Document Title	Specification of Module XCP
Document Category	Software specification (SWS)
Document UID	412
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R26-11

Document Change History

Date	Release	Changed by	Description
2026-11-26	R26-11	AUTOSAR Release Management	• Support DMA transfer approach • Removal of RS dependency • Editorial changes
2025-11-27	R25-11	AUTOSAR Release Management	• Add NvM_ReadBlock and NvM_WriteBlock APIs as optional interfaces • Removal of R24-11 obsolete elements • Editorial changes
2024-11-27	R24-11	AUTOSAR Release Management	• adapted interaction with lower layer to LSduR
2023-11-23	R23-11	AUTOSAR Release Management	• Removed all HandleId configuration parameters
2022-11-24	R22-11	AUTOSAR Release Management	• Editorial changes
2021-11-25	R21-11	AUTOSAR Release Management	• No content changes
2020-11-30	R20-11	AUTOSAR Release Management	• Improve the structure of the 'error sections'
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Update XCP on CAN version to support CAN FD

△

2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes • Update development errors • Adapt XCP with CAN return types • Add new configuration container XcpChannel
2016-11-30	4.3.0	AUTOSAR Release Management	• Editorial changes • Harmonize descriptions of identical API functions • Removal of unused artifacts and obsolete elements
2015-07-31	4.2.2	AUTOSAR Release Management	• Debugging support marked as obsolete • Editorial changes • Modifications in some parameters multiplicity of XcpDaqlist container
2014-10-31	4.2.1	AUTOSAR Release Management	• Editorial changes • Minor corrections • Changed the multiplicity of XcpEventChannelTriggeredDaqListRef • Remove limitation “Flash Programming for ECU development purposes”
2014-03-31	4.1.3	AUTOSAR Release Management	• Editorial correction for faulty references links • Minor editorial correction for SWS_Xcp_00841, SWS_Xcp_00844 • Changed Xcp_RxIndication argument from PduInfoType * to const PduInfoType *
2013-10-31	4.1.2	AUTOSAR Release Management	• Minor corrections • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Reclassify XCP_E_INIT_FAILED from class production error to development

▽

△

2013-03-15	4.1.1	AUTOSAR Administration	• Added parameters for Event Channel and Timestamp configuration • Added possibility to calculate memory consumption for ODT (DAQ & STIM) • Restructuring configuration parameters for static & dynamic ODT • Added support for deactivation of transmission capabilities
2011-12-22	4.0.3	AUTOSAR Administration	• Add chapter 7.8 (Version check), RTE limitation, OS Counter Ref • Remove InstanceID and known limitation (OS)
2010-09-30	3.1.5	AUTOSAR Administration	• Initial Release

Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction and functional overview 8
2 Acronyms and Abbreviations 9
3 Related documentation 10
3.1 Input documents & related standards and norms . . . 10
3.2 Related specification . . . 10
4 Constraints and assumptions 11
4.1 Limitations . . . 11
4.2 Applicability to car domains . . . 11
5 Dependencies to other modules 12
5.1 AUTOSAR RTE (BSW Scheduler) . . . 12
5.2 AUTOSAR PDU Routing module (PduR) . . . 12
5.3 AUTOSAR FlexRay Interface . . . 12
5.4 AUTOSAR CAN Interface . . . 12
5.5 AUTOSAR SocketAdaptor . . . 12
5.6 AUTOSAR RTE . . . 12
5.7 AUTOSAR OS . . . 13
5.8 AUTOSAR Diagnostic Event Manager . . . 13
5.9 AUTOSAR Default Error Tracer . . . 13
5.10 File structure . . . 13
5.10.1 Code file structure . . . 13
6 Requirements Tracing 14
7 Functional specification 16
7.1 XCP on CAN . . . 18
7.2 XCP on FlexRay . . . 18
7.3 XCP on Ethernet . . . 20
7.4 General Requirements . . . 21
7.5 Message copying . . . 21
7.6 Error Classification . . . 22
7.6.1 Development Errors . . . 23
7.6.2 Runtime Errors . . . 23
7.6.3 Transient Faults . . . 23
7.6.4 Production Errors . . . 23
7.6.5 Extended Production Errors . . . 23
7.7 Security Events . . . 23
8 API specification 24
8.1 Imported types . . . 24
8.2 Type definitions . . . 24
8.2.1 Xcp_ConfigType . . . 24

8.2.2 Xcp_Transmission Mode Type . . . 25
8.3 Function definitions . . . 25
8.3.1 Xcp_Init . . . 25
8.3.2 Xcp_SetTransmissionMode . . . 26
8.3.3 Xcp_RxIndication . . . 26
8.3.4 Xcp_TxConfirmation . . . 27
8.3.5 Xcp_TriggerTransmit . . . 28
8.4 Scheduled functions . . . 29
8.4.1 Xcp_MainFunction . . . 30
8.5 Expected interfaces . . . 30
8.5.1 Mandatory interfaces . . . 30
8.5.2 Optional interfaces . . . 30
8.5.3 Callout interfaces . . . 31
8.5.4 Configurable interfaces . . . 31
9 Sequence diagrams 32
9.1 XCP on FlexRay . . . 32
9.1.1 Xcp on FlexRay Transmit . . . 32
9.1.2 Xcp on FlexRay Receive Indication . . . 32
9.2 XCP on CAN . . . 33
9.2.1 Xcp on CAN Transmit . . . 33
9.2.2 Xcp on CAN Receive Indication . . . 33
9.3 XCP on Ethernet . . . 34
9.3.1 Xcp on Ethernet Transmit . . . 34
9.3.2 Xcp on Ethernet Receive Indication . . . 34
10 Configuration specification 35
10.1 How to read this chapter . . . 35
10.2 Containers and configuration parameters . . . 35
10.2.1 Xcp . . . 36
10.2.2 XcpGeneral . . . 36
10.2.3 XcpConfig . . . 45
10.2.4 XcpDaqList . . . 46
10.2.5 XcpDto . . . 48
10.2.6 XcpOdt . . . 49
10.2.7 XcpOdtEntry . . . 51
10.2.8 XcpEventChannel . . . 53
10.2.9 XcpPdu . . . 59
10.2.10 XcpRxPdu . . . 59
10.2.11 XcpTxPdu . . . 60
10.2.12 XcpCommunicationChannel . . . 61
10.2.13 XcpPageSwitching . . . 63
10.2.14 XcpSegment . . . 64
10.2.15 XcpPage . . . 66

10.3 Published Information 67
A Not applicable requirements 68
B Change history of AUTOSAR traceable items 69
B.1 Traceable item history of this document according to AUTOSAR Release R22-11 69
B.1.1 Added Specification Items in R22-11 69
B.1.2 Changed Specification Items in R22-11 71
B.1.3 Deleted Specification Items in R22-11 71
B.2 Traceable item history of this document according to AUTOSAR Release R23-11 72
B.2.1 Added Specification Items in R23-11 72
B.2.2 Changed Specification Items in R23-11 74
B.2.3 Deleted Specification Items in R23-11 74
B.3 Traceable item history of this document according to AUTOSAR Release R24-11 74
B.3.1 Added Specification Items in R24-11 74
B.3.2 Changed Specification Items in R24-11 75
B.3.3 Deleted Specification Items in R24-11 75
B.3.4 Added Constraints in R24-11 75
B.3.5 Changed Constraints in R24-11 76
B.3.6 Deleted Constraints in R24-11 76
B.4 Traceable item history of this document according to AUTOSAR Release R25-11 76
B.4.1 Added Specification Items in R25-11 76
B.4.2 Changed Specification Items in R25-11 76
B.4.3 Deleted Specification Items in R25-11 76
B.4.4 Added Constraints in R25-11 76
B.4.5 Changed Constraints in R25-11 76
B.4.6 Deleted Constraints in R25-11 77
B.5 Traceable item history of this document according to AUTOSAR Release R26-11 77
B.5.1 Added Specification Items in R26-11 77
B.5.2 Changed Specification Items in R26-11 77
B.5.3 Deleted Specification Items in R26-11 77
B.5.4 Added Constraints in R26-11 77
B.5.5 Changed Constraints in R26-11 77
B.5.6 Deleted Constraints in R26-11 77

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module XCP

XCP is a protocol description (ASAM standard) between a master (tool) and a slave (device), which provides the following basic features:

- Synchronous data acquisition (measurement)
- Synchronous data stimulation (for rapid prototyping)
- Online memory calibration (read / write access)
- Calibration data page initialization and switching
- Flash Programming for ECU development purposes
- Every feature is optional and the access can be restricted
- Various communications busses are supported

XCP was designed according to the following principles:

- Minimal Slave resource consumption (RAM, ROM, runtime)
- Efficient communication
- Simple Slave implementation

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the XCP that are not included in the [1, AUTOSAR glossary].

Acronym:	Description:
AUTOSAR	AUTomotive Open System ARchitecture
A2L	File Extension for an ASAM 2MC Language File
ASAM	Association for Standardization of Automation and Measuring Systems
BSW	Basic Software
CAN	Controller Area Network
CanIf	CAN Interface
CTO	Command Transfer Object
DAQ	Data AcQuisition, Data AcQuisition Packet
DTO	Data Transfer Object
ECU	Electronic Control Unit
FrIf	FlexRay Interface
LPDU	Data Link Layer PDU
MCD	Measurement Calibration and Diagnostics
MISRA	Motor Industry Software Reliability Association
ODT	Object Descriptor Table
PDU	Protocol Data Unit
RAM	Random Access Memory
ROM	Read Only Memory
SchM	Schedule Manager
SVN	Subversion
SRS	Software Requirements Specification
STIM	Data Stimulation packet
SW	Software
SWS	Software Specification
TCP/IP	Transfer Control Protocol / Internet Protocol
TS	Time Stamp
UDP/IP	User Datagram Protocol / Internet Protocol
URL	Uniform Resource Locator
XCP	Universal Calibration Protocol
XML	Extensible Markup Language
ISR	Interrupt Service Routine
DET	Default Error Tracer (AUTOSAR BSW module)

Table 2.1: Acronyms used in the scope of this Document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Glossary
AUTOSAR_FO_TR_Glossary

[2] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[3] ASAM XCP - The Universal Measurement and Calibration Protocol: ASAM_XCP_Part1-Overview - Version 1.1
http://www.asam.net

[4] Specification of PDU Router
AUTOSAR_CP_SWS_PDURouter

[5] Specification of Linklayer Sdu Routing Module
AUTOSAR_CP_SWS_LSduRouter

[6] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[7] ASAM XCP - Transport Layer Specification XCP on CAN: ASAM_XCP_Part3 Transport-Layer-Specification_XCPonCAN - Version 1.2
http://www.asam.net

[8] ASAM XCP - Transport Layer Specification XCP on FlexRay: ASAM_XCP_Part3-Transport-Layer-Specification_XCPonFlexRay-Version 1.1
http://www.asam.net

[9] ASAM XCP - Transport Layer Specification XCP on Ethernet: ASAM_XCP_Part3-Transport-Layer-Specification_XCPonEthernet (TCP_IP&UDP_IP) - Version 1.1
http://www.asam.net

[10] System Template
AUTOSAR_CP_TPS_SystemTemplate

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [2, SWS BSW General], which is also valid for XCP.

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

4 Constraints and assumptions

4.1 Limitations

The following XCP features are currently out of scope:

- The SET_DAQ_ID command according to the XCP CAN Transport Layer Specification is not part of the AUTOSAR XCP module
- Currently, the AUTOSAR RTE does not offer APIs for direct communication with XCP
- For further details concerning the supported feature set, please refer to [3]
- NAX is only configurable through the ASAM configuration file A2L.

Please note:

For the communications bus LIN, no ASAM XCP is specified.

4.2 Applicability to car domains

n/a

5 Dependencies to other modules

This section describes the relations to other modules and files within the AUTOSAR basic software architecture. It contains brief descriptions of configuration information and services, which are required by the XCP module from other modules.

5.1 AUTOSAR RTE (BSW Scheduler)

The BSW Scheduler calls the main functions of the Xcp, which are necessary for the cyclic processes of the Xcp.

5.2 AUTOSAR PDU Routing module (PduR)

The PDU Routing module (PduR) [4] is the lower layer module of the XCP module for the data flow. Please note: The Linklayer Sdu Routing Module (LSduR) [5] is the lower layer of the PduR. XCP data are excanged via the PduR and LSduR with the according Bus interfaces.

5.3 AUTOSAR FlexRay Interface

The FlexRay Interface is used to transmit and receive XCP PDUs via FlexRay.

5.4 AUTOSAR CAN Interface

The CAN Interface is used to transmit and receive XCP PDUs via CAN.

5.5 AUTOSAR SocketAdaptor

The SocketAdaptor is used to transmit and receive XCP PDUs via Ethernet.

5.6 AUTOSAR RTE

The RTE is used for copying calibration parameters from ROM/FLASH to RAM and to use the double pointered method.

5.7 AUTOSAR OS

In order to be able to use the time stamped feature of XCP, an AUTOSAR OS Counter is used.

5.8 AUTOSAR Diagnostic Event Manager

In order to be able to report production errors, the XCP has to have access to the Diagnostic Event Manager.

5.9 AUTOSAR Default Error Tracer

In order to be able to report default errors, the XCP has to have access to the error hook of the Default Error Tracer.

5.10 File structure

5.10.1 Code file structure

[SWS_Xcp_00501]

Upstream requirements: SRS_BSW_00419, SRS_BSW_00383, SRS_BSW_00346, SRS_BSW_00380

⌈The code file structure shall not be defined within this specification completely. At this point it shall be pointed out that the code-file structure shall include the following files named:

- Xcp.c - general source code file of the module XCP
- Xcp_Cfg.c - for pre-compile time configurable parameters
- Xcp_Lcfg.c - for link time configurable parameters and
- Xcp_PBcfg.c - for post build time configurable parameters.

⌋

These files shall contain all link time and post-build time configurable parameters.

