

# MICROSAR COM

## Technical Reference

Version 2.11.01

Authors	Gunnar Meiss, Hartmut Hörner, Klaus Emmert, Hannes Haas, Michael Bissinger, Dominik Biber
Status	Released

# 1 Document Information

## 1.1 History

Author	Date	Version	Remarks
Gunnar Meiss	2006-07-31	0.1	Creation
Hartmut Hörner	2006-09-13	0.2	Update to new template version
Klaus Emmert	2006-09-25	0.3	New Illustrations
Hartmut Hörner	2006-10-13	0.4	Post build process figure updated, several minor changes based on review comments
Hartmut Hörner	2006-10-24	0.5	Added configuration chapter and list of DET error codes.
Hartmut Hörner	2006-11-06	1.0	Minor changes based on review comments.
Gunnar Meiss	2007-01-08	1.1	ESCAN00018727, ESCAN00018724, ESCAN00018738, ESCAN00018721, ESCAN00018723
Gunnar Meiss	2007-03-13	1.2	ESCAN00019913
Hannes Haas	2007-04-26	1.2	ESCAN00019913: Added Signal Gateway Update to new template General rework
Gunnar Meiss	2007-08-01	1.2	ESCAN00021344: Update EcuC Description Removed error message list Removed COM_IPDU_DIRECTION , COM_NETWORK_SIGNAL_DIRECTION Updated EcuC attribute names Updated EcuC Import/Export Updated GenSigSendType
Michael Bissinger	2007-10-05	1.3	Added signal conversion description ESCAN00022013, ESCAN00022628, ESCAN00022050
Michael Bissinger	2007-11-23	1.4	Added I-PDU callouts ESCAN00023159 Added indication and Rx timeout flags ESCAN00023392
Hannes Haas	2008-01-10	1.5	Added support for TP I-PDU communication

Michael Bissinger	2008-02-13	1.6	Added Transmission Mode Selector chapter 3.7.2 Added Transmit Signal Filters chapter 3.7.3 Added new send type mappings to chapter 6.1 ESCAN00024194
Gunnar Meiss	2008-03-18	2.0	AUTOSAR 3 Updated API Descriptions Restructured Documentation
Gunnar Meiss	2008-07-07	2.1	Added Tms320 Limitation Updated API Description Updated EcuC Description Upgraded Technical Reference Template Updated Callout Description
Gunnar Meiss	2008-07-21	2.2	Rework with CIWI Updated Chapter 6.5.2.1 Automatic routing relations Updated 7.2.2 Code Generator and Configuration Updated Chapter 6.5 Configuration with GENy Updated Chapter 6.4 Configuration in EcuC Data Base Updated Chapter 5.2 Services provided by COM
Gunnar Meiss	2008-11-28	2.3	Added Signal Invalidation API Updated Chapter 6.5 Configuration with GENy Updated Chapter 6.4 Configuration in EcuC Data Base Updated Chapter 5.2 Services provided by COM
Michael Bissinger	2008-12-12		Added 3.8.3 Reception of Invalid Signal Values
Gunnar Meiss	2008-12-22	2.4	Added First Timeout Time
Gunnar Meiss	2009-03-04	2.5	Added 4.3 Compiler Abstraction and Memory Mapping (ESCAN00030888) ESCAN00032806 ESCAN00033565
Dominik Biber	2009-11-13	2.6	ESCAN00038580 ESCAN00038562
Dominik Biber	2009-11-20	2.6	ESCAN00039294 Updated chapter 6.5.2.4
Dominik Biber	2009-11-30	2.6	Updated Chapter 1.2 Reference Documents Updated Chapter 2 Introduction
Dominik Biber	2010-03-04	2.6.1	Signalgroup support in LDF files
Dominik Biber	2010-03-04	2.6.1	Support Optional Invalidation (F334)

Dominik Biber	2010-04-19	2.7.0	<p>ESCAN00039525 Add 'Service Ids' and 'Errors reported to DET' table</p> <p>ESCAN00042150 Support StateOn flag provider</p> <p>ESCAN00042149 Pre-compile optimization for flag provider macro access</p> <p>ESCAN00042386 Support preconfiguration of all parameters</p>
Dominik Biber	2010-05-03	2.8.0	<p>ESCAN00040927</p> <p>ESCAN00041028</p> <p>ESCAN00042542</p> <p>ESCAN00042075</p> <p>ESCAN00042007</p> <p>ESCAN00039799</p>
Dominik Biber	2010-08-06	2.9.0	<p>Added 3.7.5 Transmission Deadline Monitoring (ESCAN00043067, ESCAN00043739)</p> <p>Adapted 3.7.2 Transmission Mode Selector (ESCAN00043947)</p> <p>ESCAN00044396, ESCAN00044109, ESCAN00043651</p>
Dominik Biber	2010-09-27	2.10.0	<p>Updated 3.1 Features</p> <p>Updated 3.3 States</p> <p>Added 3.7.1 Transmission of a Signal Group</p> <p>Added 3.8.1 Reception of a Signal Group</p> <p>Updated 6.4 Configuration in EcuC Data Base</p> <p>Updated 6.5.1 Com Parameters (ESCAN00044954, ESCAN00045321)</p>
Dominik Biber	2010-12-20	2.11.00	<ul style="list-style-type: none"> <li>&gt; Removed "TMS Trigger Enter False" (ESCAN00045552)</li> <li>&gt; Changed description of Tx I-PDU callout call context in 5.5.2 Callout Functions (ESCAN00045253)</li> <li>&gt; Support PduInfoType instead of the DataPtr (ESCAN00046124). Changed following API descriptions <ul style="list-style-type: none"> <li>5.4.1 Com_RxIndication</li> <li>5.4.2 Com_TriggerTransmit</li> <li>5.5.2 Callout Functions</li> </ul> </li> <li>&gt; Support Dynamic DLC (ESCAN00047020) Added 3.8.4 Dynamic DLC</li> <li>&gt; Added configuration restriction to 7.1.6 Transport Protocol API (CanTp) (ESCAN00046398)</li> <li>&gt; Updated figure Figure 3-11 Minimum Delay Time (ESCAN00046068)</li> <li>&gt; Added 3.3.2 Multiple I-PDU group reference (ESCAN00044036)</li> </ul>

Dominik Biber	2011-01-27	2.11.00	> Corrected definition of the signal filter F_MaskedNewDiffersMaskedOld (3.7.3 Transmit Signal Filters, ESCAN00047878)
Dominik Biber	2011-02-21	2.11.00	> Added 3.3.1 I-PDU group Configuration and updated 3.3.2 Multiple I-PDU group reference (ESCAN00048770)
Dominik Biber	2011-07-13	2.11.01	> Added restrictions to signal group access APIs (3.8.1 Reception of a Signal Group, 5.2.16 Com_ReceiveShadowSignal, 3.7.1 Transmission of a Signal Group, 5.2.20 Com_UpdateShadowSignal, ESCAN00050107) > Changed description of main functions (3.4 Main Functions, ESCAN00051633)

Table 1-1 History of the document

## 1.2 Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_COM.pdf	3.0.1
[2]	AUTOSAR	AUTOSAR_SWS_DET.pdf	2.2.0
[3]	AUTOSAR	AUTOSAR_BasicSoftwareModules.pdf	1.1.0
[4]	Vector	TechnicalReferencePostbuildProcess.pdf	0.2
[5]	Vector	AN-ISC-8-1118_MICROSAR_BSW_Compatibility_Check.pdf	1.00.00

Table 1-2 Reference documents



### Please note

We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector's release of the programs delivered to your company is expressly restricted to the configuration you have specified in the questionnaire.

## Contents

<b>1</b>	<b>Document Information .....</b>	<b>2</b>
1.1	History .....	2
1.2	Reference Documents .....	5
<b>2</b>	<b>Introduction .....</b>	<b>14</b>
2.1	Architecture Overview .....	15
<b>3</b>	<b>Functional Description .....</b>	<b>17</b>
3.1	Features .....	17
3.2	Initialization .....	18
3.3	States .....	19
3.3.1	I-PDU group Configuration .....	20
3.3.2	Multiple I-PDU group reference .....	23
3.4	Main Functions .....	26
3.5	Error Handling .....	27
3.5.1	Development Error Reporting .....	27
3.5.2	Production Code Error Reporting .....	31
3.6	Signal Types .....	32
3.7	Transmission of a Signal .....	32
3.7.1	Transmission of a Signal Group .....	34
3.7.2	Transmission Mode Selector .....	35
3.7.3	Transmit Signal Filters .....	36
3.7.4	Minimum Send Distance of an I-PDU .....	37
3.7.5	Transmission Deadline Monitoring .....	37
3.8	Reception of a Signal .....	39
3.8.1	Reception of a Signal Group .....	40
3.8.2	Reception Deadline Monitoring .....	41
3.8.3	Reception of Invalid Signal Values .....	41
3.8.4	Dynamic DLC .....	41
3.9	Signal Gateway .....	42
3.9.1	Signal routing requirements .....	43
3.9.2	Routing of signal groups .....	43
3.9.3	Routing of signals with update bits .....	43
3.9.4	Adding and removing gateway signals .....	43
3.9.5	Routing latency .....	43
<b>4</b>	<b>Integration .....</b>	<b>45</b>
4.1	Scope of Delivery .....	45

4.1.1	Static Files .....	45
4.1.2	Dynamic Files .....	45
4.2	Include Structure .....	46
4.3	Compiler Abstraction and Memory Mapping .....	46
4.4	Critical Sections .....	47
4.4.1	BSW Scheduler .....	47
4.4.2	Vector Standard Library .....	50
4.5	Operating System Requirements .....	50
<b>5</b>	<b>API Description .....</b>	<b>51</b>
5.1	Type Definitions .....	51
5.2	Services provided by COM .....	52
5.2.1	Com_InitMemory .....	52
5.2.2	Com_Init .....	52
5.2.3	Com_DelInit .....	53
5.2.4	Com_IpduGroupStart .....	53
5.2.5	Com_IpduGroupStop .....	54
5.2.6	Com_EnableReceptionDM .....	54
5.2.7	Com_DisableReceptionDM .....	55
5.2.8	Com_GetStatus .....	55
5.2.9	Com_GetConfigurationId .....	55
5.2.10	Com_GetConfigurationStringPtr .....	56
5.2.11	Com_GetVersionInfo .....	56
5.2.12	Com_MainFunctionRx .....	57
5.2.13	Com_MainFunctionTx .....	57
5.2.14	Com_MainFunctionRouteSignals .....	58
5.2.15	Com_ReceiveSignal .....	58
5.2.16	Com_ReceiveShadowSignal .....	59
5.2.17	Com_ReceiveSignalGroup .....	59
5.2.18	Com_SendSignal .....	60
5.2.19	Com_SendSignalGroup .....	61
5.2.20	Com_UpdateShadowSignal .....	61
5.2.21	Com_TriggerIPDUSend .....	62
5.2.22	Com_InvalidateSignal .....	62
5.2.23	Com_InvalidateSignalGroup .....	63
5.3	Services used by COM .....	63
5.4	Callback Functions .....	64
5.4.1	Com_RxIndication .....	64
5.4.2	Com_TriggerTransmit .....	65
5.4.3	Com_TxConfirmation .....	65
5.5	Configurable Interfaces .....	66

5.5.1	Notifications .....	66
5.5.1.1	Indication Notification.....	66
5.5.1.2	Confirmation Notification.....	67
5.5.1.3	Rx Timeout Notification.....	67
5.5.2	Callout Functions .....	68
5.5.2.1	I-PDU Callout.....	68
<b>6</b>	<b>Configuration.....</b>	<b>71</b>
6.1	Configuration in Dbc Data Base.....	72
6.2	Configuration in Ldf Data Base .....	78
6.3	Configuration in FIBEX Data Base.....	80
6.4	Configuration in EcuC Data Base .....	82
6.4.1	Com .....	82
6.4.1.1	ComConfig .....	82
6.4.1.2	ComGwMapping .....	83
6.4.1.3	ComGwDestination.....	83
6.4.1.4	ComGwSignal.....	84
6.4.1.5	ComGwSource .....	84
6.4.1.6	ComGwSignal.....	84
6.4.1.7	ComIPdu .....	85
6.4.1.8	ComTxIPdu .....	87
6.4.1.9	ComTxModeFalse .....	88
6.4.1.10	ComTxMode .....	88
6.4.1.11	ComTxModeTrue .....	89
6.4.1.12	ComTxMode .....	90
6.4.1.13	ComIPduGroup.....	91
6.4.1.14	ComSignal .....	91
6.4.1.15	ComFilter .....	96
6.4.1.16	ComSignalGroup .....	97
6.4.1.17	ComGroupSignal .....	101
6.4.1.18	ComFilter .....	104
6.4.1.19	ComGeneral .....	105
6.5	Configuration with GENy .....	116
6.5.1	Com Parameters.....	117
6.5.2	Signal gateway configuration.....	160
6.5.2.1	Automatic routing relations .....	160
6.5.2.2	Manual routing relations .....	160
6.5.2.3	Deleting manual routing relations .....	162
6.5.2.4	Gateway only signals.....	162
<b>7</b>	<b>AUTOSAR Standard Compliance.....</b>	<b>164</b>



7.1	Additions/ Extensions .....	164
7.1.1	Signal Conversion.....	164
7.1.1.1	Com_ConvertSignal.....	165
7.1.1.2	Com_Convert_SignedBusToEcu .....	165
7.1.1.3	Com_Convert_UnsignedBusToEcu .....	166
7.1.1.4	Com_Convert_SignedEcuToBus .....	167
7.1.1.5	Com_Convert_UnsignedEcuToBus .....	167
7.1.1.6	Configuration with GENy .....	169
7.1.2	Rx Signal Notification Flags.....	170
7.1.2.1	Com_GetRxSigIndicationFlag .....	171
7.1.2.2	Com_GetRxSigGrpIndicationFlag .....	171
7.1.2.3	Com_ClrRxSigIndicationFlag.....	172
7.1.2.4	Com_ClrRxSigGrpIndicationFlag.....	173
7.1.2.5	Configuration .....	174
7.1.3	Rx Timeout Flags.....	176
7.1.3.1	Com_GetRxSigTimeoutFlag .....	177
7.1.3.2	Com_GetRxSigGrpTimeoutFlag .....	177
7.1.3.3	Com_ClrRxSigTimeoutFlag .....	178
7.1.3.4	Com_ClrRxSigGrpTimeoutFlag .....	179
7.1.3.5	Configuration .....	179
7.1.4	Rx State On Flags.....	182
7.1.4.1	Com_GetRxSigStateOnFlag .....	182
7.1.4.2	Com_GetRxSigGrpStateOnFlag .....	182
7.1.4.3	Configuration .....	183
7.1.5	Com_IpduGroupTransmit .....	184
7.1.6	Transport Protocol API (CanTp).....	184
7.1.6.1	Com_TpProvideRxBuffer .....	185
7.1.6.2	Com_TpRxIndication .....	185
7.1.6.3	Com_TpProvideTxBuffer .....	186
7.1.6.4	Com_TpTxConfirmation.....	186
7.1.7	Gateway: Signal routing.....	187
7.1.8	Gateway: Rx signal timeout handling without update bits.....	187
7.1.9	TMS Switch Support .....	188
7.1.10	COM without Confirmations .....	188
7.1.11	Large Signals.....	188
7.2	Limitations.....	188
7.2.1	Component .....	188
7.2.2	Code Generator and Configurator .....	189
<b>8</b>	<b>Glossary and Abbreviations.....</b>	<b>190</b>
8.1	Glossary.....	190

8.2 Abbreviations ..... 193

**9 Contact..... 195**

## Illustrations

Figure 2-1	AUTOSAR architecture .....	15
Figure 2-2	Interfaces to adjacent modules of the COM .....	16
Figure 3-1	State Machine of an I-PDU Group .....	19
Figure 3-2	Creating an I-PDU Group .....	20
Figure 3-3	Removing an I-Pdu Group .....	21
Figure 3-4	Configuration of I-PDU Group Hierarchy .....	22
Figure 3-5	Assigning I-PDUs to I-PDU Groups .....	23
Figure 3-6	Creating an I-PDU group reference .....	24
Figure 3-7	Assigning I-PDUs to multiple I-PDU Groups .....	25
Figure 3-8	Removing an I-PDU group reference .....	26
Figure 3-9	Periodic Transmission Mode .....	33
Figure 3-10	Direct Transmission Mode .....	33
Figure 3-11	Minimum Delay Time .....	37
Figure 3-12	Transmission Deadline Monitoring - Variant 1 - Cyclic Transmission .....	38
Figure 3-13	Transmission Deadline Monitoring - Variant 1 - Direct Transmission .....	39
Figure 3-14	Transmission Deadline Monitoring - Variant 2 .....	39
Figure 3-15	Reception of a periodic signal .....	40
Figure 3-16	Signal routing allows routing of individual signals .....	42
Figure 4-1	Include structure .....	46
Figure 6-1	Component Selection in the System Configuration .....	116
Figure 6-2	Creating a manual routing relation .....	161
Figure 6-3	Signal routing parameters .....	161
Figure 6-4	1:N routings are configured as several 1:1 routing relations .....	162
Figure 6-5	Deleting a manual routing relation .....	162
Figure 6-6	Marking a signal as gateway only signal ("Internal") .....	162
Figure 7-1	Signal conversion parameters .....	169
Figure 7-2	Import of an ASAP2 data base .....	170
Figure 7-3	Signal indication flag configuration .....	174
Figure 7-4	Protect Indication Flag Access parameter .....	175
Figure 7-5	Indication flag macro postfix .....	175
Figure 7-6	Signal indication flag configuration .....	180
Figure 7-7	Timeout Flag Access parameter .....	181
Figure 7-8	Timeout flag macro postfix .....	181
Figure 7-9	Signal state on flag configuration .....	183
Figure 7-10	State On flag macro postfix .....	184

## Tables

Table 1-1	History of the document .....	5
Table 1-2	Reference documents .....	5
Table 3-1	Supported SWS features .....	18
Table 3-2	Not supported SWS features .....	18
Table 3-3	Main functions that have to be called cyclically .....	27
Table 3-4	Service IDs .....	28
Table 3-5	Errors reported to DET .....	31
Table 4-1	Static files .....	45
Table 4-2	Generated files .....	45
Table 4-3	Compiler abstraction and memory mapping .....	47
Table 5-1	Type definitions .....	51
Table 5-2	Com_InitMemory .....	52

Table 5-3	Com_Init.....	52
Table 5-4	Com_DeInit.....	53
Table 5-5	Com_IpduGroupStart.....	54
Table 5-6	Com_IpduGroupStop.....	54
Table 5-7	Com_EnableReceptionDM.....	55
Table 5-8	Com_DisableReceptionDM.....	55
Table 5-9	Com_GetStatus.....	55
Table 5-10	Com_GetConfigurationId.....	56
Table 5-11	Com_GetConfigurationStringPtr.....	56
Table 5-12	Com_GetVersionInfo.....	57
Table 5-13	Com_MainFunctionRx.....	57
Table 5-14	Com_MainFunctionTx.....	58
Table 5-15	Com_MainFunctionRouteSignals.....	58
Table 5-16	Com_ReceiveSignal.....	59
Table 5-17	Com_ReceiveShadowSignal.....	59
Table 5-18	Com_ReceiveSignalGroup.....	60
Table 5-19	Com_SendSignal.....	61
Table 5-20	Com_SendSignalGroup.....	61
Table 5-21	Com_UpdateShadowSignal.....	62
Table 5-22	Com_TriggerIPDUSend.....	62
Table 5-23	Com_InvalidateSignal.....	63
Table 5-24	Com_InvalidateSignalGroup.....	63
Table 5-25	Services used by the COM.....	64
Table 5-26	Com_RxIndication.....	64
Table 5-27	Com_TriggerTransmit.....	65
Table 5-28	Com_TxConfirmation.....	66
Table 5-29	Indication Notification.....	66
Table 5-30	Confirmation Notification.....	67
Table 5-31	Timeout Notification.....	67
Table 5-32	Rx I-PDU Callout with SDU pointer.....	68
Table 5-33	Tx I-PDU Callout with SDU pointer.....	69
Table 5-34	Rx I-PDU Callout with PduInfo pointer.....	70
Table 5-35	Tx I-PDU Callout with PduInfo pointer.....	70
Table 6-1	Dbc Data base attributes.....	74
Table 6-2	Mapping of AUTOSAR EcuC and Dbc data base attributes.....	77
Table 6-3	Mapping of AUTOSAR EcuC and Ldf data base attributes.....	79
Table 6-4	Mapping of AUTOSAR EcuC and FIBEX data base attributes.....	81
Table 7-1	Main COM variants.....	164
Table 7-2	Com_ConvertSignal.....	165
Table 7-3	Com_Convert_SignedBusToEcu.....	165
Table 7-4	Com_Convert_UnsignedBusToEcu.....	166
Table 7-5	Com_Convert_SignedEcuToBus.....	167
Table 7-6	Com_Convert_UnsignedEcuToBus.....	168
Table 7-7	Com_GetRxSigIndicationFlag.....	171
Table 7-8	Com_GetRxSigGrpIndicationFlag.....	172
Table 7-9	Com_ClrRxSigIndicationFlag.....	173
Table 7-10	Com_ClrRxSigGrpIndicationFlag.....	173
Table 7-11	Com_GetRxSigTimeoutFlag.....	177
Table 7-12	Com_GetRxSigGrpTimeoutFlag.....	178
Table 7-13	Com_ClrRxSigTimeoutFlag.....	178
Table 7-14	Com_ClrRxSigGrpTimeoutFlag.....	179
Table 7-15	Com_GetRxSigStateOnFlag.....	182
Table 7-16	Com_GetRxSigGrpStateOnFlag.....	183
Table 7-17	Com_IpduGroupTransmit.....	184

Table 7-18	Com_TpProvideRxBuffer .....	185
Table 7-19	Com_TpRxIndication.....	186
Table 7-20	Com_TpProvideTxBuffer.....	186
Table 7-21	Com_TpTxConfirmation .....	187
Table 8-1	Glossary .....	193
Table 8-2	Abbreviations .....	194

## 2 Introduction

This document describes the functionality, API and configuration of the AUTOSAR BSW module COM as specified in [1].

<b>Supported AUTOSAR Release*:</b>	3	
<b>Supported Configuration Variants:</b>	pre-compile, link-time, post-build	
<b>Vendor ID:</b>	COM_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
<b>Module ID:</b>	COM_MODULE_ID	50 decimal (according to ref. [3])

\* For the precise AUTOSAR Release 3.x please see the release specific documentation.

The main purpose of the AUTOSAR COM component is to provide a signal-based interface to the upper layer. In an AUTOSAR system this is the RTE. In a non-AUTOSAR system this is the application interfaces with the COM layer directly.

It is possible to use the COM layer with different underlying bus systems since they are encapsulated by the PDU Router. Figure 2-1 shows how the component is embedded in the AUTOSAR layered architecture.

The main features of the COM component are:

- > Provision of interface for signed and unsigned signals to the RTE
- > Packing and unpacking of signals in I-PDUs
- > Handling of transmission modes
- > Minimum distance between transmit I-PDUs
- > Communication control by starting and stopping of I-PDU groups
- > Rx deadline monitoring
- > Notification mechanisms
- > Initial value support

To allow for late configuration changes in a complete ECU the COM component supports the post-build time configuration concept (Configuration conformance class CCC3).

The implementation is based on the AUTOSAR COM specification [1]. It is assumed that the reader is familiar with this document and other related AUTOSAR specifications.

## 2.1 Architecture Overview

Figure 2-1 shows where the COM is located in the AUTOSAR architecture.

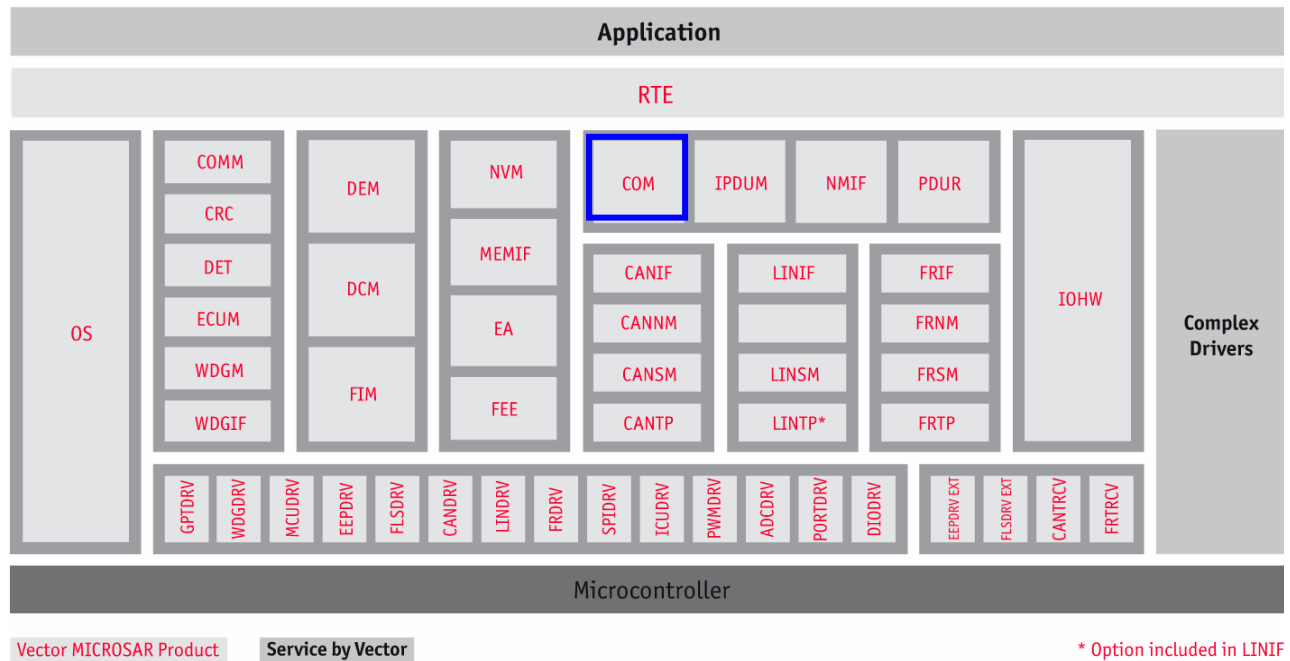


Figure 2-1 AUTOSAR architecture

The interfaces to adjacent modules of the COM is shown in Figure 2-2. These interfaces are described in chapter 5.

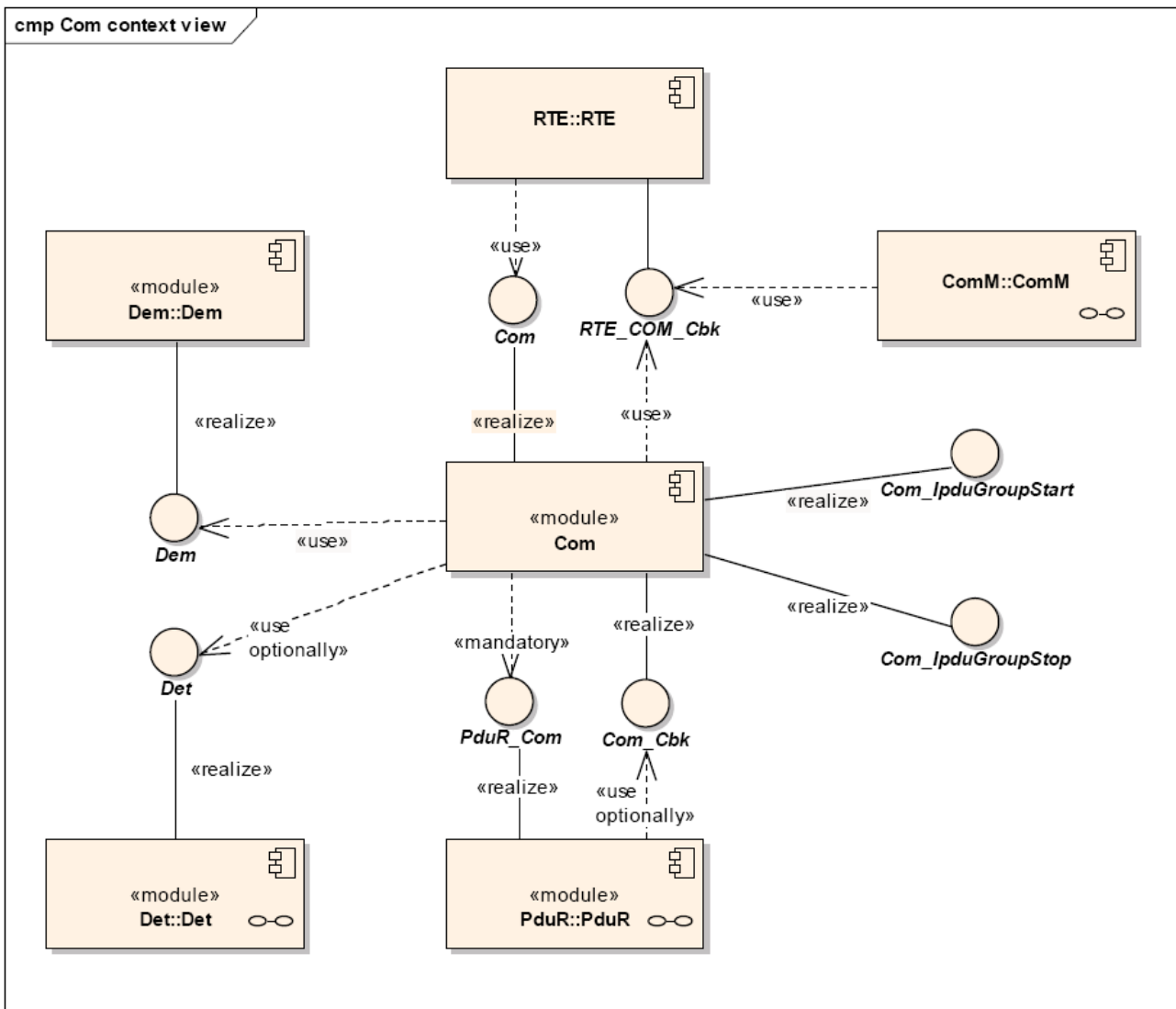


Figure 2-2 Interfaces to adjacent modules of the COM

Applications do not access the services of the BSW modules directly. They use the service ports provided by the BSW modules via the RTE.



## 3 Functional Description

### 3.1 Features

The features listed in this chapter cover the complete functionality specified in [1]. It has been designed and tested for the CAN, LIN and FlexRay bus.

The "supported" and "not supported" features are presented in the following two tables. For further information of not supported features also see chapter 7.

The following features described in [1] are supported:

Supported Feature
<b>Configuration</b>
Support for AUTOSAR xml format
Support for Vector IL CANdb attributes
<b>Deadline Monitoring</b>
Rx based on reception of I-PDUs
Rx based on update bits
Tx based on I-PDU transmission
<b>General</b>
AUTOSAR specific error handling
AUTOSAR specific file structure
Callouts
Error Notification of non confirmed transmitted I-PDUs
Hierarchical I-PDU Groups
<b>Layer Management</b>
Support for I-PDU groups
Initialization and De-Initialization service
Cyclic main functions
Services for status and version information
<b>Lower Layer Interface</b>
AUTOSAR PDU Router
Vector CAN Driver

<b>Signal Interface</b>
Support for initialization values
Support for signal groups
Support for signed and unsigned signals
Support for update bits
Configurable call context of signal indications
Signal filters
Signal conversion
Signal Invalidation
<b>Transmission modes</b>
Support for the basic transmission modes Direct, Periodic and Mixed
Minimum send distance between I-PDUs
Operation without confirmation
Multiple Tx modes per I-PDU
Replication of Tx requests (n-times mode)
Signal Gateway functionality

Table 3-1 Supported SWS features

The following features described in [1] are not supported:

<b>Not Supported Feature</b>
<b>Deadline Monitoring</b>
Signal Filter OneEveryN
Rx Signal Filters (only 'OnChange' is supported)

Table 3-2 Not supported SWS features

## 3.2 Initialization

Before the COM layer can be used it has to be initialized by Com\_Init() which performs the basic initialization but does not enable the transmission or reception of signals and I-PDUs. Initialization, starting and stopping of the layer and its I-PDU groups is normally driven by the Communication Manager. If this software component is not available a similar component has to be provided by the integrator.

### 3.3 States

Starting and stopping of I-PDUs is performed on I-PDU group basis. These groups are configured in the configuration tool. At least one group for Rx and one group for Tx I-PDUs is required on each physical channel. The groups are started and stopped by the API functions `Com_IpduGroupStart` and `Com_IpduGroupStop`.

The Rx deadline monitoring (s. 3.8.2) is also enabled and disabled on an I-PDU group basis. The resulting state chart of an Rx I-PDU group is shown in the following figure:

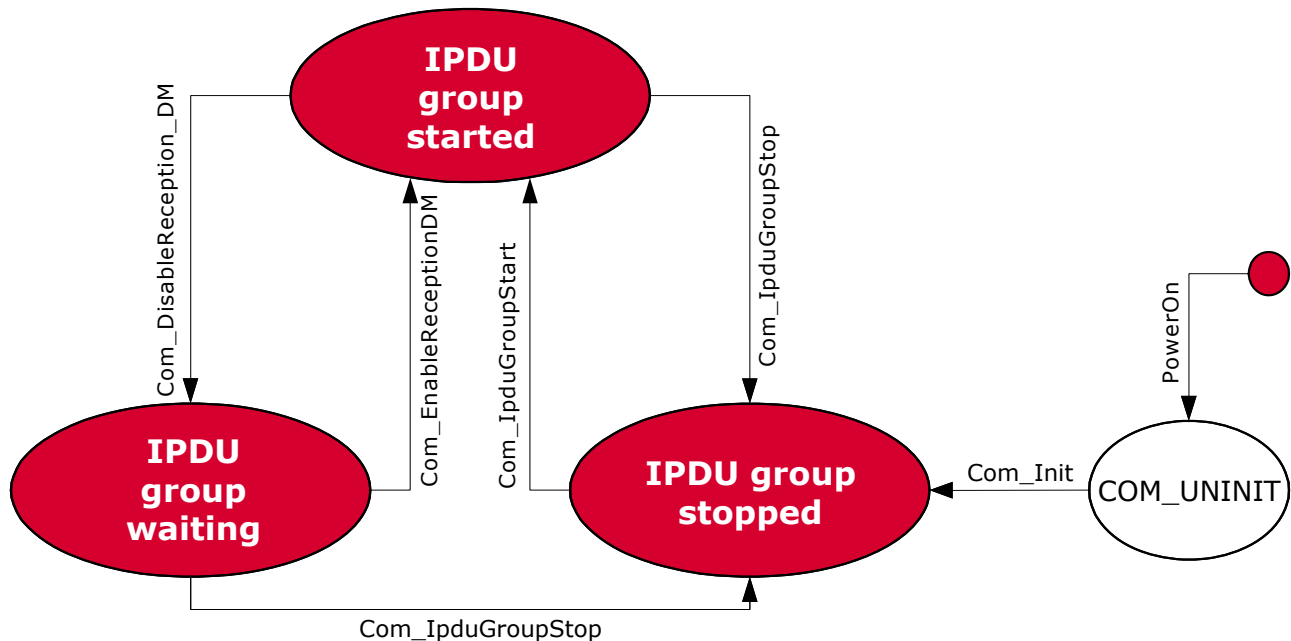


Figure 3-1 State Machine of an I-PDU Group

The Vector COMM implementation requires at least one I-PDU group both for Rx and Tx on each physical channel with the following naming:

Single channel:

```
#define COM_ALLTXIPDUS_CH0
```

```
#define COM_ALLRXIPDUS_CH0
```

Multiple channels:

```
#define COM_ALLTXIPDUS_CH0
```

```
#define COM_ALLRXIPDUS_CH0
```

```
#define COM_ALLTXIPDUS_CH1
```

```
#define COM_ALLRXIPDUS_CH1
```

### 3.3.1 I-PDU group Configuration

- > To create a new I-PDU group, right-click on the “IPdu Groups” node in the GENy components view and select “Add new IPdu Group” as shown in Figure 3-2. Configure a unique name for each newly created I-Pdu group.

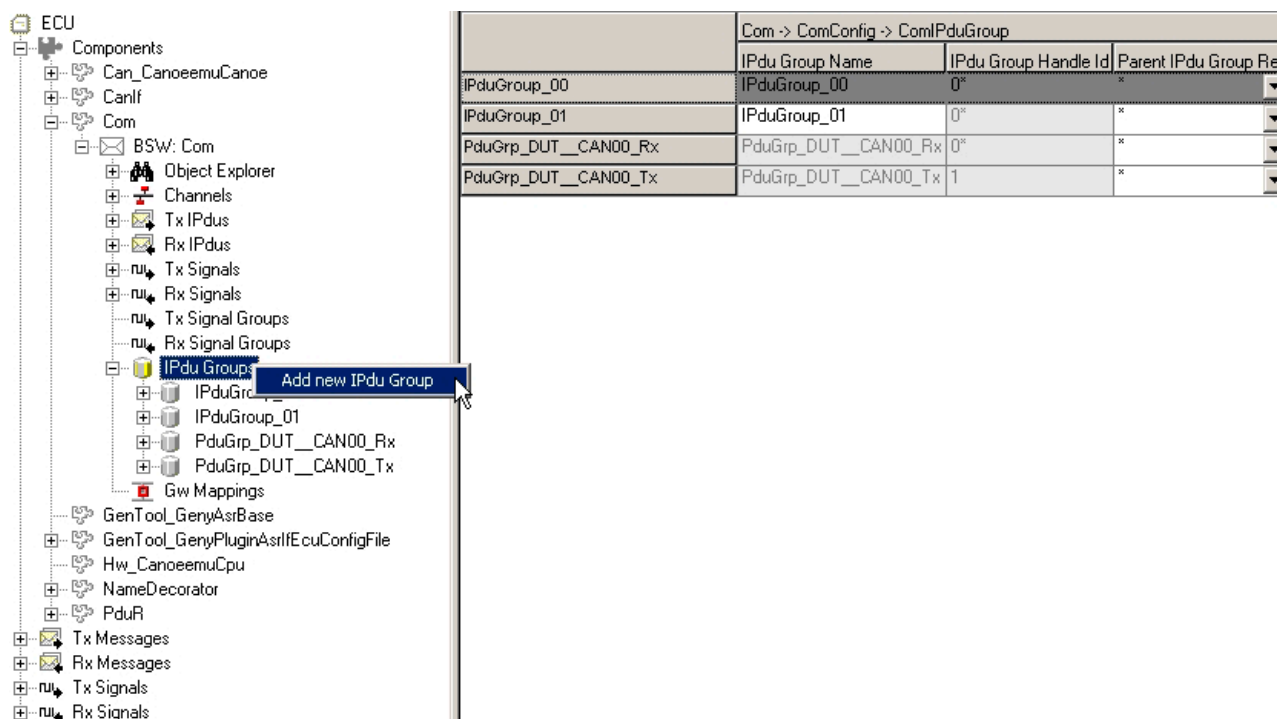


Figure 3-2 Creating an I-PDU Group

- > Remove an I-PDU group by right-clicking on the I-PDU group and selecting “Remove Ipdu Group” as shown in Figure 3-3.



#### Note

Removing an I-PDU group is only possible if the I-PDU group is not derived from the system extract. Otherwise the “Remove Ipdu Group” action is greyed out.

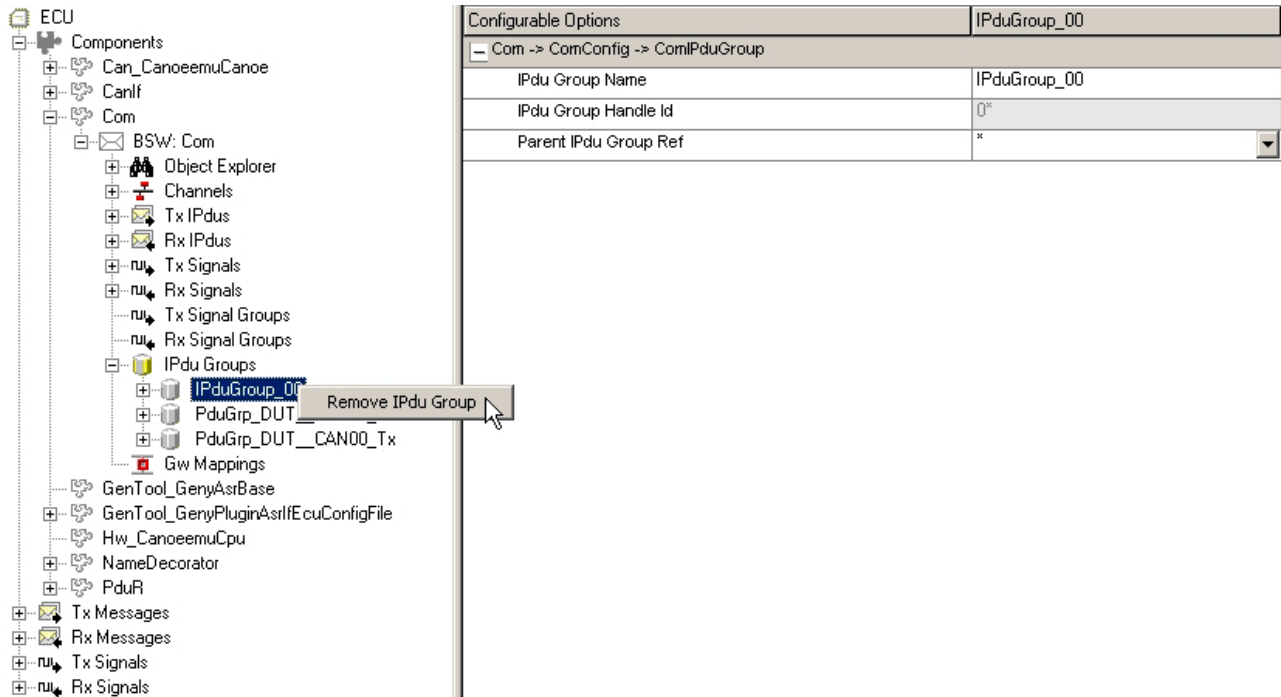


Figure 3-3 Removing an I-Pdu Group

- > An I-PDU group hierarchy can be configured by selecting the parent I-PDU group as shown in Figure 3-4 for an I-PDU group.



#### Note

Only I-PDU groups

- > without assigned I-PDUs
  - > or with assigned I-PDUs with the same ComIPduDirection (SEND/RECEIVE)
- are selectable as parent I-PDU groups.

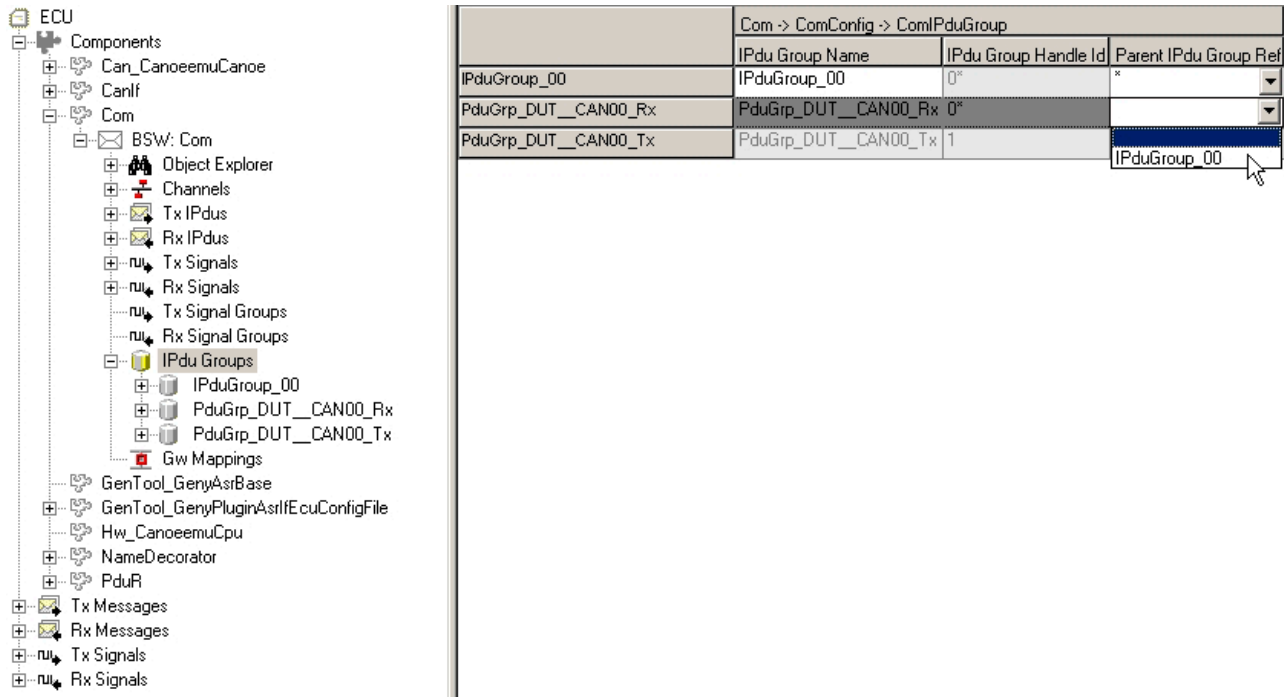


Figure 3-4 Configuration of I-PDU Group Hierarchy

- > To assign an I-PDU to an I-PDU Group select the corresponding I-PDU Group Ref for the I-PDU in the Rx or Tx I-PDUs list as shown in Figure 3-5.



