Document Title	Specification of I-PDU Multiplexer
Document Category	Software specification (SWS)
Document UID	182
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R26-11

Document Change History

Date	Release	Changed by	Description
2026-11-26	R26-11	AUTOSAR Release Management	• Added support for DMA transfer approach • Removed IpduM_GetVersionInfo() • Removed PduR bypass option • Editorial changes
2025-11-27	R25-11	AUTOSAR Release Management	• Introduced "maximum fill-up strategy" for ContanerPdus • Clarified trigger conditions to transmit a ContainerIpdu • Editorial changes
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes
2023-11-23	R23-11	AUTOSAR Release Management	• Added information about automatic handle IDs to configuration • Changed document name to include “CP”
2022-11-24	R22-11	AUTOSAR Release Management	• Improved retransmission of PDUs • Clarified priority handling for contained PDUs • Fixed error tables • Converted to LaTeX
2021-11-25	R21-11	AUTOSAR Release Management	• Rework of assembly strategy of a ContainerIpdu that contains ContainedIpdus with Last-is-Best semantic • Shorting of IPDUs if possible • Avoid possible interpretation of queue bypass for containers

▽

△

2020-11-30	R20-11	AUTOSAR Release Management	• Improved the structure of the Error sections • Clarified `IpduMRxDynamicPart` lower multiplicity
2019-11-28	R19-11	AUTOSAR Release Management	• Introduced BSW Distribution Concept (CONC 643) as draft • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Introduce priority for Tx ContainedIPdus with LastIsBest collection semantics • Header File Cleanup • Limitations on Container PDU with MDT • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2017-12-08	4.3.1	AUTOSAR Release Management	• introduced static multiple-PDU-to-Container handling as draft • revised error definitions • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2016-11-30	4.3.0	AUTOSAR Release Management	• updated TX-confirmation handling • added support for MetaData • Com-Stack API harmonization • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2015-07-31	4.2.2	AUTOSAR Release Management	• minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2014-10-31	4.2.1	AUTOSAR Release Management	• Added Multiple PDU to Container Mapping • Extension of `IpduMSelectorFieldLength`
2014-03-31	4.1.3	AUTOSAR Release Management	• Editorial changes and minor corrections • No major functional change

▽

△

2013-10-31	4.1.2	AUTOSAR Release Management	• Revised configuration structure of dynamic and static segments to enforce layout constraints already by the configuration structure • Few bug fixes and clarifications • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Reworked according to the new SWS BSW General, harmonization of post-build configuration • Allowing reception of nothing but the static part
2011-12-22	4.0.3	AUTOSAR Administration	• Minor bug fixes and editorial changes • Added configurable JIT-update
2010-09-30	3.1.5	AUTOSAR Administration	• Updated: tables for mandatory and optional interfaces, [SWS_IpduM_00020], [SWS_IpduM_00027], [SWS_IpduM_00028], [SWS_IpduM_00032], [SWS_IpduM_00060], [SWS_IpduM_00068], [SWS_IpduM_00083], [SWS_IpduM_00104], [ECUC_IpduM_00112], [IPDUM117_Conf], [SWS_IpduM_00143] and [IPDUM154] • Removed: [IPDUM013], [IPDUM030], [IPDUM050_Conf], [IPDUM051_Conf], [IPDUM063], [IPDUM064], [IPDUM065], [IPDUM072], [IPDUM099] and [IPDUM154] • Added: re-compile configuration variant (Chapter 10), [ECUC_IpduM_00162], [ECUC_IpduM_00163], [ECUC_IpudM_00164] and [SWS_IpduM_00165]

▽

△

2010-02-02	3.1.4	AUTOSAR Administration	• Converted to LaTeX • Harmonization of FIBEX multiplexing and AUTOSAR multiplexing • Many small corrections based on conformance tests and validation activities • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	• Fixed generated figures and captions
2007-12-21	3.0.1	AUTOSAR Administration	• SWS improvements by AUTOSAR Technical Office • Defined maximum I-PDU size for FlexRay to 254 bytes • Document meta information extended • Small layout adaptations made
2007-01-24	2.1.15	AUTOSAR Administration	• Integrated into BSW Scheduler header file struture • Sequence diagrams clarified • Superfluous text removed • Maximum IPDU size clarified • Signature for `IpduM_Transmit` made consistent with rest of stack. • “Advice for users” revised • “Revision Inforawion” added • Legal disclaimer revised
2006-05-16	2.0	AUTOSAR Administration	• Initial release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction and functional overview 10
2 Acronyms and Abbreviations 11
3 Related documentation 12
3.1 Input documents & related standards and norms . . . 12
3.2 Related specification . . . 12
4 Constraints and assumptions 13
4.1 Limitations . . . 13
4.2 Applicability to car domains . . . 13
4.3 Applicability to safety related environments . . . 13
5 Dependencies to other modules 14
5.1 AUTOSAR OS . . . 14
5.2 RTE (BSW Scheduler) . . . 15
5.3 PDU Router . . . 15
5.4 COM . . . 15
5.5 File structure . . . 16
5.5.1 Code file structure . . . 16
6 Requirements Tracing 17
7 Functional specification 18
7.1 General . . . 18
7.2 I-PDU Multiplexing . . . 18
7.2.1 Definitions and Layout . . . 18
7.2.2 General . . . 20
7.2.3 Initialization . . . 20
7.2.4 Transmission . . . 21
7.2.4.1 Transmission request . . . 22
7.2.4.2 Transmission trigger . . . 22
7.2.4.3 Just-In-Time update of parts . . . 23
7.2.4.4 Transmission confirmation . . . 24
7.2.5 Reception . . . 24
7.2.6 Metadata handling . . . 25
7.3 Multiple-PDU-to-Container handling . . . 25
7.3.1 Dynamic Container Layout . . . 26
7.3.2 Static Container Layout . . . 27
7.3.3 Transmission . . . 27
7.3.3.1 Transmission request . . . 28
7.3.3.2 Transmission trigger condition . . . 28
7.3.3.3 Assembly of Container PDUs . . . 31
7.3.3.4 Transmission confirmation . . . 33

7.3.3.5 Queueing . . . 34
7.3.3.6 Last-is-Best collection semantics . . . 35
7.3.4 Transmission of Dynamic Containers . . . 36
7.3.4.1 Queued collection semantics . . . 36
7.3.4.2 Triggered Transmission and Last-is-Best semantics . . . 37
7.3.4.2.1 Order and re-sorting . . . 38
7.3.5 Transmission of Static Containers . . . 40
7.3.6 Reception . . . 41
7.3.6.1 Queueing . . . 42
7.3.7 Reception of Dynamic Containers . . . 42
7.3.8 Reception of Static Containers . . . 44
7.3.9 Errorhandling . . . 44
7.3.10 Metadata handling . . . 45
7.3.11 Local buffer asynchronous copy approach . . . 45
7.4 Basic Software Distribution . . . 47
7.5 Error Classification . . . 47
7.5.1 Development Errors . . . 47
7.5.2 Runtime Errors . . . 48
7.5.3 Production Errors . . . 48
8 API specification 49
8.1 Imported types . . . 49
8.2 Type definitions . . . 49
8.2.1 IpduM_ConfigType . . . 49
8.3 Function definitions . . . 50
8.3.1 IpduM_Init . . . 50
8.3.2 IpduM_Transmit . . . 51
8.3.3 IpduM_RxIndication . . . 51
8.3.4 IpduM_TxConfirmation . . . 52
8.3.5 IpduM_TriggerTransmit . . . 53
8.4 Scheduled functions . . . 54
8.5 Expected interfaces . . . 55
8.5.1 Mandatory Interfaces . . . 55
8.5.2 Optional Interfaces . . . 55
8.5.3 Callout Interfaces . . . 56
8.5.4 Configurable interfaces . . . 56
9 Sequence diagrams 57
9.1 Transmission of a multiplexed I-PDU and Transmit confirmation . . . 57
9.2 Transmission of a multiplexed I-PDU without Trigger . . . 59
9.3 Reception of the multiplexed I-PDU . . . 59
9.4 Trigger Transmit . . . 60

10 Configuration specification 62
10.1 How to read this chapter . . . 62
10.2 Containers and configuration parameters . . . 62
10.2.1 Configuration overview . . . 63
10.2.2 IpduM . . . 63
10.2.3 IpduMConfig . . . 64
10.2.4 IpduMGeneral . . . 66
10.2.5 IpduMTxPathway . . . 70
10.2.6 IpduMTxRequest . . . 73
10.2.7 IpduMTxDynamicPart . . . 75
10.2.8 IpduMTxDynamicSegment . . . 77
10.2.9 IpduMTxStaticPart . . . 78
10.2.10 IpduMTxStaticSegment . . . 80
10.2.11 IpduMTxSelectorField . . . 80
10.2.12 IpduMRxPathway . . . 81
10.2.13 IpduMRxIndication . . . 82
10.2.14 IpduMRxDynamicPart . . . 84
10.2.15 IpduMRxDynamicSegment . . . 86
10.2.16 IpduMRxStaticPart . . . 87
10.2.17 IpduMRxStaticSegment . . . 88
10.2.18 IpduMRxSelectorField . . . 88
10.2.19 IpduMContainerRxPdu . . . 91
10.2.20 IpduMContainedRxPdu . . . 94
10.2.21 IpduMContainerTxPdu . . . 98
10.2.22 IpduMContainedTxPdu . . . 103
10.2.23 IpduMMainFunctionRx . . . 107
10.2.24 IpduMMainFunctionTx . . . 108
10.3 Published Information . . . 109
10.4 Configuration Rules . . . 109
10.4.1 Selector Field . . . 109
10.4.2 Placement of static and dynamic parts . . . 110
10.4.3 Multiple PDU to Container Mapping . . . 110
10.4.4 Priority . . . 112
A Not applicable requirements 113
B Change History of AUTOSAR Traceable Items 114
B.1 Traceable Item History of this Document According to AUTOSAR Release R26-11 . . . 114
B.1.1 Added Specification Items in R26-11 . . . 114
B.1.2 Changed Specification Items in R26-11 . . . 114
B.1.3 Deleted Specification Items in R26-11 . . . 114
B.1.4 Added Constraints in R26-11 . . . 114
B.1.5 Changed Constraints in R26-11 . . . 115

B.1.6 Deleted Constraints in R26-11 . . . 115
B.2 Traceable Item History of this Document According to AUTOSAR Release R25-11 . . . 115
B.2.1 Added Specification Items in R25-11 . . . 115
B.2.2 Changed Specification Items in R25-11 . . . 115
B.2.3 Deleted Specification Items in R25-11 . . . 115
B.2.4 Added Constraints in R25-11 . . . 115
B.2.5 Changed Constraints in R25-11 . . . 115
B.2.6 Deleted Constraints in R25-11 . . . 115
B.3 Traceable Item History of this Document According to AUTOSAR Release R24-11 . . . 116
B.3.1 Added Specification Items in R24-11 . . . 116
B.3.2 Changed Specification Items in R24-11 . . . 116
B.3.3 Deleted Specification Items in R24-11 . . . 116
B.3.4 Added Constraints in R24-11 . . . 116
B.3.5 Changed Constraints in R24-11 . . . 116
B.3.6 Deleted Constraints in R24-11 . . . 116
B.4 Traceable Item History of this Document According to AUTOSAR Release R23-11 . . . 116
B.4.1 Added Specification Items in R23-11 . . . 116
B.4.2 Changed Specification Items in R23-11 . . . 116
B.4.3 Deleted Specification Items in R23-11 . . . 117
B.4.4 Added Constraints in R23-11 . . . 117
B.4.5 Changed Constraints in R23-11 . . . 117
B.4.6 Deleted Constraints in R23-11 . . . 117

1 Introduction and functional overview

This specification describes the functionality, APIs and the configuration of the AUTOSAR Basic Software module I-PDU Multiplexer IpduM.

PDU multiplexing means using the same PCI (Protocol Control Information) of a PDU (Protocol Data Unit) with more than one unique layout of its SDU (Service Data Unit). A selector field is a piece of the SDU of the multiplexed PDU. It is used to distinguish the contents of the multiplexed PDUs from each other.

Multiplexing of PDUs is currently known from CAN, but is not restricted to this communication system.

On sender-side, the I-PDU Multiplexer module is responsible to combine appropriate I-PDUs from COM to new, multiplexed I-PDUs and send them back to the PDU Router. On receiver-side, it is responsible to interpret the content of multiplexed I-PDUs and provide COM with its appropriate separated I-PDUs taking into account the value of the selector field.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the I-PDU Multiplexer module that are not included in the [1, AUTOSAR glossary].

Abbreviation / Acronym:	Description:
COM I-PDU	I-PDU assembled in the COM module out of COM Signals
contained I-PDU	I-PDU assembled into or extracted from a Container PDU
Container PDU	PDU containing I-PDUs and headers
dynamic part	see [2]
instance of an I-PDU	IpduM I-PDU with one specific layout and content
Instances of a Container	Instances of the same Container PDU
IpduM	I-PDU Multiplexer
IpduM I-PDU	I-PDU assembled in the IpduM module out of two COM I-PDUs
multiplexed I-PDU	see IpduM I-PDU
segment	The static or dynamic part may consist of more than one piece. These pieces are called segments. See also Chapter 7.2.1 and Figure 7.1.
selector field	see [2]
signal	see [3]
signal group	see [3]
static part	see [2]

Table 2.1: Descriptions of abbreviations and acronyms used in this document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Glossary
AUTOSAR_FO_TR_Glossary

[2] Specification of I-PDU Multiplexer
AUTOSAR_CP_SWS_IPDUMultiplexer

[3] Specification of Communication
AUTOSAR_CP_SWS_COM

[4] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[5] Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 1: General requirements (see Functional Safety and IEC 61508)

[6] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture

[7] Specification of RTE Software
AUTOSAR_CP_SWS_RTE

[8] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[9] Specification of Diagnostic Event Manager
AUTOSAR_CP_SWS_DiagnosticEventManager

[10] Specification of PDU Router
AUTOSAR_CP_SWS_PDURouter

[11] System Template
AUTOSAR_CP_TPS_SystemTemplate

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [4, SWS BSW General], which is also valid for I-PDU Multiplexer.

Thus, the specification SWS BSW General shall be considered as additional and required specification for I-PDU Multiplexer.

4 Constraints and assumptions

4.1 Limitations

For transmission of multiplexed I-PDUs, minimum delay time observation cannot be taken into account. For more details, see [3] and Chapter 7.2.4.1.

For transmission of container PDUs with static layout, minimum delay time cannot be ensured if two or more contained PDUs have each MDT configuration.

4.2 Applicability to car domains

No restrictions.

4.3 Applicability to safety related environments

This document has been created in absence of a safety case and a safety plan. Thus, the direct results of this document can only be used within safety relevant systems after repeating certain process steps as required in the [5, IEC 61508].

5 Dependencies to other modules

This chapter lists all the features from other modules that are used by the AUTOSAR IpduM and functionalities that are provided by AUTOSAR IpduM to other modules.

Because the IpduM module deals with PDUs that are either sourced or sunk by other modules, care must be taken that shared configuration items are consistent between the modules.

The IpduM is arranged next to the PDU Router in the layered architecture of AUTOSAR[6]; see Figure 5.1.

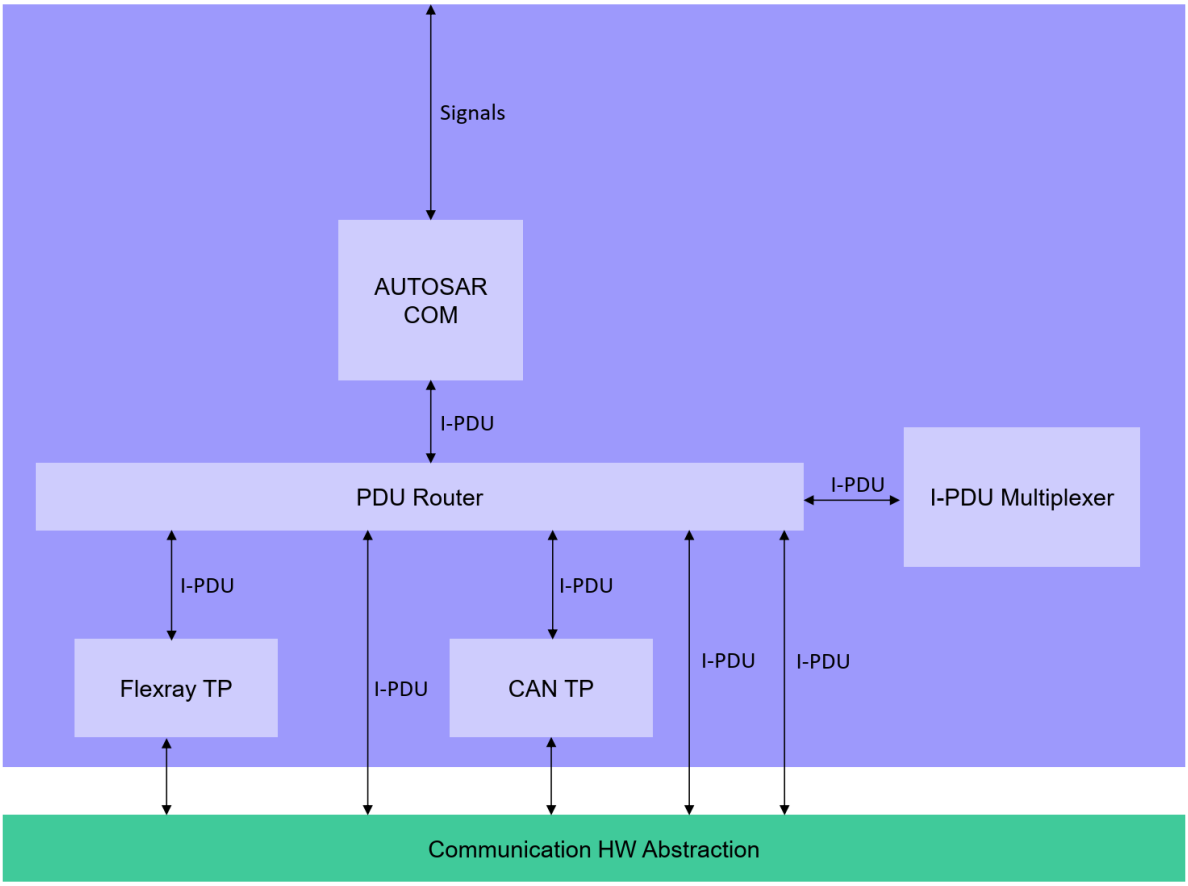


Figure 5.1: I-PDU Multiplexer in the AUTOSAR Architecture

5.1 AUTOSAR OS

[SWS_IpduM_00107]

Upstream requirements: SRS_BSW_00429

⌈The IpduM shall not directly access the AUTOSAR OS.⌋

5.2 RTE (BSW Scheduler)

The RTE includes the BSW-Scheduler (see [7]).

The IpduM module relies on the BSW-scheduler calling `IpduM_MainFunctionRx` and `IpduM_MainFunctionTx` at a period as configured in `IpduMMainRxTimeBase` or `IpduMMainTxTimeBase` respectively.

5.3 PDU Router

The following summarizes the functionality IpduM needs from the PDU Router (for more details see Chapter 8.5):

- indication of incoming multiplexed or contained I-PDUs
- sending interface for outgoing I-PDUs (Container or Multiplexed PDUs)
- confirmation of I-PDUs which went out

The following list summarizes the functionality provided by the IpduM module for the PDU Router module:

- indication interface for incoming I-PDUs, which are de-multiplexed and for incoming Container-PDUs, which are to be disassembled
- sending interface for to be multiplexed I-PDUs and I-PDUs, which are to be assembled into a Container PDU
- confirmation interface for transmitted I-PDUs

The configuration of the PDU Router module (e.g. look-up tables) must be such that the I-PDUs, which belong to multiplexed I-PDUs and represent a static or a dynamic part of a multiplexed I-PDU, are routed to the IpduM module.

The configuration of the PDU-Router module (e.g. look-up tables) must be such that the relevant I-PDUs are routed to IpduM. These are:

- I-PDUs, which belong to multiplexed I-PDUs and represent a static or a dynamic part of a multiplexed I-PDU
- I-PDUs, which consist of static and dynamic parts to be de-multiplexed
- I-PDUs, which are to be assembled into a Container PDU
- Container PDUs to be disassembled

5.4 COM

The configuration of the IpduM module relies on a corresponding configuration of the AUTOSAR COM module. For each multiplexed I-PDU, there need to be different I-

PDUs for the static part and each layout of the dynamic part. For further information configured in the COM module, see Chapter 7.1 and especially Figure 7.1.

The IpduM further assumes that the correct selector field values are already contained in the COM's modules I-PDU representing the dynamic parts. See also [SWS_IpduM_00098].

The configuration of Container PDUs/ contained I-PDUs does not depend on the COM configuration.

5.5 File structure

5.5.1 Code file structure

This IpduM SWS does not define the code file structure completely.

6 Requirements Tracing

The following table references requirements specified in [8] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[SRS_BSW_00009]	All Basic SW Modules shall be documented according to a common standard.	[SWS_IpduM_00104] [SWS_IpduM_00105]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_IpduM_00032] [SWS_IpduM_00033]
[SRS_BSW_00323]	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	[SWS_IpduM_00028]
[SRS_BSW_00337]	Classification of development errors	[SWS_IpduM_91003]
[SRS_BSW_00344]	BSW Modules shall support link-time configuration	[SWS_IpduM_00032]
[SRS_BSW_00357]	For success/failure of an API call a standard return type shall be defined	[SWS_IpduM_00102]
[SRS_BSW_00369]	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	[SWS_IpduM_00032] [SWS_IpduM_00040] [SWS_IpduM_00043] [SWS_IpduM_00044] [SWS_IpduM_00060]
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[SWS_IpduM_00032]
[SRS_BSW_00406]	API handling in uninitialized state	[SWS_IpduM_00084]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[SWS_IpduM_91003]
[SRS_BSW_00425]	The BSW module description template shall provide means to model the defined trigger conditions of schedulable objects	[SWS_IpduM_00252]
[SRS_BSW_00429]	Access to OS is restricted	[SWS_IpduM_00107]
[SRS_BSW_00438]	Configuration data shall be defined in a structure	[SWS_IpduM_00159]
[SRS_BSW_00478]	Timing limits of main functions	[SWS_IpduM_00252]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 General

There are two different approaches of multiplexing several I-PDUs into one resulting PDU being transferred on the bus:

I-PDU Multiplexing means using the same I-PDU ID transferred from the PDU Router to the Communication Hardware Abstraction Layer with more than one unique layout of this I-PDU; see also [6].

Multiple PDU to Container Mapping means collecting several I-PDUs into one Container PDU. This Container PDU is then transferred via PduR as one (large) I-PDU. This way advantage of the larger frame sizes of newer bus systems can be taken, allowing an efficient usage of the bandwidth in combination with smaller I-PDU sizes (usually 8 bytes).

[SWS_IpduM_00097] ⌈The IpduM shall be implemented so that no other modules depend on it and that it is be possible to build a system without the IpduM module if it is not needed.⌋

7.2 I-PDU Multiplexing

7.2.1 Definitions and Layout

A multiplexed I-PDU consists of a static part and a dynamic part, where the static part consists of zero or more signals or signal groups. The dynamic part consists of the selector field and one or more signals or signal groups; see Figure 7.1.

The dynamic part of an I-PDU is comparable with a union of the programming language C. Depending on the value of the selector field inside the I-PDU, the actual layout of the I-PDU is selected.

The position of the static and the dynamic part are configurable per I-PDU. The static and the dynamic part can be subdivided into different segments.

Only one selector field can be defined for each multiplexed I-PDU. The value of the selector field defines how the content of the dynamic part of the I-PDU will be interpreted. The selector field has a configurable size between 1 and 16 contiguous bits and its position can be defined by configuration, see [ECUC_IpduM_00221] and [ECUC_IpduM_00222].

See Chapter 10.2.1 for an overview of the IpduM configuration. Chapter 10.4 defines the configuration rules.

Multiplexing of PDUs is originally known from CAN, but it is not restricted to this communication system. The IpduM is layered next to the PDU Router above the interface layer (Communication Hardware Abstraction) in the AUTOSAR layer architecture and

therefore this feature could be used for all bus systems, which can be handled by the PDU Router, for example FlexRay.