6 Requirements Tracing

The following tables reference the requirements specified in [6] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_Xcp_00803]
[SRS_BSW_00159]	All modules of the AUTOSAR Basic Software shall support a tool based configuration	[SWS_Xcp_00102]
[SRS_BSW_00167]	All AUTOSAR Basic Software Modules shall provide configuration rules and constraints to enable plausibility checks	[SWS_Xcp_00103] [SWS_Xcp_00104] [SWS_Xcp_00105]
[SRS_BSW_00327]	Error values naming convention	[SWS_Xcp_00763]
[SRS_BSW_00344]	BSW Modules shall support link-time configuration	[SWS_Xcp_00741]
[SRS_BSW_00345]	BSW Modules shall support pre-compile configuration	[SWS_Xcp_00742]
[SRS_BSW_00346]	All AUTOSAR Basic Software Modules shall provide at least a basic set of module files	[SWS_Xcp_00501]
[SRS_BSW_00358]	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	[SWS_Xcp_00803]
[SRS_BSW_00373]	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	[SWS_Xcp_00823]
[SRS_BSW_00380]	Configuration parameters being stored in memory shall be placed into separate c-files	[SWS_Xcp_00501]
[SRS_BSW_00383]	The Basic Software Module specifications shall specify which other configuration files from other modules they use at least in the description	[SWS_Xcp_00501]
[SRS_BSW_00404]	BSW Modules shall support post-build configuration	[SWS_Xcp_00742]
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[SWS_Xcp_00803]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[SWS_Xcp_00803]
[SRS_BSW_00419]	If a pre-compile time configuration parameter is implemented as `const` it should be placed into a separate c-file	[SWS_Xcp_00501]
[SRS_BSW_00424]	BSW module main processing functions shall not be allowed to enter a wait state	[SWS_Xcp_00823]



△

Requirement	Description	Satisfied by
[SRS_BSW_00433]	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	[SWS_Xcp_00823]

Table 6.1: Requirements Tracing

7 Functional specification

The specification of the module XCP shall define all parameters and interfaces, which are required to use the ASAM XCP protocol specification within an AUTOSAR environment.

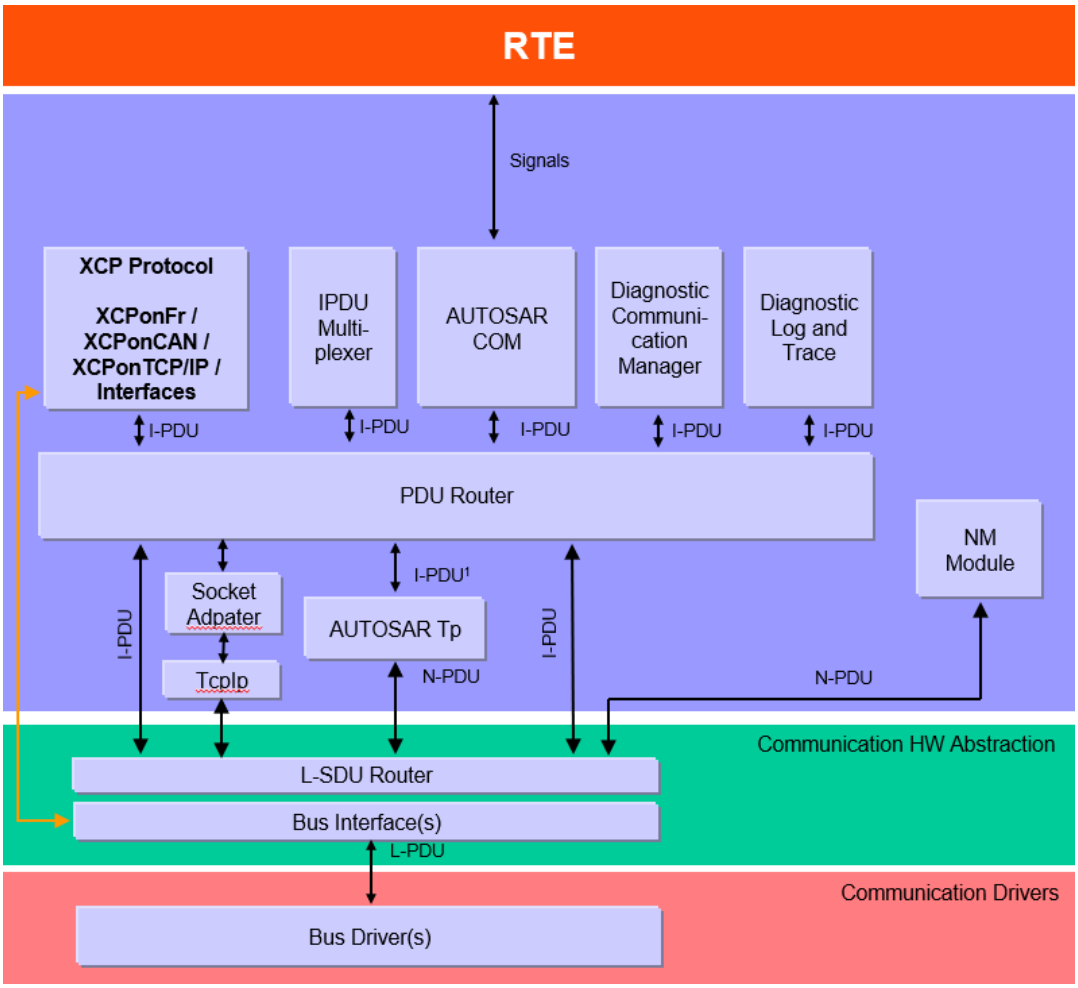


Figure 7.1: Description

Black arrows: Data Path (Signals/Pdus)
Orange arrows: Control Path (FlexRay Interface)

[SWS_Xcp_00860] Location of the XCP module in the AUTOSAR communication stack

Status: DRAFT
Replaces: SWS_Xcp_00701

⌈The AUTOSAR XCP Module and the bus specific Interfaces shall be located above the AUTOSAR PDU Routing module (PduR) [4].⌋

[SWS_Xcp_00702] ⌈For transmitting and receiving of XCP messages, unique PDU-IDs shall be used.⌋

[SWS_Xcp_00703] ⌈The AUTOSAR XCP Module shall support the ASAM XCP Specification Version 1.1, except for XCP on CAN where ASAM XCP Specification Version 1.2 shall be supported.⌋

[SWS_Xcp_00705] ⌈The AUTOSAR XCP Module shall support the basic feature "Synchronous data acquisition (measurement) ". Please refer to [3]⌋

[SWS_Xcp_00706] ⌈The AUTOSAR XCP Module shall support the feature "Dynamic DAQ Configuration". according to [3]⌋

[SWS_Xcp_00707] ⌈The AUTOSAR XCP Module shall support the basic feature "Synchronous data stimulation" according to [3]⌋

[SWS_Xcp_00708] ⌈The AUTOSAR XCP Module shall support the basic feature "Online memory calibration (read / write access) ", according to [3]⌋

[SWS_Xcp_00709] ⌈The AUTOSAR XCP Module shall support the feature "Timestamped Data Transfer", according to [3]⌋

[SWS_Xcp_00768] ⌈The ECU local time shall be derived from the AUTOSAR OS.⌋

[SWS_Xcp_00711] ⌈The AUTOSAR XCP Module shall support the feature "Block communication mode", according to [3]⌋

[SWS_Xcp_00761] ⌈The AUTOSAR XCP Module shall support the feature "Bypassing", according to [3]⌋

[SWS_Xcp_00766] ⌈The AUTOSAR XCP Module shall support the feature "Seed & Key" according to [3]⌋

[SWS_Xcp_00852] ⌈The AUTOSAR XCP Module shall support the feature "Page switching", according to [3]⌋

[SWS_Xcp_00853] ⌈The code generator of the XCP Module shall generate the A2L IF_DATA section, based on the configuration of XCP⌋

[SWS_Xcp_00854] ⌈The AUTOSAR XCP Module shall support the feature "Power-Up data transfer (RESUME MODE)", according to [3]⌋

[SWS_Xcp_00855] ⌈The AUTOSAR XCP Module shall support the flash programming (PGM) according to [3]⌋

[SWS_Xcp_00856] ⌈Indication the end of a programming sequence is supported using the optional command "PROGRAM_RESET", where the slave will go to disconnected state but without forcing a device reset⌋

[SWS_Xcp_00863] Waiting for call of `Xcp_TxConfirmation` to avoid overwritting previously transmitted data

Status: DRAFT
Replaces: SWS_Xcp_00859

⌈The XCP module shall wait for the `Xcp_TxConfirmation` (positive or negative) after each call to `PduR_XcpTransmit` to avoid overwriting previously transmitted data.⌋

7.1 XCP on CAN

[SWS_Xcp_00713] ⌈The AUTOSAR XCP Module shall support the CAN communications bus according to [7]⌋

[SWS_Xcp_00864] Use APIs of PduR for exchanging data via CAN

Status: DRAFT
Replaces: SWS_Xcp_00714

⌈XCP data sent and received via CAN, the PDUs shall be transmitted and received using the transmit and receive APIs provided by the AUTOSAR PDU Routing module (PduR) [4].⌋

Note: AUTOSAR PDU Routing module (PduR) [4] and AUTOSAR BSW Linklayer Sdu Routing Module (LSduR) route the XCP data as PDU to the AUTOSAR BSW CAN Interface: `XCP -> PduR -> LSduR -> CanIf`

Please refer to Chapter 9 for more details.

[SWS_Xcp_00715] ⌈For sending and receiving XCP data via CAN, at least two different CAN identifiers have to be configured to be used by XCP.⌋

[SWS_Xcp_00716] ⌈Performance information shall be exchanged between the XCP master and XCP slave using the parameters according to [7]⌋

[SWS_Xcp_00718] ⌈The XCP Module shall support the GET_SLAVE_ID command according to [7]⌋

7.2 XCP on FlexRay

[SWS_Xcp_00719] ⌈The AUTOSAR XCP Module shall support the FlexRay communications bus according to [8]⌋

[SWS_Xcp_00865] Use APIs of PduR for exchanging data via FlexRay

Status: DRAFT
Replaces: SWS_Xcp_00720

⌈XCP data sent and received via FlexRay, the PDUs shall be transmitted and received using the transmit and receive APIs provided by the AUTOSAR PDU Routing module (PduR) [4].⌋

Note: AUTOSAR PDU Routing module (PduR) [4] and AUTOSAR BSW Linklayer Sdu Routing Module (LSduR) route the XCP data as PDU to the AUTOSAR BSW FlexRay Interface: `XCP -> PduR -> LSduR -> FrIf`

Please refer to Chapter 9 for more details.

[SWS_Xcp_00721] ⌈All XCP on FlexRay LPDUs always are event driven. Please refer to Chapter 1.1.2 "FlexRay Frame Type" of [8]⌋

[SWS_Xcp_00722] ⌈The hardware buffers (of the FlexRay Communication Controller) XCP uses for data transmission and reception are assigned exclusively to the XCP module.⌋

Note:

This restriction prevents disturbances of ongoing FlexRay communication.

[SWS_Xcp_00723] ⌈The usage of FlexRay Communication Controller's hardware buffers shall be configured by the corresponding parameters according to [8]⌋

[SWS_Xcp_00724] ⌈The FlexRay PDU length used by the AUTOSAR XCP module shall be set using the corresponding parameters according to [8]⌋

[SWS_Xcp_00725] ⌈LPDU_IDs which shall be routed to the AUTOSAR XCP module (using the AUTOSAR Bus Interface) have to be defined by the system designer.⌋

[SWS_Xcp_00726] ⌈The ASAM MCD 2MC description file (i.e. A2L file) describes to which extent the XCP-dedicated buffers of a specific slave can be configured for XCP communication.⌋

[SWS_Xcp_00728] ⌈The XCP master gets the information about the XCP dedicated FlexRay Communication Controller buffers from the ASAM MCD 2MC description file.⌋

[SWS_Xcp_00729] ⌈Limitations due to the usage of multiple XCP slaves on the FlexRay communications bus shall be taken into consideration by the system designer. Please refer to [8].⌋

[SWS_Xcp_00730] ⌈Depending upon the requirements on sequencing correctness, alignment and net data throughput, different header types are possible. Please refer to Chapter 1.4.1 "Header" of [8]⌋

[SWS_Xcp_00731] ⌈For XCP on FlexRay, the Tail consists of a Control Field containing optional FILL bytes according to [8].⌋

[SWS_Xcp_00732] ⌈The AUTOSAR XCP module shall be able to pack multiple XCP messages into one FlexRay Frame according to [8].⌋

7.3 XCP on Ethernet

[SWS_Xcp_00733] ⌈The AUTOSAR XCP Module shall support the Ethernet communications bus according to [9]⌋

[SWS_Xcp_00866] Use APIs of PduR for exchanging data via Ethernet (i.e. TcpIp)
Status: DRAFT
Replaces: SWS_Xcp_00734

⌈XCP data sent and received via Ethernet, the PDUs have to be transmitted and received using the transmitting and receive APIs provided by the AUTOSAR PDU Router according to [4].⌋

Note: AUTOSAR PDU Routing module (PduR) [4] and AUTOSAR BSW Linklayer Sdu Routing Module (LSduR) route the XCP data as PDU to the AUTOSAR BSW Ethernet Interface: `XCP -> PduR -> SoAd -> TcpIp -> LSduR -> EthIf`

Please refer to Chapter 9 for more details.

[SWS_Xcp_00735] ⌈The AUTOSAR XCP slave connected by Ethernet and TCP/IP or UDP/IP is addressed by its IP Address and Port number.⌋

[SWS_Xcp_00736] ⌈The AUTOSAR XCP slave only accepts one connection at the time.⌋

[SWS_Xcp_00737] ⌈If the socket is closed while in XCP connected state, the slave device will perform an XCP disconnect, which means that all data acquisition will be stopped.⌋

[SWS_Xcp_00738] ⌈The addressing scheme is defined according to [9]⌋

.

[SWS_Xcp_00739] ⌈The header and tail of an XCP on Ethernet message have to be set according to [9]⌋

[SWS_Xcp_00740] ⌈The upper performance limit depends on the protocol stack of the host system. The corresponding parameters defined according to [9] have to be set.⌋

[SWS_Xcp_00710] ⌈The AUTOSAR XCP Module shall support the feature "Interleaved communication mode", according to according to [3]⌋

7.4 General Requirements

[SWS_Xcp_00741]

Upstream requirements: SRS_BSW_00344

⌈Link-time and post-build-time configuration data shall be implemented as read-only data structures. Link-time configuration data shall be immediately referenced by the implementation, the start-address of post-build-time configuration data shall be passed during module initialization⌋

[SWS_Xcp_00742]

Upstream requirements: SRS_BSW_00404, SRS_BSW_00345

⌈The XCP module shall support pre-compile time, link-time and post-build-time configuration.⌋

7.5 Message copying

The XCP module supports message copying by means of hardware-assisted data transfer (e.g. DMA). This behavior is controlled per PDU by the ECU Configuration parameter `KeepLocalPduBuffer`.

