTE The Interface Object structure is the same as specified in [06].

Data: This is the Data field of the message. The length of the data field shall depend on the Property Datatype of the Property and in case of an array structured Property value also on the number of array elements that are accessed.

[14] LTE Interface Object addressing uses Object Instance 0 as a wildcard: read/write from/to all instances; see [17].

4.1.7.3.2 M_PropRead.req

The *M_PropRead.req* message shall be applied by the management client to read the value of a Property of an Interface Object in the management server. The Interface Object of the partner shall be addressed with an Object Type and an Object Instance number. A Property within an Interface Object can be structured as an array of elements. Therefore, the Property within the Interface Object shall be addressed with a Property ID, the number of elements (NoE) and a start index (SIx). NoE shall indicate the number of array elements starting with the given SIx in the Property value that the management client wants to read. The management server shall confirm the request with a *M_PropRead.con* message. The confirmation is specified in clause 4.1.7.3.3 “M_PropRead.con”. The M_PropRead.req shall not contain any further data.

M_PropRead.req message

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**
FCh	…		≥ 1	…	> 0	…

For reading the currently valid number of array elements, NoE shall be set to 1 and start index to 0.

4.1.7.3.3 M_PropRead.con

The management server shall confirm a *M_PropRead.req* message with a *M_PropRead.con* message. This response shall contain the requested number of elements beginning at the requested start index within the requested Property of the addressed Interface Object.

M_PropRead.con message, positive response:

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Data**
FBh	as in .req		as in .req	as in .req	as in .req	as in .req	…

M_PropRead.con message, number of valid array elements:

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Number of elements**
FBh	as in .req		as in .req	as in .req	1	0	number (unsigned 16 value)

Error Handling

If the management server has a problem, e.g. Interface Object or Property does not exist, then the NoE shall be set to zero and the Start Index of the response shall be set to same value as received with the request. The data field of a negative response shall contain error information. The error information of a negative confirmation shall be a one octet long enumerated data field. For more details concerning the error handling and the Error Code Set please refer to clause 4.1.7.3.7 “cEMI Server exception handling after management service request”.

M_PropRead.con message, negative response

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Error information
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Error Code**
FBh	as in .req		as in .req	as in .req	0	as in .req	…

4.1.7.3.4 M_PropWrite.req

The *M_PropWrite.req* message shall be applied by the management client to modify the value of a Property of an Interface Object in the management server. The Interface Object of the partner shall be addressed with an Object Type and an Object Instance number. A Property within an Interface Object can be structured as an array of elements. Therefore, the Property within the Interface Object shall be addressed with a Property ID, the number of elements (NoE) and a start index (SIx). NoE shall indicate the number of array elements starting with the given SIx in the Property value that the management client wants to write. The management server shall confirm the request with a *M_PropWrite.con* message. The confirmation is described in clause 4.1.7.3.5 "M_PropWrite.con".

M_PropWrite.req message

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Data**
F6h	…		≥ 1	…	> 0	> 0	…

4.1.7.3.5 M_PropWrite.con

The management server shall confirm an *M_PropWrite.req* message with a *M_PropWrite.con* message. The (positive) response shall not contain any data, this is, data field shall not be present. The number of elements (value of NoE) and the start index (value of SIx) shall be set to the same value as in the request.

M_PropRead.con message, positive response:

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**
F5h	as in .req		as in .req	as in .req	as in .req	as in .req

Error Handling

If the management server has a problem, e.g. Interface Object or Property doesn't exist, then the NoE shall be set to zero and the Start Index of the response shall be set to same value as received with the request. The data field of a negative response shall contain error information. The error information of a negative confirmation shall be a one octet long enumerated data field. For more details concerning the error handling and the Error Code Set please refer to clause 4.1.7.3.7 "cEMI Server exception handling after management service request".

M_PropWrite.con message, negative response:

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Error information
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SIx**	**Error Code**
F5h	as in .req		as in .req	as in .req	0	as in .req	…

M_PropWrite.con shall contain information about the result of the Property access in the cEMI Server's database, not only about the writing of the Property alone, if the full error handling as specified in clause 4.1.7.3.7 is supported, especially if error code 04 "memory error" is supported.

The reaction time for the M_PropWrite.con to be sent shall be specified for the host protocol on which cEMI is used.

