Document Title	Specification of Chinese Vehicle-2-X Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1031

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R24-11

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes
2023-11-23	R23-11	AUTOSAR Release Management	• No content changes
2022-11-24	R22-11	AUTOSAR Release Management	• 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.

Contents

1 Introduction and functional overview

This document specifies the functionality, APIs and the configuration of the AUTOSAR Basic Software module Chinese Vehicle-2-X Management (CnV2xM).
The Chinese Vehicle-2-X Management (CnV2xM) together with the Chinese Vehicle-2-X Message (CnV2xMsg), Chinese Vehicle-2-X Network (CnV2xNet), Chinese Vehicle-2-X Security (CnV2xSec) and AUTOSAR BSW module Ethernet Interface (EthIf) forms the Chinese V2X stack within the AUTOSAR architecture.
The bases for this document are the Chinese LTE-V2X based standards [1] [2]. It is assumed that the reader is familiar with these standards.

1.1 Architecture Overview

Positioning of the CnV2xM module within the AUTOSAR BSW and the Layered Software architecture is shown in below.

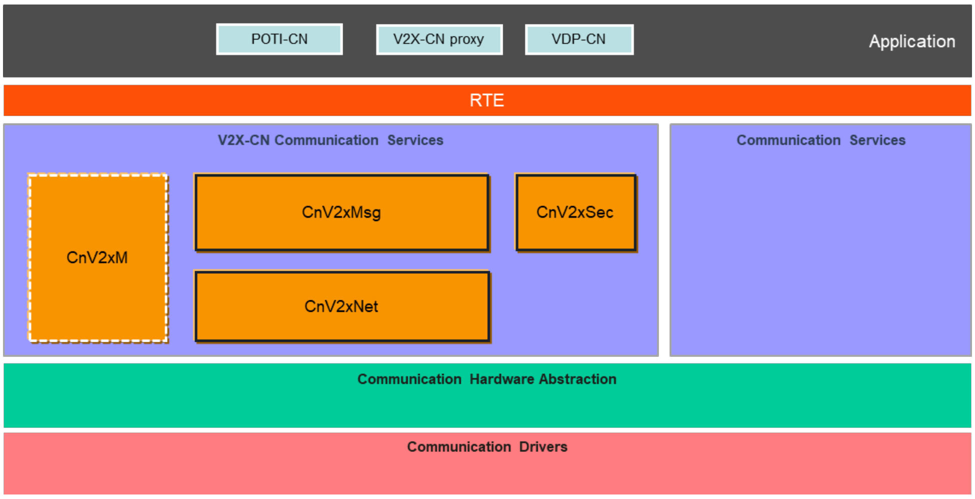

Figure 1.1: AUTOSAR BSW software architecture - CnV2xM scope

1.2 Functional Overview

The CnV2xM module support the operation of the Chinese V2X protocol stack in common V2X channel parameter acquisiton and initialization of cellular V2X driver.
In future,the CnV2xM module will implement some basic services of DME specified in [2], and will support Chinese V2X unicast services based on LTE-V2X on application level.

2 Acronyms and Abbreviations

Abbreviation / Acronym:	**Description:**
BSM	Basic safety Message
C-V2X	Cellular based Vehicle to Everything
CBR	Channel Busy Ratio
CCSA	China Communications Standards Association
CnV2xMsg	Chinese Vehicle-2-X Message
CnV2xM	Chinese Vehicle-2-X Management
CnV2xNet	Chinese Vehicle-2-X Network
CnV2xSec	Chinese Vehicle-2-X Security
DME	Dedicated Management Entity
LTE	Long Term Evolution
LTE-V2X	LTE based Vehicle to Everything
NTCAS	National Technical Committee of Auto Standardization
PC5	The reference point between the UEs (User equipment) used for control and user plane for ProSe (Proximity-based Services) Direct Communication for V2X Service
PPPP	ProSe Per-Packet Priority
TP	Transmit Power