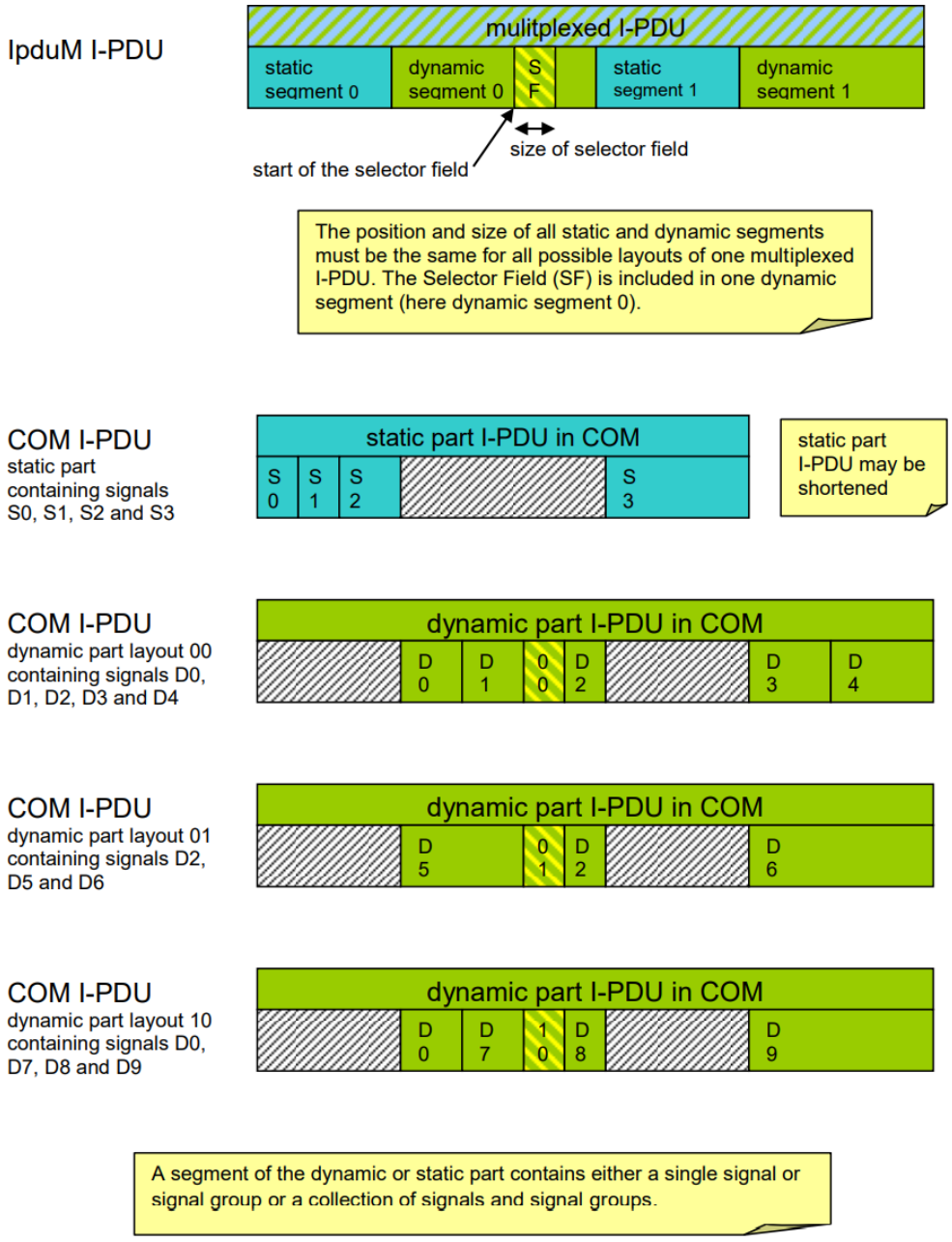


Figure 7.1: Possible layout of a multiplexed I-PDU with shortened static part

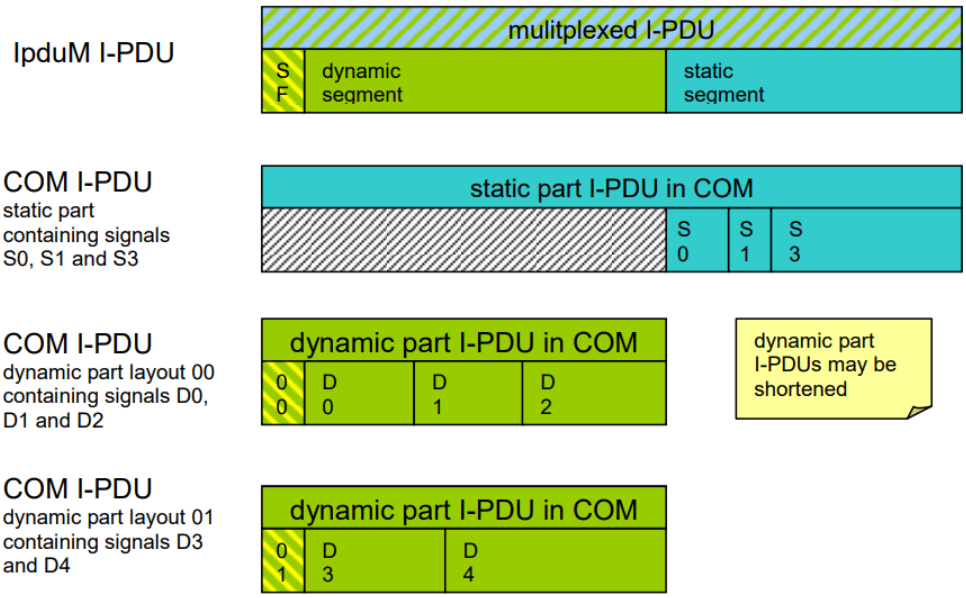


Figure 7.2: Possible layout of a multiplexed I-PDU with shortened dynamic parts

7.2.2 General

There is one COM I-PDU for the static part and one COM I-PDU for each layout of the dynamic part of one multiplexed IpduM I-PDU, so the IpduM combines at most two I-PDUs of COM.

[SWS_IpduM_00098] ⌈The IpduM module shall not set the selector field.⌋

The IpduM module relies on the configuration of the COM module. For each dynamic layout, an I-PDU needs to be configured in COM. Such I-PDUs already have to contain the correct selector field value. The selector field values in COM can be initialized by configuring them as signals that are initialized with an init value but are never written after initialization.

[SWS_IpduM_00173] ⌈The IpduM shall respect the `IpduMByteOrder` when interpreting the selector field value.⌋

For a detailed description of the transmission and reception of a multiplexed I-PDU see Chapter 7.2.4 and 7.2.5.

7.2.3 Initialization

The IpduM module provides an initialization function `IpduM_Init` defined in [SWS_IpduM_00032]. This function initializes all internal global variables and the buffers of the IpduM I-PDUs. For more details, see Chapter 8.3.1.

The environment of the IpduM shall call IpduM_Init before calling any other function of the IpduM module.

The implementer has to ensure that IPDUM_E_UNINIT is returned in development mode in case an API function (except IpduM_MainFunctionTx and IpduM_MainFunctionRx) is called before the module is initialized.

For the I-PDU data transmission pathway through the IpduM module, a buffer is allocated inside the IpduM module. This buffer needs to be initialized because it might be transmitted before it has been fully populated with data by the COM module. The initialization data of this buffer is derived from the initial values of the COM module's configuration as follows:

1. **[SWS_IpduM_00067]** ⌈The IpduM shall initialize its internal transmit buffers with the configured pattern IpduMIPduUnusedAreasDefault.⌋

2. **[SWS_IpduM_00068]** ⌈The initial signal values of the initial dynamic part shall be set according to initial values of the referenced COM I-PDU (IpduMInitialDynamicPart -> IpduMTxDynamicPart -> IpduMTxDynamicPduRef).⌋

3. **[SWS_IpduM_00143]** ⌈The initial signal values of the static part shall be set according to the intial values of the referenced COM I-PDU (IpduMTxStaticPart -> IpduMTxStaticPduRef).⌋

The selector field is contained within one segment of the intial dynamic part and therefore is initialized implicitly.

For optimization, the initial bit pattern for the buffer can be worked out at configuration-time and then copied at run-time.

7.2.4 Transmission

Inside COM, there are separated I-PDUs for the static part and one for each dynamic part of a multiplexed I-PDU.

The static part and the dynamic parts are treated in COM as separate I-PDUs with their own I-PDU IDs.

[SWS_IpduM_00015] ⌈For a multiplexed I-PDU IpduM shall merge the corresponding two COM I-PDUs representing the associated static part and the last received dynamic part into one single IpduM I-PDU with a new unique I-PDU ID. IpduM shall send out this new IpduM I-PDU to the PDU Router module.⌋

See also Figure 5.1.

For details about the trigger of the transmission, see Chapter 7.2.4.2.

All control functionalities like deadline monitoring of the COM I-PDUs and update-bit evaluation are out of the scope of the IpduM and have to be done by the COM layer. For details about the timing-behavior of the new combined I-PDU see Chapter 7.2.4.2.

7.2.4.1 Transmission request

The IpduM module provides an `IpduM_Transmit` function so that the PDU Router is able to initiate the transmission of an I-PDU; see [SWS_IpduM_00043].

[SWS_IpduM_00017] ⌈The function `IpduM_Transmit` shall assemble the multiplexed I-PDU, using the related static and dynamic part, and transmit it according to the trigger conditions/ modes as defined in [SWS_IpduM_00021] and [ECUC_IpduM_00125].⌋

As defined in Chapter 7.2.3, each outgoing I-PDU has an initial value so that, should an I-PDU be transmitted by the IpduM module before both static and dynamic parts have been sent from COM to the IpduM, a value defined by the configuration is transmitted.

[SWS_IpduM_00152] ⌈As long as no transmission confirmation for the IpduM I-PDU is received (regardless of the result), the function `IpduM_Transmit` shall return `E_NOT_OK` for any new transmission request from the upper layer with a COM I-PDU belonging to the same IpduM I-PDU.⌋

The IpduM module relies here on a configured transmission configuration in the lower layer.

In case a multiplexed I-PDU is only triggered for sending by either updating the dynamic or static part, the non-triggering part might be overwritten if updated multiple times between two transmissions.

7.2.4.2 Transmission trigger

The IpduM module receives the static and the dynamic part of a multiplexed I-PDU by separated two transmission requests as two single COM I-PDUs from the PDU Router module.

[SWS_IpduM_00021] ⌈The IpduM module shall be configurable to send a transmission request for the new multiplexed I-PDU to the PDU Router because of the following trigger conditions/modes:

- receiving a static part
- receiving a dynamic part
- receiving a static or a dynamic part

- does not trigger transmission because of receiving anything of this I-PDU (IpduMTxTriggerMode NONE) in case of TriggerTransmit

For configuration, see [ECUC_IpduM_00052].⌋

The four trigger conditions/ modes defined by [SWS_IpduM_00021] allow controlling the transmission mode of the new assembled I-PDU by the transmission modes of the single I-PDUs sent by COM, see also [3].

Not all of four trigger conditions/ modes defined by [SWS_IpduM_00021] guarantee the minimum delay time between consecutive transmissions of different instances of multiplexed I-PDUs, because if the transmission is triggered by static and dynamic part or only by the dynamic part, COM does not take care for the minimum delay time. COM treats the static part and the different dynamic parts as unrelated stand-alone I-PDUs.

The configuration "does not trigger transmission because of receiving anything" is needed if an I-PDU is only sent out because of a TriggerTransmit of a lower layer. With the API IpduM_TriggerTransmit it is possible for lower layers to trigger a send out of an I-PDU.

In case the IpduMTxTriggerMode is NONE and the lower layer triggers the transmission via IpduM_TriggerTransmit, the IpduMTxConfirmationPduId needs to be configured since this ID is also used for resolving the I-PDU in case of IpduM_TriggerTransmit, see also [ECUC_IpduM_00158].

7.2.4.3 Just-In-Time update of parts

Sometimes it may be unwanted that the IpduM module not just sends out the locally stored parts, since these parts may contain outdated information e.g. update-bits. Therefore, the IpduM supports a per part configurable just-in-time update mechanism.

[SWS_IpduM_00168] ⌈In case the transmission of a multiplexed I-PDU is triggered by the update of one part and IpduMJitUpdate is configured to true for the second part, the IpduM module shall update the second part via PduR_IpduMTriggerTransmit before the multiplexed I-PDU is sent out via PduR_IpduMTransmit.⌋

[SWS_IpduM_00169] ⌈In case the contents of a multiplexed I-PDU is requested via IpduM_TriggerTransmit, the IpduM module shall update all parts which have IpduMJitUpdate configured to true before returning the contents of the multiplexed I-PDU.⌋

[SWS_IpduM_00223] ⌈In case the IpduM shall update the dynamic part just-in-time, the latest dynamic part sent by the upper layer shall be updated or the dynamic part referenced by IpduMInitialDynamicPart if no dynamic part was sent before.⌋

[SWS_IpduM_00171] ⌈In case the transmission of a multiplexed I-PDU is triggered by the update of one part and IpduMJitUpdate is configured to true for the sec-

ond part, the multiplexed I-PDU shall not be send if the JIT-update request via `PduR_IpduMTriggerTransmit` returns `E_NOT_OK`.⌋

[SWS_IpduM_00172] ⌈In case the contents of a multiplexed I-PDU is requested via `IpduM_TriggerTransmit` and `IpduMJitUpdate` is configured to true for any multiplexed part, `IpduM_TriggerTransmit` shall return `E_NOT_OK` if any of the JIT-update requests via `PduR_IpduMTriggerTransmit` return `E_NOT_OK`.⌋

7.2.4.4 Transmission confirmation

Transmission confirmations are given to the IpduM module by the PDU Router according to the configuration of the I-PDUs in the PDU Router.

[SWS_IpduM_00022] ⌈If the IpduM receives a TxConfirmation for a specific IpduM I-PDU, it shall translate this confirmation into the corresponding confirmations for the COM I-PDUs, which were contained in the last sent out multiplexed IpduM I-PDU.⌋

Depending on the configuration of `IpduMTxDynamicConfirmation` and `IpduMTxStaticConfirmation`, the IpduM will pass zero, one or two confirmations towards COM for one send request. The number of confirmations given to the upper layer does not depend on the `IpduMTxTriggerMode`.

Examples:

- If neither `IpduMTxDynamicConfirmation` nor `IpduMTxStaticConfirmation` for the corresponding `IpduMTxRequest` is configured to true, no COM confirmation is generated.
- If `IpduMTxStaticConfirmation` is configured to true but and `IpduMTxDynamicConfirmation` is configured to false (or vice versa), then only one COM confirmation is generated.
- If both `IpduMTxStaticConfirmation` and `IpduMTxDynamicConfirmation` is configured to true, then two COM confirmations are generated; to the I-PDU representing the static part and the I-PDU representing the dynamic part.

In case two transmission confirmatios are generated, they are obviously equal, since they are derived from the same I-PDUM transmission confirmation.

7.2.5 Reception

Every I-PDU which is received by the Communication Hardware Abstraction (CAN Interface, Lin Interface, FlexRay Interface) is given to the PDU Router. The PDU Router routes multiplexed I-PDUs to the IpduM module. The IpduM module separately routes the static and dynamic parts of the multiplexed I-PDU to their destinations.

It is known at configuration-time which incoming I-PDU IDs correspond to multiplexed I-PDUs with a static part configured. The I-PDU ID is all that is necessary to work out if there is a static part present.

There are no requirements to handle or notify wrongly configured parts. Hence, if the received I-PDU contains segments not configured for reception on this ECU, they will be ignored silently. Furthermore, if an I-PDU is configured with a PduLength of 0, it will also be ignored silently, since no meaningful processing can be configured.

This situation might occur in a gateway setting, if a multiplexed I-PDU is always routed onto another bus by the PDU Router, but contains a signal in one dynamic part that must be passed to the application. In this case, the multiplexed PDU would have to be routed to the IpduM as well.

7.2.6 Metadata handling

The requirements of this section only apply if `IpduMMetaDataSupport` is configured to true.

[SWS_IpduM_00225] ⌈If `IpduMTxTriggerMode` is configured to a different value than `NONE`, the IpduM shall use the MetaData of the triggering part for sending of the multiplexed I-PDU.⌋

[SWS_IpduM_00226] ⌈If `IpduMTxTriggerMode` is configured to `NONE`, the IpduM shall use the MetaData of the last updated part for sending of the multiplexed I-PDU.⌋

[SWS_IpduM_00227] ⌈On receiver side the IpduM shall forward the received Meta-Data along with all demultiplexed parts.⌋

7.3 Multiple-PDU-to-Container handling

IpduM supports a mapping of several I-PDUs to one Container PDU. Both contained and Container PDUs are *regular* PDUs from PduR's point of view. The Container layout can either be dynamically defined using headers in front of the contained I-PDUs or statically without headers but defined static positions for contained I-PDUs.

IpduM relies on PduR being configured to forward send-PDUs mapped to a Container-PDU and received Container-PDUs to IpduM.

7.3.1 Dynamic Container Layout

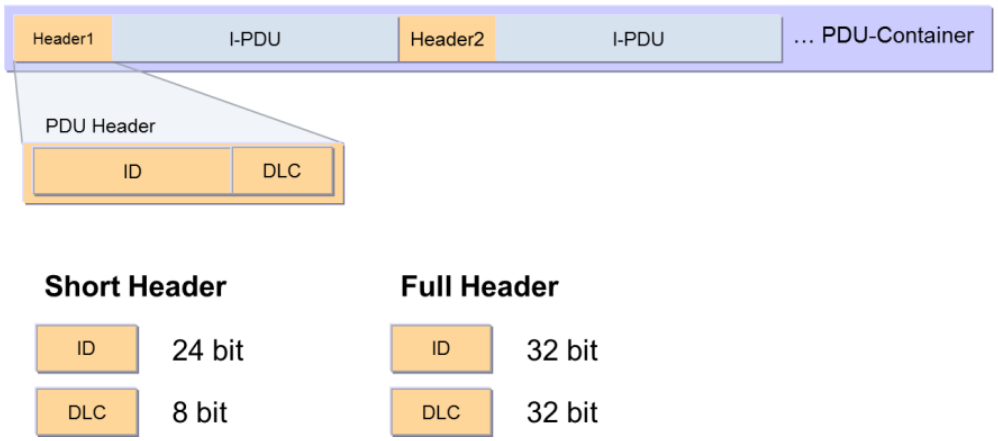


Figure 7.3: Layout of a dynamic Container

[SWS_IpduM_00175]

Status: DRAFT

⌈Inside a dynamic Container PDU IpduM shall place the header of a contained I-PDU in front of the contained I-PDU.⌋

See also Figure 7.3.

For dynamic Container PDUs, there is no configuration of the positions of contained I-PDUs inside the Container PDU, thus the position of an arbitrary contained I-PDU is determined by the length of payload (DLC) and headers of the preceding (added before) contained I-PDUs.

The number of I-PDUs contained in a Container PDU is limited by the maximum size of the Container PDU (PDULength of ECUC-PDU).

The following approaches regarding the order inside the Container I-PDU are possible:

- retained order (see [SWS_IpduM_00179] and [SWS_IpduM_00209]): the order of the contained I-PDUs inside the Container PDU will be retained. This way all contained I-PDUs are extracted in the same order as they have been put into the Container PDU.
- order by priority: the order of the contained I-PDUs inside the Container PDU is based on the configured priority.
- For Dynamic Container: Max fill-up strategy - the order of the contained I-PDUs inside a container PDU with last-is-best semantics is based on a mix to retain the order by priority and by trying to fill-up a Container PDU to its maximum size of available contained I-PDUs.

The IpduM supports two different header sizes for dynamic Container PDUs (see `IpduMContainerHeaderSize`):

- IPDUM_HEADERTYPE_SHORT with 24 bit ID and 8 bit length
- IPDUM_HEADERTYPE_LONG with 32 bit ID and 32 bit length

The header sizes are configured per Container PDU via IpduMContainerHeaderSize. Thus, it is valid for the whole Container PDU. Mixing of header sizes inside one Container PDU is not supported.

[SWS_IpduM_00177] ⌈Each I-PDU header shall consist of ID field and length field in the byte order determined by IpduMHeaderByteOrder.⌋

[SWS_IpduM_00178]

Status: DRAFT

⌈Placing of headers and payloads of contained I-PDUs inside a dynamic Container PDU shall be contiguous without any gap.⌋

Rationale: This allows iterating over a Container PDU by considering the header size and payload lengths (DLC from header).

This has to be ensured by the implementation of the container collection algorithm, since contained I-PDUs have no dedicated (configured) position inside a Container PDU.

7.3.2 Static Container Layout

To enable the static container layout, the IpduMContainerHeaderSize of the Container PDU has to be configured to IPDUM_HEADERTYPE_NONE.

[SWS_IpduM_00232]

Status: DRAFT

⌈If the IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_NONE, the IpduM module shall statically place the contained I-PDUs within the Container PDU according to their configured IpduMContainedPduOffset.⌋

For the Static Container Layout Container PDUs with IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_LAST_IS_BEST are supported only.(see [ECUC_IpduM_00198]).

7.3.3 Transmission

The requirements within this section and its subsections apply to the transmission of Dynamic Containers and Static Containers. Deviations are explicitly mentioned.

7.3.3.1 Transmission request

IpduM collects transmission requests for contained I-PDUs from the upper layers per Container PDU. Along with a transmission request the IpduM has to provide a buffer for the corresponding contained I-PDUs based on the configuration:

- In case the Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_QUEUED`, the IpduM stores the corresponding contained I-PDU internally.
- In case the Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST`, the IpduM has to store transmission requests. If a Container PDU is transmitted, the IpduM fetching the data of the contained I-PDUs according to the transmission requests from the I-PDU buffer of the corresponding upper layer.

Independent of the buffer strategy, if a transmission request for a Contained I-PDU is indicated, the specification use the term that a “transmission request of a contained I-PDU is assigned to a Container PDU”.

7.3.3.2 Transmission trigger condition

The trigger condition for a transmission of a Container PDU is influenced by several configurable transmission trigger and by transmission trigger which could occure on runtime:

- `IpduMContainedTxPduTrigger`: `IPDUM_TRIGGER_ALWAYS`, `IPDUM_TRIGGER_NEVER`
- `IpduMContainerTxFirstContainedPduTrigger`
- `IpduMContainerTxSendTimeout`
- `IpduMContainedTxPduSendTimeout`
- Only for Dynamic Container: `IpduMContainerTxSizeThreshold`
- Only for Dynamic Container: Exceed maximum size of the Container PDU
- Only for Static Container: All contained I-PDUs of the same Container PDU were updated (see [SWS_IpduM_00234])

The first occurence of a transmission trigger for an instance of a Container PDU triggers the transmission process. Any further occurence of a transmission trigger to that instance of Container PDU will not impact the transmission process.

[SWS_IpduM_00274] Behaviour for the occurence of multiple assigned transmission requests ⌈If a Container PDU has been triggered for transmission, then any further transmission trigger for that instance of Container PDU shall be ignored.⌋

The IpduM need to consider displacement of contained I-PDUs. Contained I-PDUs are called "displaced contained I-PDUs", if they fit in a Container PDU at the time when the transmission request occured, but at assembly time of the Container PDU they could not be considered to be placed in that Container PDU, due to lacking of space. Displaced contained I-PDUs could potentially occure under either the following configuration constellations:

- Dynamic Container configured with IPDUM_COLLECT_LAST_IS_BEST and contained I-PDU with dynamic PDU length
- Dynamic Container where priority handling for contained I-PDUs is enabled (IpduMContainedTxPduPriorityHandling set to TRUE)

Example for displacement of an contained I-PDU: A Container PDU is configured with IPDUM_COLLECT_LAST_IS_BEST and a contained I-PDU with dynamic PDU length. A transmission request of the contained I-PDU with dynamic PDU length has already been assigned to the Container PDU. A further transmission request of a contained I-PDU occure, that fill-up the Container PDU to its maximum size, such that this contained PDU could be placed in this Container PDU. This would trigger the transmission of that Container PDU. In the mean time the length of the contained I-PDU with dynamic PDU length has been enlargen. The Container PDU is assembled, with result that the last contained I-PDU could not be considered for this Container PDU instance. Thus, the last contained I-PDU which had been placed in the Container PDU and fit in that Container PDU at the point in time when the transmission for that contained PDU was requested, is displaced at assembly time due to the enlargend contained I-PDU with dynamic PDU length.

The IpduM need to balance between configurable transmission trigger and transmission trigger that occure on runtime (e.g. processing of time out handling) for a Container PDU. The following requirements reflect the transmission trigger condition rules.

[SWS_IpduM_00276] Qualification of displaced contained I-PDUs ⌈The IpduM shall qualify a contained I-PDU as displaced contained I-PDU, if a contained I-PDU had enough space to be placed in the Container PDU at time when the transmission request was assigned to that instance of Container PDU, but at the assembly time this contained I-PDU could not be considered, due to lack of space. Displacement of contained I-PDUs shall be considered for either the following configuration constellations:

- Dynamic Container configured with IPDUM_COLLECT_LAST_IS_BEST and contained I-PDU with dynamic PDU length
- Dynamic Container where priority handling for contained I-PDUs is enabled (IpduMContainedTxPduPriorityHandling set to TRUE)

⌋

[SWS_IpduM_00272] Overruling time out handling ⌈The following transmission trigger conditions shall always overrule the time out handling (see IpduMContainerTxSendTimeout and IpduMContainedTxPduSendTimeout) of a Container PDU:

- IPDUM_TRIGGER_ALWAYS
- IpduMContainerTxFirstContainedPduTrigger
- Only for Dynamic Container: IpduMContainerTxSizeThreshold
- Only for Dynamic Container: Exceed maximum size of the Container PDU
- Only for Static Container: All Contained PDUs of the same Container PDU were updated (see [SWS_IpduM_00234])

⌋

[SWS_IpduM_00273] Behaviour of an overruled and running time out timer ⌈If a time out handling of a Container PDU is overruled and the time out timer is already running for this Container PDU, then the time out timer shall be canceled.⌋

[SWS_IpduM_00279] Remaining time out timer value of a canceled timer at a Container PDU instance ⌈A time out timer value of a canceled timer shall remain at the Container PDU instance, where contained I-PDUs could potentially be displaced. If a time out timer already elapsed, then the remaining timer value shall be set to zero.⌋

[SWS_IpduM_00277] Time out handling for a new Container PDU instance, where a displaced contained I-PDUs is added ⌈If a displaced contained I-PDU was shifted from a Container PDU instance where a remaing time out timer value is available and this displaced container I-PDU is placed in a new Container PDU instance, then the time out handling of the new Container PDU instance shall continue with the remaining timer value of the previous Container PDU instance.⌋

[SWS_IpduM_00278] Handling for displaced contained I-PDUs where the remaining time out timer value was set to zero ⌈If a displaced contained I-PDU was shifted from an Container PDU instance where the remaining time out timer value was set to zero and this displaced container I-PDU is placed in a new Container PDU instance, then the new instance of a Container PDU shall immediately be triggered for transmission.⌋

Note: Handling of displaced contained I-PDUs could cause transmission bursts on the network

[SWS_IpduM_00181] ⌈When assigning a transmission request of a contained I-PDU to a Container PDU which has not been triggered yet, and if IpduMContainedTxPduTrigger is set to IPDUM_TRIGGER_ALWAYS, the Container PDU shall be triggered immediately.⌋

[SWS_IpduM_00201] ⌈When assigning the first transmission request of a contained I-PDU to a Container PDU with the parameter IpduMContainerTxFirstContainedPduTrigger set to TRUE, the Container PDU shall be triggered for transmission.⌋

Note: A Container PDU with transmission trigger set to IpduMContainerTxFirst-ContainedPduTrigger could be used for example to request transmission for a time-triggered bus.

