Document Title	Specification of Vehicle-2-X Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	796

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.4.0

Document Change History

Date	Release	Changed by	Change Description
2018-10-31	4.4.0	AUTOSAR Release Management	• Header file clean-up • Fixed position and time parameter names • Editorial changes
2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Initial Release

Disclaimer

Table of Contents

1 Introduction and functional overview

This document specifies the functionality, API and the configuration of the AUTOSAR Basic Software module Vehicle-2-X Management (V2xM). The Vehicle-2-X Management module together with the Vehicle-2-X Facilities (V2xFac), Vehicle-2-X Basic Transport Protocol (V2xBtp), the Vehicle-2-X GeoNetworking (V2xGn) and the communication driver layer forms the V2X stack within the AUTOSAR architecture.

V2xM is designed to be hardware independent. It controls and supports the services of V2X protocol stack entities.

Note that figures in this document are not regarded as requirements.

1.1 Architectural overview

The position of the V2xM module within the Layered Software Architecture is shown below.

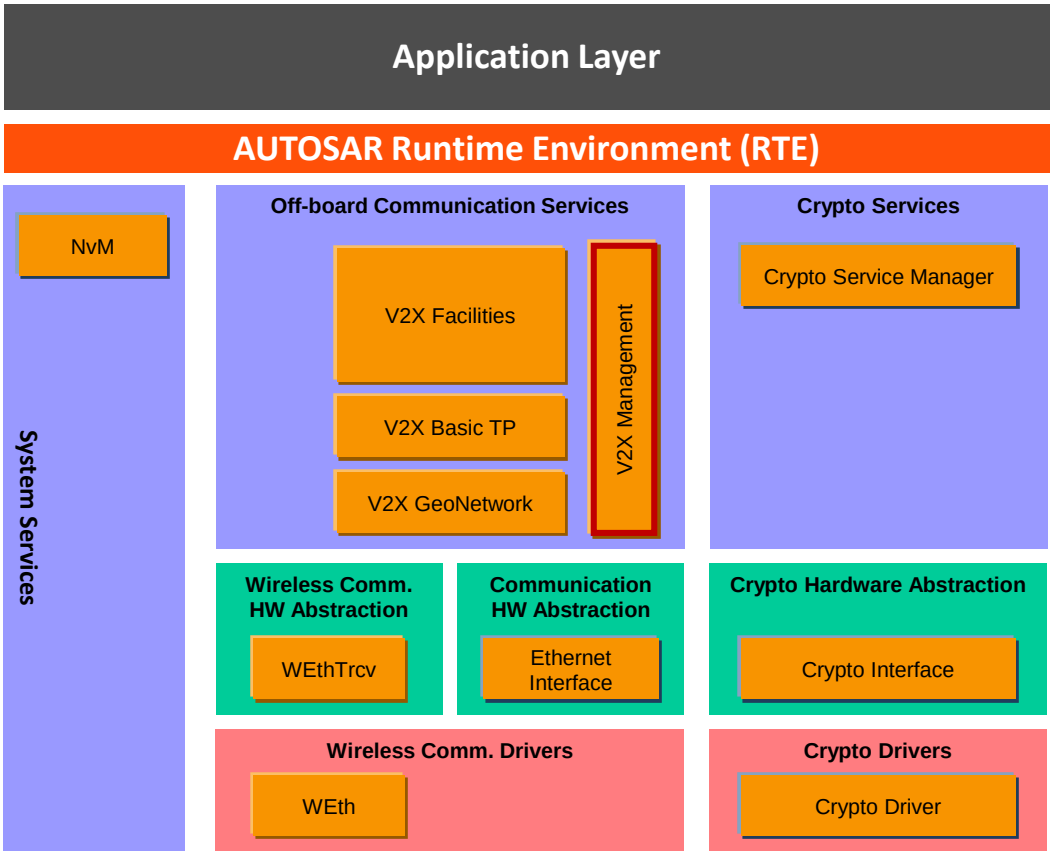

Figure 1-1 – AUTOSAR BSW architecture - V2xM scope

1.2 Functional overview

The V2xM module manages the operation of the V2X protocol stack. It does support the V2X protocol stack modules with a number of services and furthermore provide

some Application interfaces to let applications control the V2X-Stack within the limited range that the ETSI/C2C-CC Requirements left for applications..

1.2.1 Position and Time management (POTI)

Within the AUTOSAR architecture, the POTI service is a V2X Application within the Application layer. The V2xM module takes positional information from the POTI service and makes is available to the V2xFac and V2xGn modules [6].

1.2.2 Identity

A V2X Station has one identity that is used by every V2X module, that uses identity in its header information. For security and privacy reasons, the identity changes over time and travel distance. All modules that are using the identity shall be notified.

1.2.3 Security

V2xM provides standardized security services to the V2X-Stack according to ETSI specification, this includes signing and verification of messages as described in [7]. The APIs shall be implemented using CSM services provided by AUTOSAR.

1.2.4 Decentralized Congestion Control (DCC)

V2xM provides congestion control services for the V2X Stack, to provide the current V2X radio congestion state for a specific channel.