3 Related documentation

3.1 Input documents & related standards and norms

[1] GB/T: Technical requirements and test methods of vehicular communication system based on LTE-V2X direct communication (Draft Edition: 2022-04-01)
http://www.catarc.org.cn/

[2] YD/T 3707-2020: Technical requirements of network layer of LTE-based vehicular communication
http://www.ccsa.org.cn/

[3] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[4] Specification of Default Error Tracer
AUTOSAR_CP_SWS_DefaultErrorTracer

[5] Specification of ECU State Manager
AUTOSAR_CP_SWS_ECUStateManager

[6] Specification of Ethernet Interface
AUTOSAR_CP_SWS_EthernetInterface

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [3], which is also valid for CnV2xM.

Thus, the specification SWS BSW General shall be considered as additional and required specification for CnV2xM.

4 Constraints and assumptions

4.1 Limitations

- The Chinese V2X modules follow the guidance regarding the Day-1 V2X allocations defined in [1] [2], which are by NTCAS and CCSA respectively.
- Wireless communication supports LTE-V2X PC5 only. Other cellular based wireless communication can be extended in future release of AUTOSAR standard.
- DME functions specified in [2] will be implemented to support LTE-V2X based unicast service in future.

4.2 Applicability to car domains

This specification is applicable to all car domains.

5 Dependencies to other modules

This section describes the relations of CnV2xM module to other modules within the AUTOSAR basic software architecture. It outlines the modules that are required or optional for the realization of CnV2xM module and services.

5.1 AUTOSAR Default Error Tracer (DET)

In development mode, CnV2xM module reports errors through the Det_ReportError function of DET Module [4].

5.2 AUTOSAR Ecu State Manager (EcuM)

The EcuM [5] initializes the CnV2xM module by calling CnV2xM_Init specified in 8.3.1 in this document.

5.3 AUTOSAR Ethernet Interface (EthIf)

The Ethernet Interface [6] is the lower layer module of CnV2xNet module.

5.4 AUTOSAR Chinese Vehicle-2-X Message (CnV2xMsg)

The CnV2xMsg can get channel parameters by calling CnV2xM_GetChanTxParams in this document.

6 Requirements Tracing

Requirement	Description	Satisfied by
[CP_SRS_CnV2X_-00301]	The Access layer of Chinese V2X Communication shall be compliant to CCSA specification of Air Interface for LTE-based Vehicular Communication	[CP_SWS_CnV2xM_01003]
[CP_SRS_CnV2X_-00401]	The network layer of Chinese V2X communication shall support a CCSA compliant Network layer protocol of LTE-based vehicular communication	[CP_SWS_CnV2xM_00002] [CP_SWS_CnV2xM_00003] [CP_SWS_CnV2xM_00004] [CP_SWS_CnV2xM_02001] [CP_SWS_CnV2xM_02005] [CP_SWS_CnV2xM_02007] [CP_SWS_CnV2xM_02008] [CP_SWS_CnV2xM_02020]
[CP_SRS_CnV2X_-00404]	The network layer of Chinese V2X communication shall provide CBR or Max data rate to message Layer	[CP_SWS_CnV2xM_00005]
[SRS_BSW_00345]	BSW Modules shall support pre-compile configuration	[CP_SWS_CnV2xM_03001]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[CP_SWS_CnV2xM_02004]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 Startup Behavior

[CP_SWS_CnV2xM_00002]

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈The function CnV2xM_Init of the CnV2xM shall initialize the underlying MCAL/ECUAL module CV2x by EthIf_GetControllerMode and EthIf_SetControllerMode with the respective configured EthIfController CnV2xMEthIfCtrlRef.⌋

[CP_SWS_CnV2xM_00003]

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈The Ethernet State Manager (EthSm) shall not be involved in the startup of the Cellular V2X stack.⌋

7.2 Shutdown Behavior

[CP_SWS_CnV2xM_00004]