[SWS_IpduM_00184] ⌈When assigning the first transmission request of a contained I-PDU to a Container PDU and either IpduMContainerTxSendTimeout of the Container PDU or IpduMContainedTxPduSendTimeout of the contained I-PDU is configured greater than zero and this Container PDU was not already triggered for transmission (see [SWS_IpduM_00272]), the IpduM module shall start the transmission timer of the Container PDU. The timer shall be initialized with the smaller non zero value of IpduMContainerTxSendTimeout and IpduMContainedTxPdu-SendTimeout.⌋

[SWS_IpduM_00185] ⌈When a transmission request of a contained I-PDU is assigned to a Container PDU and this Container PDU was not already triggered for transmission (see [SWS_IpduM_00272]), the transmission timer of the Container PDU shall be updated with the contained I-PDU's timeout (IpduMContainedTxPduSendTimeout) if it is less than the remaining time of the Container PDU.⌋

[SWS_IpduM_00186] ⌈When the transmission timer of the Container PDU defined by [SWS_IpduM_00184] has elapsed, the Container PDU shall be triggered.⌋

Note: If neither the IpduMContainedTxPduSendTimeout nor the IpduMContainerTxSendTimeout is provided to the Container PDU and its Contained PDUs, the IpduM module will not trigger the Container PDU by the transmission timer of the Container PDU (no timeout-based triggering for the Container PDU).

[SWS_IpduM_00180] ⌈When assigning a transmission request of a contained I-PDU to a Container PDU that has not been triggered yet, and if the resulting payload of this Container PDU is bigger than IpduMContainerTxSizeThreshold, then the Container PDU shall be triggered.⌋

Note: The transmission trigger condition if exceeding the maximum size of the Container PDU is specified in [SWS_IpduM_00182], [SWS_IpduM_00183] and [SWS_IpduM_00231].

7.3.3.3 Assembly of Container PDUs

Please note, "assembly of a Container PDU" is a generic phrase to describe the delivery of a Container PDU to the lower layer. If queueing is used (see chapter 7.3.3.5), then the phrase denote to fetch an instance of a Container PDU from the internal queue. If last is best collection semantic is used (see 7.3.3.6), then the phrase denotes to fetch the data from the according upper layer.

According to the evaluated trigger conditions, a Container PDU is triggered for transmission. The transmission process depends on the configuration (IPDUM_DIRECT or IPDUM_TRIGGERTRANSMIT):

- In case IpduMContainerTxTriggerMode is set to IPDUM_DIRECT, the Container PDU is assembled when the trigger condition result in a transmission of the Container PDU.
- In case IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, the Container PDU is assembled when the lower layer indicate via IpduM_TriggerTransmit the readiness of the hardware to transmit the Container PDU.

Independent how the Container PDU is conveyed to the transmission hardware buffer, the trigger of a Container PDU result in a call of PduR_IpduMTransmit.

[SWS_IpduM_00188] ⌈When a Container PDU is triggered and IpduMContainerTxTriggerMode is set to IPDUM_DIRECT, IpduM shall assemble the Container PDU and handle the transmission process with respect to the configuration for the transmission queue (please refer to [SWS_IpduM_00261] and [SWS_IpduM_00262]).⌋

[SWS_IpduM_00254] ⌈When a Container PDU is triggered and IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, IpduM shall invoke PduR_IpduMTransmit with SduDataPtr set to NULL and SduLength according to the following preconditions:

- For static Container PDUs the SduLength shall be calculated with respect to IpduMPduUpdateBitPosition (see [SWS_IpduM_CONSTR_00241]), IpduMContainedPduOffset (see [SWS_IpduM_CONSTR_00242]) and length of last contained I-PDU, if having dynamic PDU length configured (see [SWS_IpduM_CONSTR_00246]).
- For dynamic Container PDUs the SduLength shall be calculated as specified in [SWS_IpduM_00187].

⌋

[SWS_IpduM_00256] ⌈When a Container PDU has been triggered and IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, IpduM shall assemble the Container PDU in the context of the API call IpduM_TriggerTransmit.⌋

Note:

- [SWS_IpduM_00189] has to be considered for the transmissionprocess
- Container PDUs with IpduMContainerTxTriggerMode set to IPDUM_TRIGGERTRANSMIT are assembled in the context of IpduM_TriggerTransmit. Therefore the IpduM has to consider the following points for call of PduR_IpduMTransmit:
  - The SduDataPtr has to be set to NULL, to indicate that the Container is not assembled yet.
  - The SduLength has to be set to the length with respect to the current available transmission requests. This is the minimum requested length to be

considered by the lower layer. In the subsequential corresponding IpduM_TriggerTransmit call, the lower layer has to provide an available buffer length which is equal or greater than the minimum requested length.

- Container PDUs with IpduMContainerTxTriggerMode set to IPDUM_TRIGGERTRANSMIT, transmission requests are continuously collected, even after a call of PduR_IpduMTransmit:
  - For Container PDUs where IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_QUEUED: Contained I-PDUs could be added to Container PDUs until the Container PDU is fetched (see [SWS_IpduM_00194]) or the maximum size of the Container PDU is reached.
  - For Container PDUs where IpduMContainedTxPduCollectionSemantics is set to IPDUM_COLLECT_LAST_IS_BEST: transmission requests or updates of contained PDUs are considered until the Container PDU is fetched (see [SWS_IpduM_00256]).
  - The calculated minimum requested SduLength of a Container PDU within the call of PduR_IpduMTransmit (requested SduLength) could be different from the calculated SduLength of a Container PDU, when the Container PDU is assembled within the call of IpduM_TriggerTransmit (actual SduLength):
    - If using IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_QUEUED, actual SduLength could be equal or greater as the requested SduLength.
    - If using IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_LAST_IS_BEST, actual SduLength could be equal, greater or shorter as the requested SduLength. A shorter length is possible, if a contained I-PDU with dynamic PDU length was updated to a shorter lnegth after PduR_IpduMTransmit was called and before the corresponding call of IpduM_TriggerTransmit.

7.3.3.4 Transmission confirmation

[SWS_IpduM_00189] ⌈The IpduM shall wait for the transmission confirmation (regardless of the result) of a Container PDU before invoking PduR_IpduMTransmit for the next instance of that Container PDU.⌋

Note: The IpduM module relies here on a configured transmission confirmation for that Container PDU in the lower layer.

[SWS_IpduM_00196] ⌈If the IpduM receives a transmission confirmation for a specific Container PDU, it shall translate this confirmation into the corresponding confirmations

for those contained I-PDUs having IpduMContainedTxPduConfirmation set to TRUE and were contained in the last transmission of this specific Container PDU.

This shall result in multiple TxConfirmations, if the same contained I-PDU was present more than once in this transmitted specific Container PDU.⌋

7.3.3.5 Queueing

For Container PDUs where the IpduMContainedTxPduCollectionSemantics is set to IPDUM_COLLECT_QUEUED, a queueing of Container PDUs can be used (see [SWS_IpduM_CONSTR_00260]). In this case more than one instance of a Container PDU has to be kept by the IpduM, up to IpduMContainerQueueSize instances can be stored in addition to the current instance. The current instance is one instance of the Container PDU that currently contained I-PDUs are being added to. After this instance has either been queued or copied to the lower layer, i.e. after a TriggerTransmit or Transmit API call depending on the configuration of IpduMContainerTxTrigger-Mode, no more contained I-PDUs can be added to this instance.

Note: Queued collection semantics guarantees that every instance/value of the contained I-PDU is visible on the wire.

[SWS_IpduM_00261] ⌈If a PDU Container is queued for transmission, and IpduMContainerQueueSize is not set (or set to 1), IpduM shall call PduR_IpduMTransmit.⌋

[SWS_IpduM_00262] ⌈If a PDU Container is queued for transmission, and IpduMContainerQueueSize is > 1, IpduM shall store the Container PDU in a queue.⌋

[SWS_IpduM_00195] ⌈If PduR_IpduMTransmit has returned E_NOT_OK, the same transmit request shall be repeated during the next call to IpduM_MainFunctionTx up to IpduMMaxTransmitRetries times. The instance of that Container PDU is queued in the meantime.⌋

See also [SWS_IpduM_00199].

[SWS_IpduM_00190] ⌈In case the transmission confirmation for that Container PDU was received, the IpduM shall invoke PduR_IpduMTransmit for the next oldest instance of that Container PDU during the next call to IpduM_MainFunctionTx at the latest.⌋

[SWS_IpduM_00191] ⌈In case IpduMContainerTxTriggerMode is set to IPDUM_DIRECT, and PduR_IpduMTransmit returns E_OK for that Container PDU, IpduM shall remove that instance from the queue.⌋

In this case instances of a Container-PDU can be lost, if a queue inside CanIf is used since a newer instance could overwrite a previous one. Such last-is-best behavior might not be desired in this case.

[SWS_IpduM_00199] ⌈If creating a new instance of a Container PDU would exceed IpduMContainerQueueSize the oldest instance shall be discarded. If IpduMContainerQueueSize is not configured the local instance shall be discarded. In both cases IPDUM_E_QUEUEOVFL shall be reported to [9, DET] via Det_ReportRuntimeError.⌋

[SWS_IpduM_00200] ⌈If IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, a Container PDU instance shall be dropped from the queue if it is fetched by IpduM_TriggerTransmit.⌋

[SWS_IpduM_00265] **Handling of discarded container PDUs** ⌈If an instance of a container PDU was discarded (see [SWS_IpduM_00199]), then PduR_IpduMTxConfirmation shall be called with result E_NOT_OK for every contained PDU that has IpduMContainedTxPduConfirmation enabled within that respective instance.⌋

[SWS_IpduM_00193] ⌈If IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, IpduM shall keep and provide buffered data until it is fetched by a call to IpduM_TriggerTransmit.⌋

[SWS_IpduM_00194] ⌈If IpduMContainerTxTriggerMode is set to IPDUM_TRIGGERTRANSMIT, IpduM_TriggerTransmit shall copy the oldest Container PDU instance in the queue. If the queue is empty/non-existent, the current instance of the Container PDU is copied. If the current instance of the Container PDU is empty/non-existent as well, E_NOT_OK is returned by IpduM_TriggerTransmit.⌋

7.3.3.6 Last-is-Best collection semantics

For Container PDUs where the IpduMContainedTxPduCollectionSemantics is set to IPDUM_COLLECT_LAST_IS_BEST a queueing of Container PDUs is not supported (see [SWS_IpduM_CONSTR_00260]).

Note: As soon as a contained I-PDU is configured to use last-is-best semantics, the following points have to be considered:

- not necessarily all instances/values of this contained I-PDU are visible on the wire. This behaviour becomes worse if using FlexRay bus systems and the application scheduling are not synchronized with the FlexRay cycle. This could impact communication where sequence counters are used (e.g. E2E protected communication).
- Extended interrupt runtime has to be considered, if e.g. CAN to CAN routing is used and the routing is performed in the context of the receive interrupt, since this would include at least reception, forwarding of a PDU and assembly of a Container PDU.

[SWS_IpduM_00220] ⌈For Container PDUs, with IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_LAST_IS_BEST, IpduM shall use PduR_IpduMTriggerTransmit to fetch the PDU data from its upper layer immediately before it transfers the container I-PDU to the lower layer.⌋

While it seems natural to use IpduMContainedTxPduCollectionSemantics IPDUM_COLLECT_LAST_IS_BEST in combination with IpduMContainerTxTriggerMode IPDUM_TRIGGERTRANSMIT, it may also be used in combination with IPDUM_DIRECT.

7.3.4 Transmission of Dynamic Containers

The requirements within this chapter and its subchapters complement the transmission for Dynamic Containers and do not apply to Static Containers.

[SWS_IpduM_00187] ⌈If a Container PDU is assembled (see [SWS_IpduM_00188] and [SWS_IpduM_00256]), IpduM shall calculate the overall size of the Container PDU. The total size builds up by the total of all payloads of the contained I-PDUs plus the total length of the corresponding headers. The result shall be the payload size of the Container PDU.⌋

[SWS_IpduM_00192] ⌈If a Container PDU has IpduMContainerTxTriggerMode set to IPDUM_DIRECT and this Container PDU has been assembled and is passed to PduR, the Parameter PduInfoPtr shall contain a pointer to the assembled Container PDU in SduDataPtr and the total length (according to [SWS_IpduM_00187]) in SduLength.⌋

[SWS_IpduM_00257] ⌈If a Container PDU has IpduMContainerTxTriggerMode set to IPDUM_TRIGGERTRANSMIT and this Container PDU has been assembled in the context of the IpduM_TriggerTransmit call, the Parameter PduInfoPtr shall be updated with the total length (according to [SWS_IpduM_00187]) in SduLength.⌋

7.3.4.1 Queued collection semantics

The order regarding the transmission request of the contained I-PDU is retained within the resulting Container PDU, if the Container PDU has IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_QUEUED.

Due to the following requirements IpduM will make sure that instances of a contained I-PDU (same PDU-ID) are transmitted (passed to PduR inside their Container PDUs) in exactly the same order as they are passed to IpduM.

[SWS_IpduM_00179] ⌈When a contained I-PDU is passed to IpduM via IpduM_Transmit, and the associated Container PDU is configured with IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_QUEUED (see [ECUC_Ipdu-

M_00198]), IpduM shall append the contained I-PDU to its payload even if a previous instance of the contained I-PDU is already present in that Container PDU.⌋

This way a Container PDU can include more than one instance of the same I-PDU. The resulting behavior is FIFO-like in order to preserve the order of I-PDU instances being transmitted. Thus, the upper layer(s) of the receiving IpduM can implement either last-is-best or FIFO semantics.

[SWS_IpduM_00182] ⌈If a Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_QUEUED` (see [ECUC_IpduM_00198]), the `IpduMContainerTxTriggerMode` is set to `IPDUM_DIRECT` and adding a contained I-PDU would exceed the maximum size of the Container I-PDU, first the Container PDU shall be triggered. The contained I-PDU shall be added to a new instance of the Container PDU.⌋

[SWS_IpduM_00183] ⌈If a Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_QUEUED` (see [ECUC_IpduM_00198]), the `IpduMContainerTxTriggerMode` is set to `IPDUM_TRIGGERTRANSMIT` and adding a contained I-PDU would exceed maximum size of the Container PDU, first the Container PDU shall be queued. Then the contained I-PDU shall be added to a new instance of the Container PDU.⌋

7.3.4.2 Triggered Transmission and Last-is-Best semantics

In case of Container PDU with `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST`, the IpduM collects and stores the transmission requests for the relevant contained I-PDUs.

[SWS_IpduM_00231] ⌈If a Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST`, the transmission request is updated and the total length of the payloads plus the corresponding headers of all contained I-PDUs which are requested for transmission would exceed the maximum size of the Container PDU, then the Container PDU shall be triggered.⌋

Note: the overall size of the updated I-PDUs could fluctuate, if contained I-PDUs have dynamic size configured. Exceeding maxumum size of the Container PDU leads to trigger the Container PDU. In case the Conatiner PDU is assembled in the context of the `IpduM_TriggerTransmit`, then the length of contained I-PDU configured with dynamic length colud be updated as long as the call of the `IpduM_TriggerTransmit` is pending. This could lead to two scenarios:

1. If current length < updated length of the Container I-PDU with dynamic length, then may be not all contained I-PDUs could be added to the next transmission of the Container PDU.

2. If current length > updated length of the Container I-PDU with dynamic length, then may be the primal trigger condition “exceeding the maximum size of the Container PDU” is not valid anymore

In both cases, the Container PDU is assembled and transmitted with the latest available contained I-PDU with respect to maximum size of the Container I-PDU (see [SWS_IpduM_00259]).

[SWS_IpduM_00222] ⌈If a Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST`, is assembled and if a `PduR_IpduMTriggerTransmit` returns `E_NOT_OK` for a contained I-PDU, IpduM shall omit this contained I-PDU silently and proceed with the assembly of the Container PDU.⌋

Note: The associated Container PDU has be transmitted anyway without the omitted contained I-PDU.

7.3.4.2.1 Order and re-sorting

The order regarding the transmission request of the contained I-PDU is either retained or re-sorted within the resulting Container PDU:

- In case the priority for a contained I-PDU (see `IpduMContainedTxPduPriorityHandling`) is disabled, the IpduM retains the order of the contained I-PDUs according to the passed transmission requests for the first occurrence.
- In case the priority for a contained I-PDU (see `IpduMContainedTxPduPriorityHandling`) is enabled, the IpduM resorts the order of the contained I-PDUs according to following preconditions.
  - If `IpduMContainedTxPduPriorityHandling` is set to `TRUE` and `IpduMContainerTxPduMaxFillUpStrategy` is set to `FALSE`, the contained I-PDUs of the resulting Container PDUs are ordered by the configured priorty.
  - If `IpduMContainedTxPduPriorityHandling` is set to `TRUE` and `IpduMContainerTxPduMaxFillUpStrategy` is set to `TRUE`, in the first step the contained I-PDUs of the resulting Container PDUs are ordered by the configured priority. If space is left over in the Container PDU, the remaining available contained I-PDUs (where transmission request is available) are also checked and placed in the Container PDU if they fit into the left over space.

Note: It is is not allowed to set `IpduMContainedTxPduPriorityHandling` to `FALSE` and `IpduMContainerTxPduMaxFillUpStrategy` to `TRUE`, since `IpduMContainerTxPduMaxFillUpStrategy` is only possible if `IpduMContainedTxPduPriorityHandling` is set to `TRUE` (see dependency of [ECUC_IpduM_00220]).

[SWS_IpduM_00221] ⌈When a transmission request of a contained I-PDU is assigned to a Container PDU, the Container PDU has `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST` and `IpduMContainedTxPduPriorityHandling` is set to FALSE, the IpduM shall retain the order in which the transmission requests are passed to IpduM. That is, the contained I-PDU of the first passed transmission request is placed at the beginning at the container and so on. If a transmission request of the same contained I-PDU is passed multiple times, the IpduM shall store it only once at the position matching its first occurrence.⌋

Note: Multiple triggers of the same contained I-Pdu during collection of one Container PDU will result in a Container PDU which contains exactly one instance of this contained I-PDU, in case `IpduMContainedTxPduCollectionSemantics` is set to `IPDUM_COLLECT_LAST_IS_BEST`.

[SWS_IpduM_00249] ⌈When transmission requests of contained I-PDUs are assigned to a Container PDU, `IpduMContainedTxPduCollectionSemantics` is set to `IPDUM_COLLECT_LAST_IS_BEST` and `IpduMContainedTxPduPriorityHandling` is set to TRUE, the IpduM shall collect the transmission request in respect to the `IpduMContainedTxPduPriority` (highest priority first and so on). Transmission requests of contained I-PDUs with the same `IpduMContainedTxPduPriority` shall be collected in the order the transmission request occurred.⌋

[SWS_IpduM_00258] ⌈If a container PDU is assembled and `IpduMContainedTxPduCollectionSemantics` is set to `IPDUM_COLLECT_LAST_IS_BEST`, the IpduM shall place the contained I-PDUs in the container I-PDU with respect to the configuration and resulting order according to [SWS_IpduM_00221] and [SWS_IpduM_00249].⌋

[SWS_IpduM_00259] ⌈If a container PDU is assembled where `IpduMContainedTxPduCollectionSemantics` is set to `IPDUM_COLLECT_LAST_IS_BEST` and `IpduMContainerTxPduMaxFillUpStrategy` is set to FALSE:
When placing a contained I-PDU would exceed maximum size of the Container PDU, the IpduM shall finish the assembly process and keep the remaining transmission requests for the next transmission.⌋

[SWS_IpduM_00264] Container PDU fill up strategy for Last-is-best contained I-PDUs ⌈If a container PDU is assembled where `IpduMContainedTxPduCollectionSemantics` is set to `IPDUM_COLLECT_LAST_IS_BEST` and the `IpduMContainerTxPduMaxFillUpStrategy` is set to `TRUE`:
When adding a contained I-PDU would exceed the maximum size of the Container PDU, the IpduM shall switch to iterating over the remaining transmission requests in the following order, adding all contained I-PDUs that fit in the remaining space in the container PDU:

1. Start with the oldest transmission request of the highest priority.
2. Continue through the transmission requests of the same priority unto the newest transmission request until a contained PDU does not fit.

3. Go to the next lower priority and continue with 2. until all transmission requests have been checked or the container PDU is full.

⌋

[SWS_IpduM_00264] effectively switches the behavior from strictly following FiFo and PDU priorities to filling the remaining space with any queued contained I-PDUs that will fit without reordering PDUs in the request queues.

Note: Using the maximum fill-up strategy (`IpduMContainerTxPduMaxFillUpStrategy` set to `TRUE`) may lead to a re-ordering of contained I-PDUs on the bus regarding their priority or the sequence of transmission requests. Priority inversion within a Container PDU is therefore a potential side effect of this strategy. Starvation of contained I-PDUs can be avoided by a suitable network communication design, especially by a careful assignment of contained I-PDUs.

7.3.5 Transmission of Static Containers

The requirements within this chapter and its subchatpers complement the transmission for Static Containers and do not apply to Dynamic Containers.

[SWS_IpduM_00234]

Status: DRAFT

⌈For Container PDUs with static container layout and `IpduMContainerTxTriggerMode` is set to `IPDUM_DIRECT`, the IpduM shall trigger the Container PDU when all contained I-PDUs were updated by the upper layer.⌋

Since, Static Container might include not updated contained I-PDUs, there are means to detect the currentness of contained I-PDUs on receiver side. Either update-bits for contained I-PDUs or an unsed area default pattern might be configured. For the concrete configuration and configuration rules, see Chapter 10.

[SWS_IpduM_00235]

Status: DRAFT

⌈In case a contained I-PDU has a configured `IpduMPduUpdateBitPosition`, the IpduM shall ensure that the update bit of this contained I-PDU is set if and only if the contained I-PDU was successfully updated.⌋

[SWS_IpduM_00233]

Status: DRAFT

⌈In case a Static Container has a configured `IpduMUnusedAreasDefault`, the IpduM shall ensure that all not updated areas of the Container are set to the value of `IpduMUnusedAreasDefault` before the Container PDU is sent.⌋

This allows the IpduM to handle contained I-PDUs with dynamic length also within static containers. However, the receiving IpduM is not able to detect if the SWC or the sending IpduM set the IpduMUnusedAreasDefault-value. Hence, always the complete, thus eventually filled up contained I-PDU, will be received.

It must be observed, that some bus-systems (eg. CAN-FD and FlexRay) cannot transfer PDUs of arbitrary length and might fill up the sent I-PDU to the next possible length with their own default value. Hence, the configuration of the IpduMUnusedAreasDefault value and the bus-specifc padding-pattern should be aligned.

[SWS_IpduM_00253] ⌈When sending a Static Container PDU, the IPduM shall reduce the size of the Container PDU by omitting not updated data behind the last updated PDU and the last update bit.⌋

Caveat: By reducing the size of a Container PDU and applying a padding by the underlying bus, it is possible that on receiver side a contained PDU with the padding pattern of th ebus is received. If this is not acceptable, it shoud be considered using update-bits for the contained PDUs or the system designer has to cope with this by other means.

7.3.6 Reception

There exist two possible ways how the reception of a received Container PDU can be processed by the IpduM:

- accept configured
- accept all (only for IPDUM_HEADERTYPE_LONG and IPDUM_HEADERTYPE_SHORT)

[SWS_IpduM_00202] ⌈If IpduMContainerPduProcessing is set to IPDUM_PROCESSING_IMMEDIATE, the processing of the received Container PDUs shall be executed in the context of IpduM_RxIndication. Otherwise, it is deferred to the next call to IpduM_MainFunctionRx. All deferred Container PDUs shall be processed in the order of their reception.⌋

[SWS_IpduM_00203] ⌈If by a call of IpduM_RxIndication a Container PDU is received, the contained I-PDUs shall be extracted.⌋

[SWS_IpduM_00205] ⌈If for the received Container PDU the configuration parameter IpduMContainerRxAcceptContainedPdu is set to IPDUM_ACCEPT_CONFIGURED, IpduM shall expect and match only contained I-PDUs (IpduMContainedRxPdu) that reference the IpduMContainerRxPdu in IpduMContainedRxInContainerPduRef.⌋

Note that it is well possible to define several IpduMContainedRxPdus with the same IpduMContainedRxPduLongHeaderId/IpduMContainedRxPduShortHeaderId as long as the IpduMContainedRxPdus are assigned to different IpduMContainerRxPdus (via IpduMContainedRxInContainerPduRef) with

each IpduMContainerRxPdu having IpduMContainerRxAcceptContainedPdu = IPDUM_ACCEPT_CONFIGURED.