### Note

Only I-PDU groups

- > with assigned I-PDUs with the same ComIPduDirection (SEND/RECEIVE) are selectable as I-PDU Group Ref.

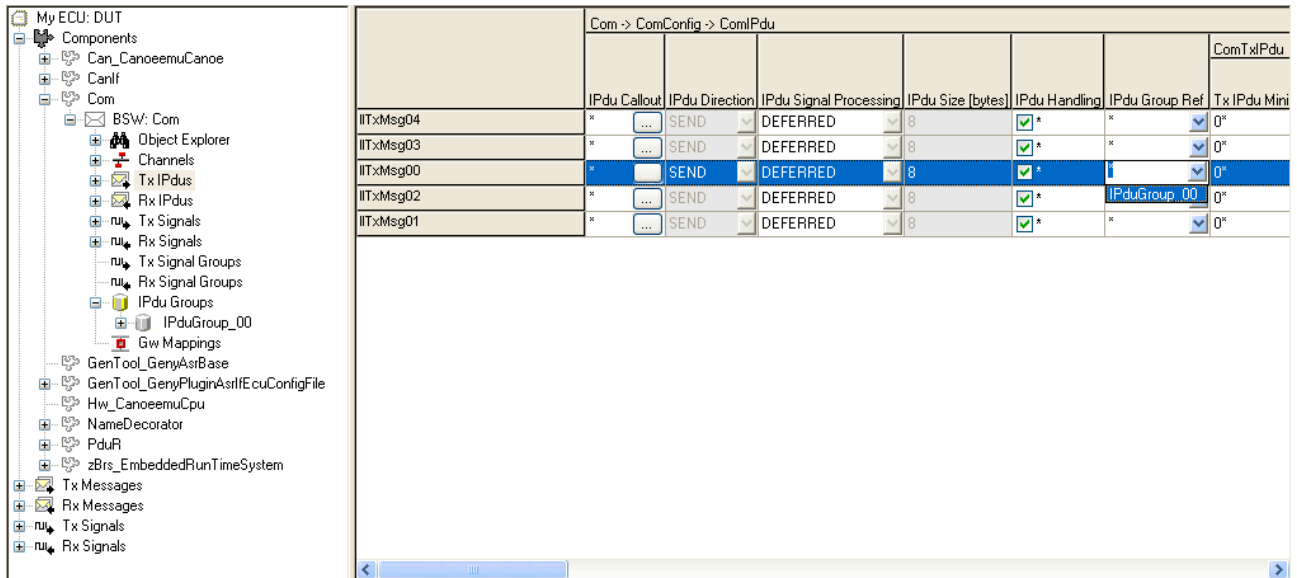


Figure 3-5 Assigning I-PDUs to I-PDU Groups

### 3.3.2 Multiple I-PDU group reference

If the 'Multiple I-Pdu Group Reference' support is enabled, one I-PDU could be contained in more than one I-PDU group.

In this case an I-PDU is

- ▶ Active  
if at least one assigned I-PDU group is started
- ▶ Inactive  
if all assigned I-PDU groups are stopped.

To assign an I-PDU to several I-PDU groups

- First, create a new I-PDU group reference object by right-clicking on to I-PDU node in GENy and selecting "Add new Ipdu Group Reference" as shown in Figure 3-6.

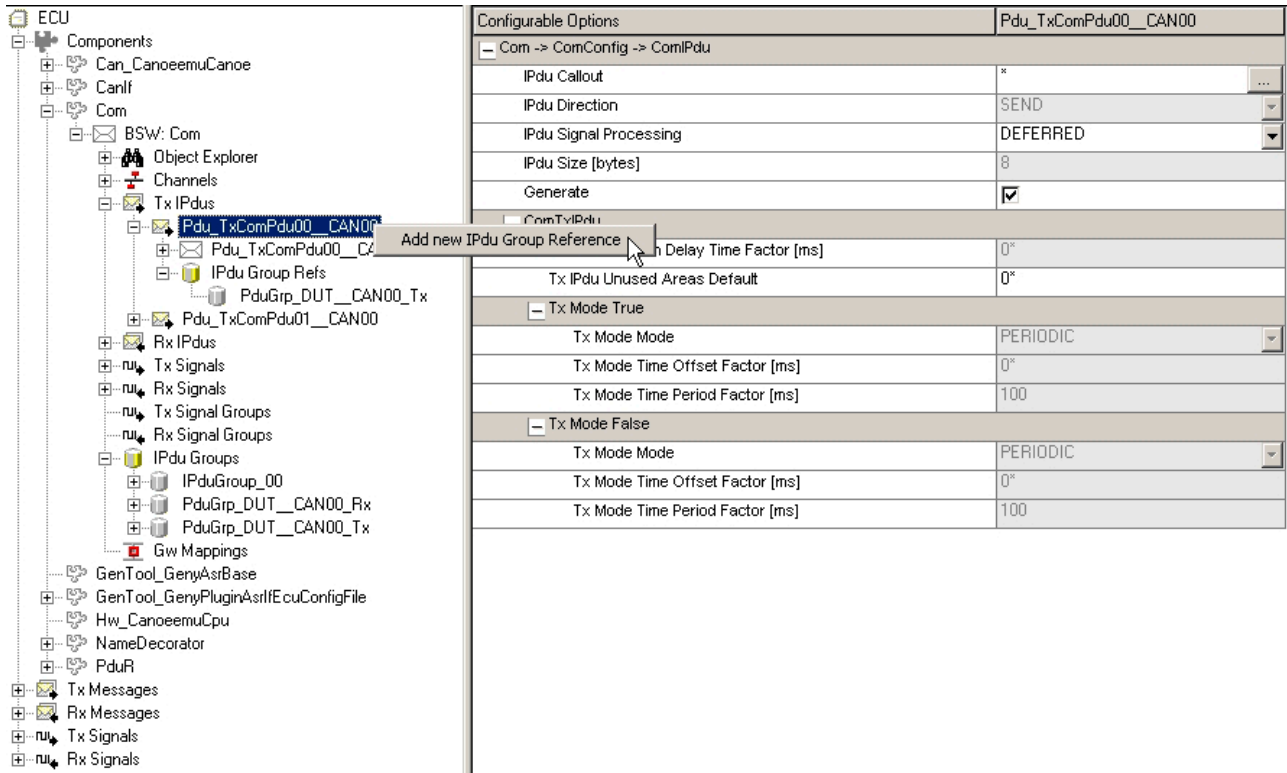


Figure 3-6 Creating an I-PDU group reference

> Second, select the I-PDU group as I-PDU group reference as shown in Figure 3-7.



### Note

Only I-PDU groups

> with assigned I-PDUs with the same ComIPduDirection (SEND/RECEIVE)  
are selectable as I-PDU Group Ref.



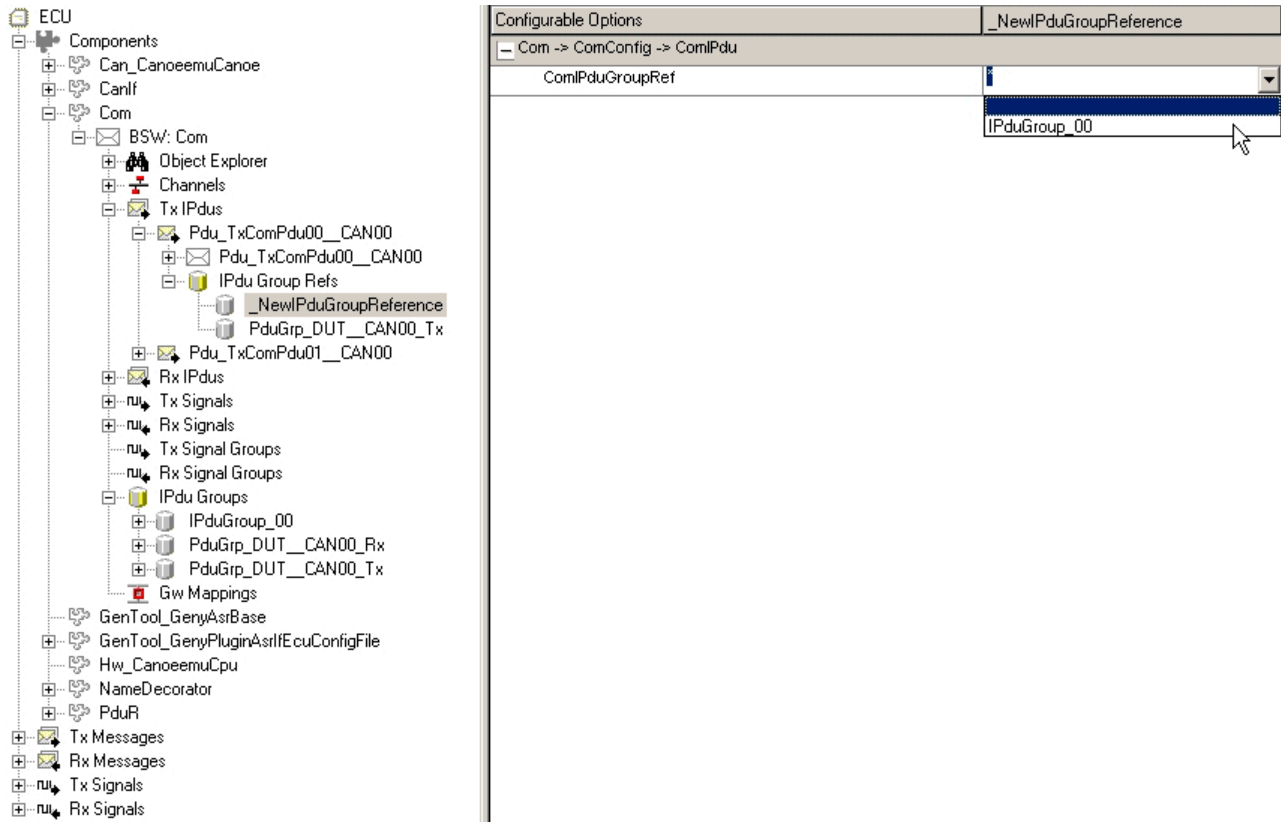


Figure 3-7 Assigning I-PDUs to multiple I-PDU Groups

- > To remove an assigned I-PDU from an I-PDU group, right-click on the I-PDU group reference object and select “Remove Ipdu Group Reference” as shown in Figure 3-8.



### Note

Removing an I-PDU group reference is only possible if the I-PDU group reference is not derived from the system extract. Otherwise the “Remove Ipdu Group Reference” action is greyed out.

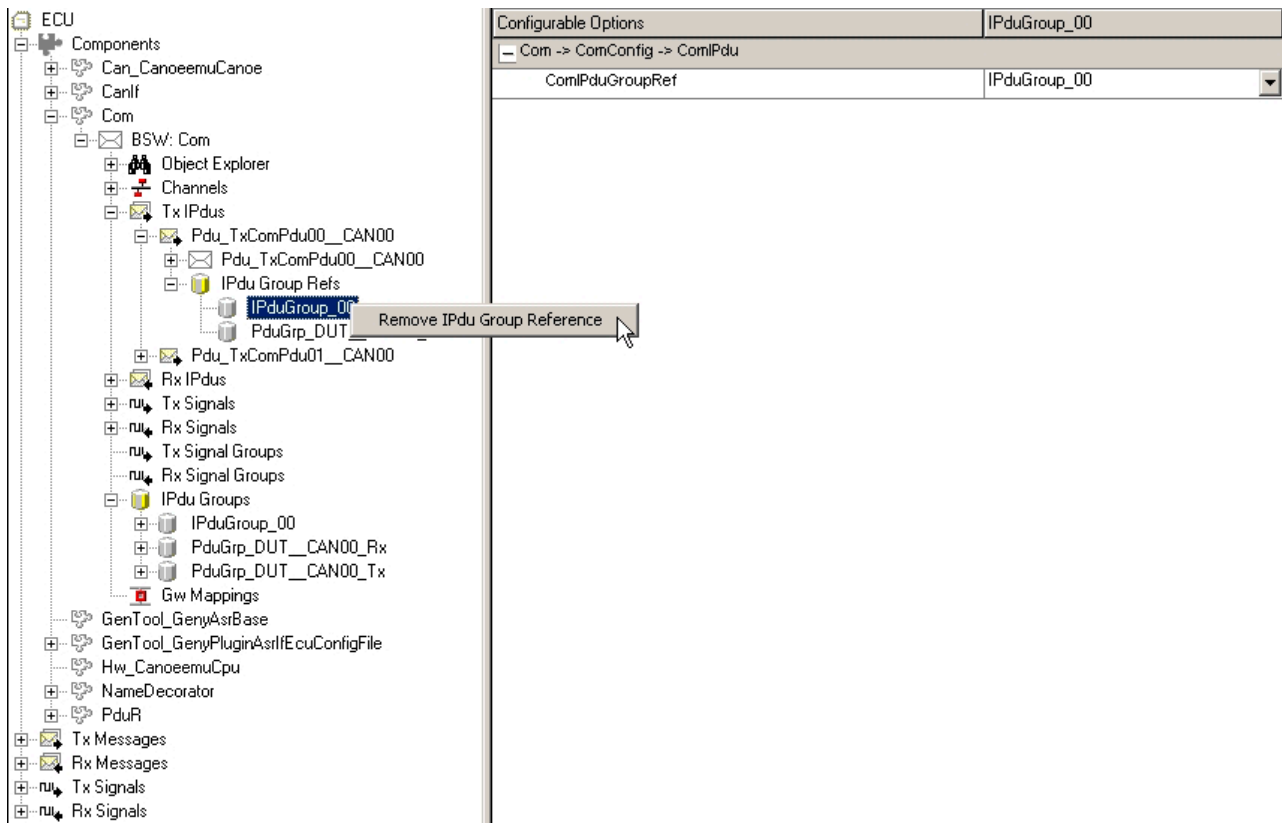


Figure 3-8 Removing an I-PDU group reference

### 3.4 Main Functions

COM provides three functions that have to be called cyclically by the Basic Software Scheduler or a similar component.

Main Functions	Description
Com_MainFunctionRx()	<p>This function performs the following reception processings</p> <ul style="list-style-type: none"> <li>&gt; Reception deadline monitoring</li> <li>&gt; Deferred reception notification</li> </ul> <p>This function must be called cyclically with a cycle time identical to the configured 'Configuration Rx Time Base'.</p>
Com_MainFunctionTx()	<p>This function performs the following transmission processings</p> <ul style="list-style-type: none"> <li>&gt; Transmission of I-PDUs</li> <li>&gt; Transmission deadline monitoring</li> <li>&gt; Deferred transmission notification</li> </ul> <p>This function must be called cyclically with a cycle time identical to the configured 'Configuration Tx Time Base'.</p>
Com_MainFunctionRouteSignals() (Gateway use case only)	<p>If the signal based gateway is used, this task has to be called in an user defined call cycle. If the function is not</p>

Main Functions	Description
	<p>called, no signal routing will be carried out.</p> <p>The call cycle is not configurable in the configuration tool but has direct effect on the maximum routing latency.</p> <p>If update bits are used, the call cycle must be faster than the fastest message reception cycle time. If this is not given, the gateway might lose update bit events. This can cause that some signals with update bits are not routed.</p>

Table 3-3 Main functions that have to be called cyclically

### 3.5 Error Handling

#### 3.5.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [2], if development error reporting is enabled (i.e. pre-compile parameter `COM_DEV_ERROR_DETECT==STD_ON`).

If another module is used for development error reporting, the function prototype for reporting the error can be configured by the integrator, but must have the same signature as the service `Det_ReportError()`.

The reported COM ID is 50.

The reported service IDs identify the services which are described in 5.2. The following table presents the service IDs and the related services:

Service ID	Service
1	COMServiceId_Init
2	COMServiceId_DeInit
3	COMServiceId_IpduGroupStart
4	COMServiceId_IpduGroupStop
5	COMServiceId_DisableReceptionDM
6	COMServiceId_EnableReceptionDM
7	COMServiceId_GetStatus
8	COMServiceId_GetConfigurationId
9	COMServiceId_GetVersionInfo
10	COMServiceId_SendSignal
11	COMServiceId_ReceiveSignal
12	COMServiceId_UpdateShadowSignal
13	COMServiceId_SendSignalGroup
14	COMServiceId_ReceiveSignalGroup
15	COMServiceId_ReceiveShadowSignal
16	COMServiceId_InvalidateSignal
17	COMServiceId_ErrorGetServiceId

Service ID	Service
19	COMServiceId_TriggerTransmit
20	COMServiceId_RxIndication
21	COMServiceId_TxConfirmation
22	COMServiceId_InvalidateShadowSignal
23	COMServiceId_TriggerIPDUSend
24	COMServiceId_MainFunctionRx
25	COMServiceId_MainFunctionTx
26	COMServiceId_MainFunctionRouteSignals
27	COMServiceId_InvalidateSignalGroup
128	COMServiceId_II_AsrComInternal
129	COMServiceId_IpduGroupTransmit
130	COMServiceId_TpProvideTxBuffer
131	COMServiceId_TpProvideRxBuffer
132	COMServiceId_TpTxConfirmation
133	COMServiceId_TpRxIndication
134	COMServiceId_GetConfigurationString

Table 3-4 Service IDs

The errors reported to DET are described in the following table:

Error Code	Description
3	Com_TxModeHdlr_Param_TriggerDeferredIpduSend The Parameter of Com_TxModeHdlr_TriggerDeferredIpduSend is too large.
4	Com_TxModeHdlr_Param_Confirmation The Parameter of Com_TxModeHdlr_Confirmation is too large.
5	Com_TxModeHdlr_Param_StopCyclicTransmission The Parameter of Com_TxModeHdlr_StopCyclicTransmission is too large.
6	Com_TxModeHdlr_Param_StartImmediateCyclicTransmission The Parameter of Com_TxModeHdlr_StartImmediateCyclicTransmission is too large.
7	Com_TxModeHdlr_Param_StartDeferredCyclicTransmission The Parameter of Com_TxModeHdlr_StartDeferred is too large.
8	Com_TxModeHdlr_Param_Stop The Parameter of Com_TxModeHdlr_Stop is too large.
9	Com_TxModeHdlr_Param_TriggerImmediateIpduSend The Parameter of Com_TxModeHdlr_TriggerImmediateIpduSend is too large.
10	Com_RxBuffer_SignalId_Param The SignalId (handled as parameter to a RxBuffer function) is too large
11	Com_RxBuffer_InvalidPointer The data pointer handled by the application to a

Error Code		Description
	r_Param	RxBuffer function is invalid
12	Com_RxBuffer_Ipduld_Param	The Ipduld (handled as parameter to a RxBuffer function) is invalid
13	Com_RxBuffer_InvalidByteOffset	The signal byte offset retrieved from a configuration table is too large
14	Com_RxBuffer_SignalGroupIld_Param	The SignalGroupIld (handled as parameter to a RxBuffer function) is too large
15	Com_TxBuffer_SignalIld_Param	The SignalIld (handled as parameter to a TxBuffer function) is too large
16	Com_TxBuffer_InvalidPointer_Param	The data pointer handled by the application to a TxBuffer function is invalid
17	Com_TxBuffer_Ipduld_Param	The Ipduld (handled as parameter to a TxBuffer function) is invalid
18	Com_TxBuffer_InvalidByteOffset	The signal byte offset retrieved from a configuration table is too large
19	Com_TxBuffer_SignalGroupIld_Param	The SignalGroupIld (handled as parameter to a TxBuffer function) is too large
20	Com_Signal_UnsupportedBusSignalType	Unsupported bus signal type. Configuration data might be corrupted
21	Com_Signal_VariableOutOfRange	The tested variable will not fit into the provided data type
22	Com_Signal_InconsistentConfigurationData	The configuration data is not consistent. The tested config value must not occur in this section
23	Com_TxLLIf_Transmit_PdulnfoNullPointer	The PdulInfo.SduDataPtr is a null pointer.
24	Com_TriggerTransmit_PdulnfoNullPointer	The PdulInfo.SduDataPtr is a null pointer.
25	Com_RxIndication_PdulInfoNullPointer	The PdulInfo.SduDataPtr is a null pointer.
26	Com_RxIndication_PdulInfoLength	The PdulInfo.SduLength is too large.
27	Com_TriggerTransmit_PdulnfoLength	The PdulInfo.SduLength is too large.
28	Com_TxLLIf_Transmit_PdulnfoLength	The PdulInfo.SduLength is too large.
29	Com_ReceiveSignal_Pduld	The Pduld retrieved from Com_Signal subcomponent is too large.
30	Com_ReceiveShadowSignal_Pduld	The Pduld retrieved from Com_Signal subcomponent is too large.
31	Com_ReceiveSignalGroup_Pduld	The Pduld retrieved from Com_Signal subcomponent is too large.
32	Com_SendSignal_Pduld	The Pduld retrieved from Com_Signal subcomponent is too large.
33	Com_SendSignalGroup_Pdu	The Pduld retrieved from Com_Signal subcomponent is

Error Code		Description
	Id	too large.
34	Com_UpdateShadowSignal_Pdul	The Pdul retrieved from Com_Signal subcomponent is too large.
35	Com_TxLLIf_Transmit_IpdulParam	The Parameter of Com_TxLLIf_Transmit is too large.
36	Com_TxSigIf_Init_SignalGroupIndex	COM_SIGNALGROUPINDEX is too large.
37	Com_UpdateShadowSignal_SigGroupId	Com_UpdateShadowSignal was called for a signal that doesn't belong to a signal group.
38	Com_RxNHdlr_IpdulParamTooLarge	The IPDU parameter of a RxNHdlr service function is > number of configured Pdus.
39	Com_RxNHdlr_TimeoutFuncIdxTooLarge	Error function pointer index too large
40	Com_RxNHdlr_NotiFuncIdxTooLarge	Notification function pointer index too large
41	Com_TxBuffer_InvalidBitPosition	The indicated bit position is not within the allowed range (0..7)
42	Com_RxBuffer_InvalidBitPosition	The indicated bit position is not within the allowed range (0..7)
43	Com_RxUbHdlr_Ipdul_Param	The Ipdul (handled as parameter to a RxUbHdlr function) is invalid
44	Com_RxUbHdlr_InvalidFctPtr	A NULL pointer is referenced within a deferred notification function call. Flag must never be set for a Ub without a configured notification.
45	Com_RxUbHdlr_InvalidUbHdl	A invalid update bit handle was detected within the IPDU to Ub Hdl ROM table
46	Com_RxDIMon_Indication_Param	The Ipdul (handled as parameter to a RxDIMon function) is invalid
47	Com_RxDIMon_Stop_Param	The Ipdul (handled as parameter to a RxDIMon function) is invalid
48	Com_TxNHdlr_IpdulParamTooLarge	The IPDU parameter of a TxNHdlr service function is > number of configured Pdus.
49	Com_TxNHdlr_NotiFuncIdxTooLarge	Notification function pointer index too large
50	Com_TxNHdlr_TimeoutFuncIdxTooLarge	Error function pointer index too large
51	Com_LMgt_NullPointerParameter	A pointer parameter is zero. This is not an allowed address.
53	Com_IpduGroupTransmit_IpduGroupStopped	The Ipdu group of an Ipdu is not started. Perform Com_IpduGroupStart for the Ipdu group, before this method is used for th Ipdu Group.
54	Com_SignalGw_InvalidGwHandle	Gateway hdl. retrieved from Com_SignalGw_RxIpdu2GwHandle is invalid
55	Com_SignalGw_InvalidSignal	Signal hdl. retrieved from a routing table is invalid

Error Code		Description
	IHandle	
56	Com_SignalGw_InvalidUpdateBitPosition	Update bit position retrieved from a routing table is invalid
57	Com_SignalGw_InvalidSignalGrpHandle	Signal group hdl. retrieved from a routing table is invalid
58	Com_RxNHdlr_InvalidSigIdForIndFlagAccess	RxSigId parameter used for indication flag access is invalid
59	Com_RxNHdlr_InvalidSigIdForToutFlagAccess	RxSigId parameter used for timeout flag access is invalid
60	Com_RxNHdlr_NotiFlagIdxTooLarge	Too large indication flag index retrieved from configuration data
61	Com_RxNHdlr_TimeoutFlagIdxTooLarge	Too large timeout flag index retrieved from configuration data
62	Com_LMgt_PduIdTypeTooSmall	The PduIdType cannot hold the number of used I-Pdus
63	Com_TxModeHdlr_Param_TriggerIpduSendOnceDeferred	The Parameter of Com_TxModeHdlr_TriggerIpduSendOnceDeferred is too large.
64	Com_Buffer_SignalId_Param	The SignalId (handled as parameter to a RxBuffer function) is too large
65	Com_Buffer_InvalidPointer_Param	The data pointer handled by the application to a RxBuffer function is invalid
66	Com_LMgt_SizeOfBoolean	Size of datatype 'boolean' must be '1' for a correct behavior of the functions Com_Signal_WriteSignal() and Com_Signal_ReadSignal()
67	Com_RxInv_NoInvalidValue	Function was called for a signal without a configured invalid value
68	Com_RxUbHdlr_TimeoutFlagIdxTooLarge	Too large update-bit timeout flag index retrieved from configuration data
69	Com_RxUbHdlr_NotificationFlagIdxTooLarge	Too large update-bit notification flag index retrieved from configuration data
70	Com_LMgt_SubIpduGroupId	The Sub I-Pdu Group ID is too large.
71	Com_LMgt_IpduId	The I-Pdu ID is too large.

Table 3-5 Errors reported to DET

### 3.5.2 Production Code Error Reporting

No production error codes are currently defined for COM.

### 3.6 Signal Types

The following signal types are supported:

- boolean
- uint8
- uint16
- uint32
- sint8
- sint16
- sint32
- uint8[n]

For signed and unsigned integers an endianness conversion is supplied depending on the endianness of the signal and the target system.

The support of signed signals is based on the B-complement.

The data type opaque is interpreted as an unsigned integer; the target system specific endianness conversion is applied.



#### Caution

Only **unsigned and signed integer** values are supported. Float and the scaling factors (as adjustable in CANdb++) are not supported and have to be interpreted by the application.

### 3.7 Transmission of a Signal

To request the transmission of a signal the upper layer uses the API `Com_SendSignal`. After performing optional parameter checks COM updates the I-PDU with the new signal value and checks if the transfer property of the signal requires a direct transmission. If yes, a flag is set which is evaluated later in the cyclic main function of the COM layer's transmit part.

Transmission modes of the I-PDU are handled in the `Com_MainFunctionTx`. This means that the actual transmit request to the underlying layer is always decoupled from the upper layer. In the transmission mode handler cyclic transmission and direct transmissions are processed.

In the following figure the transmission procedure is shown for I-PDUs with direct and periodic transmission mode.



## Periodic Transmission Mode

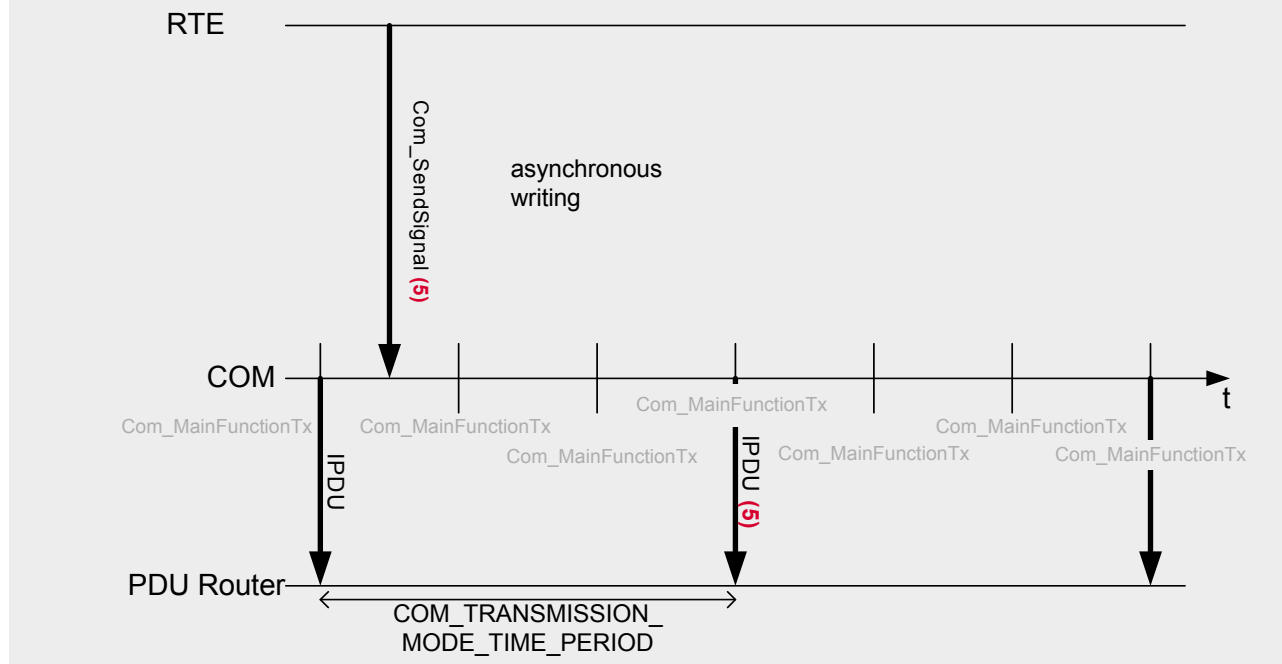


Figure 3-9 Periodic Transmission Mode

## Direct Transmission Mode

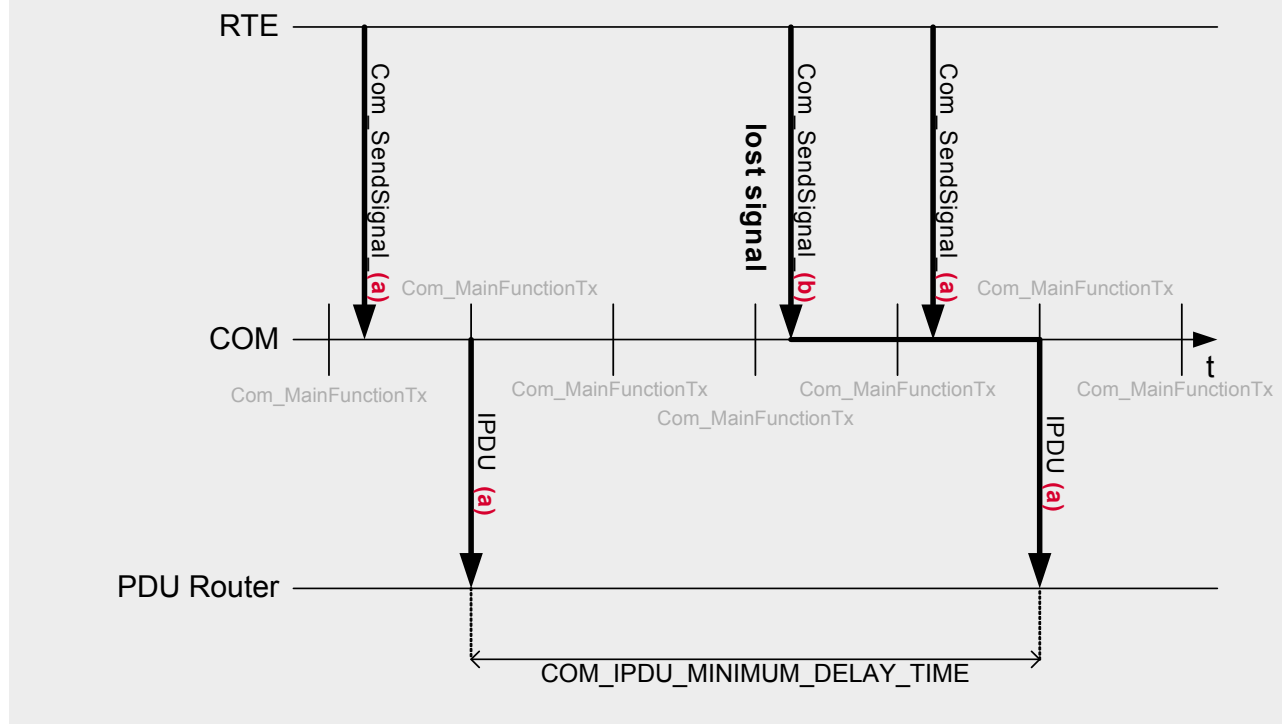


Figure 3-10 Direct Transmission Mode

The mixed transmission mode provides a combination of periodic and direct transmission mode. Note that signal values are not queued by COM. If the upper layer updates signals

with a higher rate as the I-PDU of the signal is transmitted only the most recent signal value is sent.

Transmission requests are queued until `PduR_ComTransmit` returns `E_OK` or the related I-PDU group is stopped (s. 3.3).

### 3.7.1 Transmission of a Signal Group

AUTOSAR COM provides signal groups to send several signals consistently. Signals mapped to a signal group are called group signals and should be in relationship with each other. To ensure the consistency of the group signal values a shadow buffer is provided for each signal group.

To request the transmission of a signal group with several group signals following sequence of API calls must be followed:



#### Example

```
/* Update the group signal values in the shadow buffer */
Com_UpdateShadowSignal(GroupSignal1, &SigBuffer1);
Com_UpdateShadowSignal(GroupSignal2, &SigBuffer2);
/* Copy the shadow buffer to the Tx buffer */
Com_SendSignalGroup(SignalGroupA);
```

For the transmission modes `DIRECT` or `MIXED` the evaluation of the transfer property is handled as follows

- > `ComSignalGroup.ComTransferProperty` **equals** `TRIGGERED`  
and all `ComGroupSignal.ComTransferProperty` **equals** `PENDING`
  - > `Com_UpdateShadowSignal(ComGroupSignal) -> Com_SendSignalGroup(ComSignalGroup)` will trigger an immediate transmission of the Tx I-Pdu regardless of the group signal value.
- > `ComSignalGroup.ComTransferProperty` **equals** `TRIGGERED_ON_CHANGE`  
and all `ComGroupSignal.ComTransferProperty` **equals** `PENDING`
  - > `Com_UpdateShadowSignal(ComGroupSignal) -> Com_SendSignalGroup(ComSignalGroup)` will trigger an immediate transmission of the Tx I-Pdu if at least one group signal value has changed.
- > `ComSignalGroup.ComTransferProperty` **equals** `PENDING`
  - > `ComGroupSignal.ComTransferProperty` **equals** `TRIGGERED`
    - > `Com_UpdateShadowSignal(ComGroupSignal) -> Com_SendSignalGroup(ComSignalGroup)` will trigger an immediate transmission of the Tx I-Pdu regardless of the group signal value.

- > `ComGroupSignal.ComTransferProperty` equals `TRIGGERED_ON_CHANGE`
  - > `Com_UpdateShadowSignal(ComGroupSignal) -> Com_SendSignalGroup(ComSignalGroup)` will trigger an immediate transmission of the Tx I-Pdu if the group signal value has changed.
- > `ComGroupSignal.ComTransferProperty` equals `PENDING`
  - > `Com_UpdateShadowSignal(ComGroupSignal) -> Com_SendSignalGroup(ComSignalGroup)` will not trigger an immediate transmission of the Tx I-Pdu.

**Caution**

To guarantee data consistency of the whole signal group the complete transmission of a signal group (consecutive calls of 'Com\_UpdateShadowSignal' and 'Com\_SendSignalGroup') must not be interrupted by another transmission request for the same signal group or by a call of 'Com\_InvalidateSignalGroup'.

### 3.7.2 Transmission Mode Selector

AUTOSAR COM allows configuring two different transmission modes for each I-PDU (`ComTxModeTrue` and `ComTxModeFalse` - see also chapter 6). The transmission mode of an I-PDU that is valid at a specific point in time is selected using only the filter states (see chapter 3.7.3) of the signals that are mapped to this I-PDU.

If a filter of any signal mapped to a specific I-PDU evaluates to `TRUE` this I-PDU is transmitted with transmission mode `TRUE`. The transmission mode `FALSE` is used for an I-PDU when the filters of all signals mapped to this I-PDU evaluate to `FALSE`.

If all signals mapped to a specific I-PDU have no filter assigned, the transmission mode evaluates to `TRUE` and does never change.

The transmission mode is changed as a result of a call of `Com_SendSignal` or `Com_SendSignalGroup`. The value of the signal or signal group that caused the change is already transmitted with the new transmission mode.

By a transmission mode switch to the Direct/N-times transmission mode an immediate, respecting the minimum delay time, direct/ n-times transmission to the underlying layer will be initiated, even if the transmission mode switch was triggered by a signal with `PENDING` transfer property.

By a transmission mode switch to the Cyclic or Mixed transmission mode the new cycle will start with an immediate transmission request to the underlying layers respecting the minimum delay time.

If the current transmission mode is configured to `NONE`, the COM will never initiate a transmission to the underlying layer.

### 3.7.3 Transmit Signal Filters

A signal filter can be optionally assigned to each transmit signal. The filter of a transmit signal is only used for transmission mode selection (see chapter 3.7.2) but the value of a transmit signal is never filtered out.

The following filters are supported:

■ F_Always	(TRUE)
■ F_Never	(FALSE)
■ F_MaskedNewDiffersMaskedOld	$((new\_value \& mask) \neq (old\_value \& mask))$
■ F_MaskedNewEqualsX	$((new\_value \& mask) == x)$
■ F_MaskedNewDiffersX	$((new\_value \& mask) \neq x)$
■ F_MaskedNewIsOutside	$((new\_value < min) \vee (max < new\_value))$
■ F_MaskedNewIsWithin	$((min \leq new\_value) \wedge (new\_value \leq max))$

The values for *mask*, *x*, *min* and *max* can be configured for each filter in GENy.

### 3.7.4 Minimum Send Distance of an I-PDU

In the COM specification an optional mechanism is defined to achieve a delimitation of the bus load by introducing a minimum send distance for an I-PDU. This concept is also handled in the Tx main function.

In the following figure an example for the mixed transmission mode is shown. Note that due to the minimum send distance cyclic transmissions can be delayed, however the base cycle is not modified. Direct transmissions are drawn by solid red arrows.

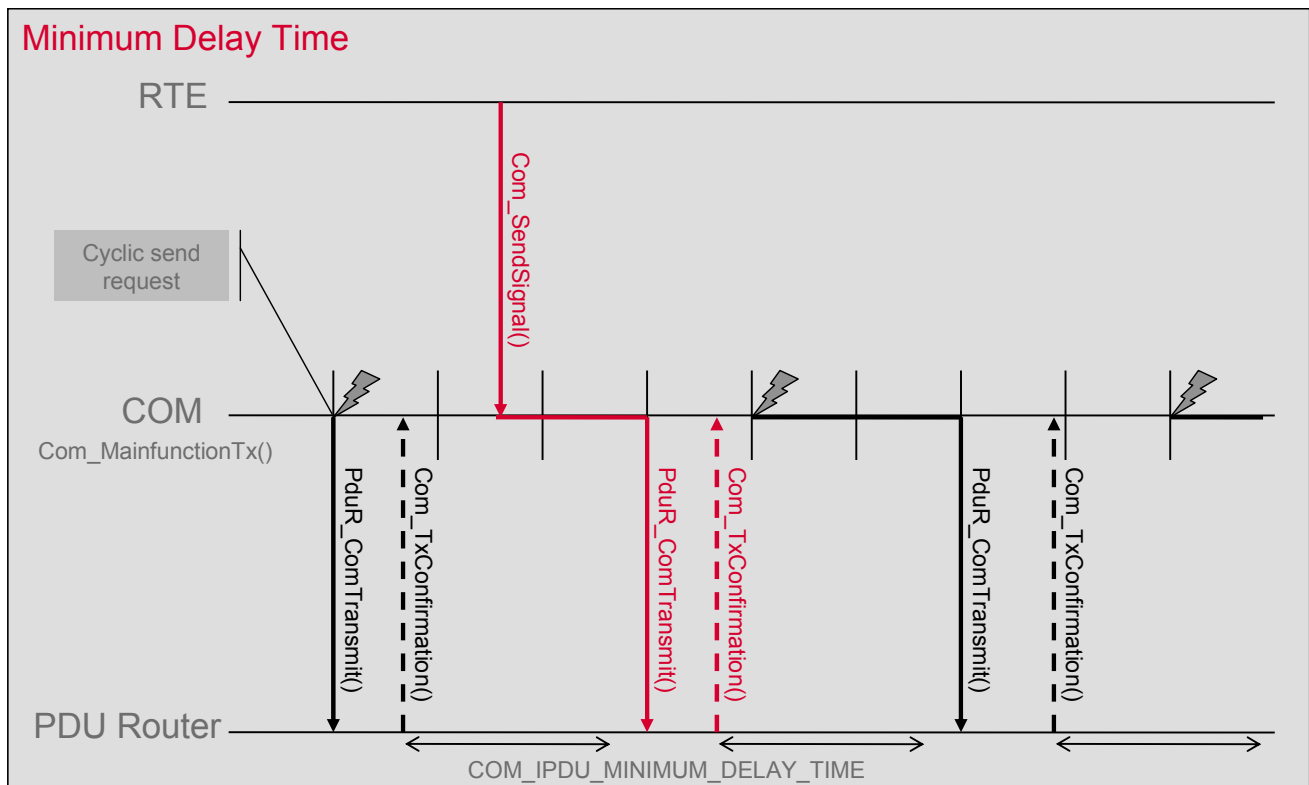


Figure 3-11 Minimum Delay Time

The handling of the minimum send distance is based on the evaluation of the confirmation by the underlying layer.

### 3.7.5 Transmission Deadline Monitoring

For Tx I-PDUs a deadline monitoring mechanism is provided to detect failures in the transmission mechanism of the lower layers.

Two different variants are supported:

- > Variant 1: If the COM triggers the transmission of the I-PDU the time between the transmission request (`PduR_ComTransmit()`) and the next Tx confirmation (`Com_TxConfirmation()`) is observed.
- > Variant 2: For I-PDUs triggered by a schedule table of a bus interface (e.g. LIN schedule table) the time between two consecutive Tx confirmations (`Com_TxConfirmation()`) is observed.

The transmission deadline monitoring Variant 1 is illustrated in Figure 3-12 for a cyclic transmission. Each time COM triggers the transmission of the I-PDU the timeout timer is started. If the timeout time (configuration parameter `ComTimeoutFactor`) is expired before the next Tx confirmation is received the timeout notification function is called.

