Document Title	Specification of Communication Stack Types
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	050
Document Classification	Standard

Document Version	3.2.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Change Description
28.10.2011	3.2.0	AUTOSAR Administration	• ComStack Artifacts have been generated from BSW Model • Update of SWS document for new traceability mechanism
11.10.2010	3.1.0	AUTOSAR Administration	• Add TPParameterType and Enumeration value TP_NORETRY in RetryInfoType • ComStack_Types.h divided into ComStack_Types.h and ComStack_Cfg.h • PduIdType and PduLengthType defined in ComStack_Cfg.h file
02.12.2009	3.0.0	AUTOSAR Administration	• Typo errors are corrected throughout the document • General return codes for NotifResultType has been added to support Tp_ChangeParameterRequest • TpDataStateType and RetryInfoType has been added to store the Tp buffer status information • Common Published information has been updated • Legal disclaimer revised
23.06.2008	2.2.1	AUTOSAR Administration	Legal disclaimer revised
27.11.2007	2.2.0	AUTOSAR Administration	• Chapter numbers in chapter 8.1 corrected • New data type NetworkHandleType created according item COMTYPE026 established • Syntax correction in PduInfoType • Document meta information extended • Small layout adaptations made

Document Change History

24.01.2007	2.1.1	AUTOSAR Administration	• "Advice for users" revised • "Revision Information" added • Changed "sender" to "receiver" at NTFRSLT_E_WFT_OVRN
14.12.2006	2.1.0	AUTOSAR Administration	• NTFRSLT_E_TIMEOUT_Bs changed NTFRSLT_E_TIMEOUT_BS • NTFRSLT_E_TIMEOUT_Cr changed to NTFRSLT_E_TIMEOUT_CR • Definitions according to compiler abstraction added • Legal disclaimer revised
21.03.2006	2.0.0	AUTOSAR Administration	• Initial release (The V1.0.0 was only as Pre-Release available within Release 1.0)

Disclaimer

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

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

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

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

1 Introduction and functional overview

This document specifies the AUTOSAR communication stack type header file. It contains all types that are used across several modules of the communication stack of the basic software and all types of all basic software modules that are platform and compiler independent.

It is strongly recommended that those communication stack type files are unique within the AUTOSAR community to guarantee unique types and to avoid type changes when changing from supplier A to B.

2 Acronyms and abbreviations

Acronyms and abbreviations that have a local scope are not contained in the AUTOSAR glossary. These must appear in a local glossary.

Acronym:	Description:
API	Application Programming Interface
DCM	Diagnostic Communication Manager
I-PDU	Interaction Layer PDU. In AUTOSAR the Interaction Layer is equivalent to the Communication Services Layer.
L-PDU	Data Link Layer PDU. In AUTOSAR the Data Link Layer is equivalent to the Communication Hardware Abstraction and Microcontroller Abstraction Layer.
N-PDU	Network Layer PDU. In AUTOSAR the Network Layer is equivalent to the Transport Protocol.
OSEK/VDX	In May 1993 OSEK has been founded as a joint project in the German automotive industry aiming at an industry standard for an open-ended architecture for distributed control units in vehicles. OSEK is an abbreviation for the German term "Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug" (English: Open Systems and the Corresponding Interfaces for Automotive Electronics). Initial project partners were BMW, Bosch, DaimlerChrysler, Opel, Siemens, VW and the IIIT of the University of Karlsruhe as co-ordinator. The French car manufacturers PSA and Renault joined OSEK in 1994 introducing their VDX-approach (Vehicle Distributed eXecutive) which is a similar project within the French automotive industry. At the first workshop on October 1995 the OSEK/VDX group presented the results of the harmonised specification between OSEK and VDX. After the 2nd international OSEK/VDX Workshop in October 1997 the 2nd versions of the specifications were published.
PDU	Protocol Data Unit
SDU	Service Data Unit - Payload of PDU
TP	Transport Protocol

Abbreviation:	Description:
Com	Communication
EcuC	ECU Configuration
e.g.	[lat.] exempli gratia = [eng.] for example
i.e.	[lat.] it est = [eng.] that is

3 Related documentation

3.1 Input documents

[GeneralSRS] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[SRSSPAL] General Requirements on SPAL
AUTOSAR_SRS_SPALGeneral.pdf