[SWS_IpduM_00250] ⌈For an IpduMContainerRxPdu withIpduMContainerRxAcceptContainedPdu = IPDUM_ACCEPT_CONFIGURED and IpduMContainerHeaderSize = IPDUM_HEADERTYPE_LONG or IPDUM_HEADERTYPE_SHORT the following constraint applies:

- IpduMContainedRxPdus with an IpduMContainedRxInContainerPduRef to that specific IpduMContainerRxPdu shall have unique IpduMContainedRxPduLongHeaderId or IpduMContainedRxPduShortHeaderId, respectively, defined in the context of that IpduMContainerRxPdu.

⌋

[SWS_IpduM_00209] ⌈Each contained I-PDU shall be notified to PduR via PduR_IpduMRxIndication. IpduM shall indicate the contained I-PDUs in the same order as the I-PDUs are located inside the Container PDU.⌋

7.3.6.1 Queueing

[SWS_IpduM_00211] ⌈If a Container PDU is received and IpduMContainerPduProcessing is set to IPDUM_PROCESSING_DEFERRED, the Container PDU shall be queued.⌋

[SWS_IpduM_00212] ⌈If receiving a new instance of a Container PDU would exceed IpduMContainerQueueSize the oldest instance shall be discarded and IPDUM_E_QUEUEOVFL shall be reported to [9, DET] via Det_ReportRuntimeError.⌋

7.3.7 Reception of Dynamic Containers

The requirements within this chapter and its subchapters complement the reception of Dynamic Containers and do not apply to Static Containers.

[SWS_IpduM_00204] ⌈For each contained I-PDU of a received Container PDU where the IpduMContainerRxPdu has IpduMContainerRxAcceptContainedPdu = IPDUM_ACCEPT_ALL the ID from the PDU header shall be used to identify the corresponding contained I-PDU:

- If the received Container PDU uses long or short header (IpduMContainerHeaderSize = IPDUM_HEADERTYPE_LONG or IPDUM_HEADERTYPE_SHORT, respectively) the ID shall be compared with the IpduMContainedRxPduLongHeaderId or IpduMContainedRxPduShortHeaderId, respectively, in the set of IpduMContainedRxPdus which do not have an IpduMContainedRxInContainerPduRef defined.

⌋

[SWS_IpduM_00206] ⌈For a received Container PDU with IpduMContainerRxAcceptContainedPdu=IPDUM_ACCEPT_ALL, IpduM shall expect and match only IpduMContainedRxPdus which do not have an IpduMContainedRxInContainerPduRef defined.⌋

[SWS_IpduM_00251] ⌈All IpduMContainedRxPdus with no IpduMContainedRxInContainerPduRef and a defined IpduMContainedRxPduLongHeaderId or IpduMContainedRxPduShortHeaderId, shall have a unique IpduMContainedRxPduLongHeaderId or IpduMContainedRxPduShortHeaderId, respectively.⌋

Note that due to [SWS_IpduM_00206] it is NOT allowed to define several IpduMContainedRxPdus with the same IpduMContainedRxPduShortHeaderId / IpduMContainedRxPduLongHeaderId and no assignment to IpduMContainerRxPdu (no IpduMContainedRxInContainerPduRef defined).

Note that it is well possible to have two IpduMContainedRxPdus with no assignment to IpduMContainerRxPdu (no IpduMContainedRxInContainerPduRef defined) which have the same header id value defined, as long as one IpduMContainedRxPdu has an IpduMContainedRxPduShortHeaderId defined and the other IpduMContainedRxPdu has an IpduMContainedRxPduLongHeaderId defined.

Note that due to two ways how IpduMContainedRxPdus can interact with IpduMContainerRxPdu (IPDUM_ACCEPT_CONFIGURED and IPDUM_ACCEPT_ALL definition at the IpduMContainerRxPdu) it is well possible that IpduMContainedRxPdus with the same IpduMContainedRxPduShortHeaderId / IpduMContainedRxPduLongHeaderId exist as long as the constraints defined in [SWS_IpduM_00250], [SWS_IpduM_00251] are fulfilled.

[SWS_IpduM_00207] ⌈If a contained I-PDU of a received IpduMContainerRxPdu with IpduMContainerRxAcceptContainedPdu = IPDUM_ACCEPT_ALL can not be matched according to [SWS_IpduM_00206] then this contained I-PDU shall be discarded silently.⌋

[SWS_IpduM_00208] ⌈For each contained I-PDU the length given in its header shall be used as the length of the corresponding I-PDU.⌋

[SWS_IpduM_00210] ⌈When processing a received Container PDU and detecting a header containing the ID 0 the processing for this Container PDU shall be stopped and the remaining bytes shall be ignored.⌋

Rationale: A header ID of 0 means that Container PDU has been filled with padding bytes and no further data is contained.

[SWS_IpduM_00210] does not mean that always a 0 has to be expected at the end of the payload.

7.3.8 Reception of Static Containers

The requirements within this chapter and its subchatpers complement the reception of Static Containers and do not apply to Dynamic Containers.

In order for the receiving IpduM module to be able to determine which of the PDUs in a received Static Container have actually been updated on the transmitter side, additional update information, so called PDU update bits within the Container Pdu, can be configured per contained I-PDU (see [ECUC_IpduM_00207]).

[SWS_IpduM_00236]

Status: DRAFT

⌈In case a received contained I-PDU has a configured update bit, the IpduM module shall only process and indicate it to the upper layer if its received update-bit is set.⌋

The above requirement results in silently ignoring contained I-PDUs with configured but not set update bits.

With respect to [SWS_IpduM_00236] expected contained I-PDUs without a configured update-bit are always processed and indicated to the upper layer. They are assumed allways valid.

7.3.9 Errorhandling

There are bus systems where it is not possible to set an arbitrary size for the transmitted L-PDU (e.g. CanFD). The valid payload length of a Container PDU can be derived from the contained headers. Therefore, the difference to the actual length of the Container PDU can be considered padding.

Assumption is that underlying bus modules are configured such that the padded values do not build up a valid header.

[SWS_IpduM_00213] ⌈When processing a received Container PDU and detecting a header where the payload length exceeds the remaining bytes of the container the processing for this Container PDU shall be stopped and the remaining bytes shall be ignored. Furthermore, `IPDUM_E_HEADER` shall be reported to [9, DET] via `Det_ReportRuntimeError`.⌋

A header with a payload length greater than the remaining byte is invalid. No further header is to be expected behind it.

[SWS_IpduM_00214] ⌈If the remaining bytes in a Container PDU are less than the configured `IpduMContainerHeaderSize` the remaining bytes shall be ignored.⌋

[SWS_IpduM_00237]

Status: DRAFT

⌈When processing a received Container PDU with IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE, the IpduM shall ignore all contained PDUs that are according to their configuration not or not completely contained in the received Container PDU. Such contained I-PDUs shall not be indicated to the upper layer. If Development Error Detection is configured (IpduMDevErrorDetect) IPDUM_E_CONTAINER shall be reported to [9, DET] via Det_ReportError.⌋

7.3.10 Metadata handling

The requirements of this section only apply if IpduMMetaDataSupport is configured to true.

[SWS_IpduM_00228] ⌈In case a Container PDU supports MetaData, the IpduM shall use the MetaData last collected from the contained I-PDUs when sending the Container PDU.⌋

[SWS_IpduM_00229] ⌈In case the IpduM receives a Container PDU with MetaData, the IpduM shall forward the MetaData of the Container PDU along with all contained I-PDU that support MetaData.⌋

The IpduM does not rearrange MetaData. Thus, it only supports contained I-PDUs assigned to the same Container PDUs, which have no MetaData or have the same MetaDataType, see [SWS_IpduM_CONSTR_00230].

7.3.11 Local buffer asynchronous copy approach

IpduM may request to release the temporary local buffer managed by a lower layer module. See ECUC_EcuC_00087 for reference.

This functionality, Release Rx Buffer on reception side is used by the IpduM to explicitly free a buffer after asynchronous copying has been finalized. On transmission side the lower layer indicates the IpduM with a call of TxConfirmation to explicitly free a Tx Buffer.

IpduM supports offloading the asynchronous copy of a PDU into its local buffer. (See [4, BSW General] Chapter 7.1.21 *"Asynchronous copy approach (DRAFT)"* for more details.) This behavior is controlled per PDU by the ECU Configuration parameter KeepLocalPduBuffer and EcucDmaThreshold.

If KeepLocalPduBuffer is set to FALSE, copying of data is done synchronously in the context of the called function. If KeepLocalPduBuffer is set to TRUE, the source module keeps its buffer until IpduM indicates that the buffer may be released and, depending on EcucDmaThreshold, the copy may be performed asynchronously.

[SWS_IpduM_CONSTR_00281] Asynchronous copy limitation for multiplexed I-PDU feature ⌈A configuration where IpduMRxPathway or IpduMTxPathway refer to the PDU that has KeepLocalPduBuffer set to TRUE, shall be rejected as invalid.⌋

Note: Asynchronous copying is not supported for multiplexed I-PDUs.

[SWS_IpduM_00282] Transmission release buffer ⌈In case IpduMContainerTxTriggerMode is set to IPDUM_DIRECT and the Container PDU has IpduMContainedTxPduCollectionSemantics set to IPDUM_COLLECT_QUEUED, if KeepLocalPduBuffer of the I-PDU is set to TRUE, the IpduM module shall release the internal buffer once TxConfirmation for that Container PDU has been received from the lower layer.⌋

[SWS_IpduM_00283] Reception local buffer copy below DMA threshold ⌈If by a call of IpduM_RxIndication a Container PDU is received and

- KeepLocalPduBuffer is set to TRUE
- SduLength is **less than or equal** to EcucDmaThreshold,

the IpduM module shall copy the I-PDU into its local buffer synchronously.⌋

[SWS_IpduM_00284] Reception local buffer copy above DMA threshold ⌈If by a call of IpduM_RxIndication a Container I-PDU is received and

- KeepLocalPduBuffer is set to TRUE
- SduLength is **greater than** EcucDmaThreshold,

the IpduM module shall copy the I-PDU into its local buffer asynchronously.⌋

[SWS_IpduM_00285] Reception release buffer ⌈If an asynchronous copy for a received IpduMContainerRxPdu was started and the copying has been finalized (successful or not), then the IpduM module shall call PduR_IpduMReleaseRxBuffer.⌋

[SWS_IpduM_00286] Error reporting when asynchronous copy fails ⌈In case an asynchronous copy fails, then the runtime error IPDUM_E_CONTAINER_LOST shall be reported to the DET module.⌋

[SWS_IpduM_00287] Tx buffer alignment for container I-PDUs with KeepLocalPduBuffer set to TRUE ⌈If IpduMContainerTxPdu refers to a I-PDU that has KeepLocalPduBuffer set to TRUE, then the provided data buffer shall respect the memory alignment that is configured at the referenced PDU (see PduBufferAlignment).⌋

Note: KeepLocalPduBuffer set to TRUE indicate, that a I-PDU could potentially be copied asynchronously by the lower layer modules. The memory alignment would support a proper processing when data is added to the hardware buffer.

7.4 Basic Software Distribution

In order to provide a load distribution amongst different partitions (cores), the different parts of the Com-Stack shall be allocated to the different partitions. Hereby it shall be supported that such a partitioning happens on a per-network-type basis, i.e., the FlexRay, CAN, and Ethernet part of the Com-Stack shall be locatable onto different distinct partitions (cores).

In order to support such a flexible allocation with reducing the amount of cross-partition communication (and thus (potentially blocking) synchronization) the main threads of execution in the IpduM module (namely the respective MainFunctions) can be split into different MainFunctions (at least one per partition). This way the flow of reception / transmission stays within the scope of a single network (and thus within a single partition) and therefore does not require special multi- partition capable communication and synchronization primitives.

In order to manage different timing requirements each MainFunction instance defines its time base individually.

[SWS_IpduM_00252]

Status: DRAFT
Upstream requirements: SRS_BSW_00425, SRS_BSW_00478

⌈IpduM Container Pdus shall be processed within the MainFunction, which is referenced by the Container Pdu configuration parameters (i.e. `IpduMContainerRxPdu` and `IpduMContainerTxPdu`).⌋

7.5 Error Classification

7.5.1 Development Errors

[SWS_IpduM_91003] Definition of development errors in module IpduM

Upstream requirements: SRS_BSW_00337, SRS_BSW_00414

⌈

Type of error	*Related error code*	*Error value*
API service called with wrong parameter	IPDUM_E_PARAM	0x10
NULL pointer checking	IPDUM_E_PARAM_POINTER	0x11
API service (except IpduM_MainFunctionTx and IpduM_MainFunctionRx) used without module initialization	IPDUM_E_UNINIT	0x20
Invalid configuration set selection	IPDUM_E_INIT_FAILED	0x21

⌋

7.5.2 Runtime Errors

[SWS_IpduM_91004] Definition of runtime errors in module IpduM ⌈

Type of error	*Related error code*	*Error value*
Erroneous header detected	IPDUM_E_HEADER	0x30
Container Queue overflow	IPDUM_E_QUEUEOVFL	0x31
Partly or erroneous container received	IPDUM_E_CONTAINER	0x32
Loss of a container instance (e.g. failure at asynchronous copy operation).	IPDUM_E_CONTAINER_LOST	0x33

⌋

7.5.3 Production Errors

There are no production errors.

8 API specification

[SWS_IpduM_00028]

Upstream requirements: SRS_BSW_00323

⌈If `IpduMDevErrorDetect` is configured to TRUE, all IpduM APIs shall check their input parameters and report detected errors to [9, DET] via `Det_ReportError`. `IPDUM_E_PARAM` shall be reported for normal parameters and `IPDUM_E_PARAM_POINTER` for pointer parameters.⌋

8.1 Imported types

This chapter lists all imported types and the corresponding modules.

[SWS_IpduM_00102] Definition of imported datatypes of module IpduM

Upstream requirements: SRS_BSW_00357

⌈

Module	*Header File*	*Imported Type*
Comtype	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
Std	Std_Types.h	Std_ReturnType

⌋

8.2 Type definitions

8.2.1 IpduM_ConfigType

[SWS_IpduM_00159] Definition of datatype IpduM_ConfigType

Upstream requirements: SRS_BSW_00438

⌈

Name	IpduM_ConfigType
Kind	Structure
Description	This is the type of the data structure containing the initialization data for the I-PDU multiplexer.
Available via	IpduM.h

⌋

8.3 Function definitions

This chapter contains all APIs provided by the module.

8.3.1 IpduM_Init

[SWS_IpduM_00032] Definition of API function IpduM_Init

Upstream requirements: SRS_BSW_00344, SRS_BSW_00405, SRS_BSW_00101, SRS_BSW_00369

⌈

Service Name	IpduM_Init	

Syntax	`void IpduM_Init (`
	`  const IpduM_ConfigType* config`
	`)`	

Service ID [hex]	0x00	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	config	Implementation specific structure with configuration parameters.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initializes the I-PDU Multiplexer. In configurations, in which IpduM is assigned to more than one partition (i.e. IpduM_Main Functions are mapped to partitions), IpduM may provide one init function per partition.	
Available via	IpduM.h	

⌋

[SWS_IpduM_00033]

Upstream requirements: SRS_BSW_00101

⌈The function `IpduM_Init` shall initialize all module-related global variables.⌋

[SWS_IpduM_00084]

Upstream requirements: SRS_BSW_00406

⌈The behavior of the IpduM is unspecified until a correct call to `IpduM_Init` is made.⌋

8.3.2 IpduM_Transmit

[SWS_IpduM_00043] Definition of API function IpduM_Transmit

Upstream requirements: SRS_BSW_00369

⌈

Service Name	IpduM_Transmit	
Syntax	`Std_ReturnType IpduM_Transmit ( PduIdType TxPduId, const PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x49	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	Identifier of the PDU to be transmitted
	PduInfoPtr	Length of and pointer to the PDU data and pointer to MetaData.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Transmit request has been accepted. `E_NOT_OK`: Transmit request has not been accepted.
Description	Requests transmission of a PDU.	
Available via	IpduM.h	

⌋

For a detailed description read Chapter 7.2.4.1.

8.3.3 IpduM_RxIndication

[SWS_IpduM_00040] Definition of callback function IpduM_RxIndication

Upstream requirements: SRS_BSW_00369

⌈

Service Name	IpduM_RxIndication	
Syntax	`void IpduM_RxIndication ( PduIdType RxPduId, const PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x42	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	Indication of a received PDU from a lower layer communication interface module.
Available via	IpduM.h

⌋

[SWS_IpduM_00041] ⌈If there is a static part configured in a multiplexed SDU received from the PDU Router, the function `IpduM_RxIndication` transforms the incoming I-PDU ID into the correct I-PDU ID for the static part's destination and then forwards the SDU via the PDU Router, see `PduR_IpduMRxIndication` in the [10, SWS PDU Router].⌋

[SWS_IpduM_00042] ⌈When a multiplexed I-PDU is received from the PDU Router the function `IpduM_RxIndication` uses the incoming I-PDU ID and the selector field to find out the correct I-PDU ID for the dynamic part's destination and then forwards the I-PDU via the PDU Router, see `PduR_IpduMRxIndication` in the [10, SWS PDU Router].⌋

[SWS_IpduM_00217] ⌈When a Container PDU is received from the PDU Router, the function `IpduM_RxIndication` forwards the contained I-PDUs via the PDU Router, using `PduR_IpduMRxIndication` (see [SWS_IpduM_00105]).⌋

[SWS_IpduM_00086] ⌈The function `IpduM_RxIndication` shall be callable in interrupt context, e.g. from receive interrupt.⌋

8.3.4 IpduM_TxConfirmation

[SWS_IpduM_00044] Definition of callback function IpduM_TxConfirmation

Upstream requirements: SRS_BSW_00369

⌈

Service Name	IpduM_TxConfirmation	
Syntax	`void IpduM_TxConfirmation ( PduIdType TxPduId, Std_ReturnType result )`	
Service ID [hex]	0x40	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the PDU that has been transmitted.
	result	`E_OK`: The PDU was transmitted successfully. `E_NOT_OK`: Transmission of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
Available via	IpduM.h

⌋

[SWS_IpduM_00088] ⌈The function `IpduM_TxConfirmation` shall translate the confirmation received from the PDU Router into confirmations for the I-PDUs which where contained in the sent multiplexed I-PDU or Container PDU.⌋

These confirmations are given again to the PDU Router that has to route them to COM.

[SWS_IpduM_00087] ⌈The function `IpduM_TxConfirmation` shall be callable in interrupt context, e.g. from a transmit interrupt.⌋

8.3.5 IpduM_TriggerTransmit

[SWS_IpduM_00060] Definition of callback function IpduM_TriggerTransmit

Upstream requirements: SRS_BSW_00369

⌈

Service Name	IpduM_TriggerTransmit	
Syntax	`Std_ReturnType IpduM_TriggerTransmit ( PduIdType TxPduId, PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the SDU that is requested to be transmitted.
Parameters (inout)	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: SDU has been copied and SduLength indicates the number of copied bytes. `E_NOT_OK`: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	
Available via	IpduM.h	

⌋

[SWS_IpduM_00090] ⌈Within the function `IpduM_TriggerTransmit`, the IpduM shall copy the contents of its I-PDU transmit buffer to the PDU buffer given by

PduInfoPtr->SduDataPtr and update PduInfoPtr->SduLength with length of the copied data accordingly.⌋

[SWS_IpduM_00091] ⌈The IpduM shall take care about the data consistency during providing the data.⌋

Use case: This function is used e.g. by the LIN Master for sending out a LIN frame. In this case, the trigger transmit can be initiated by the Master schedule table itself or a received LIN header.

This function is also used by the FlexRay Interface for requesting PDUs to be sent in static part (synchronous to the FlexRay global time).

[SWS_IpduM_00089] ⌈The function IpduM_TriggerTransmit shall be callable in interrupt context.⌋

8.4 Scheduled functions

Many of the functions of the IpduM module are called synchronous in the context of the upper layer (for transmission) and in the context of the lower layer (for reception). However, some functionality is excuted deferred. Therefore, it is performed in either IpduM_MainFunctionTx or IpduM_MainFunctionRx, dependent on the respective functionality is associated to the sender or receiver side respectively.

[SWS_IpduM_91002] Definition of scheduled function IpduM_MainFunctionTx ⌈

Service Name	IpduM_MainFunctionTx
Syntax	`void IpduM_MainFunctionTx ( void )`
Service ID [hex]	0x12
Description	This function performs the processing of the transmission activities that are not directly handled within the calls from PduR. Per configured IpduMMainFunctionTx instance one IpduM_MainFunctionTx_<shortName> shall be implemented. Hereby <shortName> is the short name of the IpduMMainFunctionTx configuration container in the ECU configuration.
Available via	IpduM_SchM.h

⌋

[SWS_IpduM_91001] Definition of scheduled function IpduM_MainFunctionRx ⌈

Service Name	IpduM_MainFunctionRx
Syntax	`void IpduM_MainFunctionRx ( void )`
Service ID [hex]	0x11

▽

△

Description	This function performs the processing of the reception activities that are not directly handled within the calls from PduR. Per configured IpduMMainFunctionRx instance one IpduM_MainFunctionRx_<shortName> shall be implemented. Hereby <shortName> is the short name of the IpduMMainFunctionRx configuration container in the ECU configuration.
Available via	IpduM_SchM.h

⌋

8.5 Expected interfaces

In this chapter, all interfaces required from other modules are listed.

8.5.1 Mandatory Interfaces

This chapter defines all interfaces that are required to fulfill the core functionality of the module.

[SWS_IpduM_00104] Definition of mandatory interfaces required by module IpduM

Upstream requirements: SRS_BSW_00009

⌈

API Function	***Header File***	***Description***
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.

⌋

8.5.2 Optional Interfaces

This chapter defines all interfaces that are required to fulfill an optional functionality of the module.

[SWS_IpduM_00105] Definition of optional interfaces requested by module IpduM

Upstream requirements: SRS_BSW_00009

⌈

API Function	***Header File***	***Description***
Det_ReportError	Det.h	Service to report development errors.
PduR_IpduMReleaseRxBuffer (draft)	PduR_IpduM.h	Indication from the upper layer to release the lower layer reception buffer.

▽

△

API Function	*Header File*	*Description*
PduR_IpduMRxIndication	PduR_IpduM.h	Indication of a received PDU from a lower layer communication interface module.
PduR_IpduMTransmit	PduR_IpduM.h	Requests transmission of a PDU.
PduR_IpduMTriggerTransmit	PduR_IpduM.h	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
PduR_IpduMTxConfirmation	PduR_IpduM.h	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.

⌋

8.5.3 Callout Interfaces

Not applicable.

8.5.4 Configurable interfaces

Not applicable.

9 Sequence diagrams

9.1 Transmission of a multiplexed I-PDU and Transmit confirmation

The following sequence chart shows a transmit request initiated by the COM layer. The transmit request is for an I-PDU which has to be transmitted within a multiplexed I-PDU. In the IpduM module is configured that this transmitted I-PDU triggers the sending of the multiplexed I-PDU.

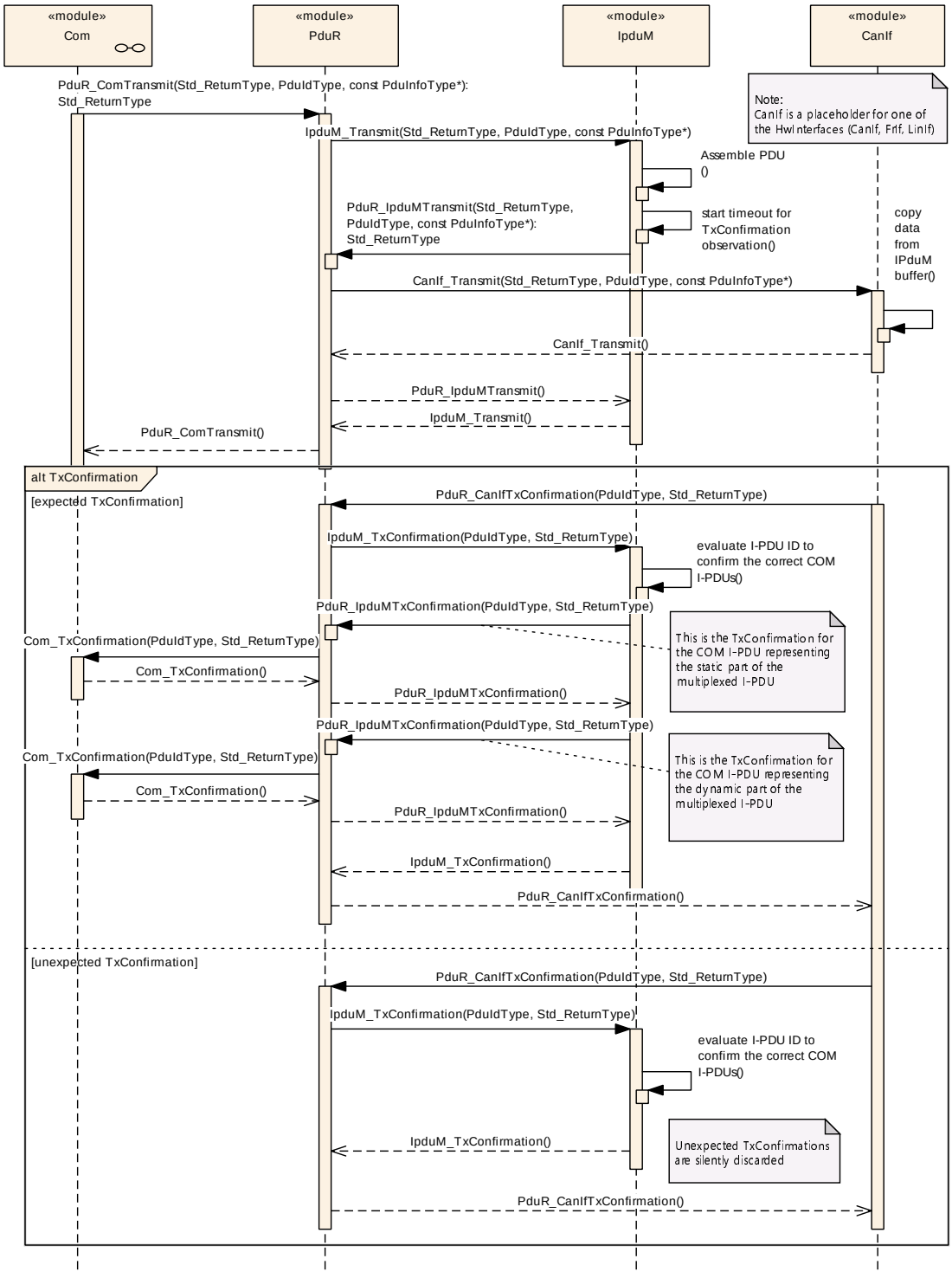


Figure 9.1: Transmission and confirmation of multiplexed I-PDU with triggering

9.2 Transmission of a multiplexed I-PDU without Trigger

The following sequence chart shows a transmit request initiated by the COM layer. Because of the configuration of the IpduM, no transmit request for the IpduM I-PDU takes place. For configuration see [ECUC_IpduM_00052].