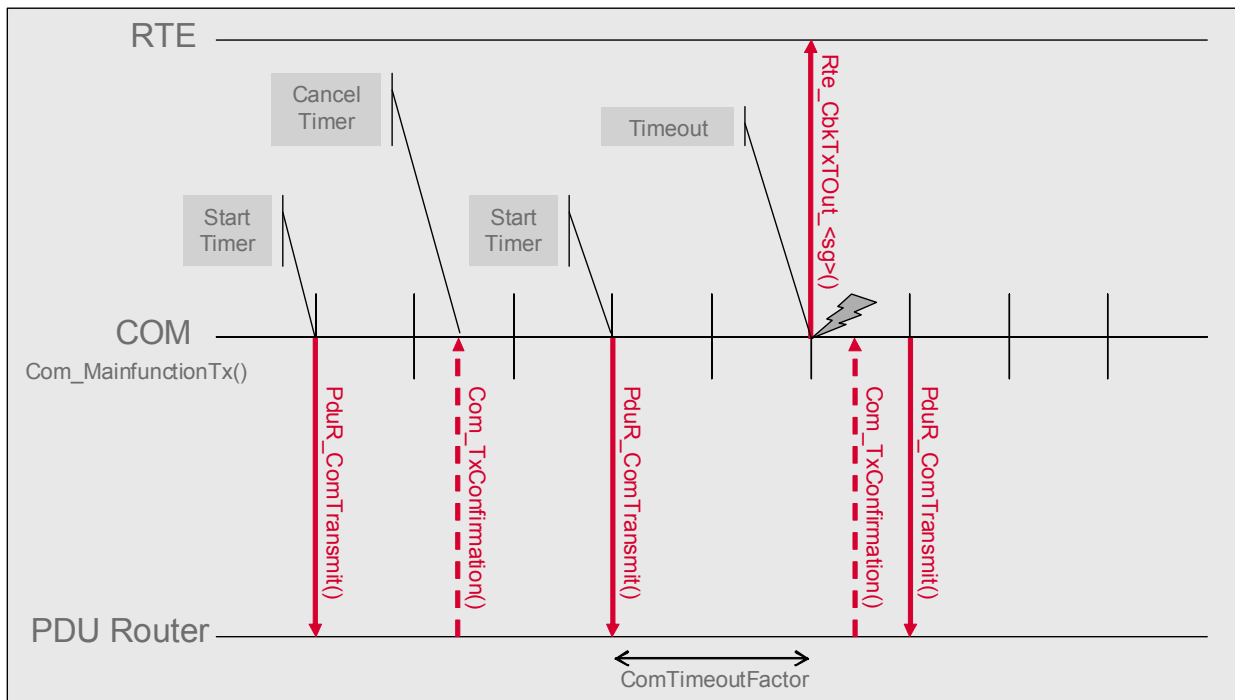


Figure 3-12 Transmission Deadline Monitoring - Variant 1 - Cyclic Transmission

Figure 3-13 illustrates transmission deadline monitoring Variant 1 for a direct transmission. If a transmission request is triggered before the next Tx confirmation is received, the timeout timer is not restarted but the former timeout monitoring is continued.

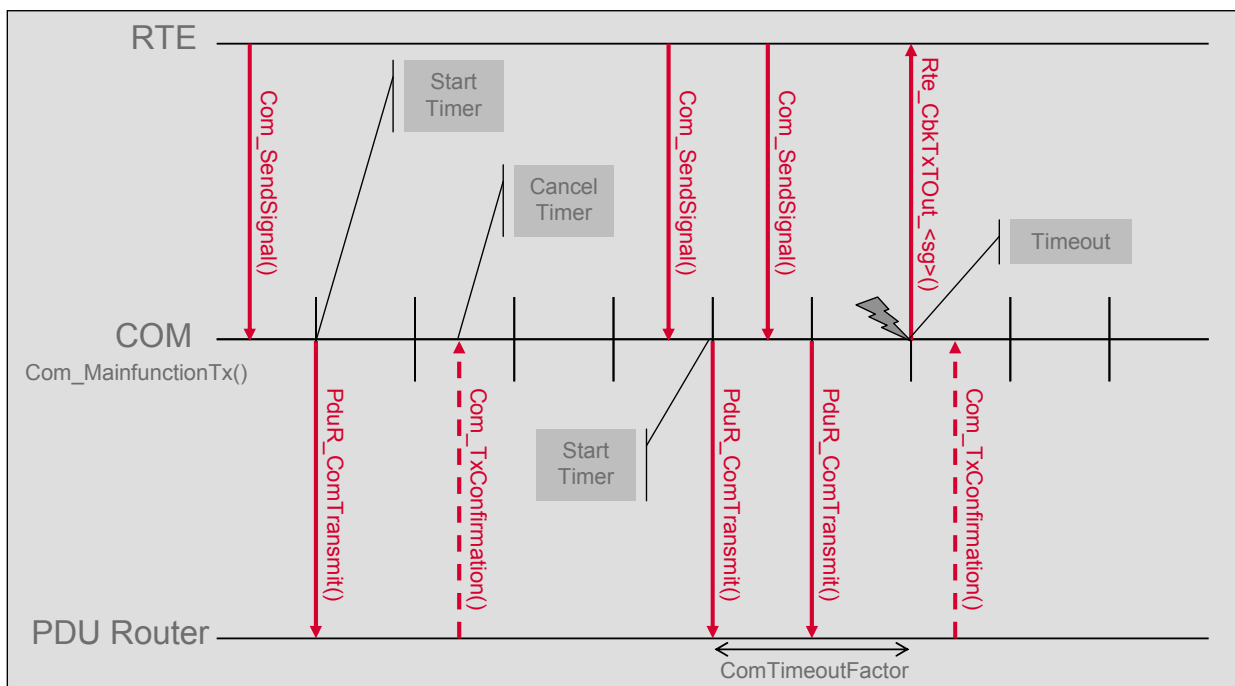


Figure 3-13 Transmission Deadline Monitoring - Variant 1 - Direct Transmission

In transmission deadline monitoring Variant 2 the timeout timers are initially started by a call to `Com_IPduGroupStart()` and are reseted each time a Tx confirmation is received. Figure 3-14 illustrates this behavior.

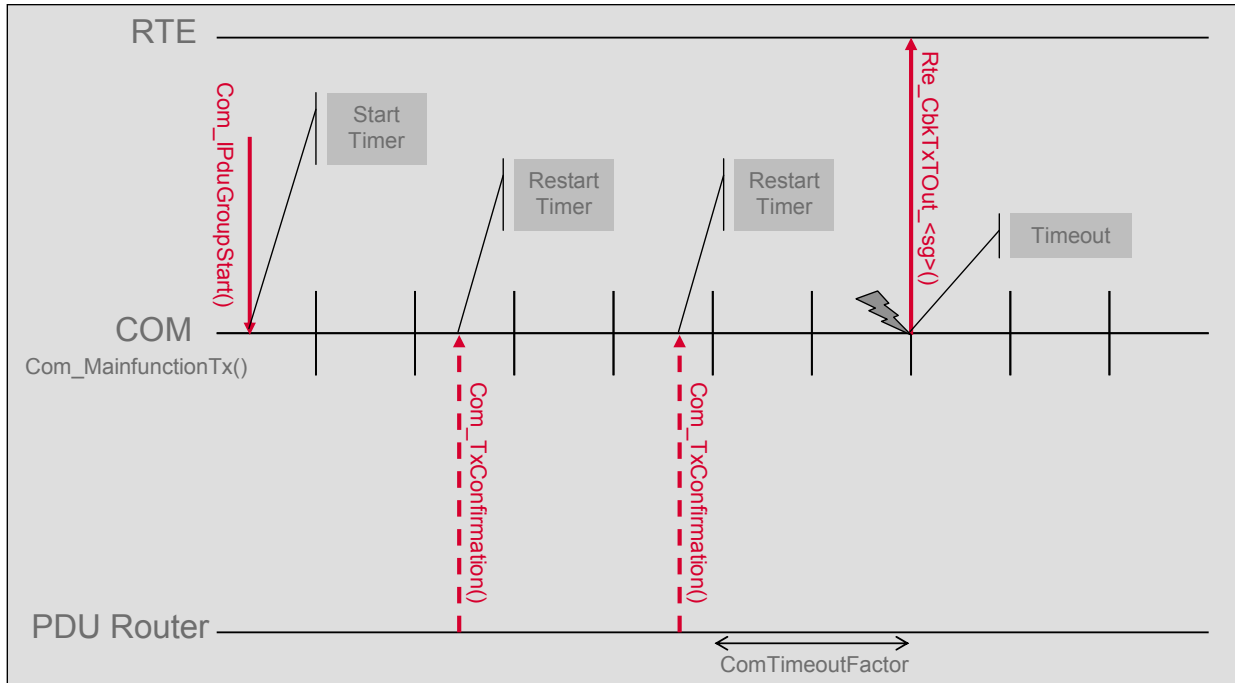


Figure 3-14 Transmission Deadline Monitoring - Variant 2

### 3.8 Reception of a Signal

To receive a signal the upper layer uses the API `Com_ReceiveSignal`. This service delivers the signal value which is contained in the latest I-PDU of the signal.

The reception procedure of the signal is usually asynchronous to the reception of the I-PDU. It is however possible to call `Com_ReceiveSignal` in the reception notification callback.

## Reception of a periodic signal

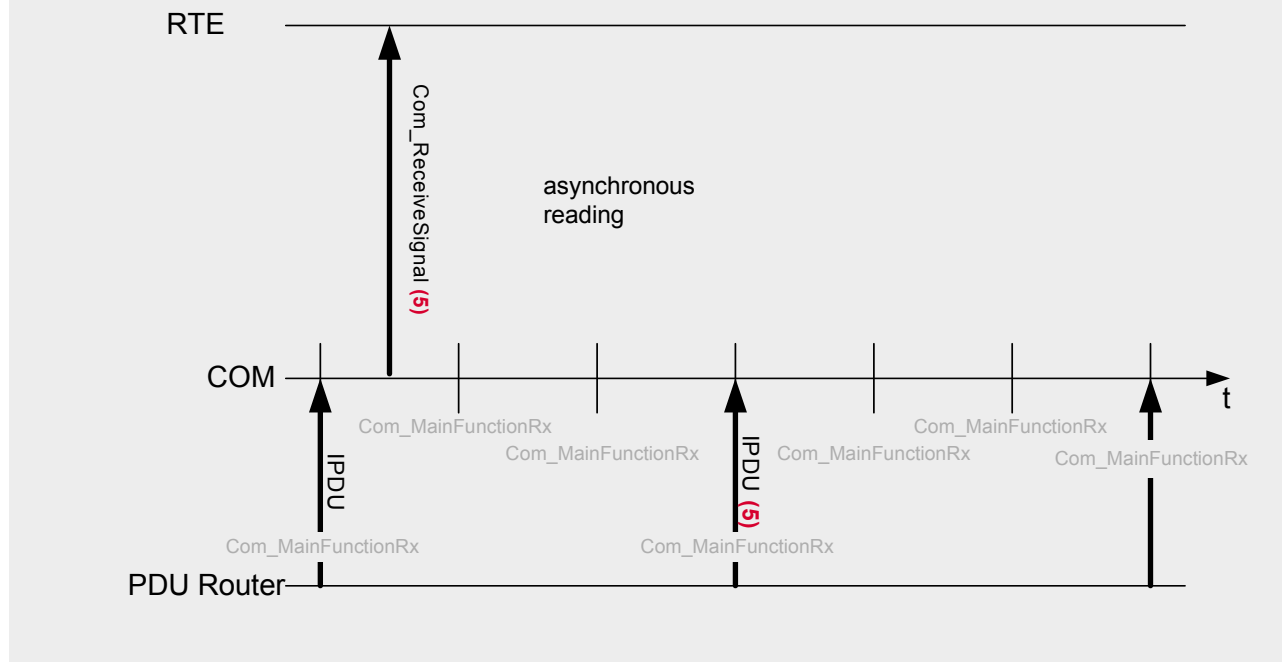


Figure 3-15 Reception of a periodic signal

A call to `Com_ReceiveSignal` always returns the last received signal value or the timeout value if a timeout occurred. This is independent of the usage of update bits.

### 3.8.1 Reception of a Signal Group

AUTOSAR COM provides signal groups to receive several signals consistently. Signals mapped to a signal group are called group signals and should be in relationship with each other. To ensure the consistency of the group signal values a shadow buffer is provided for each signal group.

To receive the values of a signal group with several group signals following sequence of API calls must be followed:



#### Example

```
/* Copy the Rx buffer to the shadow buffer */
Com_ReceiveSignalGroup(SignalGroupA);

/* Get the group signal values from the shadow buffer */
Com_ReceiveShadowSignal(GroupSignal1, &SigBuffer1);
Com_ReceiveShadowSignal(GroupSignal2, &SigBuffer2);
```



**Caution**

To guarantee data consistency of the whole signal group the complete reception of a signal group (consecutive calls of 'Com\_ReceiveSignalGroup' and 'Com\_ReceiveShadowSignal') must not be interrupted by another reception request for the same signal group.

### 3.8.2 Reception Deadline Monitoring

For Rx I-PDUs a deadline monitoring mechanism is provided. It is used to detect failures of other ECUs. In some cases this mechanism is also replaced by a CAN bus bus off handling.

Rx deadline monitoring can either be based on the reception of the I-PDU of the signal or on the reception of an update bit which is attached to the signal. The usage of update bits allows for a signal based timeout monitoring; however a timer is needed for each signal under supervision.

If a signal is not received for a configurable time a configurable timeout indication callback can be invoked.

The start of the timeout monitoring can be deferred by using the first timeout time to avoid timeout events in ECUs in the startup time of a network.

### 3.8.3 Reception of Invalid Signal Values

For each signal an action can be configured to handle the reception of the configurable invalid value.

If the action is set to REPLACE, the received invalid value is replaced by the configured initial value of this signal.

If the action is set to NOTIFY a configurable invalid notification callback is called immediately. Any indication notification callback that might also be configured for the same signal is not called for this reception. A call to `Com_ReceiveSignal` will return the last received valid value for this signal.

### 3.8.4 Dynamic DLC

If the dynamic DLC support is enabled, the Vector COM evaluates the actual received DLC of the SDU given from the lower layer interface.

Two cases are distinguished:

- > Actual received DLC is greater than or equal to the statically configured
  - > The statically configured PDU length is copied from the SDU payload data.
  - > Normal signal processing.
- > Actual received DLC is smaller than the statically configured
  - > Only the actual received PDU length is copied from the SDU payload data.

- > Only completely received signals or signal groups are processed.  
This affects:
  - > Rx indication callbacks
  - > Rx indication flags
  - > Signal routing

**Caution**

You must not read signal or signal group values without a corresponding Rx indication if the dynamic DLC support is enabled. Otherwise data consistency is not guaranteed.

**Info**

If a signal or signal group is completely received depends on the following parameters:

- > Bit position and length
- > Endianness
- > Updatebit position
- > The position of all group signals of a signal group

### 3.9 Signal Gateway

The signal gateway allows routing of signals and signal groups from an Rx I-PDU to one or several Tx I-PDU(s). To reduce interrupt runtime, signal routing is executed on task level within `Com_MainFunctionRouteSignals()`.

As signals can be accessed individually by COM, it is possible to change the signal layout of I-PDUs while routing. Further more it is possible to change the byte alignment of the routed signals and to specify any available transmission property for the Tx signal and I-PDU.

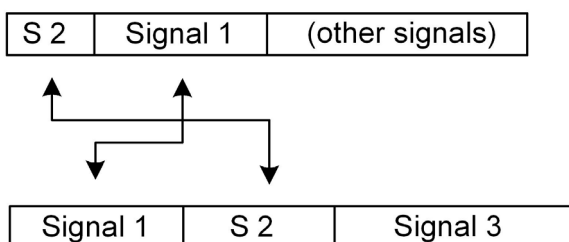


Figure 3-16 Signal routing allows routing of individual signals

The number of routes as well as the property of each routing relation can be changed at post-build time if the COM variant allows post-build configuration.

### 3.9.1 Signal routing requirements

In order to allow routing between two signals several requirements must be fulfilled regarding the compatibility of the Rx and the Tx signal:

- > Application data type must be equal
- > Rx signal bit count must not be larger as the Tx signal bit count
- > When routing byte arrays (application data type is uint[8]) the byte count must be equal
- > A Tx signal must not be used in multiple routing relations (n(Rx):1(Tx) routing is not allowed).

### 3.9.2 Routing of signal groups

Signal groups are routed consistently from the Rx I-PDU to the Tx I-PDU. In order to allow data consistency of the signals, it is required that all signals of the Tx signal group are filled with signals of one Rx signal group. If this is not given, the signal group routing is not possible.

It is not required that all signals of an Rx signal group are routed to a Tx signal group. This allows to route an Rx signal group to a Tx signal group with less group signals.

### 3.9.3 Routing of signals with update bits

The COM signal gateway supports routing of signals with update bits. Thereby it is possible that both, Rx and Tx signal have an update bit or that only one of them has an update bit.

If an Rx signal or signal group has an update bit, signal routing is only executed if the update bit is set. If a Tx signal or signal group has an update bit this will be set when the signal is routed.

### 3.9.4 Adding and removing gateway signals

This feature is only relevant for post-build configurable gateway configurations.

When adding a new ECU to a bus, it might become necessary to provide further signals on that bus and to route these signals from a neighbor bus.

This requires that adding of signal at post-build time is possible. As the application (i.e. RTE) does not support dynamic signal handles that are changeable at post-build time, COM provides a mechanism to add and remove certain signals and signal groups at post-build time. When using this mechanism, the signal handle space that is used by and accessible to the application does not change when adding and removing gateway only signals (marked as "Internal") at post-build time.

### 3.9.5 Routing latency

The maximum routing latency is influenced by several factors and cannot be guaranteed by COM. When estimating the routing latency the following factors have to be taken into account:

- > the cycle times of COM (main) functions

- > the minimum delay time of the Tx I-PDU
- > the Tx I-PDU transmission mode and cycle time
- > the Tx signal transfer mode and filter settings
- > delays caused by lower layers (such as bus access delay times)
- > other factors such as interrupt events that can delay routing execution

**Example**

Eliminating all these factors to the fastest possible configuration (minimum delay time is zero, Tx I-PDU send mode is DIRECT, Tx signal transfer property is 'TRIGGERED' ...) the maximum routing latency is the sum of the following two cycle times:

**Call cycle of Com\_MainFunctionRouteSignals():** This task polls a internal COM flag that is set in the `Com_PdurRxIndication()`. If the flag is set the signal is copied to the destination I-PDU and the transmission request flag of the Tx I-PDU is set.

**Call cycle of Com\_MainFunctionTx():** This function is used to control the transmission of I-PDUs. The function polls the transmission request flags and triggers the I-PDU transmission by calling `PduR_Transmit()` of the related Tx I-PDU.

## 4 Integration

This chapter gives necessary information for the integration of the MICROSAR COM into an application environment of an ECU.

### 4.1 Scope of Delivery

The delivery of the COM contains the files which are described in the chapters 4.1.1 and 4.1.2:

#### 4.1.1 Static Files

File Name	Description
Com.c	This is the source file of COM if ordered as source code.
Com.h	This is the header file of COM.
Com.lib	This is the library of COM if ordered. The extension of the library may change dependent on the used compiler.

Table 4-1 Static files

#### 4.1.2 Dynamic Files

The dynamic files are generated by the configuration tool [config tool].

File Name	Description
Com_Cfg.h	This is the generated header file of COM containing pre-compile switches and providing symbolic defines.
Com_Cbk.h	This is the generated header file of COM containing prototypes for lower layers.
Com_Cfg.c	This is the generated source file of COM with pre-compile-time configurable parameters
Com_Lcfg.c	This is the generated source file of COM with link-time configurable parameters
Com_PBcfg.c	This is the generated source file of COM with post-build-time configurable parameters
Rte_Cbk.h	This is the header file of COM callbacks provided for the RTE. This file is generated by GENy if no RTE is in the delivered package and the Application has to include the file to use callbacks of COM.

Table 4-2 Generated files

## 4.2 Include Structure

The following figure shows the default AUTOSAR include structure. The include structure varies dependent on the used stage of extension of COM.

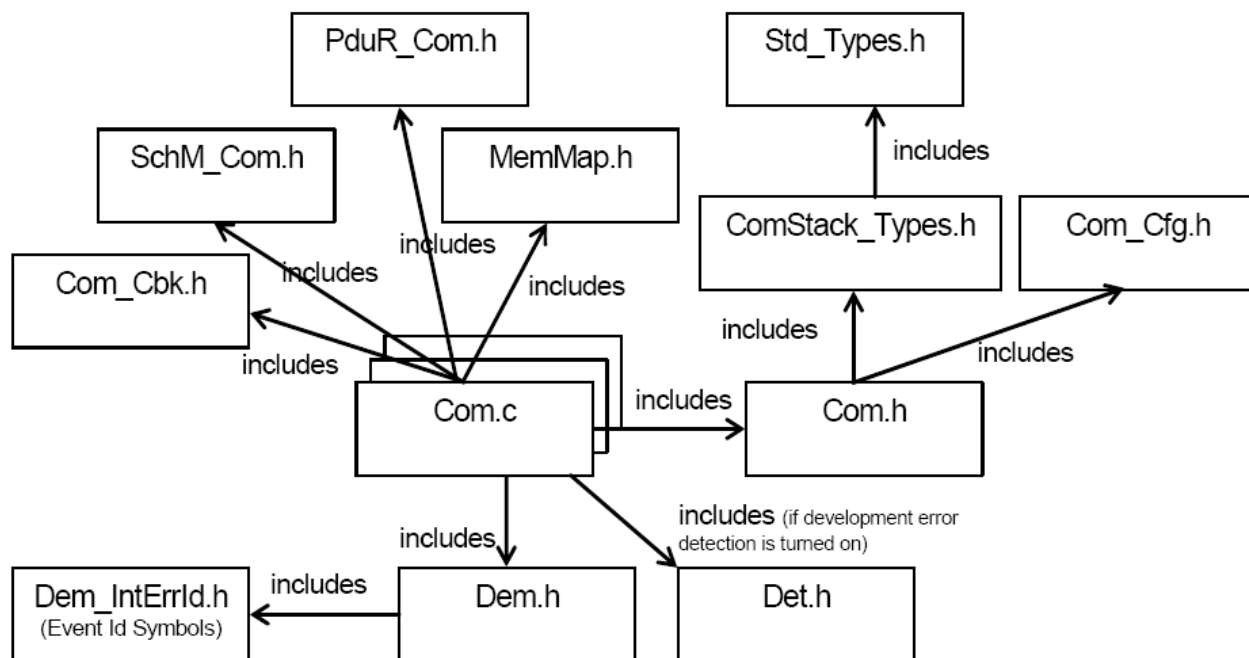


Figure 4-1 Include structure

## 4.3 Compiler Abstraction and Memory Mapping

The objects (e.g. variables, functions, constants) are declared by compiler independent definitions – the compiler abstraction definitions. Each compiler abstraction definition is assigned to a memory section.

The following table contains the memory section names and the compiler abstraction definitions defined for the COM and illustrate their assignment among each other.

Compiler Abstraction Definitions								
	COM_CONST	COM_PBCFG	COM_CODE	COM_APPL_CODE	COM_APPL_DATA	COM_APPL_VAR	COM_VAR_NOINIT	COM_VAR_INIT
Memory Mapping Sections								
COM_START_SEC_CODE			■					
COM_STOP_SEC_CODE								
COM_START_SEC_APPL_CODE				■	■	■		
COM_STOP_SEC_APPL_CODE								

COM_START_SEC_CONST_32BIT	■				■			
COM_STOP_SEC_CONST_32BIT								
COM_START_SEC_CONST_UNSPECIFIED	■				■			
COM_STOP_SEC_CONST_UNSPECIFIED								
COM_START_SEC_PBCFG_ROOT		■						
COM_STOP_SEC_PBCFG_ROOT								
COM_START_SEC_PBCFG		■			■			
COM_STOP_SEC_PBCFG								
COM_START_SEC_VAR_INIT_UNSPECIFIED					■	■		■
COM_STOP_SEC_VAR_INIT_UNSPECIFIED								
COM_START_SEC_VAR_NOINIT_BOOLEAN					■	■	■	
COM_STOP_SEC_VAR_NOINIT_BOOLEAN								
COM_START_SEC_VAR_NOINIT_8BIT					■	■	■	
COM_STOP_SEC_VAR_NOINIT_8BIT								
COM_START_SEC_VAR_NOINIT_16BIT					■	■	■	
COM_STOP_SEC_VAR_NOINIT_16BIT								
COM_START_SEC_VAR_NOINIT_UNSPECIFIED					■	■	■	
COM_STOP_SEC_VAR_NOINIT_UNSPECIFIED								

Table 4-3 Compiler abstraction and memory mapping

## 4.4 Critical Sections

The AUTOSAR standard provides with the BSW Scheduler a BSW module, which handles entering and leaving critical sections. The Vector MICROSAR COM supports additional solutions to handle these sections.

Critical sections are supported by the following two alternatives:

- > the BSW Scheduler
- > the Vector Standard Library (VStdLib)

### 4.4.1 BSW Scheduler

If you use the BSW Scheduler in your AUTOSAR stack to handle critical sections, the MICROSAR COM offers three groups of critical sections to optimize the runtime consumption.

These critical sections can be configured in the template file `_SchM_ComS.h` of the BSW Scheduler. Define the critical sections you want to use to `SCHM_EA_SUSPENDALLINTERRUPTS` and those which are covered by other critical sections or which are not used to `SCHM_COM_EA_INTERRUPTSLOCKEDBYDIFFERENTEA`.

Following critical sections are offered

> Common critical section

The critical section `COM_EXCLUSIVE_AREA_0` has to lock global interrupts to protect common critical sections.

> Tx main function

One of the critical sections `COM_EXCLUSIVE_AREA_1` or `COM_EXCLUSIVE_AREA_2` has to be used, if the function `Com_MainFunctionTx()` can be interrupted by the function `Com_TxConfirmation()`.

> `COM_EXCLUSIVE_AREA_1`

Use this critical section to lock the interrupts once per iteration over all Tx I-Pdus.

This leads to less switches of the interrupt lock but to more runtime with locked interrupts.

> `COM_EXCLUSIVE_AREA_2`

Use this critical section to lock and release the interrupts at each step while iterating over all Tx I-Pdus.

This leads to more switches of the interrupt lock but to less runtime with locked interrupts.



**Caution**

You must NOT use `COM_EXCLUSIVE_AREA_1` and `COM_EXCLUSIVE_AREA_2` at the same time.



**Example**

The critical sections for the Tx main function are used as follows:

```
SchM_Enter_Com(COM_EXCLUSIVE_AREA_1);  
for (iterate over all Tx I-Pdus)  
{  
    SchM_Enter_Com(COM_EXCLUSIVE_AREA_2);  
    /* Critical section */  
    SchM_Exit_Com(COM_EXCLUSIVE_AREA_2);  
}  
SchM_Exit_Com (COM_EXCLUSIVE_AREA_1);
```



> Rx main function

One of the critical sections `COM_EXCLUSIVE_AREA_3` or `COM_EXCLUSIVE_AREA_4` has to be used, if the function `Com_MainFunctionRx()` can be interrupted by the function `Com_RxIndication()`.

> `COM_EXCLUSIVE_AREA_3`

Use this critical section to lock the interrupts once per iteration over all Rx I-Pdus.

This leads to less switches of the interrupt lock but to more runtime with locked interrupts.

> `COM_EXCLUSIVE_AREA_4`

Use this critical section to lock and release the interrupts at each step while iterating over all Rx I-Pdus.

This leads to more switches of the interrupt lock but to less runtime with locked interrupts.



**Caution**

You must NOT use `COM_EXCLUSIVE_AREA_3` and `COM_EXCLUSIVE_AREA_4` at the same time.



**Caution**

If you use `COM_EXCLUSIVE_AREA_3` in conjunction with configured Rx notification functions or invalid value notification functions, you must ensure, that no OS functions are called within these notification functions.

**Example**

The critical sections for the Rx main function are used as follows:

```
SchM_Enter_Com(COM_EXCLUSIVE_AREA_3);  
for (iterate over all Rx I-Pdus)  
{  
    SchM_Enter_Com(COM_EXCLUSIVE_AREA_4);  
    /* Critical section */  
    SchM_Exit_Com(COM_EXCLUSIVE_AREA_4);  
    /* call notification functions */  
}  
SchM_Exit_Com (COM_EXCLUSIVE_AREA_3);
```

#### 4.4.2 Vector Standard Library

If you use the Vector Standard Library (VStdLib) all critical section are forwarded to the function `VstdSuspendAllInterrupts()` which locks the interrupts globally.

#### 4.5 Operating System Requirements

To guarantee data consistency COM uses callbacks to the basic software scheduler or to the Vector standard library ('VStdLib'). In both cases it is possible to map the interrupt disabling functions to OS services. If neither OS nor the VStdLib services are used the integrator has to provide compatible synchronization mechanisms.

Since 8-bit micro controllers are out of scope in AUTOSAR, COM has been optimized for the usage on 16- and 32-bit controllers. Therefore the target system must be able to provide atomic read and write accesses to 16-bit variables.

## 5 API Description

For an interfaces overview please see Figure 2-2.

### 5.1 Type Definitions

The types defined by the COM are described in this chapter.

Type Name	C-Type	Description	Value Range
Com_StatusType	enum	This is a status value returned by the API service Com_GetStatus().	COM_UNINIT The AUTOSAR COM module is not initialized or not usable.
			COM_INIT The AUTOSAR COM Module is initialized and usable.
Com_SignalIdType	c-type	AUTOSAR COM signal object identifier.	0..<SignalIdmax> Zero-based integer number
Com_SignalGroupIdType	c-type	AUTOSAR COM signal group object identifier.	0..<SignalGroupIdmax> Zero-based integer number
Com_PduGroupIdType	c-type	AUTOSAR COM signal group object identifier.	0..<PduGroupIdmax> Zero-based integer number
Com_SerciveldType	enum	Unique identifier of an AUTOSAR COM service.	See in Com.h
Com_LMgt_FatalErrorType	enum	Unique identifier of an AUTOSAR COM service.	See in Com.h

Table 5-1 Type definitions

## 5.2 Services provided by COM

The COM API consists of services, which are realized by function calls.

### 5.2.1 Com\_InitMemory

Prototype	
<code>void <b>Com_InitMemory</b> (void)</code>	
Parameter	
void	none
Return code	
void	none
Functional Description	
The function initialises variables, which cannot be initialised with the startup code.	
Particularities and Limitations	
The function is used by the application.	
Call Context	
The function must be called on task level.	

Table 5-2 Com\_InitMemory

### 5.2.2 Com\_Init


Prototype	
<code>void <b>Com_Init</b> (const Com_ConfigType *config)</code>	
Parameter	
config	Pointer to the COM configuration data, if COM_CONFIG_VARIANT 3 is configured.
Return code	
void	none
Functional Description	
This service initializes internal and external interfaces and variables of the AUTOSAR COM layer for the further processing. After calling this function the inter-ECU communication is still disabled.	
Particularities and Limitations	
The function is used by the Ecu State Manager	
	<b>Caution</b> Com_Init shall not pre-empt any COM function. The rest of the system must guarantee that Com_Init is not called in such a way.
Call Context	
The function must be called on task level and has not to interrupted by other administrative function calls.	

Table 5-3 Com\_Init

### 5.2.3 Com\_DeInit



Prototype	
void <b>Com_DeInit</b> (void)	
Parameter	
void	none
Return code	
void	none
Functional 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 ComDeInit COM is put into an not initialized state.	
Particularities and Limitations	
The function is used by the application.	
	<b>Caution</b> Com_DeInit shall not pre-empt any COM function. The rest of the system must guarantee that Com_DeInit is not called in such a way.
Call Context	
The function must be called on task level and has not to interrupted by other administrative function calls.	

Table 5-4 Com\_DeInit

### 5.2.4 Com\_IpduGroupStart

Prototype	
void <b>Com_IpduGroupStart</b> (Com_PduGroupIdType IpduGroupId, boolean Initialize)	
Parameter	
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
Return code	
void	none
Functional 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.	
Particularities and Limitations	
The function is used by ComM (Ccl_AsrComM, AUTOSAR Communication Manager)	
	<b>Caution</b> Note that this function is not only called by the ComM but also from other modules e.g. diagnosis.
Call Context	

The function must be called on task level and has not to interrupted by other Com\_IpduGroupStart and Com\_IpduGroupStop calls.

Table 5-5 Com\_IpduGroupStart

### 5.2.5 Com\_IpduGroupStop


Prototype	
void <b>Com_IpduGroupStop</b> (Com_PduGroupIdType IpduGroupId)	
Parameter	
IpduGroupId	ID of I-PDU group to be stopped
Return code	
void	none
Functional Description	
Stops a preconfigured I-PDU group. For example, cyclic I-PDUs will be stopped after the call of Com_IpduGroupStop().	
Particularities and Limitations	
The function is used by ComM (Ccl_AsrComM, AUTOSAR Communication Manager)	
	<p><b>Caution</b></p> <p>A call to Com_IpduGroupStop shall not be interrupted by another call to Com_IpduGroupStop or a call to Com_IpduGroupStart. Note that this function is not only called by the ComM but also from other modules e.g. diagnosis.</p>
Call Context	
The function must be called on task level.	

Table 5-6 Com\_IpduGroupStop

### 5.2.6 Com\_EnableReceptionDM

Prototype	
void <b>Com_EnableReceptionDM</b> (Com_PduGroupIdType IpduGroupId)	
Parameter	
IpduGroupId	ID of I-PDU group where reception DM shall be enabled.
Return code	
void	none
Functional Description	
Enables the reception deadline monitoring for the I-PDUs within the given I-PDU group.	
Particularities and Limitations	
The function is used by the <Bus>Sm (AUTOSAR <Bus> State Manager)	
Call Context	
The function must be called on task level and has not to interrupted by other Com_EnableReceptionDM and Com_DisableReceptionDM calls.	

Table 5-7 Com\_EnableReceptionDM

### 5.2.7 Com\_DisableReceptionDM

Prototype	
void <b>Com_DisableReceptionDM</b> (Com_PduGroupIdType IpduGroupId)	
Parameter	
IpduGroupId	ID of I-PDU group where reception DM shall be disabled.
Return code	
void	none
Functional Description	
Disables the reception deadline monitoring for the I-PDUs within the given I-PDU group.	
Particularities and Limitations	
The function is used by the <Bus>Sm (AUTOSAR <Bus> State Manager)	
Call Context	
The function must be called on task level and has not to interrupted by other Com_EnableReceptionDM and Com_DisableReceptionDM calls.	

Table 5-8 Com\_DisableReceptionDM

### 5.2.8 Com\_GetStatus

Prototype	
Com_StatusType <b>Com_GetStatus</b> (void)	
Parameter	
void	none
Return code	
Com_StatusType	Com_StatusType
Functional Description	
Returns the status of the AUTOSAR COM module.	
Particularities and Limitations	
none	
Call Context	
The function can be called on interrupt and task level.	

Table 5-9 Com\_GetStatus

### 5.2.9 Com\_GetConfigurationId

Prototype	
uint32 <b>Com_GetConfigurationId</b> (void)	

Parameter	
void	none
Return code	
uint32	uint32
Functional Description	
Provides the unique identifier of the configuration.	
Particularities and Limitations	
none	
Call Context	
The function can be called on interrupt and task level.	

Table 5-10 Com\_GetConfigurationId

### 5.2.10 Com\_GetConfigurationStringPtr

Prototype	
<code>Com_CharTypePtr Com_GetConfigurationStringPtr (void)</code>	
Parameter	
void	none
Return code	
Com_CharTypePtr	Com_CharTypePtr
Functional Description	
Provides a pointer to an configuration string identifier. The length is delimited by "\0".	
Particularities and Limitations	
none	
Call Context	
The function can be called on interrupt and task level.	

Table 5-11 Com\_GetConfigurationStringPtr

### 5.2.11 Com\_GetVersionInfo

Prototype	
<code>void Com_GetVersionInfo (Std_VersionInfoType *versioninfo)</code>	
Parameter	
versioninfo	Pointer to where to store the version information of this module.
Return code	
void	none
Functional Description	
Returns the version information of this module.	



Particularities and Limitations
none
Call Context
The function can be called on interrupt and task level.

Table 5-12 Com\_GetVersionInfo

### 5.2.12 Com\_MainFunctionRx

Prototype	
void <b>Com_MainFunctionRx</b> (void)	
Parameter	
void	none
Return code	
void	none
Functional Description	
This function shall perform the processing of the AUTOSAR COM receive processing that are not directly initiated by the calls from the RTE and PDU-R. A call to Com_MainFunctionRx returns simply if COM was not previously initialized with a call to Com_Init.	
Particularities and Limitations	
The function is called by the BSW Scheduler.	
Call Context	
The function must be called on task level.	

Table 5-13 Com\_MainFunctionRx

### 5.2.13 Com\_MainFunctionTx

Prototype	
void <b>Com_MainFunctionTx</b> (void)	
Parameter	
void	none
Return code	
void	none
Functional Description	
This function shall perform the processing of the AUTOSAR COM transmission activities that are not directly initiated by the calls from the RTE and PDU-R. A call to Com_MainFunctionTx returns simply if COM was not previously initialized with a call to Com_Init.	
Particularities and Limitations	
The function is called by the BSW Scheduler.	
Call Context	
The function must be called on task level.	

Table 5-14 Com\_MainFunctionTx

### 5.2.14 Com\_MainFunctionRouteSignals



Prototype	
void <b>Com_MainFunctionRouteSignals</b> (void)	
Parameter	
void	none
Return code	
void	None
Functional Description	
Calls the signal gateway part of COM to forward received signals to be routed. The insertion of this call is necessary for decoupling receive interrupts and signal gateway tasks. A call to Com_MainFunctionRouteSignals returns simply if COM was not previously initialized with a call to Com_Init.	
Particularities and Limitations	
The function is called by the BSW Scheduler.	
	<b>Caution</b> The time between to consecutive calls (perhaps the related task/thread cycle) affects directly the signal gateway latency.
Call Context	
The function must be called on task level.	

Table 5-15 Com\_MainFunctionRouteSignals

	<b>Caution</b> AUTOSAR COM requires the usage of a void* parameter for the signal data. Therefore this interface is not type-safe by definition. If the configured signal type does not match the variable used for storing signal values in the application wrong memory could be accessed. Please make sure that the sizes of configured signal types and the corresponding variables in the upper layer match.
---	--

### 5.2.15 Com\_ReceiveSignal

Prototype	
uint8 <b>Com_ReceiveSignal</b> (Com_SignalIdType SignalId, void *SignalDataPtr)	
Parameter	
SignalId	Id of signal to be received.
SignalDataPtr	Reference to the signal data in which to store the received data.

Return code	
uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
Functional Description	
The service Com_ReceiveSignal updates the signal referenced by SignalDataPtr with the data in the signal object identified by SignalId.	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level.	

Table 5-16 Com\_ReceiveSignal

### 5.2.16 Com\_ReceiveShadowSignal

Prototype	
void <b>Com_ReceiveShadowSignal</b> (Com_SignalIdType SignalId, void *SignalDataPtr)	
Parameter	
SignalId	Id of group signal to be received.
SignalDataPtr	Reference to the group signal data in which to store the received data.
Return code	
void	void
Functional Description	
The service Com_ReceiveShadowSignal updates the group signal which is referenced by SignalDataPtr with the data in the shadow buffer. The data in the shadow buffer should be updated before the call of Com_ReceiveShadowSignal by a call of the service Com_ReceiveSignalGroup.	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level.	
To guarantee data consistency of the whole signal group the complete reception of a signal group (consecutive calls of 'Com_ReceiveSignalGroup' and 'Com_ReceiveShadowSignal') must not be interrupted by another reception request for the same signal group.	

Table 5-17 Com\_ReceiveShadowSignal


### 5.2.17 Com\_ReceiveSignalGroup

Prototype	
uint8	<b>Com_ReceiveSignalGroup</b> (Com_SignalGroupIdType SignalGroupId)

Parameter	
SignalGroupId	Id of signal group to be received.
Return code	
uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
Functional Description	
The service Com_ReceiveSignalGroup copies 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 upper layer by a call of Com_ReceiveShadowSignal.	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level.	

Table 5-18 Com\_ReceiveSignalGroup

## 5.2.18 Com\_SendSignal

Prototype	
uint8 <b>Com_SendSignal</b> (Com_SignalIdType SignalId, const void *SignalDataPtr)	
Parameter	
SignalId	ID of signal to be sent.
SignalDataPtr	Reference to the signal data to be transmitted.
Return code	
uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
Functional Description	
The service Com_SendSignal updates the signal object identified by SignalId with the signal referenced by the SignalDataPtr parameter. If the signal has the Triggered transfer property, the update is followed by immediate transmission of the I-PDU associated with the signal except when the signal is packed into an I-PDU with Periodic transmission mode; in this case, no transmission is initiated by the call to this service. If the signal has the Pending transfer property, no transmission is caused by the update.	
Particularities and Limitations	
The function is called by the upper layer.	
	<b>Caution</b>
	<p>If the method is used on a microcontroller like the Tms320 DSP and the datatype uint8 is unsigned short the 8 MSB bits of the variable shall never be set.</p> <p>If the method is used on a microcontroller like the S12X and the datatype is uint16, sint16, uint32 or sint32 the SignalDataPtr must be word aligned.</p>
Call Context	

The function can be called on interrupt and task level and has not to be interrupted by other Com\_SendSignal and Com\_InvalidateSignal calls for the same SignalId.

Table 5-19 Com\_SendSignal

### 5.2.19 Com\_SendSignalGroup

Prototype	
uint8 <b>Com_SendSignalGroup</b> (Com_SignalGroupIdType SignalGroupId)	
Parameter	
SignalGroupId	ID of signal group to be send.
Return code	
uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
Functional Description	
The service Com_SendSignalGroup copies the content of the associated shadow buffer to the associated I-PDU. Prior to this call, all group signals should be updated in the shadow buffer by the call of Com_UpdateShadowSignal.	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level and has not to be interrupted by other Com_SendSignalGroup calls for the same SignalGroupId.	

Table 5-20 Com\_SendSignalGroup

### 5.2.20 Com\_UpdateShadowSignal

Prototype	
void <b>Com_UpdateShadowSignal</b> (Com_SignalIdType SignalId, const void *SignalDataPtr)	
Parameter	
SignalId	ID of group signal to be updated.
SignalDataPtr	Reference to the group signal data to be updated.
Return code	
void	void
Functional Description	
The service Com_UpdateShadowSignal updates a group signal with the data, referenced by SignalDataPtr. The update of the group signal data is done in the shadow buffer, not in the I-PDU. To send out the shadow buffer, Com_SendSignalGroup has to be called. Sign extension and byte swapping are performed as the group signal is inserted into the shadow buffer.	
Particularities and Limitations	
The function is called by the upper layer.	

**Caution**

If the method is used on a microcontroller like the S12X and the datatype is uint16, sint16, uint32 or sint32 the SignalDataPtr must be word aligned.

**Call Context**

The function can be called on interrupt and task level.

To guarantee data consistency of the whole signal group the complete transmission of a signal group (consecutive calls of 'Com\_UpdateShadowSignal' and 'Com\_SendSignalGroup') must not be interrupted by another transmission request for the same signal group or by a call of 'Com\_InvalidateSignalGroup'.

Table 5-21 Com\_UpdateShadowSignal

**5.2.21 Com\_TriggerIPDUSend****Prototype**

```
void Com_TriggerIPDUSend (PduIdType ComTxPduId)
```

**Parameter**

ComTxPduId	ID of AUTOSAR COM Tx I-PDU.
------------	-----------------------------

**Return code**

void	void
------	------

**Functional Description**

By a call to Com\_TriggerIPDUSend the I-PDU with the given ID is triggered for transmission.

**Particularities and Limitations**

The function shall only be called from within an I-PDU callout.

**Call Context**

The function can be called on interrupt and task level.

Table 5-22 Com\_TriggerIPDUSend

**5.2.22 Com\_InvalidateSignal****Prototype**

```
uint8 Com_InvalidateSignal (Com_SignalIdType SignalId)
```

**Parameter**

SignalId	ID of signal to be invalidated.
----------	---------------------------------

**Return code**

uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
-------	---

**Functional Description**

Sender side AUTOSAR Software Component indicates via the RTE to AUTOSAR COM that it is not able to provide a valid value for the corresponding signal (e.g. sensor is faulty). After invalidation of the current signal data a Com\_SendSignal is performed internally.

Particularities and Limitations
The function is called by the upper layer.
Call Context
The function can be called on interrupt and task level and has not to be interrupted by other Com_SendSignal and Com_InvalidateSignal calls for the same SignalId.

Table 5-23 Com\_InvalidateSignal

### 5.2.23 Com\_InvalidateSignalGroup

Prototype	
uint8 <b>Com_InvalidateSignalGroup</b> (Com_SignalGroupIdType SignalGroupId)	
Parameter	
SignalGroupId	ID of signal group to be invalidated.
Return code	
uint8	uint8 E_OK service has been accepted COM_SERVICE_NOT_AVAILABLE corresponding I-PDU group was stopped (or service failed due to development error)
Functional Description	
Sender side AUTOSAR Software Component indicates via the RTE to AUTOSAR COM that it is not able to provide a valid value for the corresponding signal group. (e.g. sensor is faulty). After invalidation of the current signal group data a Com_SendSignalGroup is performed internally.	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level and has not to be interrupted by other Com_InvalidateSignalGroup calls for the same SignalGroupId and by Com_UpdateShadowSignal calls for a SignalId which is contained in the SignalGroupId.	