See also [2, BSW General] Chapter 7.1.21 *"Asynchronous copy approach (DRAFT)"* for more details.

If `KeepLocalPduBuffer` is set to TRUE for `XcpTxPdu`, the XCP module shall prepare its local buffer to enable asynchronous copying by the lower layers.

If `KeepLocalPduBuffer` is set to TRUE for `XcpRxPdu`, the SduData shall remain unchanged until the XCP module indicates that it may be released; this enables asynchronous copying (e.g. via DMA) into the local XCP buffer when message processing is deferred to `Xcp_MainFunction`. If `KeepLocalPduBuffer` is set to FALSE, SduData shall not be available after `Xcp_RxIndication` returns; therefore, a synchronous copy into the local buffer is required.

[SWS_Xcp_00872] Tx buffer alignment when `KeepLocalPduBuffer` is set to TRUE

Status: DRAFT

⌈If `KeepLocalPduBuffer` of the `XcpTxPdu` is set to TRUE, the XCP module shall align the local buffer in accordance with the ECU configuration parameter `PduBufferAlignment`.⌋

[SWS_Xcp_00873] Tx buffer data integrity when `KeepLocalPduBuffer` is set to TRUE

Status: DRAFT

⌈If `KeepLocalPduBuffer` of the `XcpTxPdu` is set to TRUE, the XCP module shall maintain the content of the local buffer unchanged until `Xcp_TxConfirmation` is invoked.⌋

[SWS_Xcp_00874] Asynchronous copying of rx buffer

Status: DRAFT

⌈If `KeepLocalPduBuffer` of the `XcpRxPdu` is set to TRUE, its SduLength is greater than `EcucDmaThreshold`, and message processing is deferred to `Xcp_MainFunction`, the XCP module shall copy the I-PDU into its local buffer asynchronously and shall return from `Xcp_RxIndication` without waiting for the copy to finish.⌋

[SWS_Xcp_00875] Synchronous copying rx buffer

Status: DRAFT

⌈If `KeepLocalPduBuffer` of the `XcpRxPdu` is set to TRUE, its SduLength is not greater than `EcucDmaThreshold` and the message processing is deferred to `Xcp_MainFunction`, the XCP module shall copy the I-PDU into its local buffer synchronously within `Xcp_RxIndication`.⌋

[SWS_Xcp_00876] Rx buffer release when `KeepLocalPduBuffer` is set to TRUE

Status: DRAFT

⌈If `KeepLocalPduBuffer` of the `XcpRxPdu` is set to TRUE, the XCP module shall call `PduR_XcpReleaseRxBuffer` after the message is processed within `Xcp_RxIndication` (synchronous handling) or after the asynchronous copy process (e.g. via DMA) has been finalized (either successful or unsuccessful) to indicate that the buffer may be released.⌋

[SWS_Xcp_00877] Asynchronous copy failure

Status: DRAFT

⌈If the asynchronous copy of received I-PDU into the local buffer fails, the XCP module shall report the runtime error `XCP_E_RECEPTION_FAILED` to the DET module.⌋

7.6 Error Classification

[SWS_Xcp_00763]

Upstream requirements: SRS_BSW_00327

⌈The error values and EventIds are named in capital letters according to the scheme XCP_E_<NAME>, where NAME describes the error/EventId and may consist of several words separated by underscores.⌋

7.6.1 Development Errors

[SWS_Xcp_00857] Definition of development errors in module Xcp ⌈

Type of error	*Related error code*	*Error value*
Module not initialized	XCP_E_UNINIT	0x02
API call with wrong PDU ID	CP_E_INVALID_PDUID	0x03
Initialization of XCP failed	XCP_E_INIT_FAILED	0x04
Null pointer has been passed as an argument	XCP_E_PARAM_POINTER	0x12

⌋

7.6.2 Runtime Errors

[SWS_Xcp_91002] Definition of runtime errors in module Xcp ⌈

Type of error	*Related error code*	*Error value*
Reception of a PDU has failed.	XCP_E_RECEPTION_FAILED	0x20

⌋

7.6.3 Transient Faults

There are no transient faults.

7.6.4 Production Errors

There are no production errors.

7.6.5 Extended Production Errors

There are no extended production errors.

7.7 Security Events

The module does not report security events.

8 API specification

8.1 Imported types

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

[SWS_Xcp_00801] Definition of imported datatypes of module Xcp ⌈

Module	*Header File*	*Imported Type*
Comtype	ComStack_Types.h	NetworkHandleType
	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
Fr	Fr_GeneralTypes.h	Fr_ChannelType
NvM	Rte_NvM_Type.h	NvM_BlockIdType
Os	Os.h	CounterType
	Os.h	StatusType
	Os.h	TickRefType
	Os.h	TickType
Std	Std_Types.h	Std_ReturnType

⌋

8.2 Type definitions

8.2.1 Xcp_ConfigType

[SWS_Xcp_00845] Definition of datatype Xcp_ConfigType ⌈

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

⌋

8.2.2 Xcp_Transmission Mode Type

[SWS_Xcp_00846] Definition of datatype Xcp_TransmissionModeType ⌈

Name	Xcp_TransmissionModeType		
Kind	Enumeration		
Range	XCP_TX_OFF	0x00	Transmission Disabled
	XCP_TX_ON	0x01	Transmission Enabled
Description	Handles the enabling and disabling of the transmission mode		
Available via	Xcp.h		

⌋

8.3 Function definitions

This chapter contains all APIs provided by the module.

8.3.1 Xcp_Init

[SWS_Xcp_00803] Definition of API function Xcp_Init

Upstream requirements: SRS_BSW_00405, SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414

⌈

Service Name	Xcp_Init	

Syntax	`void Xcp_Init (`
	`  const Xcp_ConfigType* Xcp_ConfigPtr`
	`)`	

Service ID [hex]	0x00	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Xcp_ConfigPtr	Pointer to a selected configuration structure
Parameters (inout)	None	
Parameters (out)	None	
Return value	void	–
Description	This service initializes interfaces and variables of the AUTOSAR XCP layer.	
Available via	Xcp.h	

⌋

[SWS_Xcp_00802] ⌈The function `Xcp_Init` shall internally store the configuration address to enable subsequent API calls to access the configuration⌋

8.3.2 Xcp_SetTransmissionMode

[SWS_Xcp_00844] Definition of callback function Xcp_SetTransmissionMode ⌈

Service Name	Xcp_SetTransmissionMode	
Syntax	`void Xcp_SetTransmissionMode ( NetworkHandleType Channel, Xcp_TransmissionModeType Mode )`	
Service ID [hex]	0x05	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Channel	The Network channel for the used bus communication
	Mode	Enabled or disabled Transmission mode Parameters
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API is used to turn on and off of the TX capabilities of used communication bus channel in XCP module.	
Available via	Xcp.h	

⌋

[SWS_Xcp_00848] ⌈The XCP module shall provide this service only if XCP_SUPPRESS_TX_SUPPORT (see [ECUC_Xcp_00169]) equals TRUE.⌋

[SWS_Xcp_00849] ⌈If `Xcp_SetTransmissionMode`(Channel, Mode) is called and parameter Mode equals XCP_TX_OFF, all TxPDUs which are assigned to Channel shall not be transmitted.⌋

Note: It could be derived from <Bus>If configuration and the global PDU parameter, to which specific communication channel the PDU is assigned to.

[SWS_Xcp_00850] ⌈If `Xcp_SetTransmissionMode`(Channel, Mode) is called and parameter Mode equals XCP_TX_ON, all TxPDUs which are assigned to Channel shall be able to be transmitted.⌋

8.3.3 Xcp_RxIndication

[SWS_Xcp_00813] Definition of callback function Xcp_RxIndication ⌈

Service Name	Xcp_RxIndication
Syntax	`void Xcp_RxIndication ( PduIdType RxPduId, const PduInfoType* PduInfoPtr )`
Service ID [hex]	0x42
Sync/Async	Synchronous
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.

▽

△

Parameters (in)	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication of a received PDU from a lower layer communication interface module.	
Available via	Xcp.h	

⌋

The callback function `Xcp_RxIndication` is called by the Bus Interfaces, Ethernet Socket Adaptor or CDD and is implemented by the Xcp module.

[SWS_Xcp_00867] Development error handling in context of `Xcp_RxIndication`

Status: DRAFT
Replaces: SWS_Xcp_00847

⌈The callback function `Xcp_RxIndication` shall inform the DET, if development error detection is enabled (XCP_DEV_ERROR_DETECT is set to TRUE) and if function call has failed because of the following reasons:

- Xcp was not initialized (XCP_E_UNINIT)
- PduInfoPtr equals NULL_PTR (XCP_E_PARAM_POINTER)
- Invalid PDUID (XCP_E_INVALID_PDUID)

⌋

The function `Xcp_RxIndication` shall be called by the Xcp module's environment in an interrupt context.

8.3.4 Xcp_TxConfirmation

[SWS_Xcp_00814] Definition of callback function Xcp_TxConfirmation ⌈

Service Name	Xcp_TxConfirmation	

Syntax	`void Xcp_TxConfirmation (`
	`  PduIdType TxPduId,`
	`  Std_ReturnType result`
	`)`	

Service ID [hex]	0x40	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the PDU that has been transmitted.

▽

△

	result	E_OK: The PDU was transmitted successfully. E_NOT_OK: Transmission of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.	
Available via	Xcp.h	

▽

Note:

The callback function `Xcp_TxConfirmation` is called by the Bus Interfaces, Ethernet Socket Adaptor or CDD and is implemented by the Xcp module.

[SWS_Xcp_00868] Development error handling in context of `Xcp_TxConfirmation`

Status: DRAFT
Replaces: SWS_Xcp_00840

⌈If development error detection for the XCP module is enabled: if the function `Xcp_TxConfirmation` is called before the XCP was initialized successfully, the function `Xcp_TxConfirmation` shall raise the development error XCP_E_UNINIT and return.⌋

[SWS_Xcp_00869] Caveats on call of `Xcp_TxConfirmation`

Status: DRAFT
Replaces: SWS_Xcp_00841

⌈Caveats of `Xcp_TxConfirmation`:

- The call context is either on interrupt level (interrupt mode) or on task level
- The Xcp module is initialized correctly.

⌋

8.3.5 Xcp_TriggerTransmit

[SWS_Xcp_00835] Definition of callback function Xcp_TriggerTransmit ⌈

Service Name	Xcp_TriggerTransmit

Syntax	`Std_ReturnType Xcp_TriggerTransmit (`
	`  PduIdType TxPduId,`
	`  PduInfoType* PduInfoPtr`
	`)`

Service ID [hex]	0x41
Sync/Async	Synchronous
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.

▽

△

Parameters (in)	TxPduId	ID of the SDU that is requested to be transmitted.
Parameters (inout)	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: SDU has been copied and SduLength indicates the number of copied bytes. `E_NOT_OK`: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	
Available via	Xcp.h	

⌋

Note:

The callback function `Xcp_TriggerTransmit` is called by the Bus Interfaces, Ethernet Socket Adaptor or CDD and is implemented by the Xcp module.

[SWS_Xcp_00870] Development error handling in context of `Xcp_TriggerTransmit`
Status: DRAFT
Replaces: SWS_Xcp_00842

⌈If development error detection for the XCP module is enabled: if the function

`Xcp_TriggerTransmit` is called before the XCP was initialized successfully, the function `Xcp_TriggerTransmit` shall raise the development error XCP_E_UNINIT and return E_NOT_OK.⌋

[SWS_Xcp_00871] Caveats on call of `Xcp_TriggerTransmit`
Status: DRAFT
Replaces: SWS_Xcp_00843

⌈Caveats of `Xcp_TriggerTransmit`:

- The call context is either on interrupt level (interrupt mode) or on task level
- The Xcp module is initialized correctly.

⌋

8.4 Scheduled functions

The functions are called directly by Basic Software Scheduler. The following functions shall have no return value and no parameter. All functions shall be non reentrant.

8.4.1 Xcp_MainFunction

[SWS_Xcp_00823] Definition of scheduled function Xcp_MainFunction

Upstream requirements: SRS_BSW_00424, SRS_BSW_00433, SRS_BSW_00373

⌈

Service Name	Xcp_MainFunction
Syntax	`void Xcp_MainFunction ( void )`
Service ID [hex]	0x04
Description	Scheduled function of the XCP module
Available via	SchM_Xcp.h

⌋

[SWS_Xcp_00824] ⌈The XCP Main Function shall be called cyclically.⌋

8.5 Expected interfaces

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

8.5.1 Mandatory interfaces

[SWS_Xcp_91001] Definition of mandatory interfaces required by module Xcp ⌈

API Function	***Header File***	***Description***
There are no mandatory interfaces.		

⌋

8.5.2 Optional interfaces

[SWS_Xcp_00832] Definition of optional interfaces requested by module Xcp ⌈

API Function	***Header File***	***Description***
Det_ReportError	Det.h	Service to report development errors.
FrIf_DisableLPdu	FrIf.h	Wraps the FlexRay Driver Function Fr_DisableLPdu. It disables the hardware resource of an LPdu for transmission/reception.
FrIf_ReconfigLPdu	FrIf.h	Calls the FlexRay Driver's API Fr_ReconfigLPdu. The enum value "FR_CHANNEL_AB" shall not be used.

▽

△

API Function	*Header File*	*Description*
GetCounterValue	Os.h	This service reads the current count value of a counter (returning either the hardware timer ticks if counter is driven by hardware or the software ticks when user drives counter).
GetElapsedValue	Os.h	This service gets the number of ticks between the current tick value and a previously read tick value.
NvM_ReadBlock	NvM.h	Service to copy the data of the NV block to its corresponding RAM block.
NvM_WriteBlock	NvM.h	Service to copy the data of the RAM block to its corresponding NV block.
PduR_XcpReleaseRxBuffer (draft)	PduR_Xcp.h	Indication from the upper layer to release the lower layer reception buffer.
PduR_XcpTransmit	PduR_Xcp.h	Requests transmission of a PDU.

⌋

8.5.3 Callout interfaces

No callout interfaces.

8.5.4 Configurable interfaces

In this chapter, all interfaces are listed where the target function could be configured. The target function is usually a call-back function. The names of these kind of interfaces is not fixed because they are configurable.

The XCP module offers configurable interfaces to be used by Complex Driver(s).