4.1.7.3.6 M_PropInfo.ind

The *M_PropInfo.ind* message shall be applied by the management server to send an event notification to the management client, e.g. about a changed management Property value.

After reception of the *M_PropInfo.ind*, the management client shall check whether the contained data is relevant to one or more of the management procedures it supports. If so, these procedures shall be called with the received data. If no, the message shall be ignored.

M_PropInfo.ind shall always be an unconfirmed service.

M_PropInfo.ind message

Message Code	Interface Object Type		Object Instance	Property ID	number of elements	start index	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**NoE**	**SI**	**Data**
F7h	…		≥ 1	…	> 0	> 0	…

4.1.7.3.7 cEMI Server exception handling after management service request

4.1.7.3.7.1 Error handling

If a confirmed service (*M_PropRead.req*, *M_PropWrite.req*) can not be executed successfully by the cEMI management server, the cEMI Server shall generate a negative confirmation as an ‘Error’. Such an error shall be transmitted to the cEMI management client within the negative response PDU; see 4.1.7.3.3 “M_PropRead.con” and 4.1.7.3.5 “M_PropWrite.con”.

As the minimum requirement, an ‘Unspecified Error’ shall be returned if any problem occurs.

4.1.7.3.7.2 Error Code Set

The Error Code Set uses an 8 bit enumeration data type (N_8), as specified in Table 12.

Table 12 – Error Code Set

Error Code	Error Type	Description	Service
00h	**Unspecified Error**	unknown error	R/W
01h	**Out of Range**	write value not allowed (general, if not error 2 or 3)	W
02h	**Out of MaxRange**	write value to high	W
03h	**Out of MinRange**	write value to low	W
04h	**Memory Error**	memory can not be written or only with fault(s)	W
05h	**Read Only**	write access to a ‘read only’ or a write protected Property	W
06h	**Illegal COMMAND**	COMMAND not valid or not supported	W
07h	**Void DP**	read or write access to an non existing Property	R/W
08h	**Type Conflict**	write access with a wrong data type (Datapoint length)	W
09h	**Prop. Index Range Error**	read or write access to a non existing Property array index	R/W
0Ah	**Value temporarily not writeable**	The Property exists but can at this moment not be written with a new value	W

In case of multiple errors, the error type in the negative confirmation shall be given by the error testing sequence (specified in the following 4.1.7.3.7.3 “cEMI Server behaviour after reception of a management service request”): the 1st detected error shall be the error indication in the negative confirmation.

4.1.7.3.7.3 cEMI Server behaviour after reception of a management service request

If only the minimum requirements concerning error handling are supported, the following applies.

1) If any problem is detected, the access shall be confirmed by the negative response ‘Unspecified Error”, else continue with 2).

2) Send a positive service confirmation.

If a more sophisticated error handling is supported, providing more differentiating error identification, the following applies.

1) If the accessed (read or write) Property does not exist in the cEMI Server, the access shall be confirmed with the negative response ‘Void DP’, else continue with 2).

2) If the array field(s) within the accessed (read or write) Property causes a problem, e.g. too many elements are accessed, the access shall be confirmed with the negative response ‘Property Index/Range Error’, else continue with 3) for write, respectively 8) for read access.

3) If write access on the write accessed Property is not allowed, the write access shall be confirmed by the negative response ‘Read Only’, else continue with 4).

4) If the COMMAND [15] accompanying a Property value (write access) is not supported by the cEMI Server, the service request shall be confirmed with the negative response ‘Illegal CMD’, else continue with 5) [16].

5) If the Data (Point) Type of the received Property value does not comply with one of the Properties that is requested to be written (detected by different data lengths), the service request shall be confirmed by the negative response ‘Type Conflict’; else continue with 6).

6) If the Property value to be written is out of the allowed value range, the service request shall be confirmed by the negative response 'Out of Range', 'Out of MaxRange' or 'Out of MinRange' as appropriate; else continue with 7).

7) If the Property value to be written can not be stored successfully in the cEMI Server’s device database, the service request shall be confirmed by the negative response ‘Memory Error’; else continue with 8).

8) Send a positive service confirmation.

For each cEMI Server implementation, the supported Error Codes shall be declared in the corresponding cEMI Server profile.