2 Acronyms and abbreviations

Abbreviation / Acronym:	*Description:*
DEM	Diagnostic Event Manager
DET	Default Error Tracer
API	Application Programming Interface
BSW	Basic Software
BTP	Basic Transport Protocol
CAM	Cooperative Awareness Message
DCC	Decentralized Congestion Control
DENM	Decentralized Environmental Notification Messages
EcuM	Electronic Control Unit Manager
ITS	Intelligent Transport System
LTC	Long Term Certificate
POTI	Position and Time management
VOD	Verification on Demand

3 Related documentation

3.1 Input documents

[1] AUTOSAR Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[2] AUTOSAR General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[3] AUTOSAR General Specification for Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[4] Glossary
AUTOSAR_TR_Glossary

[5] Intelligent Transport Systems (ITS); Communications Architecture
ETSI EN 302 665 V1.1.1 (2010-09)

[6] Intelligent Transport System (ITS); Facilities layer function;
Part 3: Position and time facility specification"
ETSI TS 102 890-3

[7] Intelligent Transport Systems (ITS); OSI cross-layer topics; Part 8: Interface between security entity and network and transport layer
ETSI TS 102 723-8

[8] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer.pdf

[9] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[10] Specification of Module Vehicle-2-X Facilities
AUTOSAR_SWS_ V2XFacilities.pdf

[11] Specification of Module Vehicle-2-X Basic Transport
AUTOSAR_SWS_ V2XBasicTransport.pdf

[12] Specification of Ethernet Interface
AUTOSAR_SWS_EthernetInterface.pdf

[13] Specification of Crypto Service Manager
AUTOSAR_SWS_CryptoServiceManager.pdf

[14] Specification of Vehicle-2-X Geo Networking
AUTOSAR_SWS_V2XGeoNetworking.pdf

[15] Specification of Vehicle-2-X Facilities
AUTOSAR_SWS_V2XFacilities.pdf

[16] Specification of Vehicle-2-X Basic Transport
AUTOSAR_SWS_V2XBasicTransport.pdf

[17] Specification of Module NVRAM Manager
AUTOSAR_SWS_ NVRAMManager.pdf

3.2 Related standards and norms

[18] IEC 7498-1 The Basic Model, IEC Norm, 1994

[19] ETSI TS 103 097 V1.2.1 (2015-06)

[20] C2C-CC BSP Requirement
C2CCC_RS_2037_BSP_Requirements.docx

[21] ETSI TS 102 894-2 V1.2.1 (2014-09)

[22] ETSI EN 302 637-2 V1.3.2 (2014-11)

[23] ETSI EN 302 636-4-1 V1.2.1 (2014-07)

[24] ETSI EN 302 663 – V1.2.1 (2013-07)

[25] ETSI TS 102 723-8 V1.1.1 (2016-04)

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General) [3] which is also valid for V2xM.

Thus, the specification SWS BSW General [3] shall be considered as additional and required specification for V2xM.

4 Constraints and assumptions

4.1 Limitations

No limitations.

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 the V2xM module to other modules within the AUTOSAR basic software architecture. It outlines the modules that are required or optional for the realization of the V2xM module and the V2xM services that these modules use.

5.1 AUTOSAR DET (Default Error Tracer)

In development mode, the V2xM module reports errors through the Det_ReportError function of the DET Module, [8].

5.2 AUTOSAR EcuM (Ecu State Manager)

The EcuM [9] initializes the V2xM module.

5.3 AUTOSAR CSM (Cryptographic Service Manager)

The CSM module is used for cryptographic calculations, needed by the V2X-Stack to secure packets. Therefore, sign and verify and other services of the CSM are being used.

5.4 AUTOSAR NvM (NVRAMManager)

The NvM [17] is used by V2xM to load certificates used for pseudonyms, signature generation and verification of V2X messages. Furthermore, the last ignition-time (startup-time of the v2x stack) is stored and loaded by NvM.

5.5 AUTOSAR Math libraries (Mfl, Efx)

For mathematical calculations, the Mfl or the Efx library is needed.

5.6 File structure

5.6.1 Code file structure

For details refer to the chapter 5.1.6 “Code file structure” in *SWS_BSWGeneral* [3].

6 Requirements traceability

Note:
Requirement IDs within this document have an encoding to state where each requirement has its origin:

- SWS items starting with a leading 0 (SWS_V2xM_0xxxx) are module specific and not inherited.
- SWS items starting with a leading 2 (SWS_V2xM_2xxxx) are inherited from C2C-CC Basic System Profile

Requirement	Description	Satisfied by
SRS_BSW_00345	BSW Modules shall support pre-compile configuration	SWS_V2xM_00191
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_V2xM_00118
SRS_BSW_00457	Callback functions of Application software components shall be invoked by the Basis SW	SWS_V2xM_00163
SRS_V2X_00010	The implementation of the V2X system shall follow additional guidance given by C2C-CC requirements	SWS_V2xM_20177, SWS_V2xM_20179, SWS_V2xM_20182, SWS_V2xM_20183, SWS_V2xM_20191, SWS_V2xM_20192, SWS_V2xM_20402, SWS_V2xM_20409
SRS_V2X_00163	The "verification" of a message shall comprise at least cryptographic verification of the message's signature	SWS_V2xM_00130, SWS_V2xM_20170, SWS_V2xM_20173
SRS_V2X_00174	The V2X system shall support key origin authentication for the new (long-term or pseudonym) public keys that are provided in certificate signing requests	SWS_V2xM_20173, SWS_V2xM_20180, SWS_V2xM_20403, SWS_V2xM_20411
SRS_V2X_00176	The V2X system shall change pseudonyms	SWS_V2xM_20410
SRS_V2X_00184	The V2X system shall allow applications to block the pseudonym change	SWS_V2xM_00005, SWS_V2xM_00099
SRS_V2X_00190	The V2X system shall handle vehicle states in a consistent manner	SWS_V2xM_00095
SRS_V2X_00193	The V2X system shall use ITS time as time base	SWS_V2xM_00126
SRS_V2X_00239	The V2X DCC mechanism shall be configurable	SWS_V2xM_20239
SRS_V2X_00279	The V2X system shall support circular, rectangular and ellipsoidal geographical areas	SWS_V2xM_00113
SRS_V2X_00280	The V2X system shall use high-accuracy methods to calculate the distance between two coordinates	SWS_V2xM_00176, SWS_V2xM_00177