9 Sequence diagrams

9.1 XCP on FlexRay

9.1.1 Xcp on FlexRay Transmit

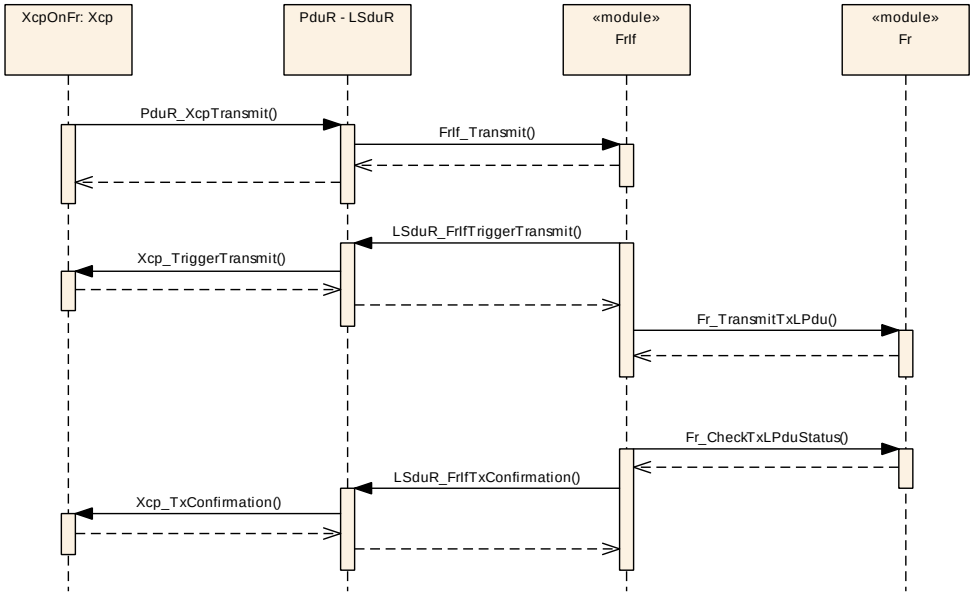


Figure 9.1: Xcp On FlexRay Transmit

9.1.2 Xcp on FlexRay Receive Indication

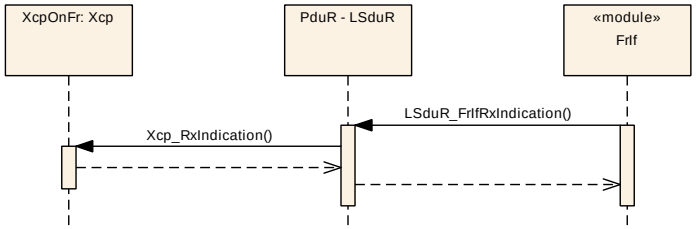


Figure 9.2: Xcp on FlexRay Receive Indication

9.2 XCP on CAN

9.2.1 Xcp on CAN Transmit

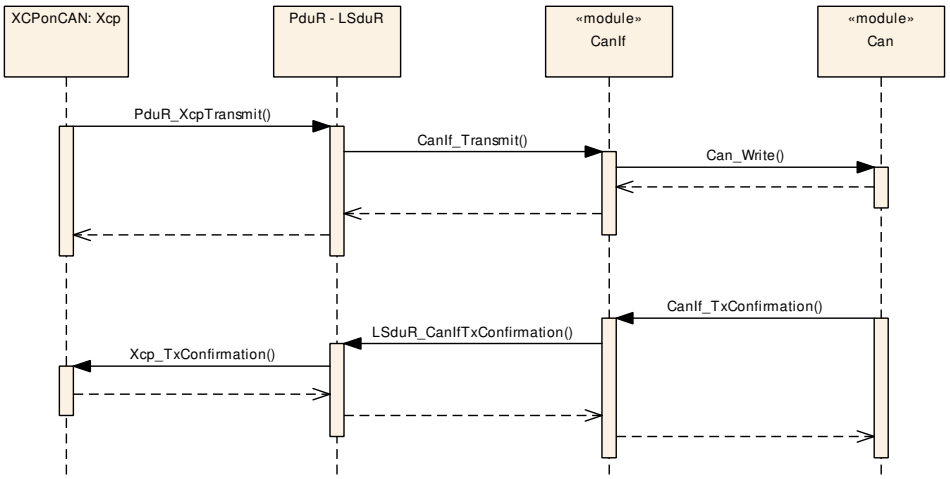


Figure 9.3: Xcp on Can Transmit

9.2.2 Xcp on CAN Receive Indication

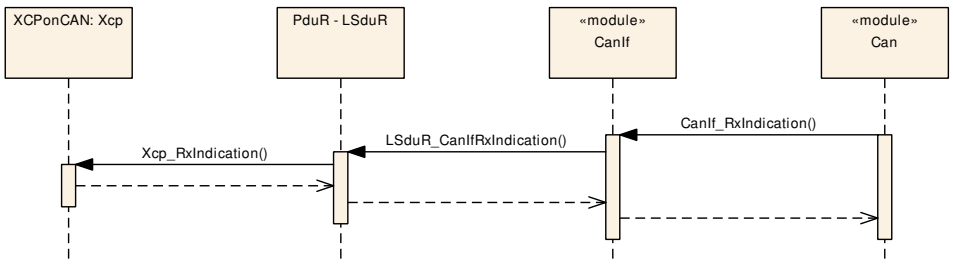


Figure 9.4: Xcp on CAN Receive Indication

9.3 XCP on Ethernet

9.3.1 Xcp on Ethernet Transmit

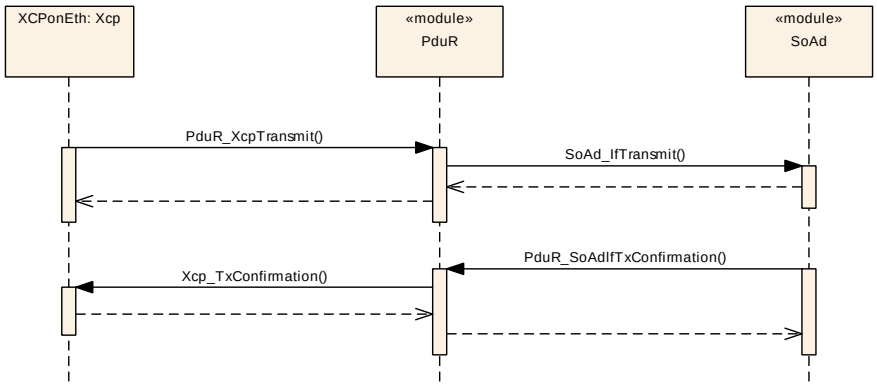


Figure 9.5: Xcp on Ethernet Transmit

9.3.2 Xcp on Ethernet Receive Indication

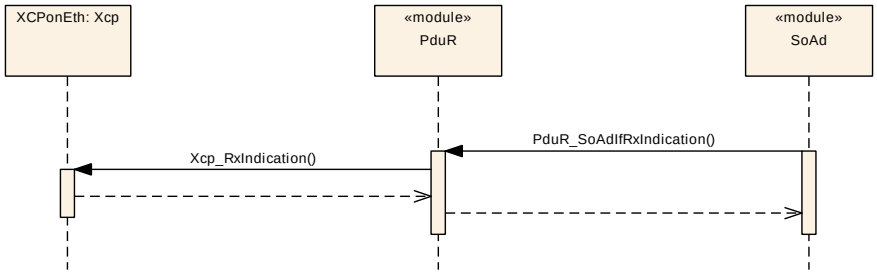


Figure 9.6: Xcp on Ethernet Receive Indication

10 Configuration specification

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

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

Chapter 10.3 specifies published information of the module XCP.

10.1 How to read this chapter

For details refer to [2] Chapter 10.1 *"Introduction to configuration specification"*.

10.2 Containers and configuration parameters

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

[SWS_Xcp_00102]

Upstream requirements: SRS_BSW_00159

⌈The listed configuration items can be derived from a network description database, which is based on the EcuConfigurationTemplate. The configuration tool shall extract all information to configure the XCP.⌋

[SWS_Xcp_00103]

Upstream requirements: SRS_BSW_00167

⌈The configuration tool must check the consistency of the configuration at configuration time.⌋

[SWS_Xcp_00104]

Upstream requirements: SRS_BSW_00167

⌈Configuration rules and constraints for plausibility checks shall be performed during configuration time, wherever possible.⌋

[SWS_Xcp_00105]

Upstream requirements: SRS_BSW_00167

⌈These dependencies between FlexRay Interface and FlexRay Driver configuration must be provided at configuration time by the configuration tools.⌋

10.2.1 Xcp

[ECUC_Xcp_00182] Definition of EcucModuleDef Xcp ⌈

Module Name	Xcp
Description	Configuration of the XCP module
Post-Build Selectable Support	true
Post-Build Loadable Support	true

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR Xcp module.
XcpGeneral	1	This container contains the general configuration parameters of the XCP.

⌋

10.2.2 XcpGeneral

[ECUC_Xcp_00001] Definition of EcucParamConfContainerDef XcpGeneral ⌈

Container Name	XcpGeneral
Parent Container	Xcp
Upstream mapping defined	–
Description	This container contains the general configuration parameters of the XCP.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpDaqConfigType	1	[ECUC_Xcp_00164]
XcpDaqCount	1	[ECUC_Xcp_00012]
XcpDevErrorDetect	1	[ECUC_Xcp_00003]
XcpFlashProgrammingEnabled	1	[ECUC_Xcp_00181]
XcpIdentificationFieldType	1	[ECUC_Xcp_00170]
XcpMainFunctionPeriod	1	[ECUC_Xcp_00014]
XcpMaxCto	1	[ECUC_Xcp_00004]
XcpMaxDto	1	[ECUC_Xcp_00005]
XcpMaxEventChannel	1	[ECUC_Xcp_00011]
XcpMinDaq	1	[ECUC_Xcp_00013]
XcpOdtCount	1	[ECUC_Xcp_00054]
XcpOdtEntriesCount	1	[ECUC_Xcp_00059]
XcpOdtEntrySizeDaq	1	[ECUC_Xcp_00177]

▽

△

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpOdtEntrySizeStim	1	[ECUC_Xcp_00178]
XcpOnCanEnabled	1	[ECUC_Xcp_00006]
XcpOnCddEnabled	1	[ECUC_Xcp_00009]
XcpOnEthernetEnabled	1	[ECUC_Xcp_00008]
XcpOnFlexRayEnabled	1	[ECUC_Xcp_00007]
XcpPrescalerSupported	1	[ECUC_Xcp_00169]
XcpSuppressTxSupport	1	[ECUC_Xcp_00176]
XcpTimestampTicks	1	[ECUC_Xcp_00167]
XcpTimestampType	1	[ECUC_Xcp_00166]
XcpTimestampUnit	1	[ECUC_Xcp_00168]
XcpCounterRef	1	[ECUC_Xcp_00162]
XcpNvRamBlockIdRef	0..1	[ECUC_Xcp_00180]

No Included Containers

⌋

[ECUC_Xcp_00164] Definition of EcucEnumerationParamDef XcpDaqConfigType ⌈

Parameter Name	XcpDaqConfigType	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Sets the DAQ_CONFIG_TYPE bit within the DAQ_PROPERTIES parameter to "static" or to "dynamic". If DAQ_STATIC is selected, the DAQ_CONFIG_TYPE bit is set to "0". If DAQ_DYNAMIC is selected, the DAQ_CONFIG_TYPE bit is set to "1". Dependency: If DAQ_CONFIG_TYPE = dynamic, MAX_DAQ equals MIN_DAQ+DAQ_COUNT.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DAQ_DYNAMIC	If DAQ_DYNAMIC is selected, the DAQ_CONFIG_TYPE bit is set to '1'
	DAQ_STATIC	If DAQ_STATIC is selected, the DAQ_CONFIG_TYPE bit is set to '0'
Pre-Compile only	true	

⌋

[ECUC_Xcp_00012] Definition of EcucIntegerParamDef XcpDaqCount ⌈

Parameter Name	XcpDaqCount
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Indicates the number of DAQ lists for dynamic configuration. Dependency: This parameter is available only if XcpDaqConfigType is set to "1" i.e DAQ_DYNAMIC
Multiplicity	1

▽

△

Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00003] Definition of EcucBooleanParamDef XcpDevErrorDetect ⌈

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

⌋

[ECUC_Xcp_00181] Definition of EcucBooleanParamDef XcpFlashProgramming Enabled ⌈

Parameter Name	XcpFlashProgrammingEnabled
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Enabling of XCP Flash programming functionality
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00170] Definition of EcucEnumerationParamDef XcpIdentification FieldType ⌈

Parameter Name	XcpIdentificationFieldType	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Type of Identification Field the slave will use when transferring DAQ Packets to the master. The master has to use the same Type of Identification Field when transferring STIM Packets to the slave.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	ABSOLUTE	Absolute ODT number

▽

△

	RELATIVE_BYTE	Relative ODT number, absolute DAQ list number (BYTE)
	RELATIVE_WORD	Relative ODT number, absolute DAQ list number (WORD)
	RELATIVE_WORD_ALIGNED	Relative ODT number, absolute DAQ list number (WORD, aligned).
Pre-Compile only	true	

⌋

[ECUC_Xcp_00014] Definition of EcucFloatParamDef XcpMainFunctionPeriod ⌈

Parameter Name	XcpMainFunctionPeriod	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	The XCP does not require this information but the BSW scheduler, which invokes the main function, needs it in order to plan its tasks.	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	]0 .. INF[	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00004] Definition of EcucIntegerParamDef XcpMaxCto ⌈

Parameter Name	XcpMaxCto	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	MAX_CTO shows the maximum length of a CTO packet in bytes.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	8 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00005] Definition of EcucIntegerParamDef XcpMaxDto ⌈

Parameter Name	XcpMaxDto	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	MAX_DTO shows the maximum length of a DTO packet in bytes.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	8 .. 65535	
Default value	–	

▽

△

Pre-Compile only	true

⌋

[ECUC_Xcp_00011] Definition of EcucIntegerParamDef XcpMaxEventChannel ⌈

Parameter Name	XcpMaxEventChannel	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Maximum number of event channels (i.e. total number of available event channels)	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00013] Definition of EcucIntegerParamDef XcpMinDaq ⌈

Parameter Name	XcpMinDaq	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Indicates the number of predefined, read only DAQ lists on the XCP slave.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00054] Definition of EcucIntegerParamDef XcpOdtCount ⌈

