Document Title	Specification of Communication
Document Category	Software specification (SWS)
Document UID	15
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	• Support DMA transfer approach • Removed ComIPduGroupGroupRef • Removed GetVersionInfo API • Minor corrections / clarifications / editorial changes
2025-11-27	R25-11	AUTOSAR Release Management	• Merged LdCom functionality as Large Data Path • Added timeout reporting to DEM • Minor corrections / clarifications / editorial changes
2024-11-27	R24-11	AUTOSAR Release Management	• Security event handling • Minor corrections / clarifications / editorial changes
2023-11-23	R23-11	AUTOSAR Release Management	• Minor corrections / clarifications / editorial changes
2022-11-24	R22-11	AUTOSAR Release Management	• Removed obsolete elements • Minor corrections / clarifications / editorial changes
2021-11-25	R21-11	AUTOSAR Release Management	• Removed I-PDU counter and I-PDU replication • Added support for independent development of CP Software Clusters • minor corrections / clarifications / editorial changes
2020-11-30	R20-11	AUTOSAR Release Management	• BSW Multicore Distribution (CONC_643) • minor corrections / clarifications / editorial changes

▽

△

2019-11-28	R19-11	AUTOSAR Release Management	• MetaData handling above RTE (CONC_650) • BSW Multicore Distribution as draft (CONC_643) • minor corrections / clarifications / editorial changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2017-12-08	4.3.1	AUTOSAR Release Management	• reverted I-PDU group control APIs to AUTOSAR Release 3 style • 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 • revised Signal Based Gateway • 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 support for Sender/ Receiver Serialization • updated to support CAN FD • minor corrections
2014-03-31	4.1.3	AUTOSAR Release Management	• Several correction, clarifications and editioral changes • No major functional changes or improvements

▽

△

2013-10-31	4.1.2	AUTOSAR Release Management	• Removed TP-restriction for dynamic length I-PDUs • Added parameter to globally enable or disable the cancellation support • Many bug fixes, clarifications and harmonization with the other COM Stack modules • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Reworked according to the new SWS BSW General • added support for postbuild loadable configuration • harmonization of COM-Stack APIs (TP-APIs and callouts) • many corrections and clarifications
2011-12-22	4.0.3	AUTOSAR Administration	• Several minor changes and bugfixes • Clarification and minor extensions of transmission mode usage and activation • Clarification and extension of transfer properties • Added retry mechanism for failed transmission requests
2010-09-30	3.1.5	AUTOSAR Administration	• corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2010-02-02	3.1.4	AUTOSAR Administration	• Added support for large data types • Added support for communication protection • Revised mode management interface • Many clarifications and corrections based on SWS improvement, validation and conformance test activities • Legal disclaimer revised
2008-08-13	3.1.1	AUTOSAR Administration	• Legal disclaimer revised
2008-02-01	3.0.2	AUTOSAR Administration	• Removed Figure 19, which was included twice

▽

△

2007-12-21	3.0.1	AUTOSAR Administration	• Removal of Internal Communication feature • Revision of AUTOSAR COM MODULE configuration • Revision of Signal Gateway configuration • Enhanced maximal I-PDU length for FlexRay • Harmonization of callback-interface to RTE • Clarifications and bug fixes • Document meta information extended • Small layout adaptations made
2007-01-24	2.1.15	AUTOSAR Administration	• Clarifications of requirements throughout the whole document. Chapter 11 contains a detailed list of the made modifications. No new major features were added since release 2.0. • "Advice for users" revised • "Revision Information" added • Legal disclaimer revised
2006-05-16	2.0	AUTOSAR Administration	• Document structure adapted to common Release 2.0 SWS Template. • Integration of signal gateway • Major updates in configuration, error handling, filtering, transmission mode switches, callouts, updatebits, deadline monitoring and initialization
2005-05-31	1.0	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 12
2 Acronyms, Abbreviations, and Definitions 13
2.1 Acronyms and Abbreviations . . . 13
2.2 Definitions . . . 14
3 Related documentation 15
3.1 Input documents & related standards and norms . . . 15
3.2 Related specification . . . 16
4 Constraints and assumptions 17
4.1 Limitations . . . 17
4.2 Applicability to car domains . . . 17
5 Dependencies to other modules 18
5.1 PDU Router . . . 18
5.2 Com Users . . . 19
5.2.1 Runtime Environment (RTE) . . . 19
5.2.2 SwCluC . . . 19
5.3 Diagnostic Event Manager (DEM) . . . 19
5.4 Default Error Tracer (DET) . . . 19
6 Requirements Tracing 20
7 Functional specification 22
7.1 General Functionality . . . 22
7.1.1 AUTOSAR COM basis . . . 22
7.1.2 Signal Values . . . 23
7.1.3 Endianness Conversion and Sign Extension . . . 23
7.1.4 Filtering . . . 25
7.1.5 Signal Gateway . . . 28
7.2 Normal Operation . . . 28
7.2.1 Start-Up Behavior . . . 28
7.2.1.1 Preconditions . . . 29
7.2.1.2 Initialization . . . 29
7.2.1.3 Initialization of Not Used Areas of an I-PDU . . . 29
7.2.1.4 Initialization of Signals and Update-Bits . . . 29
7.2.1.5 Initialization of I-PDU Groups . . . 30
7.2.2 De-Initialization . . . 30
7.2.3 Communication Modes . . . 30
7.2.3.1 Transfer Properties and I-PDU Transmission Mode . . . 30
7.2.3.2 Selection of the Transmission Mode for one specific I-PDU . . . 33
7.2.3.3 Signal Flow and Transmission Mode Selection . . . 35
7.2.3.4 Replication of Signal Transmission Requests . . . 37

7.2.3.5 Retry Failed Transmission Requests . . . 38
7.2.4 Signal Invalidation . . . 39
7.2.4.1 Transmission of an Invalidated Signal . . . 39
7.2.4.2 Reception of an Invalidated Signal . . . 39
7.2.5 Handling of I-PDUs . . . 40
7.2.5.1 I-PDU Group Definitions . . . 40
7.2.5.2 Starting of I-PDU Groups . . . 41
7.2.5.3 Stopping of I-PDU Groups . . . 43
7.2.5.4 Signal Indication (Unpacking of I-PDUs) . . . 45
7.2.5.5 Minimum Delay Timer (MDT) . . . 46
7.2.6 Deadline Monitoring . . . 48
7.2.6.1 Reception Deadline Monitoring . . . 48
7.2.6.2 Transmission Deadline Monitoring . . . 50
7.2.7 Notification handling . . . 53
7.3 Signal Groups - Complex Data Types . . . 53
7.3.1 Initialization . . . 54
7.3.2 Transmission . . . 54
7.3.3 Reception . . . 54
7.3.4 Notifications . . . 55
7.3.5 Attributes of a Signal Group . . . 55
7.3.6 UINT8-array based access to signal groups . . . 55
7.3.6.1 Basic functionality . . . 57
7.3.6.2 Initialization . . . 57
7.3.6.3 Transmission . . . 58
7.3.6.4 Reception . . . 58
7.4 Large Data Types . . . 58
7.4.1 Transmission of Large Signals / I-PDUs . . . 59
7.4.2 Reception of Large Signals / I-PDUs . . . 59
7.5 Large Data Path Support . . . 60
7.5.1 Overall . . . 61
7.5.2 Transmission . . . 62
7.5.2.1 IF . . . 63
7.5.2.2 TP . . . 63
7.5.3 Reception . . . 64
7.5.3.1 IF . . . 64
7.5.3.2 TP . . . 64
7.6 Asynchronous Copy of I-PDU Data . . . 65
7.7 Dynamic Length Signals . . . 66
7.7.1 Transmission of Dynamic Length Signals / I-PDUs . . . 66
7.7.2 Reception of Dynamic Length Signals / I-PDUs . . . 67
7.8 MetaData Handling . . . 67
7.9 Interface between AUTOSAR COM Module and the PDU Router . . . 68
7.10 Update-Bits . . . 69

7.10.1 Sender Side . . . 69
7.10.2 Receiver Side . . . 70
7.11 Signal Gateway . . . 71
7.11.1 Dealing with Signals . . . 72
7.11.2 Dealing with Group Signals . . . 72
7.11.3 Routing of Out-Timed Signals and Signal Groups . . . 73
7.11.4 Handling of Update-Bits . . . 73
7.11.5 Decoupling Signal Gateway . . . 74
7.12 Error Classification . . . 74
7.12.1 Development Errors . . . 75
7.12.2 Runtime Errors . . . 75
7.12.3 Production Errors . . . 75
7.13 AUTOSAR COM Module's Interaction Model . . . 76
7.14 Multicore Distribution . . . 80
7.15 Software Cluster Connection Layer (SwCluC) . . . 80
7.16 Security Events . . . 82
8 API specification 84
8.1 Imported types . . . 84
8.2 Type definitions . . . 84
8.2.1 Com_StatusType . . . 84
8.2.2 Com_SignalIdType . . . 85
8.2.3 Com_SignalGroupIdType . . . 85
8.2.4 Com_IpduGroupIdType . . . 85
8.2.5 Com_ConfigType . . . 86
8.3 Function definitions . . . 86
8.3.1 Return Codes . . . 86
8.3.2 Start-Up and Control Services . . . 87
8.3.2.1 Com_Init . . . 87
8.3.2.2 Com_DeInit . . . 88
8.3.2.3 Com_IpduGroupStart . . . 88
8.3.2.4 Com_IpduGroupStop . . . 89
8.3.2.5 Com_EnableReceptionDM . . . 90
8.3.2.6 Com_DisableReceptionDM . . . 90
8.3.2.7 Com_GetStatus . . . 91
8.3.3 Communication Services . . . 91
8.3.3.1 Com_SendSignal . . . 91
8.3.3.2 Com_SendDynSignal . . . 92
8.3.3.3 Com_ReceiveSignal . . . 93
8.3.3.4 Com_ReceiveDynSignal . . . 94
8.3.3.5 Com_SendSignalGroup . . . 95
8.3.3.6 Com_ReceiveSignalGroup . . . 95
8.3.3.7 Com_SendSignalGroupArray . . . 96
8.3.3.8 Com_ReceiveSignalGroupArray . . . 97

8.3.3.9 Com_ReceiveSignalWithMetaData . . . 97
8.3.3.10 Com_ReceiveDynSignalWithMetaData . . . 98
8.3.3.11 Com_ReceiveSignalGroupWithMetaData . . . 99
8.3.3.12 Com_ReceiveSignalGroupArrayWithMetaData . . . 99
8.3.3.13 Com_SendSignalWithMetaData . . . 100
8.3.3.14 Com_SendDynSignalWithMetaData . . . 101
8.3.3.15 Com_SendSignalGroupWithMetaData . . . 101
8.3.3.16 Com_SendSignalGroupArrayWithMetaData . . . 102
8.3.3.17 Com_InvalidateSignal . . . 103
8.3.3.18 Com_InvalidateSignalGroup . . . 103
8.3.3.19 Com_TriggerIPDUSend . . . 104
8.3.3.20 Com_TriggerIPDUSendWithMetaData . . . 105
8.3.3.21 Com_SwitchIpduTxMode . . . 106
8.3.3.22 Com_Transmit . . . 107
8.3.3.23 Com_ReleaseRxBuffer . . . 108
8.3.3.24 Com_TriggerTransmit . . . 108
8.3.3.25 Com_RxIndication . . . 110
8.3.3.26 Com_TpRxIndication . . . 110
8.3.3.27 Com_TxConfirmation . . . 111
8.3.3.28 Com_TpTxConfirmation . . . 112
8.3.3.29 Com_StartOfReception . . . 112
8.3.3.30 Com_CopyRxData . . . 114
8.3.3.31 Com_CopyTxData . . . 114
8.4 Scheduled functions . . . 116
8.4.1 Com_MainFunctionRx . . . 116
8.4.2 Com_MainFunctionTx . . . 116
8.4.3 Com_MainFunctionRouteSignals . . . 117
8.5 Expected interfaces . . . 117
8.5.1 Mandatory interfaces . . . 118
8.5.2 Optional interfaces . . . 118
8.5.3 Configurable interfaces . . . 119
9 Sequence diagrams 131
9.1 Interface between the AUTOSAR COM Module and the PDU Router . . . 131
9.2 Confirmation Handling between the PDU Router, the AUTOSAR COM Module and the RTE . . . 132
9.3 Indication Handling between the PDU Router, the AUTOSAR COM Module and the RTE . . . 133
9.4 Large Data Path Transmission . . . 134
9.4.1 TP API . . . 134
9.4.2 IF API . . . 135
9.4.3 TriggerTransmit . . . 136
9.5 Large Data Path Reception . . . 137
9.5.1 TP API . . . 137

9.5.2 IF API . . . 138
10 Configuration specification 139
10.1 How to read this chapter . . . 139
10.2 Containers and configuration parameters . . . 139
10.2.1 Configuration of the AUTOSAR COM Module . . . 139
10.2.2 Com . . . 141
10.2.3 ComGeneral . . . 142
10.2.4 ComSecurityEventRefs . . . 146
10.2.5 ComConfig . . . 147
10.2.6 ComMainFunctionRx . . . 149
10.2.7 ComMainFunctionTx . . . 150
10.2.8 ComMainFunctionRouteSignals . . . 151
10.2.9 ComFilter . . . 153
10.2.10 ComIPdu . . . 157
10.2.11 ComMetaDataDefaultItem . . . 163
10.2.12 ComTxIPdu . . . 164
10.2.13 ComIPduGroup . . . 167
10.2.14 ComSignal . . . 169
10.2.15 ComDemEventParameterRefs . . . 172
10.2.16 ComSignalGroup . . . 175
10.2.17 ComGroupSignal . . . 177
10.2.18 ComTxMode . . . 184
10.2.19 ComTxModeTrue . . . 187
10.2.20 ComTxModeFalse . . . 187
10.2.21 ComGwMapping . . . 188
10.2.22 ComGwSource . . . 189
10.2.23 ComGwSourceDescription . . . 190
10.2.24 ComGwDestination . . . 191
10.2.25 ComGwDestinationDescription . . . 191
10.2.26 ComGwSignal . . . 193
10.2.27 ComUserModule . . . 194
10.2.28 ComUserUriDefSet . . . 195
10.2.29 ComUserModuleCnf . . . 195
10.2.30 ComUserIPdu . . . 196
10.2.31 ComUserSignal . . . 198
10.2.32 ComUserSignalGroup . . . 199
10.2.33 ComUserCallback . . . 199
10.3 Configuration Rules . . . 201
10.3.1 Signal Configuration . . . 201
10.3.2 Signal Group Configuration . . . 203
10.3.3 Transmission Mode Configuration . . . 203
10.3.4 Signal Gateway Configuration . . . 203
10.3.5 Filter Configuration . . . 204

10.3.6 Large Data Path Configuration 204
10.3.7 Asynchronous Copy Support 207
10.3.8 Dynamic Length I-PDU Configuration 207
10.3.9 Meta Data Configuration 207
10.3.10 I-PDU Group Configuration 207
10.3.11 Configuration of Com Users 208
10.3.11.1 ECUC Partition Specific Com User 209
10.3.11.2 ECUC Partition Unspecific Com User 210
10.3.11.3 Relation to SwCluC Com Proxy 210
10.4 Published Information 211
A Not applicable requirements 212
B Appendix 213
B.1 Use Case 1 213
B.2 Use Case 2 214
B.3 Use Case 3 215
B.4 Use Case 4 216
B.5 Use Case 5 217
B.6 Use Case 6 219
B.7 Use Case 7 219
C Change history of AUTOSAR traceable items 222
C.1 Traceable item history of this document according to AUTOSAR Release R24-11 222
C.1.1 Added Specification Items in R24-11 222
C.1.2 Changed Specification Items in R24-11 222
C.1.3 Deleted Specification Items in R24-11 223
C.2 Traceable item history of this document according to AUTOSAR Release R25-11 223
C.2.1 Added Specification Items in R25-11 223
C.2.2 Changed Specification Items in R25-11 224
C.2.3 Deleted Specification Items in R25-11 225
C.2.4 Added Constraints in R25-11 226
C.2.5 Changed Constraints in R25-11 229
C.2.6 Deleted Constraints in R25-11 229
C.3 Traceable item history of this document according to AUTOSAR Release R26-11 229
C.3.1 Added Specification Items in R26-11 229
C.3.2 Changed Specification Items in R26-11 230
C.3.3 Deleted Specification Items in R26-11 230
C.3.4 Added Constraints in R26-11 230
C.3.5 Changed Constraints in R26-11 231
C.3.6 Deleted Constraints in R26-11 231

1 Introduction and functional overview

This specification is the AUTOSAR COM module Software Specification. That means that the functionality and the API of the AUTOSAR COM module are described in this document.

Within the AUTOSAR Layered Architecture the AUTOSAR COM module is placed between its users (e.g. RTE, SwCluC) and the PDU Router, see [1].

The AUTOSAR COM module is derived from [2]. For details, see Chapter 7.1.1. The AUTOSAR COM module provides signal gateway functionality. For details, see Chapter 7.1.5.

Main Features:

- Provision of signal oriented data interface for the users (e.g. RTE, SwCluC)
- Packing of AUTOSAR signals to I-PDUs to be transmitted
- Unpacking of received I-PDUs and provision of received signals to the users (e.g. RTE, SwCluC)
- Routing of signals from received I-PDUs into I-PDUs to become transmitted
- Routing of signal groups from received I-PDUs into I-PDUs to become transmitted
- Communication transmission control (start / stop of I-PDU groups)
- Replications of send requests
- Guarantee of minimum distances between transmit I-PDUs
- Monitoring of receive signals (signals timeout)
- Filter mechanisms for incoming signals
- Different notification mechanisms
- Provision of init values and update indications
- Byte order conversion
- Sign extension
- Support of two different transmission modes per I-PDU
- Signal based gateway
- Support of large and dynamic length data types

2 Acronyms, Abbreviations, and Definitions

2.1 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the COM module that are not included in the [3, AUTOSAR glossary].

Acronym	Description
AUTOSAR COM	The AUTOSAR COM module is derived from [2]. For details, see Chapter 7.1.1.
DM	Deadline Monitoring, for details see Chapter 7.2.6
I-PDU	Interaction Layer Protocol Data Unit. An I-PDU carries signals, it is defined in [2].
L-PDU	Data Link Layer Protocol Data Unit. In AUTOSAR, the Data Link Layer is equivalent to the Communication Hardware Abstraction and Microcontroller Abstraction Layer.
MDT	A detailed description of the Minimum Delay Timer (MDT) can be found in [2]. See also Chapter 7.2.5.5.
PDU Router	The PDU Router is a module transferring I-PDUs from one module to another module. The PDU Router can be utilized for gateway operations and for internal routing purposes.
SDU	Service Data Unit. For a description see [1] and [3] Chapter 4.278 *"Service Data Unit"*.
TM	Transmission Mode
TMC	Transmission Mode Condition, see Chapter 7.2.3.2
TMS	Transmission Mode Selector, see Chapter 7.2.3.2

Table 2.1: Acronyms used in the scope of this Document

2.2 Definitions

Term	Description
Confirmation	With a Confirmation, the PDU Router reports that a request by the AUTOSAR COM module has been completed successfully. It is a reaction to a request of COM. E.g. when a PDU has been successfully transmitted.
Data Invalid Value	Value sent by the AUTOSAR COM module to indicate that the sender side AUTOSAR Software Component is not able to provide a valid value.
Dynamic Length Signal	A dynamic length signal is a signal which length can vary at run-time.
Dynamic Length I-PDU	A dynamic length I-PDU is an I-PDU containing a dynamic length signal. It length varies depending on the length of the included dynamic length signal.
Group signal	A group signal is a signal that is contained in a signal group.
Indication	An Indication is asynchronous information from PDU Router to COM, e.g. to acknowledge that something has been received.
Init Value	I-PDUs and signals are set to the Init Value by the AUTOSAR COM module after start-up. This value is used until it is overwritten.
I-PDU group	An I-PDU Group is an arbitrary collection of I-PDUs of the same direction (i.e. send or receive) in the COM module.
Inter-ECU -communication	Communication between two or more ECUs for example via a CAN network.
Intra-ECU -communication	Communication between Software components that reside on the same ECU.
Large Data Path	Large Data Path (see `ComIPduLargeData`) provides an alternative Interaction Layer Mechanism. By focusing on spontaneous, non-cyclic communication without serializing, filtering and conversion in the COM module an efficient implementation can be achieved for selected I-PDUs. IF- and TP-based communication is also supported.
Large Signal	A large signal is a signal that is too large to fit into a single L-PDU of the underlying communication protocol.
Large I-PDU	Large I-PDUs are I-Signal-I-PDUs which do not fit into a single L-PDU of the underlying communication protocol. Large I-PDUs will be handled like other I-PDUs by COM, but are transmitted / received via the TP API. Please note that large I-PDUs can also be handled in a more efficient way by COM (see `ComIPduLargeData`) if they contain just one signal, or if the signals can be (de-)serialized by the RTE. But if the signals have to be treated separately or have to be routed, Large Data Path is not available.
Message	[2] uses always the synonym *message*. In AUTOSAR, *message* is replaced by *signal* but with the same meaning.
Metadata	For some I-PDUs, e.g. J1939 I-PDUs, the payload is extended with additional metadata containing for example the CAN-ID.
Notification	Information by the AUTOSAR COM module to its users (e.g. RTE, SwCluC), e.g. when new data is available, an error occurred.
Signal	A signal in the AUTOSAR COM module's context is equal to a message in [2].
Signal group	In AUTOSAR, so called complex data types are used. Inside a complex data type, there are one or more data elements (primitive data types), like in a C struct. The data consistency of such complex data types must be ensured. The user (e.g. RTE or SwCluC) decomposes the complex data type in single signals and sends them to the AUTOSAR COM module. As these signals altogether need to be treated consistently, they are called *signal group*.
Update-bit	A mechanism supported by the AUTOSAR COM module with that the receiver of a signal / signal group could identify whether the sender has updated the data in this signal / signal group before sending. See Chapter 7.10.

Table 2.2: Terms used in the scope of this Document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture

[2] ISO 17356-4: Road vehicles – Open interface for embedded automotive applications – Part 4: OSEK/VDX Communication (COM)

[3] Glossary
AUTOSAR_FO_TR_Glossary

[4] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[5] Specification of PDU Router
AUTOSAR_CP_SWS_PDURouter

[6] Specification of RTE Software
AUTOSAR_CP_SWS_RTE

[7] Specification of Software Cluster Connection module
AUTOSAR_CP_SWS_SoftwareClusterConnection

[8] Specification of Diagnostic Event Manager
AUTOSAR_CP_SWS_DiagnosticEventManager

[9] Specification of Default Error Tracer
AUTOSAR_CP_SWS_DefaultErrorTracer

[10] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[11] System Template
AUTOSAR_CP_TPS_SystemTemplate

[12] ISO 11898-1:2015 – Road vehicles – Controller area network (CAN)

[13] Specification of CAN Transport Layer
AUTOSAR_CP_SWS_CANTransportLayer

[14] Specification of FlexRay ISO Transport Layer
AUTOSAR_CP_SWS_FlexRayISOTransportLayer

[15] Specification of ECU Configuration
AUTOSAR_CP_TPS_ECUConfiguration

[16] Specification of Standard Types
AUTOSAR_CP_SWS_StandardTypes

[17] Software Component Template
AUTOSAR_CP_TPS_SoftwareComponentTemplate

[18] Generic Structure Template
AUTOSAR_FO_TPS_GenericStructureTemplate

3.2 Related specification

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

Thus, the specification SWS BSW General shall be considered as additional and required specification for COM.

4 Constraints and assumptions

4.1 Limitations

The AUTOSAR COM module is based on [2]. Nevertheless not all features of [2] are included and some features are different. See [SWS_Com_00013] for a list of not included features.

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

This chapter lists all the features from other modules that are used by the AUTOSAR COM module and functionalities that are provided by the AUTOSAR COM module to other modules. For the placement of the AUTOSAR COM module in the communication stack, see Figure 5.1.

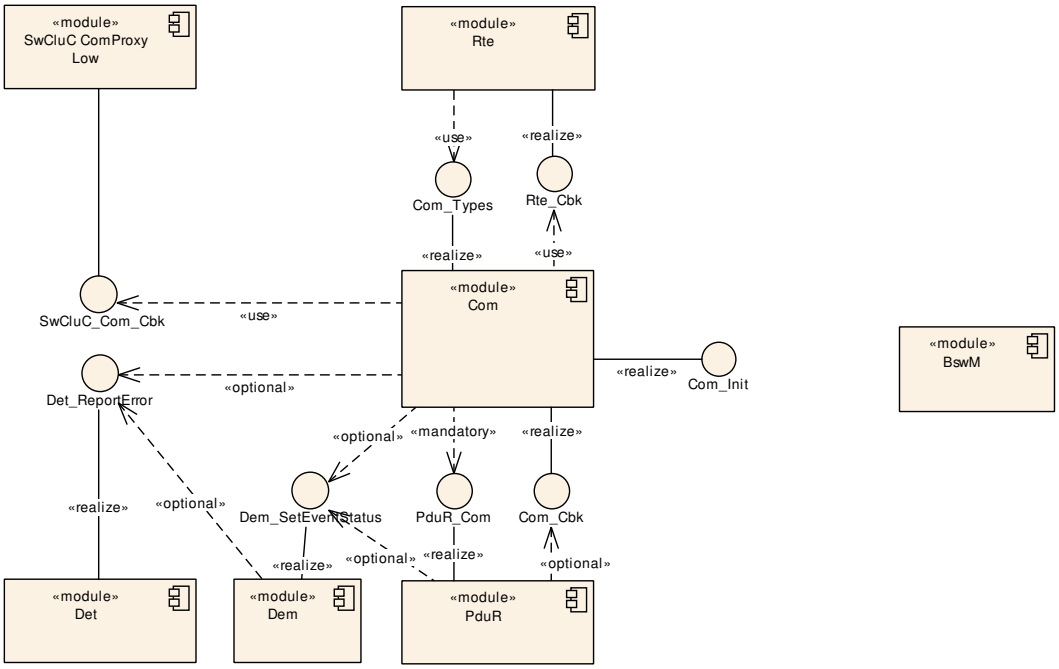


Figure 5.1: AUTOSAR COM module's context view

5.1 PDU Router

The AUTOSAR COM module uses the union of both sets of PDU Router's upper layer module APIs. That is the APIs for upper layer modules that uses TP and the APIs for upper layer modules that do not use TP. This is necessary since the AUTOSAR COM module transports I-PDUs either unfragmented via simple L-PDUs or fragmented via TP.

The following summarizes the functionality of the AUTOSAR COM module needs from the underlying layer PDU Router:

- Indication of incoming I-PDUs
- Sending interface for outgoing I-PDUs including the confirmation if an I-PDU has been sent by the communication controller
- Trigger interface to enable the PDU Router to cause a transmission from the AUTOSAR COM module
- Buffer handling for TP communication

Chapter 7.4 and Chapter 9.1 include a detailed description of the interfaces to the PDU Router. For further information, see [5].

5.2 Com Users

5.2.1 Runtime Environment (RTE)

The RTE uses the capabilities of the AUTOSAR COM module to send and receive signals. In AUTOSAR, the RTE is the higher layer above the AUTOSAR COM module. For further information, see [6].

5.2.2 SwCluC

SwCluC establishes inter-Software Cluster connection for exchanging (both providing and consuming) signals and signal groups. The Com users in Application Software Clusters can access the signals and signal groups via the SwCluC Com High Proxy. In the Host Software Cluster the SwCluC Com Low Proxy has the interface to the COM module.

Hence, the SwCluC Com Low Proxy provides callbacks for Signal- and Signal Group reception and transmission. It can also invoke the COM APIs for Signal reception and transmission. The existence of the SwCluC callbacks is configuration dependent. For further information, see [7].

5.3 Diagnostic Event Manager (DEM)

The AUTOSAR COM module reports timeout status of signals and signal groups. For further information, see [8].

5.4 Default Error Tracer (DET)

The DET provides services to report development and runtime errors, see Chapter 7.12 and [9].

6 Requirements Tracing

The following tables reference the requirements specified in [10] and links to the fulfillment of these. Requirements that are not fulfilled by this document are linked to [SWS_Com_NA_00999].

Requirement	Description	Satisfied by
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_Com_00015] [SWS_Com_00059] [SWS_Com_00098] [SWS_Com_00117] [SWS_Com_00128] [SWS_Com_00217] [SWS_Com_00432] [SWS_Com_00484] [SWS_Com_00850]
[SRS_BSW_00301]	All AUTOSAR Basic Software Modules shall only import the necessary information	[SWS_Com_00609]
[SRS_BSW_00312]	Shared code shall be reentrant	[SWS_Com_00321]
[SRS_BSW_00327]	Error values naming convention	[SWS_Com_00442]
[SRS_BSW_00331]	All Basic Software Modules shall strictly separate error and status information	[SWS_Com_00194]
[SRS_BSW_00335]	Status values naming convention	[SWS_Com_00819]
[SRS_BSW_00336]	Basic SW module shall be able to shutdown	[SWS_Com_00129] [SWS_Com_00130]
[SRS_BSW_00337]	Classification of development errors	[SWS_Com_91014]
[SRS_BSW_00344]	BSW Modules shall support link-time configuration	[SWS_Com_00432]
[SRS_BSW_00348]	All AUTOSAR standard types and constants shall be placed and organized in a standard type header file	[SWS_Com_00865]
[SRS_BSW_00358]	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	[SWS_Com_00432]
[SRS_BSW_00359]	Callback Function Return Types for AUTOSAR BSW	[SWS_Com_91013] [SWS_Com_91016] [SWS_Com_91017] [SWS_Com_91018] [SWS_Com_91019] [SWS_Com_91020] [SWS_Com_91021]
[SRS_BSW_00360]	AUTOSAR Basic Software Modules callback functions are allowed to have parameters	[SWS_Com_91013] [SWS_Com_91016] [SWS_Com_91017] [SWS_Com_91018] [SWS_Com_91019] [SWS_Com_91020] [SWS_Com_91021]
[SRS_BSW_00369]	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	[SWS_Com_00442]
[SRS_BSW_00377]	A Basic Software Module can return a module specific types	[SWS_Com_00865]
[SRS_BSW_00384]	The Basic Software Module specifications shall specify at least in the description which other modules they require	[SWS_Com_00669] [SWS_Com_00670]
[SRS_BSW_00385]	List possible error notifications	[SWS_Com_00442]
[SRS_BSW_00404]	BSW Modules shall support post-build configuration	[SWS_Com_00432] [SWS_Com_00825]



△

Requirement	Description	Satisfied by
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[SWS_Com_00432]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[SWS_Com_00432] [SWS_Com_91014]
[SRS_BSW_00425]	The BSW module description template shall provide means to model the defined trigger conditions of schedulable objects	[SWS_Com_00359] [SWS_Com_00398] [SWS_Com_00399] [SWS_Com_00400] [SWS_Com_00664] [SWS_Com_00665] [SWS_Com_00666]
[SRS_BSW_00432]	Modules should have separate main processing functions for read/receive and write/transmit data path	[SWS_Com_00359] [SWS_Com_00398] [SWS_Com_00399] [SWS_Com_00400] [SWS_Com_00466] [SWS_Com_00885] [SWS_Com_00886] [SWS_Com_00890]
[SRS_BSW_00441]	Naming convention for type, macro and function	[SWS_Com_00820] [SWS_Com_00821] [SWS_Com_00822] [SWS_Com_00825] [SWS_Com_00865]
[SRS_BSW_00452]	Classification of runtime errors	[SWS_Com_91015]
[SRS_BSW_00491]	Specification of trigger conditions and context data	[SWS_Com_00902]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 General Functionality

7.1.1 AUTOSAR COM basis

The ISO 17356-4 is the functional basis of the AUTOSAR COM module. In this document, it is referenced as [2].

[SWS_Com_00010] ⌈The AUTOSAR COM module shall implement all the functionality and all the APIs of [2] except the features and APIs mentioned in [SWS_Com_00013].⌋

[SWS_Com_00011] ⌈If this AUTOSAR COM specification defines functionality in a different way compared to definitions in [2], the AUTOSAR COM module shall implement the functionality defined in this AUTOSAR COM specification.⌋

[SWS_Com_00012] ⌈The AUTOSAR COM module shall in addition implement all those features, that are defined in this AUTOSAR COM specification and that are not part of [2].⌋

[SWS_Com_00013] ⌈The AUTOSAR COM module may implement the following features of [2]. If they are implemented in a specific AUTOSAR COM module, the configuration shall disable them by default. This also applies for all other additional features a specific implementation may provide.⌋

Feature in [2]	Rationale	Related API in [2]
Mapping of a received network message (within an I-PDU) to more than one message data objects (1:n splitting mechanism)	not required, done by the RTE, see [6]	none
Mapping of an internal message to more than one message data objects (1:n splitting mechanism)	not required, done by the RTE, see [6]	none
Mapping an only locally send message to both an external send message object and an internal receive message object (1:n splitting mechanism)	not required, done by the RTE, see [6]	none
M:1 sending; mapping of messages from multiple senders to one and the same message object	not required, ensured by RTE, see [6]	SendMessage
Queued messages	not required, done by the RTE, see [6]	GetMessageStatus
Zero size messages	it is possible to set up communication without them functionality is partly covered by `Com_TriggerTransmit`	SendZeroMessage
Notification mechanisms TASK, FLAG and EVENT	not required, done by the RTE, see [6]	none
Overlapping messages in an I-PDU	no use case, dangerous concept	none
Application modes	not needed	GetComApplicationMode

▽

△

Feature in [2]	Rationale	Related API in [2]
Start-up behavior	replaced by • Com_Init • Com_DeInit • Com_IpduGroupStart • Com_IpduGroupStop	StartCOM, StopCOM, StartCOMExtensions, InitMessage
Start and stop of periodic messages	no use case, is realized by I-PDU group mechanism	StartPeriodic, StopPeriodic
Reentrancy	Not all of the AUTOSAR API calls are reentrant. See Chapter 8.3.	See Chapter 8.3.
Interface to indirect NM	not needed	I_MessageTransfer, I_MessageTimeOut
Sender side filtering	no use case, the filter conditions are still used in the selection of the transmission mode but there is no signal filtering	none
Network-order message callout CPU-order message callout	Only I-PDU callouts with a defined AUTOSAR interface are supported by the AUTOSAR COM module. This is to avoid proprietary solutions.	none
Error hook routine	The AUTOSAR COM module defines an own interface for error reporting instead of using error hooks	COMErrorHook COMError_Name1_Name2 macros COMErrorGetServiceId
Interface for callback routines	The signatures for the used callback function of the AUTOSAR COM module will be explicitly defined within the AUTOSAR COM module's specification.	COMCallback
Internal communication	not required, ensured by RTE, see [6]	SendMessage, ReceiveMessage

Table 7.1: Excluded features of [2] in the AUTOSAR COM module

7.1.2 Signal Values

The signals sent by the AUTOSAR COM module respectively received by the AUTOSAR COM module could have the values defined in Table 7.2.

Signal value	Remark
init value	See Chapter 7.2.1.4 for details.
data invalid value	See Chapter 7.2.4 for details.
<value>	This is the normal case: A valid value after initialization phase, which is sent by the AUTOSAR COM module respectively, received by AUTOSAR COM module.

Table 7.2: Possible signal values

7.1.3 Endianness Conversion and Sign Extension

[SWS_Com_00675] ⌈The AUTOSAR COM module shall support the following data types:

- boolean
- uint8
- uint16
- uint32
- uint64
- sint8
- sint16
- sint32
- sint64
- uint8[n]
- float32
- float64

The type uint8[n] is mapped to either ComSignalType UINT8_N or UINT8_DYN.⌋

[SWS_Com_00007] ⌈The AUTOSAR COM module shall support endianness conversion for all supported signed and unsigned integer data types (see [SWS_Com_00675]).⌋

[SWS_Com_00810] ⌈The AUTOSAR COM module shall treat non-integer data types (e.g. uint8[n] or float32) either like integer data types of the matching size or leave their contents uninterpreted in case their ComSignalEndianness is configured to OPAQUE.⌋

[SWS_Com_00472] ⌈The AUTOSAR COM module shall interpret opaque data as uint8[n] and shall always map it to an n-bytes sized signal.⌋

For opaque data endianness, conversion has to be configured to OPAQUE (see ComSignalEndianness).

[SWS_Com_00674] ⌈The AUTOSAR COM module shall extend the endianness conversion defined in Clause 3.4 (Byte order conversion and message interpretation) of [2] to signed data types.⌋

Clause 3.4 (Byte order conversion and message interpretation) of [2] defines the endianness conversion for unsigned data types. The associated configurations can be found in Chapter 10. See also ComSignalType and ComSignalEndianness.

[SWS_Com_00829] ⌈The AUTOSAR COM module shall extend the endianness conversion defined in Clause 3.4 (Byte order conversion and message interpretation) of [2] to signals of signal groups.⌋

AUTOSAR COM handles signals and group signals equally with respect to endianness conversion.

[SWS_Com_00008] ⌈The AUTOSAR COM module shall extend received data to the size of the `ComSignalType` of the receive signal (sign extension).⌋

The platform specific representation of signed data has to be taken into account. Negative values of signed data will be mapped correctly.

Example: A 10-Bit signed signal is received and copied by `Com_ReceiveSignal` to a 16-Bit signed integer variable. If $(-3)_{decimal}$ is received, the received 10-Bit signal has a value of 1111111101b. While copying it to the 16-Bit integer variable the value will be extended to 1111111111111101b.

[SWS_Com_00723] ⌈The AUTOSAR COM module shall extend the init value (`ComSignalInitValue`) of a signal to the size of its `ComSignalType`.⌋

Note that the AUTOSAR COM module will not need to perform sign extensions on sender side as it is guaranteed that the values to be transmitted will always be sign-extended. See also [SWS_Com_CONSTR_00785].

[SWS_Com_00579] ⌈The AUTOSAR COM module shall not support sign extension for float32 and float64 data types.⌋

[SWS_Com_00221] ⌈The AUTOSAR COM module shall perform endianness conversion before the I-PDU callout on sender side.⌋

For an overview, see Chapter 7.13.

[SWS_Com_00352] ⌈The AUTOSAR COM module shall perform sign extensions and endianness conversion before performing filtering and notification detection on receiver side.⌋

[SWS_Com_00580] ⌈Beside endianness conversion, the AUTOSAR COM module shall not support further conversions for signals with `ComSignalType FLOAT32` or `FLOAT64`. That is endianness conversion shall be supported but complex conversion or normalizations of fractions, exponents, signs or bias values shall not be supported.⌋

7.1.4 Filtering

[SWS_Com_00694] ⌈The AUTOSAR COM module shall evaluate each filtering condition to either true or false.⌋

[SWS_Com_00695] ⌈The AUTOSAR COM module shall filter out signals only at receiver side.⌋

[SWS_Com_00602] ⌈The AUTOSAR COM module shall use filtering mechanisms on sender side for Transmission Mode Conditions (TMC) but it shall not filter out signals on sender side.⌋

For Transmission Mode Selection (TMS) see Chapters 7.2.3.2 and 7.2.3.3.

The AUTOSAR COM module only provides the following ComFilterAlgorithm of that which are defined in [2]:

- ALWAYS
- NEVER
- MASKED_NEW_EQUALS_X
- MASKED_NEW_DIFFERS_X
- MASKED_NEW_DIFFERS_MASKED_OLD
- NEW_IS_WITHIN
- NEW_IS_OUTSIDE
- ONE_EVERY_N

To reduce complexity the AUTOSAR COM module does not support all filters defined in [2]. The not supported filters are either obsolete or special cases of other filters. For example, the filter NEW_IS_DIFFERENT is a special case of MASKED_NEW_DIFFERS_MASKED_OLD with a fully set mask.

[SWS_Com_00325] ⌈The AUTOSAR COM module shall support all filter mechanisms listed in ComFilterAlgorithm, considering the exceptions defined in [SWS_Com_CONSTR_00380] and [SWS_Com_CONSTR_00439].⌋

[SWS_Com_00866] ⌈In case the configured filter condition is MASKED_NEW_DIFFERS_X, MASKED_NEW_EQUALS_X, NEW_IS_OUTSIDE or NEW_IS_WITHIN, the AUTOSAR COM module shall calculate the reception filter by using only the least significant ComBitSize bits of the filter parameters.⌋

[SWS_Com_00273] ⌈If the AUTOSAR COM module filters out a signal on receiver side, i.e. filter condition evaluates to false, the AUTOSAR COM module shall discard that signal and shall not process it. See also [SWS_Com_00303].⌋

[SWS_Com_00836] ⌈If the AUTOSAR COM module filters out a group signal on receiver side, i.e. filter condition evaluates to false, the AUTOSAR COM module shall discard the whole signal group and shall not process it.⌋

[SWS_Com_00132] ⌈The AUTOSAR COM module shall support the filtering mechanisms as defined in ComFilterAlgorithm also for signed data types.⌋

In the case a filter is evaluated before a send-API has written the corresponding signal, there needs to be a way to determine the filter state of this signal. Some of the filters

require a new_value to evaluate the filter. However, this is only available after the signal has been updated using a send-API. Therefore, it is necessary to define the value used by the filter for new_value in the period before the first send takes place.

[SWS_Com_00603] ⌈The AUTOSAR COM module shall set the old_value of the filtering mechanisms for each signal to the `ComSignalInitValue` during start-up. See also [2].⌋

[SWS_Com_00604] ⌈Until the application has not updated the new_value of the filtering mechanisms for a signal, the AUTOSAR COM module shall use the `ComSignalInitValue` as the new_value for that signal.⌋

The next two requirements clarify the definitions of [2] according to the update of the old_value of filters.

[SWS_Com_00302] ⌈If the AUTOSAR COM module evaluates a filter for a signal to true, (value is not filtered out) then the AUTOSAR COM module shall place the value of that signal into old_value (as defined in [2]).⌋

[SWS_Com_00303] ⌈When a value is being filtered, if the filter does not allow the passage of the value (i.e. the filter evaluates to false) then the AUTOSAR COM module shall not place that value into old_value (as defined in [2]).⌋

[SWS_Com_00231] ⌈In the case of `ComFilterAlgorithm` is configured to `ONE_EVERY_N`, the AUTOSAR COM module shall

- set OCCURRENCE to zero when OCCURRENCE == PERIOD
- set FILTER to true, when OCCURRENCE == OFFSET
- increment OCCURRENCE after filter processing

⌋

For definition of OCCURRENCE, FILTER, OFFSET and PERIOD see [2].

Configuring `ComFilterAlgorithm` to `ONE_EVERY_N` for a signal has the effect that the signal is passed by the filter (i.e. the filter returns true) once every PERIOD calls of the filter. If the OFFSET parameter is zero then the first time the filter is used the signal is allowed to pass (i.e. filter returns true). If the OFFSET is greater than zero then more than one signal must pass through the filter before it returns true.

In case the I-PDU is started by `Com_IpduGroupStart` with parameter initialize set to true, the OCCURRENCE is also set to zero, see [SWS_Com_00787].

This definition exists to clarify the description of the `ONE_EVERY_N` filter in [2].

The associated configuration items can be found in Chapter 10, see `ComFilter`.

[SWS_Com_00793] ⌈For a signal with a configured reception filter `MASKED_NEW_DIFFERS_MASKED_OLD`, the AUTOSAR COM module shall treat the first value re-

ceived for this signal after a reception deadline monitoring timeout occurred for this signal the same way as if the value has passed the filter criteria.⌋

Hence, the AUTOSAR COM module will let pass any value for the filter MASKED_NEW_DIFFERS_MASKED_OLD after an RX deadline timeout for the associated I-PDU.

7.1.5 Signal Gateway

The AUTOSAR COM module provides an integrated Signal Gateway for forwarding signals and group signals in a 1:n manner.

The routing relations are be statically configured via the ComGwMapping configuration container.

As shown in Figure 7.3 and Figure 7.4, the integrated Signal Gateway acts as a receiver for all signals or group signals which are configured as a source for gatewayed signals.

After the Signal Gateway received signal or group signals for routing, it acts immediately as a sender for these signals or group signals respectively. This is shown in Figure 7.2 and Figure 7.4. The signal processing does not differ if the integrated Signal Gateway forwards a signal / group signal or if a Software Component sends it.

Since the gateway relations are statically configured, an optimized Signal Gateway might skip some processing stages for concrete configurations. For example, the endianness conversion is not necessary in certain cases. Of course, such optimizations should not lead to any different logical behavior or different representations on the target bus. Optimizations may only be introduced to increase the Signal Gateway's performance.

As shown in Figure 7.3 a received signal or group signal might be received on the local ECU and be a source for a gateway action at the same time.

Figure 7.2 and Figure 7.3 show shadow buffers only for group signals that are received by the RTE. The integrated Signal Gateway has of course to ensure to handle group signals consistently but there is no predetermined way in which this must be implemented.

[SWS_Com_00370] ⌈The Signal Gateway of the AUTOSAR COM module shall scale down to no size if no signal routing functionality is needed.⌋

7.2 Normal Operation

7.2.1 Start-Up Behavior

This chapter describes the actions that will be performed during Com_Init.

[SWS_Com_00217]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module shall initialize each I-PDU during execution of `Com_Init`, firstly byte wise with the `ComTxIPduUnusedAreasDefault` value and then bit wise according to initial values (`ComSignalInitValue`) of the contained signals and the update-bits (see [SWS_Com_00117]).⌋

7.2.1.1 Preconditions

The C initialization code, also known as start-up code, initializes global and static variables with the initial values. It must be executed before any call of an AUTOSAR COM module's service.

7.2.1.2 Initialization

[SWS_Com_00128]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module's initialization function `Com_Init` shall initialize all internal data that is not yet initialized by the start-up code e.g. C-structs.⌋

This initialization chapter is not complete. Details about initialization of some AUTOSAR COM module's features are described within the different feature chapters.

7.2.1.3 Initialization of Not Used Areas of an I-PDU

[SWS_Com_00015]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module shall fill not used areas within an I-PDU with a value determined by configuration parameter `ComTxIPduUnusedAreasDefault` e.g. 0xFF.⌋

7.2.1.4 Initialization of Signals and Update-Bits

[SWS_Com_00098]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module shall initialize each signal of n-bit sized signal type on sender and receiver side with the lower n-bits of its configuration parameter `ComSignalInitValue`.⌋

The configured ComSignalInitValue is also used for the initialization of the signal in the related I-PDU; see [SWS_Com_00217].

The ComSignalInitValue of a signal can be identical to its ComSignalDataInvalidValue. These can be different for each signal.

[SWS_Com_00117]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module shall clear all update-bits during initialization. See also [SWS_Com_00059].⌋

7.2.1.5 Initialization of I-PDU Groups

[SWS_Com_00444] ⌈By default, all I-PDU groups shall be in the state stopped and they shall not be started automatically by a call to Com_Init.⌋

7.2.2 De-Initialization

The AUTOSAR COM module provides the API function Com_DeInit for de-initialization of the COM layer. This means, after de-initialization of the layer, no communication via the AUTOSAR COM module is possible and all started I-PDU groups are stopped, see also [SWS_Com_00129].

7.2.3 Communication Modes

This chapter defines the signal flow in the AUTOSAR COM module. It further defines the different transmission modes provided by the AUTOSAR COM module. Chapter 7.2.3.2 defines a mechanism to switch between two transmission modes for one I-PDU. The replication of signals is defined in Chapter 7.2.3.4.

Exemplary communication use cases that the AUTOSAR COM module can deal with are shown in Chapter B.

7.2.3.1 Transfer Properties and I-PDU Transmission Mode

7.2.3.1.1 Signals

The AUTOSAR COM module supports several transfer properties for signals and several transmission modes for I-PDUs. The definitions in this chapters are based on and are to be completed with the definitions in Clause 3.3.3 (Transfer properties for external communication) of [2].

[SWS_Com_00330] ⌈At any send request of a signal with ComTransferProperty TRIGGERED assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate ComTxModeNumberOfRepetitions plus one transmissions of the assigned I-PDU.⌋

[SWS_Com_00767] ⌈At any send request of a signal with ComTransferProperty TRIGGERED_WITHOUT_REPETITION assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate one transmission of the assigned I-PDU.⌋

[SWS_Com_00734] ⌈At a send request of a signal with ComTransferProperty TRIGGERED_ON_CHANGE assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate ComTxModeNumberOfRepetitions plus one transmissions of the assigned I-PDU, if the new sent signal differs to the locally stored (last sent or init) in length or value.⌋

[SWS_Com_00768] ⌈At a send request of a signal with ComTransferProperty TRIGGERED_ON_CHANGE_WITHOUT_REPETITION assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate one transmission of the assigned I-PDU, if the new sent signal differs to the locally stored (last sent or init) in length or value.⌋

The support of the transfer properties TRIGGERED_ON_CHANGE and TRIGGERED_ON_CHANGE_WITHOUT_REPETITION is not restricted to certain signal types. Hence, they are supported for all possible signal types. This includes even the byte array types, e.g. UINT8_N.

The details of the transmission replication mechanism are specified in Chapter 7.2.3.4.

[SWS_Com_00135] ⌈The AUTOSAR COM module shall not initiate transmissions for I-PDUs that have the ComTxModeMode NONE.⌋

It is possible to request I-PDUs with ComTxModeMode NONE via Com_TriggerTransmit.

A pending signal associated with an I-PDU is transmitted if the I-PDU's transmission is triggered for any reason. For example, if a signal with ComTransferProperty TRIGGERED within the same I-PDU is sent or the I-PDU sending is scheduled because of ComTxModeMode PERIODIC or MIXED.

The bus-timing of I-PDUs can be controlled by send requests of the users (e.g. RTE, SwCluC) in combination with the transmission mode and the transfer property as described above. Additionally, the PDU Router, especially in case of FlexRay and LIN, can control it with the service Com_TriggerTransmit. In the latter case, the PDU Router requests I-PDUs to be sent from the AUTOSAR COM module.

The function Com_TriggerTransmit can be called for any I-PDU regardless of its transmission mode. This allows LIN and FlexRay to use all the available transmission modes, particularly for sporadic communication. The NM also uses this mechanism to send user data.

7.2.3.1.2 Signal Groups

In AUTOSAR COM also signal groups and group signals may have a transfer property, defining in combination with the transmission mode, if the I-PDU is sent out in case of an update of a signal group or group signal, respectively.

[SWS_Com_00741] ⌈At any send request of a signal group with ComTransferProperty TRIGGERED assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate ComTxModeNumberOfRepetitions plus one transmissions of the assigned I-PDU.⌋

[SWS_Com_00769] ⌈At any send request of a signal group with ComTransferProperty TRIGGERED_WITHOUT_REPETITION assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall initiate one transmission of the assigned I-PDU within the next main function at the latest.⌋

[SWS_Com_00742] ⌈Regarding signal groups with ComTransferProperty TRIGGERED_ON_CHANGE which do not contain any signals that have an own ComGroupSignalTransferProperty configured:

At any send request of such a signal group assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate ComTxModeNumberOfRepetitions plus one transmissions of the assigned I-PDU, if at least one new sent group signal differs to the locally stored (last sent or init) in length or value.⌋

[SWS_Com_00743] ⌈Regarding signal groups with ComTransferProperty TRIGGERED_ON_CHANGE which contain any signals that have an own ComGroupSignalTransferProperty configured:

At any send request of such a signal group assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate ComTxModeNumberOfRepetitions plus one transmissions of the assigned I-PDU, if at least one new sent group signal configured with ComGroupSignalTransferProperty TRIGGERED_ON_CHANGE differs to the locally stored (last sent or init) in length or value.⌋

[SWS_Com_00770] ⌈At a send request of a signal group with ComTransferProperty TRIGGERED_ON_CHANGE_WITHOUT_REPETITION assigned to an I-PDU with ComTxModeMode DIRECT or MIXED, the AUTOSAR COM module shall immediately (within the next main function at the latest) initiate one transmission of the as-

signed I-PDU, if at least one new sent group signal differs to the locally stored (last sent or init) in length or value.⌋

7.2.3.2 Selection of the Transmission Mode for one specific I-PDU

I-PDUs carry signals. Because an I-PDU can contain more than one signal, in the following, a method is defined to derive the I-PDU's transmission mode from the state of the signals that are contained in one specific I-PDU.

The AUTOSAR COM module allows configuring statically two different transmission modes for each I-PDU (see [SWS_Com_00032]). The transmission mode of an I-PDU that is valid at a specific point in time is selected using only the values of the signals that are mapped to this I-PDU.

The signals of one I-PDU that contribute to the selection of one of the two transmission modes as well as the conditions used for the selection of the transmission mode are configured statically, see [SWS_Com_00676].

For the selection of the transmission mode, the AUTOSAR COM module treats group signals like normal signals.