SRS_V2X_00322	The V2X system shall provide services to avoid channel congestion of the shared media	SWS_V2xM_00188, SWS_V2xM_00189, SWS_V2xM_20238, SWS_V2xM_20239, SWS_V2xM_20240
SRS_V2X_00401	-	SWS_V2xM_00035
SRS_V2X_00406	The end-to-end security envelope shall be generated depending on the message type	SWS_V2xM_00038, SWS_V2xM_00074, SWS_V2xM_00135
SRS_V2X_00407	The signature in the end-to-end security envelope shall be generated using a private key corresponding to a valid authorization ticket (pseudonym certificate)	SWS_V2xM_00074, SWS_V2xM_00135
SRS_V2X_00412	The V2X system shall inform the driver about the expiration of the pseudonym certificates	SWS_V2xM_00095
SRS_V2X_00413	The V2X system shall inform the driver about the expiration of the Long Term Certificates	SWS_V2xM_00095
SRS_V2X_00711	The V2X system's CA basic service shall be compliant to ETSI Specification of Cooperative Awareness Basic Service	SWS_V2xM_20293

7 Functional specification

7.1 Startup behavior

[SWS_V2xM_00001] ⌈
The function V2xM_Init (refer to chapter 8.3.1) of the V2xM shall initialize the internal states of the V2xM module.
⌋ ()

[SWS_V2xM_00196] ⌈
The function V2xM_Init of the V2xM shall initialize the underlying MCAL/ECUAL modules WEth and WEthTrcv with a call to EthIf_SetControllerMode with the respective configured EthIfController V2xMEthIfCtrlRef.
⌋ ()

[SWS_V2xM_00197] ⌈
The Ethernet State Manager (EthSm) shall not be involved in the startup of the wireless communication stack.
⌋ ()

Note: See Figure 9-5 for the initialization of the wireless communication stack MCAL/ECUAL modules.

7.2 Shutdown behavior

[SWS_V2xM_00198] ⌈
The Wireless Communication is active until the ECU hardware is being shut down or reset. There are no means to stop the Vehicle-2-X wireless communication in advance.
⌋ ()

7.3 Identity management

[SWS_V2xM_00004] ⌈
The V2xM module shall implement the identity management, also known as the pseudonym. Specific V2X modules shall be notified with the current identity to ensure a consistent value is used in each layer of the V2X Stack.
⌋ ( )

[SWS_V2xM_20182] ⌈
The V2xM module shall change all addresses and identifiers of other layers transmitted over the wireless communication media (such as StationId in CAM/DENM, GeoNetworking Source Address, MAC Source Address) when the used pseudonym changes. Those changes are necessary to ensure the privacy of the user. ⌋ (SRS_V2X_00010)

Note: In V2xFac, the identity is represented in the Station Id, in V2xGn the identity is represented in the GeoNetworking address, in the Wireless Ethernet Driver the identity is represented in the MAC address.

[SWS_V2xM_20183] ⌈
All identifiers according to **[SWS_V2xM_20182]** (MAC Source Address, StationId in CAM/DENM, GN Source Address) shall be derived from the pseudonym ID. The required number of least significant bytes of the pseudonym ID shall be used as respective identifier. ⌋ (SRS_V2X_00010)

[SWS_V2xM_00005] ⌈
The V2xM module shall provide a mechanism to permit V2X modules to inhibit the identity change for a duration of maximum 15 minutes (e.g. during DENM event) via an API call to V2xM_LockPseudonymChange.
⌋ (SRS_V2X_00184)

[SWS_V2xM_00099] ⌈
The V2xM shall not inhibit an identity change when the pseudonym identity expires (i.e. when the certificate that provides the current pseudonym expires within the period where the identity change inhibit was requested). ⌋ (SRS_V2X_00184)

[SWS_V2xM_00006] ⌈
The function V2xM_Init shall initialize the identity management and provide an initial identity to the V2X protocol stack modules. ⌋ ( )

[SWS_V2xM_20410] ⌈
The V2xM identity management shall initiate a change of identity at minimum of 10 minutes and a maximum of 30 minutes randomly. ⌋ (SRS_V2X_00176)

[SWS_V2xM_20180] ⌈
V2xM shall use the pseudonym validity periods as defined by the Pseudonym CA in conformance to the rules of the Root CA. ⌋ (SRS_V2X_00174)

[SWS_V2xM_20411] ⌈
In case that an V2xM module has no valid pseudonym certificates for signing messages, it shall stop transmitting messages that use the security profiles specified in [19], clause 7.1, clause 7.2, and clause 7.3. ⌋ (SRS_V2X_00174)

[SWS_V2xM_00008] ⌈
The V2xM_MainFunction shall be used to initiate a change of the identity. ⌋ ( )

Note: The V2xM_MainFunction can also be used for software implementation specific execution of cyclic tasks.