Parameter Name	XcpOdtCount	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	This parameter indicates the amount of ODTs of a DAQ list using dynamic DAQ list configuration. Dependency: This parameter is available only if XcpDaqConfigType is set to "1" i.e DAQ_DYNAMIC	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 252	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00059] Definition of EcucIntegerParamDef XcpOdtEntriesCount ⌈

Parameter Name	XcpOdtEntriesCount	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Indicates the amount of entries into an ODT using dynamic DAQ list configuration. Dependency: This parameter is available only if XcpDaqConfigType is set to "1" i.e DAQ_DYNAMIC	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00177] Definition of EcucIntegerParamDef XcpOdtEntrySizeDaq ⌈

Parameter Name	XcpOdtEntrySizeDaq	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Indicates the size of an element described by an ODT entry to the DaqListType for a DAQ.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00178] Definition of EcucIntegerParamDef XcpOdtEntrySizeStim ⌈

Parameter Name	XcpOdtEntrySizeStim	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	Indicates the size of an element described by an ODT entry to the DaqListType for a stim.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00006] Definition of EcucBooleanParamDef XcpOnCanEnabled ⌈

Parameter Name	XcpOnCanEnabled
Parent Container	XcpGeneral
Upstream mapping defined	–

▽

△

Description	Enabling of XCPonCAN functionality
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00009] Definition of EcucBooleanParamDef XcpOnCddEnabled ⌈

Parameter Name	XcpOnCddEnabled
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Enabling of XCPonCdd functionality
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00008] Definition of EcucBooleanParamDef XcpOnEthernetEnabled ⌈

Parameter Name	XcpOnEthernetEnabled
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Enabling of XCPonEthernet functionality
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00007] Definition of EcucBooleanParamDef XcpOnFlexRayEnabled ⌈

Parameter Name	XcpOnFlexRayEnabled
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Enabling of XCPonFlexRay functionality
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00169] Definition of EcucBooleanParamDef XcpPrescalerSupported ⌈

Parameter Name	XcpPrescalerSupported
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	This parameter enables and disables the support for Prescaler support. True is Enabled, False is disabled
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00176] Definition of EcucBooleanParamDef XcpSuppressTxSupport ⌈

Parameter Name	XcpSuppressTxSupport
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	Switches the support of suppressing transmission of PDUs per communication channel on or off. TRUE: Suppressing of TxPDUs supported FALSE: Suppressing of TxPDUs not supported
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00167] Definition of EcucIntegerParamDef XcpTimestampTicks ⌈

Parameter Name	XcpTimestampTicks	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	This parameter defines the timestamp that will increment based TIMESTAMP_TICKS per unit and wrap around if an overflow occurs.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00166] Definition of EcucEnumerationParamDef XcpTimestampType ⌈

Parameter Name	XcpTimestampType	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	This parameter indicates the number of bytes used for the timestamp field. In case No_TIME_STAMP is selected the timestamp field is not available.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	FOUR_BYTE	timestamp field has the size of four byte.
	NO_TIME_STAMP	timestamp field is not available.
	ONE_BYTE	timestamp field has the size of one byte.
	TWO_BYTE	timestamp field has the size of two byte.
Pre-Compile only	true	

⌋

[ECUC_Xcp_00168] Definition of EcucEnumerationParamDef XcpTimestampUnit ⌈

Parameter Name	XcpTimestampUnit	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	This parameter indicates the resolution of the data acquisition clock of the slave when transferring data to master.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	TIMESTAMP_UNIT_100MS	Unit is 100 millisecond.
	TIMESTAMP_UNIT_100NS	Unit is 100 nanosecond.
	TIMESTAMP_UNIT_100PS	Unit is 100 picosecond.
	TIMESTAMP_UNIT_100US	Unit is 100 microsecond.
	TIMESTAMP_UNIT_10MS	Unit is 10 millisecond.
	TIMESTAMP_UNIT_10NS	Unit is 10 nanosecond.
	TIMESTAMP_UNIT_10PS	Unit is 10 picosecond.
	TIMESTAMP_UNIT_10US	Unit is 10 microsecond.
	TIMESTAMP_UNIT_1MS	Unit is 1 millisecond.
	TIMESTAMP_UNIT_1NS	Unit is 1 nonasecond.
	TIMESTAMP_UNIT_1PS	Unit is 1 picosecond.
	TIMESTAMP_UNIT_1S	Unit is 1 second.
	TIMESTAMP_UNIT_1US	Unit is 1 microsecond.
Pre-Compile only	true	

⌋

[ECUC_Xcp_00162] Definition of EcucReferenceDef XcpCounterRef ⌈

Parameter Name	XcpCounterRef
Parent Container	XcpGeneral
Upstream mapping defined	–
Description	This parameter contains a reference to the counter, which is used by XCP.
Multiplicity	1
Type	Reference to OsCounter
Pre-Compile only	true

⌋

[ECUC_Xcp_00180] Definition of EcucReferenceDef XcpNvRamBlockIdRef ⌈

Parameter Name	XcpNvRamBlockIdRef	
Parent Container	XcpGeneral	
Upstream mapping defined	–	
Description	This reference contains the link to a non-volatile memory block to be used in the feature "RESUME MODE" so this information has to be stored non volatile to be available directly after start-up of the ECU.	
Multiplicity	0..1	
Type	Symbolic name reference to NvMBlockDescriptor	
Post-Build Multiplicity	Loadable	Selectable
Post-Build Value	Loadable	Selectable

⌋

10.2.3 XcpConfig

[ECUC_Xcp_00020] Definition of EcucParamConfContainerDef XcpConfig ⌈

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

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpCommunicationChannel	0..*	This container represents the configuration of the communication channel of XCP.
XcpDaqList	1..*	This container contains the configuration of the DAQs.
XcpEventChannel	1..*	This container contains the configuration of event channels on the XCP slave.

▽

△

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpPageSwitching	0..1	This container represents configuration of the page switching feature.
XcpPdu	1..*	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.

⌋

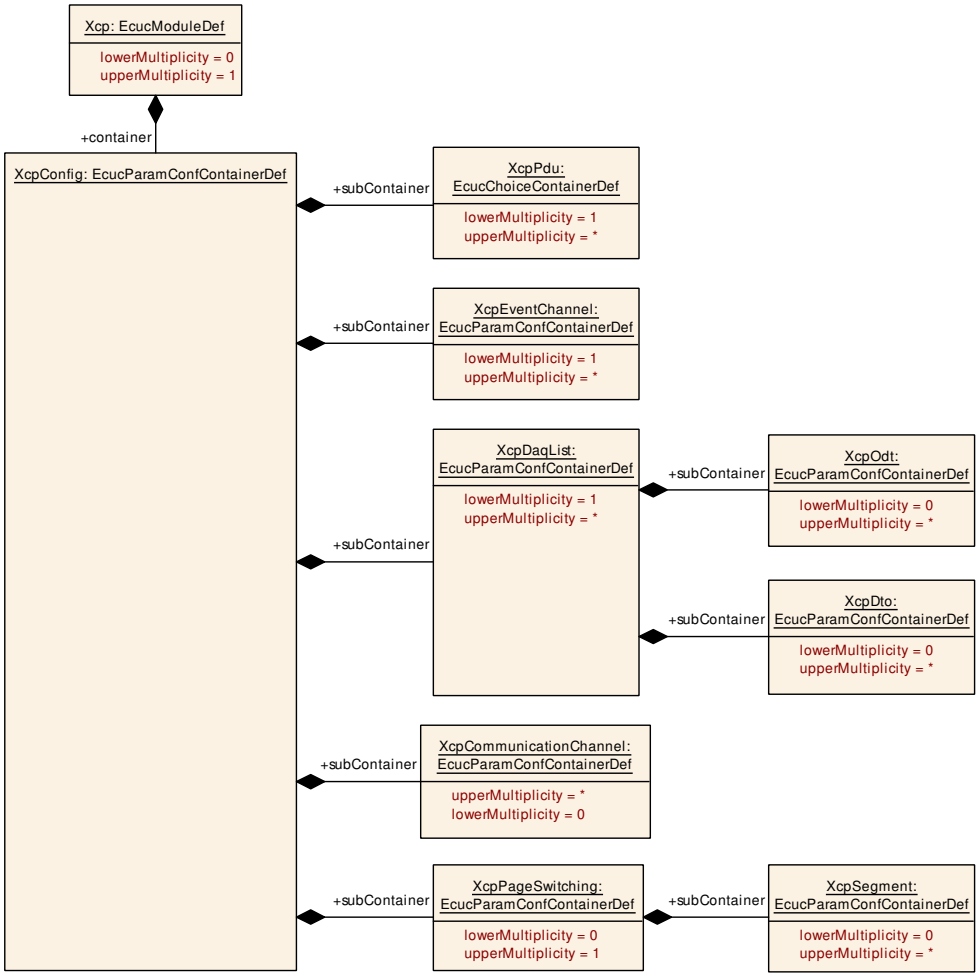


Figure 10.1: Diagram XcpConfig

10.2.4 XcpDaqList

[ECUC_Xcp_00050] Definition of EcucParamConfContainerDef XcpDaqList ⌈

Container Name	XcpDaqList
Parent Container	XcpConfig
Upstream mapping defined	–
Description	This container contains the configuration of the DAQs.
Multiplicity	1..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpDaqListNumber	1	[ECUC_Xcp_00051]
XcpDaqListType	1	[ECUC_Xcp_00052]
XcpMaxOdt	1	[ECUC_Xcp_00053]
XcpMaxOdtEntries	1	[ECUC_Xcp_00058]

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpDto	0..*	This container collects data transfer object specific parameters for the DAQ list.
XcpOdt	0..*	This container contains ODT-specific parameter for the DAQ list.

⌋

[ECUC_Xcp_00051] Definition of EcucIntegerParamDef XcpDaqListNumber ⌈

Parameter Name	XcpDaqListNumber	
Parent Container	XcpDaqList	
Upstream mapping defined	–	
Description	Index number of the DAQ list	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65534	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00052] Definition of EcucEnumerationParamDef XcpDaqListType ⌈

Parameter Name	XcpDaqListType	
Parent Container	XcpDaqList	
Upstream mapping defined	–	
Description	This indicates whether this DAQ list represents a DAQ or a STIM.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DAQ	This DAQ list is a DAQ.
	DAQ_STIM	This DAQ list can be DAQ or STIM.
	STIM	This DAQ list is a STIM.

▽

△

Pre-Compile only	true

⌋

[ECUC_Xcp_00053] Definition of EcucIntegerParamDef XcpMaxOdt ⌈

Parameter Name	XcpMaxOdt
Parent Container	XcpDaqList
Upstream mapping defined	–
Description	MAX_ODT indicates the maximum amount of ODTs in this DAQ list (STATIC configuration) Dependency: only available if XcpDaqConfigType is "DAQ_STATIC" (bit set to '0')
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 252
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00058] Definition of EcucIntegerParamDef XcpMaxOdtEntries ⌈

Parameter Name	XcpMaxOdtEntries
Parent Container	XcpDaqList
Upstream mapping defined	–
Description	This parameter indicates the maximum amount of entries in an ODT of this DAQ list (STATIC configuration). Dependency: only available if XcpDaqConfigType is "DAQ_STATIC" (bit set to '0')
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 255
Default value	–
Pre-Compile only	true

⌋

10.2.5 XcpDto

[ECUC_Xcp_00065] Definition of EcucParamConfContainerDef XcpDto ⌈

Container Name	XcpDto
Parent Container	XcpDaqList
Upstream mapping defined	–
Description	This container collects data transfer object specific parameters for the DAQ list.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpDtoPid	1	[ECUC_Xcp_00066]
XcpDto2PduMapping	1	[ECUC_Xcp_00067]

No Included Containers

⌋

[ECUC_Xcp_00066] Definition of EcucIntegerParamDef XcpDtoPid ⌈

Parameter Name	XcpDtoPid	
Parent Container	XcpDto	
Upstream mapping defined	–	
Description	Packet identifier (PID) of the DTO that identifies the ODT the content of the DTO.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 251	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00067] Definition of EcucChoiceReferenceDef XcpDto2PduMapping ⌈

Parameter Name	XcpDto2PduMapping
Parent Container	XcpDto
Upstream mapping defined	–
Description	This reference specifies the mapping of the DTO to the PDUs from the lower-layer interfaces (CanIf, FrIf, SoAd and Cdd). A reference to a XcpRxPdu is only feasible if the the DaqListType is DAQ_STIM. A reference to a XcpTxPdu is only feasible if the DaqListType is DAQ.
Multiplicity	1
Type	Choice reference to [XcpRxPdu, XcpTxPdu]
Pre-Compile only	true

⌋

10.2.6 XcpOdt

[ECUC_Xcp_00055] Definition of EcucParamConfContainerDef XcpOdt ⌈

Container Name	XcpOdt
Parent Container	XcpDaqList
Upstream mapping defined	–
Description	This container contains ODT-specific parameter for the DAQ list.

▽

△

Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpOdtEntryMaxSize	1	[ECUC_Xcp_00060]
XcpOdtNumber	0..1	[ECUC_Xcp_00057]
XcpOdt2DtoMapping	0..1	[ECUC_Xcp_00056]

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpOdtEntry	1..*	This container collects all configuration parameters that comprise an ODT entry.