4.1.7.4 Function Properties

4.1.7.4.1 M_FuncPropCommand.req

The M_FuncPropCommand.req message shall be applied by the Management Client to call a Property Function of an Interface Object in the Management Server. The Interface Object of the partner shall be addressed with an Object Type and an Object Instance number, the Property shall be addressed with the Property Identifier (PID). The content of the data part depends on the Property function.

[15] A write accessed Property value can be accompanied by a COMMAND, e.g. for ‘Set Out of Service’, ‘Override’. Examples of such parameters can be found in the HVAC Functional Block specifications (Part 7/1 of the KNX Specifications). Please refer to [09] for more details on the COMMAND octet.

[16] In today’s view (May-2002), the error ‘Illegal COMMAND’ is applicable only in configurations, where application program parameters (= Properties of an application Interface Object) are loaded locally into the BAU using the management services of the cEMI message protocol.

The Management Server shall confirm the request with a M_FuncPropCommand.con message. The data part of the confirmation is also dependant from the function called.

Message Code	Interface Object Type		Object Instance	Property Identifier	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**Data**
1 octet	2 octets		1 octet	1 octet	n octets
F8h	…		≥ 1	…	…

Figure 14 – M_FuncPropCommand.req message

4.1.7.4.2 M_FuncPropCommand.con

The Management Server shall confirm a M_FuncPropCommand.req message with an M_FuncPropCommand.con message. This response shall contain the return_code and additional data, which are both dependent on the specific Property Function that is called.

Message Code	Interface Object Type		Object Instance	Property Identifier	return_code	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**Ret**	**Data**
1 octet	2 octets		1 octet	1 octet	1 octet	n octets
FAh	as in req.		as in req.	as in req.	xx	…

Figure 15 – M_FuncPropCommand.con message

NOTE The M_FuncPropCommand.con message and the M_FuncPropStateRead.con message have the same message code and –format.

4.1.7.4.3 M_FuncPropStateRead.req

The M_FuncPropStateRead.req message shall be applied by the Management Client to perform a status read on a Property Function of an Interface Object in the Management Server. The Interface Object of the partner shall be addressed with an Object Yype and an Object Instance number, the Property shall be addressed with the Property Identifier (PID). The contents of the data part depends on the Property Function.

The Management Server shall confirm the request with a M_FuncPropStateRead.con message. The data part of the confirmation is also dependant from the function called.

Message Code	Interface Object Type		Object Instance	Property Identifier	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**Data**
1 octet	2 octets		1 octet	1 octet	n octets
F9h	…		≥ 1	…	…

Figure 16 – M_FuncPropStateRead.req message

4.1.7.4.4 M_FuncPropStateRead.con

The Management Server shall respond to an M_FuncPropStateRead.req message with an M_FuncPropStateRead.con message.

This M_FuncPropStateRead.con shall contain the return_code and additional data, which are both dependent on the specific Function Property of which the state is read.

Message Code	Interface Object Type		Object Instance	Property Identifier	return_code	Data
MC	**IOTH**	**IOTL**	**OI**	**PID**	**Ret**	**Data**
1 octet	2 octets		1 octet	1 octet	1 octet	n octets
FAh	as in req.		as in req.	as in req.	xx	…

Figure 17 – M_FuncPropStateRead.con message

NOTE The M_FuncPropCommand.con message and the M_FuncPropStateRead.con message have the same message code and –format.

4.1.7.4.5 Error and exception handling for cEMI Function Properties

If the Interface Object Property accessed by M_FuncPropCommand.req or by M_FuncPropStateRead.req is not of the Property Datatype PDT_Function, the remote application shall respond with a M_FuncPropCommand.con or M_FuncPropStateRead.con respectively, with empty return_code (i.e. the returned PDU shall not contain the field return_code) and no data (i.e. the returned PDU shall not contain the field data).

In case the remote application is able to successfully call a Function Property, the Function Property shall deliver a return_code in the field return_code. The following rules shall apply for all functions:

- return_code = 00h: function successfully executed, i.e. the return code 00h shall be the indication of the positive result of the function;
- return_code ≠ 00h: error.

Error codes are defined in a function specific way.

In case an Interface Object Property of Property Datatype PDT_Function is accessed via M_PropRead.req or M_PropWrite.req, the Application Layer shall respond with an M_PropRead.con or M_PropWrite.con with the standard error handling as specified in 4.1.7.3.7.