[SWS_V2xM_00100] ⌈
The V2xM shall initiate a change of the pseudonym within two phases. A first prepare phase and a second commit or abort phase. The second phase depends on the result of all called modules within the first phase. If the first phase was successful,

the commit phase shall be initiated, if the first phase was unsuccessful, the abort phase shall be initiated. ⌋ ( )

[SWS_V2xM_00101] ⌈
In the prepare phase, the desired API <Module>_PreparePseudonymChange() shall be called. ⌋ ( )

[SWS_V2xM_00102] ⌈
In the commit phase, the desired API <Module>_CommitPseudonymChange() shall be called. ⌋ ( )

[SWS_V2xM_00103] ⌈
In the abort phase, the desired API <Module>_AbortPseudonymChange() shall be called. ⌋ ( )

[SWS_V2xM_00104] ⌈
The modules that shall be notified with the two phase pseudonym change by V2xM are V2xGn and V2xFac. ⌋ ( )

[SWS_V2xM_00105] ⌈
The EthernetInterface and the Wireless Ethernet Driver do not support a two phase id change. Within the commit phase of the two phase pseudonym change, the API EthIf_SetPhysAddr shall be called to initiate the pseudonym change within the Wireless Ethernet Driver. ⌋ ( )

[SWS_V2xM_20403] ⌈
The maximum amount of pseudonyms per week shall be 20. ⌋ (SRS_V2X_00174)

NOTE: In case of 20 pseudonyms: For each year about 1040 pseudonyms are necessary. Within the overlapping phase, the number of valid pseudonyms could be up to 40.

[SWS_V2xM_20177] ⌈
The pseudonym used by the V2xM module shall change every time when the vehicle’s ignition is switched on except if the system gets restarted within a period of 10 minutes, the pseudonym shall not be changed. ⌋ (SRS_V2X_00010)

[SWS_V2xM_20409] ⌈
The pseudonym change after turning on ignition shall be performed within a grace period of 1 minute. ⌋ (SRS_V2X_00010)

[SWS_V2xM_20179] ⌈
Pseudonyms may be reused within their validity period. ⌋ (SRS_V2X_00010)

[SWS_V2xM_20402] ⌈
The pseudonym validity periods shall not be longer than one week + overlapping period. ⌋ (SRS_V2X_00010)

7.4 Security

[SWS_V2xM_00009] ⌈
The V2xM module shall provide the Encap and Decap services required by V2xGn and Verification On Demand (VOD) by utilizing CSM.
⌋ ( )

[SWS_V2xM_00175] ⌈
The V2xM shall disable CAM generation in case of unusable position (e.g. due to no position available, degenerated dead reckoning, time jitter/drift). This is done via a call to V2xFac_V2xM_SetCaBsOperation. ⌋ ( )

[SWS_V2xM_20170] ⌈
The V2xM module shall use for sending messages digital signatures and certificates based on ECDSA-256 using the elliptic curve NIST P-256 algorithm as defined in [19]. ⌋ (SRS_V2X_00163)

[SWS_V2xM_20173] ⌈
The V2xM module shall support key origin authentication via the creation of a signature over internally generated public key(s), where public keys for LTCs shall be signed with the module private key and public keys for PCs shall be signed with a previously registered LTC private key. ⌋ (SRS_V2X_00163, SRS_V2X_00174)
Note: The "module private key" is a vehicle specific unique private key that could be generated randomly inside the HSM when the ECU is initialized in the first place

[SWS_V2xM_00135] ⌈
The function V2xM_V2xGn_ReqEncap shall encapsulate the payload of the GeoNetworking packet to be sent as defined in [25] and [19].⌋ (SRS_V2X_00406, SRS_V2X_00407)

[SWS_V2xM_00136] ⌈
The function V2xM_V2xGn_ReqDecap shall decapsulate the payload of a received GeoNetworking packet as defined in [25] and [19].⌋ ( )

[SWS_V2xM_00130] ⌈
The function V2xM_V2xGn_ReqDecap shall invoke CSM APIs for the verification of the data given by SecuredDataPtr ⌋ (SRS_V2X_00163)

7.5 Position and Time

[SWS_V2xM_20191] ⌈
WGS 84 shall be used as the reference coordinate system as defined in [21]. Altitude information shall be interpreted as height above WGS84 Ellipsoid. ⌋ (SRS_V2X_00010)

[SWS_V2xM_20192] ⌈

Heading shall be interpreted as the direction of the horizontal velocity vector. The starting point of the velocity vector shall be the ITS Vehicle Reference Point as defined in CAM specification [22] B.19 ⌋ (SRS_V2X_00010)

[SWS_V2xM_00121] ⌈
The function V2xM_GetPositionAndTime shall provide the currently known position and time information. ⌋ ( )

[SWS_V2xM_00126] ⌈
The function V2xM_GetRefTimePtr shall provide an address pointer to 32 bit data containing the current V2X Time, i.e. the TAI milliseconds from 2004-01-01 00:00:00.000 modulo 2^32. ⌋ (SRS_V2X_00193)

[SWS_V2xM_00177] ⌈
The function V2xM_CalcDistance shall calculate the distance between two geographical points. ⌋ (SRS_V2X_00280)