⌋

[ECUC_Xcp_00060] Definition of EcucIntegerParamDef XcpOdtEntryMaxSize ⌈

Parameter Name	XcpOdtEntryMaxSize	
Parent Container	XcpOdt	
Upstream mapping defined	–	
Description	This parameter indicates the upper limit for the size of the element described by an ODT entry. Depending on the DaqListType this ODT belongs to it describes the limit for a DAQ (MAX_ODT_ENTRY_SIZE_DAQ) or a STIM (MAX_ODT_ENTRY_SIZE_STIM).	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 254	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00057] Definition of EcucIntegerParamDef XcpOdtNumber ⌈

Parameter Name	XcpOdtNumber	
Parent Container	XcpOdt	
Upstream mapping defined	–	
Description	Index number of this ODT within the DAQ list.	
Multiplicity	0..1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 251	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00056] Definition of EcucReferenceDef XcpOdt2DtoMapping ⌈

Parameter Name	XcpOdt2DtoMapping
Parent Container	XcpOdt
Upstream mapping defined	–
Description	This reference maps the ODT to the according DTO in which it will be transmitted.
Multiplicity	0..1
Type	Reference to XcpDto
Pre-Compile only	true

⌋

10.2.7 XcpOdtEntry

[ECUC_Xcp_00061] Definition of EcucParamConfContainerDef XcpOdtEntry ⌈

Container Name	XcpOdtEntry
Parent Container	XcpOdt
Upstream mapping defined	–
Description	This container collects all configuration parameters that comprise an ODT entry.
Multiplicity	1..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpOdtEntryAddress	0..1	[ECUC_Xcp_00063]
XcpOdtEntryBitOffset	0..1	[ECUC_Xcp_00179]
XcpOdtEntryLength	0..1	[ECUC_Xcp_00064]
XcpOdtEntryNumber	0..1	[ECUC_Xcp_00062]

No Included Containers

⌋

[ECUC_Xcp_00063] Definition of EcucLinkerSymbolDef XcpOdtEntryAddress ⌈

Parameter Name	XcpOdtEntryAddress
Parent Container	XcpOdtEntry
Upstream mapping defined	–
Description	Memory address that the ODT entry is referencing to.
Multiplicity	0..1
Type	EcucLinkerSymbolDef
Default value	–
maxLength	–
minLength	–

▽

△

Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00179] Definition of EcucIntegerParamDef XcpOdtEntryBitOffset ⌈

Parameter Name	XcpOdtEntryBitOffset	
Parent Container	XcpOdtEntry	
Upstream mapping defined	–	
Description	Represent the bit offset in case of the element represents status bit.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 31	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00064] Definition of EcucIntegerParamDef XcpOdtEntryLength ⌈

Parameter Name	XcpOdtEntryLength	
Parent Container	XcpOdtEntry	
Upstream mapping defined	–	
Description	Length of the referenced memory area that is referenced by the ODT entry.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	–	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00062] Definition of EcucIntegerParamDef XcpOdtEntryNumber ⌈

Parameter Name	XcpOdtEntryNumber	
Parent Container	XcpOdtEntry	
Upstream mapping defined	–	
Description	Index number of the ODT entry	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 254	
Default value	–	
Pre-Compile only	true	

⌋

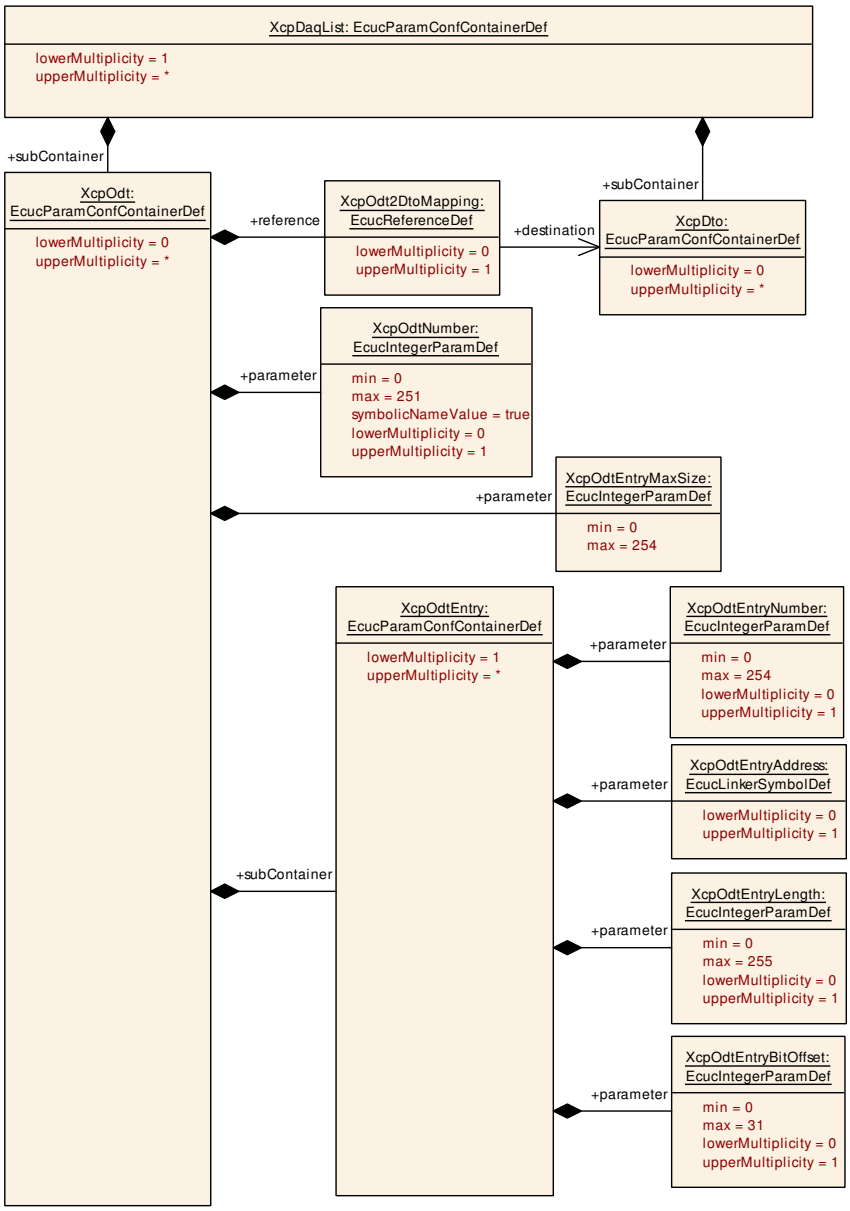


Figure 10.2: Diagram XcpOdtEntry

10.2.8 XcpEventChannel

[ECUC_Xcp_00150] Definition of EcucParamConfContainerDef XcpEventChannel ⌈

Container Name	XcpEventChannel
Parent Container	XcpConfig
Upstream mapping defined	–
Description	This container contains the configuration of event channels on the XCP slave.

▽

△

Multiplicity	1..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpEventChannelConsistency	1	[ECUC_Xcp_00171]
XcpEventChannelMaxDaqList	1	[ECUC_Xcp_00153]
XcpEventChannelNumber	1	[ECUC_Xcp_00152]
XcpEventChannelPriority	1	[ECUC_Xcp_00154]
XcpEventChannelTimeCycle	1	[ECUC_Xcp_00173]
XcpEventChannelTimeUnit	0..1	[ECUC_Xcp_00174]
XcpEventChannelType	1	[ECUC_Xcp_00172]
XcpEventChannelTriggeredDaqListRef	0..*	[ECUC_Xcp_00151]

No Included Containers

⌋

[ECUC_Xcp_00171] Definition of EcucEnumerationParamDef XcpEventChannelConsistency ⌈

Parameter Name	XcpEventChannelConsistency	
Parent Container	XcpEventChannel	
Upstream mapping defined	–	
Description	Type of consistency used by event channel	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DAQ	Consistency on DAQ list level
	EVENT	Consistency on Event Channel Level
	ODT	Consistency on ODT level (default value).
Default value	ODT	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00153] Definition of EcucIntegerParamDef XcpEventChannelMaxDaqList ⌈

Parameter Name	XcpEventChannelMaxDaqList	
Parent Container	XcpEventChannel	
Upstream mapping defined	–	
Description	Maximum amount of DAQ lists that are handled by this event channel.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	

▽

△

Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00152] Definition of EcucIntegerParamDef XcpEventChannelNumber ⌈

Parameter Name	XcpEventChannelNumber
Parent Container	XcpEventChannel
Upstream mapping defined	–
Description	Index number of the event channel.
Multiplicity	1
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)
Range	0 .. 65534
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00154] Definition of EcucIntegerParamDef XcpEventChannelPriority ⌈

Parameter Name	XcpEventChannelPriority
Parent Container	XcpEventChannel
Upstream mapping defined	–
Description	Priority of the event channel
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 255
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00173] Definition of EcucIntegerParamDef XcpEventChannelTimeCycle ⌈

Parameter Name	XcpEventChannelTimeCycle
Parent Container	XcpEventChannel
Upstream mapping defined	–
Description	The event channel time cycle indicates which sampling period is used to process this event channel. A value of 0 means 'Not cyclic'.
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 255
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00174] Definition of EcucEnumerationParamDef XcpEventChannelTimeUnit ⌈

Parameter Name	XcpEventChannelTimeUnit	
Parent Container	XcpEventChannel	
Upstream mapping defined	–	
Description	This configuration parameter indicates the unit of the event channel time cycle. Dependency: Dependent on the Parameter EventChannelTimeCycle. When this parameter is set to 0, the entire event channel time unit parameter shall be ignored.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	TIMESTAMP_UNIT_100MS	Unit is 100 millisecond.
	TIMESTAMP_UNIT_100NS	Unit is 100 nanosecond.
	TIMESTAMP_UNIT_100PS	Unit is 100 picosecond.
	TIMESTAMP_UNIT_100US	Unit is 100 microsecond.
	TIMESTAMP_UNIT_10MS	Unit is 10 millisecond.
	TIMESTAMP_UNIT_10NS	Unit is 10 nanosecond.
	TIMESTAMP_UNIT_10PS	Unit is 10 picosecond.
	TIMESTAMP_UNIT_10US	Unit is 10 microsecond.
	TIMESTAMP_UNIT_1MS	Unit is 1 millisecond.
	TIMESTAMP_UNIT_1NS	Unit is 1 nonasecond.
	TIMESTAMP_UNIT_1PS	Unit is 1 picosecond.
	TIMESTAMP_UNIT_1S	Unit is 1 second.
	TIMESTAMP_UNIT_1US	Unit is 1 microsecond.
Pre-Compile only	true	

⌋

[ECUC_Xcp_00172] Definition of EcucEnumerationParamDef XcpEventChannelType ⌈

Parameter Name	XcpEventChannelType	
Parent Container	XcpEventChannel	
Upstream mapping defined	–	
Description	This configuration parameter indicates what kind of DAQ list can be allocated to this event channel.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DAQ	only DAQ supported (default value).
	DAQ_STIM	Both DAQ and STIM supported (Simultaneously).
	STIM	only STIM supported
Default value	DAQ	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00151] Definition of EcucReferenceDef XcpEventChannelTriggeredDaqListRef ⌈

Parameter Name	XcpEventChannelTriggeredDaqListRef
Parent Container	XcpEventChannel
Upstream mapping defined	–
Description	References all DAQ lists that are trigged by this event channel.
Multiplicity	0..*
Type	Reference to XcpDaqList
Pre-Compile only	true

⌋

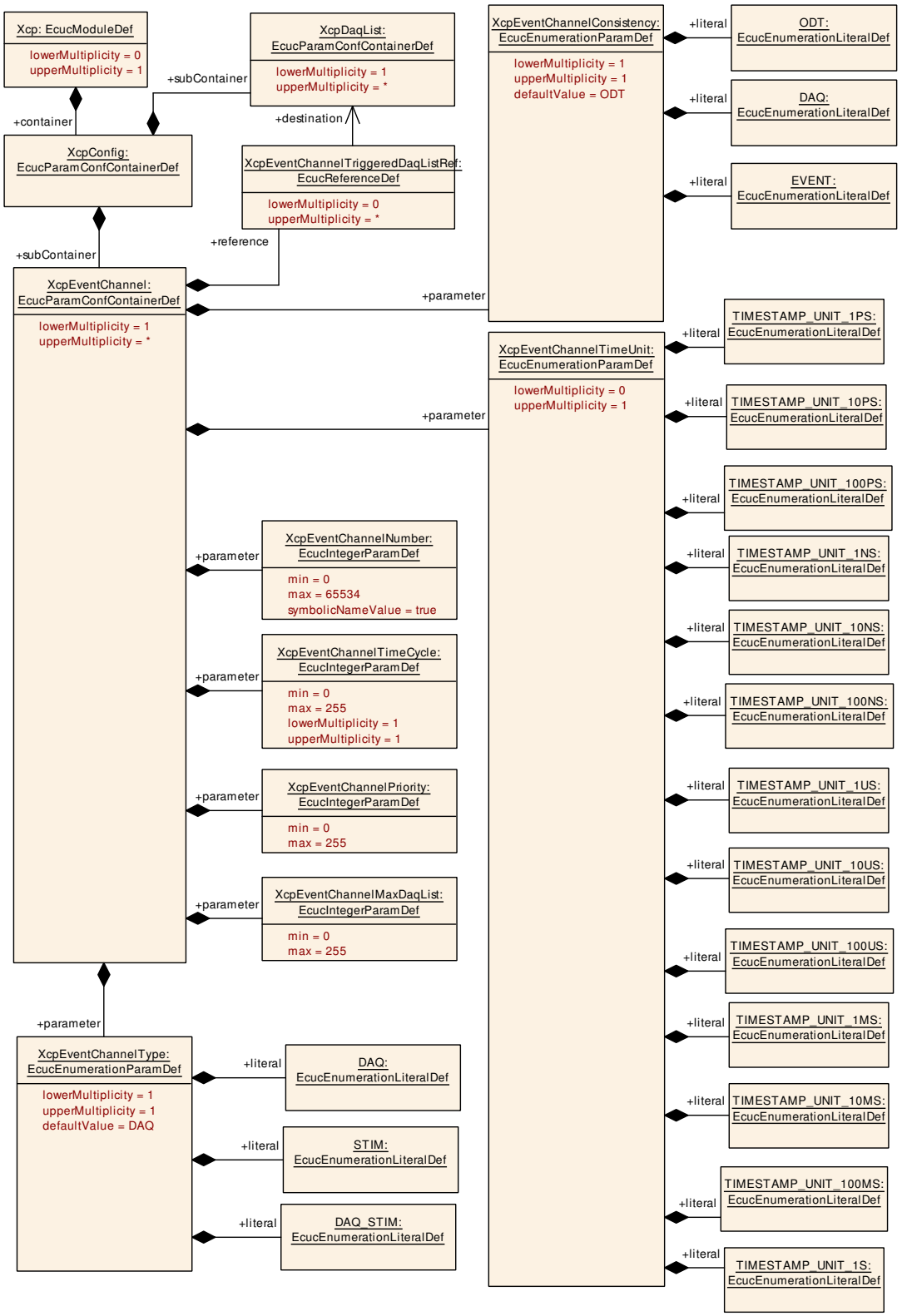


Figure 10.3: Diagram XcpChannel

10.2.9 XcpPdu

[ECUC_Xcp_00100] Definition of EcucChoiceContainerDef XcpPdu ⌈

Choice Container Name	XcpPdu
Parent Container	XcpConfig
Upstream mapping defined	[10, CP_TPS_SystemTemplate]
Description	Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.
Multiplicity	1..*
Pre-Compile only	true

No Included Parameters

Container Choices		
Container Name	**Multiplicity**	**Description**
XcpRxPdu	0..1	This container specifies received PDUs.
XcpTxPdu	0..1	This container specifies transmission PDUs.