[SWS_Com_00676] ⌈For the evaluation of a TMS, the AUTOSAR COM module shall take all those (group) signals into account whose configuration include a configured `ComFilter` container. See included containers of `ComSignal` and `ComGroupSignal`.⌋

If a signal of an I-PDU with `ComIPduDirection` configured to `SEND` has a configuration container `ComFilter` included, the signal is named to contribute to the TMS of this I-PDU. If the configuration container for that signal has no assigned configuration container `ComFilter`, then it does not contribute to a TMS.

[SWS_Com_00677] ⌈If no signal within an I-PDU contributes to the calculation of the TMS, then the AUTOSAR COM module shall evaluate the TMS of this I-PDU as true.⌋

[SWS_Com_00678] ⌈If the AUTOSAR COM module evaluates the TMC of a contributing signal as true, then the AUTOSAR COM module shall evaluate the TMS as true.⌋

[SWS_Com_00679] ⌈If the AUTOSAR COM module evaluates no TMC of a contributing signal as true, then the AUTOSAR COM module shall evaluate the TMS as false.⌋

[SWS_Com_00605] ⌈The AUTOSAR COM module shall define a Transmission Mode Selector, for each I-PDU. See definition of TMS above.⌋

[SWS_Com_00245] ⌈The AUTOSAR COM module shall re-calculate the TMS for an I-PDU, after a contained signal was updated by a call to `Com_SendSignal` or `Com_SendSignalGroup`.⌋

[SWS_Com_00763] ⌈For the calculation of the transmission mode with the configured condition MASKED_NEW_DIFFERS_MASKED_OLD, the AUTOSAR COM module shall use the least significant ComBitSize bits only.⌋

[SWS_Com_00813] ⌈For the calculation of the transmission mode with a configured condition MASKED_NEW_DIFFERS_X, MASKED_NEW_EQUALS_X, NEW_IS_OUTSIDE or NEW_IS_WITHIN, the AUTOSAR COM module shall use all bits of the configured ComSignalType.⌋

Note that a signal with ComFilterAlgorithm configured to ALWAYS, will always set the TMS of the respective I-PDU to true. Therefore, care must be taken when defining the signals that contribute to the TMS.

[SWS_Com_00032] ⌈If the TMS of an I-PDU evaluates to true, then the AUTOSAR COM module shall use the ComTxModeMode defined in configuration container ComTxModeTrue for that I-PDU.⌋

[SWS_Com_00799] ⌈If the TMS for an I-PDU evaluates to false, then the AUTOSAR COM module shall use the ComTxModeMode defined in configuration container ComTxModeFalse for that I-PDU.⌋

[SWS_Com_00238] ⌈In each of the two TMS states, the rules for combination of transfer properties of signals and transmission modes of I-PDUs shall apply as defined in Clause 3.3 (Message transmission) of [2].⌋

[SWS_Com_00239] ⌈When the TMS state of an I-PDU changes, the AUTOSAR COM module shall use the now valid transmission mode immediately (within the next main function at the latest). That means, first the AUTOSAR COM module shall perform the mode change and after that, the AUTOSAR COM module shall execute any resulting calls to PduR_ComTransmit caused by mode change.⌋

[SWS_Com_00244] ⌈If a change of the TMS causes a change of the transmission mode for an I-PDU, then the AUTOSAR COM module shall restart the timer for the cycle time of the transmission mode PERIODIC and MIXED.⌋

[SWS_Com_00495] ⌈When a call to Com_SendSignal or Com_SendSignalGroup results into a change of the transmission mode of a started I-PDU to the transmission mode PERIODIC or MIXED, then the AUTOSAR COM module shall start the first transmission immediately with respect to the minimum delay time, i.e. it shall call PduR_ComTransmit within the next main function at the latest where minimum delay time elapsed. The transmission shall be initiated regardless of the transfer property of the signal or signal group that caused the transmission mode switch.⌋

See also Figure B.6.

The ComTxModeTimeOffset is not respected. It is only respected by explicit I-PDU transmission mode switches, for example by Com_IpduGroupStart or Com_SwitchIpduTxMode.

[SWS_Com_00582] ⌈If a change of the TMS causes a change to the transmission mode `DIRECT`, an immediate (within the next main function at the latest unless shifted due to the MDT) direct / n-times transmission to the underlying layer shall be initiated.⌋

The above requirement clarifies the behavior in case the TMS-switch to `ComTxMode-Mode DIRECT` was triggered by a signal with `ComTransferProperty PENDING`.

If the `ComTxModeMode` of an I-PDU is configured to `NONE` no transmission will be initiated by AUTOSAR COM. This can be used to prevent transmitting an I-PDU, for example in case the TMS evaluates to false.

[SWS_Com_00478] ⌈The AUTOSAR COM module shall send out an I-PDU at most once within one call of `Com_MainFunctionTx`.⌋

7.2.3.3 Signal Flow and Transmission Mode Selection

After a send request from the user (e.g. RTE or SwCluC) for a specific signal, the signal is written to the appropriate I-PDU buffer as defined by configuration, and the selection of the transmission mode of the I-PDUs is done according to Chapter 7.2.3.2.

Figure 7.1 shows the signal flow (Conditions are the same as Filters in [2]):

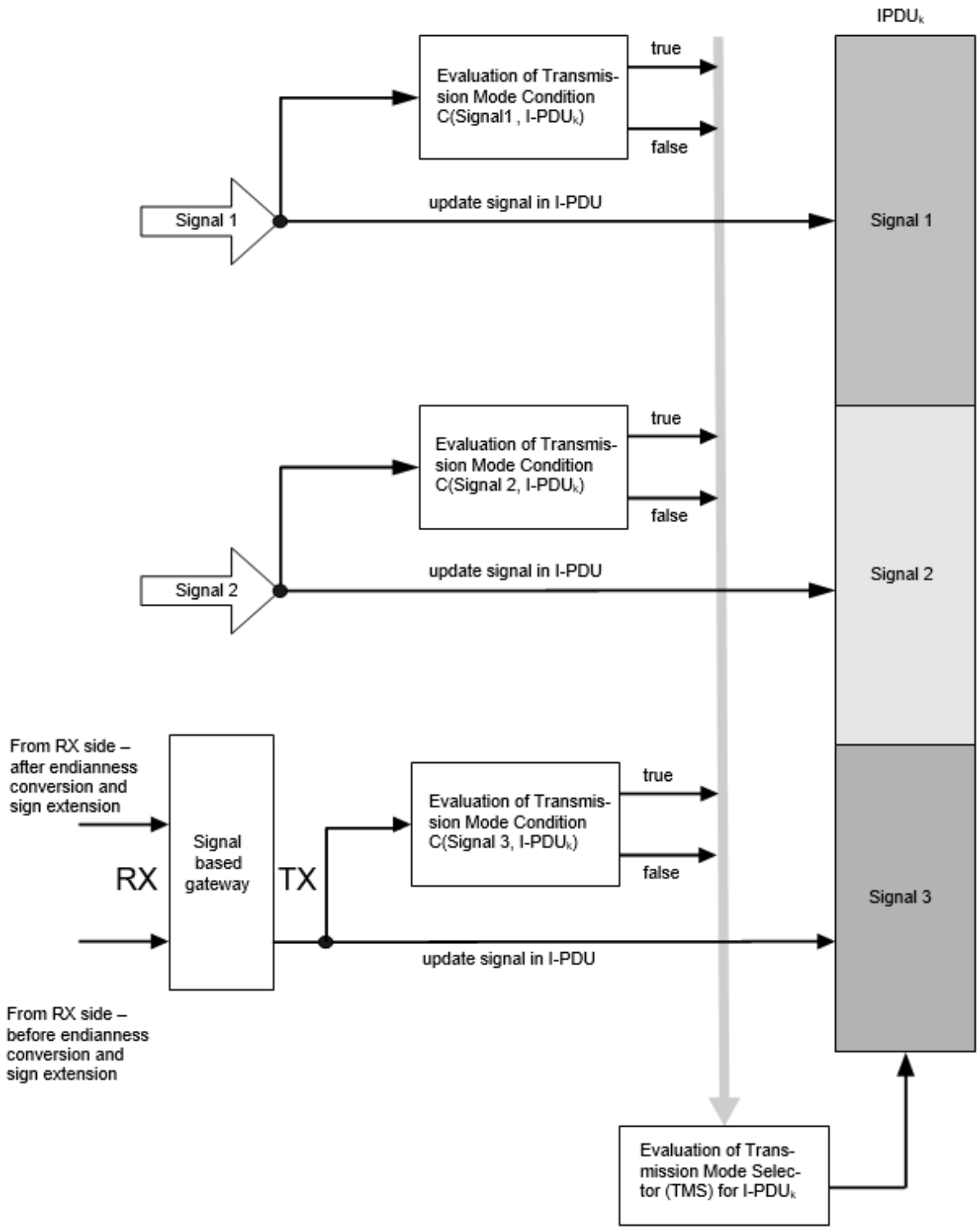


Figure 7.1: Logical signal flow in the AUTOSAR COM module shown for two signals (Signal1 and Signal2) that are mapped to one I-PDU ($IPDU_k$)

7.2.3.4 Replication of Signal Transmission Requests

The number of repetitions of transmission requests in the ComTxModeMode DIRECT or MIXED for a send request by the user (e.g. RTE or SwCluC) is defined by configuration parameter ComTxModeNumberOfRepetitions.

[SWS_Com_00467] ⌈If ComRetryFailedTransmitRequests is set to false and an I-PDU with ComTxModeMode DIRECT or MIXED and ComTxModeNumberOf-Repetitions set to 0 is triggered for sending, the AUTOSAR COM module shall invoke PduR_ComTransmit for this I-PDU just once, independently of the result of the confirmation.⌋

Configuring ComTxModeNumberOfRepetitions to 0 imitates the original direct transmission mode of [2].

[SWS_Com_00279] ⌈If a new send request is received from the user (e.g. RTE or SwCluC) while sending n transmissions belonging together (e.g. after the 3rd of 5 repetitions, see [SWS_Com_00305]) the AUTOSAR COM module shall cancel the remaining transmission repetitions and start processing the new request within the next main function at the latest.⌋

See also Use case B.4.

[SWS_Com_00305] ⌈In case of sending an I-PDU with ComTxModeMode DIRECT or MIXED and ComTxModeNumberOfRepetitions greater than 0, the AUTOSAR COM module shall call PduR_ComTransmit periodically, with period ComTxMode-RepetitionPeriod until ComTxModeNumberOfRepetitions plus one successful confirmations for this send-request are received.⌋

[SWS_Com_00305] assumes that CAN does not have a queue for these I-PDUs. There is a race condition in the interaction between the CAN driver, interface and hardware that may cause an extra transmission to occur in certain unlikely circumstances.

If the underlying layer returns E_NOT_OK while an N-Times transmission is in progress, this error notification will be ignored, unless ComRetryFailedTransmitRequests is set to true (see also [SWS_Com_00773]). As [SWS_Com_00305] specifies, only confirmed transmissions are counted for the N-Times transmission, erroneous send request can safely be ignored.

If the N-Times transmission is requested in transmission mode MIXED after a cyclic transmission of the MIXED transmission mode with a pending confirmation, the confirmation of the cyclic transmission will be assigned to the N-Times transmission. In this case, only n-1 transmissions of the new value of the N-Times request are observed. The transmission deadline monitoring timer will then be reset earliest after the N-Times request is completed. This must be respected when configuring the transmission deadline monitoring timer in conjunction with the MIXED transmission mode and N-Times transmission.

[SWS_Com_00494] ⌈If within the transmission mode MIXED an N-Times transmission request overlaps with the cyclic part of the MIXED transmission the cyclic transmission shall be counted as the corresponding transmission of the N-Times transmission request.⌋

[SWS_Com_00392] ⌈If a transmission deadline monitoring timeout occurs before the N-Times transmission is complete, then the AUTOSAR COM module shall not initiate further transmissions for this N-Times transmission.⌋

The minimum delay time will always be taken into account as defined in Clause 3.3.4 (Transmission modes) of [2].

To avoid bursts in start-up a time offset can be configured per I-PDU. See ComTxModeTimeOffset for details.

The time between two repetitions is configured by configuration parameter ComTxModeRepetitionPeriod.

If the transmission mode change leads to the start of the MIXED transmission mode by sending a triggered signal and ComTxModeNumberOfRepetitions is configured greater than or equal to 1, then there will be at least ComTxModeNumberOfRepetitions plus one transmission requests to the PDU Router at the beginning of the MIXED transmission mode. See also [SWS_Com_00305].

7.2.3.5 Retry Failed Transmission Requests

[SWS_Com_00773] ⌈If ComRetryFailedTransmitRequests is set to true and ComIPduLargeData is set to false, the return value of PduR_ComTransmit shall be evaluated. If the return value of PduR_ComTransmit is not equal to E_OK, the AUTOSAR COM Module shall invoke PduR_ComTransmit for the not sent I-PDU again within the next Com_MainFunctionTx.⌋

In case of retry, if the I-PDU is updated in between, the new data will be sent. Large Data Path avoids I-PDU buffering in COM, therefore retry is not possible in that case.

[SWS_Com_00773] may lead to multiple retries for the same failed transmission request. [SWS_Com_00773] is not affected by transmission mode changes. [SWS_Com_00773] has no influence on a potential repetition period. Any repetitions will occur as if no retry had taken place.

[SWS_Com_00774] ⌈The AUTOSAR COM module shall start the transmission deadline monitoring independently of possible retries. Subsequent retries shall not affect transmission deadline monitoring.⌋

[SWS_Com_00775] ⌈If ComRetryFailedTransmitRequests is set to true and a transmission deadline monitoring timeout occurs for an I-PDU, the AUTOSAR COM module shall expire any pending transmission request for this I-PDU.⌋

[SWS_Com_00776] ⌈The cycle timer for a cyclic transmission shall always start with the first transmit attempt.⌋

7.2.4 Signal Invalidation

The AUTOSAR COM module provides the possibility for the sender to indicate that it is not able to provide a valid value for a corresponding signal, for example in case a sensor is faulty. The AUTOSAR COM module allows defining a ComSignalDataInvalidValue during configuration.

7.2.4.1 Transmission of an Invalidated Signal

[SWS_Com_00099] ⌈By a call to Com_InvalidateSignal, the AUTOSAR COM module shall perform internally a Com_SendSignal with the configured ComSignalDataInvalidValue.⌋

The ComTransferProperty and the transmission mode determine the transmission of the ComSignalDataInvalidValue on the bus. The internally performed Com_SendSignal with the data invalid value leads to data invalid value to be used as current value for filters and TMS.

The data invalid values are configured per ComGroupSignal.

The VFB defines only one attribute for a complex data type. Therefore, the best mapping of an invalidated complex data type to an invalidated signal group is to invalidate all group signals of a signal group. Therefore, the user (e.g. RTE or SwCluC) can also request to invalidate the complete signal group by a call to Com_InvalidateSignalGroup.

7.2.4.2 Reception of an Invalidated Signal

[SWS_Com_00680] ⌈If the configured ComSignalDataInvalidValue is received for a signal and the ComDataInvalidAction is configured to NOTIFY for this signal, the AUTOSAR COM module shall notify the users (e.g. RTE, SwCluC) via the configured <ComUser_CbkInv> function. In this case, no other signal processing like filtering or the normal signal indication shall take place.⌋

The reception deadline monitoring timer is also restarted in case of receiving an invalid signal or signal group, see [SWS_Com_00738].

[SWS_Com_00681] ⌈If the configured ComSignalDataInvalidValue is received for a signal and the ComDataInvalidAction is configured to REPLACE for this signal, the AUTOSAR COM module shall replace the signal's value by its configured ComSignalInitValue. After the replacement, the normal signal processing like filtering

and notification shall take place as if the ComSignalInitValue would have been received instead of the ComSignalDataInvalidValue.⌋

[SWS_Com_00682] ⌈If the configured ComSignalDataInvalidValue is received for at least one group signal of a signal group and the ComDataInvalidAction is configured to NOTIFY for this signal group, the AUTOSAR COM module shall notify the users (e.g. RTE, SwCluC) via the configured <ComUser_CbkInv> function. In this case, no other signal group / group signal processing like the normal indication shall take place.⌋

[SWS_Com_00683] ⌈If the configured ComSignalDataInvalidValue is received for at least one group signal of a signal group and the ComDataInvalidAction is configured to REPLACE for this signal group, the AUTOSAR COM module shall replace all group signals of this signal group by their configured ComSignalInitValue values. After the replacement, the normal signal group / group signal processing like reception notification shall take place as if the ComSignalInitValue would have been received for all group signals.⌋

[SWS_Com_00717] ⌈If the configured ComSignalDataInvalidValue is received for a signal and its ComDataInvalidAction is configured to NOTIFY, the AUTOSAR COM module shall not store the received ComSignalDataInvalidValue into the signal object.⌋

The next call to Com_ReceiveSignal will return the last valid received signal or the ComSignalInitValue in case no signal was received yet respectively.

[SWS_Com_00718] ⌈If the configured ComSignalDataInvalidValue is received for at least one group signal of a signal group and its ComDataInvalidAction is configured to NOTIFY, the AUTOSAR COM module shall not store any of the received group signals into the signal objects.⌋

The next call to Com_ReceiveSignalGroup will copy the last valid received group signals or the ComSignalInitValue in case the signal group was not received yet respectively into the shadow buffer.

7.2.5 Handling of I-PDUs

7.2.5.1 I-PDU Group Definitions

For an I-PDU group the following rules apply:

1. An I-PDU can belong to any I-PDU group.
2. An I-PDU is active (started) if it belongs to a started I-PDU group (see [SWS_Com_00771]) or it does not belong to any I-PDU group and it was started implicitly (see [SWS_Com_00840]).
3. The maximum number of I-PDU groups is pre-compile configurable.

4. I-PDU groups may contain only I-PDUs of the same direction (i.e. send or receive), see [SWS_Com_CONSTR_00871].

Rules 1 and 3 are supported by the COM configuration. The maximum number of supported I-PDU groups can be configured via `ComSupportedIPduGroups`.

[SWS_Com_00771] ⌈An I-PDU is active (started) if at least one I-PDU group is active (started) it belongs to.⌋

Once again, the COM module does not know or handle any grouping of I-PDUs. Any I-PDU grouping must be handled either via configuration or by the calling module.

Further, it is expected that the complete state handling of I-PDU groups is done outside of the AUTOSAR COM module, e.g. within the Basic Software Mode Manager. In case of a state change, the module managing the I-PDU group states consistently starts or stops the I-PDU groups via `Com_IpduGroupStart` and `Com_IpduGroupStop`.

The state of I-PDUs that are not assigned to any I-PDU group cannot be changed, therefore they are started implicitly:

[SWS_Com_00840] ⌈If an I-PDU is not assigned to any I-PDU group, the AUTOSAR COM shall start this I-PDU within `Com_Init` as if it would be started by `Com_IpduGroupStart` with parameter Initialize set to true.⌋

See also Chapter 7.2.5.2.

Since the AUTOSAR COM never stops such an I-PDU implicitely and it is not assigned to any I-PDU group, such an I-PDU can never be stopped.

7.2.5.2 Starting of I-PDU Groups

By default all I-PDU groups are stopped, see [SWS_Com_00444]. A call to `Com_IpduGroupStart` starts an I-PDU group if it was previously stopped.

[SWS_Com_00114] ⌈If an I-PDU is started by `Com_IpduGroupStart`, the AUTOSAR COM module shall permit to transmit / receive its signals and signal groups.⌋

See also Table 7.4.

[SWS_Com_00787] ⌈If an I-PDU is started by `Com_IpduGroupStart`, the AUTOSAR COM module shall always initialize the following attributes of this I-PDU:

1. `ComMinimumDelayTime` of I-PDUs in transmission mode `DIRECT` or `MIXED`
2. restart all reception deadline monitoring timers for all signals with a non-zero configured `ComFirstTimeout`
3. cancel all transmission deadline monitoring timers and use `ComFirstTimeout` (if configured) as value when a transmission timer is started the first time after the I-PDU activation

4. all included update-bits shall be cleared
5. reset OCCURRENCE of filters with ComFilterAlgorithm ONE_EVERY_N

⌋

If I-PDU is already started through another I-PDU group then Com_IpduGroupStart does not change I-PDU state therefore no action is necessary.

[SWS_Com_00222] ⌈If an I-PDU is started by Com_IpduGroupStart with parameter Initialize set to true, the AUTOSAR COM module shall additionally to [SWS_Com_00787] initialize the following attributes of this I-PDU:

1. the data of the I-PDU as defined in [SWS_Com_00217]
2. the shadow buffers of included signal groups
3. old_value of the filtering mechanisms for each signal to the ComSignalInitValue
4. ComTxModeTimePeriod and ComTxModeTimeOffset of I-PDUs in PERIODIC or MIXED transmission mode

⌋

[SWS_Com_00223] ⌈If an I-PDU is started by Com_IpduGroupStart, the AUTOSAR COM module shall determine its transmission mode according to its current data content.⌋

[SWS_Com_00228] ⌈In some cases, an I-PDU is started by Com_IpduGroupStart before all its contained signals have been written. In this case, the AUTOSAR COM module shall use the ComSignalInitValue for the missing signal data.⌋

[SWS_Com_00229] ⌈When an I-PDU is started and one or more signals in that I-PDU have already been written via one of the send APIs by the upper layer, the AUTOSAR COM module shall use the most recently obtained values to determine the TMS of the I-PDU.⌋

[SWS_Com_00733] ⌈If an I-PDU is started by Com_IpduGroupStart and the I-PDU contains signals that have deadline monitoring configured ComFirstTimeout, ComTimeout, the AUTOSAR COM module shall start the deadline monitoring for these signals independently of the value of the initialize parameter.⌋

Note, in case an I-PDU is started by Com_IpduGroupStart and the parameter initialize is set to false, the internal buffers are not changed and the I-PDU contains the last the last set or init values. Particularly, if the I-PDU is started for the first time and initialize is set to false, the COM operates on the init-values set by Com_Init.

[SWS_Com_00877] ⌈If an I-PDU is not part of any I-PDU Group, it is started during the initialization of COM. Its starting transmission mode shall be evaluated according to the ComSignalInitValue of the signals contributing to its TMS.⌋

For this case, the evaluation can already be done at configuration time, since the signals cannot be written before the initialization of COM.

7.2.5.3 Stopping of I-PDU Groups

A call to Com_IpduGroupStop stops an I-PDU group, if it was previously started.

In order to disable the transmission of an I-PDU, all I-PDU groups containing this I-PDU needs to be stopped. Hence, to implement listen-only-mode, all I-PDU groups containing transmission I-PDUs needs to be stopped. Receiving of I-PDUs may also be stopped.

[SWS_Com_00334] ⌈By a call to the functions: Com_SendSignal, Com_SendSignalGroup, or Com_InvalidateSignal, the AUTOSAR COM module shall update the values of its internal buffers even for stopped I-PDUs.⌋

See also Table 7.3.

If a signal written to a stopped I-PDU would trigger the transmission of this I-PDU if it were not stopped, then this trigger is not stored. After re-starting the corresponding I-PDU group, such an old trigger does not lead to an immediate transmission of the I-PDU.

[SWS_Com_00777] ⌈If an I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall cancel any remaining transmission requests for this I-PDU. This includes cancelling any potential retries with respect to ComRetryFailedTransmitRequests.⌋

[SWS_Com_00115] ⌈If an I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall cancel the deadline monitoring for all pending confirmations.⌋

[SWS_Com_00800] ⌈The AUTOSAR COM module shall ignore any transmit confirmations for a stopped I-PDU.⌋

The AUTOSAR COM module cannot prohibit the invocation of the Com_TriggerTransmit function. However, in case of a stopped I-PDU, the function Com_TriggerTransmit returns E_NOT_OK. See [SWS_Com_00884].

[SWS_Com_00684] ⌈If an I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall disable its reception processing.⌋

[SWS_Com_00713] ⌈If a large I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall stop the reception process and ignore the partly received I-PDU.⌋

[SWS_Com_00685] ⌈If an I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall cancel its deadline monitoring.⌋

[SWS_Com_00479] ⌈If an I-PDU is stopped by Com_IpduGroupStop, the AUTOSAR COM module shall immediately invoke the configured <ComUser_CbkTxErr> for signals / signal groups of the stopped I-PDU that were triggered by the user (e.g. RTE or SwCluC) but not yet confirmed.⌋

[SWS_Com_00714] ⌈If a large I-PDU is stopped while its transmission is already in progress, the AUTOSAR COM module shall stop the transmission process immediately.⌋

Table 7.3 gives an overview of the behavior of stopped I-PDUs:

Behavior of stopped I-PDUs	
Receiver side (RX)	**Transmitter side (TX)**
• Disable RX deadline monitoring • No action on a Com_RxIndication to users (e.g. RTE, SwCluC), no storing of the I-PDU • No action on Com_StartOfReception, Com_CopyRxData, Com_TpRxIndication to users (e.g. RTE, SwCluC), no storing of the I-PDU • Return code COM_SERVICE_NOT_AVAILABLE on Com_ReceiveSignal and Com_ReceiveSignalGroup and the last known value (or init value) is given back as data	• Disable sending • Disable TX deadline monitoring • Ignore Com_TxConfirmation • Ignore Com_TpTxConfirmation • Return code BUFREQ_E_NOT_OK on Com_CopyTxData • Return code E_NOT_OK on Com_Transmit • On a call of Com_SendSignal, Com_SendSignalGroup, Com_InvalidateSignal the values in the AUTOSAR COM module's internal buffers are still updated but the return code COM_SERVICE_NOT_AVAILABLE is returned • Remaining transmission requests (e.g. N-Times) will be cancelled • Return code E_NOT_OK on Com_TriggerTransmit
	For periodic (TX)
	Do not send any more

Table 7.3: Behavior of stopped I-PDUs

Table 7.4 gives an overview of the behavior of started I-PDUs:

Behavior of started I-PDUs	
Receiver side (RX)	**Transmitter side (TX)**
• Reinitialize timeouts if Initialize==true (ComFirstTimeout, ComTimeout) • Normal reaction on Com_RxIndication / Com_TpRxIndication • Normal reaction on Com_StartOfReception, Com_CopyRxData • Normal reaction on Com_ReceiveSignal, and Com_ReceiveSignalGroup	• Normal reaction on Com_InvalidateSignal, Com_SendSignal, and Com_SendSignalGroup • No transmission timeout notification until next send • Normal reaction on Com_CopyTxData • Normal reaction on Com_Transmit • Normal reaction on Com_TxConfirmation / Com_TpTxConfirmation • Normal reaction on Com_TriggerTransmit
	For periodic (TX)
	Start at 0

Table 7.4: Behavior of started I-PDUs

7.2.5.4 Signal Indication (Unpacking of I-PDUs)

In order to support both interrupt-driven and polled systems, it can be configured when the signal indication takes place. There are two configurable signal indication modes IMMEDIATE and DEFERRED configurable via ComIPduSignalProcessing.

[SWS_Com_00300] ⌈If ComIPduSignalProcessing for an I-PDU is configured to IMMEDIATE, the AUTOSAR COM module shall invoke the configured <ComUser_CbkRxAck> for the included signals and signal groups within the Com_RxIndication, or Com_TpRxIndication function respectively.⌋

[SWS_Com_00301] ⌈If ComIPduSignalProcessing for an I-PDU is configured to DEFERRED, the AUTOSAR COM module shall first copy the relevant data (e.g. configured ComSignal) of the received I-PDU within the Com_RxIndication function or the related TP reception functions respectively from the PduR into COM. Then the AUTOSAR COM module shall invoke the configured <ComUser_CbkRxAck> for the included signals and signal groups asynchronously during the next call to Com_MainFunctionRx.⌋

For signals with dynamic length, the AUTOSAR COM has to consider the configured length according to ComSignalLength for the buffer allocation.

If in DEFERRED mode a call to Com_ReceiveSignal is made before the deferred unpacking takes place, the previous not updated values are returned.

A sequence chart with both indication options can be found in Chapter 9.3. The configuration of these modes is defined in ComIPduSignalProcessing.

[SWS_Com_00574] ⌈When unpacking an I-PDU, the AUTOSAR COM module shall check the received data length (PduInfoPtr->SduLength) and unpack and notify only completely received signals via <ComUser_CbkRxAck>.⌋

[SWS_Com_00870] ⌈In case of receiving a smaller I-PDU than the length derived via the parameter ComPduIdRef, signals with size 0 shall be received if and only if their configured position is less or equal to the received size of the partially received I-PDU.⌋

[SWS_Com_00794] ⌈In case of receiving of a smaller I-PDU than expected results into receiving a signal without its configured update-bit, the AUTOSAR COM module shall treat this signal as if its update bit was set and interpret such a signal as updated.⌋

If the received I-PDU length is smaller than the configured / expected I-PDU length, it needs to be prevented that signals are updated partially. On the other hand all completely received signals should be received and notified to the upper layer.

[SWS_Com_00575] ⌈When unpacking an I-PDU, the AUTOSAR COM module shall check the received data length (PduInfoPtr->SduLength) and in case a signal group is received only partially, such a signal group and all included group signals shall not be unpacked or notified via <ComUser_CbkRxAck>.⌋

The above requirement prevents inconsistently received signal groups and therefore inconsistently received complex data types.

The AUTOSAR COM module does not copy or handle additional received data for not configured signals in case the received data length is greater than expected.

The AUTOSAR COM module only needs to copy and handle received data that is occupied by configured signals with respect to the expected length (expected length is the length of all configured dynamic or static signals including the gaps between the signals). Received data which exceeds the expected length does not need to be copied.

Note, Large Data Path provides an alternative efficient way, received PDU will be forwarded as a whole using different callbacks, see Chapter 7.5.3. In that case the AUTOSAR COM does not perform any length check, signal unpacking and copy actions.

7.2.5.5 Minimum Delay Timer (MDT)

The AUTOSAR COM module inherits the minimum delay timer mechanism as defined [2]. This mechanism is clarified and detailed in this chapter.

When an I-PDU is started, the MDT is re-initialized. Therefore, the MDT can be violated by stopping and starting I-PDUs rapidly.

The behavior of the transmission deadline monitoring timer is not affected by any transmission delay caused by the minimum delay time supervision.

[SWS_Com_00471] ⌈No minimum delay time monitoring shall take place, if `ComMinimumDelayTime` is omitted or configured to 0.⌋

[SWS_Com_00789] ⌈If `ComEnableMDTForCyclicTransmission` is configured to false, the minimum delay timer shall not be (re)started in the following cases:

- I-PDUs with `ComTxModeMode PERIODIC`
- the cyclic part of I-PDUs with `ComTxModeMode MIXED`
- repeated transmissions for I-PDUs in case `ComTxModeNumberOfRepetitions` > 0

⌋

In case the minimum delay timer is currently not started, there is no way for the minimum delay timer to expire. Thus, optimized implementations might for example omit monitoring the minimum delay time for periodic I-PDUs completely in case `ComEnableMDTForCyclicTransmission` is configured to false, since the minimum delay timer never gets started.

[SWS_Com_00698] ⌈AUTOSAR COM module shall load and start the minimum delay time counter upon transmission of that I-PDU to the PDU Router via `PduR_ComTransmit` if

- `ComMinimumDelayTime` is configured greater than 0 and
- `ComRetryFailedTransmitRequests` is set to false or `ComIPduLargeData` is set to true.

⌋

[SWS_Com_00828] ⌈AUTOSAR COM module shall load and start the minimum delay time counter upon transmission of that I-PDU to the PDU Router via `PduR_ComTransmit` only in case `PduR_ComTransmit` returns `E_OK` if

- `ComMinimumDelayTime` is configured greater than 0 and
- `ComRetryFailedTransmitRequests` is set to true and
- `ComIPduLargeData` is set to false.

⌋

[SWS_Com_00469] ⌈If `ComMinimumDelayTime` of an I-PDU is configured greater than 0, the AUTOSAR COM module shall (re-)load an already running and not yet elapsed minimum delay time counter of that I-PDU with `ComMinimumDelayTime` on a successful transmit confirmation.⌋

The running minimum delay timer is reloaded upon the reception of a successful TX-confirmation of that I-PDU, unless the transmission was already delayed longer than `ComMinimumDelayTime` at the reception of the confirmation. In normal case, there will be no further transmission of that I-PDU by the AUTOSAR COM module unless the loaded and started minimum delay has expired. See also Figures 5 (Direct Transmission Mode with minimum delay time), 6 (Periodic Transmission Mode) and 8 (Mixed Transmission Mode with minimum delay time (MDT delays PTR)) in [2]. However, some exception exists: According to [SWS_Com_00475] `Com_TriggerTransmit` does not interfere with the minimum delay timer. Further, the minimum delay timer is reset if the transmission deadline monitoring timer expires; see Clause 3.3.4 (Transmission modes) of [2]. In addition, starting an I-PDU group resets the minimum delay time timer of the included I-PDUs.

[SWS_Com_00812] ⌈If the minimum delay timer of an I-PDU is reset because the transmission deadline monitoring timer expires and a postponed request for this I-PDU exists, the AUTOSAR COM module shall immediately transmit this I-PDU.⌋

[SWS_Com_00906] Large Data Path Minimum Delay Timer ⌈If `Com_Transmit` is called for I-PDU with `ComIPduLargeData` set to true while the minimum delay timer is running then `Com_Transmit` shall directly return COM_BUSY.⌋

If ComIPduLargeData is set to false and minimum delay time counter is running then new transmission is delayed as defined in [2].

7.2.6 Deadline Monitoring

Deadline monitoring for signals is defined in [2].

In the context of deadline monitoring for a signal group, it is handled like a signal. The deadline monitoring parameters ComFirstTimeout and ComTimeout can be defined in the configuration container ComSignal or ComSignalGroup.

The corresponding timeout notification callback functions can be defined in parameter in the configuration container ComUserCallback.

[SWS_Com_00333] ⌈If the configuration parameter ComTimeout for a signal or signal group is omitted or configured to 0, the AUTOSAR COM module shall not monitor this signal or signal group. It also shall ignore the ComFirstTimeout.⌋

7.2.6.1 Reception Deadline Monitoring

The reception deadline monitoring can be enabled and disabled by the control of I-PDU groups analogous to Chapter 7.2.5.1.

[SWS_Com_00772] ⌈The reception deadline monitoring of an I-PDU is enabled if and only if it is contained in an I-PDU group that has reception deadline monitoring enabled. Otherwise, the reception deadline monitoring of the I-PDU is disabled.⌋

[SWS_Com_00292] ⌈In the case where reception deadline monitoring is configured for signals with update-bits, the AUTOSAR COM module shall perform a separate reception deadline monitoring for each signal / signal group with an update-bit. For configuration, see ComTimeout.⌋

[SWS_Com_00290] ⌈The AUTOSAR COM module shall perform an I-PDU based reception deadline monitoring for signals without an update-bit.⌋

[SWS_Com_00291] ⌈For all signals and signal groups without update-bits within the same I-PDU, the AUTOSAR COM module shall perform the reception deadline monitoring of the I-PDU using the smallest configured non-zero timeout parameter (ComFirstTimeout, ComTimeout) of all these signals and signal groups which do not have an update-bit configured.⌋

If all signals within an I-PDU with configured reception deadline monitoring have also an update-bit configured, no reception deadline monitoring on I-PDU base needs to be performed.

In case of an Rx-timeout, the ComRxDataTimeoutAction parameter determines whether the AUTOSAR COM module replaces the signal / signal group value with the initial value or substitute value or maintains the last received value.

[SWS_Com_00470] ⌈If ComRxDataTimeoutAction is set to REPLACE, the AUTOSAR COM module shall replace the signal's value by its ComSignalInitValue when the reception deadline monitoring timer of a signal expires.⌋

In case the ComSignalInitValue replaces the last received value, the last received value is overwritten and gets lost. Therefore, the AUTOSAR COM module returns the ComSignalInitValue for this signal until a new value is received.

[SWS_Com_00875] ⌈If ComRxDataTimeoutAction is set to SUBSTITUTE, the AUTOSAR COM module shall replace the signal's value by its ComTimeoutSubstitutionValue when the reception deadline monitoring timer of a signal expires.⌋

[SWS_Com_00500] ⌈If the reception deadline monitoring timer of a signal or signal group expires and the configuration parameter ComRxDataTimeoutAction for this signal or signal group is either omitted or configured to NONE, the AUTOSAR COM module shall not replace the signal or the signals of the signal groups respectively.⌋

[SWS_Com_00513] ⌈If ComRxDataTimeoutAction is set to REPLACE, the AUTOSAR COM module shall replace the values of all included group signals by their ComSignalInitValue, when the reception deadline monitoring timer of a signal group expires.⌋

The Rx-timeout-indication can be combined and configured separately from ComRxDataTimeoutAction.

[SWS_Com_00876] ⌈If ComRxDataTimeoutAction is set to SUBSTITUTE, the AUTOSAR COM module shall replace the values of all included group signals by their ComTimeoutSubstitutionValue, when the reception deadline monitoring timer of a signal group expires.⌋

[SWS_Com_00715] ⌈When a (large) I-PDU with a configured reception deadline monitoring is received successfully, the AUTOSAR COM module shall reset the reception deadline monitoring timer for this (large) I-PDU at invocation of the function Com_RxIndication or Com_TpRxIndication respectively.⌋

[SWS_Com_00716] ⌈If the configuration parameter ComFirstTimeout for a signal or signal group is omitted or configured to 0, the AUTOSAR COM module shall not monitor the reception of this signal or signal group respectively from the start of the corresponding I-PDU until the first reception.⌋

[SWS_Com_00738] ⌈The I-PDU based reception deadline monitoring mechanism shall not take the values of the signals into account.⌋

[SWS_Com_00889] ⌈If reception deadline monitoring timer expires, AUTOSAR Com module shall restart it.⌋

AUTOSAR COM follows the approach in Clause 3.5.1 (Reception Deadline Monitoring) of [2], i.e. reception deadline monitoring timer is restarted. This also means that timeout notification is periodically reported if reception fails for a longer period.

7.2.6.1.1 En-/Disable Reception Deadline Monitoring

When reception deadline monitoring of an I-PDU is disabled and the timer expires, no error indication will be given to the users (e.g. RTE, SwCluC).

Disabling reception deadline monitoring does not stop the reception of an I-PDU.

[SWS_Com_00224] ⌈If the reception deadline monitoring of an I-PDU is enabled by `Com_EnableReceptionDM`, the AUTOSAR COM module shall set the reception deadline monitoring timer for the included signals and signal groups to the configured `ComFirstTimeout` value.⌋

[SWS_Com_00486] ⌈The AUTOSAR COM module shall silently ignore setting the reception deadline monitoring of an I-PDU to enabled by `Com_EnableReceptionDM`, in case the reception deadline monitoring is already enabled for this I-PDU.⌋

Enabling reception deadline monitoring implies that timeout notifications of deadline monitoring expiry are notified to the users (e.g. RTE, SwCluC) for signals or signal groups with a configured `<ComUser_CbkRxTOut>`.

[SWS_Com_00534] ⌈If `Com_EnableReceptionDM` or `Com_DisableReceptionDM` is invoked on an I-PDU group containing Tx-I-PDUs, then the AUTOSAR COM module shall silently ignore this request.⌋

[SWS_Com_00225] ⌈The AUTOSAR COM module shall silently ignore setting the reception deadline monitoring of an I-PDU to disabled by `Com_DisableReceptionDM`, in case the reception deadline monitoring is already disabled for this I-PDU.⌋

7.2.6.2 Transmission Deadline Monitoring

The general idea of the AUTOSAR COM transmission deadline monitoring is to supervise the lower layers and the bus but not the COM module itself. Hence, the transmission monitoring timer is generally started, when the COM module sends an I-PDU to the lower layer. This is independent of a transmission mode, a transfer property or update-bits. In generally the transmission monitoring timer is not re-started or reset if it is currently running.

[SWS_Com_00481] ⌈The AUTOSAR COM module shall perform the transmission deadline monitoring (if configured) for all signals and signal groups independently from

the transfer property and independently from the transmission modes of the I-PDU the signal / signal group belongs to.⌋

The AUTOSAR COM module makes no distinction between signals with ComTransferProperty PENDING or TRIGGERED with respect to transmission deadline monitoring.

[SWS_Com_00445] ⌈If different ComTimeout parameters of the associated signals / signal groups of an I-PDU are configured, the AUTOSAR COM module shall use the smallest value as timeout parameter for the transmission deadline monitoring of the I-PDU.⌋

Transmission deadline monitoring should only be configured in the AUTOSAR COM module for busses that support the generation of transmit confirmations. Otherwise, the transmission deadline monitoring would always notify a transmission error.

[SWS_Com_00878] ⌈The AUTOSAR COM shall start a configured transmission deadline monitoring timer of a signal (group) if it is sent (within an I-PDU) to the lower layer, unless the timer is already running.⌋

[SWS_Com_00878] does not consider a potential return code, thus the timer is started even if the sent request fails immediately.

If the timer is already running a new send request does not reset or restart a running timer, but the previous transmission is monitored further on. Unless otherwise specified, the timer is started regardless of the trigger that lead to a potential I-PDU send request. Such triggers are for example a cyclic send request of a signal, a Transmission Mode Switch, or an explicit I-PDU sent request via Com_TriggerIPDUSend.

[SWS_Com_00879] ⌈The transmission deadline monitoring timer shall be started with the configured ComFirstTimeout value if the timer is started for the first time after a (re-)start of the transmission deadline monitoring service for this I-PDU, otherwise the timer shall be started with ComTimeout value.⌋

[SWS_Com_00880] ⌈When the AUTOSAR COM receives a successful transmit confirmation (result E_OK) for an I-PDU, it shall cancel all running transmission deadline monitoring timers for all contained signals and signal groups.⌋

[SWS_Com_00696] ⌈In case transmission deadline monitoring is configured for an I-PDU with transmission mode NONE only, the AUTOSAR COM module shall start the transmission deadline monitoring timer for this I-PDU upon the start of the I-PDU group to which the I-PDU belongs to.⌋

[SWS_Com_00835] ⌈In case transmission deadline monitoring is configured for an I-PDU with transmission mode NONE and another transmission mode, the transmission deadline monitoring shall be disabled whenever the transmission mode NONE is active.⌋

[SWS_Com_00697] ⌈In case transmission deadline monitoring is configured for an I-PDU with transmission mode NONE only, the AUTOSAR COM module shall reset the transmission deadline monitoring timer for this I-PDU when a successful transmit confirmation is received.⌋

In case of a signal group, it is only possible to configure transmission deadline monitoring for the whole signal group and not for group signals, see ComSignalGroup and ComGroupSignal.

[SWS_Com_00708] ⌈In case the transmission deadline monitoring timer expires for an I-PDU with ComIPduCancellationSupport configured to true, the AUTOSAR COM module shall invoke PduR_ComCancelTransmit for that I-PDU.⌋

The invocation of PduR_ComCancelTransmit is done in addition to the normal timeout handling of Com, which notifies the SW-C, if configured.

The AUTOSAR COM ignores the return code of PduR_ComCancelTransmit.

[SWS_Com_00304] ⌈When a transmission deadline monitoring timer elapses and there was no successful transmit confirmation for an I-PDU in time, the AUTOSAR COM module shall first notify the users (e.g. RTE, SwCluC) by invoking all configured <ComUser_CbkTxTOut> for contained signals or signal groups. Afterwards, the AUTOSAR COM shall notify Dem via Dem_SetEventStatus API with DEM_EVENT_STATUS_PREFAILED if COM_E_TIMEOUT is configured for contained signals or signal groups. Besides, transmission deadline monitoring timer shall not be restarted.⌋

If the transmission deadline monitoring timer runs out, there will be a timeout notification regardless of the reason. For example, the notification will even take place, if the transmission was filtered out by an I-PDU callout.

AUTOSAR COM follows the approach in Clause 3.5.2 (Transmission Deadline Monitoring) of [2], i.e. transmission deadline monitoring timer is not restarted. This also means that timeout notification is reported only once if the transmission fails.

7.2.6.2.1 Transmission Deadline Monitoring with N-Times Transmission Mode

As defined in [2] the monitoring timer has to be started upon completion of a call to Com_SendSignal or Com_SendSignalGroup respectively if transmission deadline monitoring is configured for the corresponding signal or signal group respectively.

For the transmission mode DIRECT and MIXED, it should be ensured that all ComTxModeNumberOfRepetitions requests could be made within the configured period (see Chapter 7.2.3.4).

As defined in [2], if the monitoring timer expires the users (e.g. RTE, SwCluC) are notified with the configured notification mechanism about that failure.

[SWS_Com_00308] ⌈For an I-PDU with ComTxModeMode DIRECT and ComTxModeNumberOfRepetitions > 0, the AUTOSAR COM module shall cancel the transmission deadline monitoring timer after ComTxModeNumberOfRepetitions plus one successfully received transmit confirmations.⌋

If the timer is cancelled after ComTxModeNumberOfRepetitions plus one confirmations, the transmission was successful and then the transmission confirmation is sent to the users (e.g. RTE, SwCluC). See also [SWS_Com_00305].

[SWS_Com_00739] ⌈For an I-PDU with ComTxModeMode DIRECT and ComTxModeNumberOfRepetitions > 0, the AUTOSAR COM module shall reset an already running transmission deadline monitoring timer in case another send request for this I-PDU is initiated.⌋

7.2.7 Notification handling

The AUTOSAR COM inherits the Notification Classes 1, 2, 3 and 4 in Clause 3.6.2 (Notification classes) of [2]. The notifications are configurable via ComUserCallback and the definitions of the callbacks are given in Chapter 8.5.3.

[SWS_Com_00883] ⌈For each <ComUser_CbkTxAck> / <ComUser_CbkRxAck> that is configured for a signal or signal group, the AUTOSAR COM shall indicate each successful transmission / reception via the configured notifications.⌋

For example, for periodic transmissions each successful periodic transmission is notified. Note that for n-times transmission multiple transmissions on the bus are necessary before it is considered as successful, see also [SWS_Com_00305].

7.3 Signal Groups - Complex Data Types

To support the AUTOSAR concept of complex data types the AUTOSAR COM module provides signal groups. The AUTOSAR COM module transmits and receives signal groups consistently to provide the necessary consistency for complex data types.

Signal groups can be configured statically. For each signal group a symbolic name can be configured. See ComSignalGroup for the configuration details.

The AUTOSAR COM module achieves the consistency of a signal group by means of a shadow buffer mechanism, i.e. the user (e.g. RTE or SwCluC) accesses the group signals in the shadow buffer. If the shadow buffer needs to be synchronized with the I-PDU, the users can trigger this explicitly with Com_SendSignalGroup or Com_ReceiveSignalGroup. The synchronization is performed atomically.

An alternative approach to signal group consistency is to enable the signal group array APIs Com_SendSignalGroupArray and Com_ReceiveSignalGroupArray. The deviations of this approach are described in Chapter 7.3.6.

7.3.1 Initialization

[SWS_Com_00484]

Upstream requirements: SRS_BSW_00101

⌈By a call to `Com_Init`, the AUTOSAR COM module shall initialize the shadow buffer of a signal group on sender-side.⌋

Since it is not suspected that a well-formed SWC tries to read a group signal before a call to `Com_ReceiveSignalGroup`, [SWS_Com_00484] applies to the sender side only.

7.3.2 Transmission

If `Com_SendSignal` or `Com_InvalidateSignal` is called for a signal that belongs to a signal group, then the AUTOSAR COM will only update the shadow buffer of this signal group. There is no need for any further I-PDU processing like TMS evaluation, unless the I-PDU contents changed.

[SWS_Com_00050] ⌈If `Com_SendSignalGroup` is called for the signal group, the AUTOSAR COM module shall copy the shadow buffer atomically to the I-PDU buffer.⌋

Example with two group signals signal_a and signal_b, which belong to group_x:

- /* copy a to shadow buffer */
- `Com_SendSignal` (signal_a, &a);
- /* copy b to shadow buffer */
- `Com_SendSignal` (signal_b, &b);
- /* copy shadow buffer to I-PDU */
- `Com_SendSignalGroup` (group_x);

7.3.3 Reception

A group signal can be received from the shadow buffer by calling the function `Com_ReceiveSignal` after the signal group data has been copied to the shadow buffer by `Com_ReceiveSignalGroup`.

[SWS_Com_00051] ⌈If `Com_ReceiveSignalGroup` is called for a signal group, the AUTOSAR COM module shall copy the data atomically from the I-PDU buffer to the shadow buffer.⌋

Example with two group signals signal_a and signal_b, which belong to group_x:

- /* copy I-PDU to shadow buffer */

- Com_ReceiveSignalGroup (group_x);
- /* copy a from shadow buffer */
- Com_ReceiveSignal (signal_a, &a);
- /* copy b from shadow buffer */
- Com_ReceiveSignal (signal_b, &b);

7.3.4 Notifications

Table 7.5 shows that all notification functions are only configurable for signal groups but not for individual group signals. Chapter 7.2 defines the functional logic of the notification methods for signals and signal groups. The prototypes for the configurable notification functions are defined in Chapter 8.5.3.

7.3.5 Attributes of a Signal Group

Table 7.5 gives an overview of the attributes of a signal group:

Attribute	Per group signal	Per signal group
Update-bit	No	Yes, associated on the whole group (see Chapter 7.10])
Signal Notification (sender side)	No	Yes
Signal Notification (receiver side)	No	Yes
Error Notification (sender side)	No	Yes
Timeout Notification (sender side)	No	Yes
Timeout Notification (receiver side)	No	Yes
Invalid Notification (receiver side)	No	Yes
Data access (receiver side)	Yes	Yes, see [SWS_Com_00201]
Data access (sender side)	Yes	Yes, see [SWS_Com_00200], [SWS_Com_00557]
Data Filtering (receiver side)	Yes, see Chapter 7.1.4	No
Data Filtering (sender side)	No	No
TMS on sender side	Each signal, according to TMS selection definition. (see Chapter 7.2.3.2)	No

Table 7.5: Attributes of signal groups

7.3.6 UINT8-array based access to signal groups

An alternative approach to signal group consistency is to enable the signal group array APIs Com_SendSignalGroupArray and Com_ReceiveSignalGroupArray via the COM configuration parameter ComSignalGroupArrayAccess. The serialization of the composite data is then done outside of the COM module but resulting in the same layout as COM would use to place the group signals in the I-PDU.

In this approach, no shadow buffer is used. The serialized data is directly provided to the COM module as a uint8-array representation of the signal group.

[SWS_Com_00841] ⌈The UINT8-array based access to signal groups shall only be used if the following preconditions apply:

- uses only fix sized data types for the composite data
- signal groups, which are mapped byte aligned to the I-PDU
- signal groups, where the underlying byte array does not contain any other signals, but may contain gaps.

⌋

[SWS_Com_00842] ⌈In case a signal group has `ComSignalGroupArrayAccess` defined, no shadow buffer shall be allocated for that signal group.⌋

[SWS_Com_00843] ⌈In case a signal group has `ComSignalGroupArrayAccess` defined, the call to the following APIs is not supported for that signal group:

- `Com_SendSignal` and `Com_ReceiveSignal` for any of the group signals of that signal group
- `Com_SendSignalGroup` and `Com_ReceiveSignalGroup` for that signal group

⌋

The above requirement defines that certain APIs are not supported for signal groups with array access. That is, these APIs must not be called for signal groups with array access. If they are called illegally, the behavior is undefined.

Since the signal group is provided in the uint8-array representation, reception filtering and transmission mode selection needs to cope with this data representation when evaluating the group signals.

In order to ease the filter evaluation only a subset of the filters from [SWS_Com_00602] is supported.

For signal group array access the following two filters are excluded from reception filtering and Transmission Mode Selection:

- `NEW_IS_WITHIN`
- `NEW_IS_OUTSIDE`

[SWS_Com_00848] ⌈The signal group array processing functions `Com_SendSignalGroupArray` and `Com_ReceiveSignalGroupArray` shall support the following filters for reception filtering and Transmission Mode Selection:

- `ALWAYS`

- NEVER
- MASKED_NEW_EQUALS_X
- MASKED_NEW_DIFFERS_X
- MASKED_NEW_DIFFERS_MASKED_OLD
- ONE_EVERY_N

⌋

7.3.6.1 Basic functionality

In case the AUTOSAR Transformer approach is used the serialization and further data processing is already done in the transformer chain. Since the transformer chain already deals with a uint8-array representation of the communication data, the uint8-array based access to signal groups allows to use the already serialized data and to place it into the I-PDU without the need for COM to serialize the data again.

The AUTOSAR System Template [11] defines which signal group shall be handled in the uint8-array based approach. If the System Template enables the uint8-array based approach the Ecu configuration parameter ComSignalGroupArrayAccess is set to true.