[SWS_V2xM_00179] ⌈
The function V2xM_CalcHeadingTolerance shall calculate if the difference of two heading values are within a given tolerance value. ⌋ ( )

7.6 DCC Management

[SWS_V2xM_20240] ⌈
The V2xM module shall use the channel load measurement mechanism specified in “C2C-CC White Paper Decentralized Congestion Control (DCC) for Day One”. In particular:

- The DCC-CCA *Threshold* (S_{th}) shall be set to -85 dBm
- The channel load, *channelLoad*(S_{th}), shall be calculated as the number of channel probe samples for which the received signal strength was larger than Sth (i.e., when the channel is "busy") divided by the total number of samples that were probed. In formal terms
- The probing sample duration T_p shall be set to 8 µs, i.e., one data symbol.
- The channel load measurement, i.e. *channelLoad*(S_{th}),value shall be provided by the MAC layer synchronously every Measuring interval T_m, with T_m= *100 ms*
- The minimum receiver sensitivity for the modulation and coding schemes supported shall be *-88 dBm*. This includes a sensitivity variation margin (factoring in temperature, production, implementation and aging losses) of *3 dB*.
- The DCC Mechanism shall perform a smoothing of the reported channel load values The following filter shall be used to smooth out the channel load value (i.e., CL in **[SWS_V2xM_20238]**) that is used to control the state transitions in DCC:

CL_{now} = a**channelLoad*(S_{th})(t)+ß*CL(t-1)+y*CL(t-2), where a=ß=0.5, and y=0. ⌋ (SRS_V2X_00322)

[SWS_V2xM_20238] ⌈

The V2xM module shall use the DCC Mechanism specified in “C2C-CC White Paper Decentralized Congestion Control (DCC) for Day One.” and summarized in the table. The DCC Mechanism is located at the DCC Access Sub-layer.

DCC States	*ChannelLoad,CL (%)*	*Transmission Interval, TTX [ms]*	*Message Rate, R [Hz]*
Relaxed	***0% ⩽ CL< 19%***	***60***	***16,7***
Active_1	***19% ⩽ CL < 27%***	***100***	***10***
Active_2	***27% ⩽ CL < 35%***	***180***	***5,6***
Active_3	***35% ⩽ CL < 43%***	***260***	***3,8***
Active_4	***43% ⩽ CL < 51%***	***340***	***2,9***
Active_5	***51% ⩽ CL < 59%***	***420***	***2,4***
Restricted	***CL ⩾ 59%***	***460***	***2,2***

] (SRS_V2X_00322)

[SWS_V2xM_20239] [
The V2xM module shall implement the DCC State Machine with the parameter set defined in this table. Wireless Ethernet Transceiver specific parameters (like the maximum output power or the packet transmission interval) shall be set with respective calls of the EthIf wrapper APIs.

Parameter	***Meaning***	***Default***
P_{Tx}	Transmission power	There is no default value set in this document. For each system the default TX power will depend on what is needed to fulfill the minimum communication range requirement
P_{Toll}	Transmission power across all states and DPs when in *toll communication mode* (see NOTE 3 in [14])	P_{Toll} = 10 dBm
T_{up}	Time of sustained channel load that triggers transition to a more restrictive state	~~T_{up} = 5 s conflicting with ETSI~~ T_{up}= *1 s*
T_{down}	Time of sustained channel load that triggers transition to a less restrictive state	~~T_{down} = 1 s conflicting with ETSI~~ T_{down} = *5 s*
T_{TX_MAX}	Maximum transmission interval for all states	T_{TX_MAX} = 460 ms
T_{TX_MIN}	Minimum transmission interval for DP1-DP3 NOTE: minimum transmission interval for DP0 is 50 ms	T_{TX_MIN} = 60 ms
R_{Burst}	Maximum message rate of message bursts (additionaly to rate of DP1-DP3)	R_{Burst} = 20 messages per second
T_{Burst}	Time period over which message burst is measured	T_{Burst} = 1 second

	T_{Burst} seconds is allowed very $T_{WaitBurst}$ seconds.	
$T_{BurstPeriod}$	Time period in which one burst is allowed.	$T_{BurstPeriod}$ = 10 seconds
$R_{max\_relaxed}$	Maximum message rate in *relaxed* state	$R_{max\_relaxed}$=36,7messages/second
$R_{max\_active,k}$	Maximum message rate in *active* sub-states	The inverse of the transmission interval for each CL value. *k*=1..*n*
$R_{max\_restrictive}$	Maximum message rate in *restrictive* sub-states	See Table 4.
CL_{max}	Transition threshold between *active* and *restrictive* states	CL_{max} = *59%*
CL_{min}	Transition threshold between *relaxed* and *active* states	CL_{min} = *19%*
CL_{active_k} ,*k= 1…n*	Transition threshold between active states	CL_{active_k} , *k= 1…n*
t_j , *j=1..m*	*relaxed* (sub-)states transmission interval values as per Table 4	*m* = 1, see Table in **[SWS_V2xM_20238]**
t_k , *k=1..n*	*active* (sub-)states transmission interval values as per Table 4	*n* = 5, see Table in **[SWS_V2xM_20238]**
t_l , *l=1..q*	*restrictive* (sub-)states transmission interval values as per Table 4	*q* = 1, see Table in **[SWS_V2xM_20238]**
n	Number of *active* sub-states	*n*=5
q	Number of *restrictive (*sub-)states	*n*=1
m	Number of *relaxed (*sub-)states	*m=1*
MCS	Modulation an Coding Scheme	6 Mbps QPSK ½ for all states and DP values in Table **[SWS_V2xM_20238]**
α ,β,γ	Channel Load smoothing parameters	Default values are *α*=*β*=0.5, γ=0
$S_{th,}$ $N_{p,}$ T_{m} , T_p	Channel Load estimation parameters	Default values are T_m= 100 ms, T_p = 8 µs, N_p =12 500, and S_{th} = -85 dBm