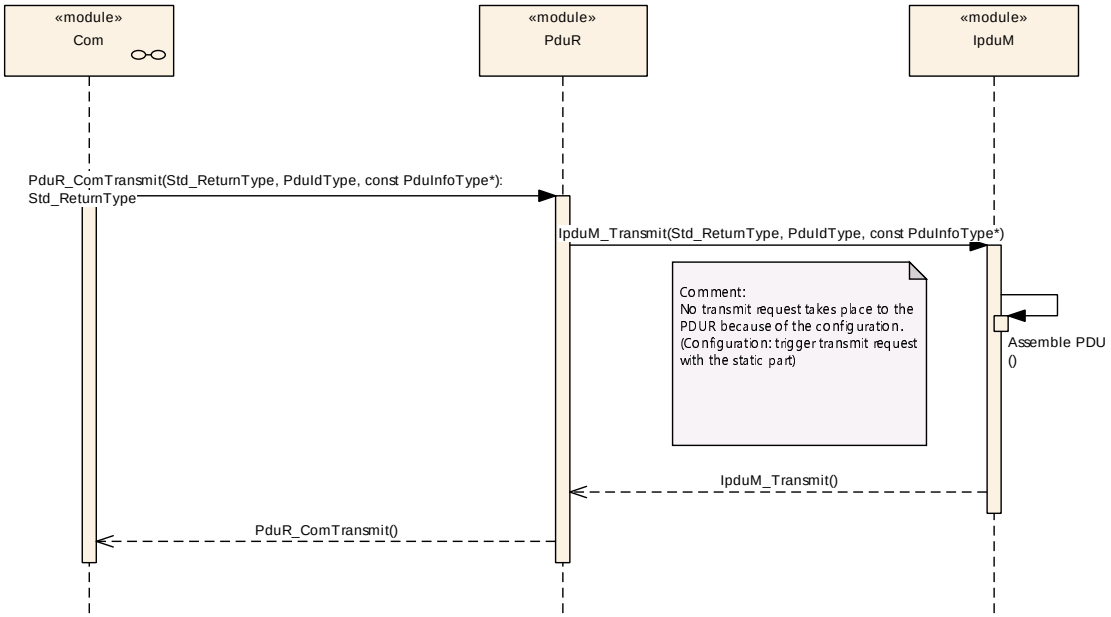


Figure 9.2: Transmission of a multiplexed I-PDU without triggering

9.3 Reception of the multiplexed I-PDU

The following sequence chart shows a reception of a multiplexed I-PDU. The I-PDU contains a static and a dynamic part and both are configured to create an RxIndication to the PDU Router module.

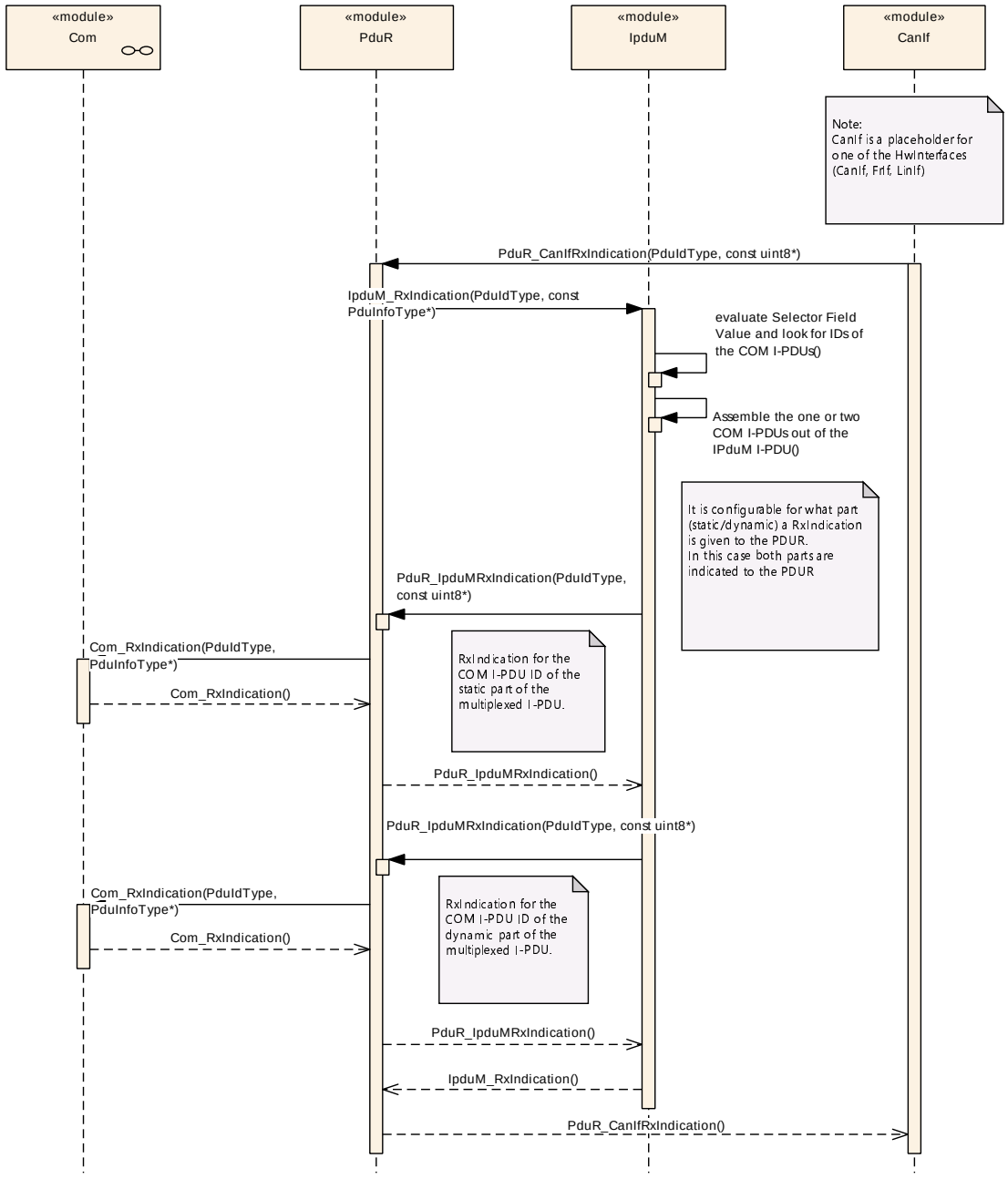


Figure 9.3: Reception of a multiplexed I-PDU

9.4 Trigger Transmit

The following sequence chart shows a Trigger Transmit request from an interface layer.

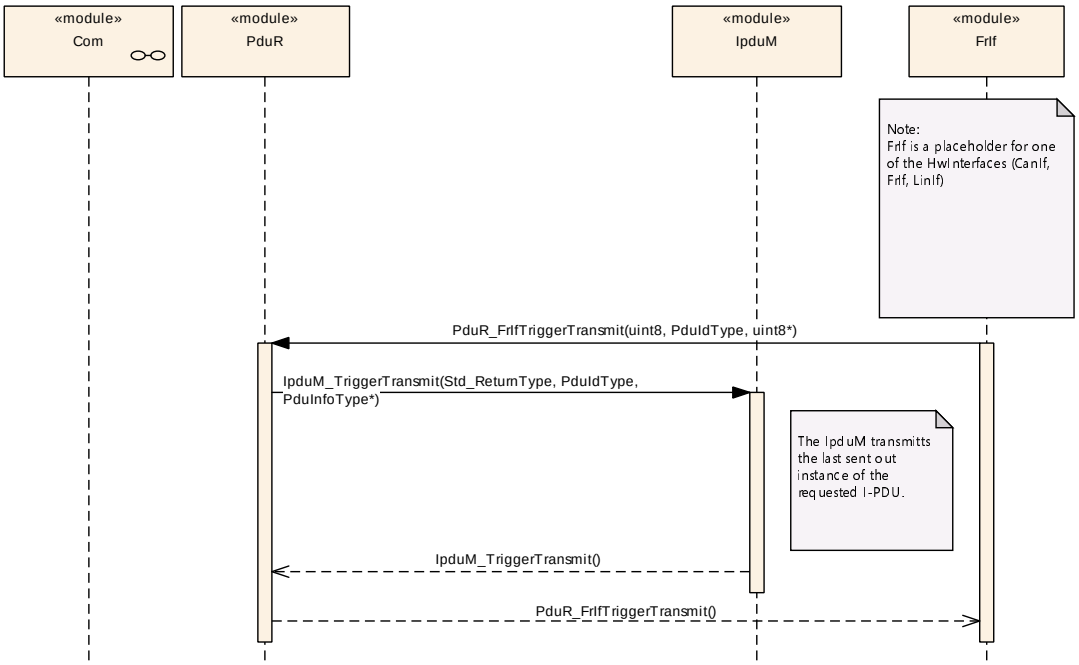


Figure 9.4: Trigger Transmit request from interface layer

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

Chapter 10.2 specifies the structure (containers) and the parameters of the module IpduM.

Chapter 10.3 specifies published information of the module IpduM.

10.1 How to read this chapter

For details refer to [4] Chapter 10.1 *"Introduction to configuration specification"*.

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapter 7 and Chapter 8.

10.2.1 Configuration overview

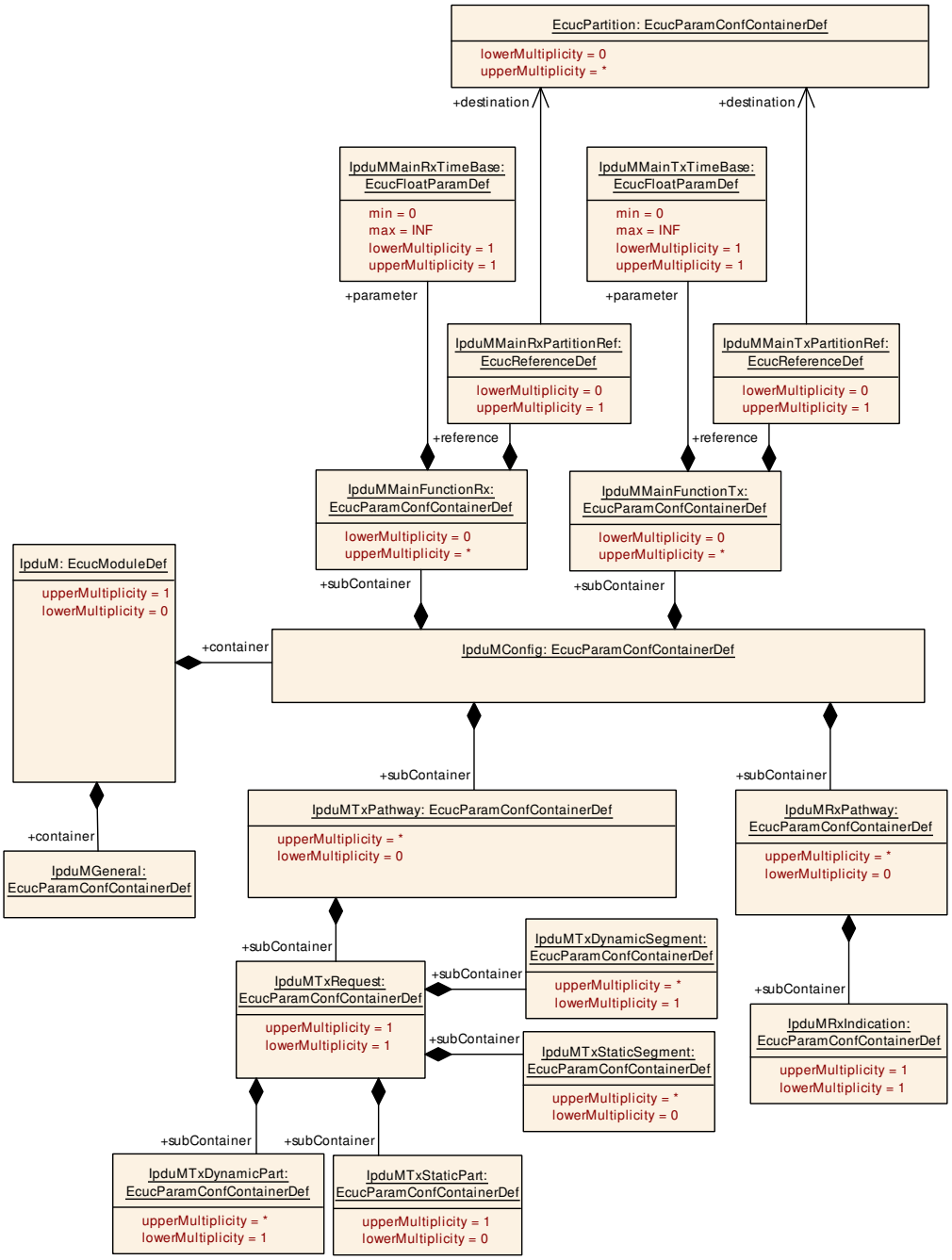


Figure 10.1: IpduM Configuration Overview (for I-PDU Multiplexing)

10.2.2 IpduM

[ECUC_IpduM_00204] Definition of EcucModuleDef IpduM ⌈

Module Name	IpduM
Description	Configuration of the IpduM (Ipdu Multiplexer) module.
Post-Build Selectable Support	true
Post-Build Loadable Support	true

Included Containers		
Container Name	Multiplicity	Description
IpduMConfig	1	This container contains the sub containers of the IpduM module. • The IpduMTxPathway subcontainer includes information about sent I-PDUs. • The IpduMRxPathway includes information about received I-PDUs. • The IpduMContainerTxPdu and IpduMContainedTxPdu include information about the sending of ContainerPdus. • The IpduMContainerRxPdu and IpduMContainedRxPdu include information about the reception of ContainerPdus.
IpduMGeneral	1	Contains the general configuration parameters of IpduM.

⌋

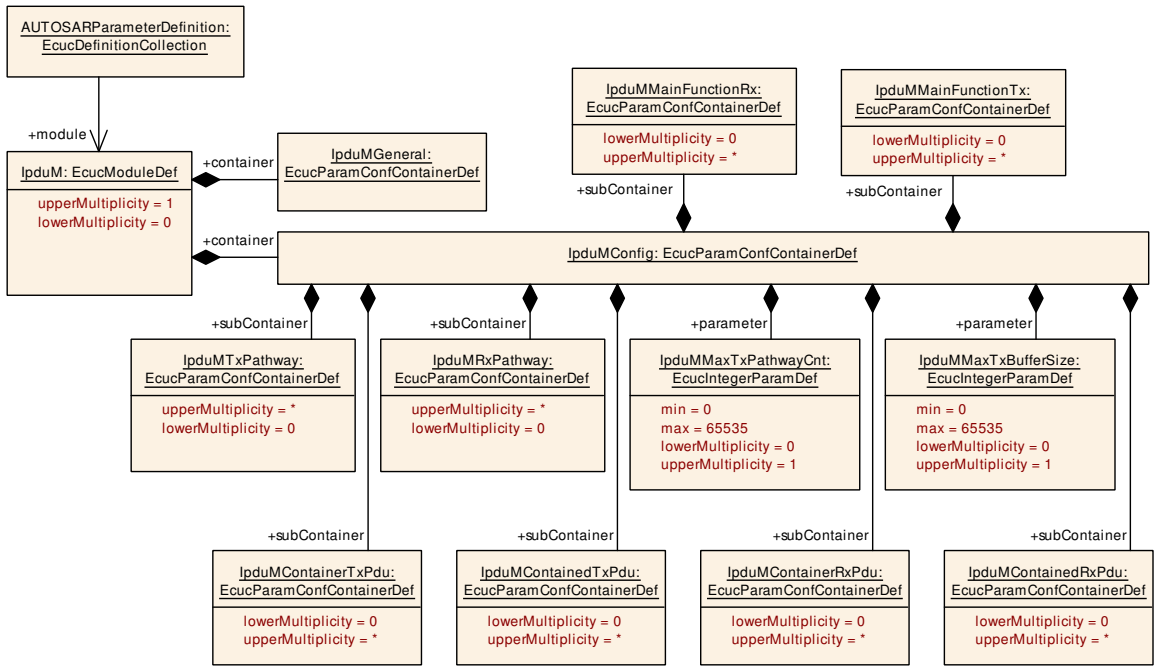


Figure 10.2: IpduM

10.2.3 IpduMConfig

[ECUC_IpduM_00059] Definition of EcucParamConfContainerDef IpduMConfig ⌈

Container Name	IpduMConfig
Parent Container	IpduM
Upstream mapping defined	–
Description	This container contains the sub containers of the IpduM module. • The IpduMTxPathway subcontainer includes information about sent I-PDUs. • The IpduMRxPathway includes information about received I-PDUs. • The IpduMContainerTxPdu and IpduMContainedTxPdu include information about the sending of ContainerPdus. • The IpduMContainerRxPdu and IpduMContainedRxPdu include information about the reception of ContainerPdus.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMMaxTxBufferSize	0..1	[ECUC_IpduM_00166]
IpduMMaxTxPathwayCnt	0..1	[ECUC_IpduM_00165]

Included Containers		
Container Name	**Multiplicity**	**Description**
IpduMContainedRxPdu	0..*	Configuration of a received contained Pdu.
IpduMContainedTxPdu	0..*	Configuration of a sender ContainedPdu.
IpduMContainerRxPdu	0..*	Configuration of a receiver ContainerPdu which may collect several ContainedPdus.
IpduMContainerTxPdu	0..*	Configuration of a transmitted container Pdu.
IpduMMainFunctionRx	0..*	Each element of this container defines one instance IpduM_Main FunctionRx, in case multi-core distribution feature is active.
IpduMMainFunctionTx	0..*	Each element of this container defines one instance IpduM_Main FunctionTx, in case multi-core distribution feature is active (mutual exclusive to ComTimeBase).
IpduMRxPathway	0..*	includes information about received I-PDUs
IpduMTxPathway	0..*	includes information about sent I-PDUs

⌋

[ECUC_IpduM_00166] Definition of EcucIntegerParamDef IpduMMaxTxBuffer Size ⌈

Parameter Name	IpduMMaxTxBufferSize	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Maximum total size of all Tx buffers. This parameter is needed only in case of post-build loadable implementation using static memory allocation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00165] Definition of EcucIntegerParamDef IpduMMaxTxPathwayCnt ⌈

Parameter Name	IpduMMaxTxPathwayCnt	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Maximum number of transmitted IPdus. This parameter is needed only in case of post-build loadable implementation using static memory allocation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

10.2.4 IpduMGeneral

[ECUC_IpduM_00130] Definition of EcucParamConfContainerDef IpduMGeneral ⌈

Container Name	IpduMGeneral
Parent Container	IpduM
Upstream mapping defined	–
Description	Contains the general configuration parameters of IpduM.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMContainedTxPduPriorityHandling	0..1	[ECUC_IpduM_00209]
IpduMContainerTxPduMaxFillUpStrategy	0..1	[ECUC_IpduM_00220]
IpduMDevErrorDetect	1	[ECUC_IpduM_00132]
IpduMHeaderByteOrder	0..1	[ECUC_IpduM_00197]
IpduMMaxTransmitRetries	1	[ECUC_IpduM_00219]
IpduMMetaDataSupport	0..1	[ECUC_IpduM_00205]
IpduMStaticPartExists	1	[ECUC_IpduM_00133]

No Included Containers

⌋

[ECUC_IpduM_00209] Definition of EcucBooleanParamDef IpduMContainedTxPduPriorityHandling ⌈

Parameter Name	IpduMContainedTxPduPriorityHandling
Parent Container	IpduMGeneral
Upstream mapping defined	–
Description	This parameter enables/disables handling of priority for IpduMContainedTxPdu's with IpduMContainedTxPduCollectionSemantics IPDUM_LAST_IS_BEST. • true: enabled • false: disabled
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_IpduM_00220] Definition of EcucBooleanParamDef IpduMContainerTxPduMaxFillUpStrategy ⌈

Parameter Name	IpduMContainerTxPduMaxFillUpStrategy
Parent Container	IpduMGeneral
Upstream mapping defined	–
Description	This parameter enables/disables handling of the maximum fill up strategy for Ipdu MContainerTxPdu's that contain IpduMContainedTxPdu with IpduMContainedTxPdu CollectionSemantics IPDUM_LAST_IS_BEST. true: enabled false: disabled Dependency: This parameter is only valid if IpduMContainedTxPduPriorityHandling is set to TRUE and IpduMContainerHeaderSize of the linked IpduMContainerTxPdu is not set to IPDUM_HEADERTYPE_NONE.
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_IpduM_00132] Definition of EcucBooleanParamDef IpduMDevErrorDetect ⌈

Parameter Name	IpduMDevErrorDetect
Parent Container	IpduMGeneral
Upstream mapping defined	–
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_IpduM_00197] Definition of EcucEnumerationParamDef IpduMHeaderByteOrder ⌈

Parameter Name	IpduMHeaderByteOrder	
Parent Container	IpduMGeneral	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This parameter defines the ByteOrder of the headers inside a Container I-PDU.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	IPDUM_BIG_ENDIAN	Headers inside a Container I-PDU shall be ordered big endian.
	IPDUM_LITTLE_ENDIAN	Headers inside a Container I-PDU shall be ordered little endian.
Pre-Compile only	true	

⌋

[ECUC_IpduM_00219] Definition of EcucIntegerParamDef IpduMMaxTransmitRetries ⌈

Parameter Name	IpduMMaxTransmitRetries	
Parent Container	IpduMGeneral	
Upstream mapping defined	–	
Description	Maximum number of retries to send a container message in case PduR_IPduMTransmit returns E_NOT_OK.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	10	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00205] Definition of EcucBooleanParamDef IpduMMetaDataSupport ⌈

Parameter Name	IpduMMetaDataSupport
Parent Container	IpduMGeneral
Upstream mapping defined	–
Description	This parameter enables/disables the support of meta-data feature. true: enabled false: disabled
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_IpduM_00133] Definition of EcucBooleanParamDef IpduMStaticPartExists ⌈

Parameter Name	IpduMStaticPartExists
Parent Container	IpduMGeneral
Upstream mapping defined	–
Description	This is to allow optimizations in the case the IpduM will never be used with a static part. Note that this is a pre-compile option. If this is set to False then it will not be possible to add static parts after compilation. True: A static part may exist. False: A static part will never exist.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