4.1.7.5 Further cEMI services for local device management

4.1.7.5.1 M_Reset.req

The *M_Reset.req* message shall be used to restart the cEMI Server device on initiation by the cEMI client. For a simple data interface device, this is without any application Interface Objects, restart means that the cEMI Server device executes the same actions as after a power up.

M_Reset.req message:

Message Code
MC
F1h

Reception of this message in the cEMI Server device shall be enabled at any time and shall lead to a re-initialisation of the cEMI Server software. This means that the communication on bus and cEMI side shall be aborted and both communication stacks shall be completely reset.

4.1.7.5.2 M_Reset.ind

The *M_Reset.ind* message shall be used to indicate to the cEMI client a reset or start-up of the cEMI Server device.

A M_Reset.ind shall be sent from the cEMI Server to the cEMI client after any of the following events:

- a M_Reset.req is received from the cEMI client and execution of this request is done;
- a bus power down-/up cycle, only in case the cEMI Server device is powered from the bus;
- after a power up of the cEMI Server device by any other reason, e.g. disconnecting/connecting cycle of the connection between cEMI Server and client, if also powering the cEMI device (e.g. USB).

M_Reset.ind message:

Message Code
MC
F0h

Exceptions

- M_Reset.ind is not mandatory for a USB powered device due to reasons on USB protocol/management level.
- M_Reset.ind is not mandatory for a KNXnet/IP server device as the execution of the reset will break the KNXnet/IP connection to the KNXnet/IP Client and the KNXnet/IP specification does not include automatic reconnecting to the KNXnet/IP client by the KNXnet/IP server.

NOTE 1 A M_Reset.con service is not applicable as the response/confirmation after a M_Reset.req. After a reset, a (simple) device does not know by what reason the reset is caused: by the M_Reset.req or any other reason. Therefore, the M_Reset.con is not applicable and therefore it’s also not specified.

NOTE 2 An A_Reset.req from the bus will be routed to the cEMI client within a L_Data-message. It is in the responsibility of the cEMI client, what has to be done after reception of such a M_Reset.req, this is to send a M_Reset.req to the cEMI Server or not.

4.2 Common EMI: Local Device Management

4.2.1 Introduction

Management and operation of the cEMI Server is defined with the approach to be as stateless as possible.

Management and operation messages to and from the cEMI Server can be interlaced, without switching the mode, e.g. from a “management mode” to any communication layer or vice versa.

A generic management interface based on Interface Objects is chosen for the management of the cEMI-Server. The management services are shown in clause 4.1.7 “Services for Local device management”.

cEMI is intended as new EMI for all future implementations. Future developments shall not use any legacy EMI implementations.

Nevertheless it is assured that existing EMI and device implementations (e.g. BCU 1 and BCU 2) can be mapped to cEMI specified in this document. Note however that this mapping task has to be performed by a component implementing the cEMI Server and plugged onto the PEI of the corresponding devices like a BCU 1 and/or a BCU 2.

These are some examples for local device management functions.

- Getting information from the connected device; e.g. Individual Address, Domain Address, maximum APDU-Length, connected medium, state of the device, possible communication modes of the device.
- Setting of the Individual Address.
- Setting of the Domain Address.
- Setting of the device’s communication layer mode; e.g. raw data (Busmonitor mode), Data Link Layer, Transport Layer.

4.2.2 Generic management based on Interface Objects

4.2.2.1 Introduction

For the generic management interface, services with own message codes are used to address the cEMI Server’s „local” Interface Objects and Properties. Access to Properties shall be done with *M_PropRead*/*Write*/*Info* services. These services are specified in detail in clause 4.1.7 “Services for Local device management”.

The following subclauses give an overview on the Interface Object Types (and their Properties) that can be used for cEMI Server management.

This document does not specify which Properties are mandatory or optional. This depends on the use of the cEMI protocol (e.g. the host protocol, the connected KNX medium, etc.) Please refer to [13] for this.

For each (open) cEMI Server implementation, the supported Properties shall be stated in a corresponding cEMI Server Profile document. If a new Profile is implemented, the manufacturer must provide the corresponding Profile document to the KNX System Department if the implemented cEMI Server device is intended to be used with any of the KNX system tools (ETS, EITT, …) as the cEMI management client. If new additional Properties are needed for a new Profile, e.g. within the cEMI Server Object, the applicant shall contact the KNX System Department for extension of the specification.