Table 5-24 Com\_InvalidateSignalGroup

## 5.3 Services used by COM

In the following table services provided by other components, which are used by the COM are listed. For details about prototype and functionality refer to the documentation of the providing component.

Component	API
PDUR	PduR_ComTransmit
DET	Det_ReportError
DEM	Dem_ReportErrorStatus
RTE/ Application	I-PDU Callout
RTE/ Application	Rx ComSignal indication notification
RTE/ Application	Rx ComSignal timeout notification

Component	API
RTE/ Application	Tx ComSignal indication notification
EcuM [5]	EcuM_GeneratorCompatibilityError

Table 5-25 Services used by the COM

**Info**

In AUTOSAR 3.0.1 the COM SWS specifies no production errors. Therefore COM will never call the DEM independent of the configuration.

## 5.4 Callback Functions

This chapter describes the callback functions that are implemented by the COM and can be invoked by other modules. The prototypes of the callback functions are provided in the header file Com\_Cbk.h by the COM.

### 5.4.1 Com\_RxIndication

Prototype	
<pre>void <b>Com_RxIndication</b> (PduIdType ComRxPduId, COM_RXIND_TYPE COM_RXIND_PARA)</pre>	
Parameter	
ComRxPduId	ID of AUTOSAR COM I-PDU that has been received. Identifies the data that has been received. Range: 0..(maximum number of I-PDU IDs received by AUTOSAR COM) - 1
COM_RXIND_PARA	<p>SduPtr Pointer to the received I-PDU data.</p> <p>This Parameter is used, if COM_USE_PDUINFOTYPE is defined to STD_OFF.</p> <p>PduInfoPtr Payload information of the received I-PDU (pointer to data and data length).</p> <p>This Parameter is used, if COM_USE_PDUINFOTYPE is defined to STD_ON.</p>
Return code	
void	none
Functional Description	
This function is called by the lower layer after an I-PDU has been received.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 5-26 Com\_RxIndication



## 5.4.2 Com\_TriggerTransmit

Prototype	
COM_TT_RETURN <b>Com_TriggerTransmit</b> (PduIdType ComTxPduId, COM_TT_TYPE COM_TT_PARA)	
Parameter	
ComTxPduId	ID of AUTOSAR COM I-PDU that is requested to be transmitted by AUTOSAR COM.
COM_TT_PARA	<p>SduPtr Pointer to the received I-PDU data.</p> <p>This Parameter is used, if COM_USE_PDUINFOTYPE is defined to STD_OFF.</p> <p>PduInfoPtr Payload information of the received I-PDU (pointer to data and data length).</p> <p>This Parameter is used, if COM_USE_PDUINFOTYPE is defined to STD_ON.</p>
Return code	
COM_TT_RETURN	<p>none No return value.</p> <p>This return value is used, if COM_USE_PDUINFOTYPE is defined to STD_OFF.</p> <p>Std_ReturnType</p> <p>This return value is used, if COM_USE_PDUINFOTYPE is defined to STD_ON.</p> <p>E_OK: The SDU has been copied and the SduLength indicates the number of copied bytes.</p> <p>E_NOT_OK: The SDU has not been copied and the SduLength has not been set.</p>
Functional Description	
<p>This function is called by the lower layer when an AUTOSAR COM I-PDU shall be transmitted. Within this function, AUTOSAR COM shall copy the contents of its I-PDU transmit buffer to the L-PDU buffer given by SduPtr. Use case: This function is used e.g. by the LIN Master for sending out a LIN frame. In this case, the trigger transmit can be initiated by the Master schedule table itself or a received LIN header. This function is also used by the FlexRay Interface for requesting PDUs to be sent in static part (synchronous to the FlexRay global time). Once the I-PDU has been successfully sent by the lower layer (PDU-Router), the lower layer must call Com_TxConfirmation().</p>	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 5-27 Com\_TriggerTransmit

## 5.4.3 Com\_TxConfirmation

Prototype
void <b>Com_TxConfirmation</b> (PduIdType ComTxPduId)

Parameter	
ComTxPduld	ID of AUTOSAR COM I-PDU that has been transmitted. Range: 0..(maximum number of I-PDU IDs transmitted by AUTOSAR COM) - 1
Return code	
void	none
Functional Description	
This function is called by the lower layer after the PDU has been transmitted on the network. A confirmation that is received for an I-PDU that does not require a confirmation is silently discarded.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326)). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 5-28 Com\_TxConfirmation

## 5.5 Configurable Interfaces

### 5.5.1 Notifications

At its configurable interfaces the COM defines notifications that can be mapped to callback functions provided by other modules. The mapping is not statically defined by the COM but can be performed at configuration time. The function prototypes that can be used for the configuration have to match the appropriate function prototype signatures, which are described in the following sub-chapters.

#### 5.5.1.1 Indication Notification

Prototype	
void [ <b>Indication Notification Name</b> ] ( void )	
Parameter	
void	none
Return code	
void	none
Functional Description	
The notification function is called after the message has been received successfully. The function can be configured in GENy for signals and signal groups with a configurable function name.	
Particularities and Limitations	
> none	
Call Context	
> The call context depends on the configuration of the parameter ComIPduSignalProcessing for the I-PDU.	

Table 5-29 Indication Notification

### 5.5.1.2 Confirmation Notification

Prototype	
void [ <b>Confirmation Notification Name</b> ] ( void )	
Parameter	
void	none
Return code	
void	none
Functional Description	
The notification function is called after successful transmission of the I-PDU containing the message. The function can be configured in GENy for signals and signal groups with a configurable function name.	
Particularities and Limitations	
> none	
Call Context	
> The call context depends on the configuration of the parameter ComIPduSignalProcessing for the I-PDU.	

Table 5-30 Confirmation Notification

### 5.5.1.3 Rx Timeout Notification

Prototype	
void [ <b>Timeout Notification Name</b> ] ( void )	
Parameter	
void	none
Return code	
void	none
Functional Description	
It is called immediately after a message reception error has been detected by the deadline monitoring mechanism. The function can be configured in GENy for signals and signal groups with a configurable function name.	
Particularities and Limitations	
> none	
Call Context	
> The function is called on task level.	

Table 5-31 Timeout Notification

## 5.5.2 Callout Functions

At its configurable interfaces the COM defines callout functions. The declarations of the callout functions are provided by the BSW module, i.e. the COM. It is the integrator's task to provide the corresponding function definitions. The definitions of the callouts can be adjusted to the system's needs. The COM callout function declarations are described in the following tables:

### 5.5.2.1 I-PDU Callout

As the Vector COM provides the possibility to use the lower layer APIs in different variants, two callout function declarations are used.

> Deactivated attribute 'Com -> Miscellaneous -> Addons -> Use Pdu Info Type'

#### Rx I-Pdu callout function

Prototype	
<pre>FUNC(boolean, COM_APPL_CODE) &lt;IPDU_CalloutName&gt;(PduIdType ComRxPduId, P2CONST(uint8, AUTOMATIC, COM_APPL_DATA) SduPtr)</pre>	
Parameter	
ComRxPduId	<p>ID of AUTOSAR COM I-PDU that has been received. Identifies the data that has been received.</p> <p>Range: 0..(maximum number of I-PDU IDs received by AUTOSAR COM) - 1</p>
SduPtr	Pointer to received SDU (payload buffer)
Return code	
boolean	The return value indicates whether the processing of this I-PDU shall continue (TRUE) or abandon (FALSE) after the callout returns.
Functional Description	
For each I-PDU a callout function can be configured and the implementation has to be provided by the application. It can be used to extend the COM functionality with application related functions (e.g. CRC or sequence counter calculation).	
Particularities and Limitations	
<ul style="list-style-type: none"> <li>Other COM APIs than Com_TriggerIPDUSend shall not be called.</li> </ul>	
Call Context	
<ul style="list-style-type: none"> <li>Rx I-PDU Callouts are called by COM directly after Com_RxIndication on task or interrupt level. The call context depends on the configuration of the Driver.</li> </ul>	

Table 5-32 Rx I-PDU Callout with SDU pointer

#### Tx I-Pdu callout function

Prototype	
<pre>FUNC(boolean, COM_APPL_CODE) &lt;IPDU_CalloutName&gt;(PduIdType ComTxPduId, P2VAR(uint8, AUTOMATIC, COM_VAR_NOINIT) SduPtr)</pre>	

Parameter	
ComTxPduId	ID of AUTOSAR COM I-PDU that should be transmitted. Identifies the data that should be transmitted.  Range: 0..(maximum number of I-PDU IDs transmitted by AUTOSAR COM) - 1
SduPtr	Pointer to transmitted SDU (payload buffer)
Return code	
boolean	The return value indicates whether the processing of this I-PDU shall continue (TRUE) or abandon (FALSE) after the callout returns.
Functional Description	
For each I-PDU a callout function can be configured and the implementation has to be provided by the application. It can be used to extend the COM functionality with application related functions (e.g. CRC or sequence counter calculation).	
Particularities and Limitations	
<ul style="list-style-type: none"> <li>Other COM APIs than Com_TriggerIPDUSend shall not be called.</li> </ul>	
Call Context	
<ul style="list-style-type: none"> <li>Tx I-PDU Callouts are called by COM directly before the call of PduR_ComTransmit on task level and in the call context of Com_TriggerTransmit. In the second case the call context depends on the configuration of the lower layer calling the API.</li> </ul>	

Table 5-33 Tx I-PDU Callout with SDU pointer

### > Activated attribute 'Com -> Miscellaneous -> Addons -> Use Pdu Info Type'

#### Rx I-Pdu callout function

Prototype	
<pre>FUNC(boolean, COM_APPL_CODE) &lt;IPDU_CalloutName&gt;(PduIdType ComRxPduId, CONSTP2CONST(PduInfoType, AUTOMATIC, COM_APPL_DATA) PduInfoPtr)</pre>	
Parameter	
ComRxPduId	ID of AUTOSAR COM I-PDU that has been received. Identifies the data that has been received.  Range: 0..(maximum number of I-PDU IDs received by AUTOSAR COM) - 1
PduInfoPtr	Contains the length (SduLength) of the received I-PDU and a pointer to a buffer (SduDataPtr) containing the I-PDU.
Return code	
boolean	The return value indicates whether the processing of this I-PDU shall continue (TRUE) or abandon (FALSE) after the callout returns.
Functional Description	
For each I-PDU a callout function can be configured and the implementation has to be provided by the application. It can be used to extend the COM functionality with application related functions (e.g. CRC or sequence counter calculation).	

### Particularities and Limitations

- Other COM APIs than Com\_TriggerIPDUSend shall not be called.

### Call Context

- Rx I-PDU Callouts are called by COM directly after Com\_RxIndication on task or interrupt level. The call context depends on the configuration of the Driver.

Table 5-34 Rx I-PDU Callout with PduInfo pointer

## Tx I-Pdu callout function

### Prototype

```
FUNC(boolean, COM_APPL_CODE) <IPDU_CalloutName>(PduIdType
ComTxPduId, CONSTP2VAR(PduInfoType, AUTOMATIC, AUTOMATIC)
PduInfoPtr)
```

### Parameter

ComTxPduId	ID of AUTOSAR COM I-PDU that should be transmitted. Identifies the data that should be transmitted.  Range: 0..(maximum number of I-PDU IDs transmitted by AUTOSAR COM) - 1
PduInfoPtr	Contains the length (SduLength) of the I-PDU to be transmitted and a pointer to a buffer (SduDataPtr) containing the I-PDU.

### Return code

boolean	The return value indicates whether the processing of this I-PDU shall continue (TRUE) or abandon (FALSE) after the callout returns.
---------	---

### Functional Description

For each I-PDU a callout function can be configured and the implementation has to be provided by the application. It can be used to extend the COM functionality with application related functions (e.g. CRC or sequence counter calculation).

### Particularities and Limitations

- Other COM APIs than Com\_TriggerIPDUSend shall not be called.

### Call Context

- Tx I-PDU Callouts are called by COM directly before the call of PduR\_ComTransmit on task level and in the call context of Com\_TriggerTransmit. In the second case the call context depends on the configuration of the lower layer calling the API.

Table 5-35 Tx I-PDU Callout with PduInfo pointer

## 6 Configuration

In the COM the attributes can be configured with the following methods:

- > Configuration in GENy for a detailed description see
- > Configuration in different database formats, for a detailed description see 6.1 and following chapters
  - DBC (CANdb) for CAN
  - LDF for LIN
  - XML (FIBEX) for FlexRay
  - EcuC for AUTOSAR

The COM layer supports the Configuration Conformance Class 3 (CCC3) which supports a post-build configuration process.

The first step however is to apply the pre-compile configuration when the COM library is built. In case of a library delivery this step is performed by Vector and sufficient information about the pre-compile settings and compiler flags have to be provided. If source code is delivered the integrator can build the COM library.

When the library is finished it can be configured by link-time configuration, i.e. a generated C source file is linked to the COM library to resolve external symbols required for configuration.

In a post-build process the configuration tool is able generate a hex file instead of a C file. The hex file can be flashed directly into an existing ECU. It must be located at the same place where configuration data for the library was located when the system was linked. A more detailed description of the post-build process can be found in Technical Reference [4].

## 6.1 Configuration in Dbc Data Base

This chapter describes the configuration of COM with database attributes of the dbc format for CAN.



### Info

Same database attributes are used as for the Vector Interaction Layer for reasons of backwards compatibility. Please refer to the technical reference of the Vector Interaction Layer for a detailed specification.

Some of the settings provided by a database can also be adjusted manually in GENy.



### Caution

Don't mix up the order of enumeration values in the attribute definition. Not the value of the attribute is interpreted but the position of the selected value in the list.

Attribute Name	Object Type	Value Type	Values <small>the default value is written in bold</small>	Description
GenMsgILSupport	Message	Enumeration	No <b>Yes</b>	Determines whether the I-PDU is handled by COM or not (e.g. for messages used by diagnosis, Network Management ...).
GenMsgSendType	Message	Enumeration	Cyclic, NotUsed, NotUsed, NotUsed, Cyclic, NotUsed, IfActive, <b>NoMsgSendType</b>	Message related transmission mode.  Note: The enumeration strings used for the GenMsgSendType are often OEM-specific and can differ from here.
GenSigSendType	Signal	Enumeration	Cyclic, OnWrite, OnWriteWithRepetition, OnChange, OnChangeWithRepetition, IfActive, IfActiveWithRepetition, <b>NoSigSendType</b>	Signal related transmission mode.
GenMsgCycleTime	Message	Integer	<b>0</b> - 65535	Time in ms between each cyclic transmission of a message.
GenMsgCycleTimeFast	Message	Integer	<b>0</b> - 65535	Time in ms between each cyclic transmission of an active message.



Attribute Name	Object Type	Value Type	Values the default value is written in bold	Description
GenMsgNrOfRepetition	Message	Integer	<b>0</b> - 65535	Number of repetitions used if the GenSigSendType is OnChangeWithRepetition or OnWriteWithRepetition for any signal in this message.
GenSigStartValue	Signal	<b>String</b> , Integer, Float	0x00 - 0xfffffffffffff (64bit: sting only)	This Value is the default value for the signal, if Com_Init() or Com_IpduGroupStart with Initialize = true is called.  The string value type can represent hexadecimal and integer values.
GenSigInactiveValue	Signal	<b>String</b> , Integer, Float	0x00 - 0xfffffffffffff (64bit: sting only)	This value determines whether a signal with GenSigSendType IfActive is active or not.  The string value type can represent hexadecimal and integer values.
GenMsgDelayTime	Message	Integer	<b>0</b> - 65535	This is the minimum time in ms between the transmission of I-PDU's with the same identifier.
GenMsgStartDelayTime	Message	Integer	<b>0</b> - 65535	This defines the time in ms between Com_IpduGroupStart() and the first transmission of the cyclic part of this I-PDU.
GenSigFirstTimeoutTime	Node – Mapped Rx Signal	Integer	<b>0</b> - 65535	First timeout time in ms used for this signal received by the Ecu relation configured in CANdb++.  If different GenSigFirstTimeoutTime values are configured for a message, the lowest timeout time (strongest definition) is used for timeout monitoring.
GenSigFirstTimeoutTime_<ECU>	Signal	Integer	<b>0</b> - 65535	First timeout time in ms used for this signal received by <ECU>.  If different GenSigFirstTimeoutTime_<ECU> values are configured for a message, the lowest timeout time (strongest definition) is used for timeout monitoring.
GenSigTimeoutTime	Node – Mapped Rx Signal	Integer	<b>0</b> - 65535	Timeout time in ms used for this signal received by the Ecu relation configured in CANdb++.  If different GenSigTimeoutTime values are configured for a message, the lowest timeout time (strongest definition) is used for timeout monitoring.

Attribute Name	Object Type	Value Type	Values the default value is written in bold	Description
GenSigTimeoutTime_ _<ECU>	Signal	Integer	<b>0</b> - 65535	Timeout time in ms used for this signal received by <ECU>.  If different GenSigTimeoutTime_ _<ECU> values are configured for a message, the lowest timeout time (strongest definition) is used for timeout monitoring.
GenSigMissingSourceValue	Signal	<b>String</b> , Integer	0x00 - 0xffffffff (64bit: sting only)	Signal Gateway only.  This attribute is used for an extension of the AUTOSAR COM specification.  The specified value is used as timeout replacement value for the Tx signal(s) of a routing relation in case of an Rx signal timeout. If the attribute is configured to its default value or if the attribute is not present in a database, no Tx signal timeout handling is carried out.
ILTxTimeout	Channel	Integer	0 – 10000 default: <b>250</b>	Transmission timeout time in ms used for all Tx signals mapped to this channel.

Table 6-1 Dbc Data base attributes

**Caution**

If update bits are specified as signals with a ‘\_UB’ postfix or infix in the DBC file, please set the database attribute GenSigSendType to NoSigSendType for these update bit “signals”.

EcuC Attribute	Dbc (CAN)
Type of the I-PDU	GenMsgILSupport is set to the Yes.
ComIPdu	A message in the DBC format.
ComIPduName	The message name in the DBC format.
ComIPduSize	The DLC of a message in bits.
ComIPduMinimumDelayFactor	The attribute value of the message attribute GenMsgDelayTime is the value. This attribute can also be modified in GENy.
ComBitSize	This is the signal length in bits.

EcuC Attribute	Dbc (CAN)
ComSignalEndianness	<p>If the Byte Order of the signal is Intel, select LITTLE_ENDIAN.</p> <p>Motorola, select BIG_ENDIAN.</p> <p>OPAQUE endianness cannot be configured.</p>
ComBitPosition	<p>This is the bit position defined in the DBC format, where the signal starts.</p>
ComTimeoutFactor	<p>Rx (RECEIVED) signals:</p> <p>One of both attributes exists in the DBC file, never both.</p> <p>GenSigFirstTimeoutMsg: The attribute value of the signal-node mapped attribute GenSigFirstTimeoutMsg is the value.</p> <p>GenSigFirstTimeoutMsg_&lt;ECU&gt;: The attribute value of the signal attribute GenSigFirstTimeoutMsg_&lt;ECU&gt; is the value.</p> <p>Tx (SEND) signals:</p> <p>The attribute value of the channel attribute ILTxTimeout is the value.</p>
ComSignal	<p>This is a signal related to a message of the current node.</p> <p>Identical to ComSignal.</p>
ComSignal shortname	<p>This is the signal name.</p> <p>Identical to ComSignalName.</p>
ComSignalTransferProperty	<p><b>Triggered:</b></p> <p>If GenSigSendType is <i>OnChange</i>, <i>OnChangeWithRepetition</i>, <i>OnWrite</i> or <i>OnWriteWithRepetition</i>.</p> <p><b>Pending:</b></p> <p>If the signal is not <i>Triggered</i>.</p> <p>This attribute can also be modified in GENy.</p>
ComSignalInitValue	<p>The attribute value is the signal attribute GenSigStartValue is the value.</p> <p>If the signal is an update bit signal, the init value is 0 (zero).</p> <p>This attribute can also be modified in GENy.</p>
ComSignalGroup	<p>This is a signal group in the DBC format.</p>
ComSignalGroup shortname	<p>This is the name of the signal group.</p>

EcuC Attribute	Dbc (CAN)
ComSignalGroupTransferProperty	<p><b>Triggered:</b></p> <p>If GenSigSendType is <i>OnChange</i>, <i>OnChangeWithRepetition</i>, <i>OnWrite</i> or <i>OnWriteWithRepetition</i> for any of the signals in the signal group.</p> <p><b>Pending:</b></p> <p>If the signal group is not <i>Triggered</i>.</p> <p>This attribute can also be modified in GENy.</p>
ComFilter	<p><b>F_MaskedNewDiffersMaskedOld:</b></p> <p>If GenSigSendType is <i>OnChange</i> or <i>OnChangeWithRepetition</i> (ComFilterMask is set to 0xFFFFFFFF)</p> <p><b>F_MaskedNewDiffersX:</b></p> <p>If GenSigSendType is <i>IfActive</i> or <i>IfActiveWithRepetition</i> (ComFilterX is set to GenSigInactiveValue)</p> <p><b>None:</b></p> <p>In any other case no filter is configured.</p> <p>This attribute can also be modified in GENy.</p>
ComTransmissionModeTrue /ComTxMode /ComTxModeTimePeriodFactor	<p>The value is defined by the database attribute</p> <p><b>GenMsgCycleTimeFast:</b></p> <p>If GenSigSendType is <i>IfActive</i> or <i>IfActiveWithRepetition</i> for any signal in this I-PDU.</p> <p>or</p> <p>If GenMsgSendType is <i>IfActive</i>.</p> <p><b>GenMsgCycleTime:</b></p> <p>If GenSigSendType is <i>Cyclic</i> for any signal in this I-PDU.</p> <p>or</p> <p>If GenMsgSendType is <i>Cyclic</i>.</p> <p>This attribute can also be modified in GENy.</p>
ComTransmissionModeFalse /ComTxMode /ComTxModeTimePeriodFactor	<p>The value is defined by the database attribute</p> <p><b>GenMsgCycleTime:</b></p> <p>If GenSigSendType is <i>Cyclic</i> for any signal in this I-PDU.</p> <p>or</p> <p>If GenMsgSendType is <i>Cyclic</i>.</p> <p>This attribute can also be modified in GENy.</p>

EcuC Attribute	Dbc (CAN)
ComTransmissionModeTrue /ComTxMode /ComTxModeMode	<p>The DBC attributes GenMsgSendType and GenSigSendType are interpreted:</p> <p><b>Periodic:</b> If GenSigSendType is <i>Cyclic</i>, <i>IfActive</i> or <i>IfActiveWithRepetition</i> for any signal in this I-PDU. or If GenMsgSendType is <i>Cyclic</i> or <i>IfActive</i></p> <p><b>Direct:</b> If GenSigSendType is <i>OnWrite</i>, <i>OnChange</i>, <i>OnWriteWithRepetition</i> or <i>OnChangeWithRepetition</i> for any signal in this I-PDU.</p> <p><b>Mixed:</b> If the I-PDU is <i>Periodic</i> and <i>Direct</i>.</p> <p><b>None:</b> If the I-PDU is neither <i>Periodic</i>, <i>Direct</i> nor <i>Mixed</i>.</p> <p>This attribute can also be modified in GENy.</p>
ComTransmissionModeFalse /ComTxMode /ComTxModeMode	<p>The DBC attributes GenMsgSendType and GenSigSendType are interpreted:</p> <p><b>Periodic:</b> If GenSigSendType is <i>Cyclic</i> for any signal in this I-PDU. or If GenMsgSendType is <i>Cyclic</i>.</p> <p><b>None:</b> If the I-PDU is not <i>Periodic</i>.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeTimeOffsetFactor	<p>The value is defined by the database attribute GenMsgStartDelayTime.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeRepetitionPeriodFactor	<p>The value is defined by the database attribute GenMsgCycleTimeFast.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeNumberOfRepetitions	<p>The value is defined by the database attribute GenMsgNrOfRepetition.</p> <p>This attribute can also be modified in GENy.</p>
ComUpdate	<p>An update bit is a signal in the same message as the other signal with the same signal name extended by the postfix or infix “_UB”.</p>
ComUpdateBitPosition	<p>This is the same value as ComSignal Bit Position of the ComSignal with the dependent name.</p>

Table 6-2 Mapping of AUTOSAR EcuC and Dbc data base attributes

## 6.2 Configuration in Ldf Data Base

This chapter describes the configuration of COM with database attributes of the Ldf format for LIN.

EcuC Attribute	Ldf (LIN)
Type of the I-PDU	All LIN frames, except Diagnostic Frames are I-PDUs.
ComIPdu	A Frame in the LDF format.
ComIPduName	This is the frame_name.
ComIPduSize	This is the frame_size.
ComIPduMinimumDelayFactor	This value is 0. This attribute can also be modified in GENy.
ComBitSize	This is the signal_size in bits.
ComSignalEndianness	The byte order of the signal is always Intel, select LITTLE_ENDIAN.
ComBitPosition	This is the signal start bit offset in bits.
ComFirstTimeoutFactor	This value is 0. This attribute can also be modified in GENy.
ComTimeoutFactor	This value is 0. This attribute can also be modified in GENy.
ComSignal	This is a signal related to a message of the current node. Identical to ComSignalName.
ComSignal shortname	This is the signal name. Identical to ComSignalName.
ComSignalTransferProperty	<b>Triggered:</b> Sporadic and event triggered LIN frames – as they require an explicit transmission trigger. <b>Pending:</b> Unconditional frames as these are sent by Trigger Transmit from LinIf. This attribute can also be modified in GENy.
ComSignalInitValue	This is the <init_value> defined in the Ldf file. This attribute can also be modified in GENy.

EcuC Attribute	Ldf (LIN)
ComSignalGroup	<p>The loader of LDF files supports files that use a naming convention to define signal groups, since LDF format has no means of defining signal groups.</p> <p>The naming convention for signals is:</p> <ul style="list-style-type: none"> <li>a) signal name SignalGroupXXX__SignalYYY produces a signal group SignalGroupXXX and a signal SignalYYY that is part of the group.</li> <li>b) signal name SignalGroupXXX__SignalYYY_ISigZZZ produces a signal group SignalGroupXXX_ISigZZZ and a signal SignalYYY_ISigZZZ that is part of the group.</li> </ul> <p>The first occurrence of __ (double underscore) marks the end of the signal group name and the beginning of the group signal name. The name of the contained signal may be any valid LIN signal name.</p>
ComSignalGroup shortname	This is the name of the signal group.
ComSignalGroupTransferProperty	<p><b>Triggered:</b></p> <p>Sporadic and event triggered LIN frames – as they require an explicit transmission trigger.</p> <p><b>Pending:</b></p> <p>Unconditional frames as these are sent by Trigger Transmit from LinIf.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeTimePeriodFactor	<p>This value is 0.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeMode	<p>This value is None.</p> <p>This attribute can also be modified in GENy.</p>
ComTxModeTimeOffsetFactor	<p>This value is 0.</p> <p>This attribute can also be modified in GENy.</p>
ComUpdate	An update bit is a signal in the same frame as the other signal with the same signal name extended by the postfix or infix “_UB”.
ComUpdateBitPosition	This is the same value as ComBitPosition of the ComSignal with the dependent name.

Table 6-3 Mapping of AUTOSAR EcuC and Ldf data base attributes

### 6.3 Configuration in FIBEX Data Base

This chapter describes the configuration of COM with database attributes of the FIBEX version 1 format for FlexRay.

EcuC Attribute	FIBEX (FlexRay)
Type of the I-PDU	All signal groups, where the “Application Type” of the frame is Application are I-PDUs.
ComIPdu	A signal group in the FIBEX format.
ComIPduName	This is the name of the signal group.
ComIPduSize	This is the length of the signal group.
ComIPduMinimumDelayFactor	This value is 0. This attribute can also be modified in GENy.
ComBitSize	This is the signal length in bits.
ComSignalEndianness	If the Byte Order of the signal is Intel, select LITTLE_ENDIAN. Motorola, select BIG_ENDIAN. OPAQUE endianness cannot be configured.
ComBitPosition	This is the bit position defined in the FIBEX format, where the signal starts decremented by the start bit of the signal group, where the signal is contained.
ComSignalTimeoutFactor	This value is 0. This attribute can also be modified in GENy.
ComSignal	This is a signal related to a frame of the current node. Identical to ComSignal.
ComSignalName	This is the signal name. Identical to ComSignalName.
ComSignalGroupTransferProperty	This value is pending. This attribute can also be modified in GENy.
ComSignalInitValue	This is the Initial Value of the signal. This attribute can also be modified in GENy.
ComSignalGroup	The FIBEX format does define signal groups, but due to this, that signal groups are representing I-PDUs, signal groups are not supported,
ComSignalGroupName	The FIBEX format does define signal groups, but due to this, that signal groups are representing I-PDUs, signal groups are not supported,
ComSignalGroupTransferProperty	The FIBEX format does define signal groups, but due to this, that signal groups are representing I-PDUs, signal groups are not supported, A default value for the attribute may be Pending. This attribute can also be modified in GENy.



EcuC Attribute	FIBEX (FlexRay)
ComTxModeTimePeriodFactor	This value is 0. This attribute can also be modified in GENy.
ComTxModeMode	This value is None. This attribute can also be modified in GENy.
ComTxModeTimeOffsetFactor	This value is 0. This attribute can also be modified in GENy.
ComUpdate	An update bit is a signal in the same signal group as the other signal with the same signal name extended by the postfix or infix “_UB”.
ComUpdateBitPosition	This is the same value as ComBitPosition of the ComSignal with the dependent name.

Table 6-4 Mapping of AUTOSAR EcuC and FIBEX data base attributes

## 6.4 Configuration in EcuC Data Base



### Caution

The ECU configuration file format has the following limitations:

- > The container ComUpdate is only exported if the UpdateBitSupport is enabled.
- > Your delivery contains in the Components directory of GENy the file Com\_bswmd.arxml. This file is the definition of the Com specific ECU configuration file format.
- > The following rules must be taken into account, before the data is imported:
  - > Com doesn't support the ComFilterAlgorithm 'F\_OneEveryN'.
  - > The imported Handles of ComIPduGroups, ComPdus, ComSignals, ComSignalGroups, ComGroupSignals are recalculated before the data is generated.
- > The signal conversion configuration is neither exported nor imported.
- > If the configuration is loaded and created with the EcuC file and the ComSignalType of the signal is UINT8\_N the ComSignalEndianness shall be configured to OPAQUE.

The following chapters contain the detailed description of the EcuC configuration parameters for COM.

### 6.4.1 Com

Container Name	Com
Path	\MICROSAR\Com
Multiplicity	0...1
Description	Configuration of the Com module.

#### 6.4.1.1 ComConfig

Container Name	ComConfig
Path	\MICROSAR\Com\ComConfig
Multiplicity	1...1
Description	This container contains the configuration parameters and sub containers of the COM module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComConfigurationId	1...1	Integer	<b>0</b> - n	This ID provides the unique identifier of the post-build configurable configuration part. At runtime the configuration ID can be retrieved by Com_GetConfigurationId().

Sub Container	Multiplicity
ComGwMapping	0...*
ComIPdu	0...*
ComIPduGroup	0...254
ComSignal	0...*
ComSignalGroup	0...*

### 6.4.1.2 ComGwMapping

Container Name	ComGwMapping
Path	\\MICROSAR\\Com\\ComConfig\\ComGwMapping
Multiplicity	0...*
Description	You can specify a name for each Gw Mapping. The name must be unique as it is used as short name for the ComGwMapping container.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComGenerate	0...1	Boolean	true <b>false</b>	The switch can be used to temporarily deactivate a Gw Mapping relation.

Sub Container	Multiplicity
ComGwDestination	1...*
ComGwSource	1...1

### 6.4.1.3 ComGwDestination

Container Name	ComGwDestination
Path	\\MICROSAR\\Com\\ComConfig\\ComGwMapping\\ComGwDestination
Multiplicity	1...*

<b>Description</b>	Each instance of this choice container allows to define one routing destination either by reference to an already configured COM signal / signal group or by a destination description container.
--------------------	---

Sub Container	Multiplicity
ComGwSignal	0...1

#### 6.4.1.4 ComGwSignal

<b>Container Name</b>	ComGwSignal
<b>Path</b>	\\MICROSAR\\Com\\ComConfig\\ComGwMapping\\ComGwDestination\\ComGwSignal
<b>Multiplicity</b>	0...1
<b>Description</b>	Select the destination signal of the Gw Mapping. If there are several signals with the same name in your databases, the unique name is used as specified by the "NameDecorator" component (i.e. the channel index is added to the signal name).

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComGwSignalRef	1...1	Choice-reference	not specified	Reference to an object of a gateway relation. Either to a ComSignal, ComGroupSignal or to a SignalGroup.

#### 6.4.1.5 ComGwSource

<b>Container Name</b>	ComGwSource
<b>Path</b>	\\MICROSAR\\Com\\ComConfig\\ComGwMapping\\ComGwSource
<b>Multiplicity</b>	1...1
<b>Description</b>	This choice container allows the definition of the gateway source signal either by reference to an already configured COM signal / signal group or by a source description container.

Sub Container	Multiplicity
ComGwSignal	0...1

#### 6.4.1.6 ComGwSignal

<b>Container Name</b>	ComGwSignal
<b>Path</b>	\\MICROSAR\\Com\\ComConfig\\ComGwMapping\\ComGwSource\\ComGwSignal
<b>Multiplicity</b>	0...1
<b>Description</b>	Select the source signal of the Gw Mapping. If there are several signals with the same name in your databases, the unique name is used as specified by the "NameDecorator" component (i.e. the channel index is added to the signal name).

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComGwSignalRef	1...1	Choice-reference	not specified	Reference to an object of a gateway relation. Either to a ComSignal, ComGroupSignal or to a SignalGroup.

#### 6.4.1.7 ComIPdu

<b>Container Name</b>	ComIPdu
<b>Path</b>	\\MICROSAR\\Com\\ComConfig\\ComIPdu
<b>Multiplicity</b>	0...*
<b>Description</b>	<p>Contains the configuration parameters of Com I-Pdus.</p> <p>The shortName is used as the symbolic name (ComIpduName) of this I-Pdu when communicating with the PduR. Is optional because the Com module might be used for internal communication only. This parameter is only stored in the XML file, and must not be used within the implementation.</p>

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComIPduCallout	0...1	Function	not specified	<p>Enter a function name if you require an IPdu callout for this PDU. If the field is left empty no function will be configured. The function will be called on task level inside Com_MainFunctionRx() or Com_MainFunctionTx().</p> <p>The prototype boolean &lt;IPdu Callout Name&gt;(PduIdType ID, uint8* ipduD) is generated to Com_Cbk.h and has to be implemented by the application.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComIPduRxHandleId	0...1	Integer	<b>0</b> - 65535	The numerical value used as the ID of this IPdu. The ComIPduRxHandleId is required by the API calls to receive IPdus from the PDUR.  It is only present for IPdu is received from the PDUR, because Com is the starting module for Tx IPdus and there is no need to define IDs for - IPdus in the Com module.
ComIPduSignalProcessing	0...1	Enumeration	<b>DEFERED</b> <b>IMMEDIATE</b>	This option determines the call context and the point of time when the signal notification function is called.  - IMMEDIATE: The notification function is called within Com_TxConfirmation() or Com_RxIndication(). Depending on the lower layer interface, this might be in interrupt context.  - DEFERRED: The notification function is called on task level during the next call cycle of Com_MainFunctionRx() or Com_MainFunctionTx().
ComIPduSize	1...1	Integer	0 - 8191	Size of the IPdu in bytes.  The maximum size is limited by the underlying communication interface.  0-8 for CAN and LIN 0-254 for FlexRay 0-8191 for Others
ComIPduDirection	1...1	Enumeration	RECEIVE SEND	The direction defines if this I-PDU, and therefore the contributing signals and signal groups, shall be send or received.
ComGenerate	0...1	Boolean	<b>true</b> false	Activate this feature to enable any Com API for the ComPdu and it ComSignals.  Deactivate this feature to disable any Com API for the ComPdu and it ComSignals. This can be used to reduce calculation time in Com, if a PDU is routed completely in the PDUR.
ComIPduGroupRef	1...*	Reference	not specified	Select the ComIPduGroup this ComPdu should be assigned to. Clear the selection to remove the assignment.
ComIPduSignalGroupRef	0...*	Reference	not specified	References to all signal groups contained in this I-Pdu
ComIPduSignalRef	0...*	Reference	not specified	References to all signals contained in this I-PDU.
PduIdRef	1...1	Reference	not specified	Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.

Sub Container	Multiplicity
---------------	--------------

Sub Container	Multiplicity
ComTxIPdu	0...1

#### 6.4.1.8 ComTxIPdu

Container Name	ComTxIPdu
Path	\\MICROSAR\\Com\\ComConfig\\ComIPdu\\ComTxIPdu
Multiplicity	0...1
Description	This container contains additional transmission related configuration parameters of COM I-PDUs

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTxIPduMinimumDelayTimeFactor	0...1	Integer	<b>0</b> - n	This attribute defines the minimum transmit delay between two subsequent messages of the same ID. A minimum delay time of '0' disables the observation. The value must be a multiple of the Com_MainFunctionTx() cycle time.
ComTxIPduUnusedAreasDefault	1...1	Integer	<b>0</b> - 255	The specified value is used to fill gaps between the IPdu signals.
ComTxIPduTimeoutContext	0...1	Enumeration	<b>COM_TRANSMIT</b>  TX_CONFIRMATION	<p>Transmission deadline monitoring context for this Tx I-Pdu.</p> <p>COM_TRANSMIT: observe the time between PduR_ComTransmit() and Com_TxConfirmation()</p> <p>TX_CONFIRMATION: observe the time between two consecutive calls to Com_TxConfirmation()</p> <p>Note: This attribute will be configured automatically depending on the bus type of the Tx I-Pdu</p> <p>CAN: timeout context COM_TRANSMIT</p> <p>LIN: timeout context TX_CONFIRMATION</p> <p>FlexRay: depends on the parameter 'FrIfNoneMode' of the FrTxPdu</p> <p>TRUE: TX_CONFIRMATION</p> <p>FALSE: COM_TRANSMIT</p> <p>RESTRICTIONS: 'ComTxIPduTimeoutContext' is only available, if 'ComTxTimeoutForTxModeNoneSupport' is set to 'true'.</p>

Sub Container	Multiplicity
ComTxModeFalse	0...1
ComTxModeTrue	0...1

#### 6.4.1.9 ComTxModeFalse

Container Name	ComTxModeFalse
Path	\MICROSAR\Com\ComConfig\ComIPdu\ComTxIPdu\ComTxModeFalse
Multiplicity	0...1
Description	This container contains the configuration parameters of COM transmission modes in the case the ComFilter evaluates to false.

Sub Container	Multiplicity
ComTxMode	1...1

#### 6.4.1.10 ComTxMode

Container Name	ComTxMode
Path	\MICROSAR\Com\ComConfig\ComIPdu\ComTxIPdu\ComTxModeFalse\ComTxMode
Multiplicity	1...1
Description	This container contains the configuration parameters of COM transmission modes.

Attribute Name	Multiplicity	Value Type	Values	Description
			Default value is typed bold	



Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTxModeMode	1...1	Enumeration	DIRECT MIXED <b>NONE</b> PERIODIC	<p>The transmission mode determines the Tx properties of the IPdu:</p> <ul style="list-style-type: none"> <li>- DIRECT: The message is transmitted in the next Com_MainFunctionTx() call after writing to a signal.</li> <li>- PERIODIC: The IPdu is transmitted with the specified cycle time (ComTxModeTimePeriodFactor).</li> <li>- MIXED: Combines the transmission modes of CYCLIC and DIRECT.</li> <li>- NONE: COM does not transmit these IPdus. A transmission must be triggered by Com_PduRTriggerTransmit().</li> </ul> <p>All transmission modes, except NONE, maintain the Minimum Delay Time (ComTxIPduMinimumDelayTimeFactor) and delay a transmission if required.</p>
ComTxModeNumberOfRepetitions	0...1	Integer	<b>0</b> - 255	If the transmission mode is DIRECT or MIXED this attribute specifies the number of repetitions for this IPdu. If the value is 0 the transmit request is triggered once but the confirmation is not evaluated.
ComTxModeRepetitionPeriodFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is DIRECT or MIXED this attribute specifies cycle time for the repetitions specified by ComTxModeNumberOfRepetitions.
ComTxModeTimeOffsetFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is PERIODIC or MIXED this attribute specifies the time offset of the first transmission after enabling the IPdu. The value must be a multiple of the Com_MainFunctionTx() cycle time.
ComTxModeTimePeriodFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is PERIODIC or MIXED this attribute specifies the message cycle time. If the transmission mode is NONE, the cycle time is not evaluated. The value must be a multiple of the Com_MainFunctionTx() cycle time.

#### 6.4.1.11 ComTxModeTrue

Container Name	ComTxModeTrue
Path	\\MICROSAR\\Com\\ComConfig\\ComIPdu\\ComTxIPdu\\ComTxModeTrue
Multiplicity	0...1
Description	This container contains the configuration parameters of COM transmission modes in the case the ComFilter evaluates to true.

Sub Container	Multiplicity
ComTxMode	1...1

#### 6.4.1.12 ComTxMode

Container Name	ComTxMode
Path	\MICROSAR\Com\ComConfig\ComIPdu\ComTxIPdu\ComTxModeTrue\ComTxMode
Multiplicity	1...1
Description	This container contains the configuration parameters of COM transmission modes.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTxModeMode	1...1	Enumeration	<b>DIRECT</b> <b>MIXED</b> <b>NONE</b> <b>PERIODIC</b>	<p>The transmission mode determines the Tx properties of the IPdu:</p> <ul style="list-style-type: none"> <li>- <b>DIRECT</b>: The message is transmitted in the next Com_MainFunctionTx() call after writing to a signal.</li> <li>- <b>PERIODIC</b>: The IPdu is transmitted with the specified cycle time (ComTxModeTimePeriodFactor).</li> <li>- <b>MIXED</b>: Combines the transmission modes of CYCLIC and DIRECT.</li> <li>- <b>NONE</b>: COM does not transmit these IPdus. A transmission must be triggered by Com_PduRTriggerTransmit().</li> </ul> <p>All transmission modes, except NONE, maintain the Minimum Delay Time (ComTxIPduMinimumDelayTimeFactor) and delay a transmission if required.</p>
ComTxModeNumberOfRepetitions	0...1	Integer	<b>0</b> - 255	If the transmission mode is DIRECT or MIXED this attribute specifies the number of repetitions for this IPdu. If the value is 0 the transmit request is triggered once but the confirmation is not evaluated.
ComTxModeRepetitionPeriodFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is DIRECT or MIXED this attribute specifies cycle time for the repetitions specified by ComTxModeNumberOfRepetitions.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTxModeTimeOffsetFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is PERIODIC or MIXED this attribute specifies the time offset of the first transmission after enabling the IPdu. The value must be a multiple of the Com_MainFunctionTx() cycle time.
ComTxModeTimePeriodFactor	0...1	Integer	<b>0</b> - 65535	If the transmission mode is PERIODIC or MIXED this attribute specifies the message cycle time. If the transmission mode is NONE, the cycle time is not evaluated. The value must be a multiple of the Com_MainFunctionTx() cycle time.

#### 6.4.1.13 ComIPduGroup

Container Name	ComIPduGroup
Path	\\MICROSAR\\Com\\ComConfig\\ComIPduGroup
Multiplicity	0...254
Description	You can specify a name for each IPdu Group. The name must be unique as it is used as short name for the ComIPduGroup container.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComIPduGroupHandleId	1...1	Integer	0 - 254	The numerical value used as the ID of this IPdu Group. The ComIPduGroupHandleId is required by the API calls to start and stop IPdu Groups.
ComIPduGroupGroupRef	0...1	Reference	not specified	If the I-PDU Group belongs to an I-PDU group, this is the name of the I-PDU group it belongs to. This I-PDU Group does not belong to another I-PDU group, if this reference is omitted.