] (SRS_V2X_00322, SRS_V2X_00239)

[SWS_V2xM_20293] [
The parameter T_GenCam_Dcc (see [2]) shall be set periodically to the value of T_{TX} of the current state v and pushed to the V2xFac module via the V2xFac_V2xM_SetTGenCamDcc API.] (SRS_V2X_00711)

[SWS_V2xM_00188] [
The current state (restrictive, active sub-state, relaxed, see [SWS_V2xM_20238]) shall be set periodically to the WEthTrcv Module to allow message bursts within the relaxed state.] (SRS_V2X_00322)

[SWS_V2xM_00189] ⌈
The current transmission interval (see [SWS_V2xM_20238]) shall be set periodically to the WEthTrcv Module to allow triggering of transmit queues. ⌋ (SRS_V2X_00322)

7.7 Error classification

7.7.1 Development Errors

[SWS_ V2xM_00031]
⌈

Type of error	*Related error code*	*Value [hex]*
API service called with wrong parameter	V2XM_E_PARAM	0x01
API service called with invalid pointer	V2XM_E_PARAM_POINTER	0x02
V2xM initialization failed	V2XM_E_INIT_FAILED	0x03
API function called before the V2xM module has been fully initialized	V2XM_E_UNINIT	0x04

⌋ ( )

7.7.2 Runtime Errors

There are no runtime errors.

7.7.3 Transient Faults

There are no transient faults.

7.7.4 Production Errors

There are no production errors.

7.7.5 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 modules are listed:

[SWS_V2xM_00033] ⌈

Module	*Header File*	*Imported Type*
Csm	<none>	Crypto_VerifyResultType
	Rte_Csm_Type.h	Crypto_OperationModeType
Gpt	Gpt.h	Gpt_ChannelType
	Gpt.h	Gpt_PredefTimerType
	Gpt.h	Gpt_ValueType
NvM	Rte_NvM_Type.h	NvM_BlockIdType
	Rte_NvM_Type.h	NvM_RequestResultType
StbM	Rte_StbM_Type.h	StbM_SynchronizedTimeBaseType
	Rte_StbM_Type.h	StbM_TimeStampExtendedType
	Rte_StbM_Type.h	StbM_TimeStampType
	Rte_StbM_Type.h	StbM_UserDataType
Std_Types	StandardTypes.h	Std_ReturnType
	StandardTypes.h	Std_VersionInfoType
V2x_GeneralTypes	Rte_V2xM_Type.h	V2xM_PositionAndTimeType
	V2x_GeneralTypes.h	V2x_ChanType
	V2x_GeneralTypes.h	V2x_PseudonymType
	V2x_GeneralTypes.h	V2x_SecProfileType
	V2x_GeneralTypes.h	V2x_SecReportType
	V2x_GeneralTypes.h	V2x_SecReturnType
WEth_GeneralTypes	WEth_GeneralTypes.h	WEthTrcv_GetChanRxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetChanRxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetChanTxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetRadioParamIdType

⌋ ()

8.2 Type definitions

[SWS_V2xM_00107] ⌈
V2xM.h shall include V2x_GeneralTypes.h for the inclusion of general V2X type declarations. ⌋ ()

8.2.1 V2xM_ConfigType

[SWS_V2xM_00110] ⌈

Name:	`V2xM_ConfigType`	
Type:	`Structure`	
Range:	`implementation specific`	The content of the configuration data structure is implementation specific.
Description:	Configuration data structure of the V2xM module.	
Available via:	`V2xM.h`	

] ()

8.2.2 V2x_GnPacketTransportType

[SWS_V2xM_00034] [

Name:	V2x_GnPacketTransportType		
Type:	Enumeration		
Range:	V2X_GN_GEOUNICAST	0x00	--
	V2X_GN_GEOANYCAST	0x01	--
	V2X_GN_GEOBROADCAST	0x02	--
	V2X_GN_TSB	0x03	--
	V2X_GN_SHB	0x04	--
Description:	Specifies the packet transport type for GeoNetworking packages. This is passed e.g. via V2xFac and V2xBtp for the transmit path.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.3 V2x_GnDestinationType

[SWS_V2xM_00112] [

Name:	V2x_GnDestinationType		
Type:	Enumeration		
Range:	V2X_GN_DESTINATION_ADDRESS	0x00	--
	V2X_GN_DESTINATION_AREA	0x01	--
Description:	Specifies the destination type for GeoNetworking packages. This is passed e.g. via V2xFac and V2xBtp for the transmit path.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.4 V2x_GnAddressType

[SWS_V2xM_00035] [

Name:	V2x_GnAddressType
Type:	uint64
Description:	The GeoNetworking address.
Available via:	V2x_GeneralTypes.h

] (SRS_V2X_00401)