[StdTypes] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf

[PltfTypes] Specification of Platform Types
AUTOSAR_SWS_PlatformTypes.pdf

[CompTypes] Specification of Compiler Abstraction
AUTOSAR_SWS_CompilerAbstraction.pdf

[CANTP] Specification of CAN Transport Layer
AUTOSAR_SWS_CANTransportLayer.pdf

[FlexRayTP] Specification of FlexRay Transport Layer
AUTOSAR_SWS_FlexRayTransportLayer.pdf

[CANTRCV] Specification of CAN Transceiver Driver
AUTOSAR_SWS_CANTransceiverDriver.pdf

[FRTRCV] Specification of FlexRay Transceiver Driver
AUTOSAR_SWS_FlexRayTransceiverDriver.pdf

[BSMDT]Basic Software Module Description Template,
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

[BSWModule]List of Basic Software Modules
AUTOSAR_TR_BSWModuleList

3.2 Related standards and norms

[CProgLang] ISO/IEC 9899:1990 Programming Language – C
[ISONM] ISO/IEC 15765-2; 2003 Diagnostics on Controller Area Networks (CAN) –
Network layer services

4 Constraints and assumptions

4.1 Limitations

No limitations.

4.2 Applicability to car domains

No limitations.

4.3 Applicability to safety related environments

No restrictions, because the subject of this specification is a header file specifying types. It does not include or implement any functionality.

5 Software Architecture

5.1 Dependencies to other modules

The communication stack type header file defines communication types based on the platform types [PltfTypes] (Platform_Types.h) and Compiler (Compiler.h) header file [CompTypes]. To prevent multiple includes of header files, the communication stack header file includes the standard types header file [StdTypes] which already includes both other files.

5.2 File structure

[COMTYPE001] ⌈The include file structure shall be as follows:

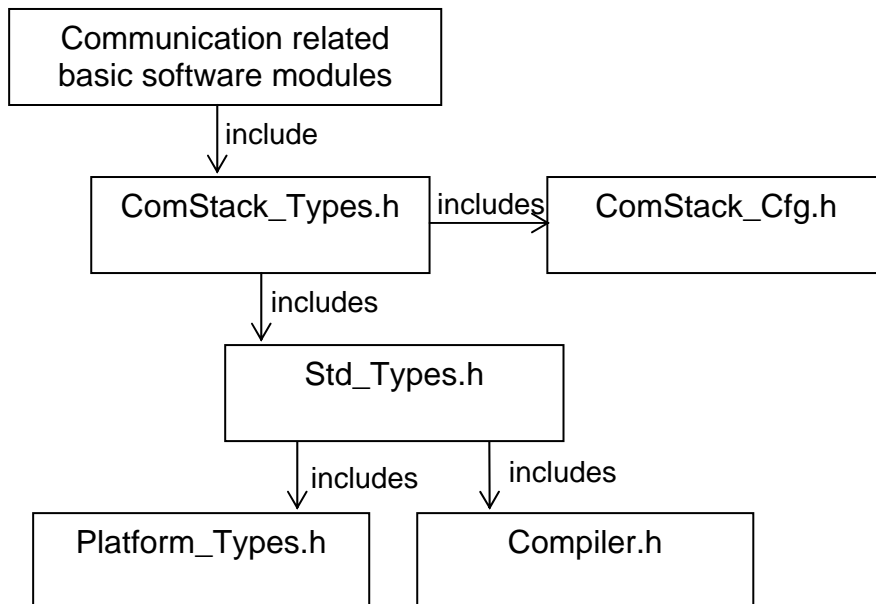**Figure 1: Include File Structure**

`ComStack_Types.h` shall include `Std_Types.h` and `ComStack_Cfg.h`
`Std_Types.h` shall include `Platform_Types.h`
`Std_Types.h` shall include `Compiler.h`
Communication related basic software modules shall include `ComStack_Types.h`
Communication related basic software modules shall <u>not</u> include `Std_Types.h` directly. ⌋ (BSW00384)