#### 6.4.1.14 ComSignal

Container Name	ComSignal
Path	\\MICROSAR\\Com\\ComConfig\\ComSignal
Multiplicity	0...*
Description	Contains the configuration parameters of Com signals.  The shortName is used as the symbolic name of the signal (ComSignalName). This name is also used as the handle name for the signal.

This parameter is only stored in the XML file, and must not be used within the implementation.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComBitPosition	1...1	Integer	0 - 65528	Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.  If the signal is a Motorola signal this indicates the MSB.  If the byte order is a Intel signal this attribute indicates the LSB.
ComBitSize	1...1	Integer	0 - 65528	Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.
ComDataInvalidAction	0...1	Enumeration	<b>NONE</b> <b>NOTIFY</b> <b>REPLACE</b>	This parameter defines the action performed upon reception of an invalid signal. If Replace is used the ComSignalInitValue will be used for the replacement.
ComErrorNotification	0...1	Function	not specified	Enter a function name if you require a Tx error notification for this signal. If the field is left empty no function will be configured.  The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.
ComFirstTimeoutFactor	0...1	Integer	<b>0</b> - 65535	First timeout time that is used for Rx deadline monitoring. '0' for the first timeout indicates that the signal shall not be timeout observed. If Update Bits are not used, the shortest first timeout time of all signals of an IPdu is used. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.
ComHandleId	1...1	Integer	not specified	This Parameter is not used in Il_AsrCom!  The numerical value used as the ID.  For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal.  For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.
ComInvalidNotification	0...1	Function	not specified	Enter a function name if you require a Rx invalid value notification for this signal. If the field is left empty no function will be configured.  The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComNotification	0...1	Function	not specified	<p>Enter a function name if you require a Rx indication or Tx confirmation for this ComSignal or ComSignalGroup. If the field is left empty no function will be configured. The call context depends on the settings made in ComIPduSignalProcessing which can either be 'DEFERRED' or 'IMMEDIATE'.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComRxDataTimeout Action	0...1	Enumerati on	<b>NONE</b>  <b>REPLACE</b>	<p>Activate this feature to write the ComSignalInitValue to the ComSignal or ComSignalGroup (REPLACE).</p> <p>Deactivate this feature to disable the timeout default value for this ComSignal or ComSignalGroup (NONE).</p>
ComSignalDataInvalidValue	0...1	Integer	not specified	<p>Specifies the invalid value of the ComSignal or ComGroupSignal.</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p>
ComSignalEndians s	1...1	Enumerati on	BIG_ENDIAN  LITTLE_ENDIAN  OPAQUE	<p>Specifies the byte order that is used for this ComSignal or ComGroupSignal. OPAQUE indicates that no byte order conversion is performed (e.g. for byte arrays).</p>
ComSignalInitValue	0...1	Integer	not specified	<p>Specify the initial value of the ComSignal or ComGroupSignal that is used as value after Com_Init().</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p> <p>Large signals (&gt;64 bit) are initialized with the pattern '0xFF'.</p>
ComSignalLength	0...1	Integer	<b>1 - 8191</b>	<p>Indicates the signal length in bytes. Depending on the signal size not all bits might be sent / received using external communication. If the signal type (ComSignalType) is unit8[n] the number of bytes must be equal to the number of bits indicated in the signal size (ComBitSize).</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComSignalType	1...1	Enumeration	BOOLEAN SINT16 SINT32 SINT8 UINT16 UINT32 UINT8 UINT8_N	Specify the application data type of the signal. If UINT8[n] (byte array) is selected the signal size must have the same size as the signal specified in the ECU description or the database.
ComTimeoutFactor	0...1	Integer	<b>0</b> - 65535	Timeout time that is used for reception or transmission deadline monitoring.  '0' indicates that the signal shall not be timeout observed. The shortest timeout time of all signals and signal groups of an IPdu is used. Rx signals and Rx signal groups with a configured update bit are not evaluated for the I-Pdu based reception timeout time. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.
ComNotificationFlag	0...1	Boolean	true false	Activate the feature to generate a macro to read and clear a indication flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup. Note: The flag is set in the call context of Com_RxIndication.  RESTRICTIONS: 'ComNotificationFlag' is only available, if 'ComEnableIndicationFlags' is set to 'true'.
ComTimeoutNotificationFlag	0...1	Boolean	true false	Activate the feature to generate a macro to read and clear a timeout flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.  RESTRICTIONS: 'ComTimeoutNotificationFlag' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.
ComStateOnFlag	0...1	Boolean	true false	Activate the feature to generate a macro to read a state on flag for the ComSignal or ComSignalGroup.  Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.  RESTRICTIONS: 'ComStateOnFlag' is only available, if 'ComEnableStateOnFlags' is set to 'true'.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTimeoutNotifica tion	0...1	Function	not specified	<p>Enter a function name if you require a timeout notification for this signal.</p> <p>If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComTransferPropert y	0...1	Enumerati on	<b>PENDING</b> TRIGGERED TRIGGERED_ON_CHANGE	<p>The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the signal can either be</p> <ul style="list-style-type: none"> <li>- TRIGGERED: the transmission of the message is triggered if the signal is written regardless of the signal value</li> <li>- TRIGGERED_ON_CHANGE: the transmission of the message is triggered if the signal is written and the signal value has changed</li> <li>- PENDING: writing to the signal does not cause direct transmission.</li> </ul> <p>RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.</p>
ComUpdateBitPositi on	0...1	Integer	<b>0</b> - 65528	<p>The update bit is located in the same IPdu. It is used to signalize that the signal was updated since the previous IPdu transmission. The update bit position must not be used by any other signal.</p> <p>RESTRICTIONS: 'ComUpdateBitPosition' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
ComIssmSignal	0...1	Boolean	true <b>false</b>	<p>Activate the feature to indicate, that the signal is handled by SysService_Issm and not by the application.</p> <p>Deactivate the feature to use the ComSignal with the application.</p>
ComMissingSource Value	0...1	Integer	not specified	<p>Specify the missing source value of the ComGwSignal that is used as value after a timeout of ComGwSource.</p> <p>RESTRICTIONS: 'ComMissingSourceValue' is only available, if 'ComMissingSourceValueHandling' is set to 'true'.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComInternal	0...1	Boolean	true <b>false</b>	<p>If enabled, the signal ID is assigned dynamically at post-build time. This requires the user of the signal (e.g. the application, RTE, gateway) to support the post-build concept as well.</p> <p>If disabled, the signal ID will be assigned statically and will not change at post-build time. This setting is typically required for signals used by the application and RTE.</p> <p>Enable this switch solely for signals that are used by components that support the post-build concept. E.g. signals that are handled by the gateway component only. Signals that have this feature enabled can be removed and added at post-build time without causing the signal ID of other non post-build signals to be changed.</p> <p>Signals that do not have this property must not be removed or added at post-build time as this can have side effects on the signal ID of other (non post-build) signal IDs.</p>
SystemTemplateSystemSignalRef	1...1	Foreign-reference	not specified	This Parameter is not used in IL_AsrCom! Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.

Sub Container	Multiplicity
ComFilter	0...1

#### 6.4.1.15 ComFilter

Container Name	ComFilter
Path	\\MICROSAR\\Com\\ComConfig\\ComSignal\\ComFilter
Multiplicity	0...1
Description	<p>This container contains the configuration parameters of COM Filters.</p> <p>Note: On sender side the container is used to specify the transmission mode conditions.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
----------------	-------------------	---------------	--	-------------



Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComFilterAlgorithm	1...1	Enumeration	<b>ALWAYS</b> <b>MASKED_NEW_DIFFERS_MASKED_OLD</b> <b>MASKED_NEW_DIFFERS_X</b> <b>MASKED_NEW_EQUALS_X</b> <b>NEVER</b> <b>NEW_IS_OUTSIDE</b> <b>NEW_IS_WITHIN</b> <b>ONE_EVERY_N</b>	<p>The supported filter algorithms for Tx signals are:</p> <ul style="list-style-type: none"> <li>- None: The signal has no filter (Multiplicity of ComFilter)</li> <li>- Always: This filter always evaluates to TRUE</li> <li>- Never: This filter always evaluates to FALSE</li> <li>- MaskedNewDiffersMaskedOld: <math>((\text{new} \&amp; \text{mask}) \neq (\text{old} \&amp; \text{mask}))</math></li> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul> <p>The supported filter algorithms for Rx signals are:</p> <ul style="list-style-type: none"> <li>- None: The signal has no filter (Multiplicity of ComFilter)</li> <li>- Always: This filter always evaluates to TRUE</li> <li>- MaskedNewDiffersMaskedOld: <math>((\text{new} \&amp; \text{mask}) \neq (\text{old} \&amp; \text{mask}))</math></li> </ul>
ComFilterMask	0...1	Integer	not specified	<p>The specified mask is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewDiffersMaskedOld: <math>((\text{new} \&amp; \text{mask}) \neq (\text{old} \&amp; \text{mask}))</math></li> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> </ul>
ComFilterMax	0...1	Integer	not specified	<p>The specified max value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul>
ComFilterMin	0...1	Integer	not specified	<p>The specified min value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul>
ComFilterX	0...1	Integer	not specified	<p>The specified value X is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> </ul>

#### 6.4.1.16 ComSignalGroup

Container Name	ComSignalGroup
Path	\\MICROSAR\\Com\\ComConfig\\ComSignalGroup

<b>Multiplicity</b>	0...*
<b>Description</b>	<p>Contains the configuration parameters of Com signal groups.</p> <p>The shortName is used as the symbolic name of the signal group (ComSignalGroupName). This name is also used as the handle name for the signal group. This parameter is only stored in the XML file, and must not be used within the implementation.</p>

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComBitPosition	1...1	Integer	0 - 65528	<p>Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.</p> <p>If the signal is a Motorola signal this indicates the MSB.</p> <p>If the byte order is a Intel signal this attribute indicates the LSB.</p>
ComBitSize	1...1	Integer	0 - 65528	Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.
ComDataInvalidAction	0...1	Enumeration	<b>NONE</b> NOTIFY REPLACE	This parameter defines the action performed upon reception of an invalid value of one of the included signals. If Replace is used the ComSignalInitValue will be used for the replacement.
ComErrorNotification	0...1	Function	not specified	<p>Enter a function name if you require a Tx error notification for this signal. If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComFirstTimeoutFactor	0...1	Integer	<b>0</b> - 65535	<p>First timeout time that is used for Rx deadline monitoring. '0' for the first timeout indicates that the signal shall not be timeout observed. If Update Bits are not used, the shortest first timeout time of all signals of an IPdu is used.</p> <p>The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.</p>
ComHandleId	1...1	Integer	not specified	<p>This Parameter is not used in Il_AsrCom!</p> <p>The numerical value used as the ID.</p> <p>For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal.</p> <p>For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComInvalidNotification	0...1	Function	not specified	<p>Enter a function name if you require a Rx invalid value notification for this signal group. If the field is left empty no function will be configured.</p> <p>The prototype void <b>_Name_</b>(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComNotification	0...1	Function	not specified	<p>Enter a function name if you require a Rx indication or Tx confirmation for this ComSignal or ComSignalGroup. If the field is left empty no function will be configured. The call context depends on the settings made in ComIPduSignalProcessing which can either be 'DEFERRED' or 'IMMEDIATE'.</p> <p>The prototype void <b>_Name_</b>(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComNotificationFlag	0...1	Boolean	true false	<p>Activate the feature to generate a macro to read and clear a indication flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup. Note: The flag is set in the call context of Com_RxIndication.</p> <p>RESTRICTIONS: 'ComNotificationFlag' is only available, if 'ComEnableIndicationFlags' is set to 'true'.</p>
ComStateOnFlag	0...1	Boolean	true false	<p>Activate the feature to generate a macro to read a state on flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>RESTRICTIONS: 'ComStateOnFlag' is only available, if 'ComEnableStateOnFlags' is set to 'true'.</p>
ComRxDataTimeout Action	0...1	Enumeration	<b>NONE</b> <b>REPLACE</b>	<p>Activate the feature to write the ComSignalInitValue to the ComSignal or ComSignalGroup (REPLACE).</p> <p>Deactivate the feature to disable the timeout default value for this ComSignal or ComSignalGroup (NONE).</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTimeoutFactor	0...1	Integer	<b>0</b> - 65535	<p>Timeout time that is used for reception or transmission deadline monitoring.</p> <p>'0' indicates that the signal shall not be timeout observed. The shortest timeout time of all signals and signal groups of an IPdu is used. Rx signals and Rx signal groups with a configured update bit are not evaluated for the I-Pdu based reception timeout time. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.</p>
ComTimeoutNotifica tion	0...1	Function	not specified	<p>Enter a function name if you require a timeout notification for this signal.</p> <p>If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComTimeoutNotifica tionFlag	0...1	Boolean	true false	<p>Activate the feature to generate a macro to read and clear a timeout flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>RESTRICTIONS: 'ComTimeoutNotificationFlag' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
ComTransferPropert y	0...1	Enumerati on	<b>PENDING</b> <b>TRIGGERED</b> <b>TRIGGERED_ON_CHANGE</b>	<p>The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the signal group can either be</p> <ul style="list-style-type: none"> <li>- TRIGGERED: the transmission of the message is triggered if at least one group signal of signal group is written regardless of the group signal value</li> <li>- TRIGGERED_ON_CHANGE: the transmission of the message is triggered if at least one group signal of signal group is written and the group signal value has changed</li> <li>- PENDING: writing to the signal group does not cause direct transmission.</li> </ul> <p>Note: If at least one group signal of the signal group has a defined transfer property equal to TRIGGERED or TRIGGERED_ON_CHANGE, the transfer property of the signal group will be ignored and the trigger will be evaluated per group signal.</p> <p>RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.</p>

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComUpdateBitPosition	0...1	Integer	<b>0</b> - 65528	<p>The update bit is located in the same IPdu. It is used to signalize that the signal group was updated since the previous IPdu transmission. The update bit position must not be used by any other signal group.</p> <p>RESTRICTIONS: 'ComUpdateBitPosition' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
ComInternal	0...1	Boolean	<b>true</b> <b>false</b>	<p>If enabled, the signal ID is assigned dynamically at post-build time. This requires the user of the signal (e.g. the application, RTE, gateway) to support the post-build concept as well.</p> <p>If disabled, the signal ID will be assigned statically and will not change at post-build time. This setting is typically required for signals used by the application and RTE.</p> <p>Enable this switch solely for signals that are used by components that support the post-build concept. E.g. signals that are handled by the gateway component only. Signals that have this feature enabled can be removed and added at post-build time without causing the signal ID of other non post-build signals to be changed.</p> <p>Signals that do not have this property must not be removed or added at post-build time as this can have side effects on the signal ID of other (non post-build) signal IDs.</p>
SystemTemplateSignalGroupRef	1...1	Foreign-reference	not specified	Reference to the ISignalToIPduMapping that contains a reference to the ISignal (SystemTemplate) which this ComSignalGroup represents.

Sub Container	Multiplicity
ComGroupSignal	0...*

#### 6.4.1.17 ComGroupSignal

Container Name	ComGroupSignal
Path	\MICROSAR\Com\ComConfig\ComSignalGroup\ComGroupSignal
Multiplicity	0...*
Description	<p>This container contains the configuration parameters of group signals. I.e. signals that are included within a signal group.</p> <p>The shortName is used as the symbolic name of the signal</p>

(ComSignalName). This name is also used as the handle name for the signal. This parameter is only stored in the XML file, and must not be used within the implementation.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComBitPosition	1...1	Integer	0 - 65528	Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.  If the signal is a Motorola signal this indicates the MSB.  If the byte order is a Intel signal this attribute indicates the LSB.
ComBitSize	1...1	Integer	0 - 65528	Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.
ComHandleId	1...1	Integer	not specified	This Parameter is not used in Il_AsrCom!  The numerical value used as the ID.  For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal.  For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.
ComSignalDataInvalidValue	0...1	Integer	not specified	Specifies the invalid value of the ComSignal or ComGroupSignal.  To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.
ComSignalEndianness	1...1	Enumeration	BIG_ENDIAN  LITTLE_ENDIAN  OPAQUE	Specifies the byte order that is used for this ComSignal or ComGroupSignal. OPAQUE indicates that no byte order conversion is performed (e.g. for byte arrays).
ComSignalInitValue	0...1	Integer	not specified	Specify the initial value of the ComSignal or ComGroupSignal that is used as value after Com_Init().  To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.  Large signals (>64 bit) are initialized with the pattern '0xFF'.
ComSignalLength	0...1	Integer	<b>1</b> - 8191	Indicates the signal length in bytes. Depending on the signal size not all bits might be sent / received using external communication. If the signal type (ComSignalType) is unit8[n] the number of bytes must be equal to the number of bits indicated in the signal size (ComBitSize).

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComSignalType	1...1	Enumeration	BOOLEAN SINT16 SINT32 SINT8 UINT16 UINT32 UINT8 UINT8_N	Specify the application data type of the signal. If UINT8[n] (byte array) is selected the signal size must have the same size as the signal specified in the ECU description or the database.
ComMissingSource Value	0...1	Integer	not specified	Specify the missing source value of the ComGwSignal that is used as value after a timeout of ComGwSource.  RESTRICTIONS: 'ComMissingSourceValue' is only available, if 'ComMissingSourceValueHandling' is set to 'true'.
ComTransferProperty	0...1	Enumeration	<b>PENDING</b> TRIGGERED TRIGGERED_ON_CHANGE	The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the group signal can either be  - TRIGGERED: the transmission of the message is triggered if the group signal is written regardless of the group signal value  - TRIGGERED_ON_CHANGE: the transmission of the message is triggered if the group signal is written and the group signal value has changed  - PENDING: writing to the group signal does not cause direct transmission.  Note: If all the transfer properties of the group signals of a signal group equals PENDING, the transfer property of the signal group will be evaluated for the group signals.  RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.
SystemTemplateSystemSignalRef	1...1	Foreign-reference	not specified	This Parameter is not used in Il_AsrCom!  Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.

Sub Container	Multiplicity
ComFilter	0...1

### 6.4.1.18 ComFilter

Container Name	ComFilter
Path	\MICROSAR\Com\ComConfig\ComSignalGroup\ComGroupSignal\ComFilter
Multiplicity	0...1
Description	This container contains the configuration parameters of COM Filters. Note: On sender side the container is used to specify the transmission mode conditions.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComFilterAlgorithm	1...1	Enumeration	ALWAYS MASKED_NEW_DIFFERS_MASKED_OLD MASKED_NEW_DIFFERS_X MASKED_NEW_EQUALS_X NEVER NEW_IS_OUTSIDE NEW_IS_WITHIN ONE_EVERY_N	The supported filter algorithms for Tx signals are: - None: The signal has no filter (Multiplicity of ComFilter) - Always: This filter always evaluates to TRUE - Never: This filter always evaluates to FALSE - MaskedNewDiffersMaskedOld: $((new \& mask) \neq (old \& mask))$ - MaskedNewEqualsX: $((new \& mask) == x)$ - MaskedNewDiffersX: $((new \& mask) \neq x)$ - NewIsOutside: $((new < min) \vee (max < new))$ - NewIsWithin: $((min \leq new) \wedge (new \leq max))$ The supported filter algorithms for Rx signals are: - None: The signal has no filter (Multiplicity of ComFilter) - Always: This filter always evaluates to TRUE - MaskedNewDiffersMaskedOld: $((new \& mask) \neq (old \& mask))$
ComFilterMask	0...1	Integer	not specified	The specified mask is used for the filters: - MaskedNewDiffersMaskedOld: $((new \& mask) \neq (old \& mask))$ - MaskedNewEqualsX: $((new \& mask) == x)$ - MaskedNewDiffersX: $((new \& mask) \neq x)$
ComFilterMax	0...1	Integer	not specified	The specified max value is used for the filters: - NewIsOutside: $((new < min) \vee (max < new))$ - NewIsWithin: $((min \leq new) \wedge (new \leq max))$



Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComFilterMin	0...1	Integer	not specified	The specified min value is used for the filters: - NewIsOutside: ((new < min)    (max < new)) - NewIsWithin: ((min <= new) && (new <= max))
ComFilterX	0...1	Integer	not specified	The specified value X is used for the filters: - MaskedNewEqualsX: ((new & mask) == x) - MaskedNewDiffersX: ((new & mask) != x)

#### 6.4.1.19 ComGeneral

Container Name	ComGeneral
Path	\MICROSAR\Com\ComGeneral
Multiplicity	1...1
Description	Contains the general configuration parameters of the Com module.

Attribute Name	Multiplicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComConfigurationString	0...1	String	not specified	This ID provides the unique identifier of the post-build configurable configuration part. At runtime the configuration ID can be retrieved by Com_GetConfigurationStringPtr().  RESTRICTIONS: 'ComConfigurationString' is only available, if 'ComGetConfigurationStringPtrApi' is set to 'true'.
ComProtectIndicationFlags	0...1	Boolean	true false	Activate the feature to enable an interrupt protection in the generated clear macro for an indication flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the generated interrupt protection. The application has to ensure the consistency of the flag access.  RESTRICTIONS: 'ComProtectIndicationFlags' is only available, if 'ComEnableIndicationFlags' is set to 'true'.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComResetTimeoutFlags	0...1	Boolean	true false	<p>Activate the feature to reset the timeout flag, if the ComSignal or ComSignalGroup is received again after a timeout occurred.</p> <p>Deactivate the feature to disable the reset of the timeout flag.</p> <p>RESTRICTIONS: 'ComResetTimeoutFlags' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
ComProtectTimeoutFlags	0...1	Boolean	true false	<p>Activate the feature to enable an interrupt protection in the generated clear macro for a timeout flag for the ComSignal or ComSignalGroup. Deactivate the feature to disable the generated interrupt protection. The application has to ensure the consistency of the flag access.</p> <p>RESTRICTIONS: 'ComProtectTimeoutFlags' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
ComEnableRx	0...1	Boolean	true false	With this option reception of all signals can be disabled.
ComEnableTx	0...1	Boolean	true false	With this option transmission of all signals can be disabled.
ComMissingSourceValueHandling	0...1	Boolean	true false	This attribute is set automatically - BeforeGeneration- depending on attribute existence in the database. If enabled, the database attribute GenSigMissingSource value is used as Tx signal value in case of a Rx signal timeout. This value is set automatically at generation time if at least one database provides this attribute.
ComEnableSignalInterface	0...1	Boolean	true false	Activate the feature to enable the signal API. Deactivate the feature to disable the feature (e.g. if all signals are grouped Com_ReceiveSignal and Com_TransmitSignal are not used).
ComGwTriggerTxOnInvalidation	0...1	Boolean	true false	Feature triggers the routed Tx Signals in case of an invalidation caused by a Rx timeout.
ComGwInvalidateTxOnRxTimeout	0...1	Boolean	true false	Feature invalidates the routed Tx Signals if a Rx timeout occurs.
ComQueueFailedTransmitRequests	0...1	Boolean	true false	<p>If the feature is activated, the COM repeats transmission requests of a Tx I-Pdu to the underlying layer if E_NOT_OK is returned.</p> <p>If the feature is deactivated, the COM will ignore all errors from the underlying communication layer as recommended by AUTOSAR.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComGwCanMost	0...1	Boolean	true false	Activate this feature to enable the CAN MOST Gateway extension.
ComGwStopTxOnRxTimeout	0...1	Boolean	true false	Feature stops transmission of Tx PDUs if the Tx PDU has a single Rx PDU only that timed out.
ComLargeComIPdus	0...1	Boolean	true false	Activate the feature to enable support for ComIPdus with ComIPduSize greater than 31 bytes.  Deactivate the feature if only ComIPdus with ComIPduSize less than 31 bytes are used.
ComBAC21Compatibility	0...1	Boolean	true false	Activate the feature to enable the BAC 2.1 compatibility mode. Deactivate the feature to disable the BAC 2.1 compatibility mode.
ComGw03ConsistencyCheck	0...1	Boolean	true false	Deactivate the feature to skip the check Gw03.
ComGetConfigurationStringPtrApi	0...1	Boolean	true false	Activate the feature to enable the API Com_GetConfigurationStringPtr(). Deactivate the feature to disable the feature.
ComMainFunctionTxApi	0...1	Boolean	true false	Activate the feature to enable the API Com_MainFunctionTx().  Deactivate the feature to disable the feature.
ComMainFunctionRxApi	0...1	Boolean	true false	Activate the feature to enable the API Com_MainFunctionRx().  Deactivate the feature to disable the feature.
ComShowError0005AsWarning	0...1	Boolean	true false	Activate the feature to generate the Error message 0005 as a Warning.
ComVStdLibMemApi	0...1	Boolean	true false	Activate the feature to use the VSdtLib to set, clear and copy data elements in Com.  Deactivate the feature to disable the feature.  Note: The feature can be disabled for the Mcs12x if the post build data is located e.g. in the GPAGE.
ComTriggerTransmissionByCom	0...1	Boolean	true false	If Com is not required to trigger the transmission of IPdus you can disable this functionality to save resources. This could be for example a Lin or FlexRay only system where transmission is triggered by the lower layer via Com_TriggerTransmit().
ComOptimizeIdTypes	0...1	Boolean	true false	Activate the feature to enable ID type optimization (e.g. use uint8 as Com_SignalIdType if sufficient) Deactivate the feature to disable the ID type optimization.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComRxSignalWriteAccess	0...1	Boolean	true false	This feature is required by some RTE configurations. The feature is not specified by AUTOSAR.
ComTsiTestCode	0...1	Boolean	true false	Activate the feature to enable test code for the integration of Com. Deactivate the feature to disable the feature.
ComIPduGroupTransmit	0...1	Boolean	true false	Activate the feature to enable the API Com_IpduGroupTransmit(). Deactivate the feature to disable the feature.
ComEnableLoTxConfirmation	0...1	Boolean	true false	Activate the feature to restart timeout counters based upon the call of Com_TxConfirmation().  Deactivate the feature to restart timeout counters after the call of PduR_ComTransmit().  This feature is necessary if confirmation functions are not supported by the PDUR.
ComTmsSupport	0...1	Boolean	true false	Activate the feature to enable the use of ComTxModeFalse. Deactivate the feature to disable the use of ComTxModeFalse and activate only the use of ComTxModeTrue. Due to the deactivation of ComTxModeFalse change of the ComFilterAlgorithm is limited to ALWAYS and MASKED_NEW_DIFFERS_MASKED_OLD (GenSigSendType Cyclic, OnWrite and OnChange are possible).
ComLargeSignalSupport	0...1	Boolean	true false	Activate the feature to enable ComSignals with ComBitSize larger than 254 bits. Deactivate the feature if only ComSignals with ComBitSize smaller than 255 bits are used.
ComTransportProtocolSupport	0...1	Boolean	true false	Activate the feature to enable the API to transmit IPdus larger than 8 bytes on CAN via CAN TP. Deactivate the feature to disable the feature.
ComEnableTimeoutFlags	0...1	Boolean	true false	Activate the feature to enable the timeout flag feature. Deactivate the feature to disable the feature.
ComEnableIndicationFlags	0...1	Boolean	true false	Activate the feature to enable the indication flag feature.  Deactivate the feature to disable the feature.
ComEnableStateOnFlags	0...1	Boolean	true false	Activate the feature to enable the state on flag macro interface.  Deactivate the feature to disable the state on flag macro interface.
ComSignalConversionInterface	0...1	Boolean	true false	Activate the feature to enable the signal conversion API.  Deactivate the feature to disable the feature.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComIsLibrary	0...1	Boolean	true false	<p>Activate the feature to indicate, that Com is delivered as a library.</p> <p>Deactivate the feature to indicate, that Com is delivered as source code.</p>
ComEnableUpdateBitSupport	0...1	Boolean	true false	<p>Activate the feature to enable the update bit handling. If a database is loaded and a ComSignal is found with the postfix or infix '_UB' and another signal in the same ComPdu is found with the same &lt;SignalName&gt;, the signal API is deactivated for the ComSignal and the signal is configured as update bit.</p> <p>Deactivate the feature to disable the update bit handling.</p> <p>Note: Update bits can be configured for ComSignals and ComSignalGroups.</p>
ComEnableSignalGroupInterface	0...1	Boolean	true false	<p>Activate the feature to enable the signal group API. Deactivate the feature to disable the feature.</p>
ComConfigurationTimeBase	1...1	Float	not specified	<p>The period between successive calls to the Main Functions (Rx, Tx, Routing) of AUTOSAR COM in seconds.</p>
ComConfigurationTimeBaseUpdate	0...1	Float	not specified	<p>This Parameter exists, if EcuCGenTool_CsAsrDbUpdate has performed an update and editable Parameters are calculated upon this value.</p>
ComConfigurationRxTimeBase	0...1	Float	not specified	<p>Configure the cycle time in which Com_MainFunctionRx() has to be called.</p>
ComConfigurationTxTimeBase	0...1	Float	not specified	<p>Configure the cycle time in which Com_MainFunctionTx() has to be called.</p>
ComConfigurationUseDet	0...1	Boolean	true <b>false</b>	<p>If 'Development Error Detection' is enabled, all development errors are reported to the Development Error Tracer (DET). The errors are described in the [TechnicalReference_Asr_Com.pdf].</p> <p>Note: In general, the development error detection is recommended during pre-test phase. It is not recommended to enable the development error detection in production code due to increased runtime and ROM needs.</p>
ComVersionInfoApi	1...1	Boolean	true false	<p>Activate this feature to enable the function Com_GetVersionInfo() to get major, minor and patch version information.</p>
ComConfigurationUseDem	0...1	Boolean	<b>true</b> false	<p>If 'Production Error Detection' is enabled, production relevant errors are reported to the Diagnostics Event Manager (DEM).</p> <p>Disable this option, if no DEM is present in your system.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComUseRte	0...1	Boolean	<b>true</b> <b>false</b>	Enable this option if the ECU uses an RTE.  If this option is disabled, callback prototypes are generated into the file Appl_Cbk.h.
ComSignalAccessMacroApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the signal access macro API.  Deactivate the feature to disable the signal access macro API.
ComTriggerTransmitApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_TriggerTransmit().  Deactivate the feature to disable the API Com_TriggerTransmit().  Note: This API is needed if the COM is used with LIN, Flexray or MOST.  RESTRICTIONS: 'ComTriggerTransmitApi' is only available, if 'ComConfigurationVariant' is not set to 'Variant 1 (Pre-compile Configuration)'.
ComTriggerIPDUSendApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_TriggerIPDUSend().  Deactivate the feature to disable the API Com_TriggerIPDUSend().
ComGetConfigurationIdApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_GetConfigurationId().  Deactivate the feature to disable the API Com_GetConfigurationId().
ComGetStatusApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_GetStatus().  Deactivate the feature to disable the API Com_GetStatus().
ComReceptionDMAApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_EnableReceptionDM() and Com_DisableReceptionDM().  Deactivate the feature to disable the API Com_EnableReceptionDM() and Com_DisableReceptionDM().
ComDelInitApi	0...1	Boolean	<b>true</b> <b>false</b>	Activate the feature to enable the API Com_DelInit().  Deactivate the feature to disable the API Com_DelInit().

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComSignalInvalidationApi	0...1	Boolean	<b>true</b> false	<p>Activate the feature to enable the API Com_InvalidateSignal() and Com_InvalidateSignalGroup() and the handling of invalid values for Rx signals if configured.</p> <p>Deactivate the feature to disable the feature.</p>
ComUserConfigFile	0...1	String	not specified	<p>A configuration file is generated by GENy. If you want to overwrite settings in the generated configuration file, you can specify a path to a user defined configuration file.</p> <p>The user defined configuration file will be included at the end of the generated file. Therefore definitions in the user defined configuration file can overwrite definitions in the generated configuration file.</p> <p>The content of the user defined configuration file is copied to the end of Com_Cfg.h.</p>
ComPBStartAddress	1...1	Integer	<b>0</b> - n	<p>This setting is mandatory when generating a configuration that will be written to the memory of your ECU at post-build time (i.e. when generating a hex-file). This setting is not relevant when generating the initial ECU configuration at link- or pre-compile time (i.e. when generating a C-file that will be linked with your project).</p> <p>The Module Start Address must specify the start address of the memory area, where the generated hex-file will be mapped to. If the ECU supports selectable post-build configuration, the Module Start Address can be any location that is accessible for the BSW module. If selectable post-build configuration is not supported by your ECU, the Module Start Address must be the same address to which the initial configuration was mapped. Please see into your linker map file to get this address.</p> <p>RESTRICTIONS: 'ComPBStartAddress' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComMaxNumberOfRxIPdus	1...1	Integer	<b>1</b> - 65535	<p>The number of IPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled IPdus at post-build time memory must be reserved.</p> <p>Configure the maximum number of Rx IPdus, which shall be able to handle at any time. The minimum number is the currently configured number of Rx IPdus.</p> <p>RESTRICTIONS: 'ComMaxNumberOfRxIPdus' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
ComMaxNumberOfTxIPdus	1...1	Integer	<b>1</b> - 65535	<p>The number of IPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled IPdus at post-build time memory must be reserved.</p> <p>Configure the maximum number of Tx IPdus, which shall be able to handle at any time. The minimum number is the currently configured number of Tx IPdus.</p> <p>RESTRICTIONS: 'ComMaxNumberOfTxIPdus' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
ComMaxNumberOfRxUpdateBits	1...1	Integer	<b>1</b> - 65535	<p>The number of Rx update bits in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled Rx update bits at post-build time memory must be reserved.</p> <p>Configure the maximum number of Rx update bits, which shall be able to handle at any time. The minimum number is the currently configured number of Rx update bits.</p> <p>RESTRICTIONS: 'ComMaxNumberOfRxUpdateBits' is only available, if 'ComEnableUpdateBitSupport' is set to 'true' and 'ComConfigurationVariant' is 'Variant 3 (Post-build Configuration)'</p>
ComBufferSize	1...1	Integer	<b>0</b> - n	<p>The number of ComPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled ComPdus at post-build time memory must be reserved at link time.</p> <p>The size of the buffer depends on the current COM configuration and on the scalability the shall be available at post-build time.</p> <p>RESTRICTIONS: 'ComBufferSize' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>



Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComGwSuspendIsr PerPdu	0...1	Boolean	true <b>false</b>	If this feature is activated, only one ISR suspension is used per routed Rx I-Pdu. Otherwise every single signal access is protected by an own ISR suspension.
ComMaxRxIPduLen gth	0...1	Integer	1... <b>8</b> ...65535	Configure the maximum length of Rx IPdus in bytes, which shall be able to handle at any time. The minimum length is the currently configured maximum length of Rx IPdus.  RESTRICTIONS: 'ComMaxRxIPduLength' is not available, if 'ComConfigurationVariant' is set to 'Variant 1 (Pre-compile Configuration)'.
ComEnableReception Filtering	0...1	Boolean	<b>true</b> false	Activate the feature to enable the signal reception filtering.  Deactivate the feature to disable the signal reception filtering.
ComMaxNumberOf TxSignalGroups	1...1	Integer	<b>1</b> - 65535	The number of Tx signal groups in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled Tx signal groups at post-build time memory must be reserved.  Configure the maximum number of Tx signal groups, which shall be able to handle at any time. The minimum number is the currently configured number of Tx signal groups.  RESTRICTIONS: 'ComMaxNumberOfTxSignalGroups' is only available, if 'ComEnableSignalGroupInterface' is set to 'true' and 'ComConfigurationVariant' is 'Variant 3 (Post-build Configuration)'
ComTxTimeoutForTx ModeNoneSupport	0...1	Boolean	true <b>false</b>	If this feature is activated the timeout context of a Tx I-Pdu is configurable by the attribute 'ComTxIPduTimeoutContext' to support transmission deadline monitoring for the transmission mode NONE.  This could be used for Tx I-Pdus which are triggered by a bus schedule table (e.g. LIN).
ComMultipleIPduGr oupRefSupport	0...1	Boolean	true <b>false</b>	If this feature is activated one I-Pdu could be contained in more than one I-Pdu group.
ComUsePduInfoType	0...1	Boolean	true <b>false</b>	If this feature is activated, PduInfoPtr instead of SduPtr is used for the APIs Com_RxIndication and Com_TriggerTransmit.
ComDynamicDlcSu pport	0...1	Boolean	true <b>false</b>	If this feature is activated, the received data length (PduInfoPtr->SduLength) is checked by the function Com_RxIndication, and only completely received signals and signal groups are processed.

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComNotiFctRxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Indication Function' name of a Rx signal or signal group, if the button for the value calculation is used.
ComNotiFctTxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Confirmation Function' name of a Tx signal or signal group, if the button for the value calculation is used.
ComTimeoutNotiFctRxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Timeout Function' name of a Rx signal or signal group, if the button for the value calculation is used.
ComInvalidNotiFctRxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Invalid Function' name of a Rx signal or signal group, if the button for the value calculation is used.
ComErrorNotiFctTxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Error Function' name of a Tx signal or signal group, if the button for the value calculation is used.
ComIpdudCalloutFctPrefix	0...1	String	not specified	This prefix is used for the calculated I-PDU callout name if the button for the value calculation is used.
ComIndicationFlagPostfix	0...1	String	not specified	<p>This postfix is used to calculate the name of the indication flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' / 'Com_Clr' + 'RxSig' / 'RxSigGrp' + &lt;ComIndicationFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComIndicationFlagPostfix' is only available, if 'Il_AsrCom.ComEnableIndicationFlags' is set to 'true'.</p>
ComTimeoutFlagPostfix	0...1	String	not specified	<p>This postfix is used to calculate the name of the timeout flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' / 'Com_Clr' + 'RxSig' / 'RxSigGrp' + &lt;ComTimeoutFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComTimeoutFlagPostfix' is only available, if 'Il_AsrCom.ComEnableTimeoutFlags' is set to 'true'.</p>
ComStateOnFlagPostfix	0...1	String	not specified	<p>This postfix is used to calculate the name of the state on flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' + 'RxSig' / 'RxSigGrp' + &lt;ComStateOnFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComStateOnFlagPostfix' is only available, if 'Il_AsrCom.ComEnableStateOnFlags' is set to 'true'.</p>

Attribute Name	Multi- plicity	Value Type	Values <small>Default value is typed bold</small>	Description
ComTimeoutNotiFct TxPrefix	0...1	String	not specified	This prefix is used for the calculated 'Timeout Function' name of a Tx signal or signal group, if the button for the value calculation is used.

## 6.5 Configuration with GENy

The COM is configured with the help of the configuration tool GENy. Most parameters are imported from the communication database (see chapter 6); some settings like notification functions to the upper layer are not part of the communication database and therefore have to be configured in the configuration tool.

For settings imported from the communication database late adjustments are sometimes useful. Please note that the databases of the different bus systems provide a different scope of parameters, i.e. it depends also on the bus system which parameters have to be configured manually. For these reasons the user interface allows to modify several settings described in chapter 6.

To use the COM layer the component must be enabled in the component selection dialog of the system configuration as shown in the following screenshot.

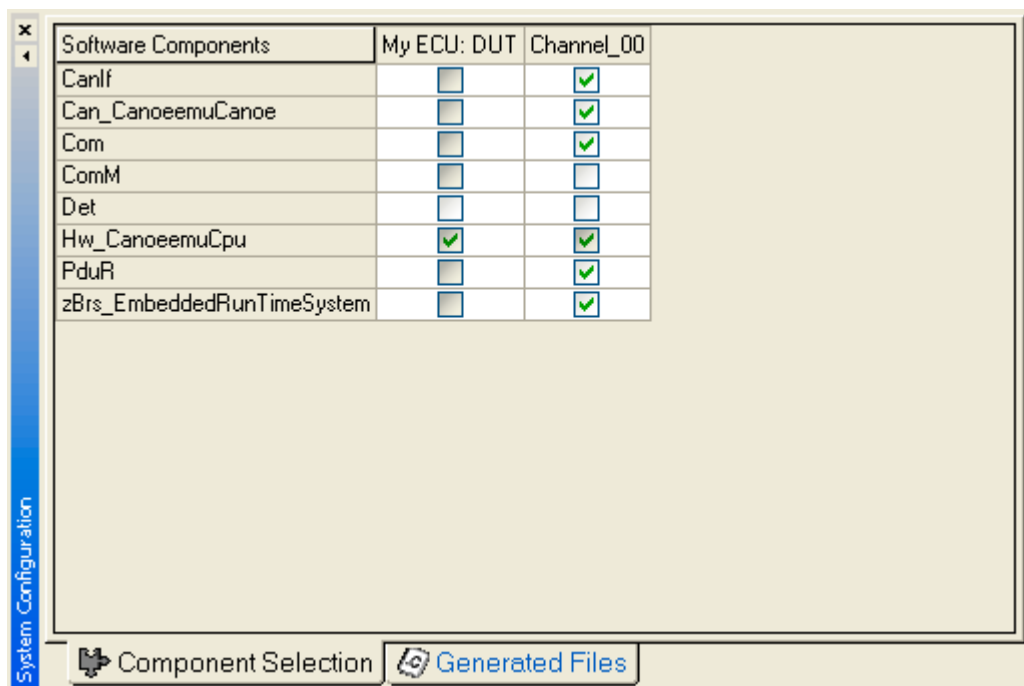


Figure 6-1 Component Selection in the System Configuration

### 6.5.1 Com Parameters


**Info**

The visibility of configuration parameters depends on the tailoring of the delivered package and is OEM dependent.