8.2.5 V2x_GnAreaShapeType

[SWS_V2xM_00113] [

Name:	V2x_GnAreaShapeType		
Type:	Enumeration		
Range:	V2X_GN_SHAPE_CIRCLE	0x00	--
	V2X_GN_SHAPE_RECT	0x01	--
	V2X_GN_SHAPE_ELLIPSE	0x02	--
Description:	Specifies the shape type for GeoNetworking Areas.		
Available via:	V2x_GeneralTypes.h		

] (SRS_V2X_00279)

8.2.6 V2x_GnDestinationAreaType

[SWS_V2xM_00036] ⌈

Name:	V2x_GnDestinationAreaType		
Type:	Structure		
Element:	sint32	latitude	Latitude [1/10 microdegree]
	sint32	longitude	Longitude [1/10 microdegree]
	uint16	distanceA	Distance a of the geometric shape [meters]
	uint16	distanceB	Distance b of the geometric shape [meters]
	uint16	angle	Angle of the geometric shape [degrees from North]
	V2x_GnAreaShapeType	shape	Shape type of the geometric area
Description:	Definition of the GeoNetworking destination area		
Available via:	V2x_GeneralTypes.h		

⌋ ()

8.2.7 V2x_GnTxResultType

[SWS_V2xM_00114] ⌈

Name:	V2x_GnTxResultType		
Type:	Enumeration		
Range:	V2X_GNTX_ACCEPTED	--	GeoNetworking transmit has been accepted
	V2X_GNTX_E_MAXSDUSIZEOVFL	--	GeoNetworking transmit has been rejected due to maximum length exceedance
	V2X_GNTX_E_MAXPACKETLIFETIME	--	GeoNetworking transmit has been rejected due to maximum lifetime exceedance
	V2X_GNTX_E_TCID	--	GeoNetworking transmit has been rejected due to unsupported Traffic Class ID
	V2X_GNTX_E_MAXGEOAREASIZE	--	GeoNetworking transmit has been rejected due to GeoArea exceeds max size
	V2X_GNTX_E_UNSPECIFIED	--	GeoNetworking transmit has been rejected due to unspecified reasons
Description:	The result code used to specify if a V2xGn_Transmit has been processed successfully.		
Available via:	V2x_GeneralTypes.h		

⌋ ()

8.2.8 V2x_SecProfileType

[SWS_V2xM_00038] ⌈

Name:	V2x_SecProfileType		
Type:	Enumeration		
Range:	V2X_SECPROF_CAM	--	Cam Security Profile
	V2X_SECPROF_DENM	--	Denm Security Profile
	V2X_SECPROF_OTHER_SIGNED	--	Security Profile for other message types that have to be signed
	V2X_SECPROF_OTHER_SIGNED_EXTERNAL	--	Security Profile for other message types that are signed externally

	V2X_SECPROF_OTHER_SIGNED_ENCRYPTED	--	Security Profile for other message types that have to be signed and encrypted
Description:	Used to describe the security service invoked by V2xM		
Available via:	V2x_GeneralTypes.h		

] (SRS_V2X_00406)

8.2.9 V2x_SecReturnType

[SWS_V2xM_00115] ⌈

Name:	V2x_SecReturnType		
Type:	Enumeration		
Range:	V2X_E_OK	--	Return with success
	V2X_E_NOT_OK	--	Failure during operation
	V2X_E_UNVERIFIED	--	Message has not been verified. Used for VoD
	V2X_E_BUF_OVFL	--	Destination buffer too small for security operation data output
Description:	Used for return values of security related functions		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.10 V2x_MaximumPacketLifetimeType

[SWS_V2xM_00039] ⌈

Name:	V2x_MaximumPacketLifetimeType		
Type:	uint16		
Range:	0..6300	--	Valid values
	6301..uint16 Max Value	--	Invalid
Description:	Specifies the maximum tolerable time (in seconds) a GeoNetworking packet can be buffered.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.11 V2x_TrafficClassIdType

[SWS_V2xM_00043] ⌈

Name:	V2x_TrafficClassIdType		
Type:	uint8		
Range:	0..63	--	Valid values
	64..uint8 Max Value	--	Invalid
Description:	Requirements on packet transport coming from ITS Facilities layer		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.12 V2x_ChanType

[SWS_V2xM_00044] ⌈

Name:	V2x_ChanType

Type:	Enumeration		
Range:	V2X_SCH4	172	Service channel 4
	V2X_SCH3	174	Service channel 3
	V2X_SCH1	176	Service channel 1
	V2X_SCH2	178	Service channel 2
	V2X_CCH	180	Control channel
Description:	Specifies the channel type to use. Channels from ITS-G5A and ITS-G5B are used. Values matching IEEE 802.11-2012 channel numbers.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.13 V2x_GnUpperProtocolType

[SWS_V2xM_00045] [

Name:	V2x_GnUpperProtocolType		
Type:	Enumeration		
Range:	V2X_ANY	--	Unspecified
	V2X_BTPA	--	Transport protocol: BTP-A (for interactive packet transport).
	V2X_BTPB	--	Transport protocol: BTP-B (for non-interactive packet transport).
	V2X_IPV6	--	IPv6 header
Description:	Specifies the GeoNetworking payload.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.14 V2x_GnLongPositionVectorType