[SWS_Com_00844] ⌈The start position of the uint8-array representation of the communication data shall be the byte offset of the group signal in this signal group with the smallest ComBitPosition.⌋

[SWS_Com_00845] ⌈The end position of the uint8-array representation of the communication data shall be the byte offset of the last byte occupied by the group signal in this signal group with the highest ComBitPosition.⌋

7.3.6.2 Initialization

[SWS_Com_00850]

Upstream requirements: SRS_BSW_00101

⌈By a call to Com_Init, the AUTOSAR COM module shall initialize the respective I-PDU buffer section of a signal group, that has ComSignalGroupArrayAccess configured true, based on the ComSignalInitValue of all included group signals.⌋

7.3.6.3 Transmission

[SWS_Com_00846] ⌈When `Com_SendSignalGroupArray` is called a data pointer to the uint8-array representation of the signal group is provided. The AUTOSAR COM module shall copy the data into the I-PDU based on the start position defined in [SWS_Com_00844] and the size defined in [SWS_Com_00845].⌋

[SWS_Com_00847] ⌈The AUTOSAR COM module shall handle a call of `Com_SendSignalGroupArray` equally to `Com_SendSignalGroup` with respect to transfer properties and I-PDU transmission mode.⌋

See Chapters 7.2.3.1.2, 7.2.3.2, 7.2.3.3.

7.3.6.4 Reception

[SWS_Com_00849] ⌈When `Com_ReceiveSignalGroupArray` is called a data pointer to the uint8-array representation of the signal group is provided. The AUTOSAR COM module shall copy the data to the data pointer position of the I-PDU based on the start position defined in [SWS_Com_00844] and the size defined in [SWS_Com_00845].⌋

7.4 Large Data Types

A large signal is a signal that is too large to fit into a single L-PDU of the underlying communication protocol. For example, a large signal on CAN would exceed 8 bytes or 64 bytes for CAN FD [12]. For FlexRay the situation is more complex, since one frame can contain several I-PDUs. Hence, a signal may be a large signal for FlexRay even if it does not exceed the size of a FlexRay frame.

Large signals in AUTOSAR require configuring a large I-PDU that will be transmitted via the transport protocol of the underlying bus. It is not imperative that a large I-PDU contains one or more large signals.

Using this concept the I-PDU length and hence the signal size will is limited according to the bus-specific transport protocol, see `ComSignalLength`.

`UINT8_N` is the only signal type that supports signals of a static size larger than 8 bytes. For dynamic length signals see Chapter 7.7.

Since the AUTOSAR COM module is not aware of the underlying bus properties, it has to be configured if an I-PDU can be transmitted within a single L-PDU, e.g. one CAN frame, or if it needs to be transmitted via TP, see also `ComIPduType`.

The AUTOSAR COM module is not able to prevent or to detect if too large I-PDUs are sent to a specific bus transport protocol, because the AUTOSAR COM module uses the bus-independent PDU Router interface. However, the AUTOSAR COM module allows

that the maximum length of the N-SDU of the underlying TPs can be used and therefore the AUTOSAR COM module does not introduce any additional length restrictions.

7.4.1 Transmission of Large Signals / I-PDUs

The AUTOSAR COM module sends large I-PDUs via the generic PduR APIs for upper layer modules that use transport protocol. For details of the AUTOSAR COM module - PDU Router interface see Chapter 7.9. According to [5] the call-sequence is:

- `PduR_ComTransmit`: for initiating the send request
- one or more callbacks to `Com_CopyTxData`: for copying the data
- callback to `Com_TpTxConfirmation`: for confirming the transmission

[SWS_Com_00662] ⌈After successful initiation of a transmission of a large I-PDU via `PduR_ComTransmit`, the AUTOSAR COM module shall not modify the data of its internal transmit buffer of this I-PDU unless the AUTOSAR COM module is notified about either the successful transmission or an error indicating that the transmission was aborted.⌋

The I-PDUs within the AUTOSAR COM module are statically configured. Therefore, the complete memory of the I-PDU, even for very large I-PDUs, will be allocated. This approach was taken in order to reduce the complexity of the large data type handling.

If the utilized transportation protocol is also used for diagnosis, or other services, it must support multiple users. This is at least supported by CAN TP [13] and FlexRay TP [14].

[SWS_Com_00863] ⌈In case a large I-PDU is currently transmitted and the same I-PDU is triggered for transmission again, the AUTOSAR COM shall skip the additionally send request and report the runtime error `COM_E_SKIPPED_TRANSMISSION`.⌋

This could for example happen in case a large I-PDU is sent out periodically and the transmission is delayed over the next cycle.

7.4.2 Reception of Large Signals / I-PDUs

The AUTOSAR COM module receives large I-PDUs via the generic PDU Router's APIs for upper layer modules that use transport protocol. For details of the AUTOSAR COM module - PDU Router interface see Chapter 7.9. According to [5] the call-sequence is:

- call to `Com_StartOfReception`: for initiating the RX request
- one or more callbacks to `Com_CopyRxData`: copying the receive data to the AUTOSAR COM module's internal receive buffer

- call to Com_TpRxIndication: indicating the correct, or incorrect end of the reception process

[SWS_Com_00838] ⌈In case Com_TpRxIndication is called with value E_NOT_OK for the parameter result, the AUTOSAR COM module shall treat all signals of the received large I-PDU exactly like invalidated signals. That is, the AUTOSAR COM module shall execute all configured ComDataInvalidAction for all included signals and signal groups.⌋

When receiving large I-PDUs the internal I-PDU buffer is overwritten gradually. Thus, unless the old signal is not stored somewhere else (for example in a shadow buffer of a signal group), the COM module runs the risk to lose signal data information that might be needed in certain scenarios, for example:

- receiving a signal with an unset update-bit
- providing the last value in case the RX-filter is not passed
- providing the last value in case there is only an invalid notification (but no replacement) is defined and the invalid value is received

[SWS_Com_00882] ⌈If the AUTOSAR COM module receives signals via large I-PDUs and these signals have features configured that need to keep the old value in certain cases, the AUTOSAR COM module shall correctly provide the old value for signals received via a large I-PDU in the same way as for signals received via normal I-PDUs.⌋

It is open to the implementation to find efficient means for providing the old values and to derive from the configuration, which signals or I-PDUs need an additional handling (e.g. buffering).

This does not mean it is necessary to implement a queue.

7.5 Large Data Path Support

The AUTOSAR COM module provides an alternative Interaction Layer Mechanism. By focusing on spontaneous, non-cyclic communication without serializing, filtering and conversion an efficient implementation can be achieved for selected I-PDUs.

Provided features are the following:

- Provision of signal oriented data interface to users (RTE, SwCluC_ComProxy)
- Provision of received signals to users (RTE, SwCluC_ComProxy)
- Support of large and dynamic length data types
- Support of IF- and TP-based communication
- Transmission and reception side deadline monitoring and notification
- Guarantee of minimum distances between transmit I-PDUs

- Communication control (start / stop of I-PDU groups)

See also parameters ComLargeDataSupport, ComIPduLargeData and Chapter 10.3.6 for limitations.

7.5.1 Overall

See Table 7.6: API to Parameter mapping for a mapping of Large Data Path relevant API names to the ECUC Parameter containing the actual name configured for this API per signal. The Com user callback handle Id (ComUserCbkHandleId) parameter identifies the corresponding signal/I-PDU.

API Name	ECUC Parameter
<ComUser_ComCbkCopyTxData>	ComUserCallbackName with ComUserCallbackType set to COM_TP_COPY_TX_DATA
<ComUser_ComCbkTpTxConfirmation>	ComUserCallbackName with ComUserCallbackType set to COM_TP_TX_CONFIRMATION
<ComUser_ComCbkRxIndication>	ComUserCallbackName with ComUserCallbackType set to COM_RX_INDICATION
<ComUser_ComCbkStartOfReception>	ComUserCallbackName with ComUserCallbackType set to COM_RX_START_OF_RECEPTION
<ComUser_ComCbkCopyRxData>	ComUserCallbackName with ComUserCallbackType set to COM_TP_COPY_RX_DATA
<ComUser_ComCbkTpRxIndication>	ComUserCallbackName with ComUserCallbackType set to COM_TP_RX_INDICATION
<ComUser_ComCbkTriggerTransmit>	ComUserCallbackName with ComUserCallbackType set to COM_TX_TRIGGER_TRANSMIT
<ComUser_ComCbkTxConfirmation>	ComUserCallbackName with ComUserCallbackType set to COM_TX_CONFIRMATION

Table 7.6: API to Parameter mapping (Large Data Path)

Even if the concept of Com user provides a lot of flexibility to support access by multiple users including their notifications some limitations needs to be considered.

In general, multiple writers can cause race conditions if the writers are not coordinated. In addition, neither the behavior of TriggerTransmit interfacing nor the TP interfacing does support notification towards multiple users for the same I-PDU.

[SWS_Com_CONSTR_00911] Large Data Path I-PDU User Configuration

Previous identifiers: SWS_LdCom_CONSTR_00001

⌈Sent I-PDUs with ComIPduLargeData set to true shall be owned by at most one Com user.⌋

Nevertheless, reading an I-PDU by several Com users in the same or different Software Clusters is possible but the partititon assignment of the I-PDUs needs to be respected.

[SWS_Com_CONSTR_00912] Large Data Path I-PDU User EcucPartition

Previous identifiers: SWS_LdCom_CONSTR_00002

⌈All Com users registering notifications for I-PDUs with `ComIPduLargeData` set to true shall reside on the EcucPartition on which the COM module handles the related I-PDU.⌋

[SWS_Com_00913] Large Data Path Excluded API Functions ⌈In case a signal is contained by `ComIPdu` with `ComIPduLargeData` set to true, the call to the following APIs is not supported for that signal:

- `Com_SendSignal`
- `Com_SendDynSignal`
- `Com_ReceiveSignal`
- `Com_ReceiveDynSignal`
- `Com_ReceiveSignalWithMetaData`
- `Com_ReceiveDynSignalWithMetaData`
- `Com_SendSignalWithMetaData`
- `Com_SendDynSignalWithMetaData`

⌋

Large Data Path avoids buffering in COM, therefore signals cannot be accessed with the functions above.

7.5.2 Transmission

If `ComIPduLargeData` is set to true, transmission is initiated by the Com user (e.g. RTE, SwCluC_ComProxy) by invoking `Com_Transmit` or PduR (invoking `Com_TriggerTransmit`) but not by COM module on its own.

[SWS_Com_00914] Large Data Path IF Transmission

Previous identifiers: SWS_LdCom_00010, SWS_LdCom_00012

⌈When `Com_Transmit` is invoked, AUTOSAR COM module shall call `PduR_ComTransmit` if the I-PDU is started and minimum delay time counter is not running.⌋

7.5.2.1 IF

[SWS_Com_00915] Large Data Path IF TriggerTransmit

Previous identifiers: SWS_LdCom_00060

⌈When Com_TriggerTransmit is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkTrigger-Transmit> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkTriggerTransmit>, see [SWS_Com_CONSTR_00911].

[SWS_Com_00916] Large Data Path IF TxConfirmation

Previous identifiers: SWS_LdCom_00061

⌈When Com_TxConfirmation is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (Com-IPduHandleId) to derive the corresponding <ComUser_ComCbkTxConfirmation> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkTxConfirmation>, see [SWS_Com_CONSTR_00911].

The Large Data Path avoids buffering in COM; therefore, each transmit I-PDU with ComIPduType NORMAL and ComIPduLargeData enabled can benefit from asynchronous copy with hardware support, i.e. DMA transfer.

In this case, lower layer indicates via Com_TxConfirmation that the transmitted data has been copied.

7.5.2.2 TP

[SWS_Com_00917] Large Data Path TP Transmission

Previous identifiers: SWS_LdCom_00063

⌈When Com_CopyTxData and Com_TpTxConfirmation are invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkCopyTxData> or <ComUser_ComCbkTpTxConfirmation> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkCopyTxData> or <ComUser_ComCbkTpTxConfirmation>, see [SWS_Com_CONSTR_00911].

7.5.3 Reception

7.5.3.1 IF

[SWS_Com_00918] Large Data Path IF Reception

Previous identifiers: SWS_LdCom_00064

⌈When Com_RxIndication is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkRxIndication> user notification callbacks and call them with the according Com user callback handle Id.⌋

The Large Data Path avoids buffering in COM; therefore, each receive I-PDU with ComIPduType NORMAL and ComIPduLargeData enabled can benefit from asynchronous copy with hardware support, i.e. DMA transfer.

In this case, upper layer indicates via Com_ReleaseRxBuffer that the received data has been copied.

[SWS_Com_00950] Large Data Path IF Reception with DMA Copy Approach
⌈When Com_ReleaseRxBuffer is invoked, AUTOSAR COM module shall call PduR_ComReleaseRxBuffer.⌋

7.5.3.2 TP

[SWS_Com_00919] Large Data Path TP StartOfReception

Previous identifiers: SWS_LdCom_00065

⌈When Com_StartOfReception is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkStartOfReception> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkStartOfReception>.

[SWS_Com_00920] Large Data Path TP CopyRxData

Previous identifiers: SWS_LdCom_00066

⌈When Com_CopyRxData is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as Handle Id (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkCopyRxData> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkCopyRxData>.

[SWS_Com_00921] Large Data Path TP TpRxIndication

Previous identifiers: SWS_LdCom_00067

⌈When Com_TpRxIndication is invoked for an I-PDU with ComIPduLargeData set to true, AUTOSAR COM module shall use the passed I-PDU ID as (ComIPduHandleId) to derive the corresponding <ComUser_ComCbkTpRxIndication> user notification callback and call it with the according Com user callback handle Id.⌋

Only a single Com user can be notified with <ComUser_ComCbkTpRxIndication>.

7.6 Asynchronous Copy of I-PDU Data

See also [4, BSW General] Chapter 7.1.21 *"Asynchronous copy approach (DRAFT)"* for more details.

[SWS_Com_00951] Transmit Buffer Alignment for Asynchronous Copy ⌈Whenever COM needs to transmit a buffered I-PDU, and the PDU referenced by ComPduIdRef has KeepLocalPduBuffer set to TRUE, COM shall provide the data at a memory address that complies with the configured PduBufferAlignment of that PDU (see [15]).⌋

[SWS_Com_00952] Release Transmit Buffer ⌈If transmission of a buffered I-PDU is requested by a call to PduR_ComTransmit, and the PDU referenced by ComPduIdRef has KeepLocalPduBuffer set to TRUE, COM shall release the I-PDU buffer upon the transmit confirmation for that I-PDU or when the I-PDU is stopped.⌋

[SWS_Com_00953] Error Reporting when I-PDU Stopped ⌈If a buffered I-PDU is stopped before transmit confirmation, and the PDU referenced by ComPduIdRef has KeepLocalPduBuffer set to TRUE, COM shall report COM_E_IPDU_LOST.⌋

[SWS_Com_00954] Asynchronous Copy to Receive Buffer ⌈If an I-PDU that needs to be buffered is received, and the length of the I-PDU exceeds the configured EcucHardware.EcucDmaThreshold, and the PDU referenced by ComPduIdRef has KeepLocalPduBuffer set to TRUE, COM shall initiate an asynchronous copy with hardware support (i.e. DMA transfer) to transfer the I-PDU data to the local I-PDU buffer.⌋

[SWS_Com_00955] Release Receive Buffer ⌈If an asynchronous copy with hardware support (i.e. DMA transfer) for an I-PDU that needs to be copied to a local buffer (see [SWS_Com_00954]) has completed, regardless of whether successfully or unsuccessfully, COM shall call PduR_ComReleaseRxBuffer.⌋

[SWS_Com_00956] Error Reporting when Asynchronous Copy Fails ⌈Whenever an asynchronous copy with hardware support (i.e. DMA transfer) fails, COM shall report the runtime error COM_E_IPDU_LOST.⌋

7.7 Dynamic Length Signals

A dynamic length signal is a signal which length can vary at run-time. The maximal length of a dynamic length signal needs to be specified at configuration time. A dynamic length signal has to be transmitted in a dynamic length I-PDU.

[SWS_Com_00753] ⌈For dynamic length signals, the AUTOSAR COM module shall only support the `ComSignalType UINT8_DYN`.⌋

Restricting the type to a UINT8-array type also restricts the placement of the dynamic length signal to byte boundaries.

All other signals, potentially including large signals, must be packed in front of the dynamic length signal (see [SWS_Com_CONSTR_00754], [SWS_Com_CONSTR_00755] and [SWS_Com_CONSTR_00756]). Therefore, it is not required to encode the length of the dynamic length signal within the I-PDU. On receiver side, the length of the dynamic length signal can be calculated from the length of the received I-PDU.

It is allowed to configure an update-bit for a dynamic length signal. In this case, the update-bit must be located in front of the dynamic length signal, see also [SWS_Com_CONSTR_00755].

There is no restriction in the AUTOSAR COM module preventing the usage of signal groups in combination with large or dynamic length signals.

Dynamic length signals are not restricted to be sent via transport protocol. However, the maximum length of the I-PDU containing the dynamic length signal must respect the bus-specific restrictions.

In case of I-PDUs with dynamic length signals that shall be sent via CAN FD [12] it is recommended to use the Multi PDU Mapping feature of the IpduM. This avoids misinterpretation of the received length due to the discrete data lengths of CAN FD frames.

7.7.1 Transmission of Dynamic Length Signals / I-PDUs

For sending a dynamic length signal, the user (e.g. RTE or SwCluC) needs to specify the actual length at the send call. The corresponding API is `Com_SendDynSignal`.

[SWS_Com_00757] ⌈At a call to `Com_SendDynSignal`, the AUTOSAR COM module shall set the length of the corresponding dynamic length I-PDU to the smallest length that includes the dynamic length signal. The length of the dynamic signal shall be equal to the value of input parameter "Length" passed in the API `Com_SendDynSignal`.⌋

The AUTOSAR COM module is able to send dynamic length I-PDUs via normal or large I-PDUs. In any case, it needs to pass the length of the dynamic I-PDU to the lower layers.

[SWS_Com_00832] ⌈When the AUTOSAR COM module invokes PduR_ComTransmit for a dynamic length I-PDU, the SduLength of the PduInfoType shall be set according the last update (or init value if not yet sent) of the included dynamic length signal.⌋

7.7.2 Reception of Dynamic Length Signals / I-PDUs

For receiving a dynamic length signal, the users (e.g. RTE, SwCluC) need to be informed about the actual size.

[SWS_Com_00758] ⌈At reception of a dynamic length I-PDU, the AUTOSAR COM module shall calculate the length of the contained dynamic length signal, by:

<dynamic signal length in bytes> = <received I-PDU length in bytes>

- <ComBitPosition of dynamic length signal> / 8⌋

Com_ReceiveDynSignal returns the length of a dynamic length signal, calculated as defined above.

7.8 MetaData Handling

COM supports MetaData in PDUs received from and transmitted to PduR, and in Signals exchanged with the user (e.g. RTE or SwCluC). One possibility to use MetaData during transmission is to use Com_TriggerIPDUSendWithMetaData or Com_Transmit to trigger transmission of an I-PDU with a defined set of Meta Data. The other is to use the APIs Com_SendSignalWithMetaData, Com_SendDynSignalWithMetaData, Com_SendSignalGroupWithMetaData, Com_SendSignalGroupArrayWithMetaData to provide MetaData for the next transmission of the I-PDU containing the signal. Received MetaData will be stored alongside the I-PDU and can be accessed using Com_ReceiveDynSignalWithMetaData, Com_ReceiveSignalWithMetaData, Com_ReceiveSignalGroupWithMetaData, Com_ReceiveSignalGroupArrayWithMetaData.

[SWS_Com_00887] Transmission Side Meta Data Handling ⌈If MetaData is configured for a transmitted I-PDU, the I-PDU shall be transmitted with the MetaData last provided by the user (e.g. RTE or SwCluC), or, if no MetaData has been provided by the user, with the configured default values specified by ComMetaDataDefaultItem containers. MetaData provided by Com_TriggerIPDUSendWithMetaData shall just be used for the transmission triggered by this call.⌋

[SWS_Com_00888] Reception Side Meta Data Handling ⌈If MetaData is configured for a received I-PDU (with ComIPduLargeData set to false), received MetaData shall be stored for access by the user (e.g. RTE or SwCluC).⌋

[SWS_Com_00893] Reception Side Meta Data Initialization ⌈If MetaData is configured for a received I-PDU, the stored MetaData shall be initialized from the configured ComMetaDataDefaultItems whenever values of the signals of the I-PDU are initialized according to [SWS_Com_00217] or [SWS_Com_00222].⌋

The configured default values specified by ComMetaDataDefaultItem are therefore provided in Com_ReceiveDynSignalWithMetaData, Com_ReceiveSignalWithMetaData, Com_ReceiveSignalGroupWithMetaData, Com_ReceiveSignalGroupArrayWithMetaData calls before an I-PDU is received.

7.9 Interface between AUTOSAR COM Module and the PDU Router

[2] leaves the interface between COM and the lower layers undefined. In AUTOSAR, the only lower layer that the AUTOSAR COM module interfaces to is the PDU Router.

The AUTOSAR COM module uses the PDU Router in two different modes, depending on the type of the I-PDU it will be transported with the bus-specific transport protocol or without. If an I-PDU is supposed to be sent via TP, is configured per configuration parameter ComIPduType.

[SWS_Com_00138] ⌈The AUTOSAR COM module shall send out I-PDUs by a calling the PduR_ComTransmit function.⌋

[SWS_Com_00759] ⌈If the AUTOSAR COM module sends out an I-PDU with ComIPduType configured to NORMAL, the AUTOSAR COM module shall expect to be called by the PDU Router using Com's interfaces for modules that do not use TP.⌋

See column normal I-PDUs in Table 7.7.

[SWS_Com_00760] ⌈If the AUTOSAR COM module sends out an I-PDU with ComIPduType configured to TP, the AUTOSAR COM module shall expect to be called by the PDU Router using Com's interfaces for modules that use TP.⌋

See column TP I-PDUs in Table 7.7.

The interaction diagram in Chapter 9.1 shows the interaction between the PDU Router module and the AUTOSAR COM module for normal I-PDUs.

Table 7.7 gives an overview, which API between the AUTOSAR COM module and the PDU Router is used for a concrete I-PDU with respect to the configured value of ComIPduType. The API description itself is specified in Chapter 7.14.

API	normal I-PDUs	TP I-PDUs
PduR_ComTransmit	X	X
Com_TxConfirmation	X	-
Com_RxIndication	X	-

▽

△

API	normal I-PDUs	TP I-PDUs
Com_TriggerTransmit	X	-
Com_TpTxConfirmation	-	X
Com_TpRxIndication	-	X
Com_StartOfReception	-	X
Com_CopyRxData	-	X
Com_CopyTxData	-	X

Table 7.7: API to PDU Router with respect to ComIPduType

7.10 Update-Bits

To enable the receiver of a signal / signal group to identify whether the sender has updated the data in this signal / signal group before sending, the AUTOSAR COM module supports update-bits.

The update-bits indicate whether the user (e.g. RTE or SwCluC) on sender-side has updated a signal value, before the I-PDU, containing that signal, was transmitted to the PDU Router.

Update-bits are not allowed if transmission mode DIRECT is used with ComTxModeNumberOfRepetitions greater or equal 1 (see [SWS_Com_CONSTR_00310]).

By configuration on sender- and on receiver-side, it is possible to add separately for each signal and / or separately for each signal group at most one update-bit.

The position of the update-bit is configurable by configuration parameter ComUpdateBitPosition. ComUpdateBitPosition is included within the configuration container ComSignal. Hence, it is ensured that the signal / signal group and the corresponding update-bit are always part of the same I-PDU.

[SWS_Com_00055] ⌈The AUTOSAR COM module shall handle the update-bit only internally and not as part of the signal or signal group.⌋

The update-bits are not directly visible or accessible by AUTOSAR Software Components.

[SWS_Com_00059]

Upstream requirements: SRS_BSW_00101

⌈The AUTOSAR COM module shall interpret the update-bit as follows: 0 indicates "cleared / data has not been updated", 1 indicates "set / data has been updated".⌋

7.10.1 Sender Side

The initialization of update-bits is defined by [SWS_Com_00117].

[SWS_Com_00061] ⌈If the user (e.g. RTE or SwCluC) updates the value of a signal by calling Com_SendSignal, the AUTOSAR COM module shall set the update-bit of this signal.⌋

[SWS_Com_00801] ⌈If the user (e.g. RTE or SwCluC) updates a signal group by calling Com_SendSignalGroup, the AUTOSAR COM module shall set the update-bit of this signal group.⌋

[SWS_Com_00062] ⌈If the parameter ComTxIPduClearUpdateBit of an I-PDU is configured to Transmit, the AUTOSAR COM module shall clear all update-bits of all contained signals and signal groups after this I-PDU was sent out via PduR_ComTransmit and PduR_ComTransmit returned E_OK.⌋

[SWS_Com_00577] ⌈If the parameter ComTxIPduClearUpdateBit of an I-PDU is configured to Confirmation, the AUTOSAR COM module shall clear all update-bits of all contained signals and signal groups after this I-PDU was sent out via PduR_ComTransmit, PduR_ComTransmit returned E_OK and the I-PDU was successfully confirmed.⌋

[SWS_Com_00578] ⌈If the parameter ComTxIPduClearUpdateBit of an I-PDU is configured to TriggerTransmit, the AUTOSAR COM module shall clear all update-bits of all contained signals and signal groups after the contents of this I-PDU was successfully requested by Com_TriggerTransmit.⌋

7.10.2 Receiver Side

[SWS_Com_00324] ⌈If a signal or signal group with an attached update bit is received, the AUTOSAR COM module shall only process this signal, i.e. filter, notification, signal based gateway, byte swapping, if the signal's update bit is set.⌋

[SWS_Com_00802] ⌈If a signal or signal group with an attached update bit is received and the update-bit is not set, the AUTOSAR COM module shall discard this signal.⌋

If the signal has not been updated, the signal will not be routed via the signal gateway. It will only be discarded.

If the user (e.g. RTE or SwCluC) reads a signal with an associated cleared update-bit, the init value or the last received value is returned.

[SWS_Com_00067] ⌈The AUTOSAR COM module shall interpret a signal / signal group as updated if the signal has an update-bit attached, and the value of the update-bit is set.⌋

For the behavior of reception deadline monitoring on signals with update-bits, see Chapter 7.2.6.1.

7.11 Signal Gateway

The signal gateway is an integrated part of the AUTOSAR COM module. Any external modules, except the cyclic task call, cannot access the signal gateway.

The AUTOSAR COM module's signal gateway works with signals and signal groups.

The AUTOSAR COM module's signal gateway supports only static routing for the ComGwMapping configuration container. It is only possible to configure routes independently of the contents of the routed signals and group signals.

The destination of a signal or group signal can be configured either per ComGwDestinationDescription container or via ComGwSignal reference. The transmission mode will be selected based on the TMCs of other associated signals of the I-PDU. In case no signal contributes to the TMS calculation, the transmission mode ComTxModeTrue [SWS_Com_00677] will be used.

[SWS_Com_00377] ⌈The AUTOSAR COM module's signal gateway shall copy the value of routed signals / group signals to the signals / group signals for transmission according to configuration, see ComGwMapping.⌋

The ComGwMapping configuration allows to configure routing a signal / group signal from one source signal / group signal to zero (no signal gateway functionality) or more destinations (1:n).

[SWS_Com_00872] ⌈The Signal Gateway shall support the following I-PDU / signal-processing stages on the receiver side:

1. reset reception deadline monitoring timer for I-PDU based monitoring
2. I-PDU callout
3. check of update-bits
4. endianness conversion and sign extension

⌋

[SWS_Com_00873] ⌈The Signal Gateway shall support the following I-PDU / signal-processing stages on the sender side:

1. set of update-bits
2. endianness conversion and sign extension
3. I-PDU transmission mode selection
4. I-PDU callout

⌋

See also Figure 7.4.

7.11.1 Dealing with Signals

[SWS_Com_00357] ⌈The AUTOSAR COM module shall forward signals to be routed from received I-PDUs to transmit I-PDUs. For configuration, see configuration container ComGwMapping.⌋

Stopping the receive I-PDU after the signals to be routed have been forwarded to the transmit I-PDU will not prevent the transmit I-PDU from being transmitted unless it is stopped as well.

[SWS_Com_00360] ⌈If the endianness of a received signal to be routed differs from the endianness of a related destination signal, the AUTOSAR COM module shall convert its endianness to the endianness of the destination signal.⌋

7.11.2 Dealing with Group Signals

[SWS_Com_00361] ⌈The AUTOSAR COM module shall forward routed group signals from received I-PDUs to transmit I-PDUs. See also configuration container ComGwMapping.⌋

Stopping the receive I-PDU after the group signals to be routed have been forwarded to the transmit I-PDU will not prevent the transmit I-PDU from being transmitted unless it is stopped as well.

[SWS_Com_00383] ⌈The AUTOSAR COM module shall route all group signals belonging to the same signal group in a consistent manner. Therefore, the AUTOSAR COM module shall transfer the data of a signal group as one consistent (sub-)set of data during a routing operation.⌋

[SWS_Com_00735] ⌈The AUTOSAR COM module shall support routing consistently a subset of group signals of a source signal group into a reduced target signal group.⌋

[SWS_Com_00833] ⌈The AUTOSAR COM module shall support routing of individual group signals of a source signal group to a individual target signals (which are not part of signal group).⌋

[SWS_Com_00362] ⌈If the endianness of a received group signal to be routed differs from the endianness of a related destination (group) signal, the AUTOSAR COM module shall convert its endianness to the endianness of the destination (group) signal.⌋

7.11.3 Routing of Out-Timed Signals and Signal Groups

[SWS_Com_00701] ⌈The AUTOSAR COM module's signal gateway shall route signal and group signals even if any configured reception deadline monitoring timeout expired.⌋

In case of a not in time received signal or group signal the AUTOSAR COM module's signal gateway will route these signal or group signal anyway.

7.11.4 Handling of Update-Bits

The routing relations for signal groups are solely defined on the group signals. However, with respect to the update-bit the integrated signal gateway has to respect the update-bits of the enclosing signal groups. Hence, when update-bits of signal groups are referenced within this chapter the signal groups that include the configured group signals are meant.

[SWS_Com_00702] ⌈If both, the received signal / signal group and the destination signal / signal group have an update-bit (ComUpdateBitPosition) configured and the update-bit of the received signal / signal group is set, the AUTOSAR COM module shall route the signal / signal group with the set update-bit and clear the update-bit of the destination signal / signal group after it was sent.⌋

[SWS_Com_00703] ⌈If the received signal / signal group and the destination signal / signal group have an update-bit (ComUpdateBitPosition) configured, and the update-bit of the received signal / signal group is not set, the AUTOSAR COM module shall not route this signal / signal group.⌋

[SWS_Com_00704] ⌈If the received signal / signal group has an update-bit (ComUpdateBitPosition) configured, but the destination signal has no update-bit configured, and the update-bit is set, the AUTOSAR COM module shall route this signal / signal group without the update-bit.⌋

[SWS_Com_00705] ⌈If the received signal / signal group has an update-bit (ComUpdateBitPosition) configured, but the destination signal has no update-bit configured, and the update-bit is not set, the AUTOSAR COM module shall not route this signal / signal group.⌋

[SWS_Com_00706] ⌈If the received signal / signal group has no update-bit (ComUpdateBitPosition) configured and the destination signal / signal group has an update-bit configured, the AUTOSAR COM module shall set the update-bit of the destination signal when a new signal / signal group was received and clear it after sending of the destination signal / signal group.⌋

7.11.5 Decoupling Signal Gateway

To protect interrupt routines used for I-PDU reception from incalculable and perhaps expensive time usage, it is necessary to decouple the AUTOSAR COM module's signal gateway from interrupt routines.

[SWS_Com_00359]

Upstream requirements: SRS_BSW_00425, SRS_BSW_00432

⌈The AUTOSAR COM module shall execute all functions of its signal gateway during the function call `Com_MainFunctionRouteSignals`. During this function call, the AUTOSAR COM module's signal gateway shall check received routed signals and group signals and copy them from the related receive I-PDUs to the related transmit I-PDUs.⌋

[SWS_Com_00466]

Upstream requirements: SRS_BSW_00432

⌈Within `Com_MainFunctionRouteSignals`, the AUTOSAR COM module shall evaluate the transfer properties and transmission modes and perform its routing actions in the following sequence:

1. copy all gated signals from the source to the target I-PDUs
2. evaluate the TMC of all gated signals
3. evaluate the TMS for the target I-PDUs

⌋

See also Figure 7.1.

In the case, that a target I-PDU contains gated signals with `ComTransferProperty TRIGGERED` that are newly received, the sending may be performed in either `Com_MainFunctionTx` or `Com_MainFunctionRouteSignals`.

[SWS_Com_00539] ⌈The AUTOSAR COM module shall sent out an I-PDU at most once while one call to `Com_MainFunctionRouteSignals`.⌋

7.12 Error Classification

Chapter [4, General Specification of Basic Software Modules] 7.2 *"Error Handling"* describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

7.12.1 Development Errors

[SWS_Com_91014] Definition of development errors in module Com

Upstream requirements: SRS_BSW_00337, SRS_BSW_00414

⌈

Type of error	*Related error code*	*Error value*
API service called with wrong parameter (invalid signal, signal group, I-PDU, I-PDU group identifier or e.g., Com_SendSignal for a dynamic length signal, Com_SendDynSignal for a fix length signal)	COM_E_PARAM	0x01
Error code if any API service (except Com_Get Status, Com_MainFunctionRx, Com_Main FunctionTx and Com_MainFunctionRouteSignals) is called before the AUTOSAR COM module was initialized with Com_Init or after a call to Com_De Init	COM_E_UNINIT	0x02
NULL pointer checking	COM_E_PARAM_POINTER	0x03
Invalid configuration set selection	COM_E_INIT_FAILED	0x04

⌋

7.12.2 Runtime Errors

[SWS_Com_91015] Definition of runtime errors in module Com

Upstream requirements: SRS_BSW_00452

⌈

Type of error	*Related error code*	*Error value*
Transmission request was skipped	COM_E_SKIPPED_TRANSMISSION	0x05
COM I-PDU lost during transmission or reception	COM_E_IPDU_LOST	0x06

⌋

7.12.3 Production Errors

[SWS_Com_00907] Reception Deadline Monitoring PREFAILED Reporting to Dem ⌈When the reception deadline monitoring timer of a signal or signal group expires, the AUTOSAR COM module shall notify Dem via `Dem_SetEventStatus` API with DEM_EVENT_STATUS_PREFAILED if `COM_E_TIMEOUT` is configured.⌋

[SWS_Com_00908] Reception Deadline Monitoring PREPASSED Reporting to Dem ⌈When a signal or signal group with active deadline monitoring is received, the AUTOSAR COM module shall notify Dem via `Dem_SetEventStatus` API with DEM_EVENT_STATUS_PREPASSED if `COM_E_TIMEOUT` is configured.⌋

[SWS_Com_00909] Transmission Deadline Monitoring PREPASSED Reporting to Dem ⌈When the AUTOSAR COM receives a successful transmit confirmation for an I-PDU with running transmission deadline monitoring, the AUTOSAR COM shall notify Dem via `Dem_SetEventStatus` API with DEM_EVENT_STATUS_PREPASSED if `COM_E_TIMEOUT` is configured.⌋

7.13 AUTOSAR COM Module's Interaction Model

This chapter corresponds to the Clause 3.8 (Functional model of the Interaction Layer) of [2]. The following figures illustrate the behavior of the Interaction layer for external reception and external transmission if `ComIPduLargeData` is set to false. The complete functionality is shown but it depends on the configuration what parts are present / used in a concrete implementation.

[SWS_Com_00396] ⌈A received signal or signal group can be configured for various processing steps. The AUTOSAR COM module shall execute the configured processing steps in the following order:

1. reset reception deadline monitoring timer for I-PDU based monitoring
2. check update-bits
3. endianness conversion and sign extension
4. data invalidation
5. reception filtering
6. reset reception deadline monitoring timer for signal based monitoring
7. user callback notification

⌋

If `ComIPduLargeData` is set to true then update-bit handling, endianness conversion, sign extension, data invalidation, reception filtering steps are skipped.

[SWS_Com_00744] ⌈The AUTOSAR COM module shall execute the configured timeout handling steps in the following order:

1. perform the configured `ComRxDataTimeoutAction`
2. call the configured `<ComUser_CbkRxTOut>`
3. notify Dem if `COM_E_TIMEOUT` is configured

⌋

Timeout handling can be configured to have various steps when reception deadline monitoring timer expires.

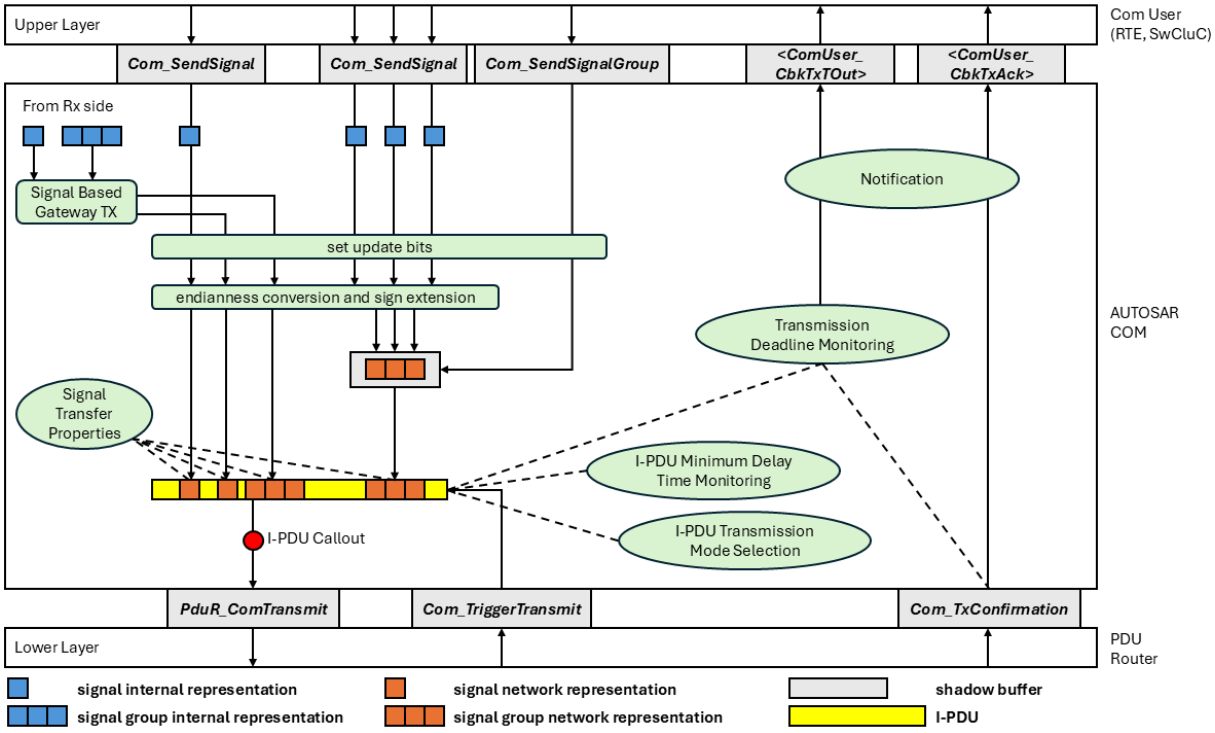


Figure 7.2: The AUTOSAR COM module's interaction model for transmission

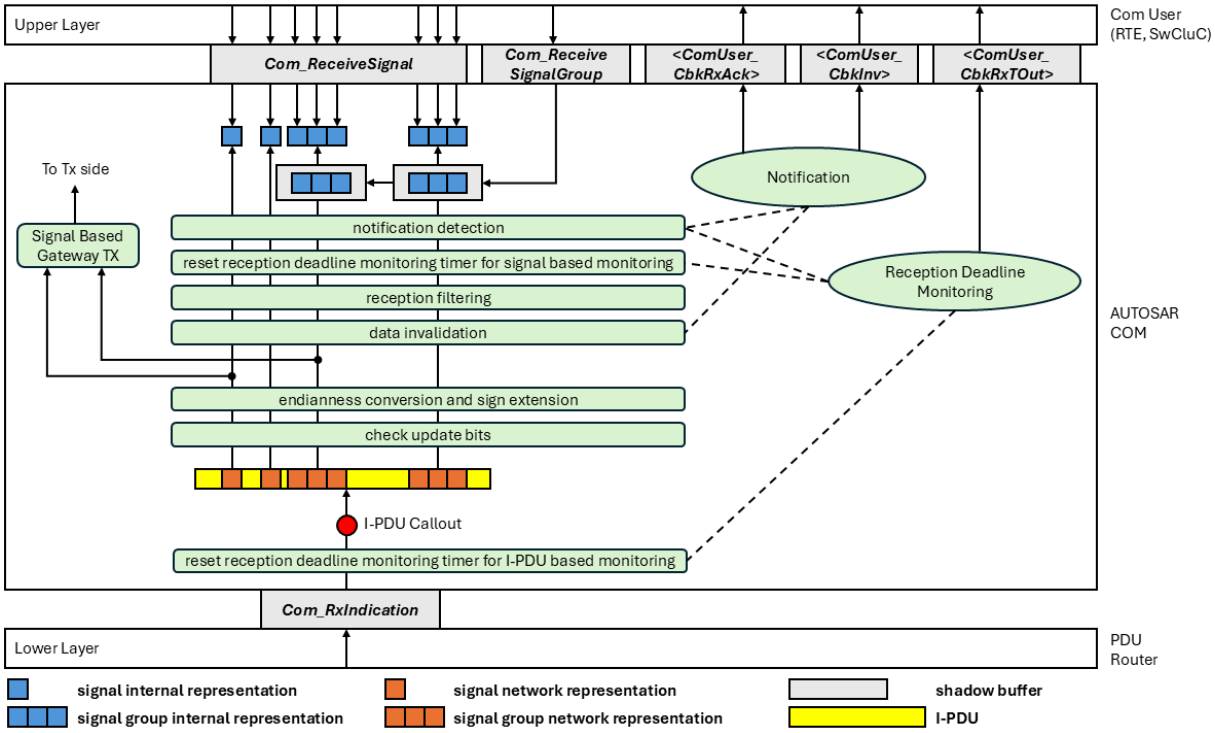


Figure 7.3: The AUTOSAR COM module's interaction model for reception

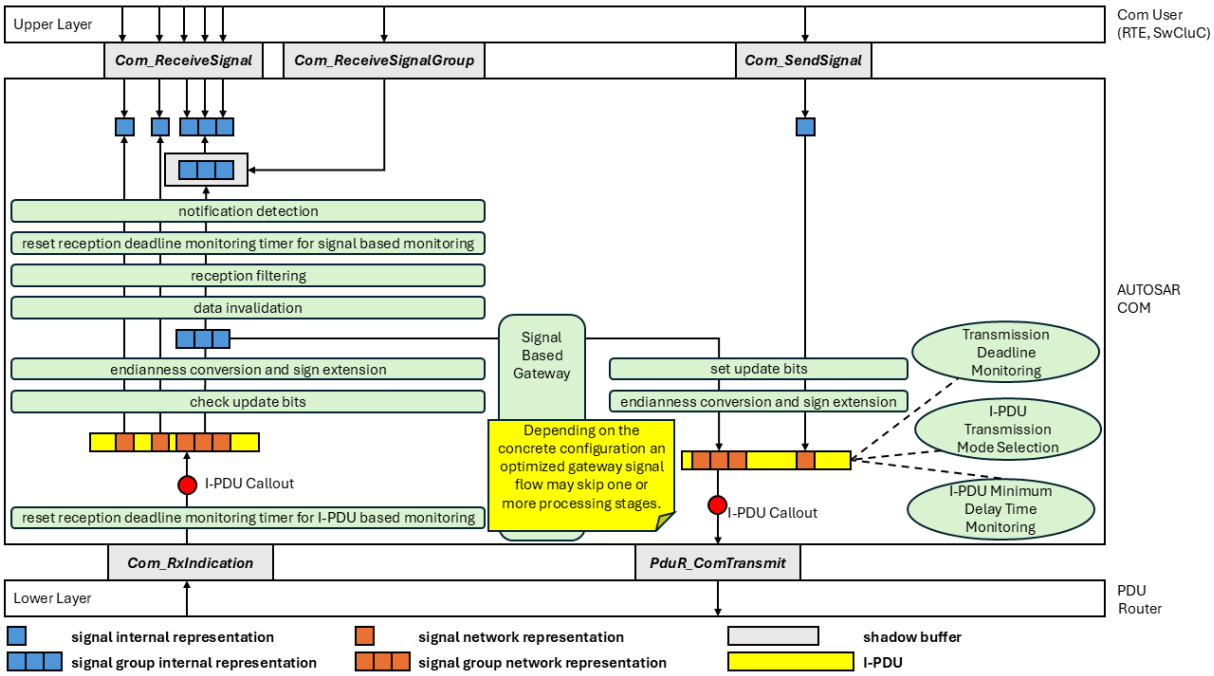


Figure 7.4: The AUTOSAR COM module's interaction model for integrated Signal Gateway

The endianness conversion and sign extension on receiver side are needed to feed the TMS with a correct data format. This endianness conversion is only necessary if the endianness of the Rx-bus differs from the endianness of the CPU. The endianness conversion on the sender side is only necessary if the endianness of the Rx-bus differs from the endianness of the Tx-bus.

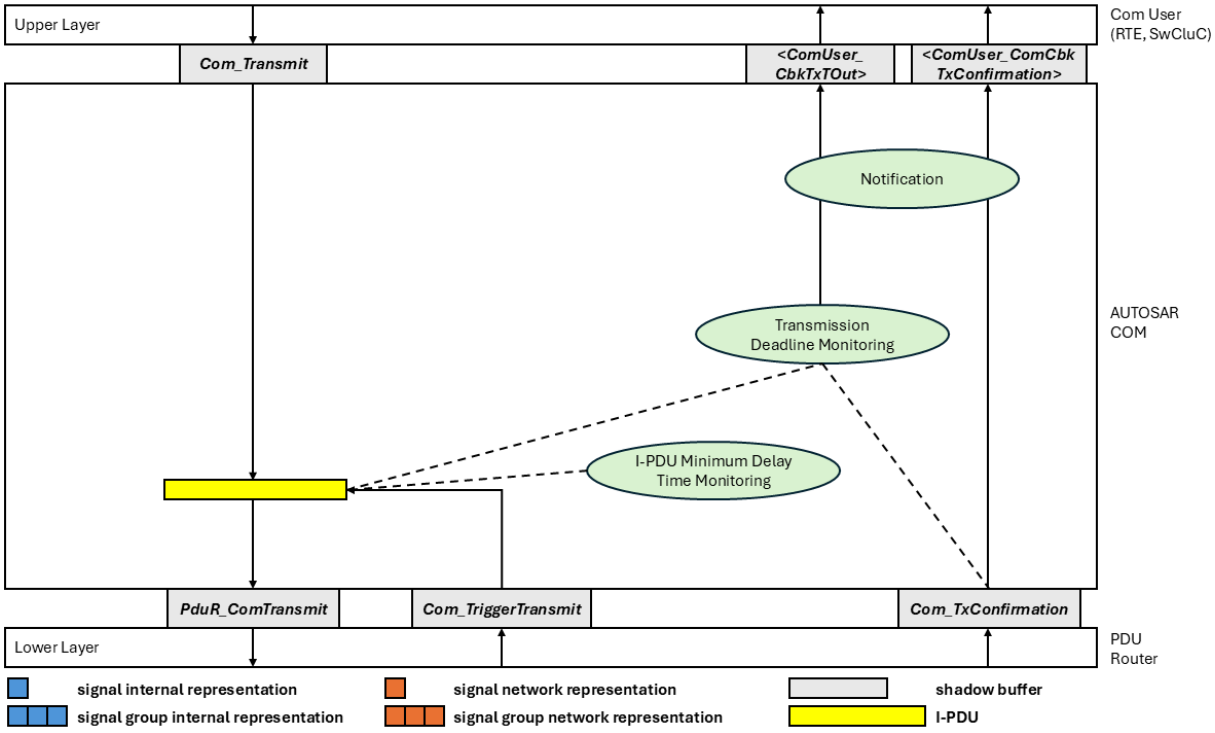


Figure 7.5: The AUTOSAR COM module's interaction model for transmission with ComIPduLargeData enabled

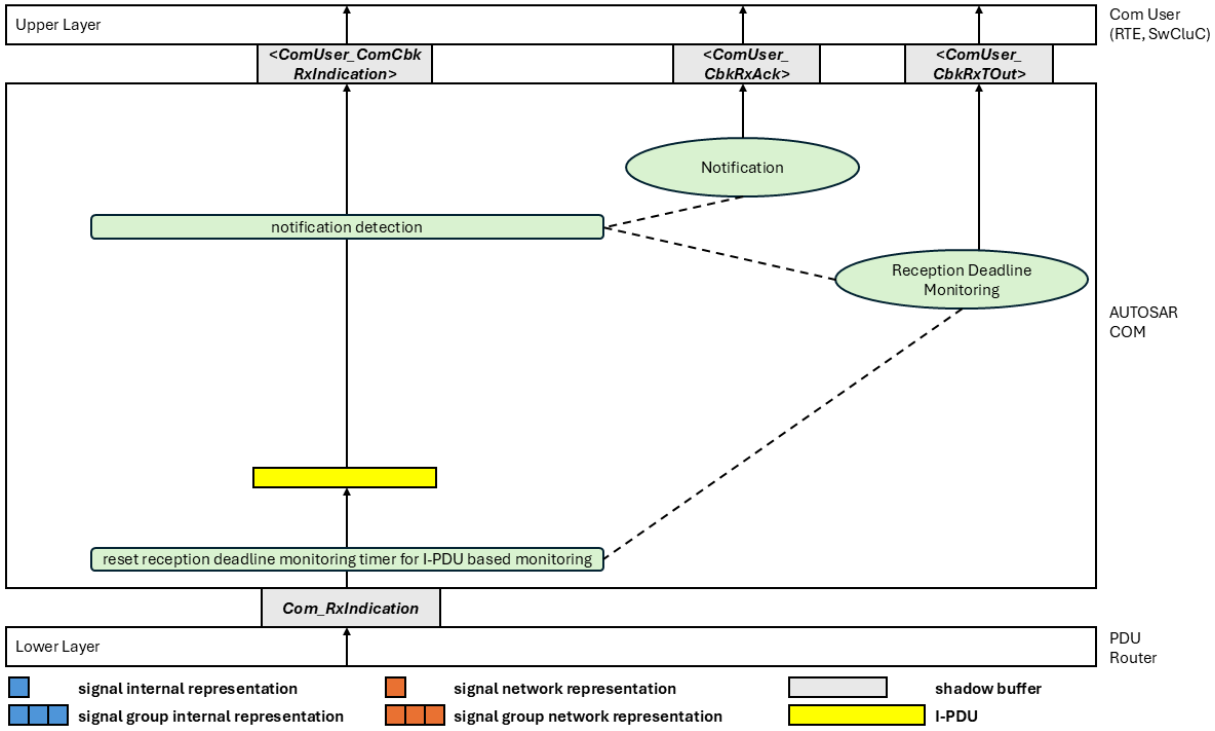


Figure 7.6: The AUTOSAR COM module's interaction model for reception with ComIPdu-LargeData enabled

7.14 Multicore 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 Flex Ray, 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 Com 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. The only exception in the Com module is the signal gateway which needs special consideration.

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

[SWS_Com_00885]

Upstream requirements: SRS_BSW_00432

⌈ComIPdus shall be processed within the MainFunction, which is referenced via `ComIPduMainFunctionRef`.⌋

[SWS_Com_00886]

Upstream requirements: SRS_BSW_00432

⌈If `ComPreparationNotification` is configured, the respective `Com_CbkTxPrep` shall be called within MainFunctionTx before I-PDU processing is started.⌋

[SWS_Com_00890]

Upstream requirements: SRS_BSW_00432

⌈`Com_MainFunctionRouteSignals` shall only handle ComIPdus and their gateway signals routings which reference to the `Com_MainFunctionRouteSignals` via Com MainFunctionRouteSignalsRef.⌋

7.15 Software Cluster Connection Layer (SwCluC)

SwCluC groups the AUTOSAR embedded software into cluster (SWCL). It describes Host- and Application Cluster. Per machine, there is one Host-Cluster which contains most of or even all the BSW modules. The Application Software Cluster contains mostly application SW-C.