Attribute Name	Value Type	Default Value	Description
ModuleInstance			
Indication Function	String	Appl_CO MCbk_	<p>MICROSAR PARAMETER DEFINITION: 'ComNotiFctRxPrefix'</p> <p>This prefix is used for the calculated 'Indication Function' name of a Rx signal or signal group, if the button for the value calculation is used.</p>
Confirmation Function	String	Appl_CO MCbkTAc ck_	<p>MICROSAR PARAMETER DEFINITION: 'ComNotiFctTxPrefix'</p> <p>This prefix is used for the calculated 'Confirmation Function' name of a Tx signal or signal group, if the button for the value calculation is used.</p>
Rx Timeout Function	String	Appl_CO MCbkTO ut_	<p>MICROSAR PARAMETER DEFINITION: 'ComTimeoutNotiFctRxPrefix'</p> <p>This prefix is used for the calculated 'Timeout Function' name of a Rx signal or signal group, if the button for the value calculation is used.</p>
Tx Timeout Function	String	Appl_CO MCbkTO ut_	<p>MICROSAR PARAMETER DEFINITION: 'ComTimeoutNotiFctTxPrefix'</p> <p>This prefix is used for the calculated 'Timeout Function' name of a Tx signal or signal group, if the button for the value calculation is used.</p>
Invalid Function	String	Appl_CO MCbkInv —	<p>MICROSAR PARAMETER DEFINITION: 'ComInvalidNotiFctRxPrefix'</p> <p>This prefix is used for the calculated 'Invalid Function' name of a Rx signal or signal group, if the button for the value calculation is used.</p>
Error Function	String	Appl_CO MCbkTER r_	<p>MICROSAR PARAMETER DEFINITION: 'ComErrorNotiFctTxPrefix'</p> <p>This prefix is used for the calculated 'Error Function' name of a Tx signal or signal group, if the button for the value calculation is used.</p>

Attribute Name	Value Type	Default Value	Description
I-PDU Callout	String	Appl_CO MCoUt_	<p>MICROSAR PARAMETER DEFINITION: 'ComIpduCalloutFctPrefix'</p> <p>This prefix is used for the calculated I-PDU callout name if the button for the value calculation is used.</p>
Indication Flag	String	Indicatio nFlag	<p>MICROSAR PARAMETER DEFINITION: 'ComIndicationFlagPostfix'</p> <p>This postfix is used to calculate the name of the indication flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' / 'Com_Clr' + 'RxSig' / 'RxSigGrp' + &lt;ComIndicationFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComIndicationFlagPostfix' is only available, if 'II_AsrCom.ComEnableIndicationFlags' is set to 'true'.</p>
Timeout Flag	String	TimeoutF lag	<p>MICROSAR PARAMETER DEFINITION: 'ComTimeoutFlagPostfix'</p> <p>This postfix is used to calculate the name of the timeout flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' / 'Com_Clr' + 'RxSig' / 'RxSigGrp' + &lt;ComTimeoutFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComTimeoutFlagPostfix' is only available, if 'II_AsrCom.ComEnableTimeoutFlags' is set to 'true'.</p>
State On Flag	String	StateOnF lag	<p>MICROSAR PARAMETER DEFINITION: 'ComStateOnFlagPostfix'</p> <p>This postfix is used to calculate the name of the state on flag provider macro.</p> <p>The macro name is made up as follows: 'Com_Get' + 'RxSig' / 'RxSigGrp' + &lt;ComStateOnFlagPostfix&gt;</p> <p>RESTRICTIONS: 'ComStateOnFlagPostfix' is only available, if 'II_AsrCom.ComEnableStateOnFlags' is set to 'true'.</p>
ComIPdu			
IPdu Callout	String	N.a.	AUTOSAR PARAMETER DEFINITION: 'ComIPduCallout'

Attribute Name	Value Type	Default Value	Description
			<p>Enter a function name if you require an IPdu callout for this PDU. If the field is left empty no function will be configured. The function will be called on task level inside Com_MainFunctionRx() or Com_MainFunctionTx().</p> <p>The prototype boolean &lt;IPdu Callout Name&gt;(PduIdType ID, uint8* ipduD) is generated to Com_Cbk.h and has to be implemented by the application.</p>
IPdu Rx Handle Id	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComIPduRxHandleId'</p> <p>The numerical value used as the ID of this IPdu. The ComIPduRxHandleId is required by the API calls to receive IPdus from the PDUR.</p> <p>It is only present for IPdu is received from the PDUR, because Com is the starting module for Tx IPdus and there is no need to define IDs for - IPdus in the Com module.</p>
IPdu Signal Processing	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComIPduSignalProcessing'</p> <p>This option determines the call context and the point of time when the signal notification function is called.</p> <ul style="list-style-type: none"> <li>- IMMEDIATE: The notification function is called within Com_TxConfirmation() or Com_RxIndication(). Depending on the lower layer interface, this might be in interrupt context.</li> <li>- DEFERRED: The notification function is called on task level during the next call cycle of Com_MainFunctionRx() or Com_MainFunctionTx().</li> </ul>
IPdu Size [bytes]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComIPduSize'</p> <p>Size of the IPdu in bytes.</p> <p>The maximum size is limited by the underlying communication interface.</p> <p>0-8 for CAN and LIN 0-254 for FlexRay 0-8191 for Others</p>
Generate	Bool	true	<p>MICROSAR PARAMETER DEFINITION: 'ComIPdu.ComGenerate'</p> <p>Activate this feature to enable any Com API for the ComPdu and it ComSignals.</p> <p>Deactivate this feature to disable any Com API for the ComPdu and it ComSignals. This can be used to reduce</p>

Attribute Name	Value Type	Default Value	Description
			calculation time in Com, if a PDU is routed completely in the PDUR.
IPdu Group Ref	Object	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComIPduGroupRef'</p> <p>Select the ComIPduGroup this ComPdu should be assigned to. Clear the selection to remove the assignment.</p>
Tx IPdu Unused Areas Default	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxIPduUnusedAreasDefault'</p> <p>The specified value is used to fill gaps between the IPdu signals.</p>
ComIPdu > ComRxIPdu			
IPdu Direction	Enum	0	<p>MICROSAR PARAMETER DEFINITION: 'ComIPduDirection'</p> <p>The direction defines if this IPdu, and therefore the contributing signals and signal groups, shall be send or received.</p>
ComIPdu > ComTxIPdu			
IPdu Direction	Enum	1	<p>MICROSAR PARAMETER DEFINITION: 'ComIPduDirection'</p> <p>The direction defines if this IPdu, and therefore the contributing signals and signal groups, shall be send or received.</p>
Tx IPdu Minimum Delay Time Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxIPduMinimumDelayTimeFactor'</p> <p>This attribute defines the minimum transmit delay between two subsequent messages of the same ID. A minimum delay time of '0' disables the observation. The value must be a multiple of the Com_MainFunctionTx() cycle time.</p>
Timeout Context	Enum	0	<p>MICROSAR PARAMETER DEFINITION: 'ComTxIPduTimeoutContext'</p> <p>Transmission deadline monitoring context for this Tx I-Pdu.</p> <p>COM_TRANSMIT: observe the time between PduR_ComTransmit() and Com_TxConfirmation()</p> <p>TX_CONFIRMATION: observe the time between two consecutive calls to Com_TxConfirmation()</p> <p>Note: This attribute will be configured automatically</p>



Attribute Name	Value Type	Default Value	Description
			<p>depending on the bus type of the Tx I-Pdu</p> <p>CAN: timeout context COM_TRANSMIT</p> <p>LIN: timeout context TX_CONFIRMATION</p> <p>FlexRay: depends on the parameter 'FrIfNoneMode' of the FrTxPdu</p> <p>TRUE: TX_CONFIRMATION</p> <p>FALSE: COM_TRANSMIT</p> <p>RESTRICTIONS: 'ComTxIPduTimeoutContext' is only available, if 'ComTxTimeoutForTxModeNoneSupport' is set to 'true'.</p>
Tx Mode Mode	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeTrue.ComTxMode.ComTxModeMode'</p> <p>The transmission mode determines the Tx properties of the IPdu:</p> <ul style="list-style-type: none"> <li>- DIRECT: The message is transmitted in the next Com_MainFunctionTx() call after writing to a signal.</li> <li>- PERIODIC: The IPdu is transmitted with the specified cycle time (ComTxModeTimePeriodFactor).</li> <li>- MIXED: Combines the transmission modes of CYCLIC and DIRECT.</li> <li>- NONE: COM does not transmit these IPdus. A transmission must be triggered by Com_PduRTriggerTransmit().</li> </ul> <p>All transmission modes, except NONE, maintain the Minimum Delay Time (ComTxIPduMinimumDelayTimeFactor) and delay a transmission if required.</p>
Tx Mode Mode	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeFalse.ComTxMode.ComTxModeMode'</p> <p>The transmission mode determines the Tx properties of the IPdu:</p> <ul style="list-style-type: none"> <li>- DIRECT: The message is transmitted in the next Com_MainFunctionTx() call after writing to a signal.</li> <li>- PERIODIC: The IPdu is transmitted with the specified cycle time (ComTxModeTimePeriodFactor).</li> <li>- MIXED: Combines the transmission modes of CYCLIC and DIRECT.</li> <li>- NONE: COM does not transmit these IPdus. A transmission must be triggered by Com_PduRTriggerTransmit().</li> </ul> <p>All transmission modes, except NONE, maintain the</p>

Attribute Name	Value Type	Default Value	Description
			Minimum Delay Time (ComTxIPduMinimumDelayTimeFactor) and delay a transmission if required.
Tx Mode Number Of Repetitions	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeTrue.ComTxMode.ComTxModeNumberOfRepetitions'</p> <p>If the transmission mode is DIRECT or MIXED this attribute specifies the number of repetitions for this IPdu. If the value is 0 the transmit request is triggered once but the confirmation is not evaluated.</p>
Tx Mode Repetition Period Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeTrue.ComTxMode.ComTxModeRepetitionPeriodFactor'</p> <p>If the transmission mode is DIRECT or MIXED this attribute specifies cycle time for the repetitions specified by ComTxModeNumberOfRepetitions.</p>
Tx Mode Time Offset Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeTrue.ComTxMode.ComTxModeTimeOffsetFactor'</p> <p>If the transmission mode is PERIODIC or MIXED this attribute specifies the time offset of the first transmission after enabling the IPdu. The value must be a multiple of the Com_MainFunctionTx() cycle time.</p>
Tx Mode Time Period Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeTrue.ComTxMode.ComTxModeTimePeriodFactor'</p> <p>If the transmission mode is PERIODIC or MIXED this attribute specifies the message cycle time. If the transmission mode is NONE, the cycle time is not evaluated. The value must be a multiple of the Com_MainFunctionTx() cycle time.</p>
Tx Mode Number Of Repetitions	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeFalse.ComTxMode.ComTxModeNumberOfRepetitions'</p> <p>If the transmission mode is DIRECT or MIXED this attribute specifies the number of repetitions for this IPdu. If the value is 0 the transmit request is triggered once but the confirmation is not evaluated.</p>
Tx Mode Repetition Period Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeFalse.ComTxMode.ComTxModeRepetitionPeriodFactor'</p>

Attribute Name	Value Type	Default Value	Description
			If the transmission mode is DIRECT or MIXED this attribute specifies cycle time for the repetitions specified by ComTxModeNumberOfRepetitions.
Tx Mode Time Offset Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeFalse.ComTxMode.ComTxModeTimeOffsetFactor'</p> <p>If the transmission mode is PERIODIC or MIXED this attribute specifies the time offset of the first transmission after enabling the IPdu. The value must be a multiple of the Com_MainFunctionTx() cycle time.</p>
Tx Mode Time Period Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComTxModeFalse.ComTxMode.ComTxModeTimePeriodFactor'</p> <p>If the transmission mode is PERIODIC or MIXED this attribute specifies the message cycle time. If the transmission mode is NONE, the cycle time is not evaluated. The value must be a multiple of the Com_MainFunctionTx() cycle time.</p>
ComSignal			
Bit Position	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComBitPosition'</p> <p>Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.</p> <p>If the signal is a Motorola signal this indicates the MSB.</p> <p>If the byte order is a Intel signal this attribute indicates the LSB.</p>
Bit Size	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComBitSize'</p> <p>Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.</p>
Handle Id	Int	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComHandleId'</p> <p>The numerical value used as the ID.</p> <p>For signals it is required by the API calls Com_UpdateShadowSignal(), Com_ReceiveShadowSignal() and Com_InvalidateShadowSignal().</p> <p>For signals groups it is required by the Com_SendSignalGroup() and Com_ReceiveSignalGroup() calls.</p>

Attribute Name	Value Type	Default Value	Description
Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComNotification'</p> <p>Enter a function name if you require a Rx indication or Tx confirmation for this ComSignal or ComSignalGroup. If the field is left empty no function will be configured. The call context depends on the settings made in ComIPduSignalProcessing which can either be 'DEFERRED' or 'IMMEDIATE'.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Signal Endianness	Enum	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComSignalEndianness'</p> <p>Specifies the byte order that is used for this ComSignal or ComGroupSignal. OPAQUE indicates that no byte order conversion is performed (e.g. for byte arrays).</p>
Signal Init Value	String	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComSignalInitValue'</p> <p>Specify the initial value of the ComSignal or ComGroupSignal that is used as value after Com_Init().</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p> <p>Large signals (&gt;64 bit) are initialized with the pattern '0xFF'.</p>
Enable	Bool	true	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of 'ComSignal.ComSignalDataInvalidValue'</p> <p>Indicates whether the ComSignal has a configured invalid value or not.</p> <p>If enabled, the ComSignal has a configured invalid value. If disabled, the ComSignal has no configured invalid value.</p>
Signal Data Invalid Value	String	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComSignalDataInvalidValue'</p> <p>Specifies the invalid value of the ComSignal or ComGroupSignal.</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p>
Signal Length [bytes]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComSignalLength'</p>

Attribute Name	Value Type	Default Value	Description
			Indicates the signal length in bytes. Depending on the signal size not all bits might be sent / received using external communication. If the signal type (ComSignalType) is unit8[n] the number of bytes must be equal to the number of bits indicated in the signal size (ComBitSize).
Signal Type	Enum	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComSignalType'</p> <p>Specify the application data type of the signal. If UINT8[n] (byte array) is selected the signal size must have the same size as the signal specified in the ECU description or the database.</p>
First Timeout Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFirstTimeoutFactor'</p> <p>First timeout time that is used for Rx deadline monitoring. '0' for the first timeout indicates that the signal shall not be timeout observed. If Update Bits are not used, the shortest first timeout time of all signals of an IPdu is used. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.</p>
Timeout Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComTimeoutFactor'</p> <p>Timeout time that is used for reception or transmission deadline monitoring.</p> <p>'0' indicates that the signal shall not be timeout observed. The shortest timeout time of all signals and signal groups of an IPdu is used. Rx signals and Rx signal groups with a configured update bit are not evaluated for the I-Pdu based reception timeout time. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.</p>
Timeout Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComTimeoutNotification'</p> <p>Enter a function name if you require a timeout notification for this signal.</p> <p>If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Transfer Property	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComTransferProperty'</p>

Attribute Name	Value Type	Default Value	Description
			<p>The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the signal can either be</p> <ul style="list-style-type: none"> <li>- TRIGGERED: the transmission of the message is triggered if the signal is written regardless of the signal value</li> <li>- TRIGGERED_ON_CHANGE: the transmission of the message is triggered if the signal is written and the signal value has changed</li> <li>- PENDING: writing to the signal does not cause direct transmission.</li> </ul> <p>RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComUpdateBitPosition</p> <p>Activate the feature to activate an update bit for this ComSignal.</p> <p>Deactivate the feature to deactivate an update bit for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasUpdateBit' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
Update Bit Position	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComUpdateBitPosition'</p> <p>The update bit is located in the same IPdu. It is used to signalize that the signal was updated since the previous IPdu transmission. The update bit position must not be used by any other signal.</p> <p>RESTRICTIONS: 'ComUpdateBitPosition' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
Issm Signal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComIssmSignal'</p> <p>Activate the feature to indicate, that the signal is handled by SysService_Issm and not by the application.</p> <p>Deactivate the feature to use the ComSignal with the application.</p>
Internal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignal.ComInternal'</p>

Attribute Name	Value Type	Default Value	Description
			<p>If enabled, the signal ID is assigned dynamically at post-build time. This requires the user of the signal (e.g. the application, RTE, gateway) to support the post-build concept as well.</p> <p>If disabled, the signal ID will be assigned statically and will not change at post-build time. This setting is typically required for signals used by the application and RTE.</p> <p>Enable this switch solely for signals that are used by components that support the post-build concept. E.g. signals that are handled by the gateway component only. Signals that have this feature enabled can be removed and added at post-build time without causing the signal ID of other non post-build signals to be changed.</p> <p>Signals that do not have this property must not be removed or added at post-build time as this can have side effects on the signal ID of other (non post-build) signal IDs.</p>
Value	String	0	<p>MICROSAR PARAMETER DEFINITION: 'ComSignal.ComMissingSourceValue'</p> <p>Specify the missing source value of the ComGwSignal that is used as value after a timeout of ComGwSource.</p> <p>RESTRICTIONS: 'ComMissingSourceValue' is only available, if 'ComMissingSourceValueHandling' is set to 'true'.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComMissingSourceValue</p> <p>Set the check box to activate the missing source value for this ComSignal.</p> <p>Clear the check box to deactivate the missing source value for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasMissingSourceValue' is only available, if 'ComEnableMissingSourceValueHandling' is set to 'true'.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute enables the signal conversion. The signal conversion is used to convert signal values from BUS representation to ECU internal representation and vice versa. NOTE: Signal conversion is not specified by AUTOSAR.</p>
Source	Enum	0	<p>MICROSAR PARAMETER DEFINITION: N.a.</p>

Attribute Name	Value Type	Default Value	Description
			This attribute specifies the source for the conversion information from BUS to physical signal data representation. When selecting "Database" the factor and offset values are read from the database file. By selecting "Manual" the values for factor and offset can be manually changed.
Factor	Float	1	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the scaling factor for the conversion from BUS to physical data representation.
Offset	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the offset for the conversion from BUS to physical data representation.
ASAP2-DB	Object	N.a.	MICROSAR PARAMETER DEFINITION: N.a.  This attribute provides a list of all known transformation rules for conversion from physical to ECU internal signal data representation. To use this feature an ASAP2-DB has to be imported (see Com attribute ASAP2-DB).  If no predefined transformation rule is selected the values for factor and offset can be manually specified.
Factor	Float	1	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the scaling factor for the conversion from physical to ECU internal data representation.
Offset	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the offset for the conversion from physical to ECU internal data representation.
Factor	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated scaling factor for the conversion from BUS to ECU internal data representation.
Offset	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated offset for the conversion from BUS to ECU internal data representation.
Quality	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated quality for the conversion from BUS to ECU internal data representation.



Attribute Name	Value Type	Default Value	Description
			The quality is calculated as the quotient of the integer result and the floating point result. The quality is in the range from 0.0 to 1.0 where 1.0 is best.
ComSignal > ComRxSignal			
Data Invalid Action	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComDataInvalidAction'</p> <p>This parameter defines the action performed upon reception of an invalid signal. If Replace is used the ComSignalInitValue will be used for the replacement.</p>
Invalid Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComInvalidNotification'</p> <p>Enter a function name if you require a Rx invalid value notification for this signal. If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Rx Data Timeout Action	Bool	false	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComRxDataTimeoutAction'</p> <p>Activate this feature to write the ComSignalInitValue to the ComSignal or ComSignalGroup (REPLACE).</p> <p>Deactivate this feature to disable the timeout default value for this ComSignal or ComSignalGroup (NONE).</p>
Indication Flag	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignal.ComNotificationFlag'</p> <p>Activate the feature to generate a macro to read and clear a indication flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>Note: The flag is set in the call context of Com_RxIndication.</p> <p>RESTRICTIONS: 'ComNotificationFlag' is only available, if 'ComEnableIndicationFlags' is set to 'true'.</p>
Timeout Flag	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignal.ComTimeoutNotificationFlag'</p> <p>Activate the feature to generate a macro to read and clear a timeout flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this</p>

Attribute Name	Value Type	Default Value	Description
			ComSignal or ComSignalGroup.  RESTRICTIONS: 'ComTimeoutNotificationFlag' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.
State On Flag	Bool	false	MICROSAR PARAMETER DEFINITION: 'ComSignal.ComStateOnFlag'  Activate the feature to generate a macro to read a state on flag for the ComSignal or ComSignalGroup.  Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.  RESTRICTIONS: 'ComStateOnFlag' is only available, if 'ComEnableStateOnFlags' is set to 'true'.
Filter Algorithm	Enum	0	AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterAlgorithm'  The supported filter algorithms for Rx signals are: - None: The signal has no filter (Multiplicity of ComFilter) - Always: This filter always evaluates to TRUE - MaskedNewDiffersMaskedOld: ((new & mask) != (old & mask))
Filter Mask	Int	-1	AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMask'  The specified mask is used for the filters: - MaskedNewDiffersMaskedOld: ((new & mask) != (old & mask)) - MaskedNewEqualsX: ((new & mask) == x) - MaskedNewDiffersX: ((new & mask) != x)
Filter Max	Int	0	AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMax'  The specified max value is used for the filters: - NewIsOutside: ((new < min)    (max < new)) - NewIsWithin: ((min <= new) && (new <= max))
Filter Min	Int	0	AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMin'  The specified min value is used for the filters: - NewIsOutside: ((new < min)    (max < new))

Attribute Name	Value Type	Default Value	Description
			- NewIsWithin: ((min <= new) && (new <= max))
Filter X	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterX'</p> <p>The specified value X is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> </ul>
ComISignal > ComTxISignal			
Error Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComErrorNotification'</p> <p>Enter a function name if you require a Tx error notification for this signal. If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Filter Algorithm	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterAlgorithm'</p> <p>The supported filter algorithms for Tx signals are:</p> <ul style="list-style-type: none"> <li>- None: The signal has no filter (Multiplicity of ComFilter)</li> <li>- Always: This filter always evaluates to TRUE</li> <li>- Never: This filter always evaluates to FALSE</li> <li>- MaskedNewDiffersMaskedOld: ((new &amp; mask) != (old &amp; mask))</li> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter Mask	Int	-1	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMask'</p> <p>The specified mask is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewDiffersMaskedOld: ((new &amp; mask) != (old &amp; mask))</li> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> </ul>
Filter Max	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMax'</p>

Attribute Name	Value Type	Default Value	Description
			<p>The specified max value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter Min	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterMin'</p> <p>The specified min value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter X	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignal.ComFilter.ComFilterX'</p> <p>The specified value X is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> </ul>
ComSignalGroup			
Bit Position	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComBitPosition'</p> <p>Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.</p> <p>If the signal is a Motorola signal this indicates the MSB.</p> <p>If the byte order is a Intel signal this attribute indicates the LSB.</p>
Bit Size	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComBitSize'</p> <p>Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.</p>
Handle Id	Int	N.a.	<p>MICROSAR PARAMETER DEFINITION: ComHandleId</p> <p>The numerical value used as the ID.</p> <p>For signals it is required by the API calls Com_UpdateShadowSignal(), Com_ReceiveShadowSignal() and Com_InvalidateShadowSignal().</p> <p>For signals groups it is required by the Com_SendSignalGroup() and Com_ReceiveSignalGroup() calls.</p>
Notification	String	N.a.	AUTOSAR PARAMETER DEFINITION:

Attribute Name	Value Type	Default Value	Description
			<p>'ComSignalGroup.ComNotification'</p> <p>Enter a function name if you require a Rx indication or Tx confirmation for this ComSignal or ComSignalGroup. If the field is left empty no function will be configured. The call context depends on the settings made in ComIPduSignalProcessing which can either be 'DEFERRED' or 'IMMEDIATE'.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Signal Endianness	Enum	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalEndianness'</p> <p>Specify the byte order that is used for this ComSignal or ComGroupSignal. OPAQUE indicates that no byte order conversion is performed (e.g. for byte arrays).</p>
Signal Init Value	String	0	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalInitValue'</p> <p>Specify the initial value of the ComSignal or ComGroupSignal that is used as value after Com_Init().</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p> <p>Large signals (&gt;64 bit) are initialized with the pattern '0xFF'.</p>
Signal Length [bytes]	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalLength'</p> <p>Indicates the signal length in bytes. Depending on the signal size not all bits might be sent / received using external communication. If the signal type (ComSignalType) is unit8[n] the number of bytes must be equal to the number of bits indicated in the signal size (ComBitSize).</p>
Signal Type	Enum	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalType'</p> <p>Specify the application data type of the signal. If UINT8[n] (byte array) is selected the signal size must have the same size as the signal specified in the ECU description or the database.</p>
First Timeout Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComFirstTimeoutFactor'</p> <p>First timeout time that is used for Rx deadline monitoring.</p>

Attribute Name	Value Type	Default Value	Description
			'0' for the first timeout indicates that the signal shall not be timeout observed. If Update Bits are not used, the shortest first timeout time of all signals of an IPdu is used. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.
Timeout Factor [ms]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComTimeoutFactor'</p> <p>Timeout time that is used for reception or transmission deadline monitoring.</p> <p>'0' indicates that the signal shall not be timeout observed. The shortest timeout time of all signals and signal groups of an IPdu is used. Rx signals and Rx signal groups with a configured update bit are not evaluated for the I-Pdu based reception timeout time. The value must be a multiple of the Com_MainFunctionRx() and Com_MainFunctionTx() cycle times.</p>
Timeout Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComTimeoutNotification'</p> <p>Enter a function name if you require a timeout notification for this signal.</p> <p>If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Transfer Property	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComTransferProperty'</p> <p>The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the signal group can either be</p> <ul style="list-style-type: none"> <li>- TRIGGERED: the transmission of the message is triggered if at least one group signal of signal group is written regardless of the group signal value</li> <li>- TRIGGERED_ON_CHANGE: the transmission of the message is triggered if at least one group signal of signal group is written and the group signal value has changed</li> <li>- PENDING: writing to the signal group does not cause direct transmission.</li> </ul> <p>Note: If at least one group signal of the signal group has a defined transfer property equal to TRIGGERED or TRIGGERED_ON_CHANGE, the transfer property of the signal group will be ignored and the trigger will be evaluated per group signal.</p>

Attribute Name	Value Type	Default Value	Description
			RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComUpdateBitPosition</p> <p>Activate the feature to activate an update bit for this ComSignal.</p> <p>Deactivate the feature to deactivate an update bit for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasUpdateBit' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
Update Bit Position	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComUpdateBitPosition'</p> <p>The update bit is located in the same IPdu. It is used to signalize that the signal group was updated since the previous IPdu transmission. The update bit position must not be used by any other signal group.</p> <p>RESTRICTIONS: 'ComUpdateBitPosition' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
Issm Signal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComIssmSignal'</p> <p>Activate the feature to indicate, that the signal is handled by SysService_Issm and not by the application.</p> <p>Deactivate the feature to use the ComSignal with the application.</p>
Filter Algorithm	Enum	0	<p>MICROSAR PARAMETER DEFINITION: 'ComFilterAlgorithm'</p> <p>The supported filter algorithms are:</p> <ul style="list-style-type: none"> <li>- None: The signal has no filter (Multiplicity of ComFilter)</li> <li>- Always: This filter always evaluates to TRUE</li> <li>- Never: This filter always evaluates to FALSE</li> <li>- MaskedNewDiffersMaskedOld: <math>((\text{new} \&amp; \text{mask}) \neq (\text{old} \&amp; \text{mask}))</math></li> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul>

Attribute Name	Value Type	Default Value	Description
Filter Mask	Int	-1	<p>MICROSAR PARAMETER DEFINITION: 'ComFilterMask'</p> <p>The specified mask is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewDiffersMaskedOld: <math>((\text{new} \&amp; \text{mask}) \neq (\text{old} \&amp; \text{mask}))</math></li> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> </ul>
Filter Max	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComFilterMax'</p> <p>The specified max value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul>
Filter Min	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComFilterMin'</p> <p>The specified min value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: <math>((\text{new} &lt; \text{min}) \parallel (\text{max} &lt; \text{new}))</math></li> <li>- NewIsWithin: <math>((\text{min} \leq \text{new}) \&amp;\&amp; (\text{new} \leq \text{max}))</math></li> </ul>
Filter X	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComFilterX'</p> <p>The specified value X is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewEqualsX: <math>((\text{new} \&amp; \text{mask}) == x)</math></li> <li>- MaskedNewDiffersX: <math>((\text{new} \&amp; \text{mask}) \neq x)</math></li> </ul>
Internal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalGroup.ComInternal'</p> <p>If enabled, the signal ID is assigned dynamically at post-build time. This requires the user of the signal (e.g. the application, RTE, gateway) to support the post-build concept as well.</p> <p>If disabled, the signal ID will be assigned statically and will not change at post-build time. This setting is typically required for signals used by the application and RTE.</p> <p>Enable this switch solely for signals that are used by components that support the post-build concept. E.g. signals that are handled by the gateway component only. Signals that have this feature enabled can be removed and added at post-build time without causing the signal ID of other non post-build signals to be changed.</p> <p>Signals that do not have this property must not be removed or added at post-build time as this can have side effects on the signal ID of other (non post-build) signal IDs.</p>



Attribute Name	Value Type	Default Value	Description
Value	String	0	<p>MICROSAR PARAMETER DEFINITION: 'ComMissingSourceValue'</p> <p>Specify the missing source value of the ComGwSignal that is used as value after a timeout of ComGwSource.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComMissingSourceValue</p> <p>Activate the feature to enable the missing source value for this ComSignal.</p> <p>Deactivate the feature to disable the missing source value for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasMissingSourceValue' is only available, if 'ComEnableMissingSourceValueHandling' is set to 'true'.</p>
ComISignalGroup > ComRxISignalGroup			
Data Invalid Action	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComDataInvalidAction'</p> <p>This parameter defines the action performed upon reception of an invalid value of one of the included signals. If Replace is used the ComSignalInitValue will be used for the replacement.</p>
Invalid Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComInvalidNotification'</p> <p>Enter a function name if you require a Rx invalid value notification for this signal group. If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
Rx Data Timeout Action	Bool	false	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComRxDataTimeoutAction'</p> <p>Activate the feature to write the ComSignalInitValue to the ComSignal or ComSignalGroup (REPLACE).</p> <p>Deactivate the feature to disable the timeout default value for this ComSignal or ComSignalGroup (NONE).</p>
Indication Flag	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalGroup.ComNotificationFlag'</p>

Attribute Name	Value Type	Default Value	Description
			<p>Activate the feature to generate a macro to read and clear a indication flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>Note: The flag is set in the call context of Com_RxIndication.</p> <p>RESTRICTIONS: 'ComNotificationFlag' is only available, if 'ComEnableIndicationFlags' is set to 'true'.</p>
Timeout Flag	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalGroup.ComTimeoutNotificationFlag'</p> <p>Activate the feature to generate a macro to read and clear a timeout flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>RESTRICTIONS: 'ComTimeoutNotificationFlag' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
State On Flag	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalGroup.ComStateOnFlag'</p> <p>Activate the feature to generate a macro to read a state on flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the feature for this ComSignal or ComSignalGroup.</p> <p>RESTRICTIONS: 'ComStateOnFlag' is only available, if 'ComEnableStateOnFlags' is set to 'true'.</p>
ComISignalGroup > ComTxISignalGroup			
Error Notification	String	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComSignalGroup.ComErrorNotification'</p> <p>Enter a function name if you require a Tx error notification for this signal. If the field is left empty no function will be configured.</p> <p>The prototype void _Name_(void) will be generated to Rte_Cbk.h or Appl_Cbk.h and has to be implemented by the application.</p>
ComIGroupSignal			
Bit Position	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComBitPosition'</p>

Attribute Name	Value Type	Default Value	Description
			<p>Specifies the start bit of the ComSignal, ComSignalGroup or ComGroupSignal.</p> <p>If the signal is a Motorola signal this indicates the MSB.</p> <p>If the byte order is a Intel signal this attribute indicates the LSB.</p>
Bit Size	Int	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComBitSize'</p> <p>Specifies the size of the ComSignal, ComSignalGroup or ComGroupSignal in bit.</p>
Handle Id	Int	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComHandleId'</p> <p>The numerical value used as the ID.</p> <p>For signals it is required by the API calls Com_UpdateShadowSignal(), Com_ReceiveShadowSignal() and Com_InvalidateShadowSignal().</p> <p>For signals groups it is required by the Com_SendSignalGroup() and Com_ReceiveSignalGroup() calls.</p>
Signal Endianess	Enum	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComSignalEndianess'</p> <p>Specifies the byte order that is used for this ComSignal or ComGroupSignal. OPAQUE indicates that no byte order conversion is performed (e.g. for byte arrays).</p>
Signal Init Value	String	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComSignalInitValue'</p> <p>Specify the initial value of the ComSignal or ComGroupSignal that is used as value after Com_Init().</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p> <p>Large signals (&gt;64 bit) are initialized with the pattern '0xFF'.</p>
Enable	Bool	true	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of 'ComGroupSignal.ComSignalDataInvalidValue'</p> <p>Indicates whether the ComGroupSignal has a configured invalid value or not.</p> <p>If enabled, the ComGroupSignal has a configured invalid value.</p>

Attribute Name	Value Type	Default Value	Description
			If disabled, the ComGroupSignal has no configured invalid value.
Signal Data Invalid Value	String	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComSignalDataInvalidValue'</p> <p>Specifies the invalid value of the ComSignal or ComGroupSignal.</p> <p>To configure negative values, enter a HEX value in the two's complement with a bit count of the ComSignalType.</p>
Signal Length [bytes]	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComSignalLength'</p> <p>Indicates the signal length in bytes. Depending on the signal size not all bits might be sent / received using external communication. If the signal type (ComSignalType) is unit8[n] the number of bytes must be equal to the number of bits indicated in the signal size (ComBitSize).</p>
Signal Type	Enum	N.a.	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComSignalType'</p> <p>Specify the application data type of the signal. If UINT8[n] (byte array) is selected the signal size must have the same size as the signal specified in the ECU description or the database.</p>
Transfer Property	Enum	0	<p>MICROSAR PARAMETER DEFINITION: 'ComGroupSignal.ComTransferProperty'</p> <p>The transfer behavior for the ComTxModeMode DIRECT or MIXED when writing to the group signal can either be</p> <ul style="list-style-type: none"> <li>- TRIGGERED: the transmission of the message is triggered if the group signal is written regardless of the group signal value</li> <li>- TRIGGERED_ON_CHANGE: the transmission of the message is triggered if the group signal is written and the group signal value has changed</li> <li>- PENDING: writing to the group signal does not cause direct transmission.</li> </ul> <p>Note: If all the transfer properties of the group signals of a signal group equals PENDING, the transfer property of the signal group will be evaluated for the group signals.</p> <p>RESTRICTIONS: 'TRIGGERED_ON_CHANGE' is only available, if 'ComTmsSupport' is set to 'true'.</p>

Attribute Name	Value Type	Default Value	Description
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComUpdateBitPosition</p> <p>Activate the feature to activate an update bit for this ComSignal.</p> <p>Deactivate the feature to deactivate an update bit for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasUpdateBit' is only available, if 'ComEnableUpdateBitSupport' is set to 'true'.</p>
Update Bit Position	Int	0	<p>MICROSAR PARAMETER DEFINITION: 'ComUpdateBitPosition'</p> <p>The update bit is located in the same IPdu. It is used to signalize that the signal was updated since the previous IPdu transmission. The update bit position must not be used by any other signal.</p>
Issm Signal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComIssmSignal'</p> <p>Activate the feature to indicate, that the signal is handled by SysService_Issm and not by the application.</p> <p>Deactivate the feature to use the ComSignal with the application.</p>
Internal	Bool	false	<p>MICROSAR PARAMETER DEFINITION: 'ComInternal'</p> <p>If enabled, the signal ID is assigned dynamically at post-build time. This requires the user of the signal (e.g. the application, RTE, gateway) to support the post-build concept as well.</p> <p>If disabled, the signal ID will be assigned statically and will not change at post-build time. This setting is typically required for signals used by the application and RTE.</p> <p>Enable this switch solely for signals that are used by components that support the post-build concept. E.g. signals that are handled by the gateway component only. Signals that have this feature enabled can be removed and added at post-build time without causing the signal ID of other non post-build signals to be changed.</p> <p>Signals that do not have this property must not be removed or added at post-build time as this can have side effects on the signal ID of other (non post-build) signal IDs.</p>
Value	String	0	<p>MICROSAR PARAMETER DEFINITION: 'ComGroupSignal.ComMissingSourceValue'</p> <p>Specify the missing source value of the ComGwSignal that</p>

Attribute Name	Value Type	Default Value	Description
			<p>is used as value after a timeout of ComGwSource.</p> <p>RESTRICTIONS: 'ComMissingSourceValue' is only available, if 'ComMissingSourceValueHandling' is set to 'true'.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: Multiplicity of ComMissingSourceValue</p> <p>Activate the feature to activate the missing source value for this ComSignal.</p> <p>Deactivate the feature to deactivate the missing source value for this ComSignal.</p> <p>RESTRICTIONS: 'ComHasMissingSourceValue' is only available, if 'ComEnableMissingSourceValueHandling' is set to 'true'.</p>
Enable	Bool	false	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute enables the signal conversion. The signal conversion is used to convert signal values from BUS representation to ECU internal representation and vice versa. NOTE: Signal conversion is not specified by AUTOSAR.</p>
Source	Enum	0	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute specifies the source for the conversion information from BUS to physical signal data representation. When selecting "Database" the factor and offset values are read from the database file. By selecting "Manual" the values for factor and offset can be manually changed.</p>
Factor	Float	1	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute specifies the scaling factor for the conversion from BUS to physical data representation.</p>
Offset	Float	0	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute specifies the offset for the conversion from BUS to physical data representation.</p>
ASAP2-DB	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute provides a list of all known transformation rules for conversion from physical to ECU internal signal data representation. To use this feature an ASAP2-DB has</p>

Attribute Name	Value Type	Default Value	Description
			to be imported (see Com attribute ASAP2-DB). If no predefined transformation rule is selected the values for factor and offset can be manually specified.
Factor	Float	1	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the scaling factor for the conversion from physical to ECU internal data representation.
Offset	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute specifies the offset for the conversion from physical to ECU internal data representation.
Factor	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated scaling factor for the conversion from BUS to ECU internal data representation.
Offset	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated offset for the conversion from BUS to ECU internal data representation.
Quality	Float	0	MICROSAR PARAMETER DEFINITION: N.a.  This attribute shows the calculated quality for the conversion from BUS to ECU internal data representation. The quality is calculated as the quotient of the integer result and the floating point result. The quality is in the range from 0.0 to 1.0 where 1.0 is best.
ComIGroupSignal > ComRxIGroupSignal			
Filter Algorithm	Enum	0	AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterAlgorithm'  The supported filter algorithms for Rx signals are: - None: The signal has no filter (Multiplicity of ComFilter) - Always: This filter always evaluates to TRUE - MaskedNewDiffersMaskedOld: ((new & mask) != (old & mask))
Filter Mask	Int	-1	AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMask'  The specified mask is used for the filters: - MaskedNewDiffersMaskedOld: ((new & mask) != (old & mask))

Attribute Name	Value Type	Default Value	Description
			<ul style="list-style-type: none"> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> </ul>
Filter Max	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMax'</p> <p>The specified max value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter Min	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMin'</p> <p>The specified min value is used for the filters:</p> <ul style="list-style-type: none"> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter X	Int	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterX'</p> <p>The specified value X is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> </ul>
ComIGroupSignal > ComTxIGroupSignal			
Filter Algorithm	Enum	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterAlgorithm'</p> <p>The supported filter algorithms for Tx signals are:</p> <ul style="list-style-type: none"> <li>- None: The signal has no filter (Multiplicity of ComFilter)</li> <li>- Always: This filter always evaluates to TRUE</li> <li>- Never: This filter always evaluates to FALSE</li> <li>- MaskedNewDiffersMaskedOld: ((new &amp; mask) != (old &amp; mask))</li> <li>- MaskedNewEqualsX: ((new &amp; mask) == x)</li> <li>- MaskedNewDiffersX: ((new &amp; mask) != x)</li> <li>- NewIsOutside: ((new &lt; min)    (max &lt; new))</li> <li>- NewIsWithin: ((min &lt;= new) &amp;&amp; (new &lt;= max))</li> </ul>
Filter Mask	Int	-1	<p>AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMask'</p> <p>The specified mask is used for the filters:</p> <ul style="list-style-type: none"> <li>- MaskedNewDiffersMaskedOld: ((new &amp; mask) != (old &amp;</li> </ul>



Attribute Name	Value Type	Default Value	Description
			mask)) - MaskedNewEqualsX: ((new & mask) == x) - MaskedNewDiffersX: ((new & mask) != x)
Filter Max	Int	0	AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMax'  The specified max value is used for the filters: - NewIsOutside: ((new < min)    (max < new)) - NewIsWithin: ((min <= new) && (new <= max))
Filter Min	Int	0	AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterMin'  The specified min value is used for the filters: - NewIsOutside: ((new < min)    (max < new)) - NewIsWithin: ((min <= new) && (new <= max))
Filter X	Int	0	AUTOSAR PARAMETER DEFINITION: 'ComGroupSignal.ComFilter.ComFilterX'  The specified value X is used for the filters: - MaskedNewEqualsX: ((new & mask) == x) - MaskedNewDiffersX: ((new & mask) != x)
ComIPduGroup			
IPdu Group Name	String	N.a.	MICROSAR PARAMETER DEFINITION: 'ComIPduGroup'  You can specify a name for each IPdu Group. The name must be unique as it is used as short name for the ComIPduGroup container.
IPdu Group Handle Id	Int	N.a.	AUTOSAR PARAMETER DEFINITION: 'ComIPduGroupHandleId'  The numerical value used as the ID of this IPdu Group. The ComIPduGroupHandleId is required by the API calls to start and stop IPdu Groups.
Parent IPdu Group Ref	Object	N.a.	AUTOSAR PARAMETER DEFINITION: 'ComIPduGroupGroupRef'  If the I-PDU Group belongs to an I-PDU group, this is the name of the I-PDU group it belongs to. This I-PDU Group does not belong to another I-PDU group, if this reference is omitted.
ComGwMapping			

Attribute Name	Value Type	Default Value	Description
Generate	Bool	true	<p>MICROSAR PARAMETER DEFINITION: 'ComGwMapping.ComGenerate'</p> <p>The switch can be used to temporarily deactivate a Gw Mapping relation.</p>
Gw Mapping Name	String	SignalRoutingRelation	<p>MICROSAR PARAMETER DEFINITION: 'ComGwMapping'</p> <p>You can specify a name for each Gw Mapping. The name must be unique as it is used as short name for the ComGwMapping container.</p>
Channel	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Select the source channel of the Gw Mapping relation. This channel serves as a filter for the source signal selection. If no channel is selected all signals from all channels can be selected.</p>
Pdu	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Select the source IPdu of the Gw Mapping. This IPdu serves as a filter for the source signal selection. If no IPdu is selected all signals from all IPdus can be selected.</p>
Gw Signal	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComGwSource.ComGwSignal'</p> <p>Select the source signal of the Gw Mapping. If there are several signals with the same name in your databases, the unique name is used as specified by the "NameDecorator" component (i.e. the channel index is added to the signal name).</p>
Channel	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Select the destination channel of the Gw Mapping. This channel serves as a filter for the destination signal selection. If no channel is selected all signals from all channels can be selected.</p>
Pdu	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Select the destination IPdu of the Gw Mapping. This IPdu serves as a filter for the destination signal selection. If no IPdu is selected all signals from all IPdus can be selected.</p>
Gw Signal	Object	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComGwDestination.ComGwSignal'</p>

Attribute Name	Value Type	Default Value	Description
			Select the destination signal of the Gw Mapping. If there are several signals with the same name in your databases, the unique name is used as specified by the "NameDecorator" component (i.e. the channel index is added to the signal name).
ComObjectFilter			
Filter Pattern	String		<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>The filter pattern defines the objects to be displayed. All objects where this pattern matches the object's name are displayed.</p>
Recursive	Bool	true	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>This attribute enables the recursive pattern matching. This means an object is displayed if the pattern matches this object or any contained object. For example a ComSignalGroup is displayed if the pattern matches the ComSignalGroup itself or any ComGroupSignal contained in this ComSignalGroup.</p>
Whole Word Only	Bool	false	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>The pattern matches only if the object's name is equal to the pattern string.</p>
ModuleInstance			
Create Signal Routing Relation	Void	N.a.	<p>AUTOSAR PARAMETER DEFINITION: N.a.</p> <p>Creates a new signal routing relation. A blank routing relation is added to the routing relations tree where further settings have to be made.</p>
ModuleInstance			
Configuration Variant	Enum	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Select which configuration variant shall be applied to your BSW module.</p> <p>The supported variants are module dependent and are specified in the [TechnicalReference_Asr_Com.pdf]. Please refer to the [TechnicalReference_Asr_Com.pdf] for details about the configuration variants.</p>
Version Info Api	Boolean	true	<p>AUTOSAR PARAMETER DEFINITION: 'ComVersionInfoApi'</p> <p>Activate this feature to enable the function Com_GetVersionInfo() to get major, minor and patch version information.</p>

Attribute Name	Value Type	Default Value	Description
Dev Error Detect	Boolean	false	<p>AUTOSAR PARAMETER DEFINITION: 'ComConfigurationUseDet'</p> <p>If 'Development Error Detection' is enabled, all development errors are reported to the Development Error Tracer (DET). The errors are described in the [TechnicalReference_Asr_Com.pdf].</p> <p>Note: In general, the development error detection is recommended during pre-test phase. It is not recommended to enable the development error detection in production code due to increased runtime and ROM needs.</p>
Prod Error Detect	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComConfigurationUseDem'</p> <p>If 'Production Error Detection' is enabled, production relevant errors are reported to the Diagnostics Event Manager (DEM).</p> <p>Disable this option, if no DEM is present in your system.</p>
User Config File	String	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComUserConfigFile'</p> <p>A configuration file is generated by GENy. If you want to overwrite settings in the generated configuration file, you can specify a path to a user defined configuration file.</p> <p>The user defined configuration file will be included at the end of the generated file. Therefore definitions in the user defined configuration file can overwrite definitions in the generated configuration file.</p> <p>The content of the user defined configuration file is copied to the end of Com_Cfg.h.</p>
PB Start Address	Integer	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComPBStartAddress'</p> <p>This setting is mandatory when generating a configuration that will be written to the memory of your ECU at post-build time (i.e. when generating a hex-file). This setting is not relevant when generating the initial ECU configuration at link- or pre-compile time (i.e. when generating a C-file that will be linked with your project).</p> <p>The Module Start Address must specify the start address of the memory area, where the generated hex-file will be mapped to. If the ECU supports selectable post-build</p>