6 Requirements traceability

Requirement	Satisfied by
-	COMTYPE005
-	COMTYPE004
-	COMTYPE006
-	COMTYPE022
-	COMTYPE015
-	COMTYPE018
-	COMTYPE007
-	COMTYPE029
-	COMTYPE017
-	COMTYPE008
-	COMTYPE027
-	COMTYPE021
-	COMTYPE030
-	COMTYPE012
-	COMTYPE003
-	COMTYPE014
-	COMTYPE031
-	COMTYPE026
-	COMTYPE028
-	COMTYPE013
-	COMTYPE016
-	COMTYPE011
-	COMTYPE019
-	COMTYPE020
-	COMTYPE010
BSW00300	COMTYPE035
BSW00301	COMTYPE035
BSW00302	COMTYPE035
BSW00304	COMTYPE035
BSW00306	COMTYPE035
BSW00307	COMTYPE035
BSW00308	COMTYPE035
BSW00309	COMTYPE035
BSW00310	COMTYPE035
BSW00312	COMTYPE035
BSW00314	COMTYPE035

Document ID 050: AUTOSAR_SWS_CommunicationStackTypes

BSW00321	COMTYPE035
BSW00323	COMTYPE035
BSW00324	COMTYPE035
BSW00325	COMTYPE035
BSW00326	COMTYPE035
BSW00327	COMTYPE035
BSW00328	COMTYPE035
BSW00329	COMTYPE035
BSW00330	COMTYPE035
BSW00331	COMTYPE035
BSW00333	COMTYPE035
BSW00334	COMTYPE035
BSW00335	COMTYPE035
BSW00336	COMTYPE035
BSW00337	COMTYPE035
BSW00338	COMTYPE035
BSW00339	COMTYPE035
BSW00341	COMTYPE035
BSW00342	COMTYPE035
BSW00343	COMTYPE035
BSW00344	COMTYPE035
BSW00345	COMTYPE035
BSW00346	COMTYPE035
BSW00347	COMTYPE035
BSW00348	COMTYPE035
BSW00350	COMTYPE035
BSW00353	COMTYPE035
BSW00355	COMTYPE035
BSW00357	COMTYPE035
BSW00358	COMTYPE035
BSW00359	COMTYPE035
BSW00360	COMTYPE035
BSW00361	COMTYPE035
BSW00369	COMTYPE035
BSW00370	COMTYPE035
BSW00371	COMTYPE035
BSW00373	COMTYPE035
BSW00374	COMTYPE035
BSW00375	COMTYPE035
BSW00376	COMTYPE035
BSW00377	COMTYPE035

BSW00378	COMTYPE035
BSW00379	COMTYPE035
BSW00380	COMTYPE035
BSW00381	COMTYPE035
BSW00383	COMTYPE035
BSW00384	COMTYPE001
BSW00385	COMTYPE035
BSW00386	COMTYPE035
BSW00387	COMTYPE035
BSW00388	COMTYPE035
BSW00389	COMTYPE035
BSW00390	COMTYPE035
BSW00391	COMTYPE035
BSW00392	COMTYPE035
BSW00393	COMTYPE035
BSW00394	COMTYPE035
BSW00395	COMTYPE035
BSW00396	COMTYPE035
BSW00397	COMTYPE035
BSW00398	COMTYPE035
BSW00399	COMTYPE035
BSW00400	COMTYPE035
BSW00401	COMTYPE035
BSW00404	COMTYPE035
BSW00405	COMTYPE035
BSW00406	COMTYPE035
BSW00407	COMTYPE035
BSW00408	COMTYPE035
BSW00409	COMTYPE035
BSW00410	COMTYPE035
BSW00411	COMTYPE035
BSW00412	COMTYPE035
BSW00414	COMTYPE035
BSW00415	COMTYPE035
BSW00416	COMTYPE035
BSW00417	COMTYPE035
BSW00420	COMTYPE035
BSW00421	COMTYPE035
BSW00422	COMTYPE035
BSW00423	COMTYPE035
BSW00424	COMTYPE035

Document ID 050: AUTOSAR_SWS_CommunicationStackTypes

BSW00425	COMTYPE035
BSW00426	COMTYPE035
BSW00427	COMTYPE035
BSW00428	COMTYPE035
BSW00429	COMTYPE035
BSW00431	COMTYPE035
BSW00432	COMTYPE035
BSW00433	COMTYPE035
BSW00434	COMTYPE035
BSW005	COMTYPE035
BSW006	COMTYPE035
BSW007	COMTYPE035
BSW009	COMTYPE035
BSW010	COMTYPE035
BSW101	COMTYPE035
BSW158	COMTYPE035
BSW159	COMTYPE035
BSW160	COMTYPE035
BSW161	COMTYPE035
BSW162	COMTYPE035
BSW164	COMTYPE035
BSW167	COMTYPE035
BSW168	COMTYPE035
BSW170	COMTYPE035
BSW171	COMTYPE035
BSW172	COMTYPE035