⌋

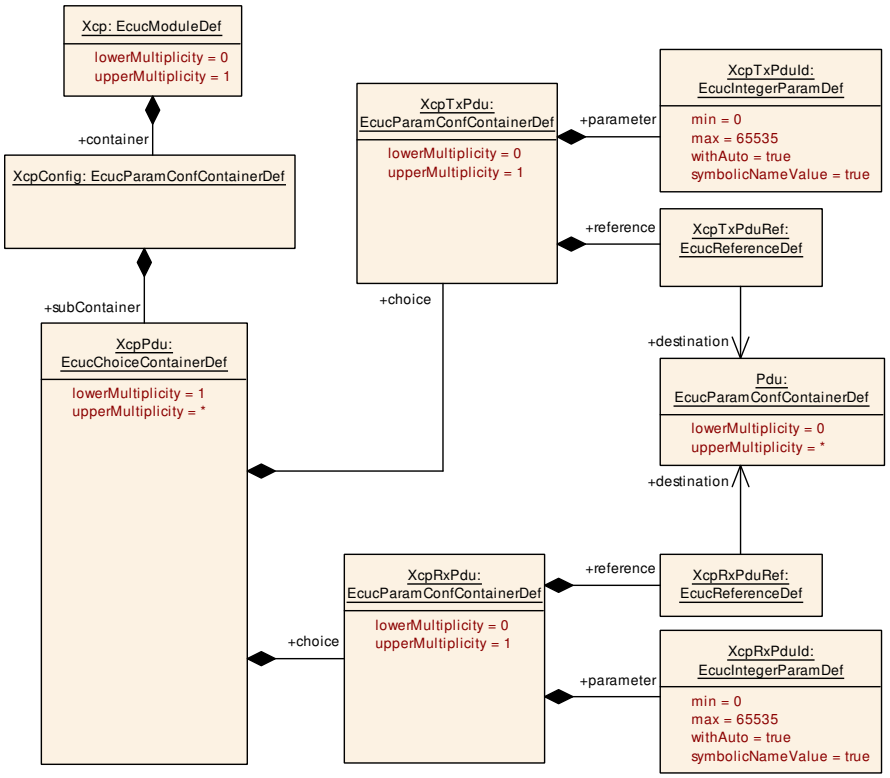


Figure 10.4: Diagram XcpPdu

10.2.10 XcpRxPdu

[ECUC_Xcp_00105] Definition of EcucParamConfContainerDef XcpRxPdu ⌈

Container Name	XcpRxPdu
Parent Container	XcpPdu
Upstream mapping defined	[10, CP_TPS_SystemTemplate]
Description	This container specifies received PDUs.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
XcpRxPduId	1	[ECUC_Xcp_00106]
XcpRxPduRef	1	[ECUC_Xcp_00107]

No Included Containers

⌋

[ECUC_Xcp_00106] Definition of EcucIntegerParamDef XcpRxPduId ⌈

Parameter Name	XcpRxPduId	
Parent Container	XcpRxPdu	
Upstream mapping defined	–	
Description	ID of the PDU that will be received via a Xcp_<module>RxIndication.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00107] Definition of EcucReferenceDef XcpRxPduRef ⌈

Parameter Name	XcpRxPduRef	
Parent Container	XcpRxPdu	
Upstream mapping defined	–	
Description	Reference to the Rx PDU	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.11 XcpTxPdu

[ECUC_Xcp_00101] Definition of EcucParamConfContainerDef XcpTxPdu ⌈

Container Name	XcpTxPdu
Parent Container	XcpPdu
Upstream mapping defined	[10, CP_TPS_SystemTemplate]
Description	This container specifies transmission PDUs.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
XcpTxPduId	1	[ECUC_Xcp_00103]
XcpTxPduRef	1	[ECUC_Xcp_00104]

No Included Containers

⌋

[ECUC_Xcp_00103] Definition of EcucIntegerParamDef XcpTxPduId ⌈

Parameter Name	XcpTxPduId	
Parent Container	XcpTxPdu	
Upstream mapping defined	–	
Description	The PDU identifier, which has to be used by the lower layer BSW module for Tx Confirmations or TriggerTransmits.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 65535	
Default value	–	
With Auto	true	
Pre-Compile only	true	

⌋

[ECUC_Xcp_00104] Definition of EcucReferenceDef XcpTxPduRef ⌈

Parameter Name	XcpTxPduRef	
Parent Container	XcpTxPdu	
Upstream mapping defined	–	
Description	Reference to the external PDU definition.	
Multiplicity	1	
Type	Reference to Pdu	
Post-Build Value	Loadable	Selectable

⌋

10.2.12 XcpCommunicationChannel

[ECUC_Xcp_00183] Definition of EcucParamConfContainerDef XcpCommunicationChannel ⌈

Container Name	XcpCommunicationChannel
Parent Container	XcpConfig
Upstream mapping defined	[10, CP_TPS_SystemTemplate]
Description	This container represents the configuration of the communication channel of XCP.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
XcpChannelRxPduRef	0..1	[ECUC_Xcp_00185]
XcpChannelTxPduRef	1	[ECUC_Xcp_00184]
XcpComMChannelRef	1	[ECUC_Xcp_00186]

No Included Containers

⌋

[ECUC_Xcp_00185] Definition of EcucReferenceDef XcpChannelRxPduRef ⌈

Parameter Name	XcpChannelRxPduRef	
Parent Container	XcpCommunicationChannel	
Upstream mapping defined	–	
Description	Optional reference to the XCP Rx PDU.	
Multiplicity	0..1	
Type	Reference to XcpRxPdu	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Xcp_00184] Definition of EcucReferenceDef XcpChannelTxPduRef ⌈

Parameter Name	XcpChannelTxPduRef	
Parent Container	XcpCommunicationChannel	
Upstream mapping defined	–	
Description	Reference to the XCP Tx PDU.	
Multiplicity	1	
Type	Reference to XcpTxPdu	
Post-Build Value	Loadable	Selectable

⌋

[ECUC_Xcp_00186] Definition of EcucReferenceDef XcpComMChannelRef ⌈

Parameter Name	XcpComMChannelRef
Parent Container	XcpCommunicationChannel
Upstream mapping defined	–
Description	Reference to the ComM channel the PDUs belong to.

▽

△

Multiplicity	1	
Type	Reference to ComMChannel	
Post-Build Value	Loadable	Selectable

⌋

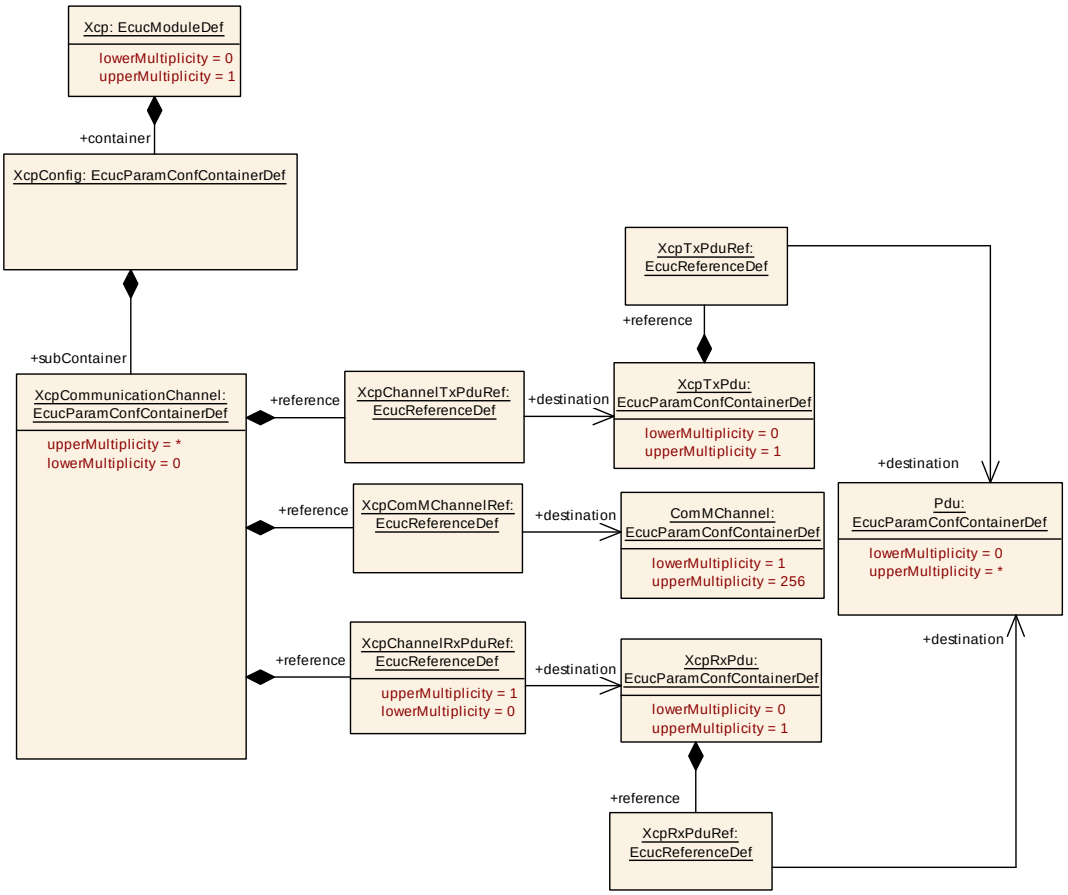


Figure 10.5: Diagram XcpCommunicationChannel

10.2.13 XcpPageSwitching

[ECUC_Xcp_00187] Definition of EcucParamConfContainerDef XcpPageSwitching ⌈

Container Name	XcpPageSwitching
Parent Container	XcpConfig
Upstream mapping defined	–
Description	This container represents configuration of the page switching feature.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpSegment	0..*	This container represents configuration of the page switching segment element.

⌋

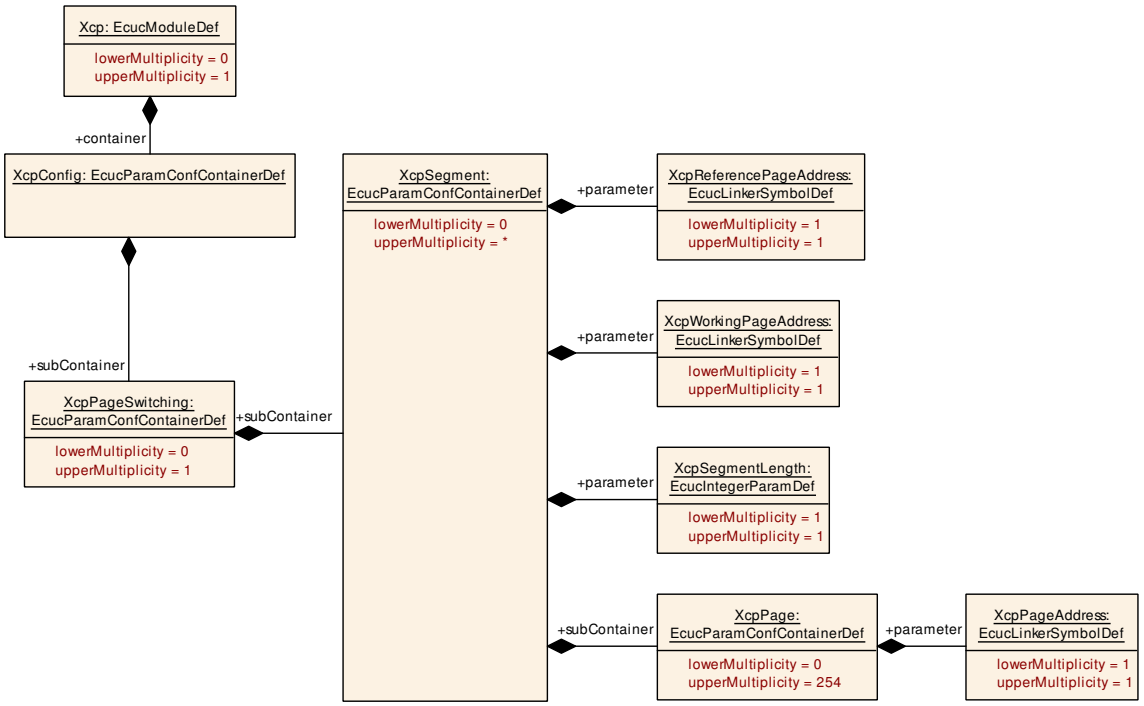


Figure 10.6: Diagram XcpPageSwitching

10.2.14 XcpSegment

[ECUC_Xcp_00188] Definition of EcucParamConfContainerDef XcpSegment ⌈

Container Name	XcpSegment
Parent Container	XcpPageSwitching
Upstream mapping defined	–
Description	This container represents configuration of the page switching segment element.
Multiplicity	0..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpReferencePageAddress	1	[ECUC_Xcp_00189]
XcpSegmentLength	1	[ECUC_Xcp_00191]
XcpWorkingPageAddress	1	[ECUC_Xcp_00190]

Included Containers		
Container Name	**Multiplicity**	**Description**
XcpPage	0..254	This container represents configuration of the optional page element.