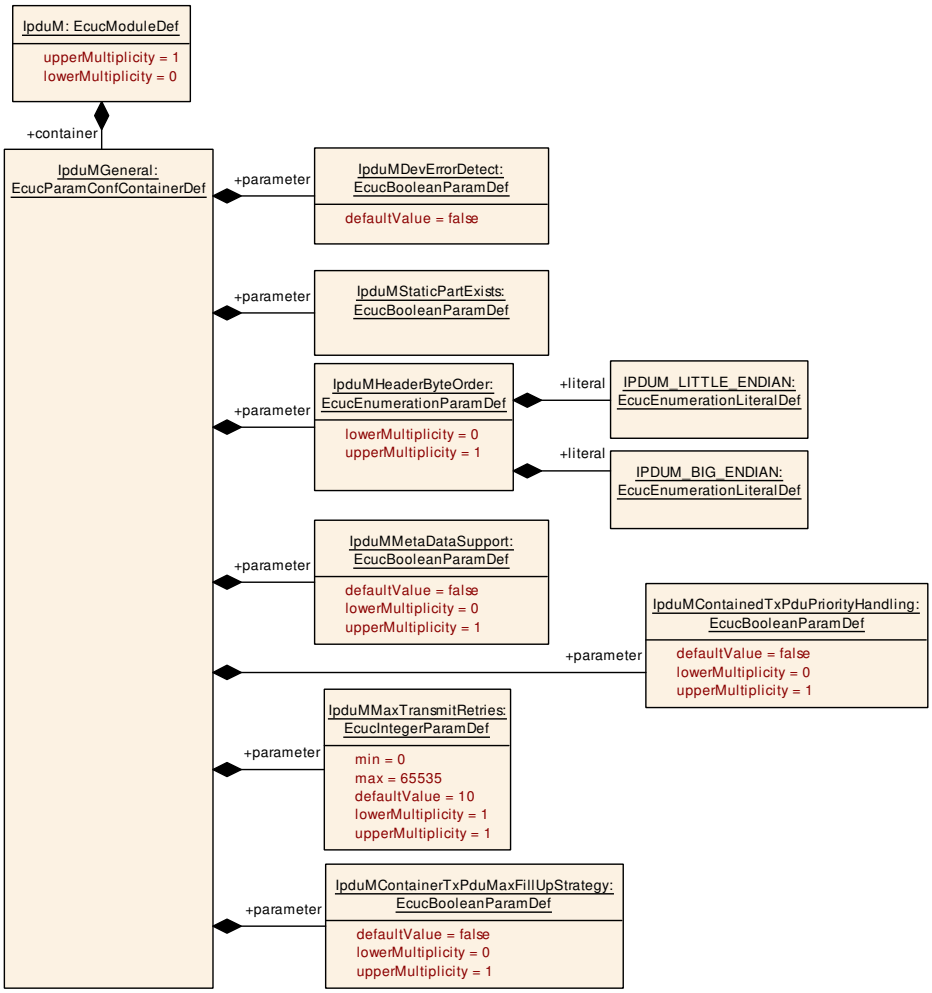


Figure 10.3: IpduMGeneral

10.2.5 IpduMTxPathway

[ECUC_IpduM_00070] Definition of EcucParamConfContainerDef IpduMTxPathway ⌈

Container Name	IpduMTxPathway	
Parent Container	IpduMConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters transmitted I-PDUs by the IpduM module.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Description**
IpduMTxRequest	1	configuration for a TxRequest

⌋

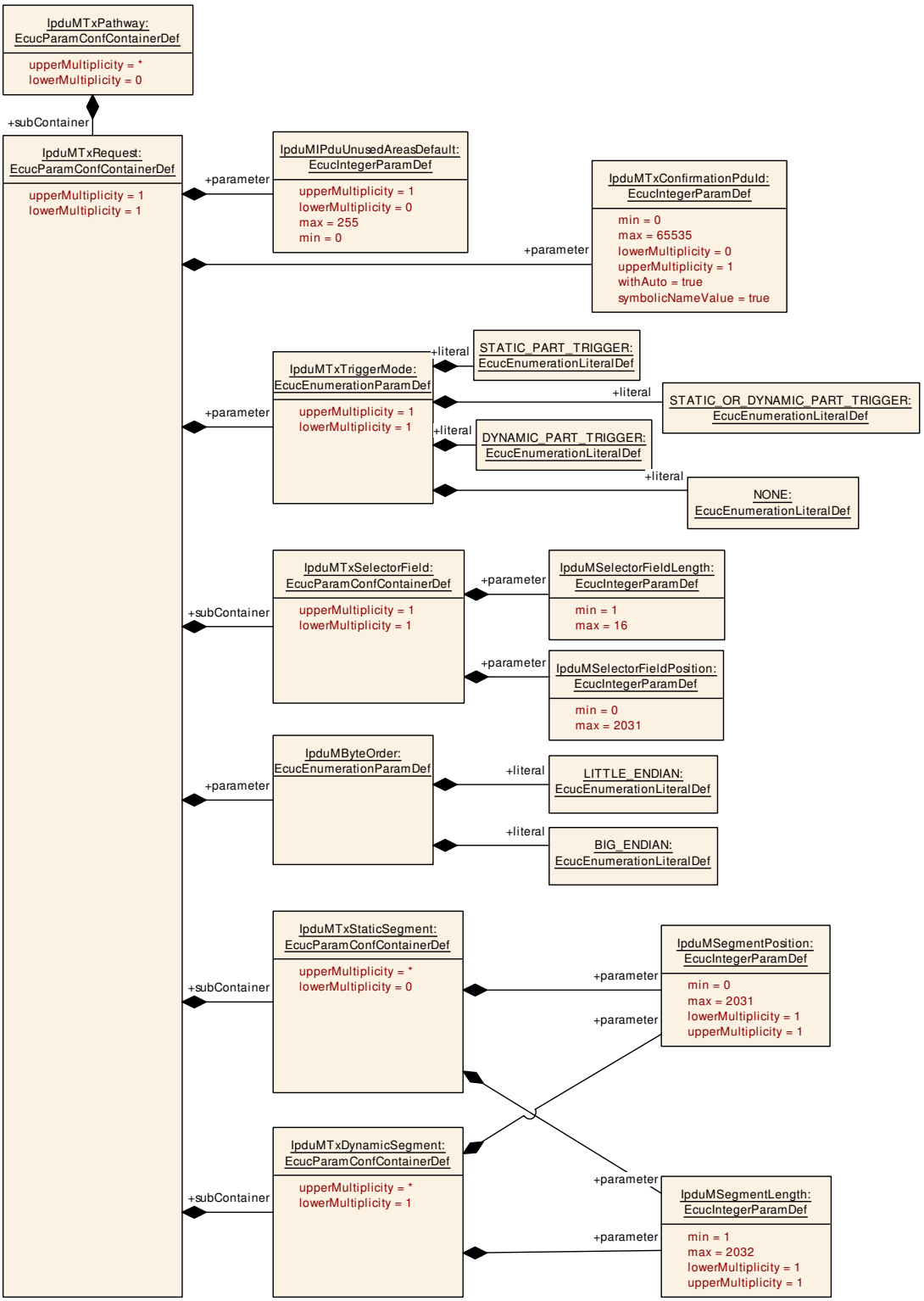


Figure 10.4: IpduMTxPathwayRequest part 1

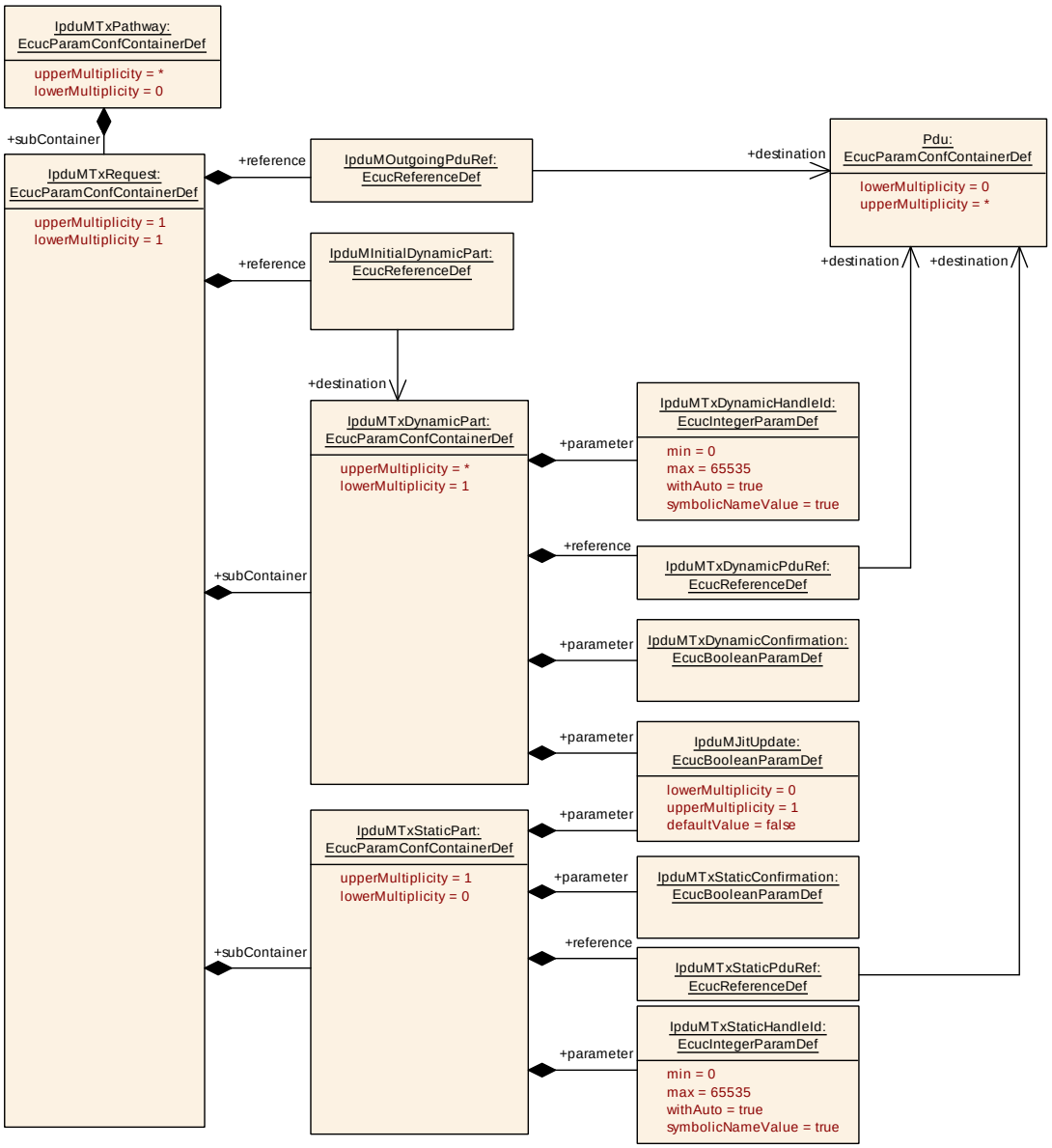


Figure 10.5: IpduMTxPathwayRequest part 2

10.2.6 IpduMTxRequest

[ECUC_IpduM_00052] Definition of EcucParamConfContainerDef IpduMTxRequest ⌈

Container Name	IpduMTxRequest
Parent Container	IpduMTxPathway
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This container is used to specify the configuration for Transmit requests. There will be one instance of this container for each I-PDU that can be requested for transmission (the outgoing I-PDUs) by the IpduM.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMByteOrder	1	[ECUC_IpduM_00162]
IpduMIPduUnusedAreasDefault	0..1	[ECUC_IpduM_00121]
IpduMTxConfirmationPduId	0..1	[ECUC_IpduM_00158]
IpduMTxTriggerMode	1	[ECUC_IpduM_00125]
IpduMInitialDynamicPart	1	[ECUC_IpduM_00157]
IpduMOutgoingPduRef	1	[ECUC_IpduM_00120]

Included Containers		
Container Name	**Multiplicity**	**Description**
IpduMTxDynamicPart	1..*	This (These) included container(s) must exist for each unique selector field value for this outgoing IpduM I-PDU.
IpduMTxDynamicSegment	1..*	The dynamic part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPduRef) can be separated into several segments. For each segment one IpduMTxDynamicSegment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied out of the source I-Pdu that is referenced in the IpduMTxDynamicPart container and will be copied to the same location in the multiplexed outgoing I-Pdu. The segment layout for all dynamic Parts is always identical.
IpduMTxSelectorField	1	This contains the location and the length of the selector field.
IpduMTxStaticPart	0..1	This included container configures the static part, if present.
IpduMTxStaticSegment	0..*	The static part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPduRef) can be separated into several segments. For each segment one IpduMTxStaticSegment container shall be created that contains the location and the length of the segment. Please note that each segment in the source I-Pdu that is referenced in the IpduMTxStaticPart container will be copied to the same location in the multiplexed outgoing I-Pdu.

⌋

For parameter table [ECUC_IpduM_00162] IpduMByteOrder, see definition below container IpduMRxIndication.

[ECUC_IpduM_00121] Definition of EcucIntegerParamDef IpduMIPduUnusedAreasDefault ⌈

Parameter Name	IpduMIPduUnusedAreasDefault	
Parent Container	IpduMTxRequest	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	IpduM module fills not used areas of an I-PDU with this bit-pattern If this attribute is omitted the IpduM module does not fill the I-PDU.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00158] Definition of EcucIntegerParamDef IpduMTxConfirmationPduId ⌈

Parameter Name	IpduMTxConfirmationPduId	
Parent Container	IpduMTxRequest	
Upstream mapping defined	–	
Description	Handle Id used by the PduR for confirmation (IpduM_TxConfirmation) and for Trigger Transmit (IpduM_TriggerTransmit). The existence of this parameter is essential for the PduR generation tool to actually find a symbolicNameValue for the OutgoingPdu.	
Multiplicity	0..1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00125] Definition of EcucEnumerationParamDef IpduMTxTriggerMode ⌈

Parameter Name	IpduMTxTriggerMode	
Parent Container	IpduMTxRequest	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Selects whether to send the multiplexed I-PDU immediately or at some later date.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DYNAMIC_PART_TRIGGER	Writing the I-PDU representing the dynamic part does trigger a sending of the I-PDU.
	NONE	Only the buffer in the IpduM are written but not send is triggered, used for IpduM I-PDUs which are requested by TriggerTransmit.

▽

△

	STATIC_OR_DYNAMIC_PART_TRIGGER	Writing the I-PDU representing the static or the dynamic part does trigger a sending of the I-PDU.
	STATIC_PART_TRIGGER	Writing the I-PDU representing the static part does trigger a sending of the I-PDU.
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00157] Definition of EcucReferenceDef IpduMInitialDynamicPart

⌈

Parameter Name	IpduMInitialDynamicPart	
Parent Container	IpduMTxRequest	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to the dynamic part that shall be used to initialize this multiplexed TX-I-PDU.	
Multiplicity	1	
Type	Reference to IpduMTxDynamicPart	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00120] Definition of EcucReferenceDef IpduMOutgoingPduRef ⌈

Parameter Name	IpduMOutgoingPduRef	
Parent Container	IpduMTxRequest	
Upstream mapping defined	–	
Description	Reference to the PDU defining the outgoing I-PDU. When the outgoing I-PDU is sent this is the I-PDU ID to give it. It is the IpduM I-PDU ID of the assembled I-PDU.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.7 IpduMTxDynamicPart

[ECUC_IpduM_00056] Definition of EcucParamConfContainerDef IpduMTxDynamicPart ⌈

Container Name	IpduMTxDynamicPart
Parent Container	IpduMTxRequest
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Configuration parameters for an instance of a TxRequest call into the IpduM. When a Tx Request with the IpduMTxDynamicHandleId is received by the IpduM, all segments (defined in the IpduMDynamicSegment container) are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honored. This container is used by the dynamic part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the dynamic part.

▽

△

Multiplicity	1..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMJitUpdate	0..1	[ECUC_IpduM_00167]
IpduMTxDynamicConfirmation	1	[ECUC_IpduM_00163]
IpduMTxDynamicHandleId	1	[ECUC_IpduM_00127]
IpduMTxDynamicPduRef	1	[ECUC_IpduM_00126]

No Included Containers

⌋

[ECUC_IpduM_00167] Definition of EcucBooleanParamDef IpduMJitUpdate ⌈

Parameter Name	IpduMJitUpdate	
Parent Container	IpduMTxDynamicPart, IpduMTxStaticPart	
Upstream mapping defined	–	
Description	If configured to true fetch the data of this part Just-In-Time via the triggerTransmit API of the PduR.	
Multiplicity	0..1	
Type	EcucBooleanParamDef	
Default value	false	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00163] Definition of EcucBooleanParamDef IpduMTxDynamicConfirmation ⌈

Parameter Name	IpduMTxDynamicConfirmation	
Parent Container	IpduMTxDynamicPart	
Upstream mapping defined	–	
Description	A transmit request can be confirmed by the lower layer. If this parameter is set to true a confirmation of the I-PDU in COM representing the dynamic part is generated.	
Multiplicity	1	
Type	EcucBooleanParamDef	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00127] Definition of EcucIntegerParamDef IpduMTxDynamicHandleId ⌈

Parameter Name	IpduMTxDynamicHandleId	
Parent Container	IpduMTxDynamicPart	
Upstream mapping defined	–	
Description	This defines an incoming handle id. When the handle of an incoming Tx Request matches this id, the configured dynamic segments are copied and the IpduMTxTrigger Mode is honored.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00126] Definition of EcucReferenceDef IpduMTxDynamicPduRef ⌈

Parameter Name	IpduMTxDynamicPduRef	
Parent Container	IpduMTxDynamicPart	
Upstream mapping defined	–	
Description	Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.8 IpduMTxDynamicSegment

[ECUC_IpduM_00168] Definition of EcucParamConfContainerDef IpduMTxDynamicSegment ⌈

Container Name	IpduMTxDynamicSegment
Parent Container	IpduMTxRequest
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	The dynamic part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPdu Ref) can be separated into several segments. For each segment one IpduMTxDynamic Segment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied out of the source I-Pdu that is referenced in the IpduMTxDynamicPart container and will be copied to the same location in the multiplexed outgoing I-Pdu. The segment layout for all dynamic Parts is always identical.
Multiplicity	1..*

▽

△

Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSegmentLength	1	[ECUC_IpduM_00114]
IpduMSegmentPosition	1	[ECUC_IpduM_00159]

No Included Containers

⌋

For parameter table [ECUC_IpduM_00114] IpduMSegmentLength, see definition below container IpduMRxDynamicSegment.

For parameter table [ECUC_IpduM_00159] IpduMSegmentPosition, see definition below container IpduMRxDynamicSegment.

10.2.9 IpduMTxStaticPart

[ECUC_IpduM_00082] Definition of EcucParamConfContainerDef IpduMTxStaticPart ⌈

Container Name	IpduMTxStaticPart	
Parent Container	IpduMTxRequest	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Configuration parameters for an instance of a Tx_Request call into the IpduM. When a Tx Request with the IpduMTxStaticHandleId is received by the IpduM, all segments (defined in the IpduMStaticSegment container) are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honored. This container is used for the static part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the static part if it exists.	
Multiplicity	0..1	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMJitUpdate	0..1	[ECUC_IpduM_00167]
IpduMTxStaticConfirmation	1	[ECUC_IpduM_00164]
IpduMTxStaticHandleId	1	[ECUC_IpduM_00129]
IpduMTxStaticPduRef	1	[ECUC_IpduM_00128]

No Included Containers

⌋

For parameter table [ECUC_IpduM_00167] IpduMJitUpdate, see definition below container IpduMTxDynamicPart.

[ECUC_IpduM_00164] Definition of EcucBooleanParamDef IpduMTxStaticConfirmation ⌈

Parameter Name	IpduMTxStaticConfirmation	
Parent Container	IpduMTxStaticPart	
Upstream mapping defined	–	
Description	A transmit request can be confirmed by the lower layer. If this parameter is set to true a confirmation of the I-PDU in COM representing the static part is generated.	
Multiplicity	1	
Type	EcucBooleanParamDef	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00129] Definition of EcucIntegerParamDef IpduMTxStaticHandleId ⌈

Parameter Name	IpduMTxStaticHandleId	
Parent Container	IpduMTxStaticPart	
Upstream mapping defined	–	
Description	This defines an incoming handle id. When the handle of an incoming Tx Request matches this id, the configured static segments are copied and the IpduMTxTrigger Mode is honored.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00128] Definition of EcucReferenceDef IpduMTxStaticPduRef ⌈

Parameter Name	IpduMTxStaticPduRef	
Parent Container	IpduMTxStaticPart	
Upstream mapping defined	–	
Description	Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.10 IpduMTxStaticSegment

[ECUC_IpduM_00171] Definition of EcucParamConfContainerDef IpduMTxStaticSegment ⌈

Container Name	IpduMTxStaticSegment	
Parent Container	IpduMTxRequest	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The static part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPdu Ref) can be separated into several segments. For each segment one IpduMTxStatic Segment container shall be created that contains the location and the length of the segment. Please note that each segment in the source I-Pdu that is referenced in the IpduMTx StaticPart container will be copied to the same location in the multiplexed outgoing I-Pdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSegmentLength	1	[ECUC_IpduM_00114]
IpduMSegmentPosition	1	[ECUC_IpduM_00159]

No Included Containers

⌋

For parameter table [ECUC_IpduM_00114] IpduMSegmentLength, see definition below container IpduMRxDynamicSegment.

For parameter table [ECUC_IpduM_00159] IpduMSegmentPosition, see definition below container IpduMRxDynamicSegment.

10.2.11 IpduMTxSelectorField

[ECUC_IpduM_00221] Definition of EcucParamConfContainerDef IpduMTxSelectorField ⌈

Container Name	IpduMTxSelectorField
Parent Container	IpduMTxRequest
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This contains the location and the length of the selector field.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSelectorFieldLength	1	[ECUC_IpduM_00160]
IpduMSelectorFieldPosition	1	[ECUC_IpduM_00161]

No Included Containers

⌋

For parameter table [ECUC_IpduM_00160] IpduMSelectorFieldLength, see definition below container IpduMRxSelectorField.

For parameter table [ECUC_IpduM_00161] IpduMSelectorFieldPosition, see definition below container IpduMRxSelectorField.

10.2.12 IpduMRxPathway

[ECUC_IpduM_00071] Definition of EcucParamConfContainerDef IpduMRxPathway ⌈

Container Name	IpduMRxPathway	
Parent Container	IpduMConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters received I-PDUs by the IpduM module.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Description**
IpduMRxIndication	1	configuration for RxIndication

⌋

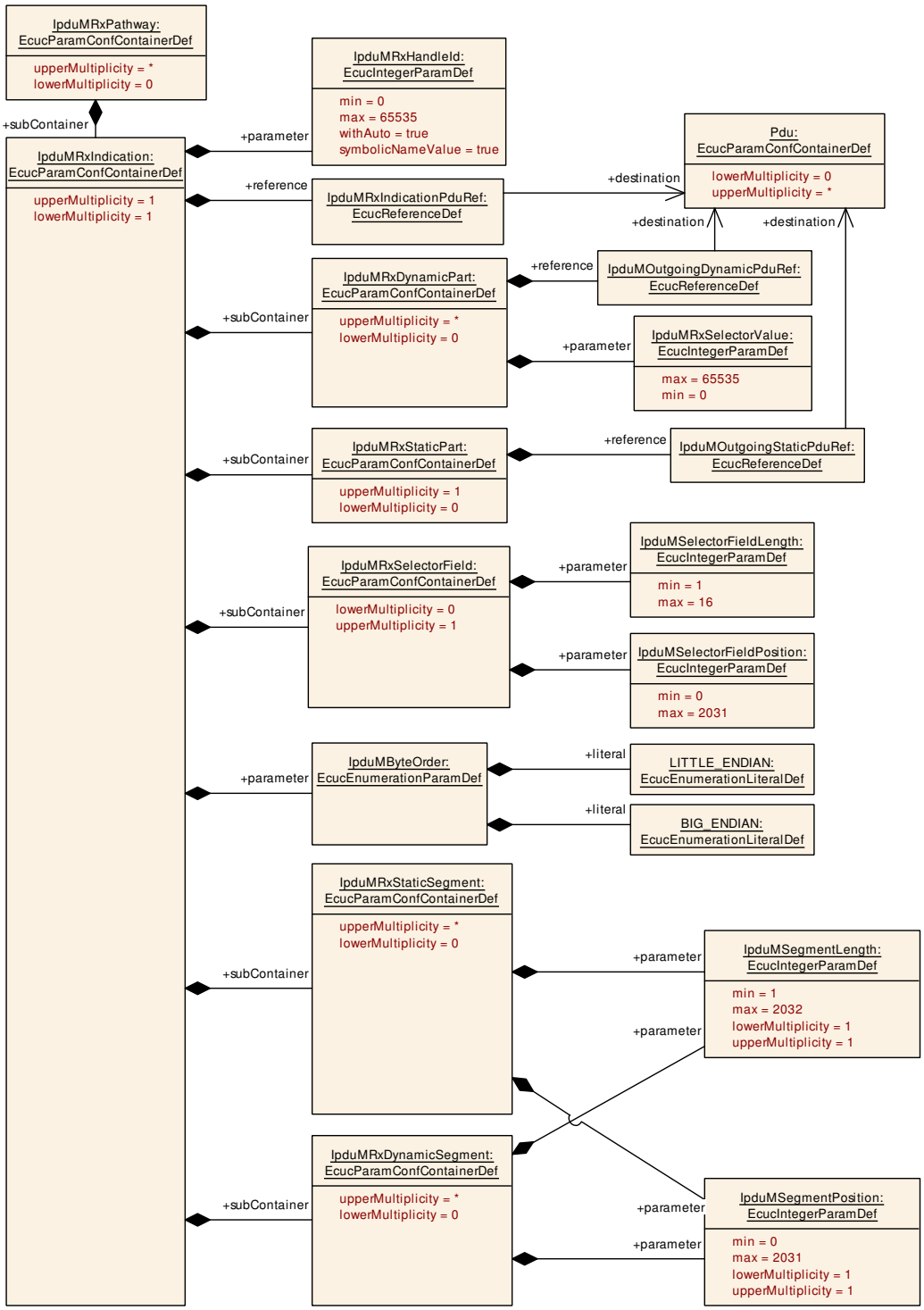


Figure 10.6: IpduMRxPathwayIndication

10.2.13 IpduMRxIndication

[ECUC_IpduM_00047] Definition of EcucParamConfContainerDef IpduMRxIndication ⌈

Container Name	IpduMRxIndication
Parent Container	IpduMRxPathway
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Contains the configuration for incoming RxIndication calls.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMByteOrder	1	[ECUC_IpduM_00162]
IpduMRxHandleId	1	[ECUC_IpduM_00109]
IpduMRxIndicationPduRef	1	[ECUC_IpduM_00108]

Included Containers		
Container Name	**Multiplicity**	**Description**
IpduMRxDynamicPart	0..*	Each of these containers contains the configuration for one value of the selector field for the incoming I-PDU's dynamic part.
IpduMRxDynamicSegment	0..*	The dynamic part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndicationPduRef) can be separated into several segments. For each segment one IpduMRxDynamicSegment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxDynamicPart container and will be copied from the same location in the multiplexed incoming I-Pdu. The segment layout for all dynamic Parts is always identical.
IpduMRxSelectorField	0..1	This contains the location and the length of the selector field.
IpduMRxStaticPart	0..1	This contains the configuration for the incoming I-PDU's static part. If the incoming I-PDU has no static part then this is omitted.
IpduMRxStaticSegment	0..*	The static part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndicationPduRef) can be separated into several segments. For each segment one IpduMRxStaticSegment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxStaticPart container and will be copied from the same location in the multiplexed incoming I-Pdu.