Attribute Name	Value Type	Default Value	Description
			<p>configuration, the Module Start Address can be any location that is accessible for the BSW module. If selectable post-build configuration is not supported by your ECU, the Module Start Address must be the same address to which the initial configuration was mapped. Please see into your linker map file to get this address.</p> <p>RESTRICTIONS: 'CompPBStartAddress' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
Max Number Of Rx IPdus	Integer	1	<p>MICROSAR PARAMETER DEFINITION: 'ComMaxNumberOfRxIPdus'</p> <p>The number of IPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled IPdus at post-build time memory must be reserved.</p> <p>Configure the maximum number of Rx IPdus, which shall be able to handle at any time. The minimum number is the currently configured number of Rx IPdus.</p> <p>RESTRICTIONS: 'ComMaxNumberOfRxIPdus' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
Max Number Of Tx IPdus	Integer	1	<p>MICROSAR PARAMETER DEFINITION: 'ComMaxNumberOfTxIPdus'</p> <p>The number of IPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled IPdus at post-build time memory must be reserved.</p> <p>Configure the maximum number of Tx IPdus, which shall be able to handle at any time. The minimum number is the currently configured number of Tx IPdus.</p> <p>RESTRICTIONS: 'ComMaxNumberOfTxIPdus' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
Max Number Of Rx Update Bits	Integer	1	<p>MICROSAR PARAMETER DEFINITION: 'ComMaxNumberOfRxUpdateBits'</p> <p>The number of Rx update bits in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled Rx update bits at post-build time memory must be reserved.</p> <p>Configure the maximum number of Rx update bits, which shall be able to handle at any time. The minimum number</p>

Attribute Name	Value Type	Default Value	Description
			<p>is the currently configured number of Rx update bits.</p> <p>RESTRICTIONS: 'ComMaxNumberOfRxUpdateBits' is only available, if 'ComEnableUpdateBitSupport' is set to 'true' and 'ComConfigurationVariant' is 'Variant 3 (Post-build Configuration)'</p>
Max Number Of Tx Signal Groups	Integer	1	<p>MICROSAR PARAMETER DEFINITION: 'ComMaxNumberOfTxSignalGroups'</p> <p>The number of Tx signal groups in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled Tx signal groups at post-build time memory must be reserved.</p> <p>Configure the maximum number of Tx signal groups, which shall be able to handle at any time. The minimum number is the currently configured number of Tx signal groups.</p> <p>RESTRICTIONS: 'ComMaxNumberOfTxSignalGroups' is only available, if 'ComEnableSignalGroupInterface' is set to 'true' and 'ComConfigurationVariant' is 'Variant 3 (Post-build Configuration)'</p>
Buffer Size [byte]	Integer	0	<p>MICROSAR PARAMETER DEFINITION: 'ComBufferSize'</p> <p>The number of ComPdus in an AUTOSAR system can be changed at post-build time (Variant 3). To allow increasing the number of handled ComPdus at post-build time memory must be reserved at link time.</p> <p>The size of the buffer depends on the current COM configuration and on the scalability the shall be available at post-build time.</p> <p>RESTRICTIONS: 'ComBufferSize' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>
Compute COM Buffer Size	Void	N.a.	<p>MICROSAR PARAMETER DEFINITION: N.a.</p> <p>Press this button to set the "COM Buffer Size" to the buffer size currently used by COM.</p> <p>If further memory shall be allocated for post-build time usage, this has to be added manually.</p> <p>RESTRICTIONS: 'ComComputeCurrentComBufferSizeAction' is only available, if 'ComConfigurationVariant' is set to 'Variant 3 (Post-build Configuration)'.</p>

Attribute Name	Value Type	Default Value	Description
Use Rte	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComUseRte'</p> <p>Enable this option if the ECU uses an RTE.</p> <p>If this option is disabled, callback prototypes are generated into the file Appl_Cbk.h.</p>
Delnit Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComDelnitApi'</p> <p>Activate the feature to enable the API Com_Delnit().</p> <p>Deactivate the feature to disable the API Com_Delnit().</p>
Reception DM Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComReceptionDMApi'</p> <p>Activate the feature to enable the API Com_EnableReceptionDM() and Com_DisableReceptionDM().</p> <p>Deactivate the feature to disable the API Com_EnableReceptionDM() and Com_DisableReceptionDM().</p>
Get Status Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComGetStatusApi'</p> <p>Activate the feature to enable the API Com_GetStatus().</p> <p>Deactivate the feature to disable the API Com_GetStatus().</p>
Trigger IPDU Send Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComTriggerIPDUSendApi'</p> <p>Activate the feature to enable the API Com_TriggerIPDUSend().</p> <p>Deactivate the feature to disable the API Com_TriggerIPDUSend().</p>
Trigger Transmit Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComTriggerTransmitApi'</p> <p>Activate the feature to enable the API Com_TriggerTransmit().</p> <p>Deactivate the feature to disable the API Com_TriggerTransmit().</p> <p>Note: This API is needed if the COM is used with LIN, Flexray or MOST.</p>

Attribute Name	Value Type	Default Value	Description
			RESTRICTIONS: 'ComTriggerTransmitApi' is only available, if 'ComConfigurationVariant' is not set to 'Variant 1 (Pre-compile Configuration)'.
Com_MainFunctionRx API	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComMainFunctionRxApi'</p> <p>Activate the feature to enable the API Com_MainFunctionRx().</p> <p>Deactivate the feature to disable the feature.</p>
Com_MainFunctionTx API	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComMainFunctionTxApi'</p> <p>Activate the feature to enable the API Com_MainFunctionTx().</p> <p>Deactivate the feature to disable the feature.</p>
Signal Invalidation Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalInvalidationApi'</p> <p>Activate the feature to enable the API Com_InvalidateSignal() and Com_InvalidateSignalGroup() and the handling of invalid values for Rx signals if configured.</p> <p>Deactivate the feature to disable the feature.</p>
Max Length Of Rx IPdus	Integer	8	<p>MICROSAR PARAMETER DEFINITION: 'ComMaxRxIPduLength'</p> <p>Configure the maximum length of Rx IPdus in bytes, which shall be able to handle at any time. The minimum length is the currently configured maximum length of Rx IPdus.</p> <p>RESTRICTIONS: 'ComMaxRxIPduLength' is not available, if 'ComConfigurationVariant' is set to 'Variant 1 (Pre-compile Configuration)'.</p>
Show Error 0005 as Warning	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComShowError0005AsWarning'</p> <p>Activate the feature to generate the Error message 0005 as a Warning.</p>
Get Configuration Id Api	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComGetConfigurationIdApi'</p> <p>Activate the feature to enable the API Com_GetConfigurationId().</p>



Attribute Name	Value Type	Default Value	Description
			Deactivate the feature to disable the API Com_GetConfigurationId().
Configuration Id	Integer	0	<p>AUTOSAR PARAMETER DEFINITION: 'ComConfigurationId'</p> <p>This ID provides the unique identifier of the post-build configurable configuration part. At runtime the configuration ID can be retrieved by Com_GetConfigurationId().</p>
Com_GetConfigurationStringPtr API	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComGetConfigurationStringPtrApi'</p> <p>Activate the feature to enable the API Com_GetConfigurationStringPtr().</p> <p>Deactivate the feature to disable the feature.</p>
Configuration String	String	N.a.	<p>MICROSAR PARAMETER DEFINITION: 'ComConfigurationString'</p> <p>This ID provides the unique identifier of the post-build configurable configuration part. At runtime the configuration ID can be retrieved by Com_GetConfigurationStringPtr().</p> <p>RESTRICTIONS: 'ComConfigurationString' is only available, if 'ComGetConfigurationStringPtrApi' is set to 'true'.</p>
Configuration Time Base [s]	Float	0.001	<p>AUTOSAR PARAMETER DEFINITION: 'ComConfigurationTimeBase'</p> <p>Configure the time base for time factors, which are saved in the EcuC file.</p>
Configuration Rx Time Base [s]	Float	0.01	<p>MICROSAR PARAMETER DEFINITION: 'ComConfigurationRxTimeBase'</p> <p>Configure the cycle time in which Com_MainFunctionRx() has to be called.</p>
Configuration Tx Time Base [s]	Float	0.01	<p>MICROSAR PARAMETER DEFINITION: 'ComConfigurationTxTimeBase'</p> <p>Configure the cycle time in which Com_MainFunctionTx() has to be called.</p>
Com_IpduGroupTransmit API	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComIPduGroupTransmit'</p>

Attribute Name	Value Type	Default Value	Description
			<p>Activate the feature to enable the API Com_IpduGroupTransmit().</p> <p>Deactivate the feature to disable the feature.</p>
Lower Layer Tx Confirmation Functions	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableLoTxConfirmation'</p> <p>Activate the feature to restart timeout counters based upon the call of Com_TxConfirmation().</p> <p>Deactivate the feature to restart timeout counters after the call of PduR_ComTransmit().</p> <p>This feature is necessary if confirmation functions are not supported by the PDUR.</p>
Indication Flag Interface	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableIndicationFlags'</p> <p>Activate the feature to enable the indication flag feature.</p> <p>Deactivate the feature to disable the feature.</p>
Protect Indication Flag Access	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComProtectIndicationFlags'</p> <p>Activate the feature to enable an interrupt protection in the generated clear macro for an indication flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the generated interrupt protection. The application has to ensure the consistency of the flag access.</p> <p>RESTRICTIONS: 'ComProtectIndicationFlags' is only available, if 'ComEnableIndicationFlags' is set to 'true'.</p>
Timeout Flag Interface	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableTimeoutFlags'</p> <p>Activate the feature to enable the timeout flag feature.</p> <p>Deactivate the feature to disable the feature.</p>
Protect Timeout Flag Access	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComProtectTimeoutFlags'</p> <p>Activate the feature to enable an interrupt protection in the generated clear macro for a timeout flag for the ComSignal or ComSignalGroup.</p> <p>Deactivate the feature to disable the generated interrupt</p>

Attribute Name	Value Type	Default Value	Description
			<p>protection. The application has to ensure the consistency of the flag access.</p> <p>RESTRICTIONS: 'ComProtectTimeoutFlags' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
Reset Timeout Flag	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComResetTimeoutFlags'</p> <p>Activate the feature to reset the timeout flag, if the ComSignal or ComSignalGroup is received again after a timeout occurred.</p> <p>Deactivate the feature to disable the reset of the timeout flag.</p> <p>RESTRICTIONS: 'ComResetTimeoutFlags' is only available, if 'ComEnableTimeoutFlags' is set to 'true'.</p>
State On Flag Interface	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableStateOnFlags'</p> <p>Activate the feature to enable the state on flag macro interface.</p> <p>Deactivate the feature to disable the state on flag macro interface.</p>
Signal Conversion Interface	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalConversionInterface'</p> <p>Activate the feature to enable the signal conversion API.</p> <p>Deactivate the feature to disable the feature.</p>
Large Transport Protocol IPdus	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComTransportProtocolSupport'</p> <p>Activate the feature to enable the API to transmit IPdus larger than 8 bytes on CAN via CAN TP.</p> <p>Deactivate the feature to disable the feature.</p>
Support for Signals larger 254bit	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComLargeSignalSupport'</p> <p>Activate the feature to enable ComSignals with ComBitSize larger than 254 bits.</p> <p>Deactivate the feature if only ComSignals with ComBitSize smaller than 255 bits are used.</p>
Support for ComIPdus larger 31bytes	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComLargeComIPdus'</p>

Attribute Name	Value Type	Default Value	Description
			<p>Activate the feature to enable support for ComIPdus with ComIPduSize greater than 31 bytes.</p> <p>Deactivate the feature if only ComIPdus with ComIPduSize less than 31 bytes are used.</p>
Optimize Id Types	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComOptimizeldTypes'</p> <p>Activate the feature to enable ID type optimization (e.g. use uint8 as Com_SignalIdType if sufficient)</p> <p>Deactivate the feature to disable the ID type optimization.</p>
Com Transmission Triggering	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComTriggerTransmissionByCom'</p> <p>If Com is not required to trigger the transmission of IPdus you can disable this functionality to save resources. This could be for example a Lin or FlexRay only system where transmission is triggered by the lower layer via Com_TriggerTransmit().</p>
Transmission Mode Switch	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComTmsSupport'</p> <p>Activate the feature to enable the use of ComTxModeFalse.</p> <p>Deactivate the feature to disable the use of ComTxModeFalse and activate only the use of ComTxModeTrue. Due to the deactivation of ComTxModeFalse change of the ComFilterAlgorithm is limited to ALWAYS and MASKED_NEW_DIFFERS_MASKED_OLD (GenSigSendType Cyclic, OnWrite and OnChange are possible).</p>
Integration Test Code	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComTsiTestCode'</p> <p>Activate the feature to enable test code for the integration of Com.</p> <p>Deactivate the feature to disable the feature.</p>
Write Access for Receive Signals	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComRxSignalWriteAccess'</p> <p>Enable/disable the support for write access to receive signals, signal groups and group signals.</p> <p>This feature is required by some RTE configurations. The</p>

Attribute Name	Value Type	Default Value	Description
			feature is not specified by AUTOSAR.
Signal Access Macro Api	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComSignalAccessMacroApi'</p> <p>Activate the feature to enable the signal access macro API. Deactivate the feature to disable the signal access macro API.</p>
BAC 2.1 Compatibility	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComBAC21Compatibility'</p> <p>Activate the feature to enable the BAC 2.1 compatibility mode. Deactivate the feature to disable the BAC 2.1 compatibility mode.</p>
Queue Failed Transmission Requests	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComQueueFailedTransmitRequests'</p> <p>If the feature is activated, the COM repeats transmission requests of a Tx I-Pdu to the underlying layer if E_NOT_OK is returned. If the feature is deactivated, the COM will ignore all errors from the underlying communication layer as recommended by AUTOSAR.</p>
Tx Timeout For Tx Mode None	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComTxTimeoutForTxModeNoneSupport'</p> <p>If this feature is activated the timeout context of a Tx I-Pdu is configurable by the attribute 'ComTxIPduTimeoutContext' to support transmission deadline monitoring for the transmission mode NONE. This could be used for Tx I-Pdus which are triggered by a bus schedule table (e.g. LIN).</p>
Multiple I-Pdu Group Reference Support	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComMultipleIPduGroupRefSupport'</p> <p>If this feature is activated one I-Pdu could be contained in more than one I-Pdu group.</p>
Use Pdu Info Type	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComUsePduInfoType'</p> <p>If this feature is activated, PduInfoPtr instead of SduPtr is used for the APIs Com_RxIndication and Com_TriggerTransmit.</p>
Dynamic DLC	Boolean	false	MICROSAR PARAMETER DEFINITION:

Attribute Name	Value Type	Default Value	Description
Support			<p>'ComDynamicDlcSupport'</p> <p>If this feature is activated, the received data length (PduInfoPtr-&gt;SduLength) is checked by the function Com_RxIndication, and only completely received signals and signal groups are processed.</p>
Signal Interface	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableSignalInterface'</p> <p>Activate the feature to enable the signal API.</p> <p>Deactivate the feature to disable the feature (e.g. if all signals are grouped Com_ReceiveSignal and Com_TransmitSignal are not used).</p>
Signal Group Interface	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableSignalGroupInterface'</p> <p>Activate the feature to enable the signal group API.</p> <p>Deactivate the feature to disable the feature.</p>
Update Bit Support	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableUpdateBitSupport'</p> <p>Activate the feature to enable the update bit handling. If a database is loaded and a ComSignal is found with the postfix or infix '_UB' and another signal in the same ComPdu is found with the same &lt;SignalName&gt;, the signal API is deactivated for the ComSignal and the signal is configured as update bit.</p> <p>Deactivate the feature to disable the update bit handling.</p> <p>Note: Update bits can be configured for ComSignals and ComSignalGroups.</p>
Use VStdLib Memory API	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComVStdLibMemApi'</p> <p>Activate the feature to use the VStdLib to set, clear and copy data elements in Com.</p> <p>Deactivate the feature to disable the feature.</p> <p>Note: The feature can be disabled for the Mcs12x if the post build data is located e.g. in the GPAGE.</p>
Signal Reception Filtering	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComEnableReceptionFiltering'</p> <p>Activate the feature to enable the signal reception filtering.</p> <p>Deactivate the feature to disable the signal reception</p>

Attribute Name	Value Type	Default Value	Description
			filtering.
Library	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComIsLibrary'</p> <p>Activate the feature to indicate, that Com is delivered as a library.</p> <p>Deactivate the feature to indicate, that Com is delivered as source code.</p>
Gw03 Consistency Check	Boolean	true	<p>MICROSAR PARAMETER DEFINITION: 'ComGw03ConsistencyCheck'</p> <p>Deactivate the feature to skip the check Gw03.</p>
Missing Source Value	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComMissingSourceValueHandling'</p> <p>This attribute is set automatically -BeforeGeneration- depending on attribute existence in the database.</p> <p>If enabled, the database attribute GenSigMissingSource value is used as Tx signal value in case of a Rx signal timeout.</p> <p>This value is set automatically at generation time if at least one database provides this attribute.</p>
Stop Tx PDU Transmission on Rx Timeout	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComGwStopTxOnRxTimeout'</p> <p>Feature stops transmission of Tx PDUs if the Tx PDU has a single Rx PDU only that timed out.</p>
CAN MOST Gateway	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComGwCanMost'</p> <p>Activate this feature to enable the CAN MOST Gateway extension.</p>
Invalidate Tx Signal on Rx Timeout	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComGwInvalidateTxOnRxTimeout'</p> <p>Feature invalidates the routed Tx Signals if a Rx timeout occurs.</p>
Trigger Tx Signal on Invalidation	Boolean	false	<p>MICROSAR PARAMETER DEFINITION: 'ComGwTriggerTxOnInvalidation'</p> <p>Feature triggers the routed Tx Signals in case of an invalidation caused by a Rx timeout.</p>
Suspend ISR Per Pdu	Boolean	false	MICROSAR PARAMETER DEFINITION:

Attribute Name	Value Type	Default Value	Description
			'ComGwSuspendIsrPerPdu'  If this feature is activated, only one ISR suspension is used per routed Rx I-Pdu. Otherwise every single signal access is protected by an own ISR suspension.

## 6.5.2 Signal gateway configuration

Routing relations are always configured on signal level. Thus, in order to route between signal groups, multiple routing relations between the signals of these signal groups have to be configured. The reason is that the location of the signals within a signal group can be change during routing.

### 6.5.2.1 Automatic routing relations

Routing relations between Rx and Tx signals can be detected automatically. Automatic routing relations cannot be deleted as they are an integral part of the database setup. It is possible to temporarily disable a routing relation by removing the “Generate” flag of the routing relation as illustrated by Figure 6-3 Signal routing parameters.

In order to configure an automatic routing relation between two signal groups, it is required that the signals of these signal groups have similar names.



#### Info

The activation of the automatic detection of routing relations depends on the tailoring of the delivered package and is OEM dependent.

### 6.5.2.2 Manual routing relations

To allow configuration of routing relations that cannot be detected by means of similar signal names, it is possible to add further routing relations manually.

Routing relations can be added by right-clicking on the “Signal Routing Relations” node in the GENy components view and selecting “Add new routing relation”.



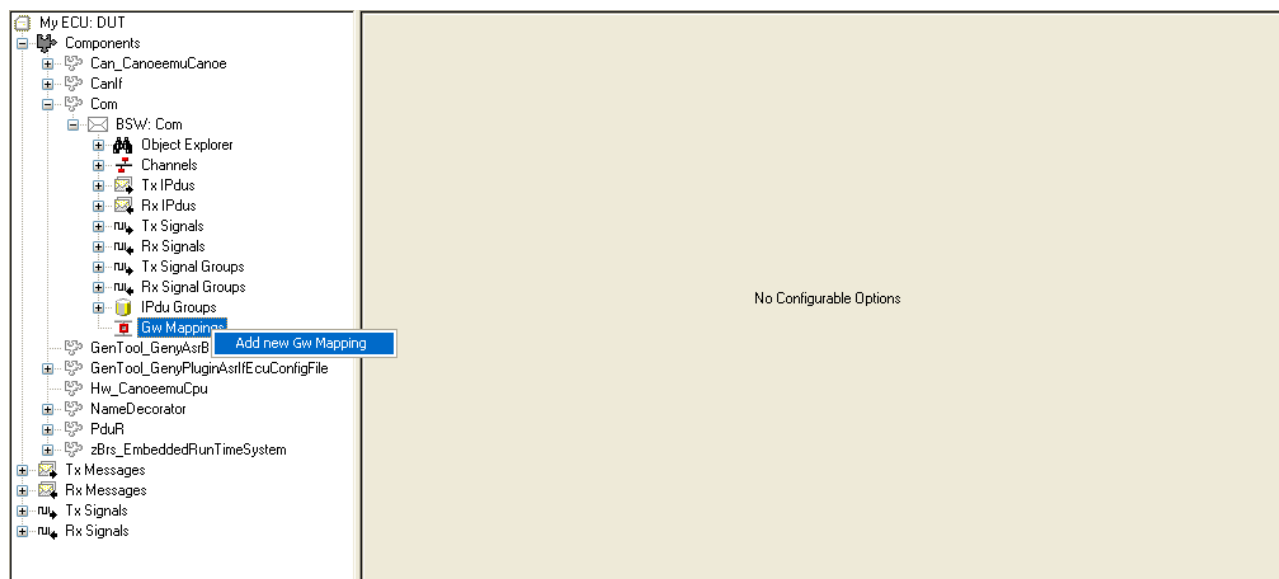


Figure 6-2 Creating a manual routing relation

A blank routing relation will be added to the “Manual Routing Relations” tree that can now be configured.

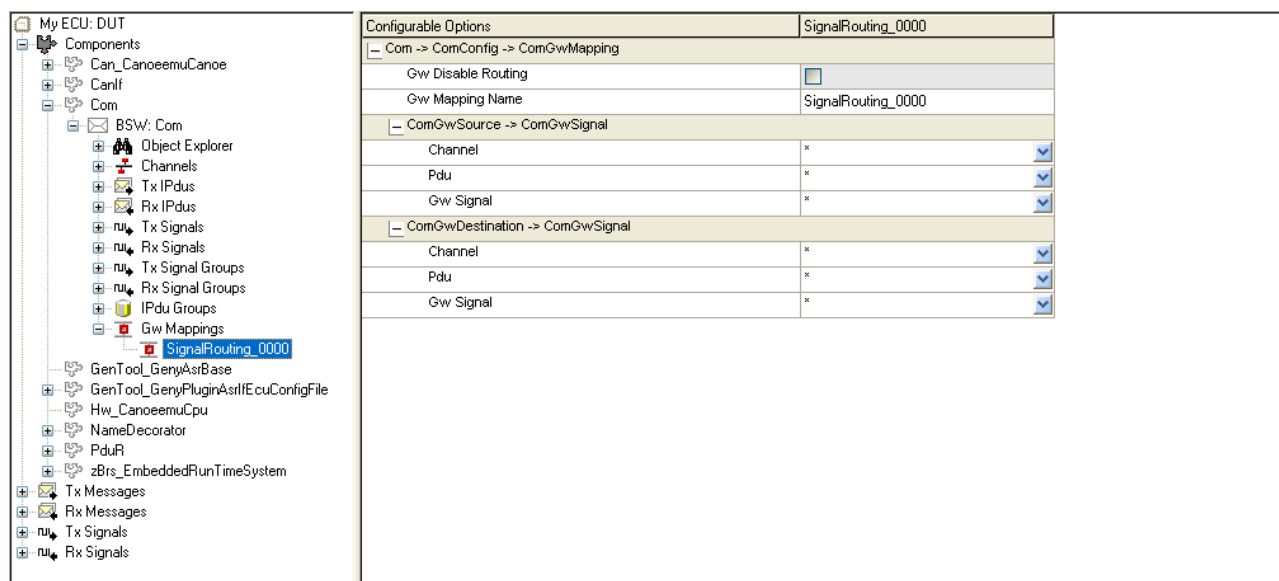


Figure 6-3 Signal routing parameters

To configure an 1:n routing – routing from one Rx signal to several Tx signals – several (n) 1:1 routing relations have to be configured.

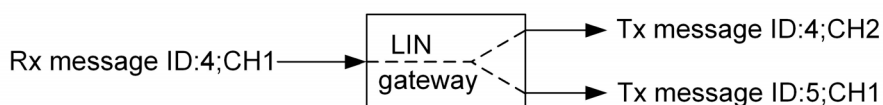


Figure 6-4 1:N routings are configured as several 1:1 routing relations

### 6.5.2.3 Deleting manual routing relations

Manual routing relations can be deleted by right-clicking on the appropriate routing relation and selecting “Remove routing relation”.

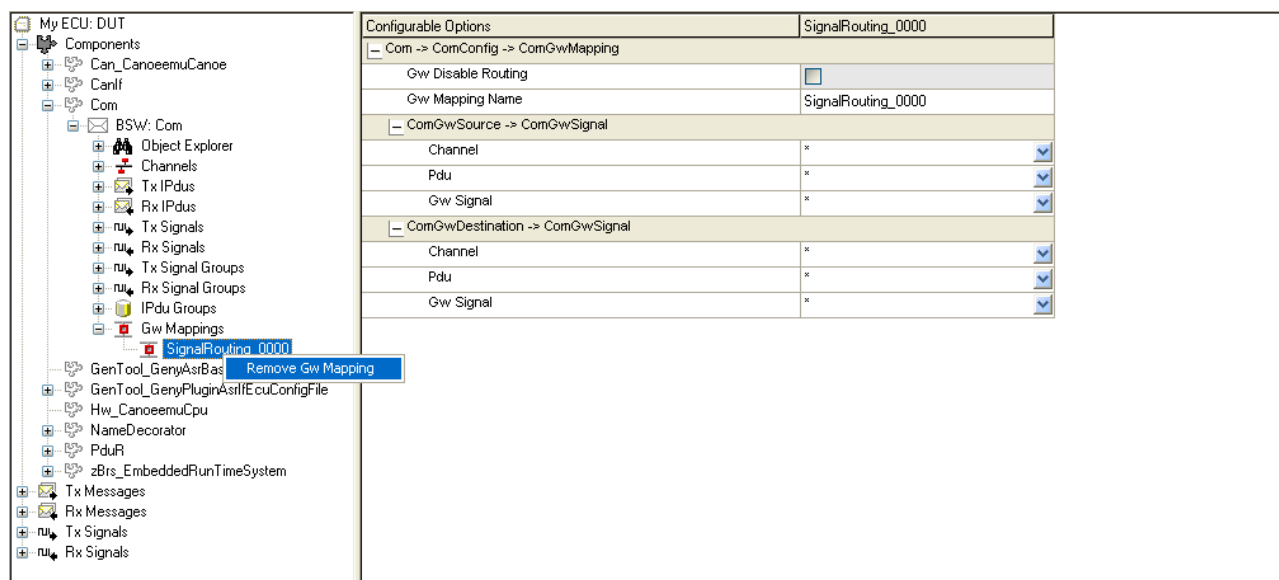


Figure 6-5 Deleting a manual routing relation

### 6.5.2.4 Gateway only signals

Signals and signal groups that are not used by the ECU application (i.e. RTE) can be marked as gateway only signals (“Internal”). Gateway only signals can be removed and added at post-build time.

The setting “Internal” is available in the GENy signal lists for all Tx and Rx signals and signal groups handled by COM. For further details on the usage of gateway only signals please see chapter 3.9.4 (Adding and removing gateway signals).

Com -> ComConfig -> ComSignal				
	Internal	Filter	Update Bit	
		Filter Algorithm	Enable	Update Bit Position
TxComPdu00ComSignal00	<input type="checkbox"/> *	NoFilter	<input type="checkbox"/> *	0*
TxComPdu00ComSignal01	<input checked="" type="checkbox"/>	NoFilter	<input type="checkbox"/> *	0*

Figure 6-6 Marking a signal as gateway only signal (“Internal”)



#### Caution

To allow removing a signal at post-build time, it is required that the signal has been marked as gateway only signal (“Internal”) at link time.



## 7 AUTOSAR Standard Compliance

### 7.1 Additions/ Extensions

This chapter describes add-ons and extensions implemented in Vector COM. These features are preconfigured in the Integration of the Package to tailor specific use cases.

For questions regarding to these Preconfigurations, ask your OEM or your delivery contact at Vector. These features do not directly contradict with an AUTOSAR requirement and are of no extra runtime or memory costs to the AUTOSAR COM Standard as long as these features are not activated. Some mechanisms have significant resource requirements only the ones which are actually needed should be enabled to build the library.

COM Variant	Details
Deadline Monitoring	Rx based on reception of I-PDUs Rx based on update bits
Error Handling	Development error detection (DET) Interface to the DEM
Layer Management	Services for status and version information
Signal Interface	Support for Rx or Tx only configuration Support for signal groups Support for update bits
Notification handling	Support for Rx indication functions
Signal Gateway Type of the I-PDU	Signal gateway support

Table 7-1 Main COM variants

#### 7.1.1 Signal Conversion

The signal conversion performs the data transformation from bus representation to ECU internal representation and vice versa via the `Com_ConvertSignal` API. The conversion is currently limited to bus signals with at most 16 bit size and the ECU data types: `uint8`, `sint8`, `uint16`, `sint16`, `uint32`, `sint32`. Only linear transformations are supported.

Values with special meaning (e.g. invalid value) can not be converted and have to be checked before the conversion.

### 7.1.1.1 Com\_ConvertSignal

Prototype	
<pre>void <b>Com_ConvertSignal</b> (Com_SignalIdType SignalId, const void *FromDataPtr, void *ToDataPtr)</pre>	
Parameter	
SignalId	ID of AUTOSAR COM signal to be converted.
FromDataPtr	Reference to the signal data to be converted.
ToDataPtr	Reference to the memory where the result is stored.
Return code	
void	none
Functional Description	
<p>The service Com_ConvertSignal() converts bus signal data to ecu internal data and vice versa. If the SignalId parameter refers to a receive signal the data is converted from BUS representation to ECU internal representation. If the SignalId refers to a transmit signal the data is converted from ECU internal representation to BUS representation.</p>	
Particularities and Limitations	
The function is called by the upper layer.	
Call Context	
The function can be called on interrupt and task level.	

Table 7-2 Com\_ConvertSignal

### 7.1.1.2 Com\_Convert\_SignedBusToEcu

Prototype	
<pre>sint32 <b>Com_Convert_SignedBusToEcu</b> (Com_Signal_SignalConvRefType ConversionPtr, sint32 BusData)</pre>	
Parameter	
ConversionPtr	Pointer to the conversion record (see Com_Signal_GetConversion).
BusData	The unconverted value.
Return code	
sint32	sint32 The converted value.
Functional Description	
<p>The service Com_Convert_SignedBusToEcu() is normally called by Com_ConvertSignal. This method can be called if the parameters are all known to the caller in order to save the additional function call.</p>	
Particularities and Limitations	
The function is called by Com_ConvertSignal or the upper layer.	
Call Context	
The function can be called on interrupt and task level.	

Table 7-3 Com\_Convert\_SignedBusToEcu

**Example**

This is an example of the reception of the signed signal s3 without invalid values. The global variable g\_s3 contains the ECU internal data.

```
void f_rx_s3(void)
{
    sint16 x;
    Com_ReceiveSignal(s3, &x);
    g_s3 =
    Com_Convert_SignedBusToEcu(Com_Signal_GetConversion(s3),
    x);
}
```

### 7.1.1.3 Com\_Convert\_UnsignedBusToEcu

#### Prototype

```
uint32 Com_Convert_UnsignedBusToEcu (Com_Signal_SignalConvRefType
ConversionPtr, uint32 BusData)
```

#### Parameter

ConversionPtr	Pointer to the conversion record (see Com_Signal_GetConversion).
BusData	The unconverted value.

#### Return code

uint32	uint32 The converted value.
--------	-----------------------------

#### Functional Description

The service Com\_Convert\_UnsignedBusToEcu() is normally called by Com\_ConvertSignal. This method can be called if the parameters are all known to the caller in order to save the additional function call.

#### Particularities and Limitations

The function is called by Com\_ConvertSignal or the upper layer.

#### Call Context

The function can be called on interrupt and task level.

Table 7-4 Com\_Convert\_UnsignedBusToEcu

**Example**

This is an example of the reception of the unsigned signal s4 without invalid values. The global variable g\_s4 contains the ECU internal data.

```
void f_rx_s4(void)
{
    uint8 x;
    Com_ReceiveSignal(s4, &x);
    g_s4 =
    Com_Convert_UnsignedBusToEcu(Com_Signal_GetConversion(s4), x);
}
```

#### 7.1.1.4 Com\_Convert\_SignedEcuToBus

Prototype	
<code>sint32 Com_Convert_SignedEcuToBus (Com_Signal_SignalConvRefType ConversionPtr, sint32 EcuData)</code>	
Parameter	
ConversionPtr	Pointer to the conversion record (see Com_Signal_GetConversion).
EcuData	The unconverted value.
Return code	
sint32	sint32 The converted value.
Functional Description	
<p>The service Com_Convert_SignedEcuToBus() is normally called by Com_ConvertSignal. This method can be called if the parameters are all known to the caller in order to save the additional function call.</p>	
Particularities and Limitations	
<p>The function is called by Com_ConvertSignal or the upper layer.</p>	
Call Context	
<p>The function can be called on interrupt and task level.</p>	

Table 7-5 Com\_Convert\_SignedEcuToBus



#### Example

This is an example of the transmission of the signed signal s5 without invalid values. The global variable g\_s5 contains the ECU internal data.

```
void f_tx_s5(void)
{
    sint8 x;
    x =
    Com_Convert_SignedEcuToBus(Com_Signal_GetConversion(s5),
    g_s5);
    Com_SendSignal(s5, &x);
}
```

#### 7.1.1.5 Com\_Convert\_UnsignedEcuToBus

Prototype	
<code>uint32 Com_Convert_UnsignedEcuToBus (Com_Signal_SignalConvRefType ConversionPtr, uint32 EcuData)</code>	
Parameter	
ConversionPtr	Pointer to the conversion record (see Com_Signal_GetConversion).

EcuData	The unconverted value.
<b>Return code</b>	
uint32	uint32 The converted value.
<b>Functional Description</b>	
The service Com_Convert_UnsignedEcuToBus() is normally called by Com_ConvertSignal. This method can be called if the parameters are all known to the caller in order to save the additional function call.	
<b>Particularities and Limitations</b>	
The function is called by Com_ConvertSignal or the upper layer.	
<b>Call Context</b>	
The function can be called on interrupt and task level.	

Table 7-6 Com\_Convert\_UnsignedEcuToBus

**Example**

This is an example of the transmission of the unsigned signal s6 without invalid values. The global variable g\_s6 contains the ECU internal data.

```
void f_tx_s6(void)
{
    uint32 x;
    x =
    Com_Convert_UnsignedEcuToBus(Com_Signal_GetConversion(s6), g_s6);
    Com_SendSignal(s6, &x);
}
```

**Example**

This is an example of the transmission of the signal named s1. The global variable g\_s1 contains the ECU internal data and g\_s1\_valid flag represents the validity of the signal.

```
void f_tx_s1(void)
{
    uint16 x;
    if(!g_s1_valid)
    {
        x = 0xffff;
    }
    else
    {
        Com_ConvertSignal(s1, &g_s1, &x);
    }
    Com_SendSignal(s1, &x);
}
```





### Example

This is an example of the reception of the signal s2. The global variable g\_s2 contains the ECU internal data and g\_s2\_valid flag represents the validity of the signal.

```
void f_rx_s2(void)
{
    uint16 x;
    Com_ReceiveSignal(s2, &x);
    if(x == 0xffff)
    {
        g_s2_valid = false;
    }
    else
    {
        g_s2_valid = true;
        Com_ConvertSignal(s2, &x, &g_s2);
    }
}
```

### 7.1.1.6 Configuration with GENy

For each signal the conversion from bus specific representation to ECU internal representation can be configured as shown in the following screenshot.

My ECU: DUT

Components

Can\_CanoeemuCanoe

CanIf

Com

BSW: Com

Object Explorer

Channels

Tx IPdus

Rx IPdus

Tx Signals

Rx Signals

Tx Signal Groups

Rx Signal Groups

IPdu Groups

Gw Mappings

GenTool\_GenYAsrBase

GenTool\_GenYPluginAsrIfEcu

Hw\_CanoeemuCpu

NameDecorator

PduR

zBrs\_EmbeddedRunTimeSys

Tx Messages

Rx Messages

Tx Signals

Rx Signals

Signal Conversion

Bus to Phys

Phys to Ecu

Bus to Ecu

	Enable	Source	Factor	Offset	ASAP2-DB	Factor	Offset	Factor	Offset	Quality
IIRxMsg04Sig	<input checked="" type="checkbox"/>	Database	1.0000*	0.0000*	*	1.0000*	0.0000*	1.0000	0.0000*	1.0000
IIRxMsg03Sig	<input checked="" type="checkbox"/>	Manual	1.0000*	0.0000*	*	1.0000*	0.0000*	1.0000	0.0000*	1.0000
IIRxMsg02Sig	<input checked="" type="checkbox"/>	Database	1.0000*	0.0000*	*	1.0000*	0.0000*	1.0000	0.0000*	1.0000
IIRxMsg01Sig	<input type="checkbox"/>									
IIRxMsg00Sig	<input type="checkbox"/>	*								

Figure 7-1 Signal conversion parameters

An ASAP2 (\*.a2l) database can be imported by selecting the file using the button shown in the following screenshot.

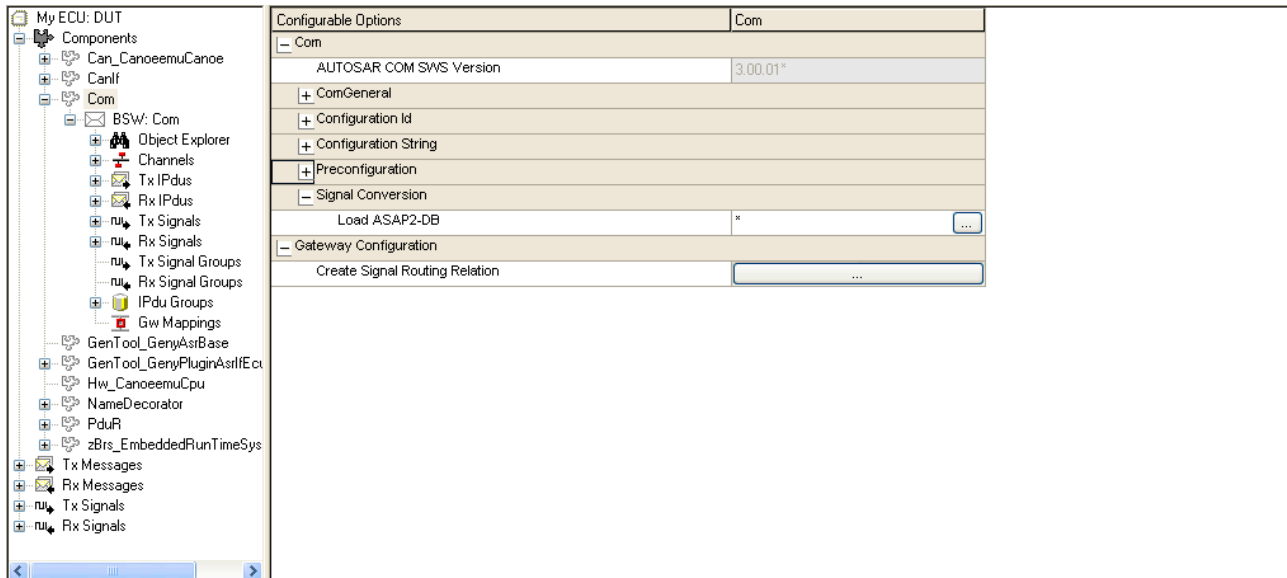


Figure 7-2 Import of an ASAP2 data base

The ASAP2 database file is not required anymore after importing it into the GENy configuration. The database can be updated by repeating the import process. Removed measurement objects produce an error message if they are still selected for any signal. The whole database is imported even if there are error messages shown during import. All reported inconsistencies should be manually corrected after import.

### 7.1.2 Rx Signal Notification Flags

AUTOSAR does currently not specify flags in case an Rx signal was received. The Vector COM offers a signal indication flag which is set for each signal and signal group of a received I-PDU. Optionally configured update bits are also evaluated.



#### Example

```
This is an example of the reception with indication flag of the signal s1.
if (TRUE == Com_GetRxSigIndicationFlag(s1)) /* check for
reception */
{
    uint16 x;
    Com_ReceiveSignal(s1, &x);                /* get the
signal value */
    Com_ClrRxSigIndicationFlag(s1);            /* clear the
flag */
    ...                                        /* further
processing */
}
```

The context, in which the notification flags are set, depends on the attribute 'IPdu Signal Processing' of the corresponding Rx I-Pdu.

If the attribute is configured to 'DEFERRED' the notification flag of a signal or signal group is set on task-level within the next call of the 'Com\_MainfunctionRx()' after the signal or signal was received.

If the attribute is configured to 'IMMEDIATE' the notification flag of a signal or signal group is set on interrupt-level within the call context of the function 'Com\_RxIndication()'.

The notification flags are initialized to 'FALSE' by a call to these function:

- > 'Com\_Init()'
- > 'Com\_Delnit()'
- > 'Com\_IpduGroupStart()' with parameter 'Initialize' set to 'TRUE'.

In the configuration variant 'Pre-Compile' an optimized flag access is provided by directly accessing the state of the flags within the RAM array.

### 7.1.2.1 Com\_GetRxSigIndicationFlag

Prototype	
boolean Com_GetRxSigIndicationFlag(RxSigId)	
Parameter	
RxSigId	Symbolic name or ID of AUTOSAR COM signal.
Return code	
boolean	TRUE Indication flag is set FALSE Indication flag is not set
Functional Description	
This function like macro returns the state of the indication flag of the signal.	
Particularities and Limitations	
Pre-Compile Variant: The macro could not be resolved by the pre-processor, if it is called for a signal that has not assigned an indication flag or for a none existing RxSigId. Link-Time/Post-Build: Due to the macro might be used in an expression no DET error is logged. If the macro is called with a parameter that is out of range, the result is undefined. The postfix of the macro-name is configurable via Il_AsrComNames::ComIndicationFlagPostfix	
Call Context	
The function can be called on interrupt and task level.	

Table 7-7 Com\_GetRxSigIndicationFlag

### 7.1.2.2 Com\_GetRxSigGrpIndicationFlag

Prototype
boolean Com_GetRxSigGrpIndicationFlag(RxSigGrpId)

Parameter	
RxSigGrpId	symbolic name or ID of AUTOSAR COM signal group.
Return code	
boolean	TRUE Indication flag is set
	FALSE Indication flag is not set
Functional Description	
This function like macro returns the state of the indication flag of the signal group.	
Particularities and Limitations	
<p>Pre-Compile Variant:</p> <p>The macro could not be resolved by the pre-processor, if it is called for a signal group that has not assigned a FirstValue flag or for a none existing RxSigGrpId.</p> <p>Link-Time/Post-Build:</p> <p>Due to the macro might be used in an expression no DET error is logged.</p> <p>If the macro is called with a parameter that is out of range, the result is undefined.</p> <p>The postfix of the macro-name is configurable via Il_AsrComNames::ComIndicationFlagPostfix</p>	
Call Context	
The function can be called on interrupt and task level.	

Table 7-8 Com\_GetRxSigGrpIndicationFlag

### 7.1.2.3 Com\_ClrRxSigIndicationFlag

Prototype	
boolean Com_ClrRxSigIndicationFlag(RxSigId)	
Parameter	
RxSigId	symbolic name or ID of AUTOSAR COM signal.
Return code	
void	none
Functional Description	
<p>This function like macro clears the indication flag of the signal.</p> <p>The clear-operation is interrupt protected if "Protect Indication Flag Access" is set in Geny.</p> <p>If "Protect Indication Flag Access" is not set, the caller of the macro has to care for data consistency.</p>	

### Particularities and Limitations

#### Pre-Compile Variant:

The macro could not be resolved by the pre-processor, if it is called for a signal that has not assigned a FirstValue flag or for a none existing RxSigId.

#### Link-Time/Post-Build:

If the macro is called with a parameter that is out of range, the result is undefined and a DET error will be logged.

The postfix of the macro-name is configurable via `Il_AsrComNames::ComIndicationFlagPostfix`

### Call Context

The function can be called on interrupt and task level.

Table 7-9 Com\_ClrRxSigIndicationFlag

## 7.1.2.4 Com\_ClrRxSigGrpIndicationFlag

### Prototype

```
boolean Com_ClrRxSigGrpIndicationFlag (RxSigGrpId)
```

### Parameter

RxSigGrpId	symbolic name or ID of AUTOSAR COM signal group.
------------	--

### Return code

void	none
------	------

### Functional Description

This function like macro clears the indication flag of the signal group.