⌋

[ECUC_Xcp_00189] Definition of EcucLinkerSymbolDef XcpReferencePageAddress ⌈

Parameter Name	XcpReferencePageAddress
Parent Container	XcpSegment
Upstream mapping defined	–
Description	Memory address of the reference page (Page ID = 0).
Multiplicity	1
Type	EcucLinkerSymbolDef
Default value	–
maxLength	–
minLength	–
Regular Expression	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00191] Definition of EcucIntegerParamDef XcpSegmentLength ⌈

Parameter Name	XcpSegmentLength
Parent Container	XcpSegment
Upstream mapping defined	–
Description	Length of the segment in bytes.
Multiplicity	1
Type	EcucIntegerParamDef
Range	0 .. 18446744073709551615
Default value	–
Pre-Compile only	true

⌋

[ECUC_Xcp_00190] Definition of EcucLinkerSymbolDef XcpWorkingPageAddress ⌈

Parameter Name	XcpWorkingPageAddress
Parent Container	XcpSegment
Upstream mapping defined	–
Description	Memory address address of the working page (Page ID = 1).
Multiplicity	1
Type	EcucLinkerSymbolDef
Default value	–
maxLength	–
minLength	–
Regular Expression	–
Pre-Compile only	true

⌋

10.2.15 XcpPage

[ECUC_Xcp_00192] Definition of EcucParamConfContainerDef XcpPage ⌈

Container Name	XcpPage
Parent Container	XcpSegment
Upstream mapping defined	–
Description	This container represents configuration of the optional page element.
Multiplicity	0..254
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
XcpPageAddress	1	[ECUC_Xcp_00193]

No Included Containers

⌋

[ECUC_Xcp_00193] Definition of EcucLinkerSymbolDef XcpPageAddress ⌈

Parameter Name	XcpPageAddress
Parent Container	XcpPage
Upstream mapping defined	–
Description	Memory address of the optional page (Page ID = 2 ... 255).
Multiplicity	1
Type	EcucLinkerSymbolDef
Default value	–

▽

△

maxLength	–
minLength	–
Regular Expression	–
Pre-Compile only	true

⌋

10.3 Published Information

For details refer to [2] Chapter 10.3 *“Published Information”*.

A Not applicable requirements

[SWS_Xcp_NA_00999]

Upstream requirements: SRS_BSW_00004, SRS_BSW_00168, SRS_BSW_00171, SRS_BSW_00170, SRS_BSW_00336, SRS_BSW_00369, SRS_BSW_00375, SRS_BSW_00384, SRS_BSW_00386, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00395, SRS_BSW_00403, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00425, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00432, SRS_BSW_00461, SRS_BSW_00469, SRS_BSW_00471, SRS_BSW_00472, SRS_BSW_00478, SRS_BSW_00490, SRS_BSW_00491,

⌈These requirements are not applicable to this specification.⌋

B Change history of AUTOSAR traceable items

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

B.1 Traceable item history of this document according to AUTOSAR Release R22-11

B.1.1 Added Specification Items in R22-11

Number	Heading
[SWS_Xcp_00102]	
[SWS_Xcp_00103]	
[SWS_Xcp_00104]	
[SWS_Xcp_00105]	
[SWS_Xcp_00501]	
[SWS_Xcp_00701]	
[SWS_Xcp_00702]	
[SWS_Xcp_00703]	
[SWS_Xcp_00705]	
[SWS_Xcp_00706]	
[SWS_Xcp_00707]	
[SWS_Xcp_00708]	
[SWS_Xcp_00709]	
[SWS_Xcp_00710]	
[SWS_Xcp_00711]	
[SWS_Xcp_00712]	
[SWS_Xcp_00713]	
[SWS_Xcp_00714]	
[SWS_Xcp_00715]	
[SWS_Xcp_00716]	
[SWS_Xcp_00718]	
[SWS_Xcp_00719]	
[SWS_Xcp_00720]	
[SWS_Xcp_00721]	
[SWS_Xcp_00722]	
[SWS_Xcp_00723]	
[SWS_Xcp_00724]	

▽

△

Number	Heading
[SWS_Xcp_00725]	
[SWS_Xcp_00726]	
[SWS_Xcp_00728]	
[SWS_Xcp_00729]	
[SWS_Xcp_00730]	
[SWS_Xcp_00731]	
[SWS_Xcp_00732]	
[SWS_Xcp_00733]	
[SWS_Xcp_00734]	
[SWS_Xcp_00735]	
[SWS_Xcp_00736]	
[SWS_Xcp_00737]	
[SWS_Xcp_00738]	
[SWS_Xcp_00739]	
[SWS_Xcp_00740]	
[SWS_Xcp_00741]	
[SWS_Xcp_00742]	
[SWS_Xcp_00761]	
[SWS_Xcp_00763]	
[SWS_Xcp_00766]	
[SWS_Xcp_00768]	
[SWS_Xcp_00801]	Definition of imported datatypes of module Xcp
[SWS_Xcp_00802]	
[SWS_Xcp_00803]	Definition of API function Xcp_Init
[SWS_Xcp_00807]	Definition of API function Xcp_GetVersionInfo
[SWS_Xcp_00813]	Definition of callback function Xcp_<Lo>RxIndication
[SWS_Xcp_00814]	Definition of callback function Xcp_<Lo>TxConfirmation
[SWS_Xcp_00823]	Definition of scheduled function Xcp_MainFunction
[SWS_Xcp_00824]	
[SWS_Xcp_00825]	
[SWS_Xcp_00832]	Definition of optional interfaces in module Xcp
[SWS_Xcp_00835]	Definition of callback function Xcp_<Lo>TriggerTransmit
[SWS_Xcp_00836]	
[SWS_Xcp_00840]	
[SWS_Xcp_00841]	
[SWS_Xcp_00842]	
[SWS_Xcp_00843]	

▽

△

Number	Heading
[SWS_Xcp_00844]	Definition of callback function Xcp_SetTransmissionMode
[SWS_Xcp_00845]	Definition of datatype Xcp_ConfigType
[SWS_Xcp_00846]	Definition of datatype Xcp_TransmissionModeType
[SWS_Xcp_00847]	
[SWS_Xcp_00848]	
[SWS_Xcp_00849]	
[SWS_Xcp_00850]	
[SWS_Xcp_00852]	
[SWS_Xcp_00853]	
[SWS_Xcp_00854]	
[SWS_Xcp_00855]	
[SWS_Xcp_00856]	
[SWS_Xcp_00857]	Definiton of development errors in module Xcp
[SWS_Xcp_00859]	
[SWS_Xcp_91001]	Definition of mandatory interfaces in module Xcp
[SWS_Xcp_NA_00999]	

Table B.1: Added Specification Items in R22-11

B.1.2 Changed Specification Items in R22-11

none

B.1.3 Deleted Specification Items in R22-11

none

B.2 Traceable item history of this document according to AUTOSAR Release R23-11

B.2.1 Added Specification Items in R23-11

Number	Heading
[SWS_Xcp_00102]	
[SWS_Xcp_00103]	
[SWS_Xcp_00104]	
[SWS_Xcp_00105]	
[SWS_Xcp_00501]	
[SWS_Xcp_00701]	
[SWS_Xcp_00702]	
[SWS_Xcp_00703]	
[SWS_Xcp_00705]	
[SWS_Xcp_00706]	
[SWS_Xcp_00707]	
[SWS_Xcp_00708]	
[SWS_Xcp_00709]	
[SWS_Xcp_00710]	
[SWS_Xcp_00711]	
[SWS_Xcp_00712]	
[SWS_Xcp_00713]	
[SWS_Xcp_00714]	
[SWS_Xcp_00715]	
[SWS_Xcp_00716]	
[SWS_Xcp_00718]	
[SWS_Xcp_00719]	
[SWS_Xcp_00720]	
[SWS_Xcp_00721]	
[SWS_Xcp_00722]	
[SWS_Xcp_00723]	
[SWS_Xcp_00724]	
[SWS_Xcp_00725]	
[SWS_Xcp_00726]	
[SWS_Xcp_00728]	
[SWS_Xcp_00729]	
[SWS_Xcp_00730]	
[SWS_Xcp_00731]	

▽

△

Number	Heading
[SWS_Xcp_00732]	
[SWS_Xcp_00733]	
[SWS_Xcp_00734]	
[SWS_Xcp_00735]	
[SWS_Xcp_00736]	
[SWS_Xcp_00737]	
[SWS_Xcp_00738]	
[SWS_Xcp_00739]	
[SWS_Xcp_00740]	
[SWS_Xcp_00741]	
[SWS_Xcp_00742]	
[SWS_Xcp_00761]	
[SWS_Xcp_00763]	
[SWS_Xcp_00766]	
[SWS_Xcp_00768]	
[SWS_Xcp_00801]	Definition of imported datatypes of module Xcp
[SWS_Xcp_00802]	
[SWS_Xcp_00803]	Definition of API function Xcp_Init
[SWS_Xcp_00807]	Definition of API function Xcp_GetVersionInfo
[SWS_Xcp_00813]	Definition of callback function Xcp_<Lo>RxIndication
[SWS_Xcp_00814]	Definition of callback function Xcp_<Lo>TxConfirmation
[SWS_Xcp_00823]	Definition of scheduled function Xcp_MainFunction
[SWS_Xcp_00824]	
[SWS_Xcp_00825]	
[SWS_Xcp_00832]	Definition of optional interfaces in module Xcp
[SWS_Xcp_00835]	Definition of callback function Xcp_<Lo>TriggerTransmit
[SWS_Xcp_00836]	
[SWS_Xcp_00840]	
[SWS_Xcp_00841]	
[SWS_Xcp_00842]	
[SWS_Xcp_00843]	
[SWS_Xcp_00844]	Definition of callback function Xcp_SetTransmissionMode
[SWS_Xcp_00845]	Definition of datatype Xcp_ConfigType
[SWS_Xcp_00846]	Definition of datatype Xcp_TransmissionModeType
[SWS_Xcp_00847]	
[SWS_Xcp_00848]	
[SWS_Xcp_00849]	
[SWS_Xcp_00850]	

▽

△

Number	Heading
[SWS_Xcp_00852]	
[SWS_Xcp_00853]	
[SWS_Xcp_00854]	
[SWS_Xcp_00855]	
[SWS_Xcp_00856]	
[SWS_Xcp_00857]	Definiton of development errors in module Xcp
[SWS_Xcp_00859]	
[SWS_Xcp_91001]	Definition of mandatory interfaces in module Xcp
[SWS_Xcp_NA_00999]	

Table B.2: Added Specification Items in R23-11

B.2.2 Changed Specification Items in R23-11

none

B.2.3 Deleted Specification Items in R23-11

none

B.3 Traceable item history of this document according to AUTOSAR Release R24-11

B.3.1 Added Specification Items in R24-11

Number	Heading
[SWS_Xcp_00860]	Location of the XCP module in the AUTOSAR communication stack
[SWS_Xcp_00862]	Sending and receiving of calibration data
[SWS_Xcp_00863]	Waiting for call of `Xcp_TxConfirmation` to avoid overwritting previously transmitted data
[SWS_Xcp_00864]	Use APIs of PduR for exchanging data via CAN
[SWS_Xcp_00865]	Use APIs of PduR for exchanging data via FlexRay
[SWS_Xcp_00866]	Use APIs of PduR for exchanging data via Ethernet (i.e. TcpIp)
[SWS_Xcp_00867]	Development error handling in context of `Xcp_RxIndication`
[SWS_Xcp_00868]	Development error handling in context of `Xcp_TxConfirmation`
[SWS_Xcp_00869]	Caveats on call of `Xcp_TxConfirmation`

▽

△

Number	Heading
[SWS_Xcp_00870]	Development error handling in context of `Xcp_TriggerTransmit`
[SWS_Xcp_00871]	Caveats on call of `Xcp_TriggerTransmit`

Table B.3: Added Specification Items in R24-11

B.3.2 Changed Specification Items in R24-11

Number	Heading
[SWS_Xcp_00701]	
[SWS_Xcp_00712]	
[SWS_Xcp_00714]	
[SWS_Xcp_00720]	
[SWS_Xcp_00734]	
[SWS_Xcp_00813]	Definition of callback function Xcp_RxIndication
[SWS_Xcp_00814]	Definition of callback function Xcp_TxConfirmation
[SWS_Xcp_00832]	Definition of optional interfaces requested by module Xcp
[SWS_Xcp_00835]	Definition of callback function Xcp_TriggerTransmit
[SWS_Xcp_00840]	
[SWS_Xcp_00841]	
[SWS_Xcp_00842]	
[SWS_Xcp_00843]	
[SWS_Xcp_00847]	
[SWS_Xcp_00859]	

Table B.4: Changed Specification Items in R24-11

B.3.3 Deleted Specification Items in R24-11

none

B.3.4 Added Constraints in R24-11

Number	Heading
[SWS_Xcp_CONSTR_00861]	`XcpRxPdu` and `XcpTxPdu` constraint for keeping the local buffer

Table B.5: Added Constraints in R24-11

B.3.5 Changed Constraints in R24-11

none

B.3.6 Deleted Constraints in R24-11

none

B.4 Traceable item history of this document according to AUTOSAR Release R25-11

B.4.1 Added Specification Items in R25-11

none

B.4.2 Changed Specification Items in R25-11

[ECUC_Xcp_00001] [ECUC_Xcp_00020] [ECUC_Xcp_00050] [ECUC_Xcp_00055] [ECUC_Xcp_00061] [ECUC_Xcp_00065] [ECUC_Xcp_00100] [ECUC_Xcp_00101] [ECUC_Xcp_00105] [ECUC_Xcp_00150] [ECUC_Xcp_00183] [ECUC_Xcp_00187] [ECUC_Xcp_00188] [ECUC_Xcp_00192] [SWS_Xcp_00715] [SWS_Xcp_00801] [SWS_Xcp_00832] [SWS_Xcp_00860]

B.4.3 Deleted Specification Items in R25-11

[SWS_Xcp_00701] [SWS_Xcp_00712] [SWS_Xcp_00714] [SWS_Xcp_00720] [SWS_Xcp_00734] [SWS_Xcp_00840] [SWS_Xcp_00841] [SWS_Xcp_00842] [SWS_Xcp_00843] [SWS_Xcp_00847] [SWS_Xcp_00859] [SWS_Xcp_00862]

B.4.4 Added Constraints in R25-11

none

B.4.5 Changed Constraints in R25-11

none

B.4.6 Deleted Constraints in R25-11

none

B.5 Traceable item history of this document according to AUTOSAR Release R26-11

B.5.1 Added Specification Items in R26-11

[SWS_Xcp_00872] [SWS_Xcp_00873] [SWS_Xcp_00874] [SWS_Xcp_00875] [SWS_Xcp_00876] [SWS_Xcp_00877] [SWS_Xcp_91002]

B.5.2 Changed Specification Items in R26-11

none

B.5.3 Deleted Specification Items in R26-11

[ECUC_Xcp_00002] [SWS_Xcp_00807] [SWS_Xcp_00825] [SWS_Xcp_00836]

B.5.4 Added Constraints in R26-11

none

B.5.5 Changed Constraints in R26-11

none

B.5.6 Deleted Constraints in R26-11

Number	Heading
[SWS_Xcp_CONSTR_00861]	`XcpRxPdu` and `XcpTxPdu` constraint for keeping the local buffer

Table B.6: Deleted Constraints in R26-11