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈The Cellular V2X Communication shall be active unless the ECU hardware is being shut down or reset. There are no means to stop the Cellular Vehicle-2-X communication in advance.⌋

7.3 Common Channel Parameter Acquisition

[CP_SWS_CnV2xM_00005]

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00404

⌈The CnV2xM module shall implement cellular V2X Channel parameter acquisition via API CnV2xM_GetChanTxParams.⌋

7.4 Error Classification

7.4.1 Development Errors

[CP_SWS_CnV2xM_00006] Definiton of development errors in module CnV2xM ⌈

Type of error	*Related error code*	*Error value*
API service called with invalid parameter	CNV2XM_E_PARAM	0x01
API service called with invalid pointer	CNV2XM_E_PARAM_POINTER	0x02
API service used withou module initialization	CNV2XM_E_UNINIT	0x03
API service called with invalid configuration pointer	CNV2XM_E_INIT_FAILED	0x04

⌋

7.4.2 Runtime Errors

There are no runtime errors

7.4.3 Production Errors

There are no production errors.

7.4.4 Extended Production Errors

There are no extended production errors.

8 API specification

8.1 Imported types

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

[CP_SWS_CnV2xM_01001] Definition of imported datatypes of module CnV2xM
⌈

Module	**Header File**	**Imported Type**
CV2x	CV2x_GeneralTypes.h	CV2x_GetChanTxParamIdType (draft)
Eth	Eth_GeneralTypes.h	Eth_ModeType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType
V2xM	V2xM.h	V2xM_ConfigType

⌋

8.2 Type definitions

8.2.1 CnV2xM_ConfigType

[CP_SWS_CnV2xM_01002] Definition of datatype CnV2xM_ConfigType

Status: DRAFT

⌈

Name	CnV2xM_ConfigType (draft)	
Kind	Structure	
Elements	implementation specific	
	Type	V2xM_ConfigType
	Comment	The content of the configuration data structure is implementation specific.
Description	Configuration data structure of the CnV2xM module. **Tags:** atp.Status=draft	
Available via	CnV2xM.h	

⌋

8.2.2 CnV2xM_ChanType

[CP_SWS_CnV2xM_01003] Definition of datatype CnV2xM_ChanType

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00301

⌈

Name	CnV2xM_ChanType (draft)		
Kind	Enumeration		
Range	CN_V2X_CH1	–	Channel of 5905-5925MHz band
Description	Specifies the channel assigned for LTE based V2X in China. **Tags:** atp.Status=draft		
Available via	CnV2xM.h		

⌋

8.3 Function definitions

This is a list of functions provided for upper layer modules.

8.3.1 CnV2xM_Init

[CP_SWS_CnV2xM_02001] Definition of API function CnV2xM_Init

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈

Service Name	CnV2xM_Init (draft)	

Syntax	`void CnV2xM_Init (`
	`  const CnV2xM_ConfigType* CfgPtr`
	`)`	

Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CfgPtr	ConfigPtr Pointer to the selected configuration set
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initialize the CnV2xM module **Tags:** atp.Status=draft	
Available via	CnV2xM.h	

⌋

[CP_SWS_CnV2xM_02002]

Status: DRAFT

⌈The function CnV2xM_Init shall store the access to the configuration structure for subsequent API calls.⌋

[CP_SWS_CnV2xM_02003]

Status: DRAFT

⌈If development error detection is enabled: The function CnV2xM_Init shall check the parameter CfgPtr for containing a valid configuration. If the check fails, the function shall raise the development error CNV2XM_E_INIT_FAILED.⌋

[CP_SWS_CnV2xM_02004]

Status: DRAFT
Upstream requirements: SRS_BSW_00414

⌈The Configuration pointer configPtr shall always have a NULL_PTR value.⌋

8.3.2 CnV2xM_GetVersionInfo

[CP_SWS_CnV2xM_02005] Definition of API function CnV2xM_GetVersionInfo

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈

Service Name	CnV2xM_GetVersionInfo (draft)	