⌋

[ECUC_IpduM_00162] Definition of EcucEnumerationParamDef IpduMByteOrder

⌈

Parameter Name	IpduMByteOrder
Parent Container	IpduMRxIndication, IpduMTxRequest
Upstream mapping defined	[11, CP_TPS_SystemTemplate]

▽

△

Description	This parameter defines the ByteOrder for all segments (static and dynamic part) and for the selectorField within the MultiplexedPdu. The absolute position of a segment in the MultiplexedIPdu is determined by the definition of the ByteOrder parameter: If BIG_ENDIAN is specified, the Segment Position indicates the bit position of the most significant bit in an IPDU. If LITTLE_ ENDIAN is specified, the SegmentPosition indicates the bit position of the least significant bit in an IPDU.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	BIG_ENDIAN	Most significant byte shall be stored at the lowest address.
	LITTLE_ENDIAN	Most significant byte shall be stored at the highest address.
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00109] Definition of EcucIntegerParamDef IpduMRxHandleId ⌈

Parameter Name	IpduMRxHandleId	
Parent Container	IpduMRxIndication	
Upstream mapping defined	–	
Description	This is the I-PDU ID of the incoming I-PDU. If an incoming RxIndication's I-PDU ID matches this value then it is unpacked according to the specification in this container.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00108] Definition of EcucReferenceDef IpduMRxIndicationPduRef ⌈

Parameter Name	IpduMRxIndicationPduRef	
Parent Container	IpduMRxIndication	
Upstream mapping defined	–	
Description	Reference to the received Pdu representation in the ECU Configuration Description exchange file.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.14 IpduMRxDynamicPart

[ECUC_IpduM_00048] Definition of EcucParamConfContainerDef IpduMRxDynamicPart ⌈

Container Name	IpduMRxDynamicPart	
Parent Container	IpduMRxIndication	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IpduMRxSelector Value, the new outgoing I-PDU for the dynamic part is constructed as defined by the segments (defined in the IpduMDynamicSegment container) and sent out with the I-PDU ID referenced by IpduMOutgoingDynamicPduRef. In case no dynamic part shall be extracted from this received I-PDU this container does not exist. This use-case can occur in case a MultiplexedIPdu is received by an ECU which is only interested in the static part of the MultiplexedIPdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
IpduMRxSelectorValue	1	[ECUC_IpduM_00113]
IpduMOutgoingDynamicPduRef	1	[ECUC_IpduM_00112]

No Included Containers

⌋

[ECUC_IpduM_00113] Definition of EcucIntegerParamDef IpduMRxSelectorValue ⌈

Parameter Name	IpduMRxSelectorValue	
Parent Container	IpduMRxDynamicPart	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This is the selector value that this container refers to.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00112] Definition of EcucReferenceDef IpduMOutgoingDynamic PduRef ⌈

Parameter Name	IpduMOutgoingDynamicPduRef	
Parent Container	IpduMRxDynamicPart	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent PDU representation in the ECU Configuration Description exchange file.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.15 IpduMRxDynamicSegment

[ECUC_IpduM_00170] Definition of EcucParamConfContainerDef IpduMRxDynamicSegment ⌈

Container Name	IpduMRxDynamicSegment	
Parent Container	IpduMRxIndication	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The dynamic part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndication PduRef) can be separated into several segments. For each segment one IpduMRx DynamicSegment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxDynamicPart container and will be copied from the same location in the multiplexed incoming I-Pdu. The segment layout for all dynamic Parts is always identical.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSegmentLength	1	[ECUC_IpduM_00114]
IpduMSegmentPosition	1	[ECUC_IpduM_00159]

No Included Containers

⌋

[ECUC_IpduM_00114] Definition of EcucIntegerParamDef IpduMSegmentLength ⌈

Parameter Name	IpduMSegmentLength	
Parent Container	IpduMRxDynamicSegment, IpduMRxStaticSegment, IpduMTxDynamicSegment, IpduMTxStaticSegment	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Length of the segment in bits.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 2032	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00159] Definition of EcucIntegerParamDef IpduMSegmentPosition ⌈

Parameter Name	IpduMSegmentPosition
Parent Container	IpduMRxDynamicSegment, IpduMRxStaticSegment, IpduMTxDynamicSegment, IpduMTxStaticSegment

▽

△

Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Segments bit position in the multiplexed Pdu.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 2031	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

10.2.16 IpduMRxStaticPart

[ECUC_IpduM_00049] Definition of EcucParamConfContainerDef IpduMRxStaticPart ⌈

Container Name	IpduMRxStaticPart	
Parent Container	IpduMRxIndication	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This container contains the configuration for the static part of incoming RxIndication calls. On reception, the new outgoing I-PDU for the static part is constructed as defined by the segments (defined in the IpduMStaticSegment container) and sent out with the I-PDU ID referenced by IpduMOutgoingStaticPduRef.	
Multiplicity	0..1	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMOutgoingStaticPduRef	1	[ECUC_IpduM_00115]

No Included Containers

⌋

[ECUC_IpduM_00115] Definition of EcucReferenceDef IpduMOutgoingStaticPduRef ⌈

Parameter Name	IpduMOutgoingStaticPduRef	
Parent Container	IpduMRxStaticPart	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.17 IpduMRxStaticSegment

[ECUC_IpduM_00169] Definition of EcucParamConfContainerDef IpduMRxStaticSegment ⌈

Container Name	IpduMRxStaticSegment	
Parent Container	IpduMRxIndication	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The static part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndicationPdu Ref) can be separated into several segments. For each segment one IpduMRxStatic Segment container shall be created that contains the location and the length of the segment. Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxStaticPart container and will be copied from the same location in the multiplexed incoming I-Pdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSegmentLength	1	[ECUC_IpduM_00114]
IpduMSegmentPosition	1	[ECUC_IpduM_00159]

No Included Containers

⌋

For parameter table [ECUC_IpduM_00114] IpduMSegmentLength, see definition below container IpduMRxDynamicSegment.

For parameter table [ECUC_IpduM_00159] IpduMSegmentPosition, see definition below container IpduMRxDynamicSegment.

10.2.18 IpduMRxSelectorField

[ECUC_IpduM_00222] Definition of EcucParamConfContainerDef IpduMRxSelectorField ⌈

Container Name	IpduMRxSelectorField	
Parent Container	IpduMRxIndication	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This contains the location and the length of the selector field.	
Multiplicity	0..1	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMSelectorFieldLength	1	[ECUC_IpduM_00160]
IpduMSelectorFieldPosition	1	[ECUC_IpduM_00161]

No Included Containers

⌋

[ECUC_IpduM_00160] Definition of EcucIntegerParamDef IpduMSelectorFieldLength ⌈

Parameter Name	IpduMSelectorFieldLength	
Parent Container	IpduMRxSelectorField, IpduMTxSelectorField	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Length of the selector field in bits.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 16	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00161] Definition of EcucIntegerParamDef IpduMSelectorFieldPosition ⌈

Parameter Name	IpduMSelectorFieldPosition	
Parent Container	IpduMRxSelectorField, IpduMTxSelectorField	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Selector field bit position in the multiplexed Pdu. Range: 0..63 for CAN/ LIN I-PDUs, 0..511 for CAN FD I-PDUs, 0..2031 for FlexRay I-PDUs.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 2031	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

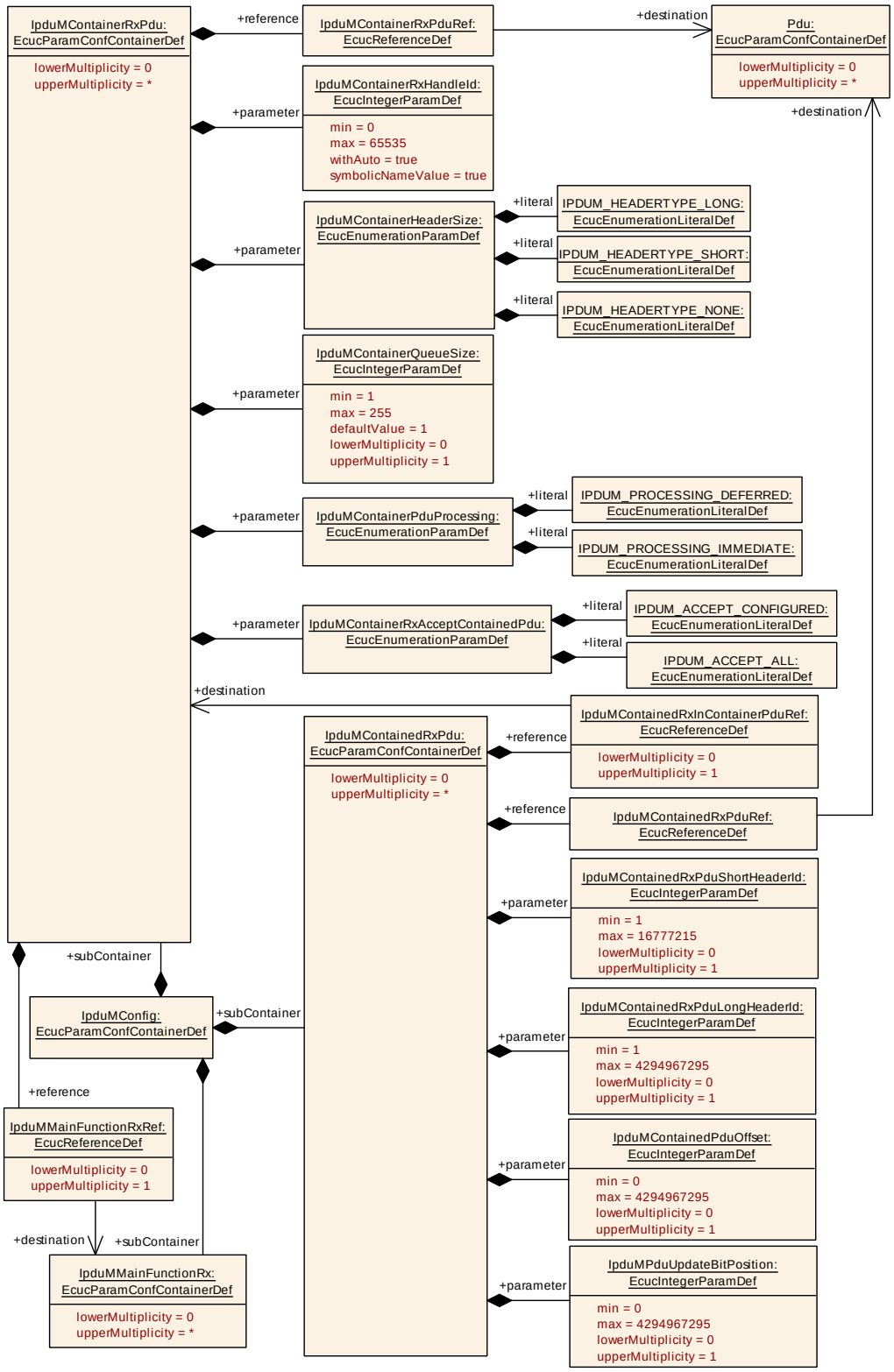


Figure 10.7: Configuration Overview RxContainer

10.2.19 IpduMContainerRxPdu

[ECUC_IpduM_00188] Definition of EcucParamConfContainerDef IpduMContainerRxPdu ⌈

Container Name	IpduMContainerRxPdu	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Configuration of a receiver ContainerPdu which may collect several ContainedPdus.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMContainerHeaderSize	1	[ECUC_IpduM_00183]
IpduMContainerPduProcessing	1	[ECUC_IpduM_00184]
IpduMContainerQueueSize	0..1	[ECUC_IpduM_00185]
IpduMContainerRxAcceptContainedPdu	1	[ECUC_IpduM_00186]
IpduMContainerRxHandleId	1	[ECUC_IpduM_00187]
IpduMContainerRxPduRef	1	[ECUC_IpduM_00189]
IpduMMainFunctionRxRef	0..1	[ECUC_IpduM_00212]

No Included Containers

⌋

[ECUC_IpduM_00183] Definition of EcucEnumerationParamDef IpduMContainerHeaderSize ⌈

Parameter Name	IpduMContainerHeaderSize	
Parent Container	IpduMContainerRxPdu, IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the layout of the header information (header id and length).	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IPDUM_HEADERTYPE_LONG	Header size is 64 bit: • Header Id 32 bit • Dlc 32 bit
	IPDUM_HEADERTYPE_NONE	Static Container Layout **Tags:** atp.Status=draft
	IPDUM_HEADERTYPE_SHORT	Header size is 32 bit: • Header Id 24 bit • Dlc 8 bit
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00184] Definition of EcucEnumerationParamDef IpduMContainerPduProcessing ⌈

Parameter Name	IpduMContainerPduProcessing	
Parent Container	IpduMContainerRxPdu	
Upstream mapping defined	–	
Description	Defines whether the handling of this ContainerPdu shall be done in the context of the caller (IMMEDIATE) or in the next call to IpduM_MainFunctionRx (DEFERRED).	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IPDUM_PROCESSING_DEFERRED	Deferred
	IPDUM_PROCESSING_IMMEDIATE	Immediate
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00185] Definition of EcucIntegerParamDef IpduMContainerQueueSize ⌈

Parameter Name	IpduMContainerQueueSize	
Parent Container	IpduMContainerRxPdu, IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines a local queue for handling of each ContainerPdu. Defined in number of instances of this ContainerPdu.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 255	
Default value	1	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00186] Definition of EcucEnumerationParamDef IpduMContainerRxAcceptContainedPdu ⌈

Parameter Name	IpduMContainerRxAcceptContainedPdu
Parent Container	IpduMContainerRxPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Defines for the received IpduMContainerRxPdu whether the list of referencing IpduMContainedRxPdus (via the reference IpduMContainedPduContainerRefRx) is a closed set.
Multiplicity	1
Type	EcucEnumerationParamDef

▽

△

Range	IPDUM_ACCEPT_ALL	The IpduMContainedRxPdus which are referencing this IpduMContainerRxPdu are expected inside this IpduMContainerRxPdu, but there may also occur other Pdus inside this Ipdu MContainerRxPdu as well. This also supports the case where no IpduMContainedRxPdu references the IpduMContainerRxPdu.
	IPDUM_ACCEPT_CONFIGURED	Only the IpduMContainedRxPdus which are referencing this IpduMContainerRxPdu are expected inside this IpduMContainerRxPdu.
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00187] Definition of EcucIntegerParamDef IpduMContainerRxHandleId ⌈

Parameter Name	IpduMContainerRxHandleId	
Parent Container	IpduMContainerRxPdu	
Upstream mapping defined	–	
Description	Handle Id used by the PduR for RxIndication.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00189] Definition of EcucReferenceDef IpduMContainerRxPduRef ⌈

Parameter Name	IpduMContainerRxPduRef	
Parent Container	IpduMContainerRxPdu	
Upstream mapping defined	–	
Description	Reference to the Pdu which represents the container and is used for reception.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00212] Definition of EcucReferenceDef IpduMMainFunctionRxRef ⌈

Parameter Name	IpduMMainFunctionRxRef
Parent Container	IpduMContainerRxPdu
Upstream mapping defined	–

▽

△

Description	Reference to the IpduM_MainFunctionRx instance this container PDU belongs to. Mandatory, if more than one IpduM_MainFunctionRx is defined.	
Multiplicity	0..1	
Type	Reference to IpduMMainFunctionRx	
Post-Build Multiplicity	-	Selectable
Post-Build Value	-	Selectable

⌋

10.2.20 IpduMContainedRxPdu

[ECUC_IpduM_00174] Definition of EcucParamConfContainerDef IpduMContainedRxPdu ⌈

Container Name	IpduMContainedRxPdu	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Configuration of a received contained Pdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMContainedPduOffset	0..1	[ECUC_IpduM_00206]
IpduMContainedRxPduLongHeaderId	0..1	[ECUC_IpduM_00203]
IpduMContainedRxPduShortHeaderId	0..1	[ECUC_IpduM_00202]
IpduMPduUpdateBitPosition	0..1	[ECUC_IpduM_00207]
IpduMContainedRxInContainerPduRef	0..1	[ECUC_IpduM_00173]
IpduMContainedRxPduRef	1	[ECUC_IpduM_00175]

No Included Containers

⌋

[ECUC_IpduM_00206] Definition of EcucIntegerParamDef IpduMContainedPduOffset

Status: DRAFT

⌈

Parameter Name	IpduMContainedPduOffset
Parent Container	IpduMContainedRxPdu, IpduMContainedTxPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]

▽

△

Description	Static offset (in bytes) of the ContainedPdu. **Tags:** atp.Status=draft Dependency: - only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_NONE. - only the ContainedPdu with the highest offset within a ContainerPdu may have variable length.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00203] Definition of EcucIntegerParamDef IpduMContainedRxPduLongHeaderId ⌈

Parameter Name	IpduMContainedRxPduLongHeaderId	
Parent Container	IpduMContainedRxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	LongHeader Id which is part of the ContainerPdu when this ContainedPdu is inside. Dependency: Only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_LONG	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 4294967295	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00202] Definition of EcucIntegerParamDef IpduMContainedRxPduShortHeaderId ⌈

Parameter Name	IpduMContainedRxPduShortHeaderId	
Parent Container	IpduMContainedRxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	ShortHeader Id which is part of the ContainerPdu when this ContainedPdu is inside. Dependency: Only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_SHORT	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 16777215	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00207] Definition of EcucIntegerParamDef IpduMPduUpdateBit Position

Status: DRAFT

⌈

Parameter Name	IpduMPduUpdateBitPosition	
Parent Container	IpduMContainedRxPdu, IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This value specifies where the PDU's Update-Bit is stored in the Container PDU (bit location of PDU's Update-Bit in the Container PDU). **Tags:** atp.Status=draft Dependency: - only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_NONE.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00173] Definition of EcucReferenceDef IpduMContainedRxInContainerPduRef ⌈

Parameter Name	IpduMContainedRxInContainerPduRef	
Parent Container	IpduMContainedRxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Optional reference to an IpduMContainerRxPdu this IpduMContainedRxPdu may be received in. If this IpduMContainedRxPdu shall be received in exactly one IpduMContainerRxPdu with IpduMContainerRxAcceptContainedPdu=IPDUM_ACCEPT_CONFIGURED then the IpduMContainedRxInContainerPduRef shall be defined. If this IpduMContainedRxPdu can be received in any IpduMContainerRxPdu with IpduMContainerRxAcceptContainedPdu=IPDUM_ACCEPT_ALL then the IpduMContainedRxInContainerPduRef shall NOT be defined.	
Multiplicity	0..1	
Type	Reference to IpduMContainerRxPdu	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00175] Definition of EcucReferenceDef IpduMContainedRxPduRef ⌈

Parameter Name	IpduMContainedRxPduRef
Parent Container	IpduMContainedRxPdu
Upstream mapping defined	–
Description	Reference to the Pdu which represents this ContainedPdu and is used for reception indication.
Multiplicity	1

▽

△

Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

┘

10.2.21 IpduMContainerTxPdu

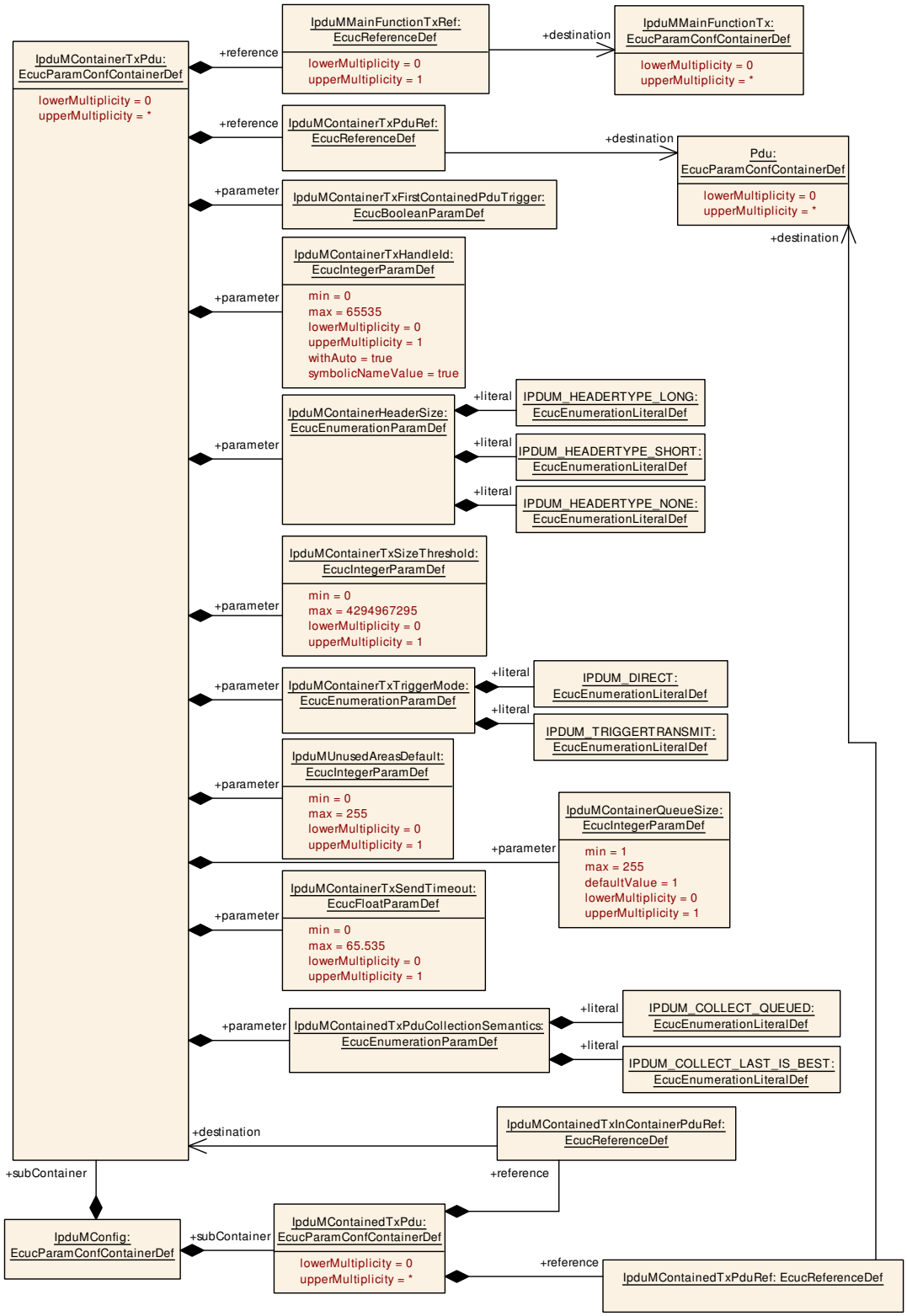


Figure 10.8: Configuration Overview TxContainer

[ECUC_IpduM_00192] Definition of EcucParamConfContainerDef IpduMContainerTxPdu ⌈

Container Name	IpduMContainerTxPdu	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Configuration of a transmitted container Pdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMContainedTxPduCollectionSemantics	1	[ECUC_IpduM_00198]
IpduMContainerHeaderSize	1	[ECUC_IpduM_00183]
IpduMContainerQueueSize	0..1	[ECUC_IpduM_00185]
IpduMContainerTxFirstContainedPduTrigger	1	[ECUC_IpduM_00199]
IpduMContainerTxHandleId	0..1	[ECUC_IpduM_00191]
IpduMContainerTxSendTimeout	0..1	[ECUC_IpduM_00194]
IpduMContainerTxSizeThreshold	0..1	[ECUC_IpduM_00195]
IpduMContainerTxTriggerMode	1	[ECUC_IpduM_00196]
IpduMUnusedAreasDefault	0..1	[ECUC_IpduM_00208]
IpduMContainerTxPduRef	1	[ECUC_IpduM_00193]
IpduMMainFunctionTxRef	0..1	[ECUC_IpduM_00214]

No Included Containers

⌋

[ECUC_IpduM_00198] Definition of EcucEnumerationParamDef IpduMContainedTxPduCollectionSemantics ⌈

Parameter Name	IpduMContainedTxPduCollectionSemantics	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines whether this IpduMContainerTxPdu shall be collector using a last-is-best or queued semantics.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IPDUM_COLLECT_LAST_IS_BEST	Collected IpduMContainedTxPdu data by this IpduMContainerTxPdu will be fetched via Trigger Transmit just before the transmission executes.
	IPDUM_COLLECT_QUEUED	Relevant IpduMContainedTxPdu data will instantly be stored to this IpduMContainerTxPdu in the context of the Transmit API.
Post-Build Value	Loadable	Selectable

⌋

For parameter table [ECUC_IpduM_00183] IpduMContainerHeaderSize, see definition below container IpduMContainerRxPdu.

For parameter table [ECUC_IpduM_00185] IpduMContainerQueueSize, see definition below container IpduMContainerRxPdu.