7 Functional specification

7.1 General issues

[COMTYPE003] ⌈The file name of the communication stack type header file shall be 'ComStack_Types.h' and configuration header file shall be 'ComStack_Cfg.h'.⌋ ( )

[COMTYPE004] ⌈It is not allowed to add any project or supplier specific extension to this file. Any extension invalidates the AUTOSAR conformity. ⌋ ( )

[COMTYPE015] ⌈Because many of the communication stack type are depending on the appropriate ECU, this file shall be generated dependent on the specific ECU configuration for each ECU independently. ⌋ ( )

[COMTYPE016] ⌈The communication stack type header files shall be protected against multiple inclusion:
ComStack_Types.h
```
#ifndef COMSTACK_TYPES_H
#define COMSTACK_TYPES_H
..
/*
 * Contents of file
 */
..
#endif /* COMSTACK_TYPES_H */
```

ComStack_Cfg.h
```
#ifndef COMSTACK_CFG_H
#define COMSTACK_CFG_H
..
/*
 * Contents of file
 */
..
#endif /* COMSTACK_CFG_H */
```
⌋ ( )

[COMTYPE029] ⌈ComStack_Cfg.h shall be generated by the generator to generate the type definition of the PduIdType and PduLengthType from the EcuC Virtual Layer based on the configuration e.g. typedef uint8 PduIdType if number of PDUs are less than 256. ⌋ ( )

[COMTYPE030] ⌈The value of PduIdType and PduLengthType shall be derived from the 'PduIdTypeEnum' and 'PduLengthTypeEnum' of the EcuCPduCollection container respectively. ⌋ ( )

8 API specification

8.1 Type definitions

8.1.1 PduIdType

Name:	PduIdType	
Type:	uint8, uint16	
Range:	0...<PduIdmax>	– Zero-based integer number – The size of this global type depends on the maximum number of PDUs used within one software module. This parameter shall be generated by the generator tool depending on the value configured in EcuC virtual layer. This parameter shall be generated in ComStack_Cfg.h file Example : If '''no''' software module deals with more PDUs that 256, this type can be set to uint8. If at least one software module handles more than 256 PDUs, this type must globally be set to uint16.
Description:	COMTYPE005: This type is used within the entire AUTOSAR Com Stack except for bus drivers. COMTYPE006: Variables of this type serve as a unique identifier of a PDU within a software module or a set thereof, and also for interaction of two software modules where the PduId of the corresponding target module is being used for referencing. COMTYPE007: In order to be able to perform table-indexing within a software module, variables of this type shall be zero-based and consecutive. There might be several ranges of PduIds in a module, one for each type of operation performed within that module (e.g. sending and receiving). COMTYPE014: PduIdmax, the maximum number of a PduId range, is the number -1 of PDUs dealt with in the corresponding type of operation within that module.	

8.1.2 PduLengthType

Name:	PduLengthType	
Type:	uint8, uint16, uint32	
Range:	0...<PduLengthmax>	– Zero-based integer number – The size of this global type depends on the maximum length of PDUs to be sent by an ECU. This parameter shall be generated by the generator tool depending on the value configured in EcuC virtual layer. This parameter shall be generated in ComStack_Cfg.h file Example : If no segmentation is used the length depends on the maximum payload size of a frame of the underlying communication system (for FlexRay maximum size is 255, therefore uint8). If segmentation is used it depends on the maximum length of a segmented N-PDU (in general uint16 is used)
Description:	COMTYPE008: This type shall be used within the entire AUTOSAR Com Stack of	

an ECU except for bus drivers. COMTYPE010: Variables of this type serve as length information of a PDU. The length information is provided in number of bytes. COMTYPE017: PduLengthmax, the maximum length of a Pdu, is the length of the largest (possibly segmented) PDU to be sent by the ECU.

8.1.3 PduInfoType

Name:	PduInfoType		
Type:	Structure		
Element:	uint8*	SduDataPtr	pointer to the SDU (i.e. payload data) of the PDU. The type of this pointer depends on the memory model being used at compile time.
	PduLengthType	SduLength	length of the SDU in bytes
Description:	COMTYPE011: Variables of this type shall be used to store the basic information about a PDU of any type, namely a pointer variable pointing to it's SDU (payload), and the corresponding length of the SDU in bytes.		