Syntax	`void CnV2xM_GetVersionInfo (`
	`  Std_VersionInfoType* VersionInfoPtr`
	`)`	

Service ID [hex]	0x02	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	VersionInfoPtr	Pointer to where to store the version information of this module.
Return value	None	
Description	Returns the version information of this module. **Tags:** atp.Status=draft	
Available via	CnV2xM.h	

⌋

[CP_SWS_CnV2xM_02006]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetVersionInfo shall check the parameter VersionInfoPtr for being valid. If the check fails, the function shall raise the development error CNV2XM_E_PARAM_POINTER.⌋

8.3.3 CnV2xM_GetChanTxParams

[CP_SWS_CnV2xM_02007] Definition of API function CnV2xM_GetChanTxParams

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈

Service Name	CnV2xM_GetChanTxParams (draft)	

Syntax	`Std_ReturnType CnV2xM_GetChanTxParams (`
	`  uint8 CtrlId,`
	`  const CnV2xM_ChanType ChannelId,`
	`  const CV2x_GetChanTxParamIdType* ParamIds,`
	`  uint32* ParamValues,`
	`  uint8 NumParams`
	`)`	

Service ID [hex]	0x03	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlId	Index of the controller within the context of the Cellular V2X Driver
	ChannelId	Index of Transceiver's Radio Channel
	ParamIds	IDs of the Parameters to read
	NumParams	Number of parameters to read
Parameters (inout)	None	
Parameters (out)	ParamValues	Value of the requested Parameters
Return value	Std_ReturnType	`E_OK`: success `E_NOT_OK`: failed setting parameter
Description	Read values related to the receive direction of the channel. For example, this could be a Channel Busy Ratio(CBR) **Tags:** atp.Status=draft	
Available via	CnV2xM.h	

⌋

[CP_SWS_CnV2xM_02008]

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈The function CnV2xM_GetChanTxParams shall provide Tx Channel parameters.⌋

[CP_SWS_CnV2xM_02009]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetChanTxParams shall check that the service CnV2xM_Init was previously called. If the check fails, the function shall raise the development error CNV2XM_E_UNINIT.⌋

[CP_SWS_CnV2xM_02010]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetChanTxParams shall check the parameter CtrlId for being valid. If the check fails, the function shall raise the development error CNV2XM_E_PARAM.⌋

[CP_SWS_CnV2xM_02011]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetChanTxParams shall check the parameter ChannelId for being valid. If the check fails, the function shall raise the development error CNV2XM_E_PARAM.⌋

[CP_SWS_CnV2xM_02012]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetChanTxParams shall check the parameter ParamIds for being valid. If the check fails, the function shall raise the development error CNV2XM_E_PARAM_POINTER.⌋

[CP_SWS_CnV2xM_02013]

Status: DRAFT

⌈If development error detection is enabled: the function CnV2xM_GetChanTxParams shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error CNV2XM_E_PARAM_POINTER.⌋

8.4 Callback notifications

The CnV2xM does not provide any callback functions.

8.5 Scheduled functions

8.5.1 CnV2xM_MainFunction

[CP_SWS_CnV2xM_02020] Definition of scheduled function CnV2xM_MainFunction

Status: DRAFT
Upstream requirements: CP_SRS_CnV2X_00401

⌈

Service Name	CnV2xM_MainFunction (draft)

Syntax	`void CnV2xM_MainFunction (`
	`  void`
	`)`

Service ID [hex]	0x04
Description	Main function of the CnV2xM module for periodical execution of protocol operations. **Tags:** atp.Status=draft
Available via	SchM_CnV2xM.h

⌋

[CP_SWS_CnV2xM_02021]

Status: DRAFT

⌈The function CnV2xM_MainFunction shall be used for getting Tx channel parameters via EthIf_GetChanCV2xPC5TxParams API call from Celluar V2X Driver.⌋

8.6 Expected interfaces

8.6.1 Mandatory interfaces

This chapter defines all external interfaces, which are required to fulfill the core functionality of the module.