4.2.2.2 Device Object

Some of the relevant local management Properties are placed within the *Device Object*. The *Device Object* is one of the System Interface Objects. It is specified in [07]. Please refer to [07] for the specification of the Device Object and the below Properties.

Please refer to [06] for Interface Object addressing information (Object Types and Property Identifiers) and to [10] for the used Property Datatypes (PDTs).

For cEMI Server local management, the Device Object may hold the Properties as specified in Table 13.

Table 13 – Properties in the Device Object for the cEMI Server

Property name	PID	Description, remark
PID_SERVICE_CONTROL	8	Device flags
PID_SERIAL_NUMBER	11	KNX Serial Number
PID_DEVICE_CONTROL	14	Device flags
PID_MAX_APDULENGTH	56	Maximum APDU-Length (for extended frame format)
PID_SUBNET_ADDR	57	To manage the Individual Address (Subnetwork Address part).
PID_DEVICE_ADDR	58	To manage the Individual Address (Device Address part).
PID_DOMAIN_ADDR	70	Domain Address of a PL medium (cEMI Server) device. The value of the Property shall be the Domain Address of the cEMI Server device itself, this is if it should be in only one domain as a management server, seen from the bus media.
PID_IO_LIST	71	List of Interface Objects in the (cEMI Server) device.

This lists only the cEMI Server relevant Properties of the Device Object. For the specification of these and other Properties in the Device Object, please refer to [07].

4.2.2.3 PID_IO_LIST - Object scan mechanism

For cEMI Server devices supporting more than the minimum required two Interface Objects (Device Object and cEMI Server Object), it shall be possible to scan the available Interface Objects

EXAMPLE It shall be possible for a tool to check which Interface Objects are located in the device.

To this purpose the Property *PID_IO_LIST in the Device Object shall be used*. Please refer to [07] for the specification of PID_IO_LIST.

The Property PID_IO_LIST is mandatory for cEMI Server devices supporting other Interface Objects than only the Device Objects and the cEMI Server Object.

The cEMI client shall read out the present Interface Objects with the Local Device Management services specified in clause 4.1.7 “Services for Local device management”. If the ¨Property is not present, then only the Device Object and the cEMI Server Object shall be present in the cEMI Server device.

4.2.2.4 cEMI Server Object

4.2.2.4.1 Overview

For cEMI Server local management, the cEMI Server Object may hold the Properties as specified in Table 13.

Table 14 – Properties in the cEMI Server Object for the cEMI Server

Property Name	**PID**	**Description, remark**
PID_MEDIUM_TYPE	51	Media Type(s) supported by cEMI Server
PID_COMM_MODE	52	Data Link Layer / Raw (Busmonitor) / Transport L.
PID_MEDIUM_AVAILABILITY	53	Bus available (1) or not (0) ?
PID_ADD_INFO_TYPES	54	cEMI supported Additional Information Types
PID_TIME_BASE	55	Time base used in Extended relative timestamp.
PID_TRANSP_ENABLE	56	LL Transparency Mode of cEMI Server
PID_CLIENT_SNA	57	Reserved for cEMI Client’s Subnetwork Address.
PID_CLIENT_DEVICE_ADDRESS	58	Reserved for cEMI Client’s Device Address.
reserved	*61*	*DoA Filter, see 4.2.2.5.2*

For the specification of these Properties in the cEMI Server Object, please refer to [07]. Some additional requirements are given below. Please refer to [13] for the mandatory and optional Properties and the default, recommended access rights.

4.2.2.4.2 Communication Mode (Data Link Layer/Raw/Transport Layer) (PID_COMM_MODE) (PID = 52)

4.2.2.4.2.1 General requirements

- Property name: Communication Mode
- Property Datatype: PDT_ENUM8
- Datapoint Type: DPT_CommMode (DPT_ID = 20.1000)

The Property PID_COMM_MODE shall control the communication mode of the cEMI Server.

The Management Client shall set PID_COMM_MODE to change the communication mode of the cEMI Server. It shall also read PID_COMM_MODE to learn the current communication mode of the cEMI Server.

A change of the communication mode in the cEMI Server shall change the presentation of the frames from the cEMI Server to the cEMI Client. Every cEMI frame shall contain a Message Code; this Message Code shall be related 1-to-1 with the Communication Mode, except for the “Services for Local device management” (M_PropRead, M_PropWrite).