From within a Software Cluster, the COM APIs are not accessed directly. Instead, accesses are done via a Proxy module ("Low Proxy"). For COM this means that it can have more than one upper layer users:

- The RTE (of the Host Software Cluster)
- The SwCluC Low Proxy
- User defined

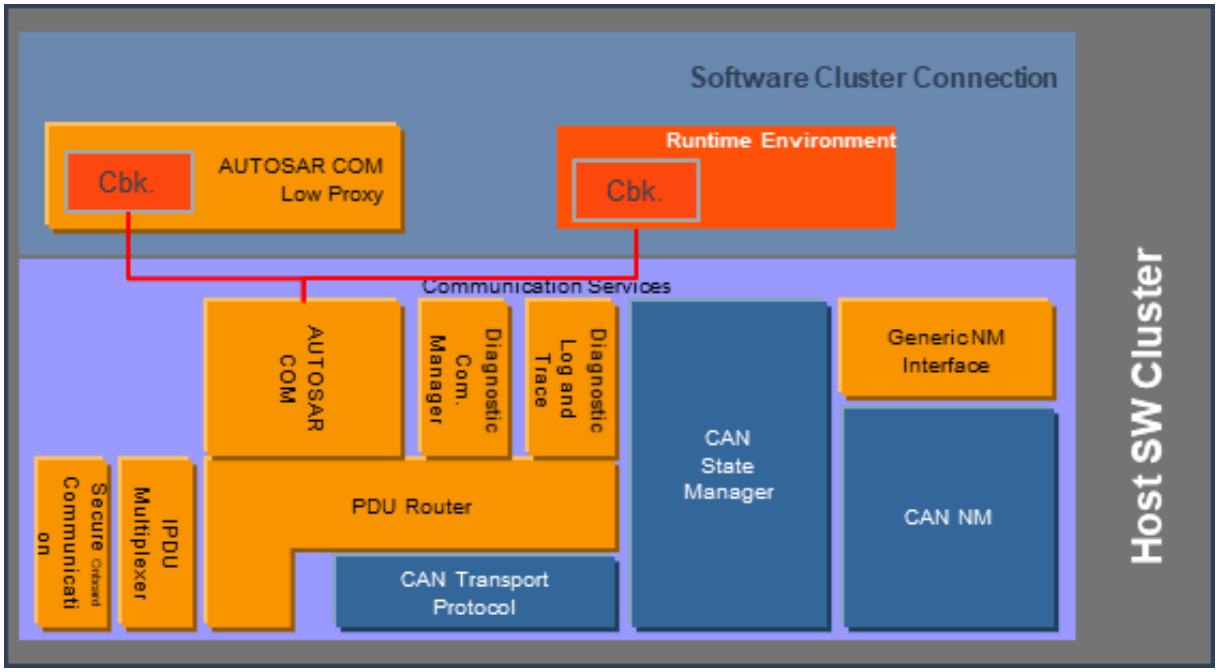


Figure 7.7

For the callback-interface, COM can serve an arbitrary number of users, which each define their individual set of callback functions.

There is a Low proxy on every ECUC partition, where COM services are used by the Application Software Component. Proxies and COM only interact in the same partition scope. Any partition crossing of a signal or signal group has to be done within the the SWC or within the Low Proxy.

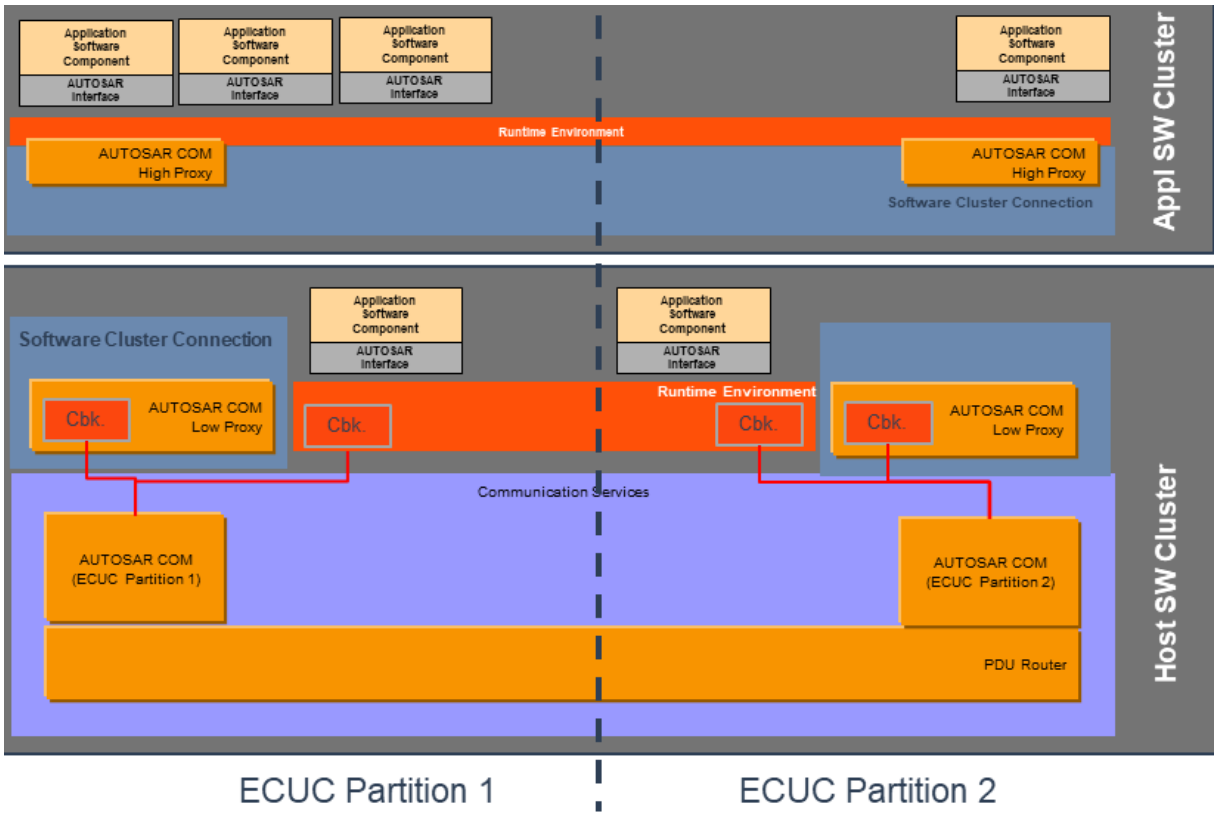


Figure 7.8

7.16 Security Events

[SWS_Com_00900] Security events for Com ⌈

Name	*Description*	*ID*
SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED	Signal or group signal is received with unexpected value.	89

⌋

[SWS_Com_00901] Security event reporting ⌈If security event reporting is enabled by ComEnableSecurityEventReporting, the respective security events shall be reported to the IdsM via the interfaces defined in [4, SWS BSW General].⌋

[SWS_Com_00902] SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED

Upstream requirements: SRS_BSW_00491

⌈If SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED is configured and the AUTOSAR COM module filters out a signal or group signal on receiver side (exception for ONE_EVERY_N and NEVER ComFilterAlgorithm types), the module shall report the security event SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED.⌋

[SWS_Com_00903] Security event context data definition: SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED ⌈

SEV Name	*SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED*	
ID	89	
Description	Signal or group signal is received with unexpected value.	
Context Data Version	1	
Context Data	*Data Type*	*Allowed Values*
ComHandleId	uint16	

⌋

8 API specification

8.1 Imported types

In this chapter, all types included from the following files are listed:

[SWS_Com_00609] Definition of imported datatypes of module Com

Upstream requirements: SRS_BSW_00301

⌈

Module	*Header File*	*Imported Type*
Comtype	ComStack_Types.h	BufReq_ReturnType
	ComStack_Types.h	CbkHandleIdType
	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
	ComStack_Types.h	RetryInfoType
	ComStack_Types.h	TpDataStateType
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
IdsM	IdsM.h	IdsM_SecurityEventIdType
	Rte_IdsM_Type.h	IdsM_TimestampDataType
Std	Std_Types.h	Std_ReturnType

⌋

8.2 Type definitions

8.2.1 Com_StatusType

[SWS_Com_00819] Definition of datatype Com_StatusType

Upstream requirements: SRS_BSW_00335

⌈

Name	Com_StatusType		
Kind	Enumeration		
Range	COM_INIT	–	The AUTOSAR COM module is initialized and usable.
	COM_UNINIT	0x00	The AUTOSAR COM module is not initialized or not usable. This shall be the default value after reset. This status shall have the value 0.
Description	This is a status value returned by the API service Com_GetStatus().		
Available via	Com.h		

⌋

8.2.2 Com_SignalIdType

[SWS_Com_00820] Definition of datatype Com_SignalIdType

Upstream requirements: SRS_BSW_00441

⌈

Name	Com_SignalIdType		
Kind	Type		
Derived from	uint16		
Range	0..<SignalIdmax>	–	Zero-based integer number
Description	The AUTOSAR COM module's signal object identifier.		
Available via	Com.h		

⌋

8.2.3 Com_SignalGroupIdType

[SWS_Com_00821] Definition of datatype Com_SignalGroupIdType

Upstream requirements: SRS_BSW_00441

⌈

Name	Com_SignalGroupIdType		
Kind	Type		
Derived from	uint16		
Range	0..<SignalGroupIdmax>	–	Zero-based integer number
Description	The AUTOSAR COM module's signal group object identifier.		
Available via	Com.h		

⌋

8.2.4 Com_IpduGroupIdType

[SWS_Com_00822] Definition of datatype Com_IpduGroupIdType

Upstream requirements: SRS_BSW_00441

⌈

Name	Com_IpduGroupIdType		
Kind	Type		
Derived from	uint16		
Range	0..<IpduGroupId-max>	–	Zero-based integer number; where IpduGroupId-max < Com SupportedIPduGroups
Description	The AUTOSAR COM I-PDU module's group object identifier.		
Available via	Com.h		

⌋

8.2.5 Com_ConfigType

[SWS_Com_00825] Definition of datatype Com_ConfigType

Upstream requirements: SRS_BSW_00404, SRS_BSW_00441

⌈

Name	Com_ConfigType	
Kind	Structure	
Elements	implementation specific	
	Type	–
	Comment	The content of the initialization data structure is implementation specific
Description	This is the type of the data structure containing the initialization data for COM.	
Available via	Com.h	

⌋

8.3 Function definitions

This chapter contains all APIs provided by the module.

[SWS_Com_00321]

Upstream requirements: SRS_BSW_00312

⌈Non-reentrant functions do not have to check if they are called reentrant.⌋

Most of the COM functions are non-reentrant with respect to a concrete object. E.g. `Com_SendSignal` is not reentrant for the same signal, but reentrant for different signals. Hence, if there are multiple callers of the AUTOSAR COM module they need to be synchronized on a system level, for example by using disjoint sets of signals and I-PDU groups.

8.3.1 Return Codes

The AUTOSAR COM module does not define a special COM return type. The API services return errors either by using the Std_ReturnType as defined in [16] or via a uint8 value mapped according to [SWS_Com_00865].

[SWS_Com_00865] Mapping of AUTOSAR COM module's return codes

Upstream requirements: SRS_BSW_00348, SRS_BSW_00377, SRS_BSW_00441

⌈

Name	Description	Type	Value	Defined in
E_OK	the service has been accepted	#define	0x00	StandardTypes.h
E_NOT_OK	invocation of service failed	#define	0x01	StandardTypes.h
COM_SERVICE_NOT_AVAILABLE	the service is currently not available e.g. the corresponding I-PDU group is stopped	#define	0x80	Com.h
COM_BUSY	Successful execution of this function is currently not possible, because a resource is currently in use. For example, the buffer of a large I-PDU is locked. COM_BUSY might, but must not, be returned whenever a signal of a large I-PDU is accessed.	#define	0x81	Com.h

⌋

8.3.2 Start-Up and Control Services

8.3.2.1 Com_Init

[SWS_Com_00432] Definition of API function Com_Init

Upstream requirements: SRS_BSW_00344, SRS_BSW_00404, SRS_BSW_00405, SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414

⌈

Service Name	Com_Init	
Syntax	`void Com_Init ( const Com_ConfigType* config )`	
Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	config	Pointer to the AUTOSAR COM module's configuration data.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service initializes internal and external interfaces and variables of the AUTOSAR COM module layer for the further processing. After calling this function the inter-ECU communication is still disabled. In configurations, in which Com is assigned to more than one partition (i.e. Com_MainFunctions are mapped to partitions), Com may provide one init function per partition.	
Available via	Com.h	

⌋

The behavior of the AUTOSAR COM module is unspecified until a correct call to `Com_Init` is made.

Caveats of Com_Init: It is not foreseen that Com_Init pre-empts any other AUTOSAR COM module's function. The AUTOSAR COM module can relies on the rest of the system that Com_Init is not called in such a way.

8.3.2.2 Com_DeInit

[SWS_Com_00130] Definition of API function Com_DeInit

Upstream requirements: SRS_BSW_00336

⌈

Service Name	Com_DeInit
Syntax	`void Com_DeInit ( void )`
Service ID [hex]	0x02
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None
Return value	None
Description	This service stops the inter-ECU communication. All started I-PDU groups are stopped and have to be started again, if needed, after Com_Init is called. By a call to Com_DeInit the AUTOSAR COM module is put into an not initialized state.
Available via	Com.h

⌋

[SWS_Com_00129]

Upstream requirements: SRS_BSW_00336

⌈The AUTOSAR COM module's function Com_DeInit shall stop all started I-PDU groups.⌋

Caveats of Com_DeInit: It is not foreseen that Com_DeInit pre-empts any AUTOSAR COM module's function. The AUTOSAR COM module can relies on the rest of the system that Com_DeInit is not called in such a way.

8.3.2.3 Com_IpduGroupStart

[SWS_Com_91001] Definition of API function Com_IpduGroupStart ⌈

Service Name	Com_IpduGroupStart
Syntax	`void Com_IpduGroupStart ( Com_IpduGroupIdType IpduGroupId, boolean initialize )`

▽

△

Service ID [hex]	0x03	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different I-PDU groups. Non reentrant for the same I-PDU group.	
Parameters (in)	IpduGroupId	Id of I-PDU group to be started
	initialize	flag to request initialization of the data in the I-PDUs of this I-PDU group
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Starts a preconfigured I-PDU group. For example, cyclic I-PDUs will be sent out cyclically after the call of Com_IpduGroupStart(). If Initialize is true all I-PDUs of the I-PDU group shall be (re-)initialized before the I-PDU group is started. That is they shall behave like after a start-up of COM, for example the old_value of the filter objects and shadow buffers of signal groups have to be (re-)initialized.	
Available via	Com.h	

⌋

8.3.2.4 Com_IpduGroupStop

[SWS_Com_91002] Definition of API function Com_IpduGroupStop ⌈

Service Name	Com_IpduGroupStop	

Syntax	`void Com_IpduGroupStop (`
	`  Com_IpduGroupIdType IpduGroupId`
	`)`	

Service ID [hex]	0x04	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different I-PDU groups. Non reentrant for the same I-PDU group.	
Parameters (in)	IpduGroupId	Id of I-PDU group to be stopped
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Stops a preconfigured I-PDU group. For example, cyclic I-PDUs will be stopped after the call of Com_IpduGroupStop().	
Available via	Com.h	

⌋

8.3.2.5 Com_EnableReceptionDM

[SWS_Com_91004] Definition of API function Com_EnableReceptionDM ⌈

Service Name	Com_EnableReceptionDM	
Syntax	`void Com_EnableReceptionDM ( Com_IpduGroupIdType IpduGroupId )`	
Service ID [hex]	0x06	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different I-PDU groups. Non reentrant for the same I-PDU group.	
Parameters (in)	IpduGroupId	Id of I-PDU group where reception DM shall be enabled.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Enables the reception deadline monitoring for the I-PDUs within the given I-PDU group.	
Available via	Com.h	

⌋

8.3.2.6 Com_DisableReceptionDM

[SWS_Com_91003] Definition of API function Com_DisableReceptionDM ⌈

Service Name	Com_DisableReceptionDM	
Syntax	`void Com_DisableReceptionDM ( Com_IpduGroupIdType IpduGroupId )`	
Service ID [hex]	0x05	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different I-PDU groups. Non reentrant for the same I-PDU group.	
Parameters (in)	IpduGroupId	Id of I-PDU group where reception DM shall be disabled.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Disables the reception deadline monitoring for the I-PDUs within the given I-PDU group.	
Available via	Com.h	

⌋

8.3.2.7 Com_GetStatus

[SWS_Com_00194] Definition of API function Com_GetStatus

Upstream requirements: SRS_BSW_00331

⌈

Service Name	Com_GetStatus	
Syntax	`Com_StatusType Com_GetStatus ( void )`	
Service ID [hex]	0x07	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Com_StatusType	COM_UNINIT: the AUTOSAR COM module is not initialized and not usable COM_INIT: the AUTOSAR COM module is initialized and usable
Description	Returns the status of the AUTOSAR COM module.	
Available via	Com.h	

⌋

8.3.3 Communication Services

8.3.3.1 Com_SendSignal

[SWS_Com_00197] Definition of API function Com_SendSignal ⌈

Service Name	Com_SendSignal	
Syntax	`uint8 Com_SendSignal ( Com_SignalIdType SignalId, const void* SignalDataPtr )`	
Service ID [hex]	0x0a	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be sent.
	SignalDataPtr	Reference to the signal data to be transmitted.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group was stopped COM_BUSY: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignal updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter.	

▽

△

Available via	Com.h

⌋

[SWS_Com_00624] ⌈The service `Com_SendSignal` shall update the signal object identified by `SignalId` with the signal referenced by the `SignalDataPtr` parameter.⌋

[SWS_Com_00625] ⌈If the updated signal has the `ComTransferProperty TRIGGERED` and it is assigned to an I-PDU with `ComTxModeMode DIRECT` or `MIXED`, then `Com_SendSignal` shall perform an immediate transmission (within the next main function at the latest) of that I-PDU, unless the sending is delayed or prevented by other COM mechanisms.⌋

A related mechanism is for example the minimum delay timer (ComMinimumDelayTime).

8.3.3.2 Com_SendDynSignal

[SWS_Com_00627] Definition of API function Com_SendDynSignal ⌈

Service Name	Com_SendDynSignal	
Syntax	`uint8 Com_SendDynSignal ( Com_SignalIdType SignalId, const void* SignalDataPtr, uint32 Length )`	
Service ID [hex]	0x21	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be sent.
	SignalDataPtr	Reference to the signal data to be transmitted.
	Length	Length of the dynamic length signal
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `E_NOT_OK`: in case the Length is greater than the configured ComSignalLength of this sent signal `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked
Description	The service Com_SendDynSignal updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter.	
Available via	Com.h	

⌋

[SWS_Com_00628] ⌈The service `Com_SendDynSignal` shall update the signal object identified by `SignalId` with the signal referenced by the `SignalDataPtr` parameter.⌋

[SWS_Com_00629] ⌈If the updated signal (see [SWS_Com_00628]) has the ComTransferProperty TRIGGERED, then the function Com_SendDynSignal shall perform an immediate transmission (within the next main function at the latest) of the I-PDU associated with the signal except when the signal is packed into an I-PDU with PERIODIC transmission mode.⌋

[SWS_Com_00630] ⌈If the updated signal (see [SWS_Com_00628]) has the ComTransferProperty PENDING, then the function Com_SendDynSignal shall not perform a transmission of the I-PDU associated with the signal.⌋

There is no restriction regarding the minimal length of a dynamic length signal. Hence, it is allowed that a dynamic length signal has zero length.

8.3.3.3 Com_ReceiveSignal

[SWS_Com_00198] Definition of API function Com_ReceiveSignal ⌈

Service Name	Com_ReceiveSignal	
Syntax	`uint8 Com_ReceiveSignal ( Com_SignalIdType SignalId, void* SignalDataPtr )`	
Service ID [hex]	0x0b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be received.
Parameters (inout)	None	
Parameters (out)	SignalDataPtr	Reference to the location where the received signal data shall be stored
Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group was stopped COM_BUSY: in case the TP-Buffer is locked for large data types handling
Description	Com_ReceiveSignal copies the data of the signal identified by SignalId to the location specified by SignalDataPtr.	
Available via	Com.h	

⌋

[SWS_Com_00631] ⌈The service Com_ReceiveSignal shall copy the data of the signal object identified by SignalId to the position referenced by SignalDataPtr.⌋

Caveats of Com_ReceiveSignal: The AUTOSAR COM module relies on the fact the SignalDataPtr is properly aligned with respect to the data type of the signal with the given SignalId. The caller of Com_ReceiveSignal has to ensure this.

8.3.3.4 Com_ReceiveDynSignal

[SWS_Com_00690] Definition of API function Com_ReceiveDynSignal ⌈

Service Name	Com_ReceiveDynSignal	
Syntax	`uint8 Com_ReceiveDynSignal ( Com_SignalIdType SignalId, void* SignalDataPtr, uint32* Length )`	
Service ID [hex]	0x22	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be received.
Parameters (inout)	Length	in: maximum length that could be received out: length of the dynamic length signal
Parameters (out)	SignalDataPtr	reference to the location where the received signal data shall be stored
Return value	uint8	`E_OK`: service has been accepted `E_NOT_OK`: in case the Length (as in-parameter) is smaller than the received length of the dynamic length signal `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked
Description	Com_ReceiveDynSignal copies the data of the signal identified by SignalId to the location specified by SignalDataPtr and stores the length of the dynamical length signal at the position given by the Length parameter.	
Available via	Com.h	

⌋

[SWS_Com_00711] ⌈The service `Com_ReceiveDynSignal` shall copy the data of the signal object identified by `SignalId` to the position referenced by `SignalDataPtr`.⌋

[SWS_Com_00712] ⌈The service `Com_ReceiveDynSignal` shall return the calculated length (see [SWS_Com_00758]) of the dynamic length signal identified by `SignalId` in parameter Length.⌋

[SWS_Com_00724] ⌈In case, the Length (as in-parameter) is smaller than the received length of the dynamic length signal, `Com_ReceiveDynSignal` shall not copy any data to the position referenced by `SignalDataPtr`.⌋

8.3.3.5 Com_SendSignalGroup

[SWS_Com_00200] Definition of API function Com_SendSignalGroup ⌈

Service Name	Com_SendSignalGroup	
Syntax	`uint8 Com_SendSignalGroup ( Com_SignalGroupIdType SignalGroupId )`	
Service ID [hex]	0x0d	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be sent.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignalGroup copies the content of the associated shadow buffer to the associated I-PDU.	
Available via	Com.h	

⌋

[SWS_Com_00635] ⌈The service `Com_SendSignalGroup` shall copy the content of the shadow buffer referenced by parameter `SignalGroupId` to the associated I-PDU.⌋

8.3.3.6 Com_ReceiveSignalGroup

[SWS_Com_00201] Definition of API function Com_ReceiveSignalGroup ⌈

Service Name	Com_ReceiveSignalGroup	
Syntax	`uint8 Com_ReceiveSignalGroup ( Com_SignalGroupIdType SignalGroupId )`	
Service ID [hex]	0x0e	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be received.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_ReceiveSignalGroup copies the received signal group from the I-PDU to the shadow buffer.	
Available via	Com.h	

⌋

[SWS_Com_00638] ⌈The service Com_ReceiveSignalGroup shall copy the received signal group from the I-PDU to the shadow buffer.⌋

After this call, the group signals could be copied from the shadow buffer to the users (e.g. RTE, SwCluC) by calling Com_ReceiveSignal.

[SWS_Com_00461] ⌈The AUTOSAR COM module shall always copy the last known data, or the ComSignalInitValue if not yet written, of the I-PDU to the shadow buffer by a call to Com_ReceiveSignalGroup even if the I-PDU is stopped and COM_SERVICE_NOT_AVAILABLE is returned.⌋

8.3.3.7 Com_SendSignalGroupArray

[SWS_Com_00851] Definition of API function Com_SendSignalGroupArray ⌈

Service Name	Com_SendSignalGroupArray	
Syntax	`uint8 Com_SendSignalGroupArray ( Com_SignalGroupIdType SignalGroupId, const uint8* SignalGroupArrayPtr )`	
Service ID [hex]	0x23	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be sent.
	SignalGroupArrayPtr	Reference to the signal group array.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group was stopped COM_BUSY: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignalGroupArray copies the content of the provided SignalGroupArray Ptr to the associated I-PDU. The provided data shall correspond to the array representation of the signal group.	
Available via	Com.h	

⌋

[SWS_Com_00852] ⌈The service Com_SendSignalGroupArray shall copy the content designated by the SignalGroupArrayPtr to the associated I-PDU.⌋

[SWS_Com_00853] ⌈The Com_SendSignalGroupArray API shall only be available when the configuration switch ComEnableSignalGroupArrayApi is enabled.⌋

8.3.3.8 Com_ReceiveSignalGroupArray

[SWS_Com_00854] Definition of API function Com_ReceiveSignalGroupArray ⌈

Service Name	Com_ReceiveSignalGroupArray	
Syntax	`uint8 Com_ReceiveSignalGroupArray ( Com_SignalGroupIdType SignalGroupId, uint8* SignalGroupArrayPtr )`	
Service ID [hex]	0x24	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be received.
Parameters (inout)	None	
Parameters (out)	SignalGroupArrayPtr	reference to the location where the received signal group array shall be stored
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_ReceiveSignalGroupArray copies the received signal group array representation from the PDU to the location designated by SignalGroupAr-rayPtr.	
Available via	Com.h	

⌋

[SWS_Com_00855] ⌈The service `Com_ReceiveSignalGroupArray` shall copy the array representation of the received signal group from the I-PDU to the location designated by `SignalGroupArrayPtr`.⌋

[SWS_Com_00856] ⌈The `Com_ReceiveSignalGroupArray` API shall only be available when the configuration switch `ComEnableSignalGroupArrayApi` is enabled.⌋

[SWS_Com_00857] ⌈The AUTOSAR COM module shall always copy the last known data, or the `ComSignalInitValue` if not yet written, of the I-PDU to the `SignalGroupArrayPtr` by a call to `Com_ReceiveSignalGroupArray` even if the I-PDU is stopped and COM_SERVICE_NOT_AVAILABLE is returned.⌋

8.3.3.9 Com_ReceiveSignalWithMetaData

[SWS_Com_91005] Definition of API function Com_ReceiveSignalWithMetaData ⌈

Service Name	Com_ReceiveSignalWithMetaData
Syntax	`uint8 Com_ReceiveSignalWithMetaData ( Com_SignalIdType SignalId, void* SignalDataPtr, uint8* MetaDataPtr )`

▽

△

Service ID [hex]	0x49	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be received.
Parameters (inout)	None	
Parameters (out)	SignalDataPtr	Reference to the location where the received signal data shall be stored
	MetaDataPtr	Pointer to the meta data of the signal
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	Com_ReceiveSignalWithMetaData copies the data of the signal identified by SignalId to the location specified by SignalDataPtr. The received meta data is stored at the position given by the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.10 Com_ReceiveDynSignalWithMetaData

[SWS_Com_91006] Definition of API function Com_ReceiveDynSignalWithMeta Data ⌈

Service Name	Com_ReceiveDynSignalWithMetaData	

Syntax	`uint8 Com_ReceiveDynSignalWithMetaData (`
	`  Com_SignalIdType SignalId,`
	`  void* SignalDataPtr,`
	`  uint32* Length,`
	`  uint8* MetaDataPtr`
	`)`	

Service ID [hex]	0x4a	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be received.
Parameters (inout)	Length	in: maximum length that could be received out: length of the dynamic length signal
Parameters (out)	SignalDataPtr	reference to the location where the received signal data shall be stored
	MetaDataPtr	Pointer to the meta data of the signal
Return value	uint8	`E_OK`: service has been accepted `E_NOT_OK`: in case the Length (as in-parameter) is smaller than the received length of the dynamic length signal `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked
Description	Com_ReceiveDynSignalWithMetaData copies the data of the signal identified by SignalId to the location specified by SignalDataPtr and stores the length of the dynamical length signal at the position given by the Length parameter. The received meta data is stored at the position given by the MetaDataPtr parameter.	

▽

△

Available via	Com.h

⌋

8.3.3.11 Com_ReceiveSignalGroupWithMetaData

[SWS_Com_91007] Definition of API function Com_ReceiveSignalGroupWithMetaData ⌈

Service Name	Com_ReceiveSignalGroupWithMetaData	

Syntax	`uint8 Com_ReceiveSignalGroupWithMetaData (`
	`  Com_SignalGroupIdType SignalGroupId,`
	`  uint8* MetaDataPtr`
	`)`	

Service ID [hex]	0x4b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be received.
Parameters (inout)	None	
Parameters (out)	MetaDataPtr	Pointer to the meta data of the signal group
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_ReceiveSignalGroupWithMetaData copies the received signal group from the I-PDU to the shadow buffer. The received meta data is stored at the position given by the Meta DataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.12 Com_ReceiveSignalGroupArrayWithMetaData

[SWS_Com_91008] Definition of API function Com_ReceiveSignalGroupArrayWithMetaData ⌈

Service Name	Com_ReceiveSignalGroupArrayWithMetaData	

Syntax	`uint8 Com_ReceiveSignalGroupArrayWithMetaData (`
	`  Com_SignalGroupIdType SignalGroupId,`
	`  uint8* SignalGroupArrayPtr,`
	`  uint8* MetaDataPtr`
	`)`	

Service ID [hex]	0x4c	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be received.

▽

△

Parameters (inout)	None	
Parameters (out)	SignalGroupArrayPtr	reference to the location where the received signal group array shall be stored
	MetaDataPtr	Pointer to the meta data of the signal group.
Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group was stopped COM_BUSY: in case the TP-Buffer is locked for large data types handling
Description	The service Com_ReceiveSignalGroupArrayWithMetaData copies the received signal group array representation from the PDU to the location designated by SignalGroupAr-rayPtr. The received meta data is stored at the position given by the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.13 Com_SendSignalWithMetaData

[SWS_Com_91009] Definition of API function Com_SendSignalWithMetaData ⌈

Service Name	Com_SendSignalWithMetaData	

Syntax	`uint8 Com_SendSignalWithMetaData (`
	`  Com_SignalIdType SignalId,`
	`  const void* SignalDataPtr,`
	`  const uint8* MetaDataPtr`
	`)`	

Service ID [hex]	0x4d	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be sent.
	SignalDataPtr	Reference to the signal data to be transmitted.
	MetaDataPtr	Pointer to the meta data of the signal
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group was stopped COM_BUSY: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignalWithMetaData updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter. The meta data is provided via the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.14 Com_SendDynSignalWithMetaData

[SWS_Com_91010] Definition of API function Com_SendDynSignalWithMetaData ⌈

Service Name	Com_SendDynSignalWithMetaData	
Syntax	`uint8 Com_SendDynSignalWithMetaData ( Com_SignalIdType SignalId, const void* SignalDataPtr, uint32 Length, const uint8* MetaDataPtr )`	
Service ID [hex]	0x4e	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be sent.
	SignalDataPtr	Reference to the signal data to be transmitted.
	Length	Length of the dynamic length signal
	MetaDataPtr	Pointer to the meta data of the signal
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `E_NOT_OK`: in case the Length is greater than the configured ComSignalLength of this sent signal `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked
Description	The service Com_SendDynSignalWithMetaData updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter. The meta data is provided via the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.15 Com_SendSignalGroupWithMetaData

[SWS_Com_91011] Definition of API function Com_SendSignalGroupWithMeta Data ⌈

Service Name	Com_SendSignalGroupWithMetaData	
Syntax	`uint8 Com_SendSignalGroupWithMetaData ( Com_SignalGroupIdType SignalGroupId, const uint8* MetaDataPtr )`	
Service ID [hex]	0x4f	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be sent.
	MetaDataPtr	Pointer to the meta data of the signal group
Parameters (inout)	None	

▽

△

Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignalGroupWithMetaData copies the content of the associated shadow buffer to the associated I-PDU. The meta data is provided via the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.16 Com_SendSignalGroupArrayWithMetaData

[SWS_Com_91012] Definition of API function Com_SendSignalGroupArrayWithMetaData ⌈

Service Name	Com_SendSignalGroupArrayWithMetaData	

Syntax	`uint8 Com_SendSignalGroupArrayWithMetaData (`
	`  Com_SignalGroupIdType SignalGroupId,`
	`  const uint8* SignalGroupArrayPtr,`
	`  const uint8* MetaDataPtr`
	`)`	

Service ID [hex]	0x50	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be sent.
	SignalGroupArrayPtr	Reference to the signal group array.
	MetaDataPtr	Pointer to the meta data of the signal group.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group was stopped `COM_BUSY`: in case the TP-Buffer is locked for large data types handling
Description	The service Com_SendSignalGroupArrayWithMetaData copies the content of the provided SignalGroupArrayPtr to the associated I-PDU. The provided data shall correspond to the array representation of the signal group. The meta data is provided via the MetaDataPtr parameter.	
Available via	Com.h	

⌋

8.3.3.17 Com_InvalidateSignal

[SWS_Com_00203] Definition of API function Com_InvalidateSignal ⌈

Service Name	Com_InvalidateSignal	
Syntax	`uint8 Com_InvalidateSignal ( Com_SignalIdType SignalId )`	
Service ID [hex]	0x10	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalId	Id of signal to be invalidated.
Parameters (inout)	None	
Parameters (out)	None	
Return value	uint8	`E_OK`: service has been accepted `COM_SERVICE_NOT_AVAILABLE`: corresponding I-PDU group is stopped, no ComSignalDataInvalidValue is configured for the given signalId `COM_BUSY`: in case the TP-Buffer is locked
Description	The service Com_InvalidateSignal invalidates the signal with the given SignalId by setting its value to its configured ComSignalDataInvalidValue.	
Available via	Com.h	

⌋

See Chapter 7.2.4 for further information about the signal invalidation mechanism.

[SWS_Com_00642] ⌈After invaliding the signal data, the function `Com_InvalidateSignal` shall perform a call of `Com_SendSignal` internally, for details see [SWS_Com_00099].⌋

[SWS_Com_00643] ⌈`Com_InvalidateSignal` shall return COM_SERVICE_NOT_AVAILABLE in case no `ComSignalDataInvalidValue` is configured for the signal with the given `SignalId`.⌋

8.3.3.18 Com_InvalidateSignalGroup

[SWS_Com_00557] Definition of API function Com_InvalidateSignalGroup ⌈

Service Name	Com_InvalidateSignalGroup	
Syntax	`uint8 Com_InvalidateSignalGroup ( Com_SignalGroupIdType SignalGroupId )`	
Service ID [hex]	0x1b	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SignalGroupId	Id of signal group to be invalidated.
Parameters (inout)	None	
Parameters (out)	None	

▽

△

Return value	uint8	E_OK: service has been accepted COM_SERVICE_NOT_AVAILABLE: corresponding I-PDU group is stopped, no ComSignalDataInvalidValue is configured for any of the group signals COM_BUSY: in case the TP-Buffer is locked
Description	The service Com_InvalidateSignalGroup invalidates all group signals of the signal group with the given SignalGroupId by setting their values to their configured ComSignalDataInvalidValues.	
Available via	Com.h	

⌋

For further information about the signal invalidation mechanism, see Chapter 7.2.4.

[SWS_Com_00645] ⌈After invaliding the signal group data, the function Com_InvalidateSignalGroup shall perform a call of Com_SendSignalGroup internally.⌋

8.3.3.19 Com_TriggerIPDUSend

[SWS_Com_00348] Definition of API function Com_TriggerIPDUSend ⌈

Service Name	Com_TriggerIPDUSend	
Syntax	`Std_ReturnType Com_TriggerIPDUSend ( PduIdType PduId )`	
Service ID [hex]	0x17	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	PduId	The I-PDU-ID of the I-PDU that shall be triggered for sending
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: I-PDU was triggered for transmission E_NOT_OK: I-PDU is stopped, the transmission could not be triggered
Description	By a call to Com_TriggerIPDUSend the I-PDU with the given ID is triggered for transmission.	
Available via	Com.h	

⌋

[SWS_Com_00861] ⌈By a call to Com_TriggerIPDUSend, the AUTOSAR COM shall trigger the I-PDU with the given ID for transmission only if the I-PDU is started and ComIPduLargeData is set to false.⌋

In case a stopped I-PDU is triggered for transmission, just E_NOT_OK is returned. For stopped I-PDUs no triggers are stored for a later execution after a potential (re-) starting of the I-PDU.

[SWS_Com_00930] Com_TriggerIPDUSend and Com_TriggerIPDUSendWithMetaData Excluded for Large Data Path ⌈If invoked for a ComIPdu with ComIPduLargeData set to true, Com_TriggerIPDUSend and Com_TriggerIPDUSendWithMetaData shall return E_NOT_OK.⌋

[SWS_Com_00388] ⌈When an I-PDU is transmitted because of the function Com_TriggerIPDUSend or Com_TriggerIPDUSendWithMetaData, the AUTOSAR COM module shall only take the minimum delay time into account. That is the AUTOSAR COM module shall postpone transmissions if necessary and reset the minimum delay timer in case of transmissions. The AUTOSAR COM module shall not take into account all other transmission mode related parameters like ComTxModeNumberOfRepetitions.⌋

[SWS_Com_00492] ⌈If an I-PDU triggered by Com_TriggerIPDUSend or Com_TriggerIPDUSendWithMetaData has a configured ComIPduCallout, then the function Com_TriggerIPDUSend or Com_TriggerIPDUSendWithMetaData repectively shall also call this I-PDU-Callout.⌋

If within the Com_TriggerIPDUSend function, another call to Com_TriggerIPDUSend is invoked the application programmer must take care to not create infinite recursions. Both, direct recursions by invoking Com_TriggerIPDUSend with the same ComPduId again and indirect recursions with longer recursion paths are theoretically possible.

The previous paragraph is also true for Com_TriggerIPDUSendWithMetaData as well as for combinations of these two I-PDU triggering functions.

[SWS_Com_00867] ⌈In case a large I-PDU is triggered for transmission by Com_TriggerIPDUSend and the buffer of the large I-PDU is currently locked, the AUTOSAR COM shall store the transmit request and transmit the I-PDU when the buffer is available again.⌋

8.3.3.20 Com_TriggerIPDUSendWithMetaData

[SWS_Com_00858] Definition of API function Com_TriggerIPDUSendWithMetaData ⌈

Service Name	Com_TriggerIPDUSendWithMetaData
Syntax	`Std_ReturnType Com_TriggerIPDUSendWithMetaData ( PduIdType PduId, const uint8* MetaData )`
Service ID [hex]	0x28
Sync/Async	Synchronous
Reentrancy	Non Reentrant

▽

△

Parameters (in)	PduId	The I-PDU-ID of the I-PDU that shall be triggered for sending
	MetaData	Pointer to the meta data of the signal
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: I-PDU was triggered for transmission `E_NOT_OK`: I-PDU is stopped, the transmission could not be triggered
Description	By a call to Com_TriggerIPDUSendWithMetaData the AUTOSAR COM module updates its internal metadata for the I-PDU with the given ID by copying the metadata from the given position and with respect to length of the globally configured MetaDataType of this I-PDU. Then the I-PDU is triggered for transmission.	
Available via	Com.h	

⌋

[SWS_Com_00862] ⌈By a call to `Com_TriggerIPDUSendWithMetaData`, the AUTOSAR COM shall trigger the I-PDU with the given ID for transmission only if the I-PDU is started and `ComIPduLargeData` is set to false.⌋

In case a stopped I-PDU is triggered for transmission, just `E_NOT_OK` is returned. There is no mechanism in COM to store the triggering and to execute it after a potential (re-)starting of the I-PDU.

[SWS_Com_00868] ⌈In case a large I-PDU is triggered for transmission by `Com_TriggerIPDUSendWithMetaData` and the buffer of the large I-PDU is currently locked, the AUTOSAR COM shall store the transmit request with the given `MetaData` and transmit the I-PDU when the buffer is available again.⌋

8.3.3.21 Com_SwitchIpduTxMode

[SWS_Com_00784] Definition of API function Com_SwitchIpduTxMode ⌈

Service Name	Com_SwitchIpduTxMode	
Syntax	`void Com_SwitchIpduTxMode ( PduIdType PduId, boolean Mode )`	
Service ID [hex]	0x27	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	PduId	Id of the I-PDU of which the transmission mode shall be changed.
	Mode	the transmission mode that shall be set.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	The service Com_SwitchIpduTxMode sets the transmission mode of the I-PDU referenced by PduId to Mode. In case the transmission mode changes, the new mode shall immediately be effective (see [SWS_Com_00239]). In case the requested transmission mode was already active for this I-PDU, the call will have no effect.
Available via	Com.h

⌋

[SWS_Com_00881] ⌈When the transmission mode of an I-PDU is explicitly set by `Com_SwitchIpduTxMode`, the AUTOSAR COM shall defer the cyclic transmissions of this I-PDU by `ComTxModeTimeOffset`.⌋

By mixing the signal based TMS and explicit TMS by `Com_SwitchIpduTxMode` for the same I-PDU, it must be noted that the signal base TMS might switch back the manual set mode during a call to `Com_SendSignal` or `Com_SendSignalGroup` for this I-PDU. Therefore, mixing explicit and manual TMS for one I-PDU must be carefully designed, if used at all.

The new transmission mode is immediately effective (within the next main function at the latest), however, ongoing TP transmission will be finished. In this case if the new transmission mode would necessitate a new transmission then `COM_E_SKIPPED_TRANSMISSION` runtime error will be reported (see [SWS_Com_00863]).

8.3.3.22 Com_Transmit

[SWS_Com_00931] Definition of API function Com_Transmit ⌈

Service Name	Com_Transmit	
Syntax	`Std_ReturnType Com_Transmit ( PduIdType TxPduId, const PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x51	
Sync/Async	Synchronous	
Reentrancy	Reentrant Reentrant for different TxPduIds. Non reentrant for the same TxPduId.	
Parameters (in)	TxPduId	Identifier of the I-PDU to be transmitted.
	PduInfoPtr	Length of and pointer to the signal data and pointer to MetaData.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Transmit request has been accepted. `COM_BUSY`: Transmit request has been rejected due to minimum delay time violation. `E_NOT_OK`: Transmit request has not been accepted.
Description	Requests transmission of an I-PDU with Large Data Path.	
Available via	Com.h	

⌋

[SWS_Com_00932] Com_Transmit Available for Large Data Path I-PDUs Only ⌈If invoked for a ComIPdu with ComIPduLargeData set to false, Com_Transmit shall return E_NOT_OK.⌋

[SWS_Com_00933] Com_Transmit Behavior for Stopped I-PDUs ⌈Com_Transmit shall return E_NOT_OK in case the given TxPduId refers to a stopped I-PDU.⌋

8.3.3.23 Com_ReleaseRxBuffer

[SWS_Com_91033] Definition of API function Com_ReleaseRxBuffer ⌈

Service Name	Com_ReleaseRxBuffer	
Syntax	`void Com_ReleaseRxBuffer ( PduIdType RxPduId )`	
Service ID [hex]	0x53	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId	
Parameters (in)	RxPduId	Identifier of the received PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication from the upper layer to release the lower layer reception buffer.	
Available via	Com.h	

⌋

8.3.3.24 Com_TriggerTransmit

[SWS_Com_00001] Definition of callback function Com_TriggerTransmit ⌈

Service Name	Com_TriggerTransmit	
Syntax	`Std_ReturnType Com_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	Com.h	

⌋

[SWS_Com_00647] ⌈Within the function Com_TriggerTransmit, the AUTOSAR COM module shall copy the contents of its I-PDU transmit buffer to the L-PDU buffer given by PduInfoPtr->SduDataPtr and update PduInfoPtr->SduLength with length of the copied data accordingly.⌋

[SWS_Com_00869] ⌈In case the given PduInfoPtr->SduLength is smaller than the actual PDU-length, Com_TriggerTransmit shall not copy any data and return E_NOT_OK.⌋

[SWS_Com_00884] Com_TriggerTransmit Behavior for Stopped I-PDUs ⌈Com_TriggerTransmit shall not copy any data and return E_NOT_OK if the I-PDU is stopped.⌋

[SWS_Com_00475] ⌈Com_TriggerTransmit is not interfered by the I-PDU minimum delay time and shall not reset the minimum delay timer, see ComMinimumDelayTime.⌋

The function Com_TriggerTransmit behaves independent of the configured transmission mode.

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. The FlexRay Interface for requesting PDUs to be sent in static part (synchronous to the FlexRay global time) also uses this function.

Once the PDU Router has successfully sent the I-PDU, the PDU Router will call Com_TxConfirmation.

Caveats of Com_TriggerTransmit: This function might be called in interrupt context.

8.3.3.25 Com_RxIndication

[SWS_Com_00123] Definition of callback function Com_RxIndication ⌈

Service Name	Com_RxIndication	

Syntax	`void Com_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	Com.h	

⌋

Caveats of `Com_RxIndication`: This function might be called in interrupt context.

[SWS_Com_00934] Com_RxIndication Behavior for Stopped I-PDUs ⌈AUTOSAR COM shall ignore `Com_RxIndication` call in case the given `RxPduId` refers to a stopped I-PDU.⌋

8.3.3.26 Com_TpRxIndication

[SWS_Com_00650] Definition of callback function Com_TpRxIndication ⌈

Service Name	Com_TpRxIndication	

Syntax	`void Com_TpRxIndication (`
	`  PduIdType id,`
	`  Std_ReturnType result`
	`)`	

Service ID [hex]	0x45	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the received I-PDU.
	result	`E_OK`: The PDU was received successfully. `E_NOT_OK`: Reception of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	Called after an I-PDU has been received via the TP API, the result indicates whether the reception was successful or not.
Available via	Com.h

⌋

Caveats of `Com_TpRxIndication`: This function might be called in interrupt context.

[SWS_Com_00935] Com_TpRxIndication Behavior for Stopped I-PDUs ⌈AUTOSAR COM shall ignore `Com_TpRxIndication` call in case the given `id` refers to a stopped I-PDU.⌋

8.3.3.27 Com_TxConfirmation

[SWS_Com_00124] Definition of callback function Com_TxConfirmation ⌈

Service Name	Com_TxConfirmation	
Syntax	`void Com_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	Com.h	

⌋

Caveats of `Com_TxConfirmation`: This function might be called in interrupt context, e.g. from transmit interrupt.

[SWS_Com_00936] Com_TxConfirmation Behavior for Stopped I-PDUs ⌈AUTOSAR COM shall ignore `Com_TxConfirmation` call in case the given `TxPduId` refers to a stopped I-PDU.⌋

8.3.3.28 Com_TpTxConfirmation

[SWS_Com_00725] Definition of callback function Com_TpTxConfirmation ⌈

Service Name	Com_TpTxConfirmation	

Syntax	`void Com_TpTxConfirmation (`
	`  PduIdType id,`
	`  Std_ReturnType result`
	`)`	

Service ID [hex]	0x48	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the transmitted I-PDU.
	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	This function is called after the I-PDU has been transmitted on its network, the result indicates whether the transmission was successful or not.	
Available via	Com.h	

⌋

[SWS_Com_00937] Com_TpTxConfirmation Behavior for Stopped I-PDUs ⌈AUTOSAR COM shall ignore `Com_TpTxConfirmation` call in case the given `id` refers to a stopped I-PDU.⌋

8.3.3.29 Com_StartOfReception

[SWS_Com_00691] Definition of callback function Com_StartOfReception ⌈

Service Name	Com_StartOfReception	

Syntax	`BufReq_ReturnType Com_StartOfReception (`
	`  PduIdType id,`
	`  const PduInfoType* info,`
	`  PduLengthType TpSduLength,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Service ID [hex]	0x46	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the I-PDU.
	info	Pointer to a PduInfoType structure containing the payload data (without protocol information) and payload length of the first frame or single frame of a transport protocol I-PDU reception, and the MetaData related to this PDU. If neither first/single frame data nor MetaData are available, this parameter is set to NULL_PTR.
	TpSduLength	Total length of the N-SDU to be received.
Parameters (inout)	None	

▽

△

Parameters (out)	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
Return value	BufReq_ReturnType	BUFREQ_OK: Connection has been accepted. bufferSizePtr indicates the available receive buffer; reception is continued. If no buffer of the requested size is available, a receive buffer size of 0 shall be indicated by bufferSizePtr. BUFREQ_E_NOT_OK: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged. BUFREQ_E_OVFL: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.
Description	This function is called at the start of receiving an N-SDU. The N-SDU might be fragmented into multiple N-PDUs (FF with one or more following CFs) or might consist of a single N-PDU (SF). The service shall provide the currently available maximum buffer size when invoked with TpSdu Length equal to 0.	
Available via	Com.h	

⌋

[SWS_Com_00654] ⌈If id corresponds to a large I-PDU and TpSduLength is greater than the configured length of this I-PDU then the function Com_StartOfReception shall return BUFREQ_E_OVFL.⌋

[SWS_Com_00655] ⌈If id corresponds to a dynamic length I-PDU and TpSduLength is greater than the configured maximum length of this I-PDU then the function Com_StartOfReception shall return BUFREQ_E_OVFL.⌋

[SWS_Com_00657] ⌈If the internal AUTOSAR COM module's buffer is currently not locked and the parameter TpSduLength does not exceed the (maximum) expected length, the function Com_StartOfReception shall write the complete size of the AUTOSAR COM module's internal I-PDU buffer of the I-PDU with handle id to the position given by bufferSizePtr.⌋

[SWS_Com_00721] Com_StartOfReception Behavior for Stopped I-PDUs ⌈Com_StartOfReception shall return BUFREQ_E_NOT_OK, in case id refers to a stopped I-PDU.⌋

[SWS_Com_00818] ⌈In case Com_StartOfReception returns BUFREQ_E_NOT_OK or BUFREQ_E_OVFL, the AUTOSAR COM module shall not expect a call to Com_TpRxIndication and hence shall not lock its internal buffer.⌋

[SWS_Com_00874] ⌈In case Com_StartOfReception is called with TpSduLength equal to 0, the AUTOSAR COM module shall return the configured (maximum) length of this I-PDU via the bufferSizePtr.⌋

A TpSduLength equal to 0 indicates an unknown message length.

8.3.3.30 Com_CopyRxData

[SWS_Com_00692] Definition of callback function Com_CopyRxData ⌈

Service Name	Com_CopyRxData	
Syntax	`BufReq_ReturnType Com_CopyRxData ( PduIdType id, const PduInfoType* info, PduLengthType* bufferSizePtr )`	
Service ID [hex]	0x44	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the received I-PDU.
	info	Provides the source buffer (SduDataPtr) and the number of bytes to be copied (SduLength). An SduLength of 0 can be used to query the current amount of available buffer in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
Parameters (inout)	None	
Parameters (out)	bufferSizePtr	Available receive buffer after data has been copied.
Return value	BufReq_ReturnType	`BUFREQ_OK`: Data copied successfully `BUFREQ_E_NOT_OK`: Data was not copied because an error occurred.
Description	This function is called to provide the received data of an I-PDU segment (N-PDU) to the upper layer. Each call to this function provides the next part of the I-PDU data. The size of the remaining buffer is written to the position indicated by bufferSizePtr.	
Available via	Com.h	

⌋

Caveats of `Com_CopyRxData`: This function might be called in interrupt context. Therefore, data consistency must be ensured.

[SWS_Com_00782] Com_CopyRxData Behavior for Stopped I-PDUs ⌈`Com_CopyRxData` shall copy no data and return BUFREQ_E_NOT_OK in case the given `id` refers to a stopped I-PDU.⌋

8.3.3.31 Com_CopyTxData

[SWS_Com_00693] Definition of callback function Com_CopyTxData ⌈

Service Name	Com_CopyTxData	
Syntax	`BufReq_ReturnType Com_CopyTxData ( PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr )`	
Service ID [hex]	0x43	
Sync/Async	Synchronous	
Reentrancy	Reentrant	

▽

△

Parameters (in)	id	Identification of the transmitted I-PDU.
	info	Provides the destination buffer (SduDataPtr) and the number of bytes to be copied (SduLength). If not enough transmit data is available, no data is copied by the upper layer module and BUFREQ_E_BUSY is returned. The lower layer module may retry the call. An SduLength of 0 can be used to indicate state changes in the retry parameter or to query the current amount of available data in the upper layer module. In this case, the Sdu DataPtr may be a NULL_PTR.
	retry	This parameter is used to acknowledge transmitted data or to retransmit data after transmission problems. If the retry parameter is a NULL_PTR, it indicates that the transmit data can be removed from the buffer immediately after it has been copied. Otherwise, the retry parameter must point to a valid RetryInfoType element. If TpDataState indicates TP_CONFPENDING, the previously copied data must remain in the TP buffer to be available for error recovery. TP_DATACONF indicates that all data that has been copied before this call is confirmed and can be removed from the TP buffer. Data copied by this API call is excluded and will be confirmed later. TP_DATARETRY indicates that this API call shall copy previously copied data in order to recover from an error. In this case TxTpDataCnt specifies the offset in bytes from the current data copy position.
Parameters (inout)	None	
Parameters (out)	availableDataPtr	Indicates the remaining number of bytes that are available in the upper layer module's Tx buffer. availableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. FrIsoTp) to determine the size of the following CFs.
Return value	BufReq_ReturnType	BUFREQ_OK: Data has been copied to the transmit buffer completely as requested. BUFREQ_E_BUSY: Request could not be fulfilled, because the required amount of Tx data is not available. The lower layer module may retry this call later on. No data has been copied. BUFREQ_E_NOT_OK: Data has not been copied. Request failed.
Description	This function is called to acquire the transmit data of an I-PDU segment (N-PDU). Each call to this function provides the next part of the I-PDU data unless retry->TpDataState is TP_DATARETRY. In this case the function restarts to copy the data beginning at the offset from the current position indicated by retry->TxTpDataCnt. The size of the remaining data is written to the position indicated by availableDataPtr.	
Available via	Com.h	

⌋

Caveats of Com_CopyTxData: This function might be called in interrupt context.