[CP_SWS_CnV2xM_02022] Definition of mandatory interfaces required by module CnV2xM ⌈

API Function	***Header File***	***Description***
EthIf_GetChanCV2xPC5TxParams	–	Read values related to the receive direction of the channel. For example, this could be a Channel Busy Ratio(CBR)
EthIf_GetControllerMode	EthIf.h	Obtains the state of the indexed controller
EthIf_SetControllerMode	EthIf.h	Enables / disables the indexed controller

⌋

8.6.2 Optional interfaces

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

[CP_SWS_CnV2xM_02023] Definition of optional interfaces requested by module CnV2xM ⌈

API Function	***Header File***	***Description***
Det_ReportError	Det.h	Service to report development errors.

⌋

9 Sequence diagrams

9.1 CnV2xM Initialization

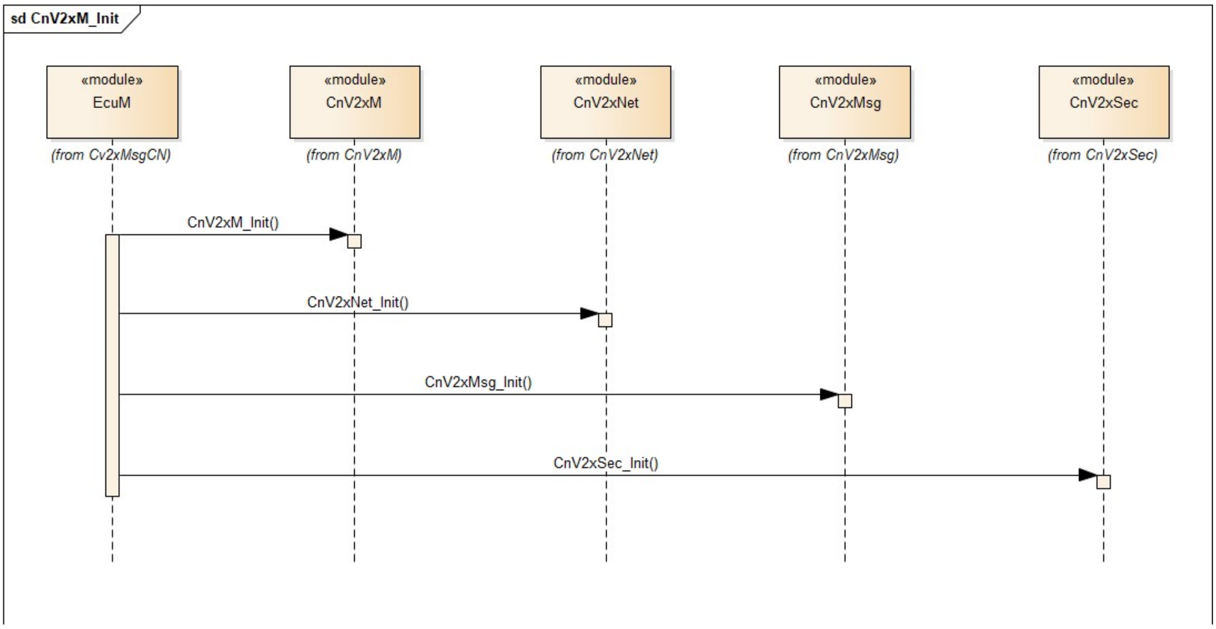