EXAMPLE 1 PID_COMM_MODE shall control whether a telegram received from the bus shall be presented to the cEMI Client as an *L_Raw.ind* or a *L_Data.ind* message.

In the direction from cEMI Client to the bus, the value of PID_COMM_MODE shall not affect anything. The ‘communication mode’ shall be included in every message received from the cEMI client: the communication layer is indicated by the used message code.

After a power on or a reset, the cEMI Server device may be in an undefined communication mode. If the cEMI Client prior to any other communication to or from the bus does not set the Property Communication Mode, then the behaviour of the cEMI Server is undefined.

If the Property is not present, the cEMI server shall only support Data Link Layer communication mode (this means it shall support the L_Data service) as the minimum requirement.

The value of the Property PID_COMM_MODE shall be formatted as an 8 bit enumeration datatype, identified as DPT_CommMode (see [09]), and shall be interpreted as specified Table 15.

The interpretation of PID_COMM_MODE depends on the device in which the cEMI Server is hosted and is specified in the following clauses.

Table 15 - Interpretation of DPT_CommMode of PID_COMM_MODE

Value	Communication Mode	Dest. Layer	Description
00h	Data Link Layer	LL	Data Link Layer
01h	Data Link Layer Busmonitor	LLB	Busmonitor [a]
02h	Data Link Layer Raw Frames	LLR	Data Link Layer Raw Frames
03h	not used	…	reserved for Network Layer
04h	not used	…	reserved for TL group oriented
05h	not used	…	reserved for TL connection oriented
06h	cEMI Transport Layer		establishes a connection to the cEMI Transport Layer
07h to EFh	not used	…	reserved for other ‘destination layers’
F0h to FEh	reserved		reserved for manufacturer specific use
FFh	“no layer”	No Layer	allowed as initial value of communication layer after a power up or after an M_Reset [b], [c]

[a] TP-UART implementations: used to put the TP-UART to “Busmonitor-Mode”. In this mode, the TP-UART is not able to send telegrams to the bus.

[b] A cEMI Client may set PID_COMM_MODE to FFh to disable the current Communication Mode and to activate the complete stack in the cEMI device again. The cEMI Server may afterwards set autonomously PID_COMM_MODE back to its default value.

EXAMPLE The cEMI Client may set PID_COMM_MODE to FFh to stop a cEMI Transport Layer connection.

If the cEMI Server is integrated in a KNX device with a full communication stack, this may enable again the communication of this full communication stack.

If the cEMI Server is part of an interface device (e.g. KNX USB interface) then the cEMI Server may afterwards autonomously set PID_COMM_MODE back to its default value, e.g. 00h “Data Link Layer”.

[c] **cEMI client minimum requirements**

The knowledge that a cEMI Server device maybe in an “undefined” communication mode after powerup/reset (value FFh, see above) leads to the following cEMI client minimum requirements:

- When the connection between a cEMI client and a cEMI Server is built up, the client shall always set the Property Communication Mode prior to any intended communication from/to the KNX bus.
- The same applies after the cEMI client detects a power up or a reset of the server device or if the client resets the server with the M_Reset service.

4.2.2.4.2.2 cEMI communication mode “cEMI Transport Layer”

cEMI Client sets PID_COMM_MODE to “cEMI Transport Layer”

The cEMI Client shall use the cEMI communication mode “cEMI Transport Layer” to activate the cEMI Transport Layer in the cEMI Server and establish a cEMI Transport Layer connection as specified in [04].

For host protocol dependent additional requirements, please refer to [15] for the KNX USB data interface and to [12] for (devices that use) KNXnet/IP Device Management.

4.2.2.4.3 cEMI Client Individual Address

Properties with PID = 57 and PID = 58 are reserved for future use as the cEMI client’s Individual Address (client SNA and Client Device Address), which may be needed to be held as copy in the cEMI Server.

4.2.2.5 Address Filtering

4.2.2.5.1 Group and LTE Addresses - Address Filter Table Object(s)

Filtering of (logical) Destination Addresses is an optional cEMI feature. Filtering in this context means, only frames in the direction from bus to cEMI client shall be filtered.