[SWS_Com_00783] Com_CopyTxData Behavior for Stopped I-PDUs ⌈Com_CopyTxData shall copy no data and return BUFREQ_E_NOT_OK in case the given id refers to a stopped I-PDU.⌋

8.4 Scheduled functions

8.4.1 Com_MainFunctionRx

[SWS_Com_00398] Definition of scheduled function Com_MainFunctionRx

Upstream requirements: SRS_BSW_00425, SRS_BSW_00432

⌈

Service Name	Com_MainFunctionRx

Syntax	`void Com_MainFunctionRx (`
	`  void`
	`)`

Service ID [hex]	0x18
Description	This function performs the processing of the AUTOSAR COM module's receive processing that are not directly handled within the COM's functions invoked by the PDU-R, for example Com_RxIndication. Per configured ComMainFunctionRx instance one Com_MainFunctionRx_<shortName> shall be implemented. Hereby <shortName> is the short name of the ComMainFunctionRx configuration container in the ECU configuration.
Available via	SchM_Com.h

⌋

[SWS_Com_00664]

Upstream requirements: SRS_BSW_00425

⌈A call to `Com_MainFunctionRx` shall simply return if the AUTOSAR COM module was not previously initialized with a call to `Com_Init`.⌋

Configuration of `Com_MainFunctionRx`: see `ComMainFunctionRx`.

8.4.2 Com_MainFunctionTx

[SWS_Com_00399] Definition of scheduled function Com_MainFunctionTx

Upstream requirements: SRS_BSW_00425, SRS_BSW_00432

⌈

Service Name	Com_MainFunctionTx

Syntax	`void Com_MainFunctionTx (`
	`  void`
	`)`

Service ID [hex]	0x19
Description	This function performs the processing of the AUTOSAR COM module's transmission activities that are not directly handled within the COM's function invoked by the users (e.g. RTE, SwClu C), for example Com_SendSignal. Per configured ComMainFunctionTx instance one Com_MainFunctionTx_<shortName> shall be implemented. Hereby <shortName> is the short name of the ComMainFunctionTx configuration container in the ECU configuration.
Available via	SchM_Com.h

⌋

[SWS_Com_00665]

Upstream requirements: SRS_BSW_00425

⌈A call to `Com_MainFunctionTx` shall simply return if the AUTOSAR COM module was not previously initialized with a call to `Com_Init`.⌋

Configuration of `Com_MainFunctionTx`: see `ComMainFunctionTx`.

8.4.3 Com_MainFunctionRouteSignals

[SWS_Com_00400] Definition of scheduled function Com_MainFunctionRouteSignals

Upstream requirements: SRS_BSW_00425, SRS_BSW_00432

⌈

Service Name	Com_MainFunctionRouteSignals
Syntax	`void Com_MainFunctionRouteSignals ( void )`
Service ID [hex]	0x1a
Description	Calls the signal gateway part of the AUTOSAR COM module to forward received signals to be routed. Per configured ComMainFunctionRouteSignals instance one Com_MainFunctionRouteSignals_<shortName> shall be implemented. Hereby <shortName> is the short name of the ComMainFunctionRouteSignals configuration container in the ECU configuration.
Available via	SchM_Com.h

⌋

[SWS_Com_00666]

Upstream requirements: SRS_BSW_00425

⌈A call to `Com_MainFunctionRouteSignals` shall simply return if the AUTOSAR COM module was not previously initialized with a call to `Com_Init`.⌋

Caveats of `Com_MainFunctionRouteSignals`:

- The time between two consecutive calls (perhaps the related task/thread cycle) affects directly the signal gateway latency.
- The insertion of this call is necessary for decoupling receive interrupts and signal gateway tasks.
- Normally a cyclic task / thread to call this function is configured. The cycle of this task / thread directly affects the latency of the signal gateway.

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, which are required to fulfill the core functionality of the module.

[SWS_Com_00669] Definition of mandatory interfaces required by module Com

Upstream requirements: SRS_BSW_00384

⌈

API Function	Header File	Description
PduR_ComTransmit	PduR_Com.h	Requests transmission of a PDU.

⌋

8.5.2 Optional interfaces

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

[SWS_Com_00670] Definition of optional interfaces requested by module Com

Upstream requirements: SRS_BSW_00384

⌈

API Function	Header File	Description
Dem_SetEventStatus	Dem.h	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value. This API will be available only if ({Dem/Dem ConfigSet/DemEventParameter/DemEvent ReportingType} == STANDARD_REPORTING)
Det_ReportError	Det.h	Service to report development errors.
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.
IdsM_ReportSecurityEvent	IdsM.h	This API is the interface to report security events to the IdsM.
PduR_ComCancelTransmit	PduR_Com.h	Requests cancellation of an ongoing transmission of a PDU in a lower layer communication module.
PduR_ComReleaseRxBuffer (draft)	PduR_Com.h	Indication from the upper layer to release the lower layer reception buffer.

⌋

The associated configuration parameter for API function `Det_ReportError` is `ComConfigurationUseDet`.

[SWS_Com_00442]

Upstream requirements: SRS_BSW_00369, SRS_BSW_00385, SRS_BSW_00327

⌈When a development error is detected, the function `Det_ReportError` of the Default Error Tracer shall be called with:

- 50 as the AUTOSAR COM's ModuleId

- 0 as InstanceId
- the service ID of the AUTOSAR COM module's API in which the error was detected as ApiId
- the error ID

⌋

See also Chapter 7.12.1 for error IDs.

The API function `PduR_ComCancelTransmit` must be included if at least one I-PDU has configured `ComIPduCancellationSupport` to true.

8.5.3 Configurable interfaces

Caveats of configurable interfaces: A callback routine runs either on interrupt level or on task level. Thus, the OS restrictions of usage of system functions for interrupt service routines as well as for tasks apply.

Callbacks are configured per Com user. It is possible that one Com user has a specific type of callback whereas another Com user does not have it. Examples of Com users are the RTE and the SwCluC Com Proxy.

API Name	ECUC Parameter
<ComUser_CbkRxAck>	ComUserCallbackName with ComUserCallbackType set to COM_RX_ACK
<ComUser_CbkInv>	ComUserCallbackName with ComUserCallbackType set to COM_RX_INV
<ComUser_CbkRxTOut>	ComUserCallbackName with ComUserCallbackType set to COM_RX_TOUT
<ComUser_CbkTxAck>	ComUserCallbackName with ComUserCallbackType set to COM_TX_ACK
<ComUser_CbkTxErr>	ComUserCallbackName with ComUserCallbackType set to COM_TX_ERR
<ComUser_CbkTxTOut>	ComUserCallbackName with ComUserCallbackType set to COM_TX_TOUT

Table 8.1: API to Parameter mapping (Large Data Path not used)

Regarding Large Data Path callbacks, see Table 7.6: API to Parameter mapping for the configuration of the actual API names.

[SWS_Com_91016] Definition of configurable interface <ComUser_CbkTxAck>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkTxAck>	

Syntax	`void <ComUser_CbkTxAck> (`
	`  CbkHandleIdType ComUserCbkHandleId`
	`)`	

Sync/Async	Synchronous	
Reentrancy	don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the signal/signal group, which transmission is acknowledged
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This callback represents Notification Class 2 in Clause 3.6.2 (Notification classes) of [2]. It is called immediately after successful transmission of the I-PDU containing the message.	
Available via	Configuration parameter ComUserHeaderInclude	

⌋

<ComUser_CbkTxAck> is called on sender side only. It can be configured for signals and signal groups. This callback function corresponds to Rte_COMCbkTAck or SwCluC_ComProxy_COMCbkTAck respectively.

The callback function type (COM_TX_ACK) and name must be configured in ComUser-Callback.

[SWS_Com_91017] Definition of configurable interface <ComUser_CbkTxErr>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkTxErr>	

Syntax	`void <ComUser_CbkTxErr> (`
	`  CbkHandleIdType ComUserCbkHandleId`
	`)`	

Sync/Async	Synchronous	
Reentrancy	don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the signal/signal group, which could not be transmitted.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This callback corresponds to Notification Class 4 in Clause 3.6.2 (Notification classes) of [2]. It is called in case the transmission is not possible because the corresponding I-PDU group is stopped.	
Available via	Configuration parameter ComUserHeaderInclude	

⌋

<ComUser_CbkTxErr> is called on sender side only. This callback function corresponds to Rte_COMCbkTErr or SwCluC_ComProxy_COMCbkTErr respectively.

The callback function type (COM_TX_ERR) and name must be configured in ComUserCallback.

[SWS_Com_91018] Definition of configurable interface <ComUser_CbkTxTOut>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkTxTOut>	
Syntax	`void <ComUser_CbkTxTOut> ( CbkHandleIdType ComUserCbkHandleId )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the signal/signal group, for which a transmission deadline violation has occurred.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This callback corresponds to Notification Class 4 in Clause 3.6.2 (Notification classes) of [2]. It is called immediately after a message transmission error has been detected by the deadline monitoring mechanism.	
Available via	Configuration parameter ComUserHeaderInclude	

⌋

<ComUser_CbkTxTOut> is called on sender side only. It can be configured for signals and signal groups. This callback function corresponds to Rte_COMCbkTxTOut or SwCluC_ComProxy_COMCbkTxTOut.

The callback function type (COM_TX_TOUT) and name must be configured in ComUserCallback.

[SWS_Com_91019] Definition of configurable interface <ComUser_CbkRxAck>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkRxAck>	
Syntax	`void <ComUser_CbkRxAck> ( CbkHandleIdType ComUserCbkHandleId )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the signal/signal group, which has been received.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This callback represents Notification Class 1 in Clause 3.6.2 (Notification classes) of [2]. It is called immediately after the message has been stored in the receiving message object.	
Available via	Configuration parameter ComUserHeaderInclude	

⌋

<ComUser_CbkRxAck> is called on receiver side only. It can be configured for signals and signal groups. This callback function corresponds to Rte_COMCbk or SwCluC_ComProxy_COMCbk.

The callback function type (COM_RX_ACK) and name must be configured in ComUserCallback.

[SWS_Com_91020] Definition of configurable interface <ComUser_CbkRxTOut>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkRxTOut>	
Syntax	`void <ComUser_CbkRxTOut> ( CbkHandleIdType ComUserCbkHandleId )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the signal/signal group, for which a reception deadline violation has occurred.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This callback corresponds to Notification Class 3 in Clause 3.6.2 (Notification classes) of [2]. It is called immediately after a message reception error has been detected by the deadline monitoring mechanism.	
Available via	Configuration parameter ComUserHeaderInclude	

⌋

<ComUser_CbkRxTOut> is called on receiver side only. It can be configured for signals and signal groups. This callback function corresponds to Rte_COMCbkRxTOut or SwCluC_ComProxy_COMCbkRxTOut.

The callback function type (COM_RX_TOUT) and name must be configured in ComUserCallback.

[SWS_Com_91021] Definition of configurable interface <ComUser_CbkInv>

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	<ComUser_CbkInv>	
Syntax	`void <ComUser_CbkInv> ( CbkHandleIdType ComUserCbkHandleId )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant don't care	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id of the invalid signal/signal group.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

▽

△

Description	This callback function corresponds to [SWS_Com_00680]. It is called after reception of an invalid signal or signal group respectively.
Available via	Configuration parameter ComUserHeaderInclude

⌋

<ComUser_CbkInv> is called on receiver side only. It can be configured for signals and signal groups. This callback function corresponds to Rte_COMCbkInv or SwCluC_ComProxy_COMCbkInv.

The callback function type (COM_RX_INV) and name must be configured in ComUser-Callback.

[SWS_Com_91013] Definition of configurable interface Com_CbkTxPrep

Upstream requirements: SRS_BSW_00359, SRS_BSW_00360

⌈

Service Name	Com_CbkTxPrep

Syntax	`void Com_CbkTxPrep (`
	`  void`
	`)`

Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None
Return value	None
Description	This callback function indicates that the signals/signal groups of a dedicated Com_Main FunctionTx instance will now be prepared for transmission.
Available via	Configuration parameter ComUserHeaderInclude

⌋

Com_CbkTxPrep is called on sender side only. It can be configured per Com_MainFunctionTx instance. This callback function corresponds to Rte_COMCbkTAck_<mn>.

[SWS_Com_91024] Definition of configurable interface <ComUser_ComCbkCopyTxData>

Previous identifiers: SWS_LDCOM_91001

⌈

Service Name	<ComUser_ComCbkCopyTxData>	

Syntax	`BufReq_ReturnType <ComUser_ComCbkCopyTxData> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  const RetryInfoType* retry,`
	`  PduLengthType* availableDataPtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the transmitted I-PDU
	info	Provides the destination buffer (SduDataPtr) and the number of bytes to be copied (SduLength). If not enough transmit data is available, no data is copied by the upper layer module and BUFREQ_E_BUSY is returned. The lower layer module may retry the call. An SduLength of 0 can be used to indicate state changes in the retry parameter or to query the current amount of available data in the upper layer module. In this case, the Sdu DataPtr may be a NULL_PTR.
	retry	Will not be handled by Com and its upper layer.
Parameters (inout)	None	
Parameters (out)	availableDataPtr	Indicates the remaining number of bytes that are available in the upper layer module's Tx buffer. availableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. FrIsoTp) to determine the size of the following CFs.
Return value	BufReq_ReturnType	`BUFREQ_OK`: Data has been copied to the transmit buffer completely as requested. `BUFREQ_E_BUSY`: Request could not be fulfilled, because the required amount of Tx data is not available. The lower layer module may retry this call later on. No data has been copied. `BUFREQ_E_NOT_OK`: Data has not been copied. Request failed.
Description	This function is called to acquire the transmit data of an I-PDU segment (N-PDU). Each call to this function provides the next part of the I-PDU data unless retry->TpDataState is TP_DATARETRY. In this case the function restarts to copy the data beginning at the offset from the current position indicated by retry->TxTpDataCnt. The size of the remaining data is written to the position indicated by availableDataPtr	
Available via	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91025] Definition of configurable interface <ComUser_ComCbkTpTxConfirmation>

Previous identifiers: SWS_LDCOM_91002

⌈

Service Name	<ComUser_ComCbkTpTxConfirmation>

Syntax	`void <ComUser_ComCbkTpTxConfirmation> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  Std_ReturnType result`
	`)`

▽

△

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the transmitted I-PDU
	result	E_OK - transmission successful E_NOT_OK - transmission not successful
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This function is called after a Signal has been transmitted via the TP-API on its network.	
Available via	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91026] Definition of configurable interface <ComUser_ComCbkStartOfReception>

Previous identifiers: SWS_LDCOM_91003

⌈

Service Name	<ComUser_ComCbkStartOfReception>	

Syntax	`BufReq_ReturnType <ComUser_ComCbkStartOfReception> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  PduLengthType TpSduLength,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the I-PDU
	info	Pointer to a PduInfoType structure containing the payload data (without protocol information) and payload length of the first frame or single frame of a transport protocol I-PDU reception, and the MetaData related to this PDU. If neither first/single frame data nor MetaData are available, this parameter is set to NULL_PTR.
	TpSduLength	Total length of the N-SDU to be received.
Parameters (inout)	None	
Parameters (out)	bufferSizePtr	Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.
Return value	BufReq_ReturnType	`BUFREQ_OK`: Connection has been accepted. bufferSizePtr indicates the available receive buffer; reception is continued. If no buffer of the requested size is available, a receive buffer size of 0 shall be indicated by bufferSizePtr. `BUFREQ_E_NOT_OK`: Connection has been rejected; reception is aborted. bufferSizePtr remains unchanged. `BUFREQ_E_OVFL`: No buffer of the required length can be provided; reception is aborted. bufferSizePtr remains unchanged.
Description	This function is called at the start of receiving an N-SDU. The N-SDU might be fragmented into multiple N-PDUs (FF with one or more following CFs) or might consist of a single N-PDU (SF). The service shall provide the currently available maximum buffer size when invoked with TpSdu Length equal to 0.	
Available via	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91027] Definition of configurable interface <ComUser_ComCbkCopyRxData>

Previous identifiers: SWS_LDCOM_91004

⌈

Service Name	<ComUser_ComCbkCopyRxData>	

Syntax	`BufReq_ReturnType <ComUser_ComCbkCopyRxData> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  const PduInfoType* info,`
	`  PduLengthType* bufferSizePtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the received I-PDU
	info	Provides the source buffer (SduDataPtr) and the number of bytes to be copied (SduLength). An SduLength of 0 can be used to query the current amount of available buffer in the upper layer module. In this case, the SduDataPtr may be a NULL_PTR.
Parameters (inout)	None	
Parameters (out)	bufferSizePtr	Available receive buffer after data has been copied.
Return value	BufReq_ReturnType	`BUFREQ_OK`: Data copied successfully `BUFREQ_E_NOT_OK`: Data was not copied because an error occurred.
Description	This function is called to provide the received data of an I-PDU segment (N-PDU) to the upper layer. Each call to this function provides the next part of the I-PDU data. The size of the remaining data is written to the position indicated by bufferSizePtr.	
Available via	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91028] Definition of configurable interface <ComUser_ComCbkTpRxIndication>

Previous identifiers: SWS_LDCOM_91005

⌈

Service Name	<ComUser_ComCbkTpRxIndication>	

Syntax	`void <ComUser_ComCbkTpRxIndication> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  Std_ReturnType result`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the received I-PDU
	result	`E_OK`: The I-PDU was received successfully. `E_NOT_OK`: Reception of the I-PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called after an I-PDU has been received via the TP API, the result indicates whether the reception was successful or not.	
Available via	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91029] Definition of configurable interface <ComUser_ComCbkTriggerTransmit>

Previous identifiers: SWS_LDCOM_91007

⌈

Service Name	<ComUser_ComCbkTriggerTransmit>	
Syntax	`Std_ReturnType <ComUser_ComCbkTriggerTransmit> ( CbkHandleIdType ComUserCbkHandleId, PduInfoType* PduInfoPtr )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the ID of the SDU that is requested to be transmitted
	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 (inout)	None	
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	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

[SWS_Com_91030] Definition of configurable interface <ComUser_ComCbkTxConfirmation>

Previous identifiers: SWS_LDCOM_91008

⌈

Service Name	<ComUser_ComCbkTxConfirmation>	
Syntax	`void <ComUser_ComCbkTxConfirmation> ( CbkHandleIdType ComUserCbkHandleId, Std_ReturnType result )`	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to the PDU that has been transmitted
	result	E_OK: The PDU was transmitted. 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	ComUserHeaderInclude ([ECUC_Com_10032])

⌋

[SWS_Com_91031] Definition of configurable interface <ComUser_ComCbkRx Indication>

Previous identifiers: SWS_LDCOM_91006

⌈

Service Name	<ComUser_ComCbkRxIndication>	

Syntax	`void <ComUser_ComCbkRxIndication> (`
	`  CbkHandleIdType ComUserCbkHandleId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Sync/Async	Asynchronous	
Reentrancy	Non Reentrant for same ComUserCbkHandleId, otherwise Reentrant	
Parameters (in)	ComUserCbkHandleId	Com user callback handle Id corresponding to received I-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	ComUserHeaderInclude ([ECUC_Com_10032])	

⌋

As stated in [SWS_Com_00013] Network-order message callout and CPU-order message callout are not supported by the AUTOSAR COM module. The only callout method supported by the AUTOSAR COM module therefore is the I-PDU callout. The AUTOSAR COM module supports I-PDU callouts on sender and on receiver side.

As specified in [2], if the I-PDU callout returns false the I-PDU will not be processed any further.

[SWS_Com_00700] Definition of configurable interface Com_RxIpduCallout ⌈

Service Name	Com_RxIpduCallout	

Syntax	`boolean Com_RxIpduCallout (`
	`  PduIdType PduId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID [hex]	0x2A	
Sync/Async	Synchronous	
Reentrancy	don't care	
Parameters (in)	PduId	ID of the received I-PDU.

▽

△

	PduInfoPtr	Contains the length (SduLength) of the received I-PDU and a pointer to the data of the I-PDU (SduDataPtr).
Parameters (inout)	None	
Parameters (out)	None	
Return value	boolean	true: I-PDU will be processed normal false: I-PDU will not be processed any further
Description	The I-PDU callout on receiver side can be configured to implement user-defined receive filtering mechanisms.	
Available via	Configuration parameter ComUserCbkHeaderFile	

⌋

The callout function name must be configured by `ComIPduCallout`.

[SWS_Com_00816] ⌈The AUTOSAR COM module shall forward all data of the received I-PDU (i.e. the complete I-PDU as provided by the PduR) in the `Com_RxIpduCallout`.⌋

[SWS_Com_00346] Definition of configurable interface Com_TxIpduCallout ⌈

Service Name	Com_TxIpduCallout	
Syntax	`boolean Com_TxIpduCallout ( PduIdType PduId, PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x29	
Sync/Async	Synchronous	
Reentrancy	don't care	
Parameters (in)	PduId	ID of the transmitted I-PDU.
Parameters (inout)	PduInfoPtr	Contains the length (SduLength) of the transmitted I-PDU and a pointer to the data of the I-PDU (SduDataPtr).
Parameters (out)	None	
Return value	boolean	true: I-PDU will be processed normal false: I-PDU will not be processed any further
Description	The I-PDU callout on sender side can be configured for example to implement user-defined transmission filtering or user-defined pre-transmission-processing of the outgoing I-PDU.	
Available via	Configuration parameter ComUserCbkHeaderFile	

⌋

The callout function name must be configured either by `ComIPduCallout` or by `ComIPduTriggerTransmitCallout`.

The AUTOSAR COM module implements no means to detect or prevent possible recursions that may occur when I-PDUs are triggered by actions within an I-PDU callout. It is solely in the responsibility of the implementer of an I-PDU callout and the system integrator that unwanted or endless chains of I-PDU callouts are avoided.

[SWS_Com_00381] ⌈The AUTOSAR COM module shall not support that other AUTOSAR COM module's APIs than `Com_TriggerIPDUSend`, `Com_TriggerIPDUSendWithMetaData`, `Com_SendSignal` and `Com_SendSignalGroup` can be called out of an I-PDU callout.⌋

[SWS_Com_00766] ⌈In case ComIPduTriggerTransmitCallout is configured for an I-PDU and the I-PDU is started, the AUTOSAR COM module shall invoke this I-PDU callout within every execution of Com_TriggerTransmit for this I-PDU.⌋

[SWS_Com_00395] ⌈When Com_TriggerTransmit is called, the AUTOSAR COM module shall ignore the return value from the ComIPduTriggerTransmitCallout.⌋

[SWS_Com_00719] ⌈In case a ComIPduCallout is configured for an I-PDU with ComIPduDirection configured to SEND, the AUTOSAR COM module shall invoke this I-PDU callout directly before the I-PDU is transmitted via PduR_ComTransmit.⌋

For TP I-PDUs, the I-PDU callout functions are handled similar to the I-PDU callout functions of NORMAL I-PDUs and are supported both on transmitter and receiver side. Note that for example the requirement [SWS_Com_00719] does not consider the configured ComIPduType.

[SWS_Com_00720] ⌈When a large I-PDU is received successfully via the TP interface, the I-PDU callout shall take place within the invocation of Com_TpRxIndication.⌋

[SWS_Com_00839] ⌈When the I-PDU callout in context of Com_TpRxIndication returns false, the I-PDU shall be considered as incorrectly received and therefore all included signals shall be treated exactly like invalidated signals. That is, the AUTOSAR COM module shall execute all configured ComDataInvalidAction for all included signals and signal groups.⌋

9 Sequence diagrams

A sequence diagram of the underlying communication stack can be found in [2].

9.1 Interface between the AUTOSAR COM Module and the PDU Router

The following chart shows the communication between the AUTOSAR COM module and the PDU Router.

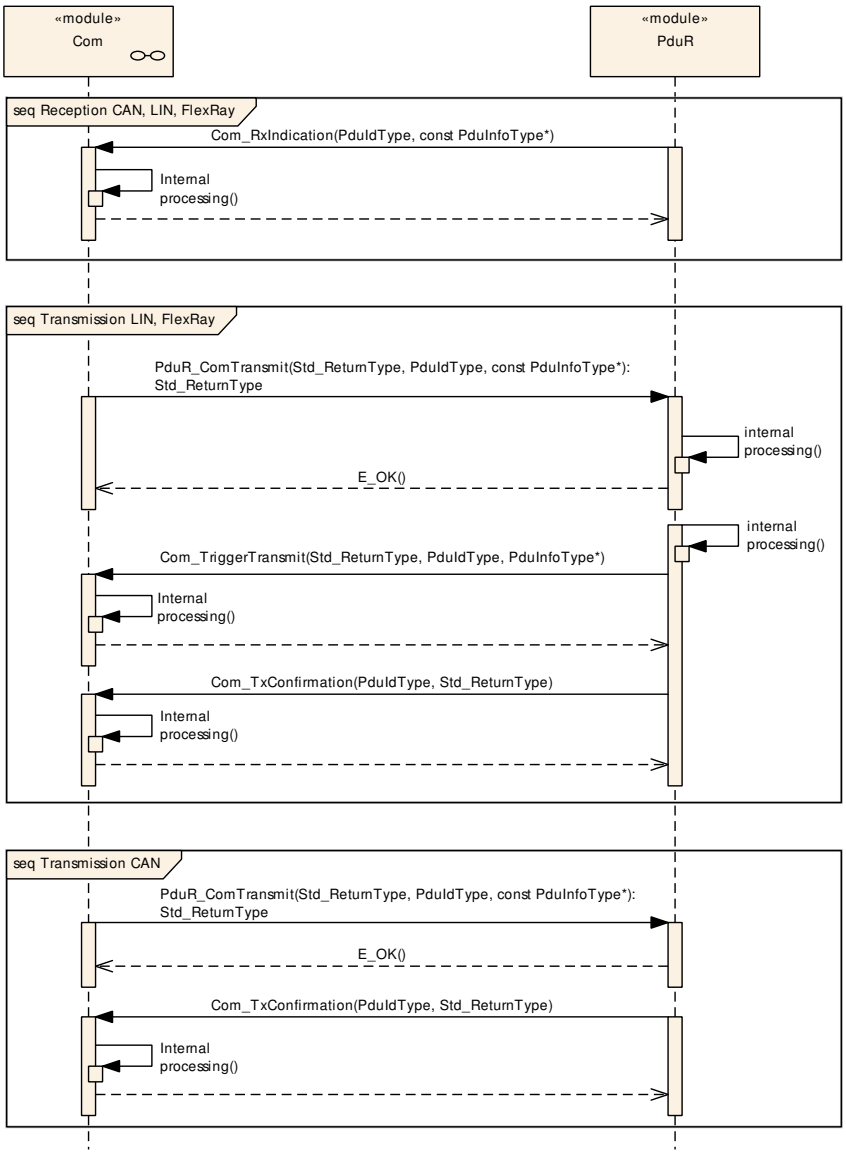


Figure 9.1: Interactions between the AUTOSAR COM module and the PDU Router

9.2 Confirmation Handling between the PDU Router, the AUTOSAR COM Module and the RTE

The following chart shows the confirmation handling with respect to the two different IPDU-processing modes. (See also Chapter 7.2.5.4.)

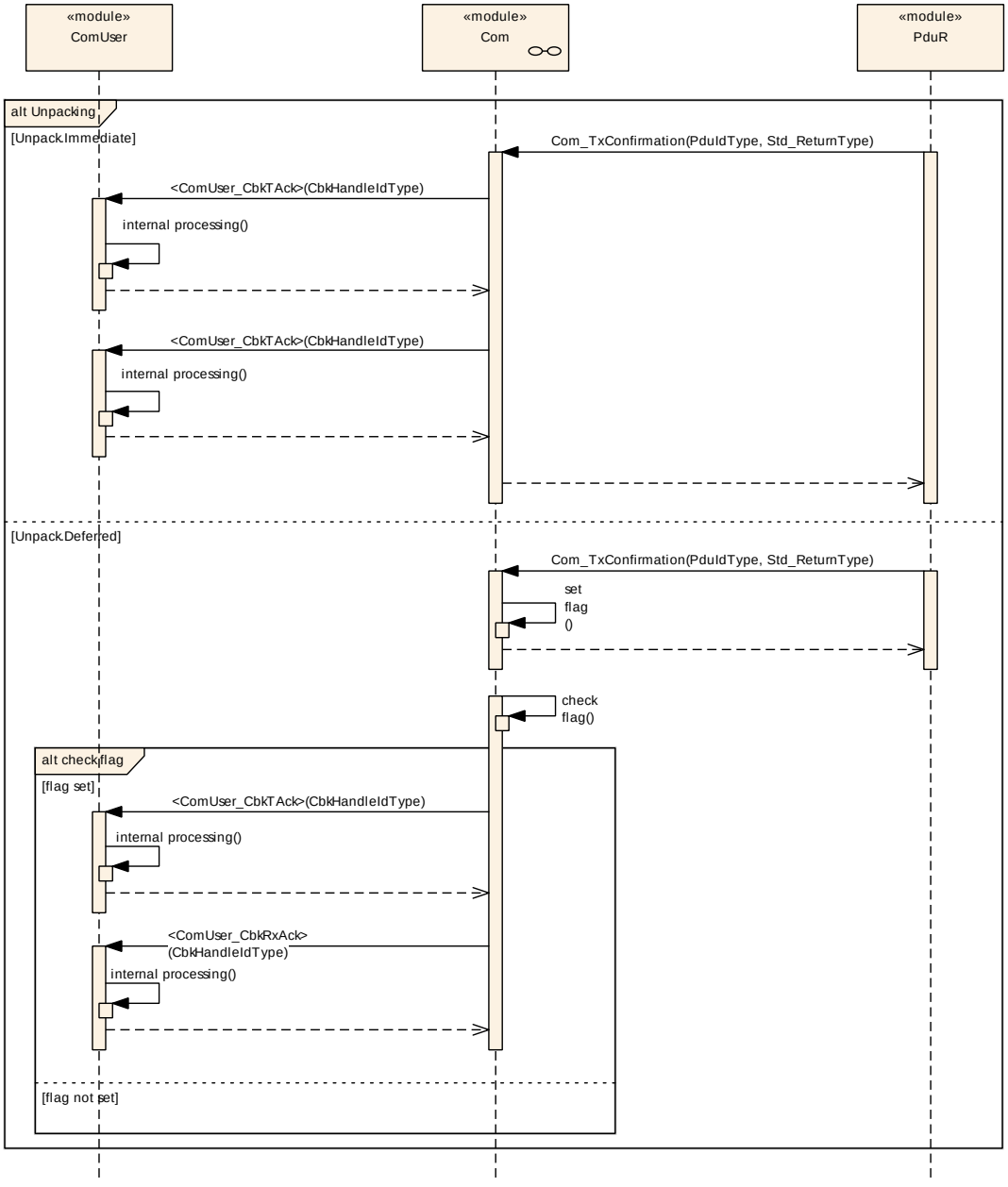


Figure 9.2: Confirmation handling between the PDU Router, the AUTOSAR COM module and the RTE

9.3 Indication Handling between the PDU Router, the AUTOSAR COM Module and the RTE

The following chart shows the indication handling with respect to the two different unpacking modes. (See also Chapter 7.2.5.4.)

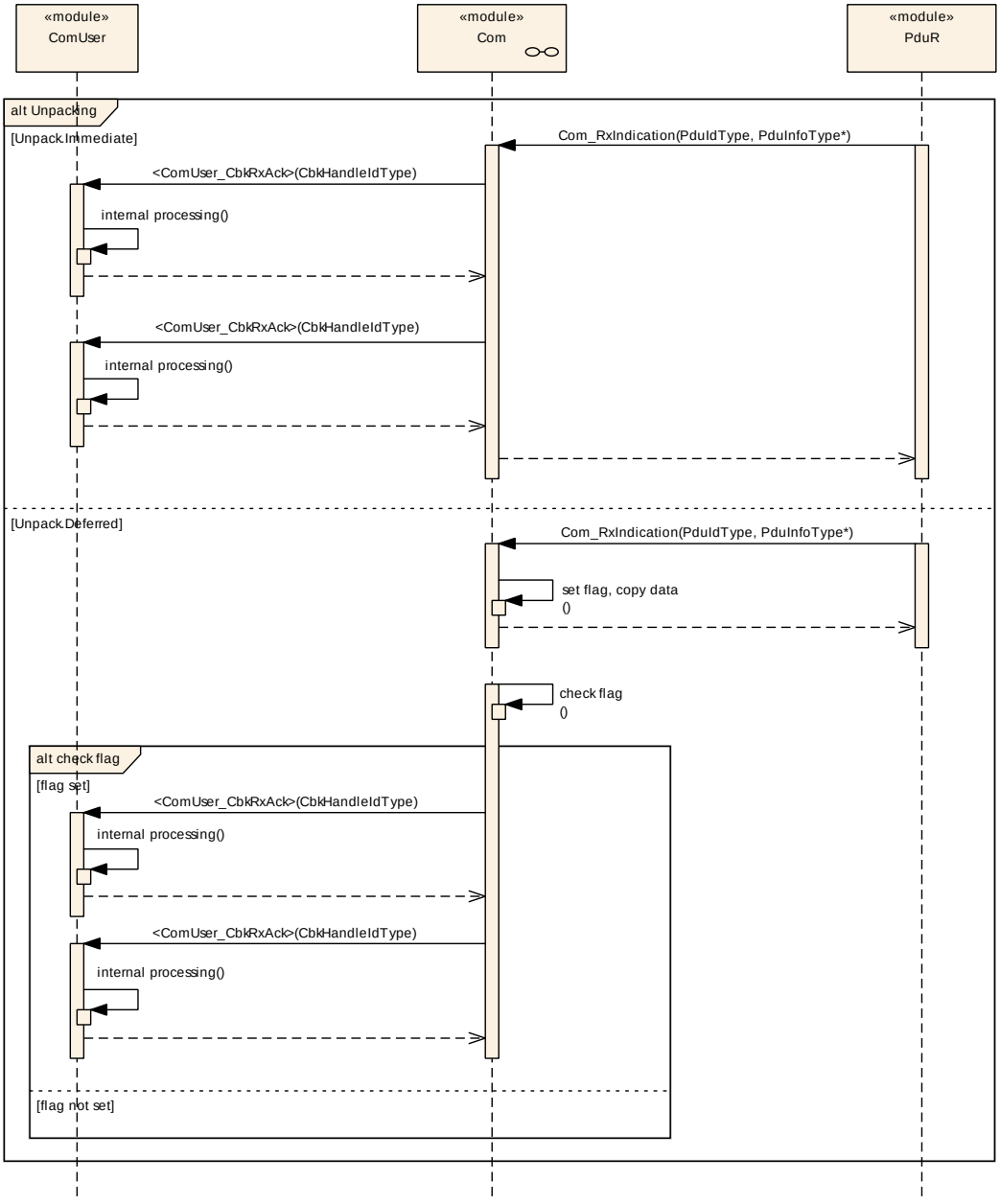


Figure 9.3: Indication handling between the PDU Router, the AUTOSAR COM module and the RTE

9.4 Large Data Path Transmission

9.4.1 TP API

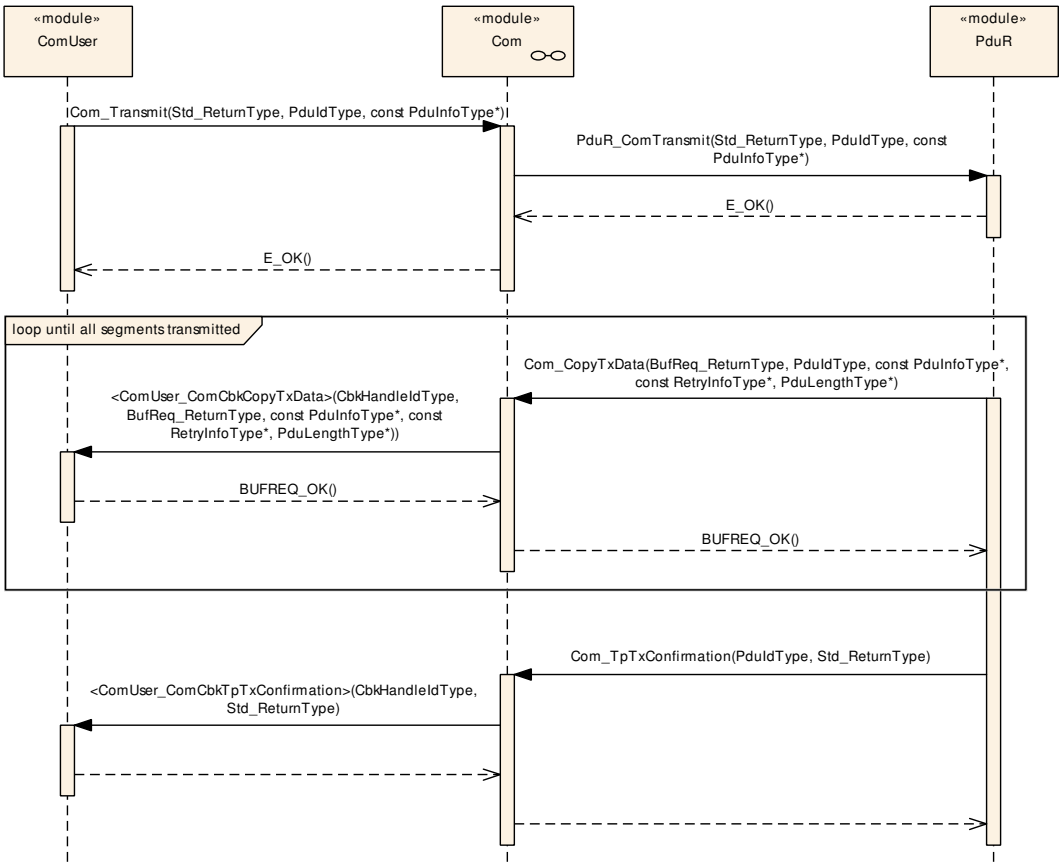


Figure 9.4: Transmission via Large Data Path using TP API

9.4.2 IF API

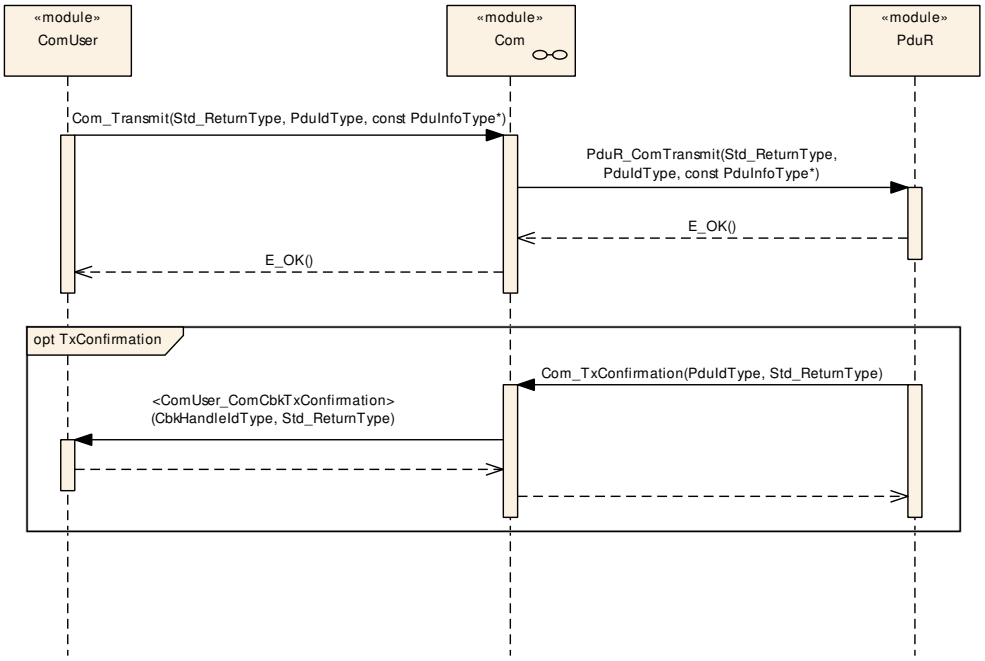


Figure 9.5: Transmission via Large Data Path using IF API

9.4.3 TriggerTransmit

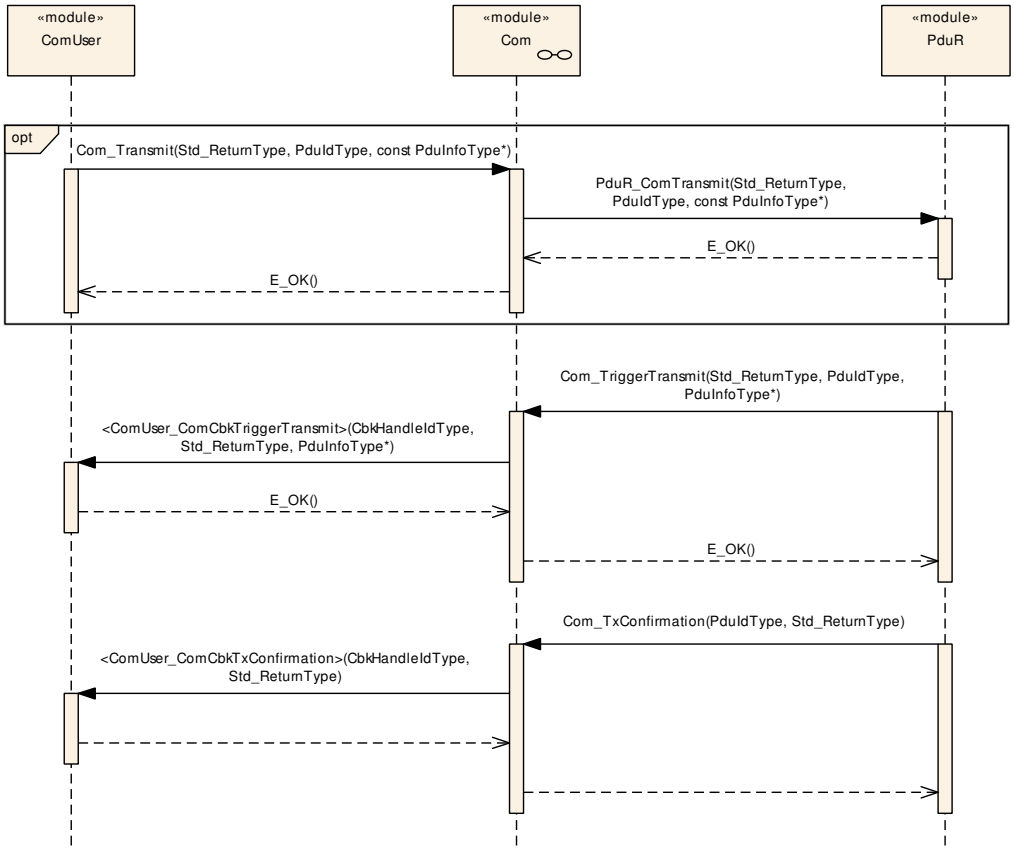


Figure 9.6: Transmission via Large Data Path using TriggerTransmit

9.5 Large Data Path Reception

9.5.1 TP API

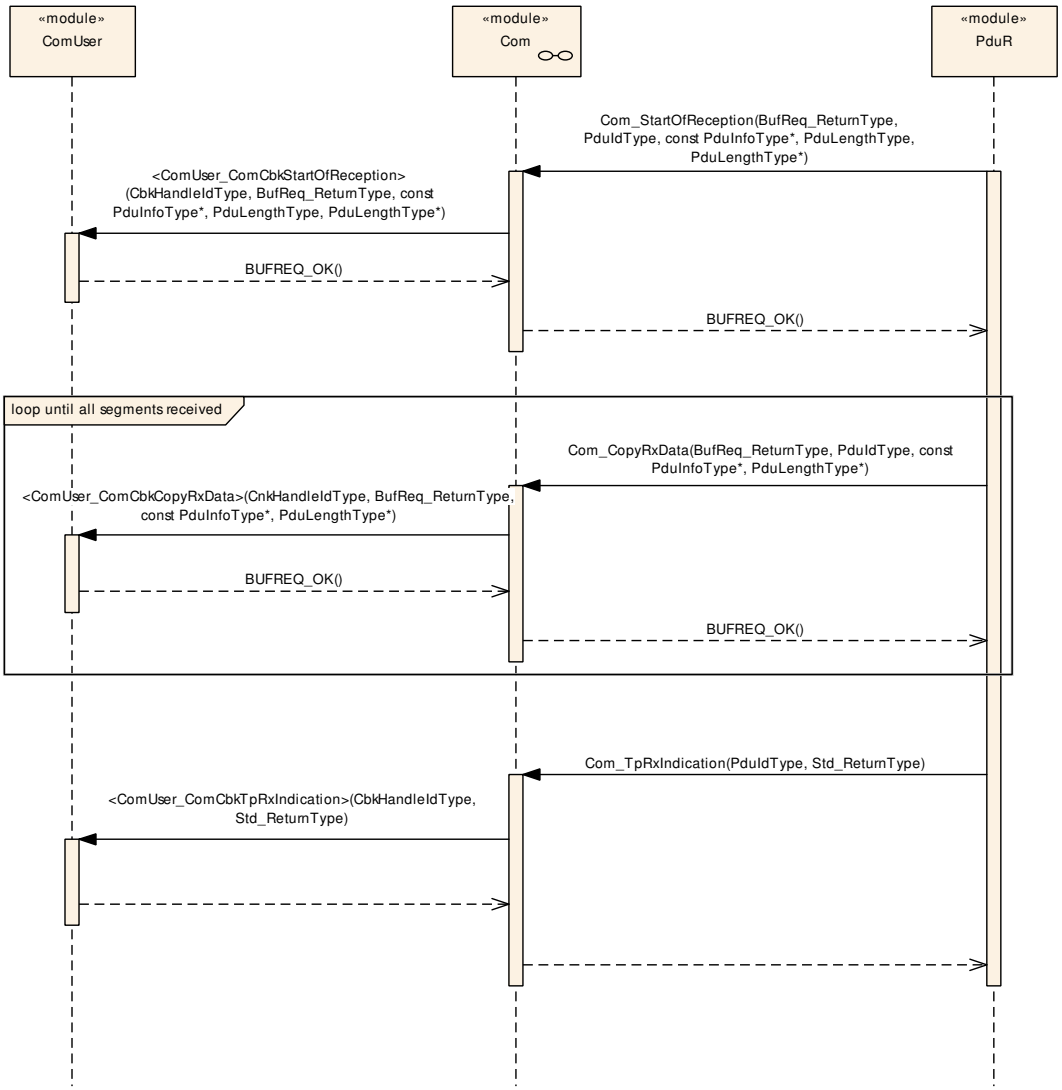


Figure 9.7: Reception via Large Data Path using TP API

9.5.2 IF API

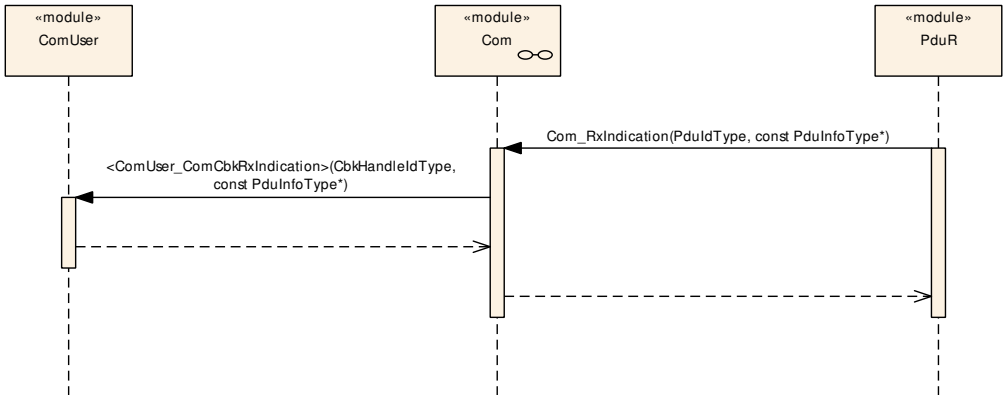


Figure 9.8: Reception via Large Data Path using IF API

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 COM.

Chapter 10.4 specifies published information of the module COM.

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 are described in Chapter 7 and Chapter 8.

10.2.1 Configuration of the AUTOSAR COM Module

For an overview of the AUTOSAR COM module's configuration, see Figure 10.1.

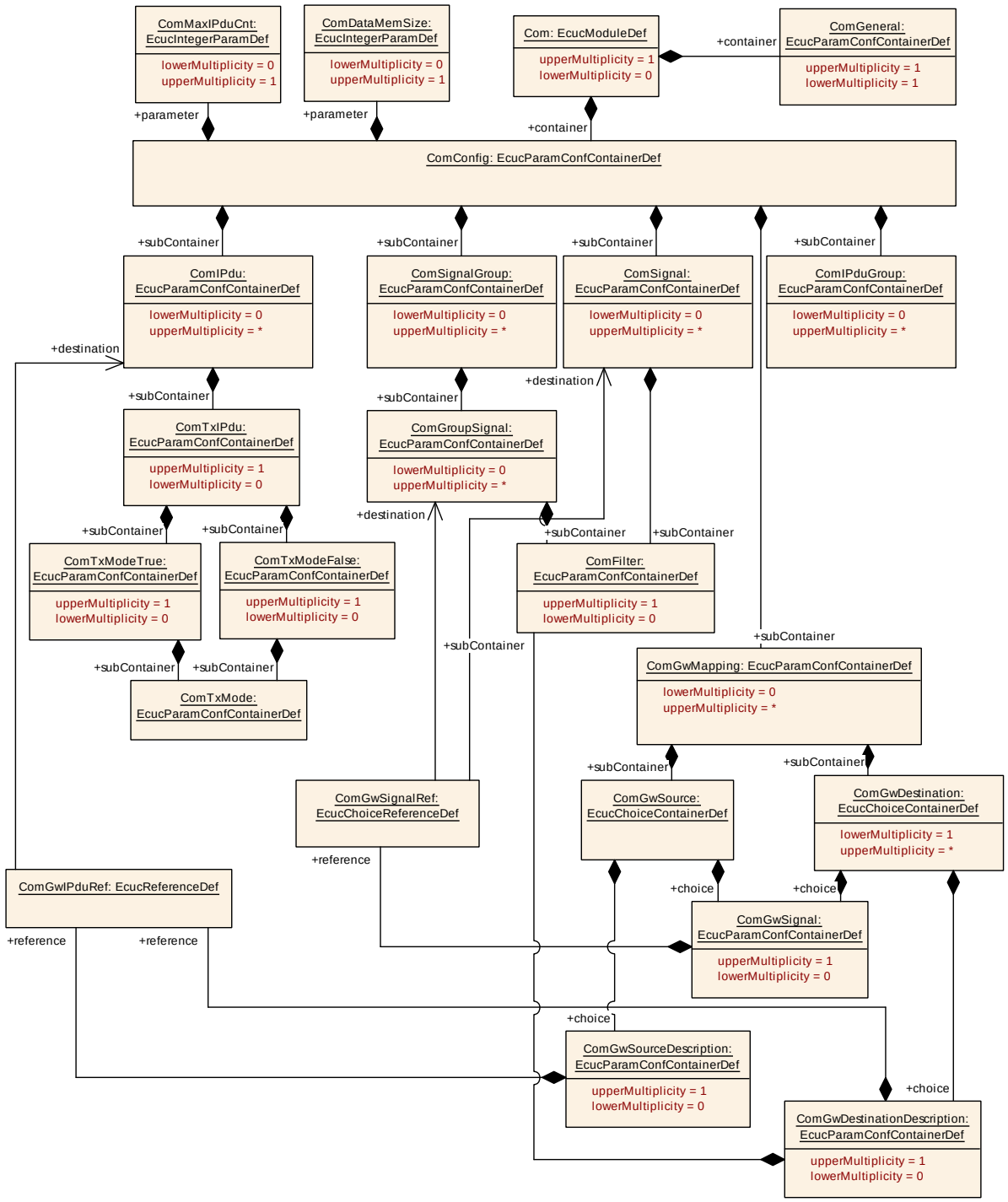


Figure 10.1: The AUTOSAR COM module's Configuration Overview

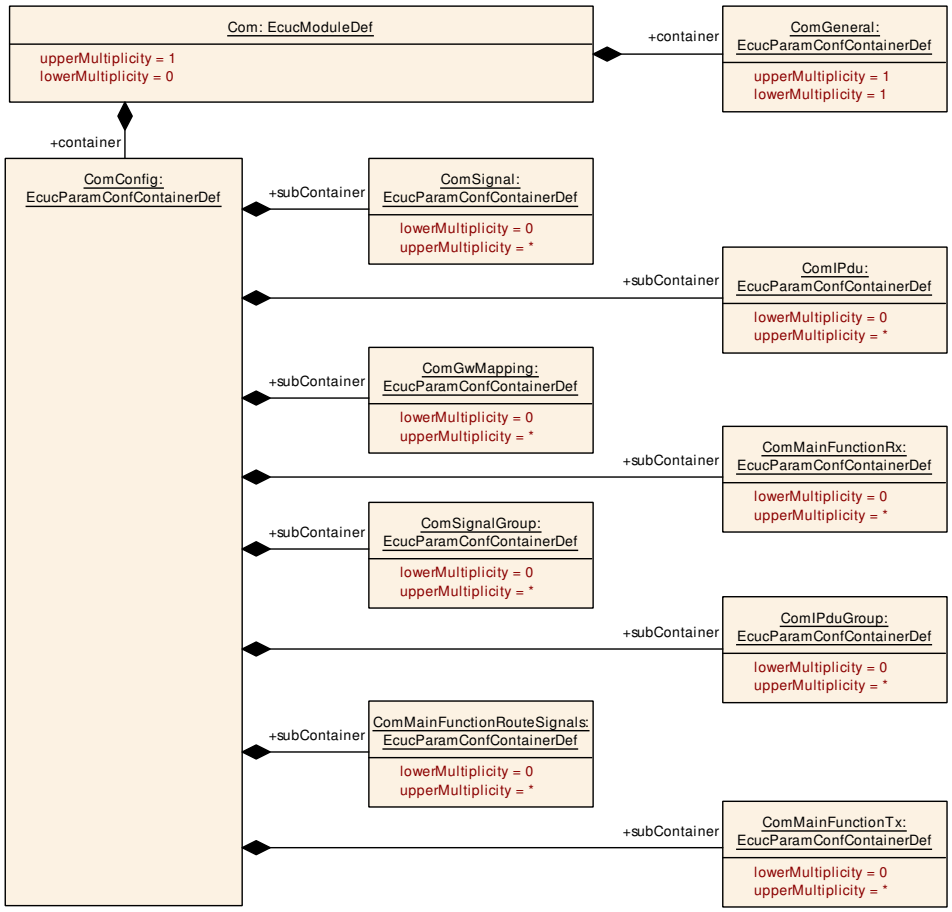


Figure 10.2: Com

10.2.2 Com

[ECUC_Com_00540] Definition of EcucModuleDef Com ⌈

Module Name	Com
Description	Configuration of the AUTOSAR COM module.
Post-Build Selectable Support	true
Post-Build Loadable Support	true

Included Containers		
Container Name	**Multiplicity**	**Description**
ComConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR COM module.
ComGeneral	1	Contains the general configuration parameters of the AUTOSAR COM module.