Figure 9.1: CnV2xM Initialization

9.2 Initialization of Cellular V2X Drivers

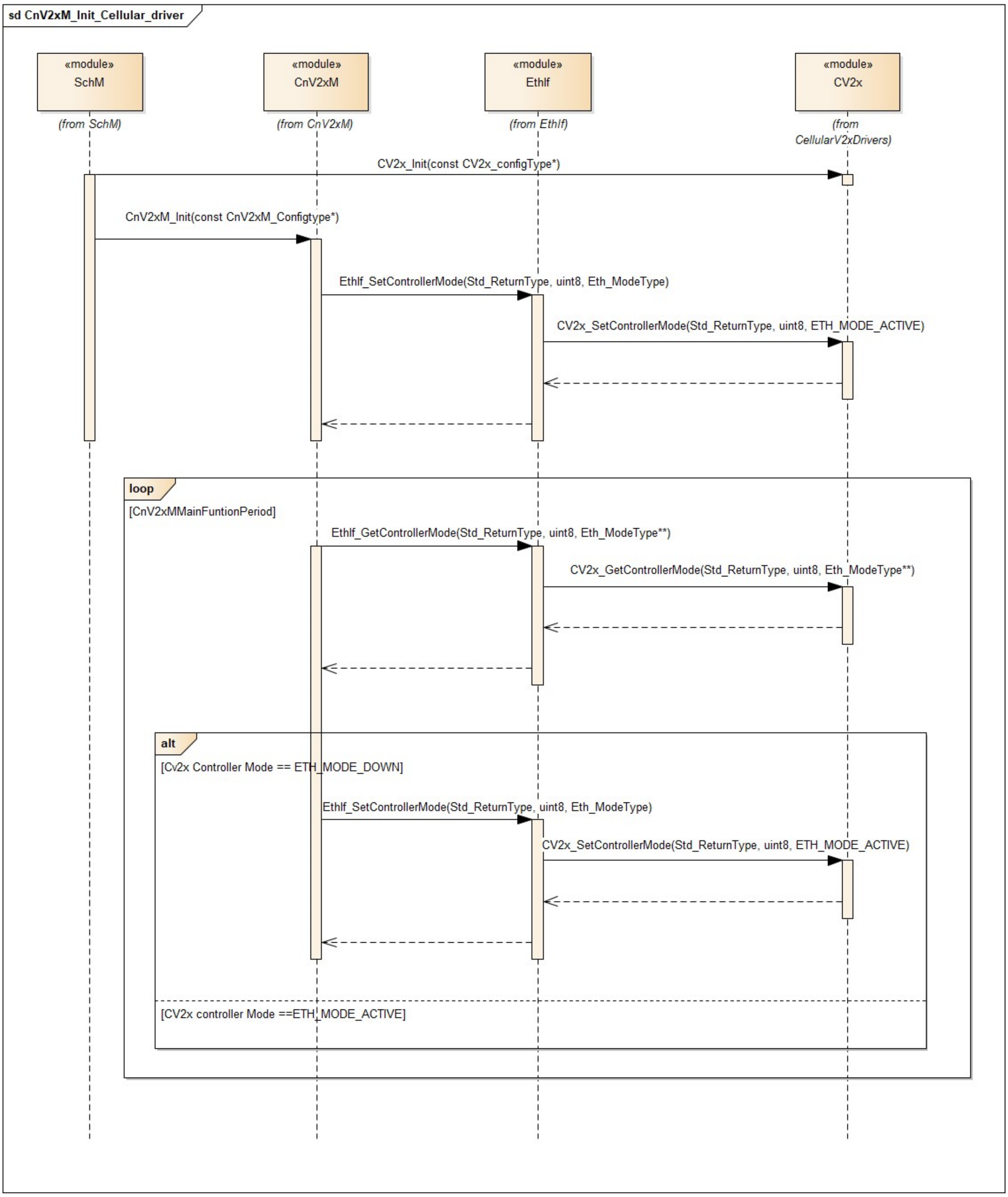
sd CnV2xM_Init_Cellular_driver
«module»
SchM
(from SchM)
«module»
CnV2xM
(from CnV2xM)
«module»
EthIf
(from EthIf)
«module»
CV2x
(from CellularV2xDrivers)
CV2x_Init(const CV2x_configType*)
CnV2xM_Init(const CnV2xM_Configtype*)
EthIf_SetControllerMode(Std_ReturnType, uint8, Eth_ModeType)
CV2x_SetControllerMode(Std_ReturnType, uint8, ETH_MODE_ACTIVE)
loop
[CnV2xMMainFuntionPeriod]
EthIf_GetControllerMode(Std_ReturnType, uint8, Eth_ModeType**)
CV2x_GetControllerMode(Std_ReturnType, uint8, Eth_ModeType**)
alt
[Cv2x Controller Mode == ETH_MODE_DOWN]
EthIf_SetControllerMode(Std_ReturnType, uint8, Eth_ModeType)
CV2x_SetControllerMode(Std_ReturnType, uint8, ETH_MODE_ACTIVE)
[CV2x controller Mode ==ETH_MODE_ACTIVE]