8.1.4 TPParameterType

Name:	TPParameterType	
Type:	Enumeration	
Range:	TP_STMIN	Separation Time
	TP_BS	Block Size
	TP_BC	The Band width control parameter used in FlexRay transport protocol module.
Description:	COMTYPE031: Specify the parameter to which the value has to be changed (BS or STmin).	

8.1.5 BufReq_ReturnType

Name:	BufReq_ReturnType	
Type:	Enumeration	
Range:	BUFREQ_OK	Buffer request accomplished successful. This status shall have the value 0.
	BUFREQ_E_NOT_OK	Buffer request not successful. Buffer cannot be accessed. This status shall have the value 1.
	BUFREQ_E_BUSY	Temporarily no buffer available. It's up the requester to retry request for a certain time. This status shall have the value 2.
	BUFREQ_E_OVFL	No Buffer of the required length can be provided. This status shall have the value 3.
Description:	COMTYPE0012: Variables of this type shall be used to store the result of a buffer request.	

8.1.6 NotifResultType

Name:	NotifResultType		
Type:	uint8		
Range:	0x00 – 0x1E	--	General return codes. A detailed specification is listed below.
	0x1F - 0x3C	--	Error notification: Error notification codes specific for the communication system CAN. For a detailed definition please refer to the AUTOSAR specification of CAN TP [CANTP].
	0x3D – 0x5A	--	Error notification: Error notification codes specific for the communication system LIN. A detailed definition is still open, because currently there is not AUTOSAR specification of Lin TP.
	0x5B – 0x78	--	Error notification: Error notification codes specific for the communication system FlexRay. For a detailed definition please refer to the AUTOSAR specification of FlexRay TP [FlexRayTP].
	> 0x78	--	Currently values in this range are invalid. In future it might be possible that further return codes are specified for other communication systems.
	NTFRSLT_E_ABORT	0x09	Error notification: Flow control (FC) N_PDU with FlowStatus = ABORT received. It indicates an abort of a transmission. A possible reason for this is that the receiver is currently busy and can not take the request at that point in time.
	NTFRSLT_E_CANCELATION_OK	0x0B	Action has been successfully finished: Requested cancellation has been executed.
	NTFRSLT_E_CANCELATION_NOT_OK	0x0C	Error notification: Due to an internal error the requested cancellation has not been executed. This will happen e.g., if the to be canceled transmission has been executed already.
	NTFRSLT_PARAMETER_OK	0x0D	The parameter change request has been successfully executed
	NTFRSLT_E_PARAMETER_NOT_OK	0x0E	The request for the change of the parameter did not complete successfully
	NTFRSLT_E_RX_ON	0x0F	The parameter change request not

Document ID 050: AUTOSAR_SWS_CommunicationStackTypes

			executed successfully due to an ongoing reception
	`NTFRSLT_E_VALUE_NOT_OK`	`0x10`	The parameter change request not executed successfully due to a wrong value
	–	`0x11-` `0x1E`	Reserved values for future usage.
Description:			COMTYPE013: Variables of this type shall be used to store the result status of a notification (confirmation or indication). (Currently this type is only used for communication between DCM and TP to enable the notification that an error has occurred and a dedicated buffer can be unlocked.)

Return code specification:

[COMTYPE018] ⌈General Codes

Return code	Value	Description
NTFRSLT_OK	0x00	Action has been successfully finished: • message sent out (in case of confirmation), • message received (in case of indication)
NTFRSLT_E_NOT_OK	0x01	Error notification: • message not successfully sent out (in case of confirmation), • message not successfully received (in case of indication)
NTFRSLT_E_TIMEOUT_A	0x02	Error notification: • timer N_Ar/N_As (according to ISO specification [ISONM]) has passed its time-out value N_Asmax/N_Armax. This value can be issued to service user on both the sender and receiver side.
NTFRSLT_E_TIMEOUT_BS	0x03	Error notification: timer N_Bs has passed its time-out value N_Bsmax (according to ISO specification [ISONM]). This value can be issued to the service user on the sender side only.
NTFRSLT_E_TIMEOUT_CR	0x04	Error notification: timer N_Cr has passed its time-out value N_Crmax. This value can be issued to the service user on the receiver side only.
NTFRSLT_E_WRONG_SN	0x05	Error notification: unexpected sequence number (PCI.SN) value