⌋

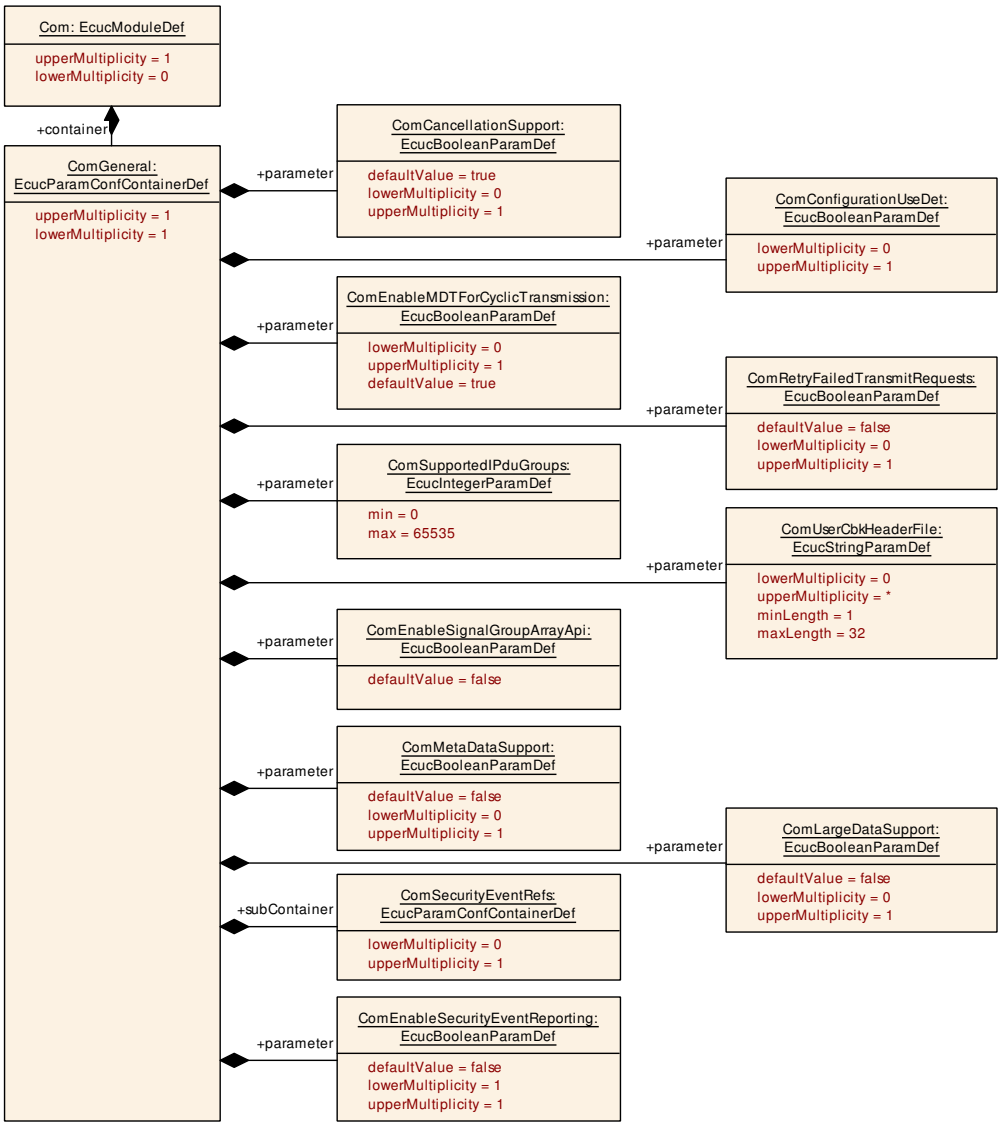


Figure 10.3: ComGeneral

10.2.3 ComGeneral

[ECUC_Com_00541] Definition of EcucParamConfContainerDef ComGeneral ⌈

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

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComCancellationSupport	0..1	[ECUC_Com_10000]
ComConfigurationUseDet	0..1	[ECUC_Com_00141]
ComEnableMDTForCyclicTransmission	0..1	[ECUC_Com_00788]
ComEnableSecurityEventReporting	1	[ECUC_Com_10042]
ComEnableSignalGroupArrayApi	1	[ECUC_Com_10002]
ComLargeDataSupport	0..1	[ECUC_Com_10049]
ComMetaDataSupport	0..1	[ECUC_Com_10004]
ComRetryFailedTransmitRequests	0..1	[ECUC_Com_00780]
ComSupportedIPduGroups	1	[ECUC_Com_00710]
ComUserCbkHeaderFile	0..*	[ECUC_Com_10001]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComSecurityEventRefs	0..1	Container for the references to IdsMEvent elements representing the security events that the COM module shall report to the IdsM in case the corresponding security related event occurs (and if ComEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events.

⌋

[ECUC_Com_10000] Definition of EcucBooleanParamDef ComCancellationSupport ⌈

Parameter Name	ComCancellationSupport
Parent Container	ComGeneral
Upstream mapping defined	–
Description	This parameter enables/disables the cancellation feature: true: enabled false: disabled
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	true
Pre-Compile only	true

⌋

[ECUC_Com_00141] Definition of EcucBooleanParamDef ComConfigurationUseDet ⌈

Parameter Name	ComConfigurationUseDet
Parent Container	ComGeneral
Upstream mapping defined	–
Description	The error hook shall contain code to call the Det. If this parameter is configured COM_DEV_ERROR_DETECT shall be set to ON as output of the configuration tool. (as input for the source code).
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	–

▽

△

Pre-Compile only	true

⌋

[ECUC_Com_00788] Definition of EcucBooleanParamDef ComEnableMDTForCyclicTransmission ⌈

Parameter Name	ComEnableMDTForCyclicTransmission
Parent Container	ComGeneral
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Enables globally for the whole Com module the minimum delay time monitoring for cyclic and repeated transmissions (ComTxModeMode=PERIODIC or ComTxMode Mode=MIXED for the cyclic transmissions, ComTxModeNumberOfRepetitions > 0 for repeated transmissions).
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	true
Pre-Compile only	true

⌋

[ECUC_Com_10042] Definition of EcucBooleanParamDef ComEnableSecurityEventReporting ⌈

Parameter Name	ComEnableSecurityEventReporting
Parent Container	ComGeneral
Upstream mapping defined	–
Description	Switches the reporting of security events to the IdsM. • true: reporting is enabled. • false: reporting is disabled.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_10002] Definition of EcucBooleanParamDef ComEnableSignalGroupArrayApi ⌈

Parameter Name	ComEnableSignalGroupArrayApi
Parent Container	ComGeneral
Upstream mapping defined	–
Description	Activate/Deactivate the signal group array access APIs (Com_SendSignalGroupArray, Com_ReceiveSignalGroupArray). true: signal group array access APIs activated false: signal group array access APIs deactivated
Multiplicity	1
Type	EcucBooleanParamDef

▽

△

Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_10049] Definition of EcucBooleanParamDef ComLargeDataSupport ⌈

Parameter Name	ComLargeDataSupport
Parent Container	ComGeneral
Upstream mapping defined	–
Description	This parameter enables/disables Large Data Path support for COM module.
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_10004] Definition of EcucBooleanParamDef ComMetaDataSupport ⌈

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

⌋

[ECUC_Com_00780] Definition of EcucBooleanParamDef ComRetryFailedTransmitRequests ⌈

Parameter Name	ComRetryFailedTransmitRequests
Parent Container	ComGeneral
Upstream mapping defined	–
Description	If this Parameter is set to true, retry of failed transmission requests is enabled. If this Parameter is not present, the default value is assumed.
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_00710] Definition of EcucIntegerParamDef ComSupportedIPdu Groups ⌈

Parameter Name	ComSupportedIPduGroups
Parent Container	ComGeneral
Upstream mapping defined	–
Description	Defines the maximum number of supported I-PDU groups.
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 65535
Default value	–
Pre-Compile only	true

⌋

[ECUC_Com_10001] Definition of EcucStringParamDef ComUserCbkHeaderFile ⌈

Parameter Name	ComUserCbkHeaderFile
Parent Container	ComGeneral
Upstream mapping defined	–
Description	Defines the header files for callback functions which shall be included by the COM module.
Multiplicity	0..*
Type	EcucStringParamDef
Default value	–
Length	1-32
Regular Expression	–
Pre-Compile only	true

⌋

10.2.4 ComSecurityEventRefs

[ECUC_Com_10041] Definition of EcucParamConfContainerDef ComSecurity EventRefs ⌈

Container Name	ComSecurityEventRefs
Parent Container	ComGeneral
Upstream mapping defined	–
Description	Container for the references to IdsMEvent elements representing the security events that the COM module shall report to the IdsM in case the corresponding security related event occurs (and if ComEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED	0..1	[ECUC_Com_10043]

No Included Containers

⌋

[ECUC_Com_10043] Definition of EcucReferenceDef SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED ⌈

Parameter Name	SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED
Parent Container	ComSecurityEventRefs
Upstream mapping defined	–
Description	Signal or group signal is received with out of range value.
Multiplicity	0..1
Type	Symbolic name reference to IdsMEvent
Pre-Compile only	true

⌋

10.2.5 ComConfig

[ECUC_Com_00337] Definition of EcucParamConfContainerDef ComConfig ⌈

Container Name	ComConfig
Parent Container	Com
Upstream mapping defined	–
Description	This container contains the configuration parameters and sub containers of the AUTOSAR COM module.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComDataMemSize	0..1	[ECUC_Com_00783]
ComMaxIPduCnt	0..1	[ECUC_Com_00782]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComGwMapping	0..*	Each instance of this container defines one mapping of the integrated Signal Gateway.
ComIPdu	0..*	See ECUC_Com_00340, if there is no ComIpdu container included no IPDUs are defined. In this case no communication via the AUTOSAR COM module is possible.

▽

△

Included Containers		
Container Name	**Multiplicity**	**Description**
ComIPduGroup	0..*	See ECUC_Com_00341, if there is no ComIPduGroup container included then no IPDU group is defined. In this case no communication via the AUTOSAR COM module is possible.
ComMainFunctionRouteSignals	0..*	Each element of this container defines one instance of Com_MainFunctionRouteSignals.
ComMainFunctionRx	0..*	Each element of this container defines one instance of Com_MainFunctionRx.
ComMainFunctionTx	0..*	Each element of this container defines one instance of Com_MainFunctionTx.
ComSignal	0..*	See ECUC_Com_00344, if there is no ComSignal container included no single signals are defined.
ComSignalGroup	0..*	See ECUC_Com_00345, if there is no ComSignalGroup container included no signal groups are defined.
ComUserModule	0..*	Contains the configuration parameters of the Com user modules.

⌋

[ECUC_Com_00783] Definition of EcucIntegerParamDef ComDataMemSize ⌈

Parameter Name	ComDataMemSize	
Parent Container	ComConfig	
Upstream mapping defined	–	
Description	Size of internal Com data in units of bytes (static memory allocation) - memory required by post-build configuration must be smaller than this constant. This parameter is needed only in case of post-build loadable implementation using static memory allocation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 18446744073709551615	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Com_00782] Definition of EcucIntegerParamDef ComMaxIPduCnt ⌈

Parameter Name	ComMaxIPduCnt	
Parent Container	ComConfig	
Upstream mapping defined	–	
Description	Maximum number of IPdus. This parameter is needed only in case of post-build loadable implementation using static memory allocation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 18446744073709551615	
Default value	–	
Pre-Compile only	true	

⌋

10.2.6 ComMainFunctionRx

[ECUC_Com_10011] Definition of EcucParamConfContainerDef ComMainFunctionRx ⌈

Container Name	ComMainFunctionRx
Parent Container	ComConfig
Upstream mapping defined	–
Description	Each element of this container defines one instance of Com_MainFunctionRx.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComMainRxTimeBase	1	[ECUC_Com_10018]
ComMainRxPartitionRef	0..1	[ECUC_Com_10017]

No Included Containers

⌋

[ECUC_Com_10018] Definition of EcucFloatParamDef ComMainRxTimeBase ⌈

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

⌋

[ECUC_Com_10017] Definition of EcucReferenceDef ComMainRxPartitionRef ⌈

Parameter Name	ComMainRxPartitionRef
Parent Container	ComMainFunctionRx
Upstream mapping defined	–
Description	Reference to EcucPartition, where the according Com_MainFunction instance is assigned to.
Multiplicity	0..1

▽

△

Type	Reference to EcucPartition
Pre-Compile only	true

⌋

10.2.7 ComMainFunctionTx

[ECUC_Com_10014] Definition of EcucParamConfContainerDef ComMainFunctionTx ⌈

Container Name	ComMainFunctionTx
Parent Container	ComConfig
Upstream mapping defined	–
Description	Each element of this container defines one instance of Com_MainFunctionTx.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
ComMainTxTimeBase	1	[ECUC_Com_10010]
ComPreparationNotification	0..1	[ECUC_Com_10020]
ComMainTxPartitionRef	0..1	[ECUC_Com_10019]

No Included Containers

⌋

[ECUC_Com_10010] Definition of EcucFloatParamDef ComMainTxTimeBase ⌈

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

⌋

[ECUC_Com_10020] Definition of EcucFunctionNameDef ComPreparationNotification ⌈

Parameter Name	ComPreparationNotification
Parent Container	ComMainFunctionTx
Upstream mapping defined	–
Description	This callback function indicates that the signals/signal groups to be sent via a dedicated Com_MainFunctionTx instance will now be prepared for transmission. If this parameter is omitted no notification shall take place.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Com_10019] Definition of EcucReferenceDef ComMainTxPartitionRef ⌈

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

⌋

10.2.8 ComMainFunctionRouteSignals

[ECUC_Com_10013] Definition of EcucParamConfContainerDef ComMainFunctionRouteSignals ⌈

Container Name	ComMainFunctionRouteSignals
Parent Container	ComConfig
Upstream mapping defined	–
Description	Each element of this container defines one instance of Com_MainFunctionRoute Signals.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComMainRouteSignalsTimeBase	1	[ECUC_Com_10016]
ComMainRouteSignalsPartitionRef	0..1	[ECUC_Com_10015]

No Included Containers

⌋

[ECUC_Com_10016] Definition of EcucFloatParamDef ComMainRouteSignals TimeBase ⌈

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

⌋

[ECUC_Com_10015] Definition of EcucReferenceDef ComMainRouteSignalsPartitionRef ⌈

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

⌋

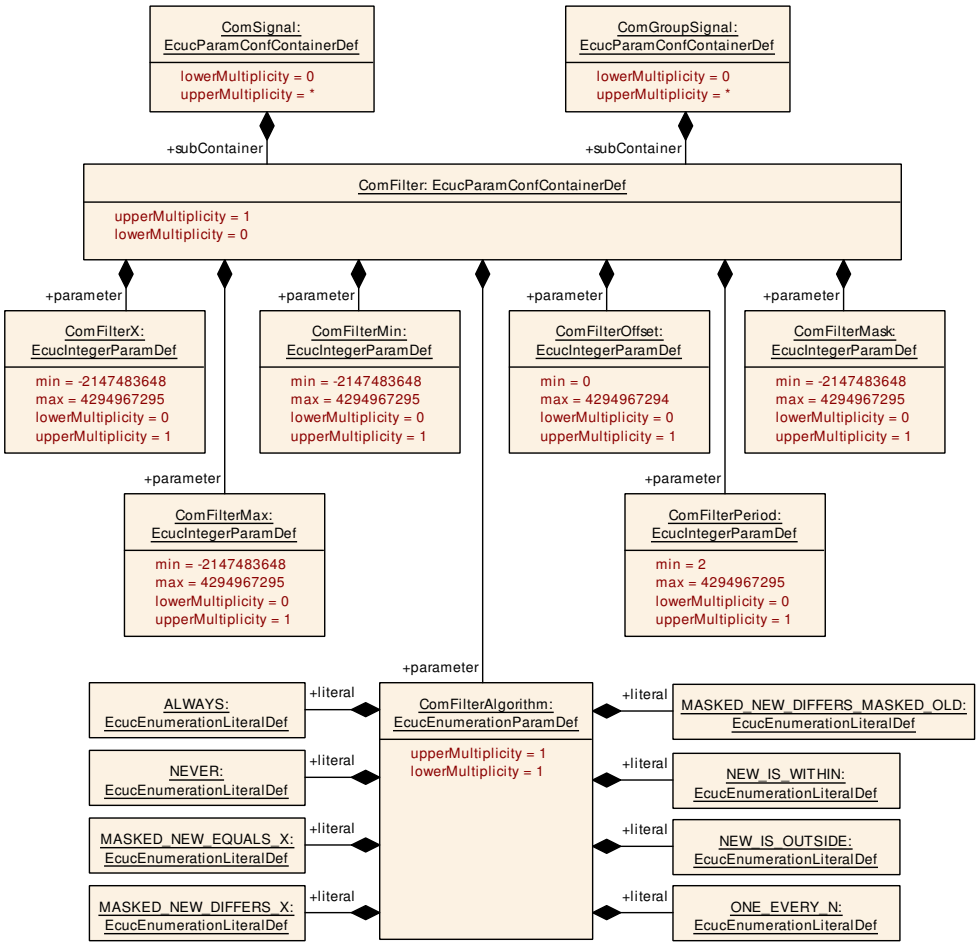


Figure 10.4: ComFilter

10.2.9 ComFilter

[ECUC_Com_00339] Definition of EcucParamConfContainerDef ComFilter ⌈

Container Name	ComFilter
Parent Container	ComGroupSignal, ComGwDestinationDescription, ComSignal
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	This container contains the configuration parameters of the AUTOSAR COM module's Filters. Note: On sender side the container is used to specify the transmission mode conditions.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComFilterAlgorithm	1	[ECUC_Com_00146]
ComFilterMask	0..1	[ECUC_Com_00235]
ComFilterMax	0..1	[ECUC_Com_00317]
ComFilterMin	0..1	[ECUC_Com_00318]
ComFilterOffset	0..1	[ECUC_Com_00313]
ComFilterPeriod	0..1	[ECUC_Com_00312]
ComFilterX	0..1	[ECUC_Com_00147]

No Included Containers

⌋

[ECUC_Com_00146] Definition of EcucEnumerationParamDef ComFilterAlgorithm ⌈

Parameter Name	ComFilterAlgorithm	
Parent Container	ComFilter	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	The range of values is specified in the Clause 3.2.3 (Reception Filtering) of [3].	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	ALWAYS	Allow all (always true)
	MASKED_NEW_DIFFERS_ MASKED_OLD	Allow if the masked value has changed
	MASKED_NEW_DIFFERS_X	Allow if the masked new value is different from a specific value
	MASKED_NEW_EQUALS_X	Allow if the masked new value equals a specific value
	NEVER	Drop all (always false)
	NEW_IS_OUTSIDE	Allow if new value is outside a range
	NEW_IS_WITHIN	Allow if new value is within a range
	ONE_EVERY_N	Allow every n-th message
Pre-Compile only	true	

⌋

[ECUC_Com_00235] Definition of EcucIntegerParamDef ComFilterMask ⌈

Parameter Name	ComFilterMask	
Parent Container	ComFilter	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	The name of this attribute corresponds to the parameter name in the Clause 3.2.3 (Reception filtering) of [3].	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	-2147483648 .. 4294967295	
Default value	–	

▽

△

Pre-Compile only	true

⌋

[ECUC_Com_00317] Definition of EcucIntegerParamDef ComFilterMax ⌈

Parameter Name	ComFilterMax
Parent Container	ComFilter
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	The name of this attribute corresponds to the parameter name in the Clause 3.2.3 (Reception filtering) of [3].
Multiplicity	0..1
Type	EcucIntegerParamDef
Range	-2147483648 .. 4294967295
Default value	–
Pre-Compile only	true

⌋

[ECUC_Com_00318] Definition of EcucIntegerParamDef ComFilterMin ⌈

Parameter Name	ComFilterMin
Parent Container	ComFilter
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	The name of this attribute corresponds to the parameter name in the Clause 3.2.3 (Reception filtering) of [3].
Multiplicity	0..1
Type	EcucIntegerParamDef
Range	-2147483648 .. 4294967295
Default value	–
Pre-Compile only	true

⌋

[ECUC_Com_00313] Definition of EcucIntegerParamDef ComFilterOffset ⌈

Parameter Name	ComFilterOffset
Parent Container	ComFilter
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	The name of this attribute corresponds to the parameter name in the Clause 3.2.3 (Reception filtering) of [3]. Range = 0..(ComFilterPeriod-1) Dependency: ECUC_Com_00312
Multiplicity	0..1
Type	EcucIntegerParamDef
Range	0 .. 4294967294
Default value	–
Pre-Compile only	true

⌋

[ECUC_Com_00312] Definition of EcucIntegerParamDef ComFilterPeriod ⌈

Parameter Name	ComFilterPeriod	
Parent Container	ComFilter	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	This parameter defines the period of the ComFilterAlgorithm ONE_EVERY_N.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	2 .. 4294967295	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00147] Definition of EcucIntegerParamDef ComFilterX ⌈

Parameter Name	ComFilterX	
Parent Container	ComFilter	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	The name of this attribute corresponds to the parameter name in the Clause 3.2.3 (Reception filtering) of [3].	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	-2147483648 .. 4294967295	
Default value	–	
Pre-Compile only	true	

⌋

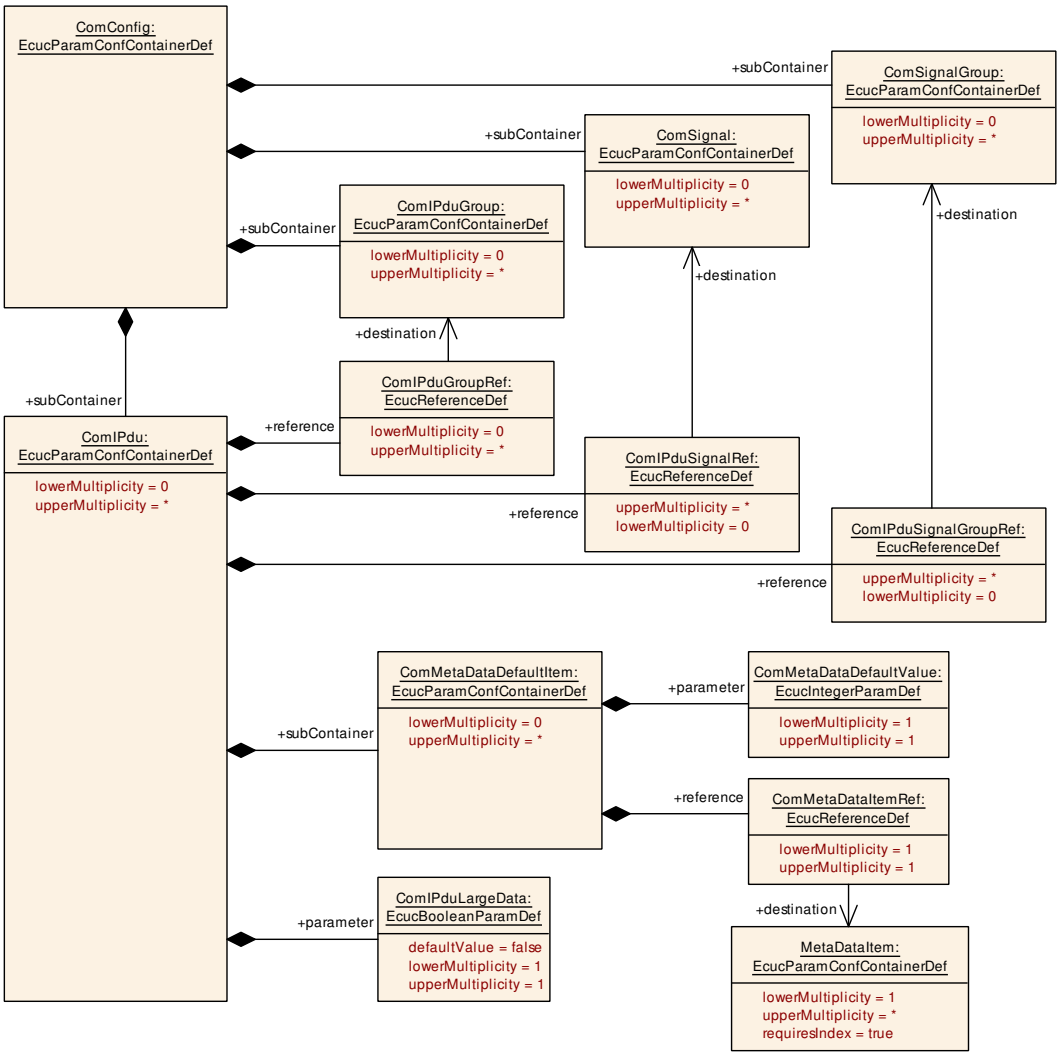


Figure 10.5: ComIPdu

10.2.10 ComIPdu

[ECUC_Com_00340] Definition of EcucParamConfContainerDef ComIPdu ⌈

Container Name	ComIPdu	
Parent Container	ComConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters of the AUTOSAR COM module's I-PDUs.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComIPduCallout	0..1	[ECUC_Com_00387]
ComIPduCancellationSupport	0..1	[ECUC_Com_00709]
ComIPduDirection	1	[ECUC_Com_00493]
ComIPduHandleId	0..1	[ECUC_Com_00175]
ComIPduLargeData	1	[ECUC_Com_10045]
ComIPduSignalProcessing	1	[ECUC_Com_00119]
ComIPduTriggerTransmitCallout	0..1	[ECUC_Com_00765]
ComIPduType	1	[ECUC_Com_00761]
ComIPduGroupRef	0..*	[ECUC_Com_00206]
ComIPduMainFunctionRef	0..1	[ECUC_Com_10012]
ComIPduSignalGroupRef	0..*	[ECUC_Com_00519]
ComIPduSignalRef	0..*	[ECUC_Com_00518]
ComMainFunctionRouteSignalsRef	0..1	[ECUC_Com_10021]
ComPduIdRef	1	[ECUC_Com_00711]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComMetaDataDefaultItem	0..*	Defines a default value for a meta data item. Used for sending an I-PDU with meta data when it is triggered spontaneously (and not by Com_TriggerIPDUSendWithMetaData), and no meta data has been provided by the RTE. Also used for reception, when e.g. Com_ReceiveSignalWithMetaData is called before an I-PDU is received. It represents a MetaDataItem of the referenced global PDU.
ComTxIPdu	0..1	This container must be included if COM_IPDU_DIRECTION is configured to SEND.

⌋

[ECUC_Com_00387] Definition of EcucFunctionNameDef ComIPduCallout ⌈

Parameter Name	ComIPduCallout
Parent Container	ComIPdu
Upstream mapping defined	–
Description	This parameter defines the existence and the name of a callout function for the corresponding I-PDU. If this parameter is omitted no I-PDU callout shall take place for the corresponding I-PDU.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Com_00709] Definition of EcucBooleanParamDef ComIPduCancellation Support ⌈

Parameter Name	ComIPduCancellationSupport	
Parent Container	ComIPdu	
Upstream mapping defined	–	
Description	Defines for I-PDUs with ComIPduType NORMAL: If the underlying IF-modul supports cancellation of transmit requests. Defines for I-PDUs with ComIPduType TP: If the underlying TP-module supports RX and TX cancellation of ongoing requests. Dependency: This parameter shall not be set to true if ComCancellationSupport is set to false	
Multiplicity	0..1	
Type	EcucBooleanParamDef	
Default value	false	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00493] Definition of EcucEnumerationParamDef ComIPduDirection ⌈

Parameter Name	ComIPduDirection	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The direction defines if this I-PDU, and therefore the contributing signals and signal groups, shall be sent or received. Dependency: If configured to Sent also a ComTxIpdu container shall be included, see ECUC_Com_00496	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	RECEIVE	Receive
	SEND	Send
Pre-Compile only	true	

⌋

[ECUC_Com_00175] Definition of EcucIntegerParamDef ComIPduHandleId ⌈

Parameter Name	ComIPduHandleId
Parent Container	ComIPdu
Upstream mapping defined	–
Description	The numerical value used as the ID of this I-PDU. The ComIPduHandleId is required by the API calls Com_RxIndication, Com_TpRxIndication, Com_StartOfReception and Com_CopyRxData to receive I-PDUs from the PduR (ComIP-duDirection: Receive), as well as the PduId passed to an Rx-I-PDU-callout. For Tx-I-PDUs (ComIPduDirection: Send), this handle Id is used for the APIs calls Com_TxConfirmation, Com_Trigger Transmit, Com_TriggerIPDUSend or Com_TriggerIPDUSendWithMetaData, Com_Copy TxData and Com_TpTxConfirmation to transmit respectively confirm transmissions of I-PDUs, as well as the PduId passed to the Tx-I-PDU-callout configured with ComIPdu Callout and/or ComIPduTriggerTransmitCallout. This ID is used also in Com_Transmit for I-PDUs with ComIPduLargeData set to true.
Multiplicity	0..1

▽

△

Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_Com_10045] Definition of EcucBooleanParamDef ComIPduLargeData ⌈

Parameter Name	ComIPduLargeData
Parent Container	ComIPdu
Upstream mapping defined	–
Description	This parameter enables/disables Large Data Path support for the I-PDU.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_00119] Definition of EcucEnumerationParamDef ComIPduSignalProcessing ⌈

Parameter Name	ComIPduSignalProcessing	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	For the definition of the two modes Immediate and Deferred.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEFERRED	signal indication / confirmations are deferred for example to a cyclic task
	IMMEDIATE	the signal indications / confirmations are performed in Com_RxIndication/ Com_Tx Confirmation
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00765] Definition of EcucFunctionNameDef ComIPduTriggerTransmitCallout ⌈

Parameter Name	ComIPduTriggerTransmitCallout
Parent Container	ComIPdu
Upstream mapping defined	–
Description	If there is a trigger transmit callout defined for this I-PDU this parameter contains the name of the callout function.
Multiplicity	0..1
Type	EcucFunctionNameDef

▽

△

Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Com_00761] Definition of EcucEnumerationParamDef ComIPduType ⌈

Parameter Name	ComIPduType	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines if this I-PDU is a normal I-PDU that can be sent unfragmented or if this is a large I-PDU that shall be sent via the Transport Protocol of the underlying bus.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	NORMAL	sent or received via normal L-PDU
	TP	sent or received via TP
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00206] Definition of EcucReferenceDef ComIPduGroupRef ⌈

Parameter Name	ComIPduGroupRef	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to the I-PDU groups this I-PDU belongs to.	
Multiplicity	0..*	
Type	Reference to ComIPduGroup	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_10012] Definition of EcucChoiceReferenceDef ComIPduMainFunctionRef ⌈

Parameter Name	ComIPduMainFunctionRef
Parent Container	ComIPdu
Upstream mapping defined	–
Description	Reference to the Com_MainFunctionRx/Com_MainFunctionTx this I-PDU belongs to. Mandatory, if multiple main functions of the relevant type are defined.
Multiplicity	0..1
Type	Choice reference to [ComMainFunctionRx, ComMainFunctionTx]
Pre-Compile only	true

⌋

[ECUC_Com_00519] Definition of EcucReferenceDef ComIPduSignalGroupRef ⌈

Parameter Name	ComIPduSignalGroupRef	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	References to all signal groups contained in this I-Pdu	
Multiplicity	0..*	
Type	Reference to ComSignalGroup	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00518] Definition of EcucReferenceDef ComIPduSignalRef ⌈

Parameter Name	ComIPduSignalRef	
Parent Container	ComIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	References to all signals contained in this I-PDU.	
Multiplicity	0..*	
Type	Reference to ComSignal	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_10021] Definition of EcucReferenceDef ComMainFunctionRouteSignalsRef ⌈

Parameter Name	ComMainFunctionRouteSignalsRef
Parent Container	ComIPdu
Upstream mapping defined	–
Description	Reference to ComMainFunctionRouteSignals which performs signal gateway related activities.
Multiplicity	0..1
Type	Reference to ComMainFunctionRouteSignals
Pre-Compile only	true

⌋

[ECUC_Com_00711] Definition of EcucReferenceDef ComPduIdRef ⌈

Parameter Name	ComPduIdRef
Parent Container	ComIPdu
Upstream mapping defined	–
Description	Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.
Multiplicity	1
Type	Reference to Pdu
Pre-Compile only	true

⌋

10.2.11 ComMetaDataDefaultItem

[ECUC_Com_10022] Definition of EcucParamConfContainerDef ComMetaDataDefaultItem ⌈

Container Name	ComMetaDataDefaultItem	
Parent Container	ComIPdu	
Upstream mapping defined	–	
Description	Defines a default value for a meta data item. Used for sending an I-PDU with meta data when it is triggered spontaneously (and not by Com_TriggerIPDUSendWithMetaData), and no meta data has been provided by the RTE. Also used for reception, when e.g. Com_ReceiveSignalWithMetaData is called before an I-PDU is received. It represents a MetaDataItem of the referenced global PDU.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComMetaDataDefaultValue	1	[ECUC_Com_10023]
ComMetaDataItemRef	1	[ECUC_Com_10024]

No Included Containers

⌋

[ECUC_Com_10023] Definition of EcucIntegerParamDef ComMetaDataDefaultValue ⌈

Parameter Name	ComMetaDataDefaultValue	
Parent Container	ComMetaDataDefaultItem	
Upstream mapping defined	–	
Description	Default value for MetaDataItem of the global PDU.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 18446744073709551615	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_10024] Definition of EcucReferenceDef ComMetaDataItemRef ⌈

Parameter Name	ComMetaDataItemRef	
Parent Container	ComMetaDataDefaultItem	
Upstream mapping defined	–	
Description	Reference to a MetaDataItem of the global PDU.	
Multiplicity	1	
Type	Reference to MetaDataItem	
Post-Build Value	Loadable	Selectable

⌋

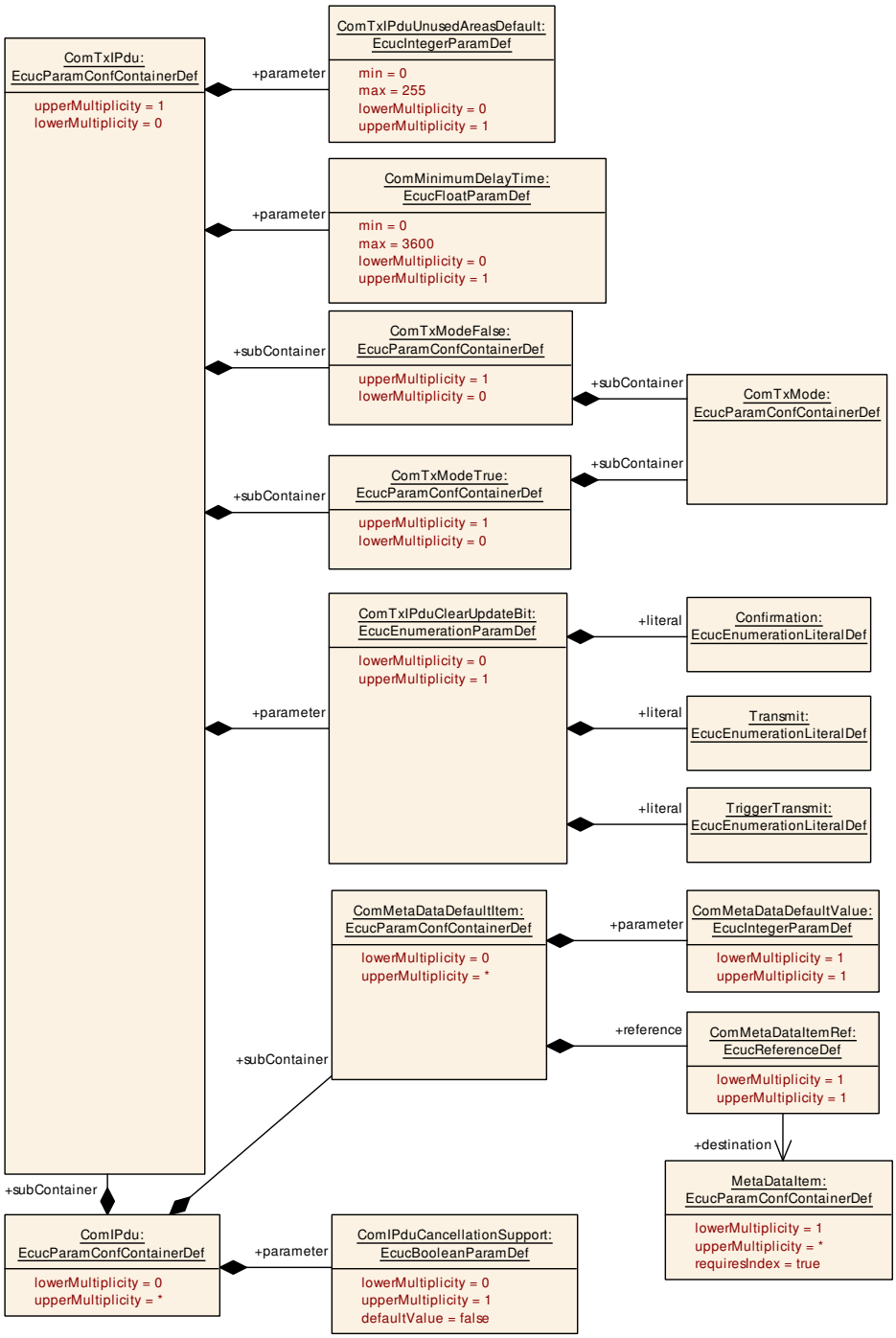


Figure 10.6: ComTxIPdu

10.2.12 ComTxIPdu

[ECUC_Com_00496] Definition of EcucParamConfContainerDef ComTxIPdu ⌈

Container Name	ComTxIPdu
Parent Container	ComIPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This container contains additional transmission related configuration parameters of the AUTOSAR COM module's I-PDUs.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComMinimumDelayTime	0..1	[ECUC_Com_00181]
ComTxIPduClearUpdateBit	0..1	[ECUC_Com_00576]
ComTxIPduUnusedAreasDefault	0..1	[ECUC_Com_00017]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComTxModeFalse	0..1	The referenced transmission mode object that is used when the filtering state for this I-PDU evaluates to false. The default is transmission mode None.
ComTxModeTrue	0..1	The referenced transmission mode object that is used when the filtering state for this I-PDU evaluates to true.

⌋

[ECUC_Com_00181] Definition of EcucFloatParamDef ComMinimumDelayTime ⌈

Parameter Name	ComMinimumDelayTime	
Parent Container	ComTxIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the Minimum Delay Time (MDT) between successive transmissions of this I-PDU in seconds. The MDT is independent of the possible different transmission modes. There is only one minimum delay time parameter for one I-PDU. The minimum delay timer is not reset by changing the transmission mode. Hence, it is not allowed to violate the minimum delay time by transmission mode changes. It is not possible to monitor the minimum delay time for I-PDUs that are requested using the Com_Trigger Transmit API.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00576] Definition of EcucEnumerationParamDef ComTxIPduClearUpdateBit ⌈

Parameter Name	ComTxIPduClearUpdateBit	
Parent Container	ComTxIPdu	
Upstream mapping defined	–	
Description	Defines when the update-bits of signals or signal groups, contained in this I-PDU, will be cleared.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	Confirmation	The update-bits are cleared when the transmission of the I-PDU was confirmed. In case of Direct/N-Times transmission mode the update bits will be cleared with respect to the confirmation behaviour of SWS_Com_00305.
	Transmit	The update-bits are cleared directly after the invocation of PduR_ComTransmit.
	TriggerTransmit	The update-bits are cleared after the I-PDU was fetched via Com_TriggerTransmit.
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00017] Definition of EcucIntegerParamDef ComTxIPduUnusedAreasDefault ⌈

Parameter Name	ComTxIPduUnusedAreasDefault	
Parent Container	ComTxIPdu	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The AUTOSAR COM module fills not used areas of an I-PDU with this byte pattern. This attribute is mandatory to avoid undefined behaviour. This byte-pattern will be repeated throughout the I-PDU before any init-values or update-bits were set.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

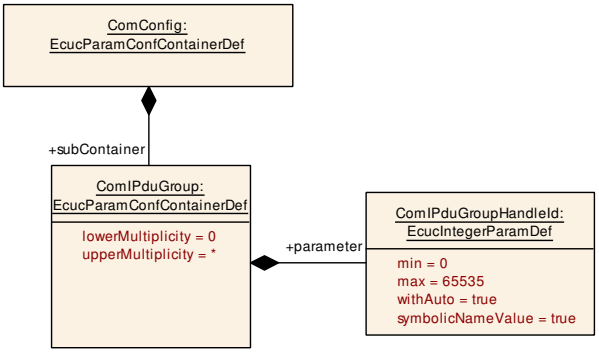


Figure 10.7: ComIPduGroup

10.2.13 ComIPduGroup

[ECUC_Com_00341] Definition of EcucParamConfContainerDef ComIPduGroup ⌈

Container Name	ComIPduGroup	
Parent Container	ComConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters of the AUTOSAR COM module's I-PDU groups.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComIPduGroupHandleId	1	[ECUC_Com_00184]

No Included Containers

⌋

[ECUC_Com_00184] Definition of EcucIntegerParamDef ComIPduGroupHandleId ⌈

Parameter Name	ComIPduGroupHandleId	
Parent Container	ComIPduGroup	
Upstream mapping defined	–	
Description	The numerical value used as the ID of this I-PDU Group . The ComIPduGroupHandleId is required by the API calls to start and stop I-PDU Groups. Range: 0 .. (ComSupportedIPduGroups-1)	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

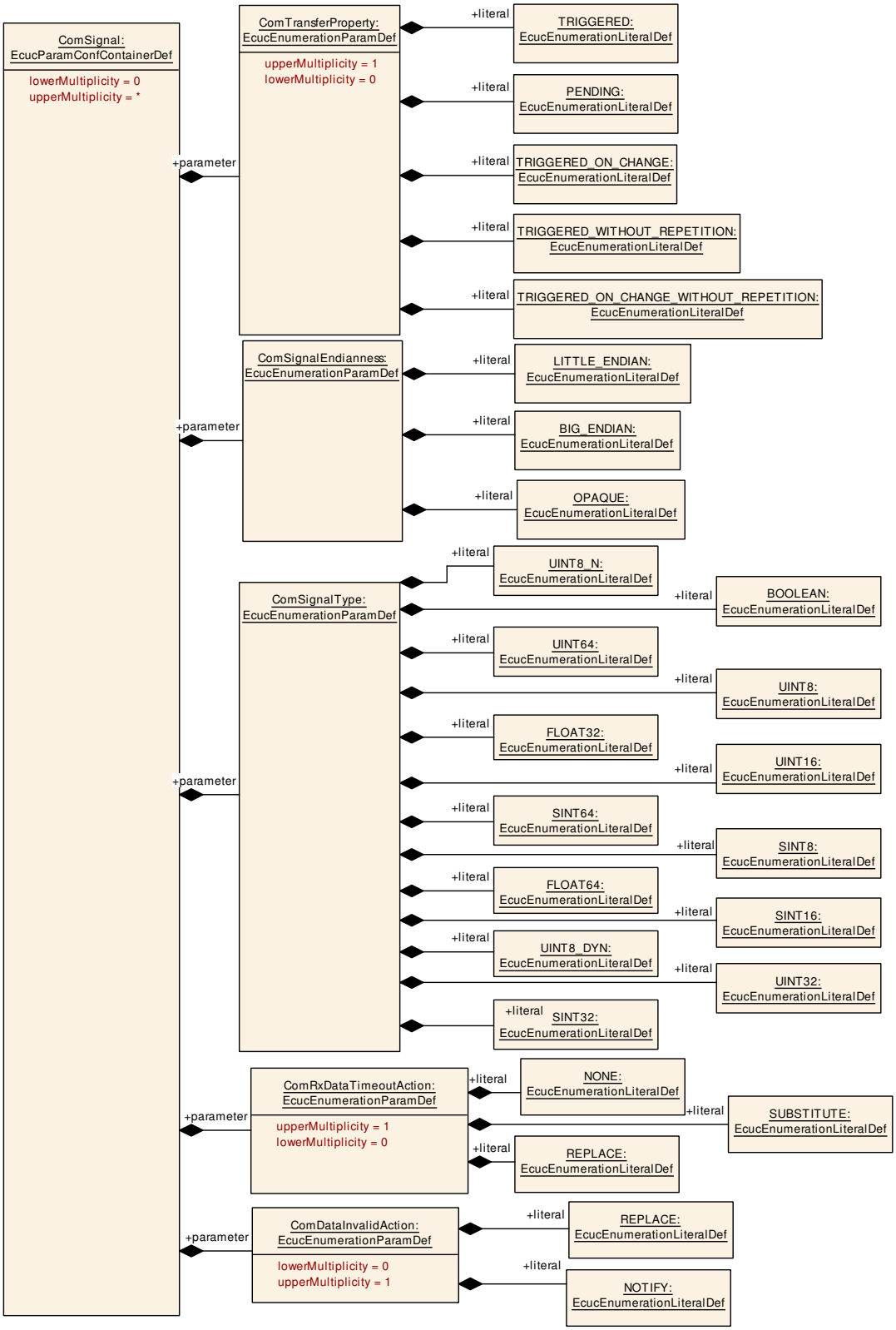


Figure 10.8: ComSignal

10.2.14 ComSignal

[ECUC_Com_00344] Definition of EcucParamConfContainerDef ComSignal ⌈

Container Name	ComSignal	
Parent Container	ComConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters of the AUTOSAR COM module's signals.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComBitPosition	1	[ECUC_Com_00259]
ComBitSize	0..1	[ECUC_Com_00158]
ComDataInvalidAction	0..1	[ECUC_Com_00314]
ComFirstTimeout	0..1	[ECUC_Com_00183]
ComHandleId	0..1	[ECUC_Com_00165]
ComInitialValueOnly	0..1	[ECUC_Com_00811]
ComRxDataTimeoutAction	0..1	[ECUC_Com_00412]
ComSignalDataInvalidValue	0..1	[ECUC_Com_00391]
ComSignalEndianness	1	[ECUC_Com_00157]
ComSignalInitValue	0..1	[ECUC_Com_00170]
ComSignalLength	0..1	[ECUC_Com_00437]
ComSignalType	1	[ECUC_Com_00127]
ComTimeout	0..1	[ECUC_Com_00263]
ComTimeoutSubstitutionValue	0..1	[ECUC_Com_10006]
ComTransferProperty	0..1	[ECUC_Com_00232]
ComUpdateBitPosition	0..1	[ECUC_Com_00257]
ComSystemTemplateSystemSignalRef	0..1	[ECUC_Com_00002]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameters DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
ComFilter	0..1	This container contains the configuration parameters of the AUTOSAR COM module's Filters. Note: On sender side the container is used to specify the transmission mode conditions.

⌋

For parameter table [ECUC_Com_00259] ComBitPosition, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00158] ComBitSize, see definition below container ComGroupSignal.

[ECUC_Com_00314] Definition of EcucEnumerationParamDef ComDataInvalidAction ⌈

Parameter Name	ComDataInvalidAction	
Parent Container	ComSignal, ComSignalGroup	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate]	
Description	This parameter defines the action performed upon reception of an invalid signal. Relating to signal groups the action in case if one of the included signals is an invalid signal. If Replace is used the ComSignalInitValue will be used for the replacement.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	NOTIFY	Notify (via callback)
	REPLACE	Signals shall be replaced by their ComSignalInit Value.
Pre-Compile only	true	

⌋

[ECUC_Com_00183] Definition of EcucFloatParamDef ComFirstTimeout ⌈

Parameter Name	ComFirstTimeout	
Parent Container	ComSignal, ComSignalGroup	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the length of the first deadline monitoring timeout period in seconds. This timeout is used immediately after start (or restart) of the deadline monitoring service. The timeout period of the successive periods is configured by ECUC_Com_00263.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

For parameter table [ECUC_Com_00165] ComHandleId, see definition below container ComGroupSignal.

[ECUC_Com_00811] Definition of EcucBooleanParamDef ComInitialValueOnly ⌈

Parameter Name	ComInitialValueOnly
Parent Container	ComSignal, ComSignalGroup
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This parameter defines that the respective signal's initial value shall be put into the respective PDU but there will not be any update of the value through the users (e.g. RTE, SwCluC). Thus the Com implementation does not need to expect any API calls for this signal (group).
Multiplicity	0..1

▽

△

Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_Com_00412] Definition of EcucEnumerationParamDef ComRxDataTimeoutAction ⌈

Parameter Name	ComRxDataTimeoutAction	
Parent Container	ComSignal, ComSignalGroup	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This parameter defines the action performed upon expiration of the reception deadline monitoring timer.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	NONE	no replacement shall take place
	REPLACE	Signals shall be replaced by their ComSignalInit Value.
	SUBSTITUTE	signals shall be replaced by their ComTimeout SubstitutionValue
Pre-Compile only	true	

⌋

For parameter table [ECUC_Com_00391] ComSignalDataInvalidValue, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00157] ComSignalEndianness, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00170] ComSignalInitValue, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00437] ComSignalLength, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00127] ComSignalType, see definition below container ComGroupSignal.

[ECUC_Com_00263] Definition of EcucFloatParamDef ComTimeout ⌈

Parameter Name	ComTimeout	
Parent Container	ComSignal, ComSignalGroup	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	Defines the length of the deadline monitoring timeout period in seconds. The period for the first timeout period can be configured separately by ECUC_Com_00183.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	

▽

△

Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

For parameter table [ECUC_Com_10006] ComTimeoutSubstitutionValue, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00232] ComTransferProperty, see definition below container ComGwDestinationDescription.

For parameter table [ECUC_Com_00257] ComUpdateBitPosition, see definition below container ComGwDestinationDescription.

For parameter table [ECUC_Com_00002] ComSystemTemplateSystemSignalRef, see definition below container ComGroupSignal.