If filtering is supported, the address filter table(s) shall be structured as the Interface Objects used in the Routers. Please refer to [07].

If filtering is not supported, all group and LTE-addressed frames shall be sent to the cEMI client.

A cEMI Server to/from media with time critical L2-acknowledge (e.g. TP1) and without multicast address filtering shall send a non-selective L2-acknowledge to all multicast-addressed frames (AT-bit set to 1), i.e. independent of the multicast Destination Address.

If an address filter table Interface Object (for Group Addresses, LTE zone addresses or both) is present in the cEMI Server device, the corresponding Properties provided by these Interface Objects shall be used for features like ‘blocking of addresses in one direction’ or other. Please refer to [07] for more details.

(Default) Filter Mechanism

- Filter algorithms shall be the same as for the filtering in Routers.
- Addresses entered in the Filter Table shall be passed; all other addresses shall not be passed.

4.2.2.5.2 PL Domain Addresses (Informative)

It may be required in the future, that a cEMI Server device is a filter for PL Domain Addresses.

The intention is to put the PL Domain Address from which messages are to be routed to the cEMI client is put in a Property in the cEMI Server Object. If the value of the Property is set to zero, then there shall be no filtering.

Current cEMI Server implementations do not foresee the PL Domain Address filter mechanism. If an applicant wants to implement a DoA filter mechanism, he is pleased to contact the KNX Association System Department for further procedure.

4.2.2.5.3 Filtering in function of RF Domain Address and KNX Serial Numbers

The cEMI Server shall filter received RF frames in function of:

- RF Domain Address, according to PID_RF_DOMAIN_ADDRESS in the Device Object, and
- KNX Serial Numbers, according to PID_SERIAL_NR_TABLE in the Device Object.

The filtering mode shall be selected through the Property PID_RF_FILTERING_MODE.

4.2.2.6 Local Loading of applications (informative)

Today’s well known BCU technology uses EMI interface devices (RS232) that can be plugged on a BCU. An application can then be loaded locally into the target device’s BCU. After download the EMI interface device is unplugged and an Application Module, e.g. a push button unit, is plugged onto the BCU.

For the future, a similar combination of a BAU device and an interface device (cEMI Server) is possible, too. For such a configuration the application relevant Interface Objects shall be loaded using the management services specified in 4.1.7 “Services for Local device management”.

Application relevant Interface Objects are:

- Group Address Table Object;
- Association Table Object;
- Application Program Object;
- Group Object Table Object;
- Application specific Interface Objects.

Please refer to [07] for a detailed specification of the above mentioned System Interface Objects.

4.3 KNX extraneous information on cEMI connection (informative)

4.3.1 General requirements

It is needed for some applications and implementations that information between the cEMI Server and cEMI client can also transfer other protocols than only KNX, e.g. the metering protocol (according EN 13757-3 and EN 13757-4), that uses the same Physical Layer on the RF medium as KNX RF does.

If any cEMI Client / Server combination is intended to handle also such KNX extraneous information, a *protocol identifier* shall be used in the cEMI host protocol's header.

The protocol identifier is usually part of a cEMI's host protocol specification.

Example: cEMI hosted on the KNX USB protocol
There shall be a dedicated separate field *Protocol ID* in the *KNX USB Transfer Protocol Header* in the *KNX HID Report Body*.
Please refer to [15] for more information.

As a more simple kind of a protocol separation, a *protocol identifier* can be placed in front of the cEMI's message code octet, see following 4.3.2 "Metering protocol".

4.3.2 Metering protocol

The following frame structure shows an **example** of an implementation how the Metering Protocol frames may be transferred on the cEMI connection link.

"Host Protocol's Header"		"Host Protocol's Body"	
Buffer Length	Protocol Identifier	Message Code	cEMI frame body (KNX) or KNX extraneous information
PL	PI	MC	
1 octet	1 octet	1 octet	var. length

The "Host protocol´s" minimum header information consists of the buffer length (typically the "Host Protocol´s Body" length) and the protocol identifier.

In case of KNX extraneous protocols the contents in the "Host Protocol's Body" are free. If there's a message code or not in the KNX extraneous protocols, is not in the scope of this common EMI (KNX) specification. If a message code is used as the 1st octet, the values can be chosen independently of the KNX cEMI message code set.