Return code	*Value*	*Description*
		received. This value can be issued to the service user on the receiver side only.
NTFRSLT_E_INVALID_FS	0x06	Error notification: invalid or unknown FlowStatus value has been received in a flow control (FC) N_PDU. This value can be issued to the service user on the sender side only.
NTFRSLT_E_UNEXP_PDU	0x07	Error notification: unexpected protocol data unit received. This value can be issued to the service user on both the sender and receiver side.
NTFRSLT_E_WFT_OVRN	0x08	Error notification: flow control WAIT frame that exceeds the maximum counter N_WFTmax received. This value can be issued to the service user on the receiver side.
NTFRSLT_E_ABORT	0x09	Error notification: Flow control (FC) N_PDU with FlowStatus = ABORT received. It indicates an abort of a transmission. A possible reason for this is that the receiver is currently busy and can not take the request at that point in time.
NTFRSLT_E_NO_BUFFER	0x0A	Error notification: Flow control (FC) N_PDU with FlowStatus = OVFLW received. It indicates that the buffer on the receiver side of a segmented message transmission cannot store the number of bytes specified by the FirstFrame DataLength (FF_DL) parameter in the FirstFrame and therefore the transmission of the segmented message was aborted.

Return code	Value	Description
		No buffer within the TP available to transmit the segmented I-PDU. This value can be issued to the service user on both the sender and receiver side.
NTFRSLT_E_CANCELATION_OK	0x0B	Action has been successfully finished: Requested cancellation has been executed.
NTFRSLT_E_CANCELATION_NOT_OK	0x0C	Error notification: Due to an internal error the requested cancellation has not been executed. This will happen e.g., if the to be canceled transmission has been executed already.
NTFRSLT_PARAMETER_OK	0x0D	The parameter change request has been successfully executed
NTFRSLT_E_PARAMETER_NOT_OK	0x0E	The request for the change of the parameter did not complete successfully
NTFRSLT_E_RX_ON	0x0F	The parameter change request not executed successfully due to an ongoing reception
NTFRSLT_E_VALUE_NOT_OK	0x10	The parameter change request not executed successfully due to a wrong value
-	0x11-0x1E	Reserved values for future usage.

⌋()

[COMTYPE019] ⌈The Communication System dependent Return codes shall be named as follows:

NTFRSLT_E_<Communication System Abbreviation>_<Error Code Name>.

Communication System Abbreviation:

CAN: for Controller area network

LIN: for Local Interconnect Network

FR: for FlexRay

Error Code Name: self explaining name of error return code.

Example for a FlexRay specific return value:

NTFRSLT_E_FR_NEG_ACK: Negative acknowledgement on received⌋()

8.1.7 BusTrcvErrorType

Name:	BusTrcvErrorType		
Type:	uint8		
Range:	0x00 – 0x1E	– General	return codes.

		– A detailed specification is listed below.
	0x1F – 0x3C	– Error notification: – Error notification codes specific for the communication system CAN. For a detailed definition please refer to the AUTOSAR specification of CAN Transceiver Driver [CANTRCV].
	0x3D – 0x5A	– Error notification: – Error notification codes specific for the communication system LIN. A detailed definition is still open, because currently there is not AUTOSAR specification of Lin Interface.
	0x5B – 0x78	– Error notification: – Error notification codes specific for the communication system FlexRay. For a detailed definition please refer to the AUTOSAR specification of FlexRay Transceiver Driver [FRTRCV].
	> 0x78	– Currently values in this range are invalid. In future it might – be possible that further return codes are specified for other communication systems.
Description:		COMTYPE020: Variables of this type shall be used to return the bus status evaluated by a transceiver.

Return code specification:

[COMTYPE021] ⌈General Codes

Return code	Value	Description
BUSTRCV_OK	0x00	There is no bus transceiver error seen by the driver or transceiver does not support the detection of bus errors.
BUSTRCV_E_ERROR	0x01	Bus transceiver detected an unclassified error.
	0x02-0x1E	Reserved values for future usage.

⌋()

[COMTYPE022] ⌈The Communication System dependent Return codes shall be named as follows:

BUSTRCV_E_<Communication System Abbreviation>_<Error Code Name>.

Communication System Abbreviation:

CAN: for Controller area network

LIN: for Local Interconnect Network

FR: for FlexRay

Error Code Name: self explaining name of error return code.

Example for a CAN specific return value:

BUSTRCV_E_CAN_SINGLE: CAN bus transceiver has detected that the fault tolerant bus is in single wire mode.