10.2.15 ComDemEventParameterRefs

[ECUC_Com_10046] Definition of EcucParamConfContainerDef ComDemEvent ParameterRefs ⌈

Container Name	ComDemEventParameterRefs
Parent Container	ComSignal, ComSignalGroup
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The Event Id is taken from the referenced DemEventParameters DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
COM_E_TIMEOUT	0..1	[ECUC_Com_10047]

No Included Containers

⌋

[ECUC_Com_10047] Definition of EcucReferenceDef COM_E_TIMEOUT ⌈

Parameter Name	COM_E_TIMEOUT
Parent Container	ComDemEventParameterRefs
Upstream mapping defined	–

▽

△

Description	Reference to configured DEM event to report timeout status.
Multiplicity	0..1
Type	Symbolic name reference to DemEventParameter
Pre-Compile only	true

⌋

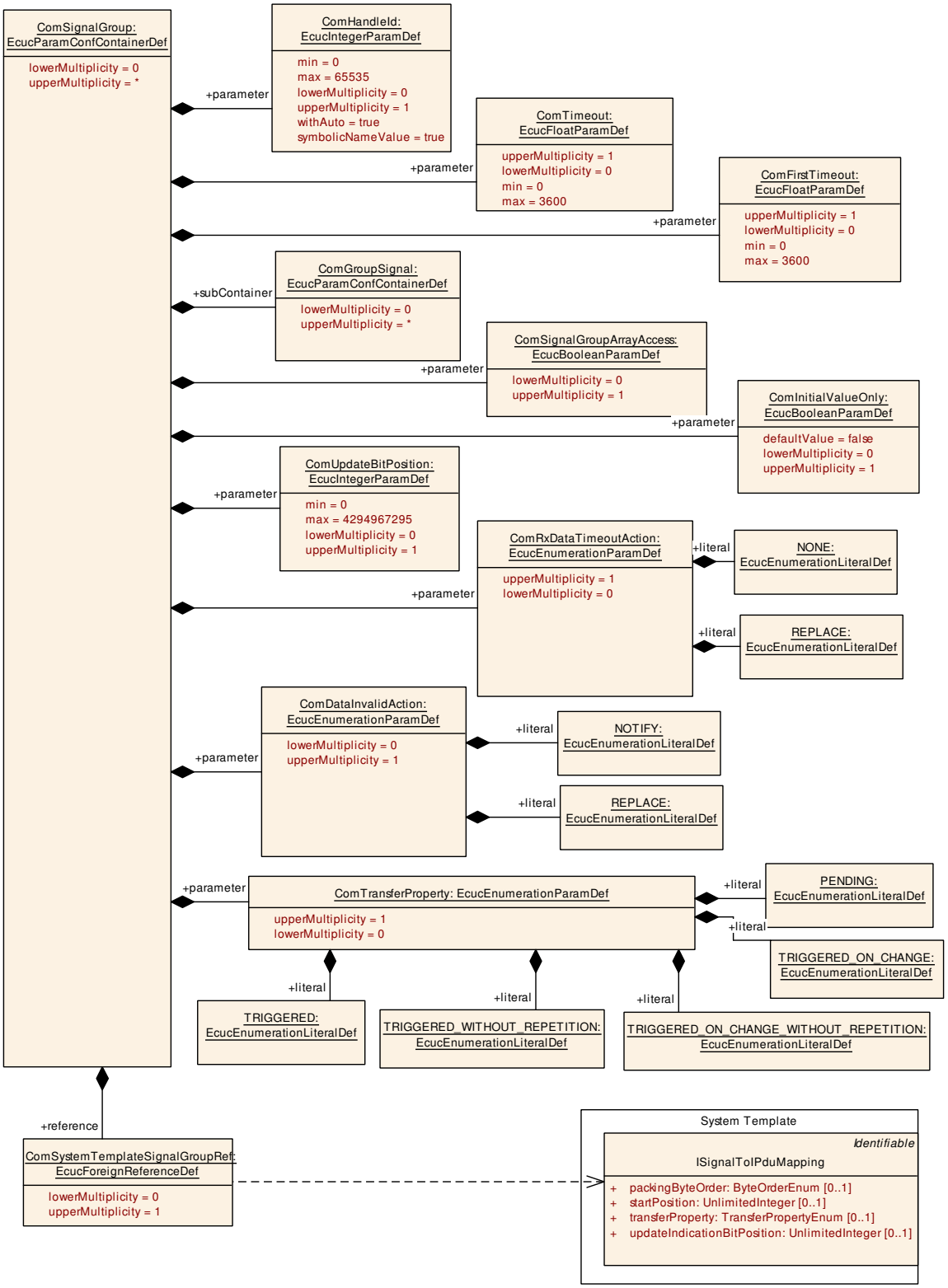


Figure 10.9: ComSignalGroup

10.2.16 ComSignalGroup

[ECUC_Com_00345] Definition of EcucParamConfContainerDef ComSignal Group ⌈

Container Name	ComSignalGroup	
Parent Container	ComConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Contains the configuration parameters of the AUTOSAR COM module's signal groups.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComDataInvalidAction	0..1	[ECUC_Com_00314]
ComFirstTimeout	0..1	[ECUC_Com_00183]
ComHandleId	0..1	[ECUC_Com_00165]
ComInitialValueOnly	0..1	[ECUC_Com_00811]
ComRxDataTimeoutAction	0..1	[ECUC_Com_00412]
ComSignalGroupArrayAccess	0..1	[ECUC_Com_10003]
ComTimeout	0..1	[ECUC_Com_00263]
ComTransferProperty	0..1	[ECUC_Com_00232]
ComUpdateBitPosition	0..1	[ECUC_Com_00257]
ComSystemTemplateSignalGroupRef	0..1	[ECUC_Com_00001]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameters DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
ComGroupSignal	0..*	This container contains the configuration parameters of group signals. I.e. signals that are included within a signal group.

⌋

For parameter table [ECUC_Com_00314] ComDataInvalidAction, see definition below container ComSignal.

For parameter table [ECUC_Com_00183] ComFirstTimeout, see definition below container ComSignal.

For parameter table [ECUC_Com_00165] ComHandleId, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00811] ComInitialValueOnly, see definition below container ComSignal.

For parameter table [ECUC_Com_00412] ComRxDataTimeoutAction, see definition below container ComSignal.

[ECUC_Com_10003] Definition of EcucBooleanParamDef ComSignalGroupArrayAccess ⌈

Parameter Name	ComSignalGroupArrayAccess
Parent Container	ComSignalGroup
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Defines whether the uint8-array based access shall be used for this ComSignalGroup.
Multiplicity	0..1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

For parameter table [ECUC_Com_00263] ComTimeout, see definition below container ComSignal.

For parameter table [ECUC_Com_00232] ComTransferProperty, see definition below container ComGwDestinationDescription.

For parameter table [ECUC_Com_00257] ComUpdateBitPosition, see definition below container ComGwDestinationDescription.

[ECUC_Com_00001] Definition of EcucForeignReferenceDef ComSystemTemplateSignalGroupRef ⌈

Parameter Name	ComSystemTemplateSignalGroupRef	
Parent Container	ComSignalGroup	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignalGroup (SystemTemplate) which this ComSignalGroup represents.	
Multiplicity	0..1	
Type	Foreign reference to I-SIGNAL-TO-I-PDU-MAPPING	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

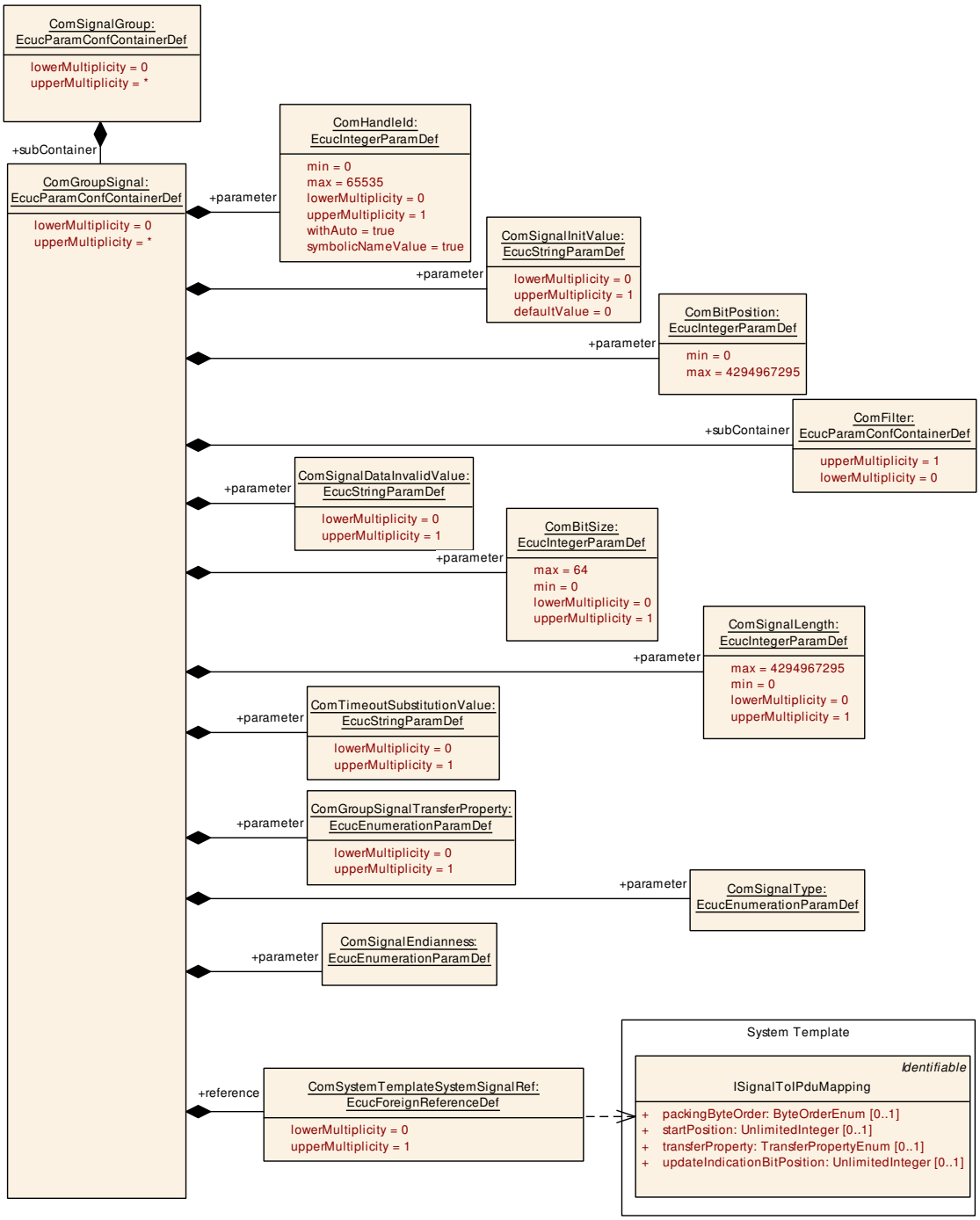


Figure 10.10: ComGroupSignal

10.2.17 ComGroupSignal

[ECUC_Com_00520] Definition of EcucParamConfContainerDef ComGroupSignal ⌈

Container Name	ComGroupSignal	
Parent Container	ComSignalGroup	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	This container contains the configuration parameters of group signals. I.e. signals that are included within a signal group.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
ComBitPosition	1	[ECUC_Com_00259]
ComBitSize	0..1	[ECUC_Com_00158]
ComGroupSignalTransferProperty	0..1	[ECUC_Com_00560]
ComHandleId	0..1	[ECUC_Com_00165]
ComSignalDataInvalidValue	0..1	[ECUC_Com_00391]
ComSignalEndianness	1	[ECUC_Com_00157]
ComSignalInitValue	0..1	[ECUC_Com_00170]
ComSignalLength	0..1	[ECUC_Com_00437]
ComSignalType	1	[ECUC_Com_00127]
ComTimeoutSubstitutionValue	0..1	[ECUC_Com_10006]
ComSystemTemplateSystemSignalRef	0..1	[ECUC_Com_00002]

Included Containers		
Container Name	Multiplicity	Description
ComFilter	0..1	This container contains the configuration parameters of the AUTOSAR COM module's filters. Note: On sender side the container is used to specify the transmission mode conditions.

⌋

[ECUC_Com_00259] Definition of EcucIntegerParamDef ComBitPosition ⌈

Parameter Name	ComBitPosition	
Parent Container	ComGroupSignal, ComGwDestinationDescription, ComGwSourceDescription, Com Signal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer. If the endianness conversion is configured to Opaque the parameter ComBitPosition shall define the bit0 of the first byte like in little endian byte order	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00158] Definition of EcucIntegerParamDef ComBitSize ⌈

Parameter Name	ComBitSize	
Parent Container	ComGroupSignal, ComGwSourceDescription, ComSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Size in bits, for integer signal types. For ComSignalType UINT8_N and UINT8_DYN the size shall be configured by ComSignalLength. For ComSignalTypes FLOAT32 and FLOAT64 the size is already defined by the signal type and therefore may be omitted.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 64	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00560] Definition of EcucEnumerationParamDef ComGroupSignal TransferProperty ⌈

Parameter Name	ComGroupSignalTransferProperty	
Parent Container	ComGroupSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Optionally defines whether this group signal shall contribute to the TRIGGERED_ON_CHANGE transfer property of the signal group. If at least one group signal of a signal group has the "ComGroupSignalTransferProperty" configured all other group signals of that signal group shall have the attribute configured as well.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	PENDING	A change of the value of this group signal shall not be considered in the evaluation of the signal groups ComTransferProperty.
	TRIGGERED_ON_CHANGE	A change of the value of this group signal shall be considered in the evaluation of the signal groups ComTransferProperty.
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00165] Definition of EcucIntegerParamDef ComHandleId ⌈

Parameter Name	ComHandleId	
Parent Container	ComGroupSignal, ComSignal, ComSignalGroup	
Upstream mapping defined	–	
Description	The numerical value used as the ID. This ID identifies signals and signal groups in the COM APIs using Com_SignalIdType or Com_SignalGroupIdType parameter respectively.	
Multiplicity	0..1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	

▽

△

With Auto	true
Pre-Compile only	true

⌋

[ECUC_Com_00391] Definition of EcucStringParamDef ComSignalDataInvalidValue ⌈

Parameter Name	ComSignalDataInvalidValue
Parent Container	ComGroupSignal, ComSignal
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	Defines the data invalid value of the signal. In case the ComSignalType is UINT8, UINT16, UINT32, UINT64, SINT8, SINT16, SINT32, SINT64 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT8_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address). For the ComSignalType UINT8_DYN the dynamic length shall be set to the number of configured characters. An empty string "" shall be interpreted as 0-sized dynamic signal. Dependency: In case of UINT8_N the length of ComSignalDataInvalidValue has to be the same as ComSignalLength.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Com_00157] Definition of EcucEnumerationParamDef ComSignalEndianness ⌈

Parameter Name	ComSignalEndianness	
Parent Container	ComGroupSignal, ComGwDestinationDescription, ComGwSourceDescription, ComSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the endianness of the signal's network representation.	
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.
	OPAQUE	Opaque data endianness
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00170] Definition of EcucStringParamDef ComSignalInitValue ⌈

Parameter Name	ComSignalInitValue	
Parent Container	ComGroupSignal, ComGwDestinationDescription, ComSignal	
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]	
Description	Initial value for this signal. In case of UINT8_N the default value is a string of length ComSignalLength with all bytes set to 0x00. In case of UINT8_DYN the initial size shall be 0. In case the ComSignalType is UINT8, UINT16, UINT32, UINT64, SINT8, SINT16, SINT32, SINT64 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT8_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address). For the ComSignalType UINT8_DYN the dynamic length shall be set to the number of configured characters. An empty string "" shall be interpreted as 0-sized dynamic signal. Dependency: In case of UINT8_N the length of ComSignalInitValue has to be the same as ComSignalLength.	
Multiplicity	0..1	
Type	EcucStringParamDef	
Default value	0	
Regular Expression	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00437] Definition of EcucIntegerParamDef ComSignalLength ⌈

Parameter Name	ComSignalLength	
Parent Container	ComGroupSignal, ComGwSourceDescription, ComSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Description: For ComSignalType UINT8_N this parameter specifies the length n in bytes. For ComSignalType UINT8_DYN it specifies the maximum length in bytes. For all other types this parameter shall be ignored. The supported maximum length is restricted by the used transportation system. For non TP-PDUs the maximum size of a PDU, and therefore also of any included signal, is limited by the concrete bus characteristic. For example, the limit is 8 bytes for CAN and LIN, 64 bytes for CAN FD and 254 for FlexRay.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Com_00127] Definition of EcucEnumerationParamDef ComSignalType ⌈

Parameter Name	ComSignalType	
Parent Container	ComGroupSignal, ComGwSourceDescription, ComSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The AUTOSAR type of the signal. Whether or not the signal is signed or unsigned can be found by examining the value of this attribute. This type could also be used to reserved appropriate storage in AUTOSAR COM.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	BOOLEAN	Type of the data is boolean.
	FLOAT32	Type of the data is float32.
	FLOAT64	Type of the data is float64.
	SINT16	Type of the data is sint16.
	SINT32	Type of the data is sint32.
	SINT64	Type of the data is sint64.
	SINT8	Type of the data is sint8.
	UINT16	Type of the data is uint16.
	UINT32	Type of the data is uint32.
	UINT64	Type of the data is uint64.
	UINT8	Type of the data is uint8.
	UINT8_DYN	Type of the data is uint8 array with dynamic length.
	UINT8_N	Type of the data is uint8 array.
Pre-Compile only	true	

⌋

[ECUC_Com_10006] Definition of EcucStringParamDef ComTimeoutSubstitution Value ⌈

Parameter Name	ComTimeoutSubstitutionValue
Parent Container	ComGroupSignal, ComSignal
Upstream mapping defined	[17, CP_TPS_SoftwareComponentTemplate], [11, CP_TPS_SystemTemplate]
Description	The signal substitution value will be used in case of a timeout and ComRxDataTimeout Action is set to SUBSTITUTE. • In case of UINT8_N, the default value is a string of length ComSignalLength with all bytes set to 0x00. • In case of UINT8_DYN, the initial size shall be 0. • In case the ComSignalType is UINT8, UINT16, UINT32, UINT64, SINT8, SINT16, SINT32, SINT64, the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. • In case the ComSignalType is FLOAT32, FLOAT64, the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. • In case the ComSignalType is BOOLEAN, the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. • In case the ComSignal is a UINT8_N, UINT8_DYN, the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address). For the ComSignalType UINT8_DYN, the dynamic length shall be set to the number of configured characters. An empty string "" shall be interpreted as 0-sized dynamic signal.

▽

△

Multiplicity	0..1	
Type	EcucStringParamDef	
Default value	–	
Regular Expression	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00002] Definition of EcucForeignReferenceDef ComSystemTemplateSystemSignalRef ⌈

Parameter Name	ComSystemTemplateSystemSignalRef	
Parent Container	ComGroupSignal, ComSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.	
Multiplicity	0..1	
Type	Foreign reference to I-SIGNAL-TO-I-PDU-MAPPING	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

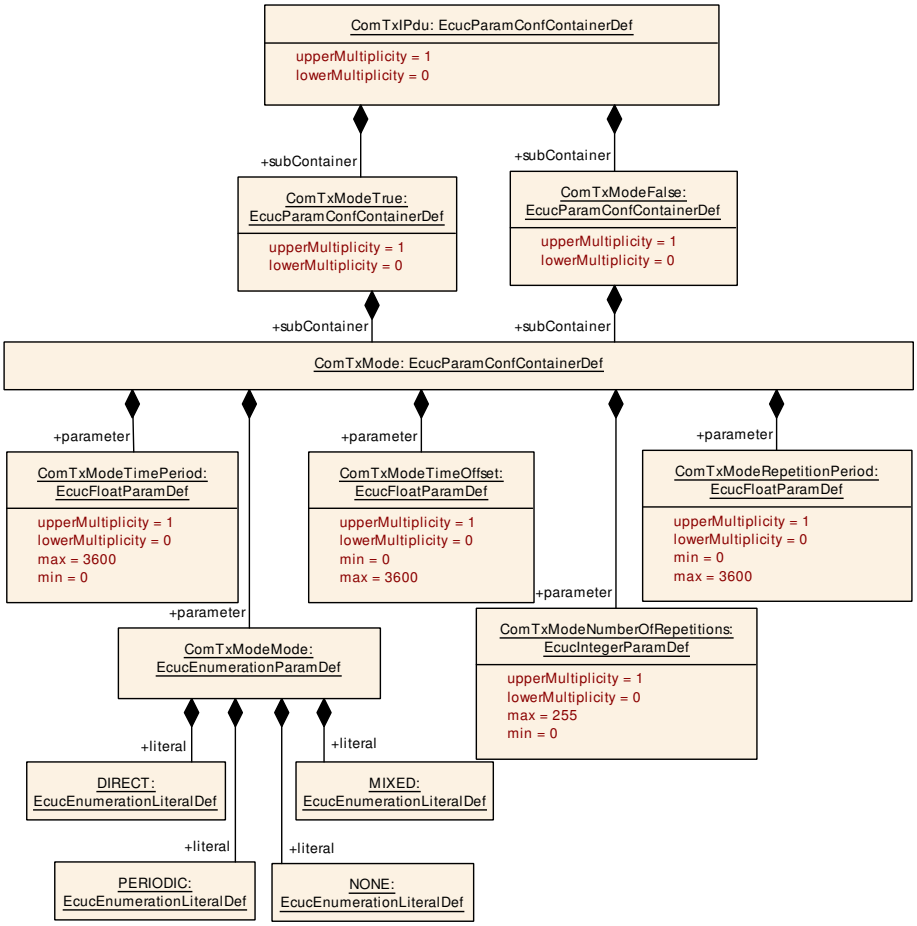


Figure 10.11: ComTxMode

10.2.18 ComTxMode

[ECUC_Com_00351] Definition of EcucParamConfContainerDef ComTxMode ⌈

Container Name	ComTxMode
Parent Container	ComTxModeFalse, ComTxModeTrue
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This container contains the configuration parameters of the AUTOSAR COM module's transmission modes.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComTxModeMode	1	[ECUC_Com_00137]
ComTxModeNumberOfRepetitions	0..1	[ECUC_Com_00281]
ComTxModeRepetitionPeriod	0..1	[ECUC_Com_00282]

▽

△

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComTxModeTimeOffset	0..1	[ECUC_Com_00180]
ComTxModeTimePeriod	0..1	[ECUC_Com_00178]

No Included Containers

⌋

[ECUC_Com_00137] Definition of EcucEnumerationParamDef ComTxModeMode ⌈

Parameter Name	ComTxModeMode	
Parent Container	ComTxMode	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	The transmission mode None shall not have any further sub-attributes in the ComTx Mode object.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DIRECT	Direct (triggered transfer)
	MIXED	Direct and periodic
	NONE	Literal for TxMode
	PERIODIC	Periodic
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00281] Definition of EcucIntegerParamDef ComTxModeNumberOfRepetitions ⌈

Parameter Name	ComTxModeNumberOfRepetitions	
Parent Container	ComTxMode	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the number of repetitions for the transmission mode DIRECT and the event driven part of transmission mode MIXED.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00282] Definition of EcucFloatParamDef ComTxModeRepetitionPeriod ⌈

Parameter Name	ComTxModeRepetitionPeriod	
Parent Container	ComTxMode	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the repetition period in seconds of the multiple transmissions in case ComTx ModeNumberOfRepetitions is configured greater than or equal to 1 and ComTxMode Mode is configured to DIRECT or MIXED. In case of the mixed transmission mode only the event driven part is affected.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00180] Definition of EcucFloatParamDef ComTxModeTimeOffset ⌈

Parameter Name	ComTxModeTimeOffset	
Parent Container	ComTxMode	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the period in seconds between the start of the I-PDU by Com_IpduGroupStart and the first transmission request in case ComTxModeMode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected. In case ComTxModeTimeOffset is omitted or configured to 0, the first periodic transmission shall be transmitted within the next invocation of Com_MainFunctionTx.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00178] Definition of EcucFloatParamDef ComTxModeTimePeriod ⌈

Parameter Name	ComTxModeTimePeriod	
Parent Container	ComTxMode	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines the repetition period in seconds of the periodic transmission requests in case ComTxModeMode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected.	
Multiplicity	0..1	
Type	EcucFloatParamDef	
Range	[0 .. 3600]	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable

▽

△

Post-Build Value	Loadable	Selectable

⌋

10.2.19 ComTxModeTrue

[ECUC_Com_00455] Definition of EcucParamConfContainerDef ComTxMode True ⌈

Container Name	ComTxModeTrue
Parent Container	ComTxIPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This container contains the configuration parameters of the AUTOSAR COM module's transmission modes in the case the ComFilter evaluates to true.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Description
ComTxMode	1	This container contains the configuration parameters of the AUTOSAR COM module's transmission modes.

⌋

10.2.20 ComTxModeFalse

[ECUC_Com_00454] Definition of EcucParamConfContainerDef ComTxMode False ⌈

Container Name	ComTxModeFalse
Parent Container	ComTxIPdu
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This container contains the configuration parameters of the AUTOSAR COM module's transmission modes in the case the ComFilter evaluates to false.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	Multiplicity	Description
ComTxMode	1	This container contains the configuration parameters of the AUTOSAR COM module's transmission modes.

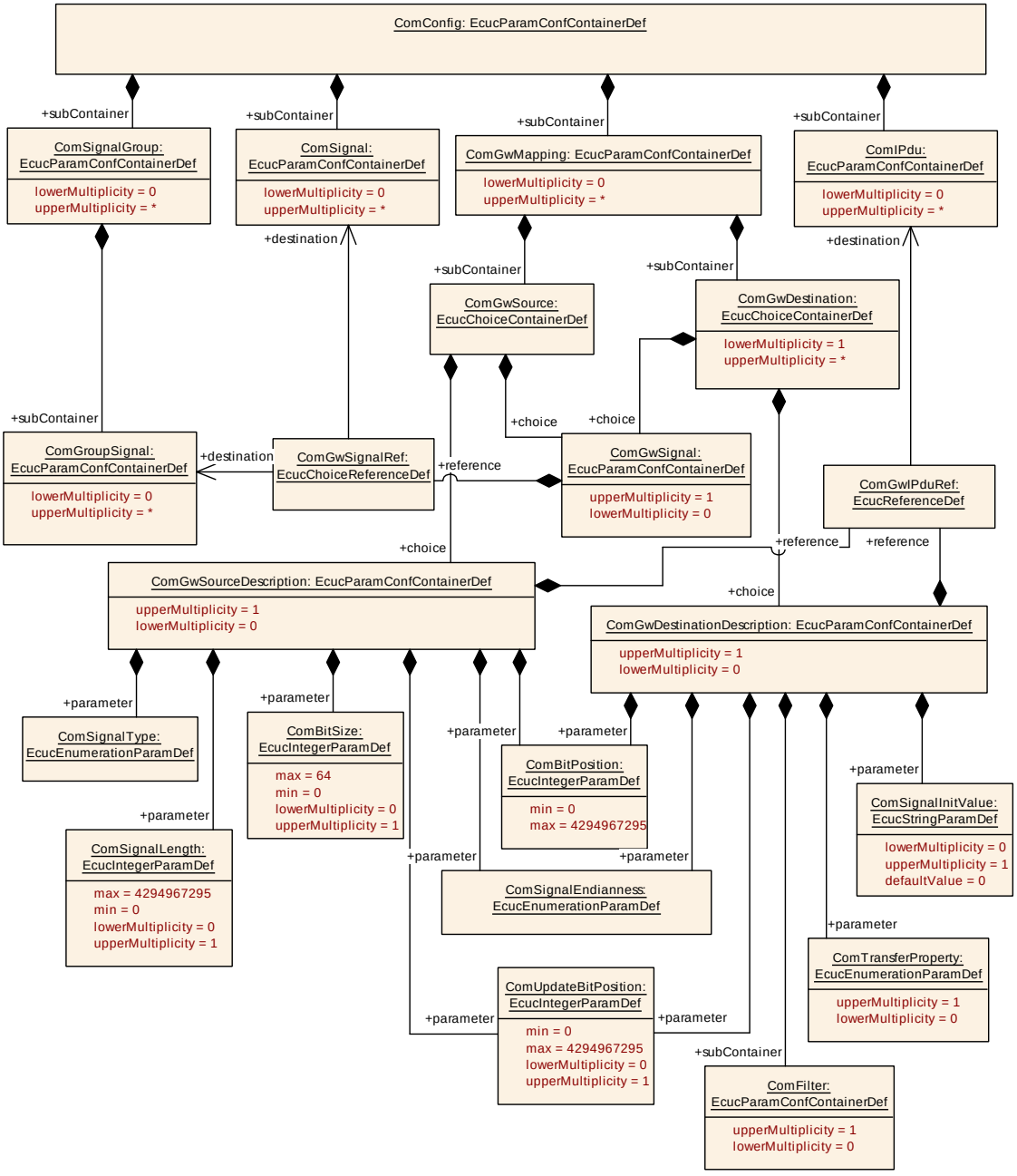


Figure 10.12: ComGwMapping

10.2.21 ComGwMapping

[ECUC_Com_00544] Definition of EcucParamConfContainerDef ComGwMapping

Container Name	ComGwMapping	
Parent Container	ComConfig	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Each instance of this container defines one mapping of the integrated Signal Gateway.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

No Included Parameters

Included Containers		
Container Name	Multiplicity	Description
ComGwDestination	1..*	Each instance of this choice container allows to define one routing destination either by reference to an already configured COM signal / group signal or by a destination description container.
ComGwSource	1	This choice container allows the definition of the gateway source signal either by reference to an already configured COM signal / group signal or by a source description container.

⌋

10.2.22 ComGwSource

[ECUC_Com_00545] Definition of EcucChoiceContainerDef ComGwSource ⌈

Choice Container Name	ComGwSource
Parent Container	ComGwMapping
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	This choice container allows the definition of the gateway source signal either by reference to an already configured COM signal / group signal or by a source description container.
Multiplicity	1
Pre-Compile only	true

No Included Parameters

Container Choices		
Container Name	Multiplicity	Description
ComGwSignal	0..1	This container allows specifying a gateway source or destination respectively with a reference to a ComSignal or a ComGroup Signal.
ComGwSourceDescription	0..1	Description of a gateway source. This container allows defining a gateway source without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.

⌋

10.2.23 ComGwSourceDescription

[ECUC_Com_00548] Definition of EcucParamConfContainerDef ComGwSource Description ⌈

Container Name	ComGwSourceDescription
Parent Container	ComGwSource
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Description of a gateway source. This container allows defining a gateway source without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComBitPosition	1	[ECUC_Com_00259]
ComBitSize	0..1	[ECUC_Com_00158]
ComSignalEndianness	1	[ECUC_Com_00157]
ComSignalLength	0..1	[ECUC_Com_00437]
ComSignalType	1	[ECUC_Com_00127]
ComUpdateBitPosition	0..1	[ECUC_Com_00257]
ComGwIPduRef	1	[ECUC_Com_00550]

No Included Containers

⌋

For parameter table [ECUC_Com_00259] ComBitPosition, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00158] ComBitSize, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00157] ComSignalEndianness, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00437] ComSignalLength, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00127] ComSignalType, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00257] ComUpdateBitPosition, see definition below container ComGwDestinationDescription.

For parameter table [ECUC_Com_00550] ComGwIPduRef, see definition below container ComGwDestinationDescription.

10.2.24 ComGwDestination

[ECUC_Com_00546] Definition of EcucChoiceContainerDef ComGwDestination ⌈

Choice Container Name	ComGwDestination	
Parent Container	ComGwMapping	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Each instance of this choice container allows to define one routing destination either by reference to an already configured COM signal / group signal or by a destination description container.	
Multiplicity	1..*	
Post-Build Multiplicity	Loadable	Selectable

No Included Parameters

Container Choices		
Container Name	**Multiplicity**	**Description**
ComGwDestinationDescription	0..1	Description of a gateway destination. This container allows defining a gateway destination without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.
ComGwSignal	0..1	This container allows specifying a gateway source or destination respectively with a reference to a ComSignal or a ComGroup Signal.

⌋

10.2.25 ComGwDestinationDescription

[ECUC_Com_00549] Definition of EcucParamConfContainerDef ComGwDestinationDescription ⌈

Container Name	ComGwDestinationDescription
Parent Container	ComGwDestination
Upstream mapping defined	[11, CP_TPS_SystemTemplate]
Description	Description of a gateway destination. This container allows defining a gateway destination without the configuration of a complete COM signal. This allows adding / changing gateway relations post build without the configuration of new signals.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComBitPosition	1	[ECUC_Com_00259]
ComSignalEndianness	1	[ECUC_Com_00157]
ComSignalInitValue	0..1	[ECUC_Com_00170]

▽

△

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComTransferProperty	0..1	[ECUC_Com_00232]
ComUpdateBitPosition	0..1	[ECUC_Com_00257]
ComGwIPduRef	1	[ECUC_Com_00550]

Included Containers		
Container Name	**Multiplicity**	**Description**
ComFilter	0..1	This container contains the configuration parameters of the AUTOSAR COM module's Filters. Note: On sender side the container is used to specify the transmission mode conditions.

⌋

For parameter table [ECUC_Com_00259] ComBitPosition, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00157] ComSignalEndianness, see definition below container ComGroupSignal.

For parameter table [ECUC_Com_00170] ComSignalInitValue, see definition below container ComGroupSignal.

[ECUC_Com_00232] Definition of EcucEnumerationParamDef ComTransferProperty ⌈

Parameter Name	ComTransferProperty	
Parent Container	ComGwDestinationDescription, ComSignal, ComSignalGroup	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Defines if a write access to this signal can trigger the transmission of the corresponding I-PDU. If the I-PDU is triggered, depends also on the transmission mode of the corresponding I-PDU.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	PENDING	A write access to this signal never triggers the transmission of the corresponding I-PDU.
	TRIGGERED	Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU.
	TRIGGERED_ON_CHANGE	Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU, but only in case the written value is different to the locally stored (last sent or initial value) in length or value.
	TRIGGERED_ON_CHANGE_WITHOUT_REPETITION	Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition, but only in case the written value is different to the locally stored (last sent or initial value) in length or value.

▽

△

	TRIGGERED_WITHOUT_REPETITION	Depending on the transmission mode, a write access to this signal can trigger the transmission of the corresponding I-PDU just once without a repetition.
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00257] Definition of EcucIntegerParamDef ComUpdateBitPosition ⌈

Parameter Name	ComUpdateBitPosition	
Parent Container	ComGwDestinationDescription, ComGwSourceDescription, ComSignal, ComSignal Group	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Bit position of update-bit inside I-PDU. If this attribute is omitted then there is no update-bit. This setting must be consistently on sender and on receiver side. Range: 0..63 for CAN and LIN, 0..511 for CAN FD, 0..2031 for FlexRay, 0..4294967295 for TP.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	–	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Com_00550] Definition of EcucReferenceDef ComGwIPduRef ⌈

Parameter Name	ComGwIPduRef	
Parent Container	ComGwDestinationDescription, ComGwSourceDescription	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to an I-PDU of a Signal Gateway source or destination description.	
Multiplicity	1	
Type	Reference to ComIPdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.26 ComGwSignal

[ECUC_Com_00551] Definition of EcucParamConfContainerDef ComGwSignal ⌈

Container Name	ComGwSignal
Parent Container	ComGwDestination, ComGwSource
Upstream mapping defined	[11, CP_TPS_SystemTemplate]

▽

△

Description	This container allows specifying a gateway source or destination respectively with a reference to a ComSignal or a ComGroupSignal.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComGwSignalRef	1	[ECUC_Com_00547]

No Included Containers

⌋

[ECUC_Com_00547] Definition of EcucChoiceReferenceDef ComGwSignalRef ⌈

Parameter Name	ComGwSignalRef	
Parent Container	ComGwSignal	
Upstream mapping defined	[11, CP_TPS_SystemTemplate]	
Description	Reference to an object of a gateway relation. Either to a ComSignal or a ComGroup Signal.	
Multiplicity	1	
Type	Choice reference to [ComGroupSignal, ComSignal]	
Post-Build Value	Loadable	Selectable

⌋

10.2.27 ComUserModule

[ECUC_Com_10031] Definition of EcucParamConfContainerDef ComUserModule ⌈

Container Name	ComUserModule
Parent Container	ComConfig
Upstream mapping defined	–
Description	Contains the configuration parameters of the Com user modules.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComUserModuleCnfRef	1	[ECUC_Com_10029]

No Included Containers

⌋

[ECUC_Com_10029] Definition of EcucUriReferenceDef ComUserModuleCnfRef ⌈

Parameter Name	ComUserModuleCnfRef
Parent Container	ComUserModule
Upstream mapping defined	–
Description	Reference to the Com user module configuration.
Multiplicity	1
Type	Reference to destinationUri ComUserUriDefSet/ComUser
Pre-Compile only	true

⌋

10.2.28 ComUserUriDefSet

[ECUC_Com_10039] Definition of EcucDestinationUriDefSet ComUserUriDefSet ⌈

EcucDestinationUriDefSet Name	ComUserUriDefSet
Description	Defines the set of DestinationUriDefs for the Com module.
Included EcucDestinationUriDefs	
Name	**Description**
ComUser	Defines the configuration container content of the Com user modules relevant settings.

⌋

[ECUC_Com_10040] Definition of EcucDestinationUriDef ComUser ⌈

EcucDestinationUriDef Name	ComUser
Destination Uri Definition Set	ComUserUriDefSet
Description	Defines the configuration container content of the Com user modules relevant settings.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Description**
ComUserModuleCnf	0..1	Contains the configuration parameters of the Com user module.

⌋

10.2.29 ComUserModuleCnf

[ECUC_Com_10030] Definition of EcucParamConfContainerDef ComUserModuleCnf ⌈

Container Name	ComUserModuleCnf
Parent Container	RteComUser, SwCluCComProxyBaseSocket
Parent Destination Uri Definition	ComUser
Upstream mapping defined	–
Description	Contains the configuration parameters of the Com user module.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
ComUserHeaderInclude	0..1	[ECUC_Com_10032]

Included Containers		
Container Name	Multiplicity	Description
ComUserCallback	0..*	This container defines a Com callback function for ComSignals, ComSignalGroups, and ComIPdus.
ComUserIPdu	0..*	Contains the configuration parameters for the ComIPdu with Large Data Path support inside a Com user module.
ComUserSignal	0..*	Contains the configuration parameters of a signal inside a Com user module. Please note that it is valid to define ComUser Signal without any callback function.
ComUserSignalGroup	0..*	Contains the configuration parameters of a signal group inside a Com user module. Please note that it is valid to define ComUser SignalGroup without any callback function.

⌋

[ECUC_Com_10032] Definition of EcucStringParamDef ComUserHeaderInclude
⌈

Parameter Name	ComUserHeaderInclude
Parent Container	ComUserModuleCnf
Upstream mapping defined	–
Description	Defines the header file where the Com user provides the function declarations for configured callbacks.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

10.2.30 ComUserIPdu

[ECUC_Com_10048] Definition of EcucParamConfContainerDef ComUserIPdu ⌈

Container Name	ComUserIPdu	
Parent Container	ComUserModuleCnf	
Upstream mapping defined	–	
Description	Contains the configuration parameters for the ComIPdu with Large Data Path support inside a Com user module.	
Multiplicity	0..*	
Post-Build Multiplicity	-	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComUserCbkHandleId	0..1	[ECUC_Com_10033]
ComUserCallbackRef	0..*	[ECUC_Com_10035]
ComUserSystemTemplateSystemSignalRef	0..1	[ECUC_Com_10025]

No Included Containers

⌋

[ECUC_Com_10033] Definition of EcucIntegerParamDef ComUserCbkHandleId ⌈

Parameter Name	ComUserCbkHandleId	
Parent Container	ComUserIPdu, ComUserSignal, ComUserSignalGroup	
Upstream mapping defined	–	
Description	The numerical value used as the Com user callback handle Id. This ID identifies signals and signal groups in the COM callbacks using ComUserCbk HandleId parameter respectively.	
Multiplicity	0..1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Com_10035] Definition of EcucReferenceDef ComUserCallbackRef ⌈

Parameter Name	ComUserCallbackRef	
Parent Container	ComUserIPdu, ComUserSignal, ComUserSignalGroup	
Upstream mapping defined	–	
Description	Reference(s) to all callback(s) of this signal or signal group or IPdu.	
Multiplicity	0..*	
Type	Reference to ComUserCallback	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	-	Selectable

⌋

[ECUC_Com_10025] Definition of EcucForeignReferenceDef ComUserSystemTemplateSystemSignalRef ⌈

Parameter Name	ComUserSystemTemplateSystemSignalRef	
Parent Container	ComUserIPdu, ComUserSignal	
Upstream mapping defined	–	
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComUserSignal (or ComUserGroupSignal) represents.	
Multiplicity	0..1	
Type	Foreign reference to I-SIGNAL-TO-I-PDU-MAPPING	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

10.2.31 ComUserSignal

[ECUC_Com_10028] Definition of EcucParamConfContainerDef ComUserSignal ⌈

Container Name	ComUserSignal	
Parent Container	ComUserModuleCnf	
Upstream mapping defined	–	
Description	Contains the configuration parameters of a signal inside a Com user module. Please note that it is valid to define ComUserSignal without any callback function.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComUserCbkHandleId	0..1	[ECUC_Com_10033]
ComUserCallbackRef	0..*	[ECUC_Com_10035]
ComUserSystemTemplateSystemSignalRef	0..1	[ECUC_Com_10025]

No Included Containers

⌋

For parameter table [ECUC_Com_10033] ComUserCbkHandleId, see definition below container ComUserIPdu.

For parameter table [ECUC_Com_10035] ComUserCallbackRef, see definition below container ComUserIPdu.

For parameter table [ECUC_Com_10025] ComUserSystemTemplateSystemSignalRef, see definition below container ComUserIPdu.

10.2.32 ComUserSignalGroup

[ECUC_Com_10027] Definition of EcucParamConfContainerDef ComUserSignalGroup ⌈

Container Name	ComUserSignalGroup	
Parent Container	ComUserModuleCnf	
Upstream mapping defined	–	
Description	Contains the configuration parameters of a signal group inside a Com user module. Please note that it is valid to define ComUserSignalGroup without any callback function.	
Multiplicity	0..*	
Post-Build Multiplicity	Loadable	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
ComUserCbkHandleId	0..1	[ECUC_Com_10033]
ComUserCallbackRef	0..*	[ECUC_Com_10035]
ComUserSystemTemplateSignalGroupRef	0..1	[ECUC_Com_10026]

No Included Containers

⌋

For parameter table [ECUC_Com_10033] ComUserCbkHandleId, see definition below container ComUserIPdu.

For parameter table [ECUC_Com_10035] ComUserCallbackRef, see definition below container ComUserIPdu.

[ECUC_Com_10026] Definition of EcucForeignReferenceDef ComUserSystemTemplateSignalGroupRef ⌈

Parameter Name	ComUserSystemTemplateSignalGroupRef	
Parent Container	ComUserSignalGroup	
Upstream mapping defined	–	
Description	Reference to the ISignalToIPduMapping that contains a reference to the ISignalGroup (SystemTemplate) which this ComUserSignalGroup represents.	
Multiplicity	0..1	
Type	Foreign reference to I-SIGNAL-TO-I-PDU-MAPPING	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

10.2.33 ComUserCallback

[ECUC_Com_10038] Definition of EcucParamConfContainerDef ComUserCallback ⌈

Container Name	ComUserCallback	
Parent Container	ComUserModuleCnf	
Upstream mapping defined	–	
Description	This container defines a Com callback function for ComSignals, ComSignalGroups, and ComIPdus.	
Multiplicity	0..*	
Post-Build Multiplicity	-	Selectable
Post-Build Value	-	Selectable
Configuration Parameters		

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
ComUserCallbackName	1	[ECUC_Com_10036]
ComUserCallbackType	1	[ECUC_Com_10034]

No Included Containers

⌋

[ECUC_Com_10036] Definition of EcucFunctionNameDef ComUserCallback Name ⌈

Parameter Name	ComUserCallbackName
Parent Container	ComUserCallback
Upstream mapping defined	–
Description	The name of the callback function to be called.
Multiplicity	1
Type	EcucFunctionNameDef
Default value	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Com_10034] Definition of EcucEnumerationParamDef ComUserCallback Type ⌈

Parameter Name	ComUserCallbackType	
Parent Container	ComUserCallback	
Upstream mapping defined	–	
Description	The type of the Com callback	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	COM_RX_ACK	Com_CbkRxAck callback which is called immediately after the message has been stored in the receiving message object. This type of callback represents a ComNotification.

▽

△

	COM_RX_INDICATION	ComCbkRxIndication Large Data Path callback indicates a received PDU from a lower layer communication interface module.
	COM_RX_INV	Com_CbkInv callback which is called after reception of an invalid signal or signal group respectively. This type of callback represents a ComInvalidNotification.
	COM_RX_START_OF_RECEPTION	ComCbkStartOfReception Large Data Path callback called at the start of receiving an N-SDU.
	COM_RX_TOUT	Com_CbkRxTOut callback which is called immediately after a message reception error has been detected by the deadline monitoring mechanism. This type of callback represents a ComTimeoutNotification.
	COM_TP_COPY_RX_DATA	ComCbkCopyRxData Large Data Path callback to provide the received data of an I-PDU segment (N-PDU) to the upper layer.
	COM_TP_COPY_TX_DATA	ComCbkCopyTxData Large Data Path callback to acquire the transmit data of an I-PDU segment.
	COM_TP_RX_INDICATION	ComCbkTpRxIndication Large Data Path callback called after an I-PDU has been received via the TP API.
	COM_TP_TX_CONFIRMATION	ComCbkTpTxConfirmation Large Data Path callback called after a Signal has been transmitted via the TP-API on its network.
	COM_TX_ACK	Com_CbkTxAck callback which is called immediately after successful transmission of the I-PDU containing the message. This type of callback represents a ComNotification.
	COM_TX_CONFIRMATION	ComCbkTxConfirmation Large Data Path callback which is called when the lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.
	COM_TX_ERR	Com_CbkTxErr callback which is called in case the transmission is not possible because the corresponding I-PDU group is stopped. This type of callback represents a ComErrorNotification.
	COM_TX_TOUT	Com_CbkTxTOut callback which is called immediately after a message transmission error has been detected by the deadline monitoring mechanism. This type of callback represents a ComTimeoutNotification.
	COM_TX_TRIGGER_TRANSMIT	Large Data Path ComCbkTriggerTransmit callback to acquire transmit data for the lower layer communication interface module.
Pre-Compile only	true	

⌋

10.3 Configuration Rules

10.3.1 Signal Configuration

More than one signal can be packed into an I-PDU as long as the following packing rules are fulfilled:

[SWS_Com_CONSTR_00102] Signals Separated in I-PDU ⌈ComSignal / ComGroupSignal are not allowed to overlap each other.⌋

It is explicitly allowed that a ComSignal / ComGroupSignal may have the size 0, see ComBitSize.

A signal of size 0 never overlaps other signals. Therefore, it can be configured to an arbitrary position within the I-PDU. The position of a zero sized signal is only relevant in case receiving an I-PDU partially.

[SWS_Com_CONSTR_00105] Byte Aligned Array Signals ⌈ComSignal / ComGroupSignal of ComSignalType UINT8_N and UINT8_DYN shall be byte aligned (ComBitPosition is a multiple of 8).⌋

[SWS_Com_CONSTR_00443] Mapping of Array Signals ⌈A ComSignal / ComGroupSignal of type uint8[n] shall always be mapped to an n-bytes sized ComSignal / ComGroupSignal.⌋

[SWS_Com_CONSTR_00553] OPAQUE Endianness of Array Signals ⌈A ComSignal / ComGroupSignal of type uint8[n] shall be configured to have OPAQUE endianness.⌋

[SWS_Com_CONSTR_00754] Dynamic Length Signal Configuration ⌈A dynamic length I-PDU shall contain at most one dynamic length signal.⌋

[SWS_Com_CONSTR_00755] Placement of Dynamic Length Signals ⌈The dynamic length signal shall be placed last in a dynamic length I-PDU.⌋

[SWS_Com_CONSTR_00756] Size Configuration of Dynamic Length Signals ⌈The ComSignalLength parameter shall be configured to the maximum size / length for dynamic length signals.⌋

[SWS_Com_CONSTR_00310] Update Bit Excluded in case of Transmission Repetitions ⌈For I-PDUs with transmission mode DIRECT or MIXED with ComTxModeNumberOfRepetitions greater or equal 1 no update-bit (ComUpdateBitPosition) shall be configured.⌋

[SWS_Com_CONSTR_00785] Bit Size of Signals ⌈The ComBitSize of a (group) signal shall not extend past the size of its configured ComSignalType.⌋

For example, the ComBitSize of a signal with ComSignalType UINT8 shall not exceed 8 bits.

[SWS_Com_CONSTR_00762] Transfer Property of Zero Length Signals ⌈The AUTOSAR COM module shall not support the transfer properties TRIGGERED_ON_CHANGE and TRIGGERED_ON_CHANGE_WITHOUT_REPETITION for signals and group signals with ComBitSize configured to 0.⌋

[SWS_Com_CONSTR_00790] Interpretation of Signal Initial and Invalid Values ⌈The configured string of ComSignalInitValue / ComSignalDataInvalidValue shall be interpreted according to the definitions of [18] for the boolean and all numerical types.⌋

[SWS_Com_CONSTR_00859] Signal Invalid Value Configuration ⌈In case a ComSignalDataInvalidValue is configured for a ComSignal that is included in a ComIPdu with ComIPduDirection RECEIVE, the ComDataInvalidAction shall also be configured for this ComSignal.⌋

10.3.2 Signal Group Configuration

[SWS_Com_CONSTR_00365] Signal Group Gateway Configuration ⌈It shall not be allowed to configure signal groups for routing with data type differences between receive and transmit signal group. A sub-setting of signal groups (as defined in [SWS_Com_00735]) shall be supported.⌋

How the signals of signal groups are placed within an I-PDU is not restricted. It is allowed to define interlaced signal groups, or to place other signals within the holes of a signal group.

[SWS_Com_CONSTR_00860] Group Signal Invalid Value Configuration ⌈In case a ComSignalDataInvalidValue is configured for at least one ComGroupSignal of a ComSignalGroup that is included in a ComIPdu with ComIPduDirection RECEIVE, the ComDataInvalidAction shall also be configured for this ComSignalGroup.⌋

10.3.3 Transmission Mode Configuration

[SWS_Com_CONSTR_00497] Mandatory ComTxIPdu for TX I-PDUs ⌈A ComTxIPdu container must be included if ComIPduDirection is configured to SEND.⌋

[SWS_Com_CONSTR_00465] Transmission Mode Configuration ⌈Every ComTxModeTrue or ComTxModeFalse that is a potential result of the configured / calculated TMS must be configured. Within the ComTxIPdu, at least one of the containers ComTxModeTrue or ComTxModeFalse has to be included.⌋

If ComIPduLargeData is set to true then containers ComTxModeTrue and ComTxModeFalse are not needed, see [SWS_Com_CONSTR_00942].

10.3.4 Signal Gateway Configuration

[SWS_Com_CONSTR_00384] Routed Signal Size Configuration ⌈The ComBitSize of a received and to be routed ComSignal shall not differ.⌋

Additionally to reception deadline monitoring of a ComSignal, to be routed by the Signal Gateway, it is possible to configure update-bits via ComUpdateBitPosition, for the transmit signal. In this case, the receiving node can detect if the sender has really updated the signal or it is just repeated by the Signal Gateway. If this is necessary depends on the use-case.