Figure 9.2: Initialization of Cellular V2X Drivers

10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of the module CnV2xM.

10.1 Containers and configuration parameters

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

10.1.1 Variants

[CP_SWS_CnV2xM_03001]

Status: DRAFT
Upstream requirements: SRS_BSW_00345

⌈The CnV2xM module only supports VARIANT-PRE-COMPILE⌋

10.1.2 CnV2xM

[ECUC_CnV2xM_00001] Definition of EcucModuleDef CnV2xM

Status: DRAFT

⌈

Module Name	CnV2xM
Description	Configuration of the CnV2xM module.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CnV2xMGeneral	1	This container contains the general configuration parameters of the BSW module CnV2xM. **Tags:** atp.Status=draft

⌋

10.1.3 CnV2xMGeneral

[ECUC_CnV2xM_00002] Definition of EcucParamConfContainerDef CnV2xMGeneral

Status: DRAFT

⌈

Container Name	CnV2xMGeneral
Parent Container	CnV2xM
Description	This container contains the general configuration parameters of the BSW module Cn V2xM. **Tags:** atp.Status=draft
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
CnV2xMDevErrorDetect	1	[ECUC_CnV2xM_00004]
CnV2xMMainFunctionPeriod	1	[ECUC_CnV2xM_00003]
CnV2xMVersionInfoApi	1	[ECUC_CnV2xM_00005]
CnV2xMEthIfCtrlRef	1	[ECUC_CnV2xM_00006]

No Included Containers

⌋

[ECUC_CnV2xM_00004] Definition of EcucBooleanParamDef CnV2xMDevErrorDetect

Status: DRAFT

⌈

Parameter Name	CnV2xMDevErrorDetect		
Parent Container	CnV2xMGeneral		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. - true: enabled (ON) - false: disabled (OFF) **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_CnV2xM_00003] Definition of EcucFloatParamDef CnV2xMMainFunctionPeriod

Status: DRAFT

⌈

Parameter Name	CnV2xMMainFunctionPeriod		
Parent Container	CnV2xMGeneral		
Description	This parameter defines the schedule period of CnV2xM_MainFunction.Unit:[s] **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. 1[		
Default value	0.1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_CnV2xM_00005] Definition of EcucBooleanParamDef CnV2xMVersionInfoApi

Status: DRAFT

⌈

Parameter Name	CnV2xMVersionInfoApi		
Parent Container	CnV2xMGeneral		
Description	Enable/disables the API for reading the version information of the CnV2xM Module. - true: enabled (ON) - false: disabled (OFF) **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_CnV2xM_00006] Definition of EcucReferenceDef CnV2xMEthIfCtrlRef

Status: DRAFT

⌈

Parameter Name	CnV2xMEthIfCtrlRef		
Parent Container	CnV2xMGeneral		
Description	This is represents the reference to the Ethernet interface taken to transmit the C-V2X packets to. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	Symbolic name reference to EthIfController		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

⌋

A Not applicable requirements

None.

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 R23-11

B.1.1 Added Specification Items in R24-11

none

B.1.2 Changed Specification Items in R24-11

none

B.1.3 Deleted Specification Items in R24-11

none

B.1.4 Added Specification Items in R23-11

none

B.1.5 Changed Specification Items in R23-11

none

B.1.6 Deleted Specification Items in R23-11

none