[ECUC_IpduM_00199] Definition of EcucBooleanParamDef IpduMContainerTx FirstContainedPduTrigger ⌈

Parameter Name	IpduMContainerTxFirstContainedPduTrigger	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines if the transmission of this IpduMContainerTxPdu shall be requested right after the first IpduMContainedTxPdu was put into it.	
Multiplicity	1	
Type	EcucBooleanParamDef	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00191] Definition of EcucIntegerParamDef IpduMContainerTx HandleId ⌈

Parameter Name	IpduMContainerTxHandleId	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	–	
Description	Handle Id used by the PduR for TxConfirmation and for TriggerTransmit of the ContainerPdu.	
Multiplicity	0..1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00194] Definition of EcucFloatParamDef IpduMContainerTxSend Timeout ⌈

Parameter Name	IpduMContainerTxSendTimeout	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	When this timeout expires the ContainerPdu is triggered for sending. The respective timer is started when the first Pdu is put into the ContainerPdu. Defined in seconds.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 65.535]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00195] Definition of EcucIntegerParamDef IpduMContainerTxSize Threshold ⌈

Parameter Name	IpduMContainerTxSizeThreshold	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the size threshold in bytes which, when exceeded, triggers the sending of the ContainerPdu although the maxium Pdu size (PduLength parameter of Pdu object) has not been reached yet. Dependency: only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_SHORT or IPDUM_HEADERTYPE_LONG	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00196] Definition of EcucEnumerationParamDef IpduMContainer TxTriggerMode ⌈

Parameter Name	IpduMContainerTxTriggerMode	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	–	
Description	Defines whether this ContainerPdu is fetched via trigger transmit.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IPDUM_DIRECT	The IpduM sends this ContainerPdu when this ContainerPdu is triggered.
	IPDUM_TRIGGERTRANSMIT	This ContainerPdu is stored in the IpduM and fetched via trigger transmit.
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00208] Definition of EcucIntegerParamDef IpduMUnusedAreas Default

Status: DRAFT

⌈

Parameter Name	IpduMUnusedAreasDefault
Parent Container	IpduMContainerTxPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	IpduM fills not updated areas of the Container PDU with this byte-pattern. **Tags:** atp.Status=draft Dependency: Only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_NONE / should be aligned to bus-specific padding value if available.
Multiplicity	0..1

▽

△

Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00193] Definition of EcucReferenceDef IpduMContainerTxPduRef

⌈

Parameter Name	IpduMContainerTxPduRef	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	–	
Description	Reference to the Pdu which represents the container and is used for transmission.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00214] Definition of EcucReferenceDef IpduMMainFunctionTxRef

⌈

Parameter Name	IpduMMainFunctionTxRef	
Parent Container	IpduMContainerTxPdu	
Upstream mapping defined	–	
Description	Reference to the IpduM_MainFunctionTx instance this container PDU belongs to. Mandatory, if more than one IpduM_MainFunctionTx is defined.	
Multiplicity	0..1	
Type	Reference to IpduMMainFunctionTx	
Post-Build Multiplicity	-	Selectable
Post-Build Value	-	Selectable

⌋

10.2.22 IpduMContainedTxPdu

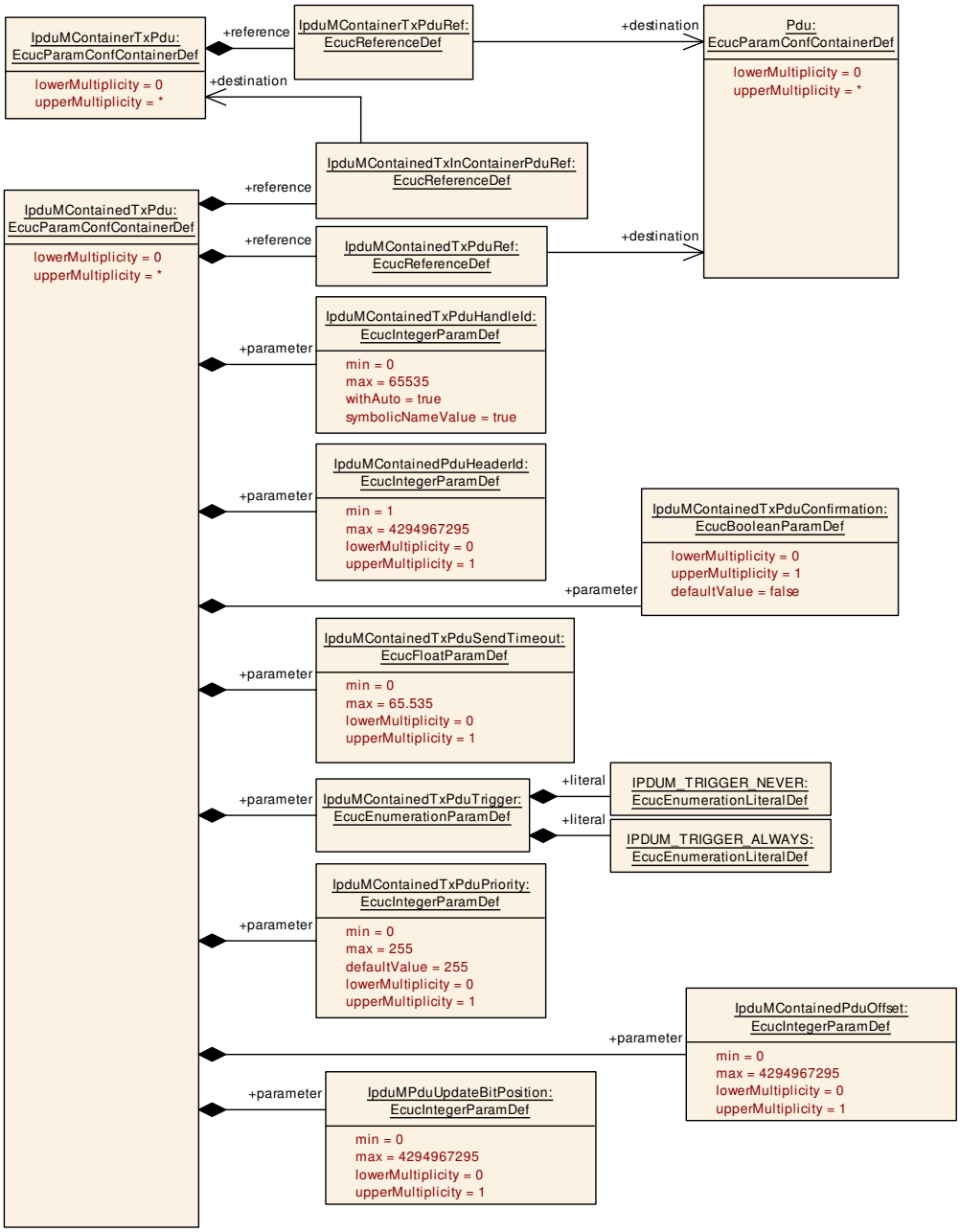


Figure 10.9: Configuration Overview Contained Tx Pdu

[ECUC_IpduM_00177] Definition of EcucParamConfContainerDef IpduMContainedTxPdu ⌈

Container Name	IpduMContainedTxPdu
Parent Container	IpduMConfig
Upstream mapping defined	–

▽

△

Description	Configuration of a sender ContainedPdu.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMContainedPduHeaderId	0..1	[ECUC_IpduM_00172]
IpduMContainedPduOffset	0..1	[ECUC_IpduM_00206]
IpduMContainedTxPduConfirmation	0..1	[ECUC_IpduM_00178]
IpduMContainedTxPduHandleId	1	[ECUC_IpduM_00179]
IpduMContainedTxPduPriority	0..1	[ECUC_IpduM_00210]
IpduMContainedTxPduSendTimeout	0..1	[ECUC_IpduM_00181]
IpduMContainedTxPduTrigger	1	[ECUC_IpduM_00182]
IpduMPduUpdateBitPosition	0..1	[ECUC_IpduM_00207]
IpduMContainedTxInContainerPduRef	1	[ECUC_IpduM_00176]
IpduMContainedTxPduRef	1	[ECUC_IpduM_00180]

No Included Containers

⌋

[ECUC_IpduM_00172] Definition of EcucIntegerParamDef IpduMContainedPduHeaderId ⌈

Parameter Name	IpduMContainedPduHeaderId	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Header Id which is part of the ContainerPdu when this ContainedPdu is inside. Dependency: only valid if IpduMContainerHeaderSize is set to IPDUM_HEADERTYPE_SHORT or IPDUM_HEADERTYPE_LONG.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 4294967295	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

For parameter table [ECUC_IpduM_00206] IpduMContainedPduOffset, see definition below container IpduMContainedRxPdu.

[ECUC_IpduM_00178] Definition of EcucBooleanParamDef IpduMContainedTxPduConfirmation ⌈

Parameter Name	IpduMContainedTxPduConfirmation
Parent Container	IpduMContainedTxPdu
Upstream mapping defined	–
Description	This Parameter determines whether for this contained I-PDU a TxConfirmation shall be provided. If set to TRUE a TxConfirmation is issued. It is not used when an I-PDU is requested using the trigger transmit API. If this Parameter is omitted, the default value shall be used.
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_IpduM_00179] Definition of EcucIntegerParamDef IpduMContainedTxPduHandleId ⌈

Parameter Name	IpduMContainedTxPduHandleId	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	–	
Description	Handle Id of the ContainedPdu.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00210] Definition of EcucIntegerParamDef IpduMContainedTxPduPriority ⌈

Parameter Name	IpduMContainedTxPduPriority	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines a priority of a ContainedTxPdu. 255 represents the lowest priority and 0 represent the highest priority. Dependency: The IpduMContainedTxPduPriority shall only be considered if IpduMContainedTxPduPriorityHandling is set to TRUE.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	255	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00181] Definition of EcucFloatParamDef IpduMContainedTxPdu SendTimeout ⌈

Parameter Name	IpduMContainedTxPduSendTimeout	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines a ContainedPdu specific sender timeout which adapts the ContainerPdu timer when this ContainedPdu is put inside the ContainerPdu: If the ContainerPdu timer is already running, it is reduced to this value, otherwise it is set to this value. Defined in seconds.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 65.535]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00182] Definition of EcucEnumerationParamDef IpduMContained TxPduTrigger ⌈

Parameter Name	IpduMContainedTxPduTrigger	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines whether this Pdu triggers the sending of the ContainerPdu.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IPDUM_TRIGGER_ALWAYS	This Pdu directly triggers the sending of the ContainerPdu.
	IPDUM_TRIGGER_NEVER	This Pdu does not triggers the sending of the ContainerPdu (other trigger criteria might still trigger sending of the ContainerPdu).
Post-Build Value	Loadable	Selectable

⌋

For parameter table [ECUC_IpduM_00207] IpduMPduUpdateBitPosition, see definition below container IpduMContainedRxPdu.

[ECUC_IpduM_00176] Definition of EcucReferenceDef IpduMContainedTxInContainerPduRef ⌈

Parameter Name	IpduMContainedTxInContainerPduRef	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to the container Pdu which this contained Pdu shall be collected in.	
Multiplicity	1	
Type	Reference to IpduMContainerTxPdu	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_IpduM_00180] Definition of EcucReferenceDef IpduMContainedTxPduRef ⌈

Parameter Name	IpduMContainedTxPduRef	
Parent Container	IpduMContainedTxPdu	
Upstream mapping defined	–	
Description	Reference to the Pdu which represents this ContainedPdu and is used for transmission.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.23 IpduMMainFunctionRx

[ECUC_IpduM_00211] Definition of EcucParamConfContainerDef IpduMMainFunctionRx ⌈

Container Name	IpduMMainFunctionRx	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Each element of this container defines one instance IpduM_MainFunctionRx, in case multi-core distribution feature is active.	
Multiplicity	0..*	
Post-Build Multiplicity	-	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMMainRxTimeBase	1	[ECUC_IpduM_00216]
IpduMMainRxPartitionRef	0..1	[ECUC_IpduM_00215]

No Included Containers

⌋

[ECUC_IpduM_00216] Definition of EcucFloatParamDef IpduMMainRxTimeBase ⌈

Parameter Name	IpduMMainRxTimeBase
Parent Container	IpduMMainFunctionRx
Upstream mapping defined	–

▽

△

Description	The period between successive calls to according instance of IpduM_MainFunctionRx in seconds. This parameter may be used by the IpduM generator to transform the values of the reception related timing configuration parameters of the IpduM module to internal implementation specific counter or tick values. The IpduM module's internal timing handling is implementation specific. The IpduM module (generator) may rely on the fact that IpduM_MainFunctionRx is scheduled according to the value configured here.	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	]0 .. INF[	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_IpduM_00215] Definition of EcucReferenceDef IpduMMainRxPartitionRef ⌈

Parameter Name	IpduMMainRxPartitionRef
Parent Container	IpduMMainFunctionRx
Upstream mapping defined	–
Description	Reference to EcucPartition, where the according IpduM_MainFunction instance is assigned to.
Multiplicity	0..1
Type	Reference to EcucPartition
Pre-Compile only	true

⌋

10.2.24 IpduMMainFunctionTx

[ECUC_IpduM_00213] Definition of EcucParamConfContainerDef IpduMMainFunctionTx ⌈

Container Name	IpduMMainFunctionTx	
Parent Container	IpduMConfig	
Upstream mapping defined	–	
Description	Each element of this container defines one instance IpduM_MainFunctionTx, in case multi-core distribution feature is active (mutual exclusive to ComTimeBase).	
Multiplicity	0..*	
Post-Build Multiplicity	-	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
IpduMMainTxTimeBase	1	[ECUC_IpduM_00218]
IpduMMainTxPartitionRef	0..1	[ECUC_IpduM_00217]

No Included Containers

⌋

[ECUC_IpduM_00218] Definition of EcucFloatParamDef IpduMMainTxTimeBase ⌈

Parameter Name	IpduMMainTxTimeBase
Parent Container	IpduMMainFunctionTx
Upstream mapping defined	–
Description	The period between successive calls to IpduM_MainFunctionTx in seconds. This parameter may be used by the IpduM generator to transform the values of the reception related timing configuration parameters of the IpduM module to internal implementation specific counter or tick values. The IpduM module's internal timing handling is implementation specific. The IpduM module (generator) may rely on the fact that IpduM_MainFunctionTx is scheduled according to the value configured here.
Multiplicity	1
Type	EcucFloatParamDef
Range	]0 .. INF[
Default value	–
Pre-Compile only	true

⌋

[ECUC_IpduM_00217] Definition of EcucReferenceDef IpduMMainTxPartitionRef ⌈

Parameter Name	IpduMMainTxPartitionRef
Parent Container	IpduMMainFunctionTx
Upstream mapping defined	–
Description	Reference to EcucPartition, where the according IpduM_MainFunction instance is assigned to.
Multiplicity	0..1
Type	Reference to EcucPartition
Pre-Compile only	true

⌋

10.3 Published Information

For details refer to [4] Chapter 10.3 *"Published Information"*.

10.4 Configuration Rules

10.4.1 Selector Field

[SWS_IpduM_CONSTR_00011] Selector field value number constraint ⌈The number of values used of the selector field, i.e. values used to distinguish between different I-PDU layouts, does not have to be the whole range of possible values.⌋

Example: The size of a selector field with 3 bits leads to 2^3 possible selector field values; it shall be allowed to use only an arbitrary subset of these values. The used subset needs no to be contiguous.

10.4.2 Placement of static and dynamic parts

[SWS_IpduM_CONSTR_00224] Identical bit placement of static and dynamic parts in multiplexed and de-multiplexed I-PDUs ⌈All static and dynamic parts shall be configured to allocate exactly the same bits in the multiplexed and the de-multiplexed I-PDUs.⌋

The above requirement assures that the IpduM does not need to shift or swap any bits or bytes but is able to construct the (de-) multiplexed I-PDUs by masking and copying operations. The IpduM handles all segments as opaque data and performs no endianness or sign conversion of the copied data.

10.4.3 Multiple PDU to Container Mapping

Container PDUs that have only I-PDUs assigned with `IPDUM_COLLECT_LAST_IS_BEST` semantic to can be realized buffer efficiently.

[SWS_IpduM_CONSTR_00260] Reject configuration with queued Container PDU and collection semantics IPDUM_COLLECT_LAST_IS_BEST ⌈IpduM shall reject configurations, in which Container PDU has both:

- a queue (`IpduMContainerQueueSize` is available) configured and
- `IpduMContainedTxPduCollectionSemantics` set to `IPDUM_COLLECT_LAST_IS_BEST`.

⌋

[SWS_IpduM_CONSTR_00230] Reject configuration with inconsistent MetaDataType ⌈IpduM shall reject configurations in which contained I-PDU supporting MetaData have a different MetaDataType from the MetaDataType of the Container PDU.⌋

The above requirement implies that multiple contained I-PDUs supporting MetaData assigned to the same Container PDU have the same MetaDataType.

[SWS_IpduM_CONSTR_00271] Reject configuration with IpduMContainerTxSendTimeout value and IpduMContainerTxFirstContainedPduTrigger set ⌈IpduM shall reject configurations in which the same Container PDU have `IpduMContainerTxFirstContainedPduTrigger` is set to TRUE and `IpduMContainerTxSendTimeout` configured.⌋

[SWS_IpduM_CONSTR_00238] Collection semantics for Container PDU with IPDUM_HEADERTYPE_NONE set

Status: DRAFT

⌈For a Container PDU with IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE, IpduMContainedTxPduCollectionSemantics shall be set to IPDUM_COLLECT_LAST_IS_BEST.⌋

[SWS_IpduM_CONSTR_00241] Contained I-PDU offset required for Container PDU with IPDUM_HEADERTYPE_NONE set

Status: DRAFT

⌈For a Container PDU with IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE, all contained I-PDUs shall have a configured IpduMContained-PduOffset.⌋

[SWS_IpduM_CONSTR_00242] Update bit position required for Container PDU with no header and no IpduMUnusedAreasDefault

Status: DRAFT

⌈For a Container PDU with IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE and IpduMUnusedAreasDefault not set, all contained I-PDUs shall have a configured IpduMPduUpdateBitPosition.⌋

[SWS_IpduM_CONSTR_00240] Update bit position only allowed for Container PDUs with IPDUM_HEADERTYPE_NONE set

Status: DRAFT

⌈Contained I-PDUs with a configured IpduMPduUpdateBitPosition shall only be assigned to Container PDUs with IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE.⌋

[SWS_IpduM_CONSTR_00246] Only last contained I-PDU of static layout Container PDU may have dynamic length

Status: DRAFT

⌈Only the last contained IPdu (according to IpduMContainedPduOffset) of a ContainerIPdu with static container layout (i.e. IpduMContainerHeaderSize set to IPDUM_HEADERTYPE_NONE) may be a dynamic length PDU (i.e, a PDU that at runtime may exhibit a length different from the one statically configured via Pdu.length of the respective Pdu). All other contained PDUs of a ContainerIPdu with static container layout have to be static length PDUs.⌋

This constraint is in line with similar constraints in Com ([SWS_Com_CONSTR_00754] and [SWS_Com_CONSTR_00755]) and in the FrIf ([SWS_FrIf_05244]).

[SWS_IpduM_CONSTR_00245] Unique update bit position for each contained I-PDU

Status: DRAFT

⌈All IpduMPduUpdateBitPositions shall be configured to their own not otherwise occupied bit position.⌋

10.4.4 Priority

[SWS_IpduM_CONSTR_00263] Priority handling required for differing priorities of contained I-PDUs in the same Container PDU ⌈The IpduMContainedTxPduPriorityHandling shall be set to TRUE, if at least one IpduMContainerTxPdu is referenced by at least two IpduMContainedTxPdu via IpduMContainedTxInContainerPduRef and the configured IpduMContainedTxPduPriority of those IpduMContainedTxPdu differ from each other.⌋

Note: If all IpduMContainedTxPdu which refer to the same IpduMContainerTxPdu have the same IpduMContainedTxPduPriority set and this applies to all configured IpduMContainedTxPdus, then a prioritization cannot be performed. In this case, this is even true, if the configured IpduMContainedTxPduPriority differ per IpduMContainerTxPdu.

[SWS_IpduM_CONSTR_00248] Priority required for contained I-PDUs of Container PDU with IPDUM_COLLECT_LAST_IS_BEST when priority handling is enabled ⌈If IpduMContainedTxPduPriorityHandling is set to TRUE, all contained I-Pdus of a Container PDU with Collection Semantic IPDUM_COLLECT_LAST_IS_BEST shall have an IpduMContainedTxPduPriority.⌋

A Not applicable requirements

[SWS_IpduM_NA_00001]

Upstream requirements: SRS_BSW_00004, SRS_BSW_00168, SRS_BSW_00170, SRS_BSW_00171, SRS_BSW_00336, SRS_BSW_00339, SRS_BSW_00375, SRS_BSW_00384, SRS_BSW_00386, SRS_BSW_00383, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00395, SRS_BSW_00403, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00419, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00427, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00437, SRS_BSW_00461, SRS_BSW_00469, SRS_BSW_00471, SRS_BSW_00472, SRS_BSW_00490, SRS_BSW_00491

⌈These requirements are not applicable to this specification.⌋

B Change History of AUTOSAR Traceable Items

Please note that the lists in this chapter also include traceable items that have been removed from the specification in a later version. These items do not appear as hyperlinks in the document.

B.1 Traceable Item History of this Document According to AUTOSAR Release R26-11

B.1.1 Added Specification Items in R26-11

[ECUC_IpduM_00221] [ECUC_IpduM_00222] [SWS_IpduM_00282] [SWS_IpduM_00283] [SWS_IpduM_00284] [SWS_IpduM_00285] [SWS_IpduM_00286] [SWS_IpduM_00287]

B.1.2 Changed Specification Items in R26-11

[ECUC_IpduM_00047] [ECUC_IpduM_00052] [ECUC_IpduM_00112] [ECUC_IpduM_00115] [ECUC_IpduM_00130] [ECUC_IpduM_00160] [ECUC_IpduM_00177] [ECUC_IpduM_00192] [ECUC_IpduM_00198] [ECUC_IpduM_00204] [SWS_IpduM_00105] [SWS_IpduM_00179] [SWS_IpduM_00182] [SWS_IpduM_00183] [SWS_IpduM_00221] [SWS_IpduM_00222] [SWS_IpduM_00231] [SWS_IpduM_00249] [SWS_IpduM_00258] [SWS_IpduM_00259] [SWS_IpduM_00264] [SWS_IpduM_91003] [SWS_IpduM_91004]

B.1.3 Deleted Specification Items in R26-11

[ECUC_IpduM_00054] [ECUC_IpduM_00134] [ECUC_IpduM_00141] [ECUC_IpduM_00142] [SWS_IpduM_00011] [SWS_IpduM_00037] [SWS_IpduM_00140] [SWS_IpduM_00219] [SWS_IpduM_00224] [SWS_IpduM_00230] [SWS_IpduM_00238] [SWS_IpduM_00240] [SWS_IpduM_00241] [SWS_IpduM_00242] [SWS_IpduM_00245] [SWS_IpduM_00246] [SWS_IpduM_00248] [SWS_IpduM_00260]

B.1.4 Added Constraints in R26-11

[SWS_IpduM_CONSTR_00011] [SWS_IpduM_CONSTR_00224] [SWS_IpduM_CONSTR_00230] [SWS_IpduM_CONSTR_00238] [SWS_IpduM_CONSTR_00240] [SWS_IpduM_CONSTR_00241] [SWS_IpduM_CONSTR_00242] [SWS_IpduM_CONSTR_00245] [SWS_IpduM_CONSTR_00246] [SWS_IpduM_CONSTR_00248] [SWS_IpduM_CONSTR_00260] [SWS_IpduM_CONSTR_00281]

B.1.5 Changed Constraints in R26-11

none

B.1.6 Deleted Constraints in R26-11

none

B.2 Traceable Item History of this Document According to AUTOSAR Release R25-11

B.2.1 Added Specification Items in R25-11

[ECUC_IpduM_00220] [SWS_IpduM_00264] [SWS_IpduM_00265] [SWS_IpduM_00272] [SWS_IpduM_00273] [SWS_IpduM_00274] [SWS_IpduM_00276] [SWS_IpduM_00277] [SWS_IpduM_00278] [SWS_IpduM_00279]

B.2.2 Changed Specification Items in R25-11

[ECUC_IpduM_00130] [SWS_IpduM_00184] [SWS_IpduM_00185] [SWS_IpduM_00201] [SWS_IpduM_00259]

B.2.3 Deleted Specification Items in R25-11

none

B.2.4 Added Constraints in R25-11

[SWS_IpduM_CONSTR_00271]

B.2.5 Changed Constraints in R25-11

none

B.2.6 Deleted Constraints in R25-11

none

B.3 Traceable Item History of this Document According to AUTOSAR Release R24-11

B.3.1 Added Specification Items in R24-11

none

B.3.2 Changed Specification Items in R24-11

none

B.3.3 Deleted Specification Items in R24-11

none

B.3.4 Added Constraints in R24-11

none

B.3.5 Changed Constraints in R24-11

none

B.3.6 Deleted Constraints in R24-11

none

B.4 Traceable Item History of this Document According to AUTOSAR Release R23-11

B.4.1 Added Specification Items in R23-11

none

B.4.2 Changed Specification Items in R23-11

[SWS_IpduM_00232] [SWS_IpduM_00235] [SWS_IpduM_00241] [SWS_IpduM_00254]

B.4.3 Deleted Specification Items in R23-11

none

B.4.4 Added Constraints in R23-11

none

B.4.5 Changed Constraints in R23-11

none

B.4.6 Deleted Constraints in R23-11

none