The clear-operation is interrupt protected if "Protect Indication Flag Access" is set in Geny.

If "Protect Indication Flag Access" is not set, the caller of the macro has to care for data consistency.

### Particularities and Limitations

#### Pre-Compile Variant:

The macro could not be resolved by the pre-processor, if it is called for a signal group that has not assigned a FirstValue flag or for a none existing RxSigId.

#### Link-Time/Post-Build:

If the macro is called with a parameter that is out of range, the result is undefined and a DET error will be logged.

The postfix of the macro-name is configurable via `Il_AsrComNames::ComIndicationFlagPostfix`

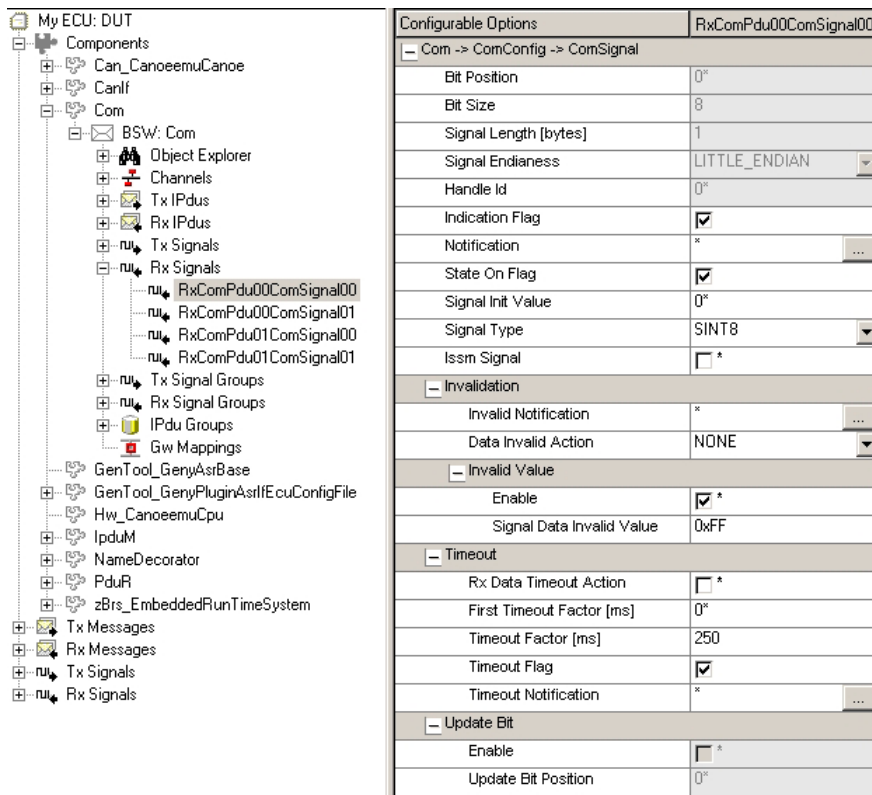
### Call Context

The function can be called on interrupt and task level.

Table 7-10 Com\_ClrRxSigGrpIndicationFlag

### 7.1.2.5 Configuration

For each signal and signal group an indication flag can be configured by checking the indication flag option shown in the following screenshot



Configurable Options		RxComPdu00ComSignal00
Com -> ComConfig -> ComSignal		
Bit Position		0*
Bit Size		8
Signal Length [bytes]		1
Signal Endianness		LITTLE_ENDIAN
Handle Id		0*
Indication Flag		<input checked="" type="checkbox"/>
Notification		x ...
State On Flag		<input checked="" type="checkbox"/>
Signal Init Value		0*
Signal Type		SINT8
Issm Signal		<input type="checkbox"/> *
Invalidation		
Invalid Notification		x ...
Data Invalid Action		NONE
Invalid Value		
Enable		<input checked="" type="checkbox"/> *
Signal Data Invalid Value		0xFF
Timeout		
Rx Data Timeout Action		<input type="checkbox"/> *
First Timeout Factor [ms]		0*
Timeout Factor [ms]		250
Timeout Flag		<input checked="" type="checkbox"/>
Timeout Notification		x ...
Update Bit		
Enable		<input checked="" type="checkbox"/> *
Update Bit Position		0*

Figure 7-3 Signal indication flag configuration

In the default configuration all flag access is protected against interrupts. Since the flags are cleared by the application, protecting each single clear-operation may cause an unacceptable overhead. Therefore the flag access protection can be configured as shown in the following screenshot. Please note that this parameter has no effect on the set-operation which is always protected.

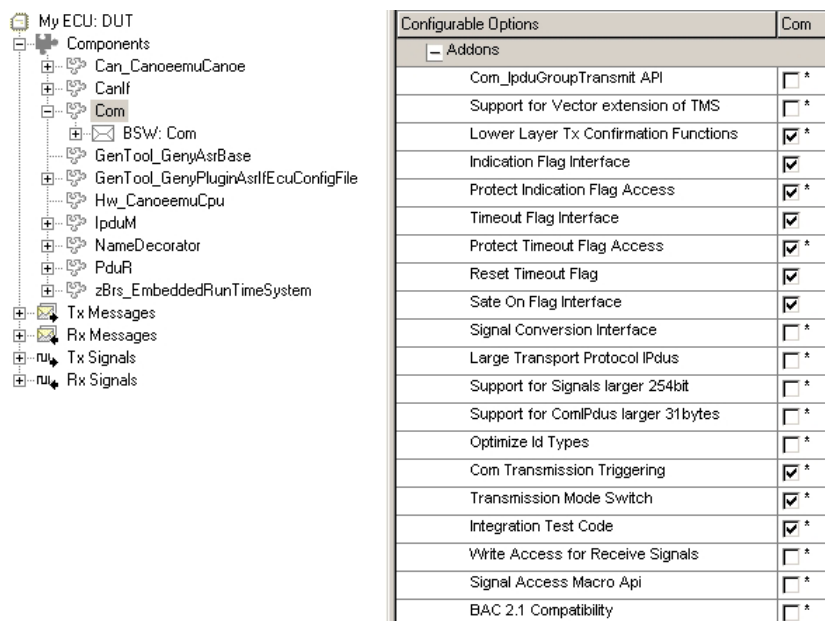


Figure 7-4 Protect Indication Flag Access parameter

The postfix of the generated get and clear macros could be configured by the parameter 'Flag Provider Postfixes -> Indication Flag' as shown in the following screenshot:

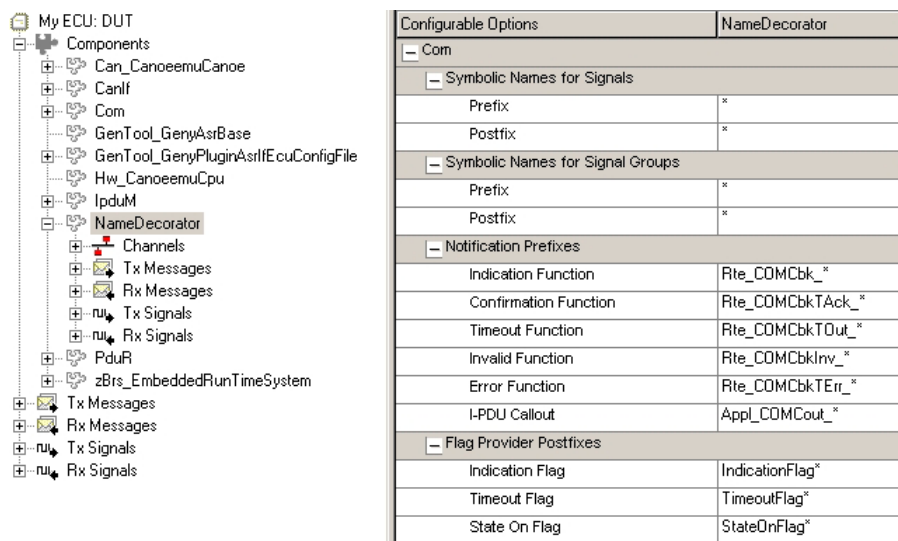


Figure 7-5 Indication flag macro postfix

The macro name is made up as follows:

'Com\_Get' / 'Com\_Clr' + 'RxSig' / 'RxSigGrp' + <ComIndicationFlagPostfix>

### 7.1.3 Rx Timeout Flags

AUTOSAR does currently not specify flags in case an Rx signal timeout was detected. The Vector COM offers a flag for each signal and signal group which is set when a timeout was detected. Optionally configured update bits are also evaluated.



#### Example

This is an example of the timeout notification via flag of the signal s1.

```
if (TRUE == Com_GetRxSigTimeoutFlag(s1))    /* check for
timeout */
{
    Com_ClrRxSigTimeoutFlag(s1);
    ...                                     /* handle
timeout */
}
else
{
    ...                                     /* normal
processing */
}
```

If the feature 'Reset Timeout Flag' is activated, the timeout flags are reseted by the next reception of the signal or signal group after an reception timeout occurred.

The context, in which the timeout flags are reseted, depends on the attribute 'IPdu Signal Processing' of the corresponding Rx I-Pdu.

If the attribute is configured to 'DEFERRED' the timeout flag of a signal or signal group is reseted on task-level within the next call of the 'Com\_MainfunctionRx()' after the signal or signal was received.

If the attribute is configured to 'IMMEDIATE' the timeout flag of a signal or signal group is reseted on interrupt-level within the call context of the function 'Com\_RxIndication()'.

The timeout flags are initialized to 'FALSE' by a call to these function:

- > 'Com\_Init()'
- > 'Com\_DeInit()'
- > 'Com\_IpduGroupStart()' with parameter 'Initialize' set to 'TRUE'.
- > 'Com\_EnableReceptionDM()'

In the configuration variant 'Pre-Compile' an optimized flag access is provided by directly accessing the state of the flags within the RAM array.



### 7.1.3.1 Com\_GetRxSigTimeoutFlag

Prototype	
boolean Com_GetRxSigTimeoutFlag(RxSigId)	
Parameter	
RxSigId	symbolic name or ID of AUTOSAR COM signal.
Return code	
boolean	TRUE    Timeout flag is set FALSE   Timeout flag is not set
Functional Description	
This function like macro returns the state of the timeout flag of the signal.	
Particularities and Limitations	
Pre-Compile Variant: The macro could not be resolved by the pre-processor, if it is called for a signal that has not assigned a timeout flag or for a none existing RxSigId. Link-Time/Post-Build: Due to the macro might be used in an expression no DET error is logged. If the macro is called with a parameter that is out of range, the result is undefined. > > The postfix of the macro-name is configurable via Il_AsrComNames::ComTimeoutFlagPostfix	
Call Context	
The function can be called on interrupt and task level.	

Table 7-11 Com\_GetRxSigTimeoutFlag

### 7.1.3.2 Com\_GetRxSigGrpTimeoutFlag

Prototype	
boolean Com_GetRxSigGrpTimeoutFlag(RxSigGrpId)	
Parameter	
RxSigGrpId	symbolic name or ID of AUTOSAR COM signal group.
Return code	
boolean	TRUE    Timeout flag is set FALSE   Timeout flag is not set
Functional Description	
This function like macro returns the state of the timeout flag of the signal group.	

### Particularities and Limitations

#### Pre-Compile Variant:

The macro could not be resolved by the pre-processor, if it is called for a signal group that has not assigned a timeout flag or for a none existing RxSigGrpId.

#### Link-Time/Post-Build:

Due to the macro might be used in an expression no DET error is logged.

If the macro is called with a parameter that is out of range, the result is undefined.

>

> The postfix of the macro-name is configurable via `Il_AsrComNames::ComTimeoutFlagPostfix`

### Call Context

The function can be called on interrupt and task level.

Table 7-12 Com\_GetRxSigGrpTimeoutFlag

### 7.1.3.3 Com\_ClrRxSigTimeoutFlag

#### Prototype

```
boolean Com_ClrRxSigTimeoutFlag(RxSigId)
```

#### Parameter

RxSigId	symbolic name or ID of AUTOSAR COM signal.
---------	--

#### Return code

void	none
------	------

#### Functional Description

This function like macro clears the timeout flag of the signal.

The clear-operation is interrupt protected if "Protect Timeout Flag Access" is set in Geny.

If "Protect Timeout Flag Access" is not set, the caller of the macro has to care for data consistency.

### Particularities and Limitations

#### Pre-Compile Variant:

The macro could not be resolved by the pre-processor, if it is called for a signal that has not assigned a timeout flag or for a none existing RxSigId.

#### Link-Time/Post-Build:

If the macro is called with a parameter that is out of range, the result is undefined and a DET error will be logged.

The postfix of the macro-name is configurable via `Il_AsrComNames::ComTimeoutFlagPostfix`

### Call Context

The function can be called on interrupt and task level.

Table 7-13 Com\_ClrRxSigTimeoutFlag

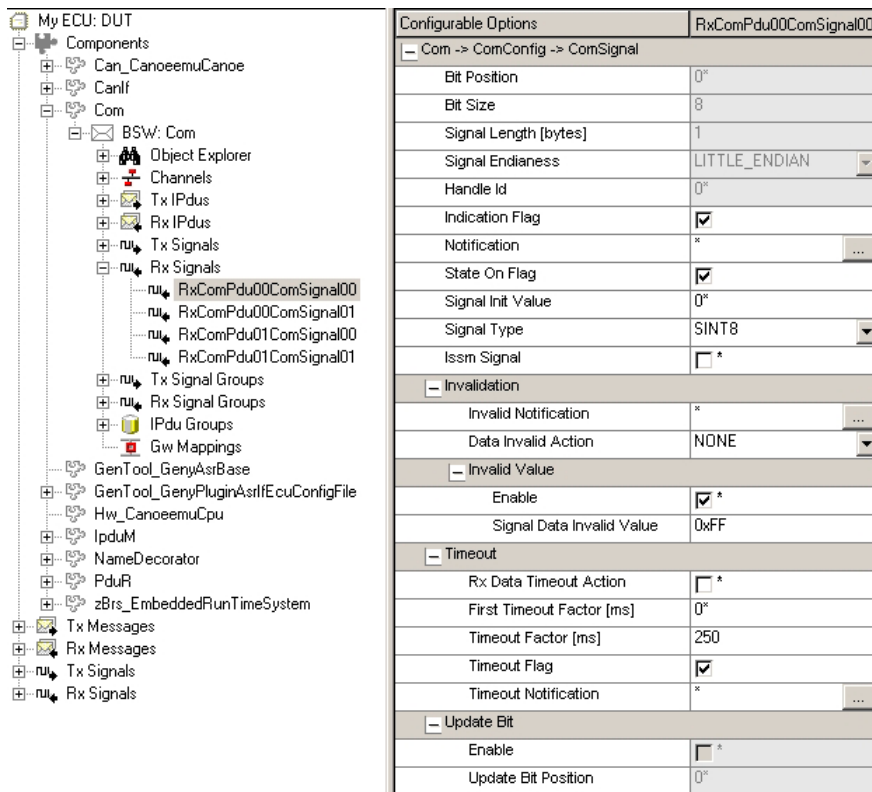
### 7.1.3.4 Com\_ClrRxSigGrpTimeoutFlag

Prototype	
boolean Com_ClrRxSigGrpTimeoutFlag (RxSigGrpId)	
Parameter	
RxSigGrpId	symbolic name or ID of AUTOSAR COM signal group.
Return code	
void	none
Functional Description	
<p>This function like macro clears the timeout flag of the signal group.</p> <p>The clear-operation is interrupt protected if "Protect Timeout Flag Access" is set in Geny.</p> <p>If "Protect Timeout Flag Access" is not set, the caller of the macro has to care for data consistency.</p>	
Particularities and Limitations	
<p>Pre-Compile Variant:</p> <p>The macro could not be resolved by the pre-processor, if it is called for a signal group that has not assigned a timeout flag or for a none existing RxSigGrpId.</p> <p>Link-Time/Post-Build:</p> <p>If the macro is called with a parameter that is out of range, the result is undefined and a DET error will be logged.</p> <p>The postfix of the macro-name is configurable via Il_AsrComNames::ComTimeoutFlagPostfix</p>	
Call Context	
The function can be called on interrupt and task level.	

Table 7-14 Com\_ClrRxSigGrpTimeoutFlag

### 7.1.3.5 Configuration

For each signal and signal group a timeout flag can be configured by checking the timeout flag option shown in the following screenshot.



Configurable Options	RxComPdu00ComSignal00
Com -> ComConfig -> ComSignal	
Bit Position	0*
Bit Size	8
Signal Length [bytes]	1
Signal Endianness	LITTLE_ENDIAN
Handle Id	0*
Indication Flag	<input checked="" type="checkbox"/>
Notification	x
State On Flag	<input checked="" type="checkbox"/>
Signal Init Value	0*
Signal Type	SINT8
Issm Signal	<input type="checkbox"/> *
Invalidation	
Invalid Notification	x
Data Invalid Action	NONE
Invalid Value	
Enable	<input checked="" type="checkbox"/> *
Signal Data Invalid Value	0xFF
Timeout	
Rx Data Timeout Action	<input type="checkbox"/> *
First Timeout Factor [ms]	0*
Timeout Factor [ms]	250
Timeout Flag	<input checked="" type="checkbox"/>
Timeout Notification	x
Update Bit	
Enable	<input type="checkbox"/> *
Update Bit Position	0*

Figure 7-6 Signal indication flag configuration

In the default configuration all flag access is protected against interrupts. Since the flags are cleared by the application, protecting each single clear-operation may cause an unacceptable overhead. Therefore the flag access protection can be configured as shown in the following screenshot. Please note that this parameter has no effect on the set-operation which is always protected.

Additionally the feature 'Reset Timeout Flag' could be activated as shown in the following screenshot.

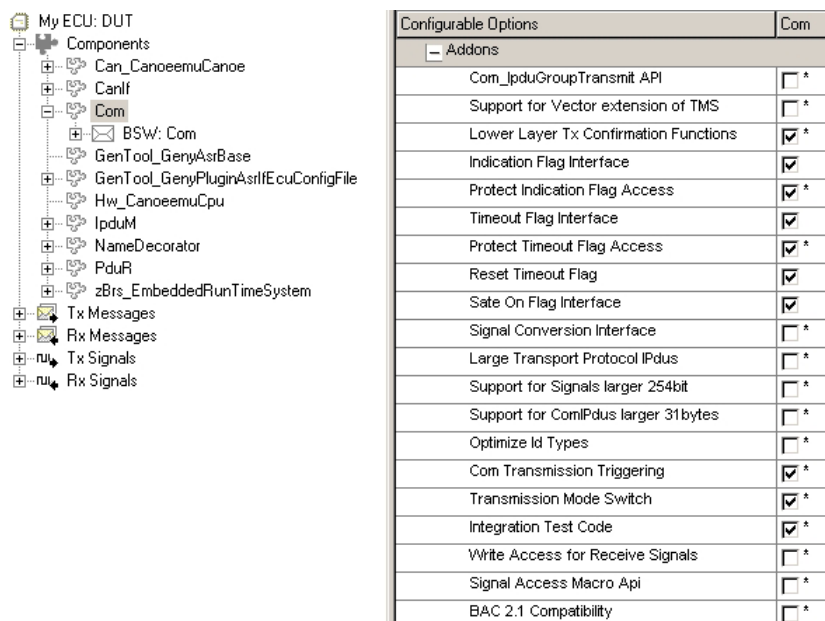


Figure 7-7 Timeout Flag Access parameter

The postfix of the generated get and clear macros could be configured by the parameter 'Flag Provider Postfixes -> Timeout Flag' as shown in the following screenshot:

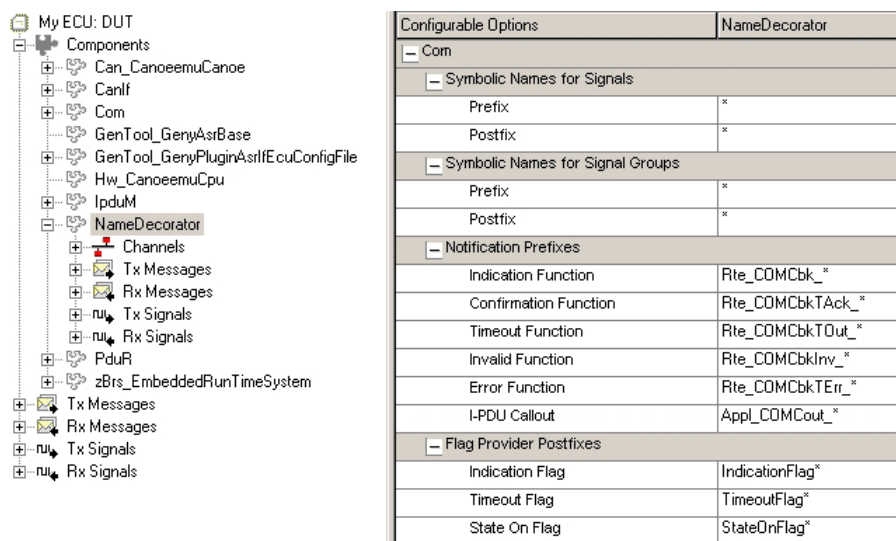


Figure 7-8 Timeout flag macro postfix

The macro name is made up as follows:

'Com\_Get' / 'Com\_Clr' + 'RxSig' / 'RxSigGrp' + <ComTimeoutFlagPostfix>

### 7.1.4 Rx State On Flags

AUTOSAR does currently not specify flags which indicates whether the corresponding I-Pdu group of a Rx signal or Rx signal group is active. The Vector COM offers a flag for each signal and signal group which indicates the current state.



#### Example

This is an example of the usage of a state on flag of the signal s1.

```
if (TRUE == Com_GetRxSigStateOnFlag(s1))    /* check the
state*/
{
    ...                                     /* processing
if ative */
}
```

#### 7.1.4.1 Com\_GetRxSigStateOnFlag

Prototype	
boolean Com_GetRxSigStateOnFlag(RxSigId)	
Parameter	
RxSigId	symbolic name or ID of AUTOSAR COM signal.
Return code	
boolean	TRUE    At least one of the mapped Rx IPduGroups is active FALSE   None of the mapped Rx IPduGroups is active
Functional Description	
This function like macro returns the state of the signal.	
Particularities and Limitations	
> The macro could not be resolved by the pre-processor, if it is called for a signal that has not assigned a StateOn flag or for a none existing RxSigId. > The postfix of the macro-name is configurable via Il_AsrComNames::ComStateOnFlagPostfix	
Call Context	
The function can be called on interrupt and task level.	

Table 7-15 Com\_GetRxSigStateOnFlag

#### 7.1.4.2 Com\_GetRxSigGrpStateOnFlag

Prototype	
boolean Com_GetRxSigGrpStateOnFlag(RxSigGrpId)	
Parameter	
RxSigGrpId	symbolic name or ID of AUTOSAR COM signal group.
Return code	
boolean	TRUE    At least one of the mapped Rx IPduGroups is active FALSE   None of the mapped Rx IPduGroups is active

Functional Description
This function like macro returns the state of the signal group.
Particularities and Limitations
<ul style="list-style-type: none"> <li>&gt; The macro could not be resolved by the pre-processor, if it is called for a signal group that has not assigned a StateOn flag or for a none existing RxSigGrpId.</li> <li>&gt; The postfix of the macro-name is configurable via <code>II_AsrComNames::ComStateOnFlagPostfix</code></li> </ul>
Call Context
The function can be called on interrupt and task level.

Table 7-16 Com\_GetRxSigGrpStateOnFlag

### 7.1.4.3 Configuration

For each signal and signal group a state on flag can be configured by checking the state on flag option shown in the following screenshot.

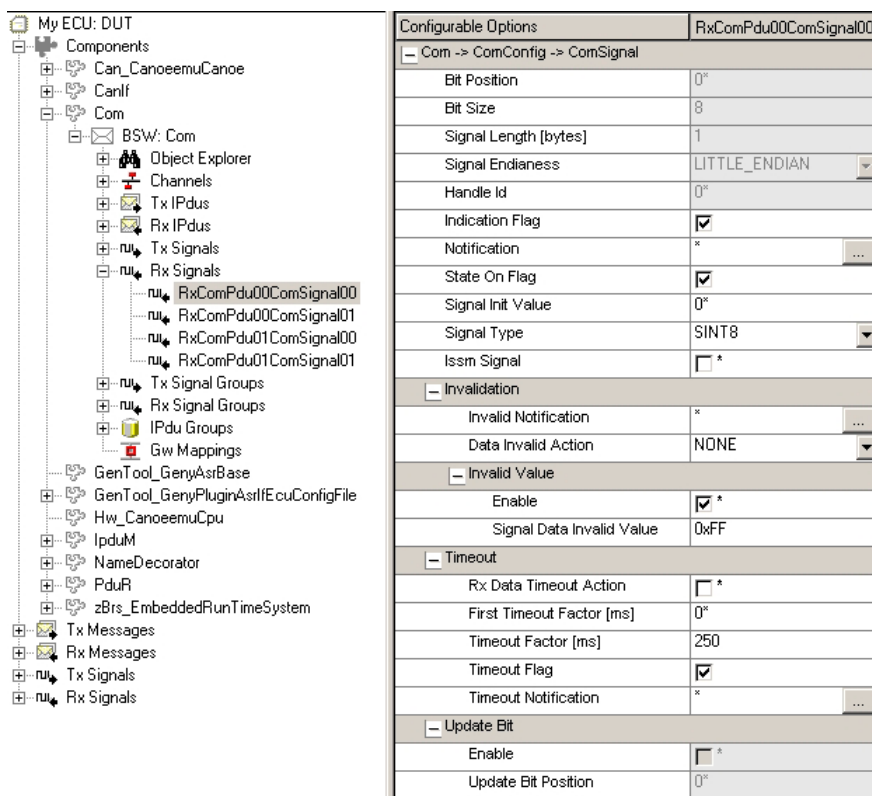


Figure 7-9 Signal state on flag configuration

The postfix of the generated get macros could be configured by the parameter 'Flag Provider Postfixes -> State On Flag' as shown in the following screenshot:

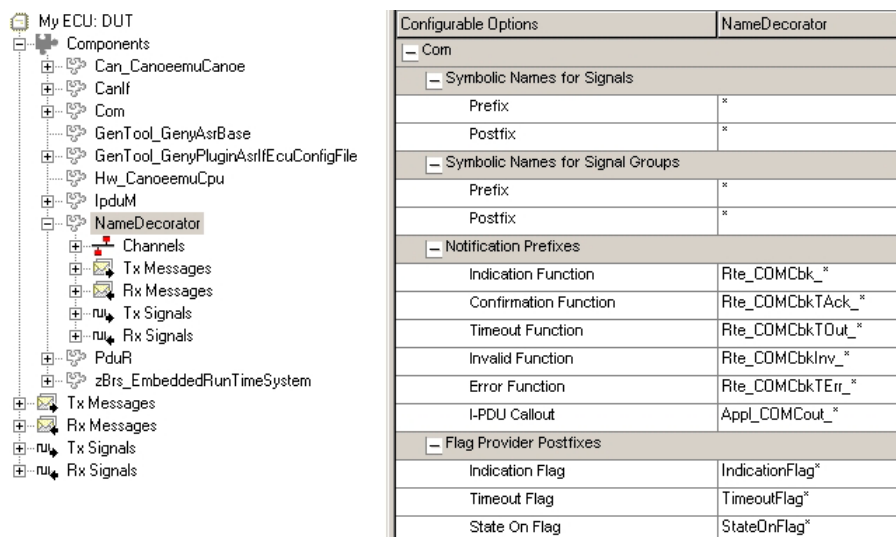


Figure 7-10 State On flag macro postfix

The macro name is made up as follows:

'Com\_Get' / 'Com\_Clr' + 'RxSig' / 'RxSigGrp' + <ComStateOnFlagPostfix>

### 7.1.5 Com\_IpduGroupTransmit

Prototype	
void <b>Com_IpduGroupTransmit</b> (Com_PduGroupIdType IpduGroupId)	
Parameter	
IpduGroupId	ID of I-PDU group to be stopped
Return code	
void	none
Functional Description	
Sets a transmission request for all I-PDUs of a preconfigured I-PDU group.	
Particularities and Limitations	
The function is used by Ccl with Il_Ni extension.	
Call Context	
The function must be called on task level.	

Table 7-17 Com\_IpduGroupTransmit

### 7.1.6 Transport Protocol API (CanTp)



**Caution**

For COM I-PDUs received by a TP the attribute 'ComIPduSignalProcessing' must be configured to 'DEFERRED' to avoid Rx indication calls with disabled interrupts.

In this case the COM should be scheduled directly after the TP (in SchM) to minimize the delay induced by the deferred handling.

### 7.1.6.1 Com\_TpProvideRxBuffer

Prototype	
BufReq_ReturnType <b>Com_TpProvideRxBuffer</b> (PduIdType ComRxPduId, PduLengthType TpSduLength, PduInfoTypeApplPtr *PduInfoPtr)	
Parameter	
ComRxPduId	ID of AUTOSAR COM I-PDU that will be received.
TpSduLength	Length of the I-PDU that will be received.
PduInfoPtr**	Reference where COM will locate a pointer to the data buffer that will be used as Rx buffer.
Return code	
BufReq_ReturnType	BUFREQ_OK if COM accepts the reception. If COM cannot receive the I-PDU BUFREQ_E_NOT_OK is returned.
Functional Description	
This function called by the lower layer to query for a receive buffer that is required to receive a TP layer I-PDU. COM will always provide a buffer that allows the TP layer to store the complete TP I-PDU. The TpSduLength must match with the COM I-PDU length to allow COM to receive the I-PDU.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 7-18 Com\_TpProvideRxBuffer

### 7.1.6.2 Com\_TpRxIndication

Prototype	
Void <b>Com_TpRxIndication</b> (PduIdType ComRxPduId, NotifResultType Result)	
Parameter	
ComRxPduId	ID of AUTOSAR COM I-PDU that has been received.
Result	NTFRSLT_OK indicates that the reception was successful. Any other value is handled as an error by COM.

Return code	
void	none
Functional Description	
This function is called by the lower layer after a COM TP layer I-PDU has been received completely (Result has to indicate NTFRSLT_OK) or if the TP layer has canceled the reception due to any error. In this case, Result has to indicate the type of error.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 7-19 Com\_TpRxIndication

### 7.1.6.3 Com\_TpProvideTxBuffer

Prototype	
BufReq_ReturnType <b>Com_TpProvideTxBuffer</b> (PduIdType ComTxPduId, PduInfoTypeApplPtr *PduInfoPtr, uint16 Length)	
Parameter	
ComTxPduId	ID of AUTOSAR COM I-PDU that will be transmitted.
PduInfoPtr**	Reference where COM will locate a pointer to the data that will be transmitted
Length	Number of bytes required by the lower layer. 0 is handed as wild card.
Return code	
BufReq_ReturnType	BUFREQ_OK if COM has provided the requested buffer. BUFREQ_E_NOT_OK indicates an error
Functional Description	
This function called by the lower layer to query for a transmit buffer containing the I-PDU data that is to be transmitted. COM will always provide a buffer that allows the TP layer to transmit the complete TP I-PDU without querying for a further buffer. The requested buffer size (Length) must either be zero (any size) or must indicate the exact length of the COM I-PDU that is to be transmitted.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 7-20 Com\_TpProvideTxBuffer

### 7.1.6.4 Com\_TpTxConfirmation

Prototype	
Void <b>Com_TpTxConfirmation</b> (PduIdType ComTxPduId, NotifResultType Result)	

Parameter	
ComTxPduld	ID of AUTOSAR COM I-PDU that has been transmitted.
Result	Indicates if the transmission has been successfull (NTFRSLT_OK) or not
Return code	
void	none
Functional Description	
This function is called by the lower layer after a COM TP layer I-PDU has been transmitted completely (Result has to indicate NTFRSLT_OK) or if the TP layer has canceled the transmission due to any error. In this case, Result has to indicate the type of error.	
Particularities and Limitations	
The function is called by the lower layer.	
Call Context	
The function can be called on interrupt and task level. It is not allowed to use CAT1 interrupts with Rte (BSW00326]). Due to this, the interrupt context shall be configured to a CAT2 interrupt if an Rte is used.	

Table 7-21 Com\_TpTxConfirmation

### 7.1.7 Gateway: Signal routing

- > If the application data types of Rx and Tx signals are equal but not uint8[n], COM allows routing of signals with a smaller bit count to signals with a larger bit count (e.g. routing of a 3bit signal on the Rx bus to an 8bit signal on the Tx bus).
- > It is possible to route an Rx signal group to a Tx signal group that encapsulates only a subset of the Rx signal groups signals. In this case, data consistency is not violated in as all signals of the Tx signal group are still written consistently from a single Rx signal group.

### 7.1.8 Gateway: Rx signal timeout handling without update bits

AUTOSAR does not specify a timeout handling of Tx signals in case a Rx signal timeout was detected. The Vector COM provides the possibility to specify a timeout value for CAN Rx signals that is written to the Tx signals in case of an Rx signal timeout event.

The timeout value is configured by the DBC attribute GenSigMissingSourceValue for the Rx signal of a routing relation.



#### Caution

When writing a missing source value to a Tx signal as a responds to an Rx signal timeout the following restrictions apply:

- > If the Tx signal uses a filter other that F\_ALWAYS and the Tx I-PDU allows sporadic transmission, the I-PDU transmission is triggered at any time a Rx timeout occurs – neglecting the signal filter.
- > If the Tx signal has also an update bit no missing source value will be written to the Tx signal as the receiver can detect a timeout event by means of observing the update bit.

### 7.1.9 TMS Switch Support

The initial implementation of Vector COM did not support different Tx Modes and signal filters. This former transmission behaviour can reproduced.

### 7.1.10 COM without Confirmations

Vector COM can be configured to work with or without confirmation of the requested I-PDU transmission by the underlying layer. This has some impacts on the transmit behavior and the notification concept. If confirmations are not used the COM layer cannot guarantee the minimum send distance on the bus since transmission might be delayed by lower layers. The Notification of successful transmission mechanism is based on the confirmation by the underlying layer. The feature is not available if the COM layer is used without confirmation.

### 7.1.11 Large Signals

Vector COM can be configured to support signals with ComBitSize larger 63 bits. This feature may be required for FlexRay or the Transport Protocol Interface of COM implementation.

## 7.2 Limitations



#### Info

Some specific variants and features of the Vector IL have no compatible replacement in AUTOSAR COM. This includes the support for

- multiplexed signals
- first value notification
- dynamic timeout handling
- Tx notification flags

### 7.2.1 Component

#### Component Limitations

TRUE must be defined to (boolean) 1

FALSE must be defined to (boolean) 0

NULL\_PTR must be defined to ((void \*)0)

The Signal Gateway functionality is not supported for the target Tms320. (See ESCAN00028157)

## 7.2.2 Code Generator and Configuration

### Code Generator Limitations

The following Il\_Vector DBC attribute definitions are not handled by COM: ILUsed, GenMsgCycleTimeFast, GenMsgNrOfRepetition, GenSigInactiveValue, GenSigTimeoutValue, GenMsgFastOnStart, ILTxTimeout, GenMsgNolalSupport.

The filters NewIsOutside or NewIsWithin are not supported for signals with ComSignalType SINT8 and the target Tms320. (See ESCAN00027982).

The ComSignalInitValue for large signals cannot be configured in the EcuC file and can be configured in the GUI. The code generator uses always ComSignalInitValue where every bit is set.

## 8 Glossary and Abbreviations

### 8.1 Glossary

Term	Description
Buffer	A buffer in a memory area normally in the RAM. It is an area, that the application has reserved for data storage.
Bus off	A node is in the state bus off when it is switched off from the bus due to a request of fault confinement entity. In the bus off state, a node can neither send nor receive any frames. A node can start the recovery from bus off state only upon a user request.
Callback function	This is a function provided by an application. E.g. the CAN Driver calls a callback function to allow the application to control some action, to make decisions at runtime and to influence the work of the driver.
CAN Driver	The CAN Driver encapsulates a specific CAN controller handling. It consists of algorithms for hardware initialization, CAN message transmission and reception. The application interface supports both event and polling notification and WR/RD access to the message buffers.
Channel	A channel defines the assignment (1:1) between a physical communication interface and a physical layer on which different modules are connected to (either CAN or LIN). 1 channel consists of 1..X network(s).
Communication layer	The set of all entities and elements which constitute a communication layer based on the ISO/OSI Reference Model (ISO 7498).
Component	CAN Driver, Network Management ... are software COMPONENTS in contrast to the expression module, which describes an ECU.
Confirmation	A service primitive defined in the ISO/OSI Reference Model (ISO 7498). With the service primitive 'confirmation' a service provider informs a service user about the result of a preceding service request of the service user. Notification by the CAN Driver on asynchronous successful transmission of a CAN message.
Conformance class	In each OSEK module (operating system, communication, network management) a pool of different services and processing mechanisms is provided. Various requirements of the application software for a system, and various capabilities of a specific system demand different stages of the OSEK modules. These stages are upwardly compatible to one another and described as conformance classes.
Data consistency	Data consistency means that the content of a given application message correlates unambiguously to the operation performed onto the message by the application. This means that no unforeseen sequence of operations may alter the content of a message hence rendering a message inconsistent with respect to its allowed and expected value.
EAD	Embedded Architecture Designer; generation tool for MICROSAR components
Electronic control unit	Also known as ECU. Small embedded computer system consisting of at least one CPU and corresponding periphery which is placed in one

	housing.
Event	An exclusive signal which is assigned to a certain extended task. An event can be used to send a binary information to an extended task. The meaning of events is defined by the application. Any task can set an event for an extended task. The event can only be cleared by the task which is assigned to the event.
Gateway	A gateway is designed to enable communication between different bus systems, e.g. from CAN to LIN.
GENy	Generation tool for CANbedded and MICROSAR components
Indication	A service primitive defined in the ISO/OSI Reference Model (ISO 7498). With the service primitive 'indication' a service provider informs a service user about the occurrence of either an internal event or a service request issued by another service user. Notification of application in case of events in the Vector software components, e.g. an asynchronous reception of a CAN message.
Interaction Layer	The Interaction Layer is a communication component that is responsible for separating the driver (depending on the data link layer) and the application (independent of the physical layer). It provides a logical (signal) orientated interface. The communication layer representing the communication interface to the application. The communication services of the Interaction Layer are independent of both microcontroller and network protocol.
Interrupt	Processor-specific event which can interrupt the execution of a current program section.
Interrupt level	Processing level provided for time-critical activities. To keep the interrupt latency brief, only absolutely indispensable actions should be effected in the interrupt service routine, which ensures reception of the interrupt and trigger its (post) processing within a task. Other processing levels are: Operating system level and task level.
Interrupt service routine	The function used for direct processing of an interrupt.
Network	A network defines the assignment (1:N) between a logical communication grouping and a physical layer on which different modules are connected to (either CAN or LIN). One network consists of one channel, Y networks consists of 1..Z channel(s). We say network if we talk about more than one bus.
Network Management	The Network Managements manages the availability of different networks on a channel. It is responsible for the synchronized transition between the communication states 'sleep', 'prepare sleep' and 'active' for all modules. Network Management serves to ensure the safety and availability of the communications network of autonomous control units. The OSEK NM distinguishes between node-related (local) activities, e.g. initialization of the node and network-related (global) activities, e.g. coordination of global NM operating modes.
Operating system	All management and executive programs of a computer and its resources comprising an available and clearly defined interface.
PCI	PCI is the abbreviation of "Protocol Control Information". This Information is needed to pass a SDU from one instance of a specific protocol layer to another instance. E.g. it contains source and target information.

	The PCI is added by a protocol layer on the transmission side and is removed again on the receiving side.
PDU	<p>PDU is the abbreviation of “Protocol Data Unit”. The PDU contains SDU and PCI.</p> <p>On the transmission side the PDU is passed from the upper layer to the lower layer, which interprets this PDU as its SDU.</p>
Post-build	This type of configuration is possible after building the software module or the ECU software. The software may either receive parameters of its configuration during the download of the complete ECU software resulting from the linkage of the code, or it may receive its configuration file that can be downloaded to the ECU separately, avoiding a re-compilation and re-build of the ECU software modules. In order to make the post-build time re-configuration possible, the re-configurable parameters shall be stored at a known memory location of ECU storage area.
Schedule table	Table containing the sequence of LIN message identifiers to be transmitted together with the message delay.
Scheduler	The algorithm which decides whether a task switch is to be affected and which triggers all necessary internal activities of the operating system is named scheduler. The scheduler decides whether a task switch is possible according to the implemented scheduling policy. The scheduler can be considered as a resource which can be occupied and released by tasks. Thus a task can reserve the scheduler to avoid task switch until the scheduler is released.
SDU	SDU is the abbreviation of “Service Data Unit”. It is the data passed by an upper layer, with the request to transmit the data. It is as well the data which is extracted after reception by the lower layer and passed to the upper layer. A SDU is part of a PDU.
Signal	A signal is responsible for the logical transmission and reception of information depending on the restrictions of the physical layer. The definition of the signal contents is part of the database given by the vehicle manufacturer. Signals describe the significance of the individual data segments within a message. Typically bits, bytes or words are used for data segments but individual bit combinations are also possible. In the CAN data base, each data segment is assigned a symbolic name, a value range, a conversion formula and a physical unit, as well as a list of receiving nodes.
Transport Protocol	Some information that must be transferred over the CAN/LIN bus does not fit into individual message frames because the data length exceeds the maximum of 8 bytes. In this case, the sender must divide up the data into a number of messages. Additional information is necessary for the receiver to put the data together again.
Use case	A model of the usage by the user of a system in order to realize a single functional feature of the system.
User interface	A user interface (also called human machine interface or rarely man machine interface, since the political incorrectness) includes all the input and output hardware and software components which permit the interaction between the user (commonly the driver) and the on-board information systems (for instance a navigation system, a mobile telephone, etc.). Literally according to this definition, many elements can be considered as an User Interface (from the steering wheel to the LCD



of the infotainment system). Nevertheless, it is preferable to separate between the primary actuators (where all the devices to manage the drive are included, namely the pedals, the gear box, the steering wheel, etc.) and the systems to support the so-called secondary task (i.e. all the functions related to the infotainment, telematic and multimedia domains). The second ones are exactly the user interface as commonly intended within the automotive and consumer electronic worlds. Finally, all the user interfaces designed and developed for the vehicle have to absolutely be easy to use and not distractive. Therefore, the full respect of the usability requirements are the main aim of this domain.

Table 8-1 Glossary

## 8.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
ASAP2	Standardised interface for the description data defined by Arbeitskreis zur Standardisierung von Applikationshilfsmitteln (Working Group on the Standardisation of Application Tools)
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
CAN	Controller Area Network protocol originally defined for use as a communication network for control applications in vehicles.
CCC	Configuration Conformance Class in AUTOSAR. COM supports CCC3.
COM	Communication Layer
COMXXX	AUTOSAR COM Requirement ID. Replace XXX by the numeric identifier.
COMM	Communication Manager
CRC	Cyclic Redundancy Check
DBC	CAN data base format of the Vector company which is used by Vector tools
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DLC	Data Length Code, Number of data bytes of a CAN message
DM	Deadline Monitoring
DSP	Digital Signal Processing
EAD	Embedded Architecture Designer
ECU	Electronic control unit
ESCANXXXXXXXX	Vector PES Clearquest Database ID. Replace XXXXXXXX by the numeric identifier.
FIBEX	Field Bus Exchange
HIS	Hersteller Initiative Software

ICC	Implementation (Cluster) Conformance Class in AUTOSAR, this implementation supports ICC3
ID	Identifier (e.g. Identifier of a CAN message)
IL	Interaction Layer
I-PDU	Interaction Layer - Protocol Data Unit
ISR	Interrupt Service Routine
LDF	LIN Configuration Description File
LIN	Local Interconnect Network
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
MSB	Most Significant Bit
OEM	Original Equipment Manufacturer
OS	Operating System
PCI	Protocol Control Information
PDU	Protocol Data Unit
PDUR	PDU Router
PPort	Provide Port
ROM	Read-Only Memory
RPort	Require Port
RTE	Runtime Environment
SCHM	Basic Software Scheduler
SDU	Service Data Unit
SRS	Software Requirement Specification
SWC	Software Component
SWS	Software Specification
TMS	Transmission Mode Selector
TP	Transport Protocol
XML	eXtensible Markup Language

Table 8-2 Abbreviations

## 9 Contact

Visit our website for more information on

- > News
- > Products
- > Demo software
- > Support
- > Training data
- > Addresses

**[www.vector-informatik.com](http://www.vector-informatik.com)**