4.4 Further requirements

4.4.1 Quality of Service

It shall be assured that existing EMI and device implementations (e.g. BCU 1 and BCU 2) can be mapped (by a function) to cEMI. Note however that this task has to be performed by the component implementing the cEMI Server.

Especially for the management services, it may not only be important if a service request can be executed, but which side effects this service execution has. The explicit definition of the management services shall allow execution without switching modes, leaving the current non-management state and service execution unaffected.

However, for some existing and or upcoming device implementations, such behaviour can not be provided. Instead, executing management services e.g. reinitialises queues or disables parts of the bus communication like a local Transport Layer connection.

The cEMI Server shall provide adequate warning on such execution behaviour

- during the description and
- at runtime with a specific return code ("success with info").

ANNEX A

(informative)

Overview on EMI 2 & cEMI Message Codes

The following page gives an overview on the message codes used for EMI 2, cEMI.

In addition, additional codes are shown (reserved) for new services, e.g. system broadcast service and Transport Layer services for extended frame format/LTE group messages.

Legend

	bold letters	Message Codes used for cEMI messages
▨	grey background fields	Messages Codes used by EMI 2
▨	**grey background & bold letters**	LL Messages Codes used by EMI 2 & cEMI
🟥	red background	(Old) EMI 2 Services → Redefined
🟩	green background	Redefined EMI 2 Services
🟦	blue background	reserved for the extended frame format TL services (LTE Tag Group data)
🟦	light blue background	Message Codes for System Broadcast Services, as defined in Suppl. No. 16 of KNX System Spec.

NOTE The message codes in the following table are only for reference. Please refer to Table 1 for the list of message codes.

high nibble	low nibble															
	0h	1h	2h	3h	4h	5h	6h	7h	8h	9h	Ah	Bh	Ch	Dh	Eh	Fh
0h																
1h	L_raw.req L_Plain_Data.req	L_Data.req		L_Poll_Data.req				L_SystemBroadcast.req								
2h		N_Data_Individual.req	N_Data_Group.req	N_Poll_Data.req		L_Poll_Data.con	L_SystemBroadcast.con		L_SystemBroadcast.ind	L_Data.ind		L_Busmon.ind	N_Data_Broadcast.req	L_Raw.ind	L_Data.con	L_Raw.con
3h			T_Data_Group.req	T_Poll_Data.req		N_Poll_Data.con			T_Data_Tag_Group.req		N_Data_Group.ind				N_Data_Group.con	
4h		T_Data_Connected.req		T_Connect.req	T_Disconnect.req		T_LOCAL.request			N_Data_Individual.ind	T_Data_individual.req		T_Data_Broadcast.req	N_Data_Broadcast.ind	N_Data_Individual.con	N_Data_Broadcast.con
5h																
6h																
7h		U_Value_Write.req	A_Data_Group.req	A_Poll_Data.req	U_Value_Read.req	T_Poll_Data.con		T_Data_tag_Group.ind	T_Data_tag_Group.con		T_Data_Group.ind		U_Flags_Read.req		T_Data_Group.con	
8h		M_User_Data_Individual.req	M_User_Data_Connected.req		T_Data_Individual.ind	T_Connect.ind	T_Connect.con	T_Disconnect.ind	T_Disconnect.con	T_Data_Connected.ind	M_InterfaceObj_Data.req		T_Data_Individual.con	T_Data_Broadcast.ind	T_Data_Connected.con	T_Data_Broadcast.con
9h					T_Data_Individual.ind						M_InteraceObj_Data.req		T_Data_Individual.con			
Ah							PC_Set_Value.req	PEI_Identify.req	PEI_Identify.con	PEI_Switch.req		PC_Get_Value.con	PC_Get_Value.req			
Bh																
Ch	TM_Timer.ind															
Dh		M_User_Data_Connected.con	M_User_Data_Connected.ind		M_Interface_Obj_Data.ind	M_Connect.ind		M_Disconnect.ind		M_user_Data_Individual.ind			M_Interface_Obj_Data.con		M_User_Data_Individual.con	
Eh					U_Value_Read.con	A_Poll_Data.con		U_Event.ind			A_Data_Group.ind		U_Flags_Read.con		A_Data_Group.con	
Fh	M_Reset.ind	M_Reset.req				M_PropWrite.con	M_PropWrite.req	M_PropInfo.ind				M_PropRead.con	M_PropRead.req			