8.1.8 TpDataStateType

Name:	TpDataStateType	
Type:	Enumeration	
Range:	TP_DATACONF	TP_DATACONF indicates that all data, that have been

		copied so far, are confirmed and can be removed from the TP buffer. Data copied by this API call are excluded and will be confirmed later.
	TP_DATARETRY	TP_DATARETRY indicates that this API call shall copy already copied data in order to recover from an error. In this case TxTpDataCnt specifies the offset of the first byte to be copied by the API call.
	TP_CONFPENDING	TP_CONFPENDING indicates that the previously copied data must remain in the TP
Description:		COMTYPE027: Variables of this type shall be used to store the state of TP buffer.

8.1.9 RetryInfoType

Name:	RetryInfoType		
Type:	Structure		
Element:	TpDataStateType	TpDataState	The enum type to be used to store the state of Tp buffer.
	PduLengthType	TxTpDataCnt	length of the SDU in bytes
Description:	Variables of this type shall be used to store the information about Tp buffer handling.		

8.1.10   NetworkHandleType

Name:	NetworkHandleType		
Type:	uint8		
Range:	0...255	--	Zero-based integer number
Description:	Variables of the type NetworkHandleType shall be used to store the identifier of a communication channel.		

8.2 Function definitions

Not applicable.

9 Sequence diagrams

Not applicable.

10 Configuration specification

10.1 Published parameters

[COMTYPE034] ⌈The standardized common published parameters as required by BSW00402 in the General Requirements on Basic Software Modules BSMDT shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules.⌋ (BSW00402, BSW004, BSW003, BSW00318)

Additional module-specific published parameters are listed below if applicable.

The standard common published information like

vendorId (COMSTACKTYPE_VENDOR_ID),
moduleId (COMSTACKTYPE _MODULE_ID),
arMajorVersion (COMSTACKTYPE_AR_MAJOR_VERSION),
arMinorVersion (COMSTACKTYPE_AR_MINOR_VERSION),
arPatchVersion (COMSTACKTYPE_AR_PATCH_VERSION),
swMajorVersion (COMSTACKTYPE_SW_MAJOR_VERSION),
swMinorVersion (COMSTACKTYPE_SW_MINOR_VERSION),
swPatchVersion (COMSTACKTYPE_SW_PATCH_VERSION),
vendorApiInfix (COMSTACKTYPE_VENDOR_API_INFIX)

is provided in the BSW Module Description Template (see BSMDT Figure 4.1 and Figure 7.1).

11 Not applicable requirements

[COMTYPE035] ⌈These requirements are not applicable to this specification.⌋ (BSW00344, BSW00404, BSW00405, BSW00345, BSW159, BSW167, BSW171, BSW170, BSW00380, BSW00381, BSW00412, BSW00383, BSW00387, BSW00388, BSW00389, BSW00390, BSW00391, BSW00392, BSW00393, BSW00394, BSW00395, BSW00396, BSW00397, BSW00398, BSW00399, BSW00400, BSW00375, BSW101, BSW00416, BSW00406, BSW168, BSW00407, BSW00423, BSW00424, BSW00425, BSW00426, BSW00426, BSW00427, BSW00428, BSW00429, BSW00431, BSW00432, BSW00433, BSW00434, BSW00336, BSW00337, BSW00338, BSW00369, BSW00339, BSW00421, BSW00422, BSW00420, BSW00417, BSW00323, BSW00409, BSW00385, BSW00386, BSW161, BSW162, BSW00324, BSW005, BSW00415, BSW164, BSW00325, BSW00326, BSW00342, BSW00343, BSW160, BSW007, BSW00300, BSW00347, BSW00307, BSW00310, BSW00373, BSW00327, BSW00335, BSW00350, BSW00408, BSW00410, BSW00411, BSW00346, BSW158, BSW00314, BSW00370, BSW00348, BSW00353, BSW00361, BSW00301, BSW00302, BSW00328, BSW00312, BSW006, BSW00357, BSW00377, BSW00304, BSW00355, BSW00378, BSW00306, BSW00308, BSW00309, BSW00371, BSW00358, BSW00414, BSW00376, BSW00359, BSW00360, BSW00329, BSW00330, BSW00331, BSW009, BSW00401, BSW172, BSW010, BSW00333, BSW00374, BSW00379, BSW00321, BSW00341, BSW00334)