10.3.5 Filter Configuration

[SWS_Com_CONSTR_00380] Float and Array Signal Filter Algorithm Limitations ⌈For signals with ComSignalType FLOAT32, FLOAT64, UINT8_N or UINT8_DYN, the AUTOSAR COM module shall only support ComFilterAlgorithm configured to ALWAYS, NEVER or ONE_EVERY_N.⌋

[SWS_Com_CONSTR_00439] Boolean Signal Filter Algorithm Limitations ⌈For signals with ComSignalType configured to BOOLEAN, the AUTOSAR COM module shall only support ComFilterAlgorithm configured to:

- ALWAYS
- NEVER
- MASKED_NEW_EQUALS_X
- MASKED_NEW_DIFFERS_X
- MASKED_NEW_DIFFERS_MASKED_OLD
- ONE_EVERY_N

⌋

[SWS_Com_CONSTR_00764] Zero Length Signal Filter Algorithm Limitations ⌈For signals and group signals with ComBitSize configured to 0, the AUTOSAR COM module shall not support the filter algorithm MASKED_NEW_DIFFERS_MASKED_OLD.⌋

[SWS_Com_CONSTR_00535] ONE_EVERY_N Filter Configuration ⌈For the ComFilterAlgorithm ONE_EVERY_N, the ComFilterOffset shall be configured to a value lesser than ComFilterPeriod.⌋

10.3.6 Large Data Path Configuration

[SWS_Com_CONSTR_00940] Large Data Path I-PDU Restrictions ⌈If ComIPduLargeData is set to true then the following restrictions apply for the ComIPdu:

- ComIPduHandleId shall be configured
- ComIPduSignalProcessing shall be IMMEDIATE

- exactly one ComSignal shall be referred via ComIPduSignalRef

⌋

[SWS_Com_CONSTR_00941] Large Data Path I-PDU Excluded Features ⌈If ComIPduLargeData is set to true then the following parameters and containers shall not be configured for the ComIPdu:

- ComIPduCallout
- ComIPduCancellationSupport
- ComIPduTriggerTransmitCallout
- ComIPduMainFunctionRef
- ComIPduSignalGroupRef
- ComMainFunctionRouteSignalsRef
- ComMetaDataDefaultItem

⌋

[SWS_Com_CONSTR_00942] Large Data Path TX I-PDU Excluded Features ⌈If ComIPduLargeData is set to true for a ComIPdu with ComIPduDirection SEND, then the following parameters and containers shall not be configured for the ComTxIPdu:

- ComTxIPduClearUpdateBit
- ComTxIPduUnusedAreasDefault
- ComTxModeFalse
- ComTxModeTrue

⌋

[SWS_Com_CONSTR_00943] Large Data Path Signal Restrictions ⌈If ComSignal is referred by a ComIPdu with ComIPduLargeData set to true, then the following restrictions apply for the ComSignal:

- ComSignalType shall be set to UINT8_N or UINT8_DYN
- ComSignalEndianness shall be set to OPAQUE

⌋

[SWS_Com_CONSTR_00944] Large Data Path Signal Excluded Features ⌈If ComSignal is referred by a ComIPdu with ComIPduLargeData set to true, then the following parameters and containers shall not be configured for the ComSignal:

- ComBitSize

- ComDataInvalidAction
- ComHandleId
- ComInitialValueOnly
- ComRxDataTimeoutAction
- ComSignalDataInvalidValue
- ComSignalInitValue
- ComTimeoutSubstitutionValue
- ComTransferProperty
- ComUpdateBitPosition
- ComFilter

⌋

[SWS_Com_CONSTR_00945] Large Data Path Excluded from Gateway ⌈If ComSignal is referred by a ComIPdu with ComIPduLargeData set to true, then ComSignal shall not be configured as source or destination within the ComGwMapping configuration container.⌋

[SWS_Com_CONSTR_00946] Large Data Path Excluded Signal Callbacks ⌈If ComSignal is referred by a ComIPdu with ComIPduLargeData set to true, then the following ComUserCallbackType values shall not be configured for the signal:

- COM_RX_ACK
- COM_RX_INV
- COM_RX_TOUT
- COM_TX_ACK
- COM_TX_ERR
- COM_TX_TOUT

⌋

[SWS_Com_CONSTR_00947] Large Data Path TP Transmission Callbacks

Previous identifiers: SWS_LdCom_CONSTR_00009

⌈If ComIPduLargeData is set to true for a ComIPdu with ComIPduDirection SEND and ComIPduType TP, the respective <ComUser_ComCbkCopyTxData>, <ComUser_ComCbkTpTxConfirmation> notification callbacks shall be configured too.⌋

[SWS_Com_CONSTR_00948] Large Data Path TP Reception Callbacks

Previous identifiers: SWS_LdCom_CONSTR_00010

⌈If ComIPduLargeData is set to true for a ComIPdu with ComIPduDirection RECEIVE and ComIPduType TP, the respective <ComUser_ComCbkStartOfReception>, <ComUser_ComCbkCopyRxData>, <ComUser_ComCbkTpRxIndication> notification callbacks shall be configured too.⌋

[SWS_Com_CONSTR_00949] Large Data Path NORMAL Reception Callbacks

Previous identifiers: SWS_LdCom_CONSTR_00011

⌈If ComIPduLargeData is set to true for a ComIPdu with ComIPduDirection RECEIVE and ComIPduType NORMAL, the respective <ComUser_ComCbkRxIndication> notification callback shall be configured too.⌋

10.3.7 Asynchronous Copy Support

[SWS_Com_CONSTR_00960] TP I-PDUs Excluded from Asynchronous Copy ⌈A ComIPdu with ComIPduType set to TP shall reference a PDU that has KeepLocalPduBuffer set to FALSE.⌋

Asynchronous copy with hardware support (i.e. DMA transfer) is not supported for I-PDUs that use transport protocol.

10.3.8 Dynamic Length I-PDU Configuration

Since the FlexRay Interface only supports even values for the I-PDU length, it is not recommended to configure dynamic length I-PDU to be sent via FrIf. Instead, the FlexRay TP module should be used to transport dynamic length I-PDUs via FlexRay.

10.3.9 Meta Data Configuration

[SWS_Com_CONSTR_00891] Meta Data Default Item Configuration ⌈The number of ComMetaDataDefaultItem containers of a ComIPdu shall match the globally configured number of MetaDataItems of this I-PDU.⌋

10.3.10 I-PDU Group Configuration

[SWS_Com_CONSTR_00871] Direction of I-PDU Groups ⌈All I-PDUs that are contained within one I-PDU group shall have the same ComIPduDirection.⌋

10.3.11 Configuration of Com Users

To support Com's most prominent users, the RTE and the SwCluC Com Proxy but also additional users, not known yet, Com uses Uri References (see also [15] Chapter 2.3.6.6 *"Uri Reference"*) to generically link Com to its users in the model.

To ensure, that a Com user's configuration is compatible to the Com and vice versa, the required parameters and containers are defined by the ComUserUriDefSet.

For a Com user this means, that it must have a ComUserModuleCnf container (including its sub-containers) as part of its configuration.

Each Com user gets registered once with a ComUserModuleCnfRef reference being part of the Com modules configuration values.

The ComUserModuleCnf container holds the configuration of the signals and signal groups, which are produced- and consumed by the Com user as well as the signal- and signal group callbacks (compare to chapter 8.5.3) it provides.

Those signal and signal group callbacks are generic for each Com user and the signal / signal group for which the notification applies gets differentiated by the Com user callback handle Id value defined by the parameter ComUserCbkHandleId.

With this means each user can be notified on a set of signal and signal groups with user specific handle id values. Those handle id values can be directly used to index Com user specific data structures in its implementation when the notification occurs.

Please note that for the RX case the Com module can notify multiple Com users for the same signal or signal group.

The matching between ComUserSignal / ComUserSignalGroup containers and the related ComSignal / ComSignalGroup containers is done indirectly via the matching pair of:

ComUserSystemTemplateSystemSignalRef <-> ComSystemTemplateSystemSignalRef

ComUserSystemTemplateSignalGroupRef <-> ComSystemTemplateSignalGroupRef

pointing to the identical ISignalToIPduMapping.

This way of configuration avoids circular dependencies between a Com configuration and a Com user configuration. Both sides can independently derive their configurations via upstream template information.

Please note that RTE already uses a similar approach based on ComSystemTemplateSystemSignalRef / ComSystemTemplateSignalGroupRef to identify the according ComSignal / ComSignalGroup container for dataElements implementing inter ECU communication.

A Com user may span over one or multiple ECUC partitions. How this is achieved by the Com user is implementation specific. There are two different architecture patterns,

which the Com user can implement. It is Com user dependent, which one is suited better to the Com user's requirements.

Even if the concept of Com user provides a lot of flexibility to support access by multiple users including their notifications some limitations need to be considered.

In general, multiple writers can cause race conditions if the writers are not coordinated. In addition, the required sequence of `Com_SendSignal` (of group signals) and `Com_SendSignalGroup` calls cause the risk of data inconsistency in case of preemption by multiple writers

Since such a coordination is hardly to achieve in different Software Clusters, the following restriction apply:

[SWS_Com_CONSTR_00892] Signal User Configuration ⌈Sent signals or signal groups shall be owned by a single Software Cluster only and consequently either by Com users of RTE, or SwCluC, or one CDD.⌋

Nevertheless, reading a signal or signal group by several Com Users in the same or different Software Clusters is possible.

10.3.11.1 ECUC Partition Specific Com User

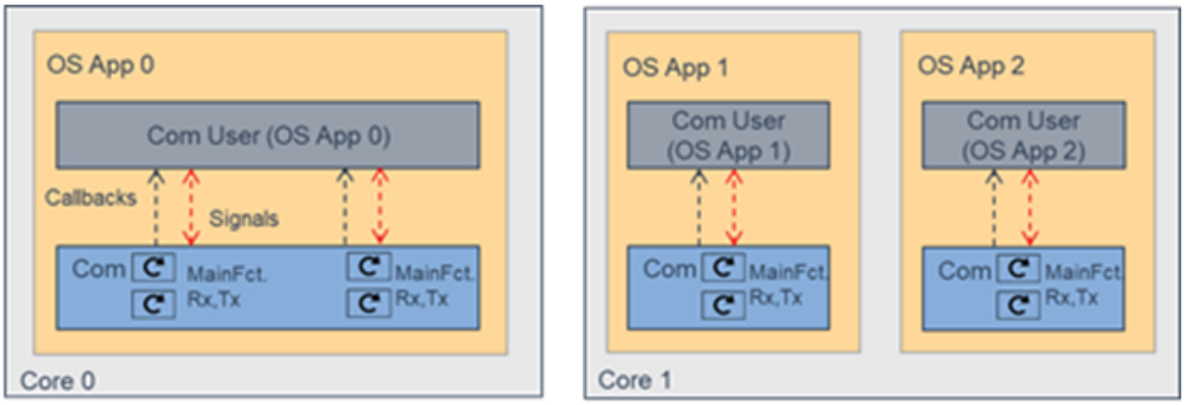


Figure 10.13: Partition specific Com users and their interaction with partition specific Com instances

In this architecture pattern, the Com user module provides dedicated instances for every configured partition, on which Com API or Callback invocation shall take place. As precondition, the Com must provide multiple main functions, that are bound to the relevant partitions. The Com user's callbacks get called in the context of only one partition. Identification of the current partition context hence can be done with a simple "callback function -> partition" lookup table.

10.3.11.2 ECUC Partition Unspecific Com User

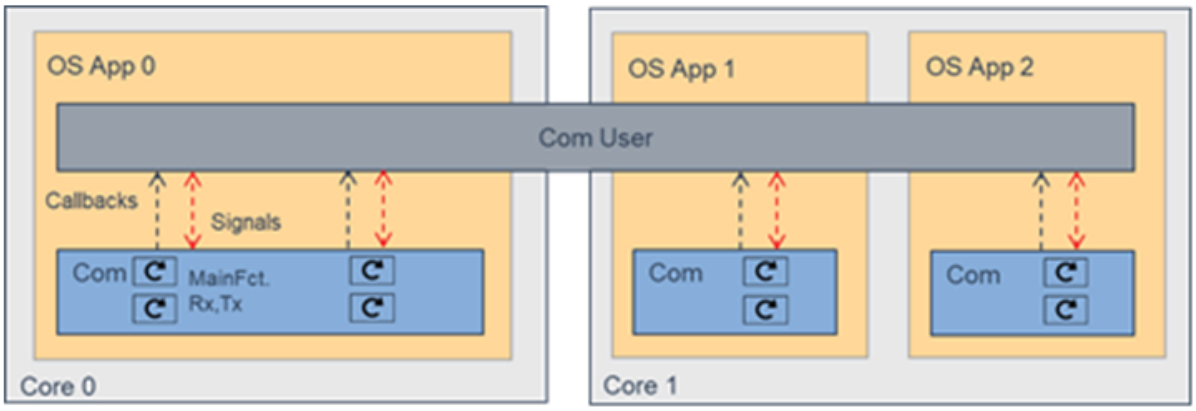


Figure 10.14: A partition unspecific Com user and its interaction with partition specific Com instances

In this architecture pattern, the com user is partition unspecific and needs to provide a reentrant implementation of its callbacks for different signals and signal groups on different ECUC partitions. The Com user needs to provide one common set of callbacks, that gets called in the context of multiple partitions.

10.3.11.3 Relation to SwCluC Com Proxy

A Com user has one or several SwCluCComBaseSockets [7]. A Base Socket is required for each partition, in which the Com user either requires access to the Com APIs for `ComSignal` / `ComSignalGroup` transmission / reception or provides signal- and signal group callbacks.

A Base Socket binds the Com callbacks- and a set of Com signals / Com signal groups of the Com in the Host Software Cluster to a EcucPartition. This means, that the Com in the Host Software Cluster invokes the callbacks in the context of this EcucPartition and provides the Com signals / Com signal groups only in this EcucPartition.

This requires the following preconditions to be fulfilled:

- The Com IPdus that contain the Com signals / Com signal groups of a Base Socket shall be assigned to a Com main function on the same EcucPartition as configured for the Base Socket. Details on the EcucPartition mapping in Com is described in `ComIPduMainFunctionRef`, Chapter 7.14
- The system shall provide the required ECUC partitions in the Application and the Host Software Cluster. This is a requirement which must be considered during system design
- The Com user shall be either of the following variants:
  - The Com user is partition unspecific and provides a common instance, which APIs can be called on multiple partitions

- The Com user is partition specific and provides dedicated instances for the configured EcuCPartitions

10.4 Published Information

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

A Not applicable requirements

[SWS_Com_NA_00999]

Upstream requirements: SRS_BSW_00004, SRS_BSW_00168, SRS_BSW_00170, SRS_BSW_00171, SRS_BSW_00375, SRS_BSW_00383, SRS_BSW_00386, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00395, SRS_BSW_00403, SRS_BSW_00409, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00419, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00424, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00433, SRS_BSW_00437, SRS_BSW_00461, SRS_BSW_00469, SRS_BSW_00471, SRS_BSW_00472, SRS_BSW_00478, SRS_BSW_00490

⌈These requirements are not applicable to this specification.⌋

B Appendix

This appendix contains use cases with different transmission modes and the necessary configuration for these. Table B.1 and Figures B.1, B.2, B.3 show the notation of the use case diagrams.

Notation	Description
t_c, t_{c1}, t_{c2}	cycle times
t_d	cycle time of N-Times sent signals
t_r	minimum SW reaction time of COM-Layer due to internal main cycle time
V	Value: x stands for an arbitrary value/ value range, a...w for specific values / value ranges, defined by the user, with a <> b, range a is disjoint from range b.
dt	minimum delay time: minimum distance between two requests to PDU Router
without TMS switch	without switching of the TMS (see Chapter 7.2.3.2) from true to false or vice versa
with TMS switch	with switching of the TMS (see Chapter 7.2.3.2) from true to false or vice versa (from TM 1 to TM 2); one TM is named before the "+" and one behind in the description

Table B.1: Legend for use case diagrams.



Figure B.1: Request from RTE to the COM-Layer



Figure B.2: Request from COM-Layer to PDU Router



Figure B.3: Potential but skipped request from COM-Layer to PDU Router (e.g. because of a new send request by the RTE or delayed due to minimum delay time)

B.1 Use Case 1

Use case 1 shows an I-PDU that is sent out cyclically with a cycle time t_c. This I-PDU consists of signals that all have the `ComTransferProperty PENDING`. It is configured that the transmission takes place when the TMS evaluates to true.

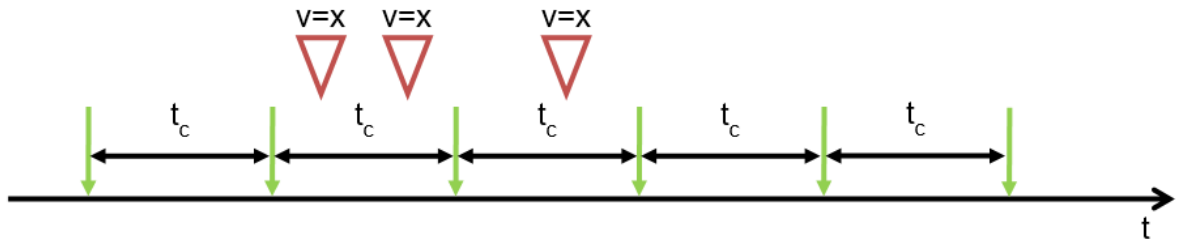


Figure B.4: Use case 1, TM `PERIODIC` (without TMS switch)

`ComTransferProperty`	`PENDING` or `TRIGGERED` (`TRIGGERED` has no influence)
`ComFilterAlgorithm`	`ALWAYS`
`ComIPduDirection`	`SEND`
`ComTxModeTrue` / `ComTxModeMode`	`PERIODIC`
`ComTxModeTrue` / `ComTxModeTimePeriod`	t_c
`ComMinimumDelayTime`	0 or omitted

Table B.2: Relevant I-PDU transmission configuration for use case 1

Because of the configuration of the parameter `ComFilterAlgorithm` `ALWAYS` of the `ComFilter`, there is no need to configure a transmission mode for the case that the TMS evaluates to false.

It does not make any difference in the behavior whether the `ComFilterAlgorithm` parameter of the `ComFilter` is set for all the signals within the I-PDU to `ALWAYS` or if the `ComFilter` is not defined (does not contribute to the evaluation of the TMS).

B.2 Use Case 2

Use case 2 shows an I-PDU which is sent out three times whenever a value is given by the upper (`Com_SendSignal` or `Com_SendSignalGroup`). The time between two transmissions is t_d. This I-PDU consists of signals, which all have the `ComTransfer-Property`. It is configured that the transmission takes place when the TMS evaluates to true.

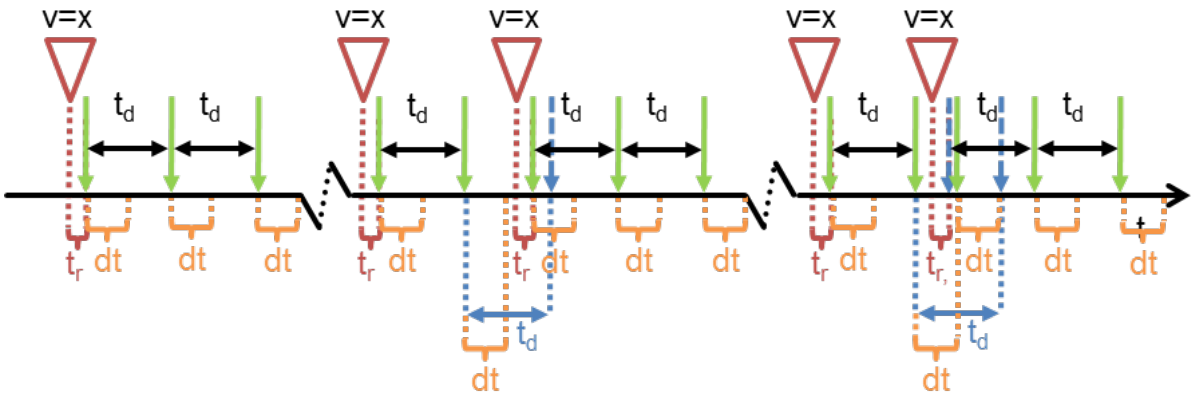


Figure B.5: Use case 2, TM `DIRECT` (N-Times) here n = 2 (without TMS switch)

ComEnableMDTForCyclicTransmission	true
ComTransferProperty	TRIGGERED
ComFilterAlgorithm	ALWAYS
ComIPduDirection	SEND
ComTxModeTrue / ComTxModeMode	DIRECT
ComTxModeTrue / ComTxModeNumberOfRepetitions	2
ComTxModeTrue / ComTxModeRepetitionPeriod	t_d
ComMinimumDelayTime	dt

Table B.3: Relevant I-PDU transmission configuration for use case 2

If there is a new send request by the RTE before the ongoing repetition cycle is completed, the new repetition cycle is started and the rest of the last one is discarded.

B.3 Use Case 3

Use case 3 shows an I-PDU which is sent out cyclically with a cycle time t_{c1} if value v = a (TMS evaluates to true) and with a cycle time t_{c2} if value v = b (TMS evaluates to false). The I-PDU consists of signals, which all have the `ComTransferProperty PENDING`.

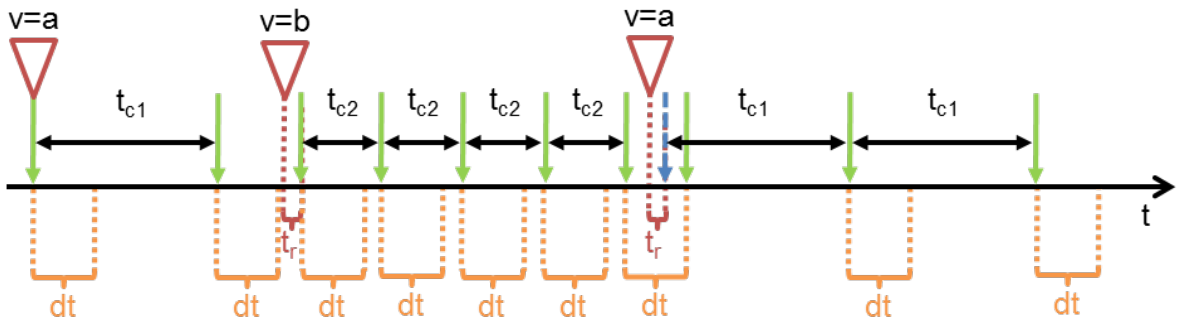


Figure B.6: Use case 3, TM `PERIODIC` + `PERIODIC` (with TMS switch)

ComEnableMDTForCyclicTransmission	true
ComTransferProperty	PENDING or TRIGGERED (TRIGGERED has no influence)
ComFilterAlgorithm	any filter mapping value a to true and value b to false
ComIPduDirection	SEND
ComTxModeTrue / ComTxModeMode	PERIODIC
ComTxModeTrue / ComTxModeTimePeriod	t_{c1}
ComTxModeFalse / ComTxModeMode	PERIODIC
ComTxModeFalse / ComTxModeTimePeriod	t_{c2}
ComMinimumDelayTime	dt

Table B.4: Relevant I-PDU transmission configuration for use case 3

The TMS switch caused by the new value v = b, starts a new cycle within the next main function and the new value is sent out. Nevertheless, the minimum delay time dt has to be taken into account, since ComEnableMDTForCyclicTransmission is set to true.

For the parameter ComFilterAlgorithm, the following filters defined in [2] can be used for this use-case:

- MASKED_NEW_EQUALS_X
- MASKED_NEW_DIFFERS_X
- MASKED_NEW_DIFFERS_MASKED_OLD
- NEW_IS_WITHIN
- NEW_IS_OUTSIDE

B.4 Use Case 4

Use case 4 shows an I-PDU which is sent out cyclically with a cycle time t_c if value v = a (TMS evaluates to true) and if value v = b (TMS evaluates to false) it is sent out three times whenever the value is given by the RTE. The time between two transmissions is t_d. The I-PDU consists of signals that all have the ComTransferProperty TRIGGERED.

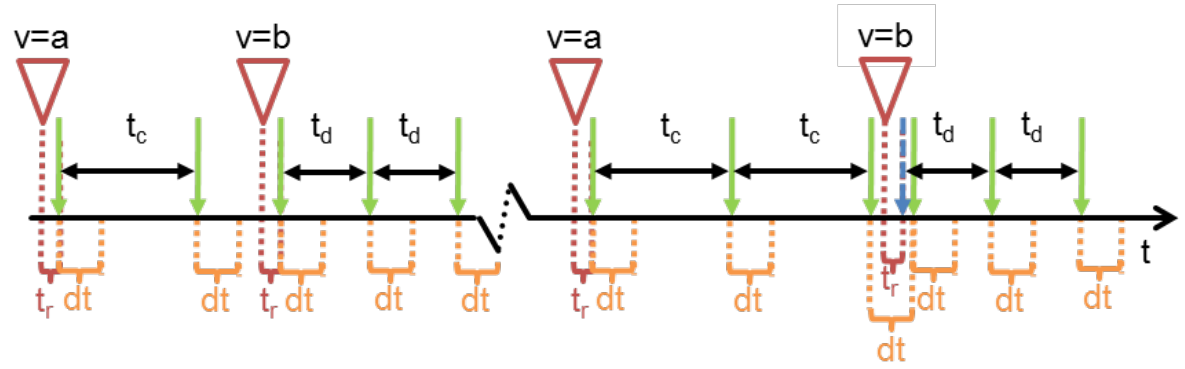


Figure B.7: Use case 4 scenario a, TM `PERIODIC` + `DIRECT`/N-Times, here n = 2 (with TMS switch)

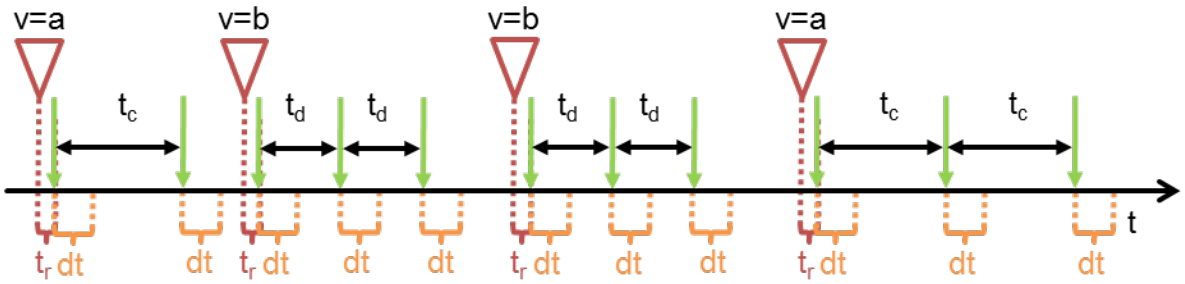


Figure B.8: Use case 4 scenario b, TM `PERIODIC` + `DIRECT`/N-Times, here n = 2 (with TMS switch)

`ComEnableMDTForCyclicTransmission`	true
`ComTransferProperty`	`TRIGGERED`
`ComFilterAlgorithm`	any filter mapping value a to true and value b to false
`ComIPduDirection`	`SEND`
`ComTxModeTrue` / `ComTxModeMode`	`PERIODIC`
`ComTxModeTrue` / `ComTxModeTimePeriod`	t_c
`ComTxModeFalse` / `ComTxModeMode`	`DIRECT`
`ComTxModeFalse` / `ComTxModeNumberOfRepetitions`	2
`ComTxModeFalse` / `ComTxModeTimePeriod`	t_d
`ComMinimumDelayTime`	dt

Table B.5: Relevant I-PDU transmission configuration for use case 4

After the switch from `ComTxModeMode` `DIRECT` to `PERIODIC` the cycle is started within the next main function and the new value a is sent out with respect to `ComMinimumDelayTime` dt.

B.5 Use Case 5

Use case 5 shows an I-PDU which is sent out cyclically with a cycle time t_c and if the value (the same or a new one) is given by the RTE it is also sent out directly three

times. The time between two of these three transmissions is always t_d. The I-PDU consists of signals that all have the ComTransferProperty TRIGGERED.

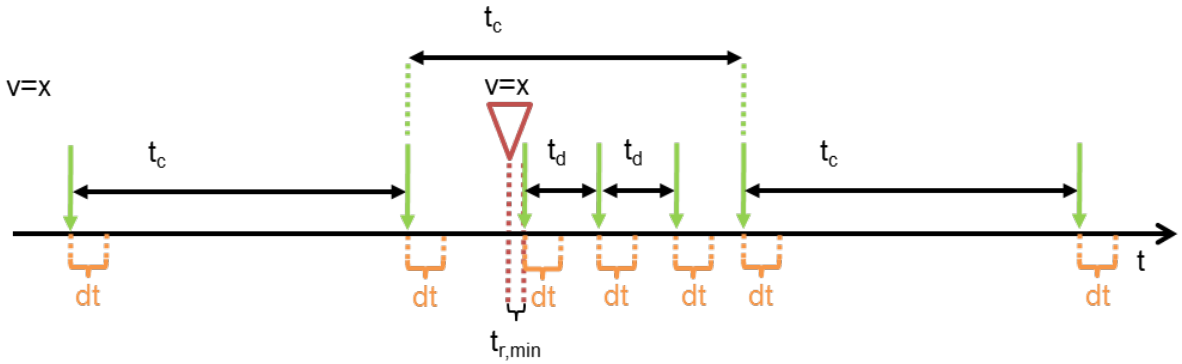


Figure B.9: Use case 5 scenario a, TM MIXED, here n = 2 (without TMS switch)

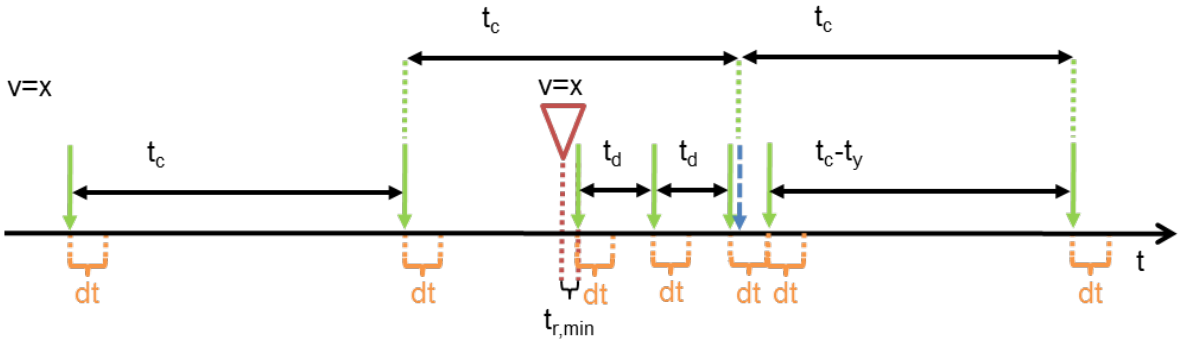


Figure B.10: Use case 5 scenario b, TM MIXED, here n = 2 (without TMS switch)

ComEnableMDTForCyclicTransmission	true
ComTransferProperty	TRIGGERED
ComFilterAlgorithm	ALWAYS
ComIPduDirection	SEND
ComTxModeTrue / ComTxModeMode	MIXED
ComTxModeTrue / ComTxModeTimePeriod	t_c
ComTxModeTrue / ComTxModeNumberOfRepetitions	2
ComTxModeTrue / ComTxModeRepetitionPeriod	t_d
ComMinimumDelayTime	dt

Table B.6: Relevant I-PDU transmission configuration for use case 5

If the next transmission caused by the periodic part of the ComTxModeMode MIXED should take place within the timeout dt (ComMinimumDelayTime) after a transmission of the DIRECT (N-Times) part, this sent out is delayed until the minimum delay time is elapsed. However, after that the next period of the periodic part is shortened so that there is only an intermediate phase shift of the periodic part but no continuous one.

B.6 Use Case 6

Use case 6 shows an I-PDU which is sent out cyclically with a cycle time t_{c2} if value v = b (TMS evaluates to false). If value v = a (TMS evaluates to true) it is sent out cyclically with a cycle time t_{c1} and whenever the value v = a is given by the RTE it is also sent out directly three times. The time between two of these three transmissions is always t_d. The I-PDU consists of signals that all have the `ComTransferProperty` `TRIGGERED`.

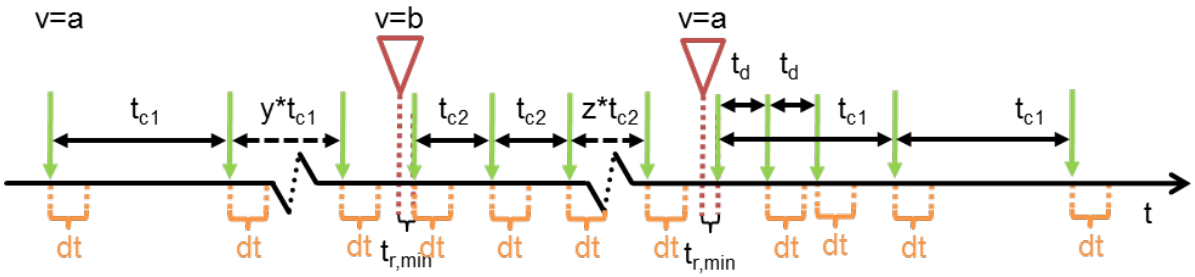


Figure B.11: Use Case 6, TM `MIXED`, here n= 2 + `PERIODIC` (with TMS switch)

`ComEnableMDTForCyclicTransmission`	true
`ComTransferProperty`	`TRIGGERED`
`ComFilterAlgorithm`	any filter mapping value a to true and value b to false
`ComIPduDirection`	`SEND`
`ComTxModeTrue` / `ComTxModeMode`	`MIXED`
`ComTxModeTrue` / `ComTxModeTimePeriod`	t_{c1}
`ComTxModeTrue` / `ComTxModeNumberOfRepetitions`	2
`ComTxModeTrue` / `ComTxModeRepetitionPeriod`	t_d
`ComTxModeFalse` / `ComTxModeMode`	`PERIODIC`
`ComTxModeTrue` / `ComTxModeTimePeriod`	t_{c2}
`ComMinimumDelayTime`	dt

Table B.7: Relevant I-PDU transmission configuration for use case 6

A usage of this in practice is for example the signal of the button that controls the window-lift motor. If the button is not pressed, there is a long cycle time t_{c1} with this information. If it is pressed this information is distributed with a short cycle time t_{c2}. If the button is released again, starting with the next main function this information is distributed three times with t_d and after that, again the long cycle time is used.

B.7 Use Case 7

Use case 7 is similar to use case 5 but with `ComEnableMDTForCyclic-Transmission` disabled and `ComTxModeNumberOfRepetitions` set to one.

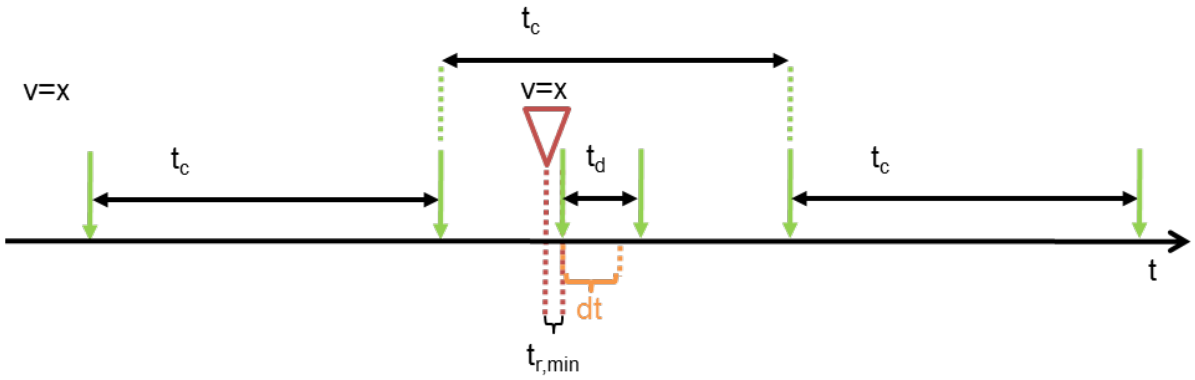


Figure B.12: Use case 7 scenario a, TM MIXED, here n = 1 (without TMS switch)

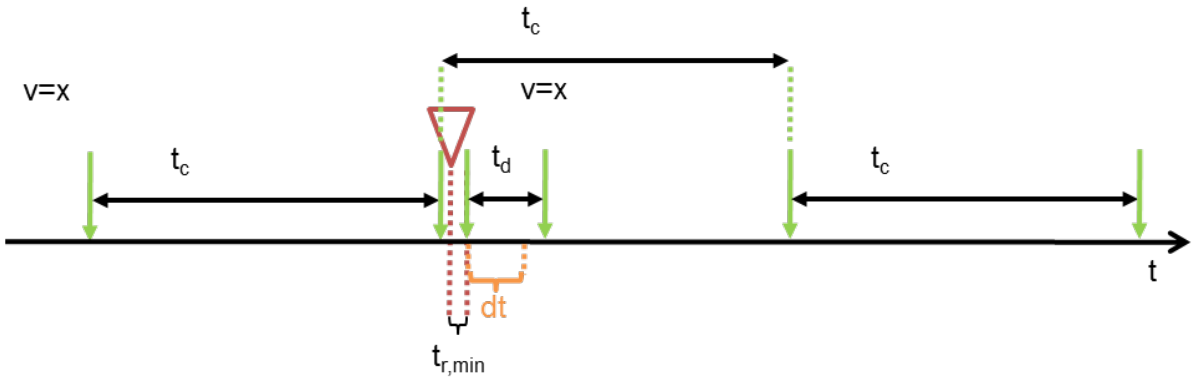


Figure B.13: Use case 7 scenario b, TM MIXED, here n = 1 (without TMS switch)

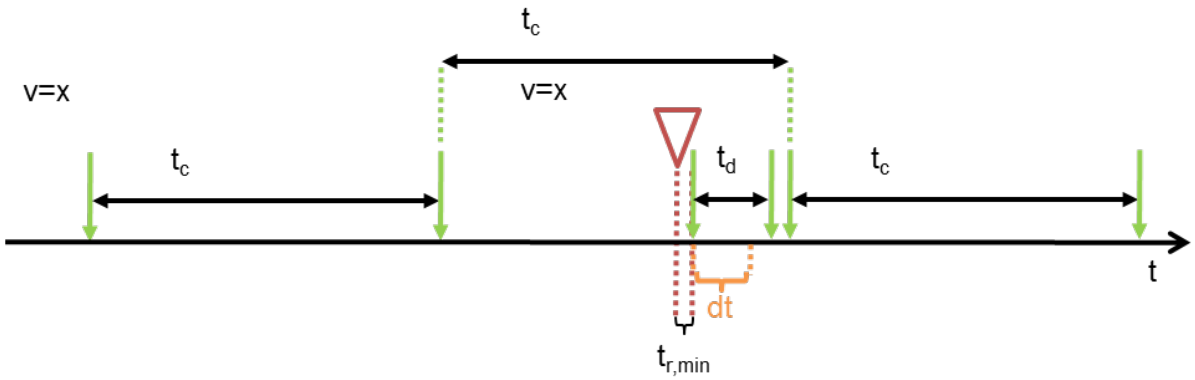


Figure B.14: Use case 7 scenario c, TM MIXED, here n = 1 (without TMS switch)

ComEnableMDTForCyclicTransmission	false
ComTransferProperty	TRIGGERED
ComFilterAlgorithm	ALWAYS
ComIPduDirection	SEND
ComTxModeTrue / ComTxModeMode	MIXED

▽

△

ComTxModeTrue / ComTxModeTimePeriod	t_c
ComTxModeTrue / ComTxModeNumberOfRepetitions	1
ComTxModeTrue / ComTxModeRepetitionPeriod	t_d
ComMinimumDelayTime	dt

Table B.8: Relevant I-PDU transmission configuration for use case 7

In contrary to use case 5, here the minimum delay timer is neither started for the periodic transmissions of the cyclic part nor for the repeated transmissions of the direct part of the MIXED mode. This can result in two consecutive transmissions within the timespan dt as shown in the scenarios b and c.

C 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.

C.1 Traceable item history of this document according to AUTOSAR Release R24-11

C.1.1 Added Specification Items in R24-11

Number	Heading
[ECUC_Com_10041]	Definition of EcucParamConfContainerDef ComSecurityEventRefs
[ECUC_Com_10042]	Definition of EcucBooleanParamDef ComEnableSecurityEventReporting
[ECUC_Com_10043]	Definition of EcucReferenceDef SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED
[SWS_Com_00900]	Security events for Com
[SWS_Com_00901]	Security event reporting
[SWS_Com_00902]	SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED
[SWS_Com_00903]	Security event context data definition: SEV_COM_RX_SIGNAL_VALUE_UNEXPECTED

Table C.1: Added Specification Items in R24-11

C.1.2 Changed Specification Items in R24-11

Number	Heading
[ECUC_Com_00541]	Definition of EcucParamConfContainerDef ComGeneral
[SWS_Com_00304]	
[SWS_Com_00495]	
[SWS_Com_00609]	Definition of imported datatypes of module Com
[SWS_Com_00627]	Definition of API function Com_SendDynSignal
[SWS_Com_00670]	Definition of optional interfaces requested by module Com
[SWS_Com_00690]	Definition of API function Com_ReceiveDynSignal
[SWS_Com_91006]	Definition of API function Com_ReceiveDynSignalWithMetaData
[SWS_Com_91010]	Definition of API function Com_SendDynSignalWithMetaData

Table C.2: Changed Specification Items in R24-11

C.1.3 Deleted Specification Items in R24-11

none

C.2 Traceable item history of this document according to AUTOSAR Release R25-11

C.2.1 Added Specification Items in R25-11

Number	Heading
[ECUC_Com_10045]	Definition of EcucBooleanParamDef ComIPduLargeData
[ECUC_Com_10046]	Definition of EcucParamConfContainerDef ComDemEventParameterRefs
[ECUC_Com_10047]	Definition of EcucReferenceDef COM_E_TIMEOUT
[ECUC_Com_10048]	Definition of EcucParamConfContainerDef ComUserIPdu
[ECUC_Com_10049]	Definition of EcucBooleanParamDef ComLargeDataSupport
[SWS_Com_00893]	Reception Side Meta Data Initialization
[SWS_Com_00906]	Large Data Path Minimum Delay Timer
[SWS_Com_00907]	Reception Deadline Monitoring PREFAILED Reporting to Dem
[SWS_Com_00908]	Reception Deadline Monitoring PREPASSED Reporting to Dem
[SWS_Com_00909]	Transmission Deadline Monitoring PREPASSED Reporting to Dem
[SWS_Com_00913]	Large Data Path Excluded API Functions
[SWS_Com_00914]	Large Data Path IF Transmission
[SWS_Com_00915]	Large Data Path IF TriggerTransmit
[SWS_Com_00916]	Large Data Path IF TxConfirmation
[SWS_Com_00917]	Large Data Path TP Transmission
[SWS_Com_00918]	Large Data Path IF Reception
[SWS_Com_00919]	Large Data Path TP StartOfReception
[SWS_Com_00920]	Large Data Path TP CopyRxData
[SWS_Com_00921]	Large Data Path TP TpRxIndication
[SWS_Com_00930]	Com_TriggerIPDUSend and Com_TriggerIPDUSendWithMetaData Excluded for Large Data Path
[SWS_Com_00931]	Definition of API function Com_Transmit
[SWS_Com_00932]	Com_Transmit Available for Large Data Path I-PDUs Only
[SWS_Com_00933]	Com_Transmit Behavior for Stopped I-PDUs
[SWS_Com_00934]	Com_RxIndication Behavior for Stopped I-PDUs
[SWS_Com_00935]	Com_TpRxIndication Behavior for Stopped I-PDUs
[SWS_Com_00936]	Com_TxConfirmation Behavior for Stopped I-PDUs
[SWS_Com_00937]	Com_TpTxConfirmation Behavior for Stopped I-PDUs
[SWS_Com_91024]	Definition of configurable interface <ComUser_ComCbkCopyTxData>

▽

△

Number	Heading
[SWS_Com_91025]	Definition of configurable interface <ComUser_ComCbkTpTxConfirmation>
[SWS_Com_91026]	Definition of configurable interface <ComUser_ComCbkStartOfReception>
[SWS_Com_91027]	Definition of configurable interface <ComUser_ComCbkCopyRxData>
[SWS_Com_91028]	Definition of configurable interface <ComUser_ComCbkTpRxIndication>
[SWS_Com_91029]	Definition of configurable interface <ComUser_ComCbkTriggerTransmit>
[SWS_Com_91030]	Definition of configurable interface <ComUser_ComCbkTxConfirmation>
[SWS_Com_91031]	Definition of configurable interface <ComUser_ComCbkRxIndication>

Table C.3: Added Specification Items in R25-11

C.2.2 Changed Specification Items in R25-11

Number	Heading
[ECUC_Com_00017]	Definition of EcucIntegerParamDef ComTxIPduUnusedAreasDefault
[ECUC_Com_00175]	Definition of EcucIntegerParamDef ComIPduHandleId
[ECUC_Com_00340]	Definition of EcucParamConfContainerDef ComIPdu
[ECUC_Com_00344]	Definition of EcucParamConfContainerDef ComSignal
[ECUC_Com_00345]	Definition of EcucParamConfContainerDef ComSignalGroup
[ECUC_Com_00496]	Definition of EcucParamConfContainerDef ComTxIPdu
[ECUC_Com_00541]	Definition of EcucParamConfContainerDef ComGeneral
[ECUC_Com_10022]	Definition of EcucParamConfContainerDef ComMetaDataDefaultItem
[ECUC_Com_10025]	Definition of EcucForeignReferenceDef ComUserSystemTemplateSystem SignalRef
[ECUC_Com_10030]	Definition of EcucParamConfContainerDef ComUserModuleCnf
[ECUC_Com_10033]	Definition of EcucIntegerParamDef ComUserCbkHandleId
[ECUC_Com_10034]	Definition of EcucEnumerationParamDef ComUserCallbackType
[ECUC_Com_10035]	Definition of EcucReferenceDef ComUserCallbackRef
[ECUC_Com_10038]	Definition of EcucParamConfContainerDef ComUserCallback
[ECUC_Com_10040]	Definition of EcucDestinationUriDef ComUser
[SWS_Com_00304]	
[SWS_Com_00396]	
[SWS_Com_00609]	Definition of imported datatypes of module Com
[SWS_Com_00670]	Definition of optional interfaces requested by module Com
[SWS_Com_00698]	
[SWS_Com_00744]	
[SWS_Com_00773]	
[SWS_Com_00828]	

▽

△

Number	Heading
[SWS_Com_00861]	
[SWS_Com_00862]	
[SWS_Com_00880]	
[SWS_Com_00888]	Reception Side Meta Data Handling
[SWS_Com_00901]	Security event reporting
[SWS_Com_91005]	Definition of API function Com_ReceiveSignalWithMetaData
[SWS_Com_91006]	Definition of API function Com_ReceiveDynSignalWithMetaData
[SWS_Com_91008]	Definition of API function Com_ReceiveSignalGroupArrayWithMetaData
[SWS_Com_91014]	Definition of development errors in module Com

Table C.4: Changed Specification Items in R25-11

C.2.3 Deleted Specification Items in R25-11

Number	Heading
[SWS_Com_00102]	
[SWS_Com_00105]	
[SWS_Com_00310]	
[SWS_Com_00365]	
[SWS_Com_00373]	
[SWS_Com_00380]	
[SWS_Com_00384]	
[SWS_Com_00401]	
[SWS_Com_00402]	
[SWS_Com_00439]	
[SWS_Com_00443]	
[SWS_Com_00465]	
[SWS_Com_00497]	
[SWS_Com_00535]	
[SWS_Com_00553]	
[SWS_Com_00598]	
[SWS_Com_00637]	
[SWS_Com_00639]	
[SWS_Com_00732]	
[SWS_Com_00754]	
[SWS_Com_00755]	
[SWS_Com_00756]	

▽

△

Number	Heading
[SWS_Com_00762]	
[SWS_Com_00764]	
[SWS_Com_00785]	
[SWS_Com_00790]	
[SWS_Com_00859]	
[SWS_Com_00860]	
[SWS_Com_00871]	
[SWS_Com_00891]	
[SWS_Com_00892]	DRAFT

Table C.5: Deleted Specification Items in R25-11

C.2.4 Added Constraints in R25-11

Number	Heading
[SWS_Com_CONSTR_00102]	Signals Separated in I-PDU
[SWS_Com_CONSTR_00105]	Byte Aligned Array Signals
[SWS_Com_CONSTR_00310]	Update Bit Excluded in case of Transmission Repetitions
[SWS_Com_CONSTR_00365]	Signal Group Gateway Configuration
[SWS_Com_CONSTR_00373]	Post Build Configuration Update
[SWS_Com_CONSTR_00380]	Float and Array Signal Filter Algorithm Limitations
[SWS_Com_CONSTR_00384]	Routed Signal Size Configuration
[SWS_Com_CONSTR_00401]	Unique COM Model Element Names
[SWS_Com_CONSTR_00402]	Well-formedness of COM Model Element Names

▽

△

Number	Heading
[SWS_Com_CONSTR_00439]	Boolean Signal Filter Algorithm Limitations
[SWS_Com_CONSTR_00443]	Mapping of Array Signals
[SWS_Com_CONSTR_00465]	Transmission Mode Configuration
[SWS_Com_CONSTR_00497]	Mandatory ComTxIPdu for TX I-PDUs
[SWS_Com_CONSTR_00535]	ONE_EVERY_N Filter Configuration
[SWS_Com_CONSTR_00553]	OPAQUE Endianness of Array Signals
[SWS_Com_CONSTR_00598]	TP I-PDUs Excluded from Gateway
[SWS_Com_CONSTR_00732]	Callback Names Differ from COM Model Element Names
[SWS_Com_CONSTR_00754]	Dynamic Length Signal Configuration
[SWS_Com_CONSTR_00755]	Placement of Dynamic Length Signals
[SWS_Com_CONSTR_00756]	Size Configuration of Dynamic Length Signals
[SWS_Com_CONSTR_00762]	Transfer Property of Zero Length Signals
[SWS_Com_CONSTR_00764]	Zero Length Signal Filter Algorithm Limitations
[SWS_Com_CONSTR_00785]	Bit Size of Signals
[SWS_Com_CONSTR_00790]	Interpretation of Signal Initial and Invalid Values

▽

△

Number	Heading
[SWS_Com_CONSTR_00859]	Signal Invalid Value Configuration
[SWS_Com_CONSTR_00860]	Group Signal Invalid Value Configuration
[SWS_Com_CONSTR_00871]	Direction of I-PDU Groups
[SWS_Com_CONSTR_00891]	Meta Data Default Item Configuration
[SWS_Com_CONSTR_00892]	Signal User Configuration
[SWS_Com_CONSTR_00911]	Large Data Path I-PDU User Configuration
[SWS_Com_CONSTR_00912]	Large Data Path I-PDU User EcucPartition
[SWS_Com_CONSTR_00940]	Large Data Path I-PDU Restrictions
[SWS_Com_CONSTR_00941]	Large Data Path I-PDU Excluded Features
[SWS_Com_CONSTR_00942]	Large Data Path TX I-PDU Excluded Features
[SWS_Com_CONSTR_00943]	Large Data Path Signal Restrictions
[SWS_Com_CONSTR_00944]	Large Data Path Signal Excluded Features
[SWS_Com_CONSTR_00945]	Large Data Path Excluded from Gateway
[SWS_Com_CONSTR_00946]	Large Data Path Excluded Signal Callbacks
[SWS_Com_CONSTR_00947]	Large Data Path TP Transmission Callbacks

▽

△

Number	Heading
[SWS_Com_CONSTR_00948]	Large Data Path TP Reception Callbacks
[SWS_Com_CONSTR_00949]	Large Data Path NORMAL Reception Callbacks

Table C.6: Added Constraints in R25-11

C.2.5 Changed Constraints in R25-11

none

C.2.6 Deleted Constraints in R25-11

none

C.3 Traceable item history of this document according to AUTOSAR Release R26-11

C.3.1 Added Specification Items in R26-11

Number	Heading
[SWS_Com_00950]	Large Data Path IF Reception with DMA Copy Approach
[SWS_Com_00951]	Transmit Buffer Alignment for Asynchronous Copy
[SWS_Com_00952]	Release Transmit Buffer
[SWS_Com_00953]	Error Reporting when I-PDU Stopped
[SWS_Com_00954]	Asynchronous Copy to Receive Buffer
[SWS_Com_00955]	Release Receive Buffer
[SWS_Com_00956]	Error Reporting when Asynchronous Copy Fails
[SWS_Com_91033]	Definition of API function Com_ReleaseRxBuffer

Table C.7: Added Specification Items in R26-11

C.3.2 Changed Specification Items in R26-11

Number	Heading
[ECUC_Com_00520]	Definition of EcucParamConfContainerDef ComGroupSignal
[ECUC_Com_00560]	Definition of EcucEnumerationParamDef ComGroupSignalTransferProperty
[ECUC_Com_10040]	Definition of EcucDestinationUriDef ComUser
[ECUC_Com_10047]	Definition of EcucReferenceDef COM_E_TIMEOUT
[SWS_Com_00124]	Definition of callback function Com_TxConfirmation
[SWS_Com_00279]	
[SWS_Com_00479]	
[SWS_Com_00609]	Definition of imported datatypes of module Com
[SWS_Com_00650]	Definition of callback function Com_TpRxIndication
[SWS_Com_00670]	Definition of optional interfaces requested by module Com
[SWS_Com_00725]	Definition of callback function Com_TpTxConfirmation
[SWS_Com_00742]	
[SWS_Com_00743]	
[SWS_Com_00777]	
[SWS_Com_91014]	Definition of development errors in module Com
[SWS_Com_91015]	Definition of runtime errors in module Com

Table C.8: Changed Specification Items in R26-11

C.3.3 Deleted Specification Items in R26-11

Number	Heading
[ECUC_Com_00185]	Definition of EcucReferenceDef ComIPduGroupGroupRef
[ECUC_Com_00438]	Definition of EcucBooleanParamDef ComVersionInfoApi
[SWS_Com_00006]	
[SWS_Com_00426]	Definition of API function Com_GetVersionInfo

Table C.9: Deleted Specification Items in R26-11

C.3.4 Added Constraints in R26-11

Number	Heading
[SWS_Com_CONSTR_00960]	TP I-PDUs Excluded from Asynchronous Copy

Table C.10: Added Constraints in R26-11

C.3.5 Changed Constraints in R26-11

none

C.3.6 Deleted Constraints in R26-11

Number	Heading
[SWS_Com_CONSTR_00373]	Post Build Configuration Update
[SWS_Com_CONSTR_00401]	Unique COM Model Element Names
[SWS_Com_CONSTR_00402]	Well-formedness of COM Model Element Names
[SWS_Com_CONSTR_00598]	TP I-PDUs Excluded from Gateway
[SWS_Com_CONSTR_00732]	Callback Names Differ from COM Model Element Names

Table C.11: Deleted Constraints in R26-11