[SWS_V2xM_00046] [

Name:	V2x_GnLongPositionVectorType		
Type:	Structure		
Element:	V2x_GnAddressType	gnAddress	GeoNetworking Address
	uint32	timestamp	Timestamp [ms]
	sint32	latitude	Latitude [1/10 microdegree]
	sint32	longitude	Longitude [1/10 microdegree]
	boolean	pai	Positional accuracy indicator
	sint16	speed	Speed [1/100 m/s]
	uint16	heading	Heading [1/10 degrees]
Description:	Position-related information as defined within [23] chapter 8.5.2.		
Available via:	V2x_GeneralTypes.h		

] ()

8.2.15 V2x_PseudonymType

[SWS_V2xGn_00057] [

Name:	V2x_PseudonymType
Type:	uint64
Description:	Pseudonym, derived from Pseudonym Certificates. The pseudonym is distributed to different modules to support privacy within the V2X System to the outside world.
Available via:	V2x_GeneralTypes.h

] ()

8.3 Function definitions

This is a list of functions provided for upper layer modules and other V2X stack modules.

8.3.1 V2xM_Init

[SWS_V2xM_00070] ⌈

Service name:	V2xM_Init	

Syntax:	`void V2xM_Init(`
	`  const void * 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:	Initializes the V2xM module.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00116] ⌈
The function shall store the access to the configuration structure for subsequent API calls. ⌋ ( )

[SWS_V2xM_00118] ⌈
The Configuration pointer configPtr shall always have a NULL_PTR value⌋ (SRS_BSW_00414)

8.3.2 V2xM_GetVersionInfo

[SWS_V2xM_00071] ⌈

Service name:	V2xM_GetVersionInfo	

Syntax:	`void V2xM_GetVersionInfo(`
	`  Std_VersionInfoType* VersionInfoPtr`
	`)`	

Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	VersionInfoPtr	Pointer to store the version information of this module.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Provides the version information of this module.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00120] ⌈
If development error detection is enabled: the function shall check the parameter VersionInfoPtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER. ⌋ ( )

8.3.3 V2xM_GetPositionAndTime

[SWS_V2xM_00072] ⌈

Service name:	V2xM_GetPositionAndTime	
Syntax:	`Std_ReturnType V2xM_GetPositionAndTime( V2xM_PositionAndTimeType* Poti )`	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	Poti	Current position and time information including positional error information.
Return value:	Std_ReturnType	E_OK: request successful E_NOT_OK: Time and/or position not available.
Description:	Provides the instantaneous position information.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00122] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

[SWS_V2xM_00123] ⌈
If development error detection is enabled: the function shall check the parameter Poti for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

8.3.4 V2xM_GetRefTimePtr

[SWS_V2xM_00125] ⌈

Service name:	V2xM_GetRefTimePtr
Syntax:	`Std_ReturnType V2xM_GetRefTimePtr( const uint32** RefTimePtr )`
Service ID[hex]:	0x04
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None

Parameters (out):	RefTimePtr	Pointer to the current time information.
Return value:	Std_ReturnType	E_OK: request successful E_NOT_OK: request failed
Description:	Provides a pointer to the time reference of the V2X-Stack.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00127] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

[SWS_V2xM_00128] ⌈
If development error detection is enabled: the function shall check the parameter RefTimePtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

8.3.5 V2xM_V2xGn_ReqEncap

[SWS_V2xM_00074] ⌈

Service name:	V2xM_V2xGn_ReqEncap	

Syntax:	`V2x_SecReturnType V2xM_V2xGn_ReqEncap(`
	`  uint16 TransactionId16,`
	`  V2x_SecProfileType SecProfile,`
	`  uint16 UnsecuredDataLength,`
	`  const uint8* UnsecuredDataPtr,`
	`  uint16* SecuredDataLength,`
	`  uint8* SecuredDataPtr`
	`)`	

Service ID[hex]:	0x06	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId16	The request identifier that the client can use to match the response
	SecProfile	The security profile to use for encapsulation
	UnsecuredDataLength	The length of the data to use for encapsulation
	UnsecuredDataPtr	The pointer to the data to use for encapsulation
Parameters (inout):	SecuredDataLength	The length pointer containing the maximum length of secured data SecuredDataPtr at input direction. Shall contain the actual size of the secured data SecuredDataPtr at output direction.
	SecuredDataPtr	The pointer where the secured data shall be put.
Parameters (out):	None	
Return value:	V2x_SecReturnType	V2X_E_OK: request successful V2X_E_NOT_OK: request failed V2X_E_BUF_OVFL: SecuredDataLength is too small for security operation result data
Description:	This function is called by the V2xGn to sign and/or encrypt a message. An asynchronous V2xGn_V2xM_EncapConfirmation call will be used to notify V2xGn of the result.	
Available via:	`V2xM_V2xGn.h`	

] (SRS_V2X_00406, SRS_V2X_00407)

[SWS_V2xM_00131] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00132] ⌈
If development error detection is enabled: the function shall check the parameter UnsecuredDataPtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00133] ⌈
If development error detection is enabled: the function shall check the parameter SecuredDataLength for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00134] ⌈
If development error detection is enabled: the function shall check the parameter SecuredDataPtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

8.3.6 V2xM_V2xGn_ReqDecap

[SWS_V2xM_00075] ⌈

Service name:	V2xM_V2xGn_ReqDecap	

Syntax:	`V2x_SecReturnType V2xM_V2xGn_ReqDecap(`
	`  uint32 TransactionId32,`
	`  uint16 SecuredDataLength,`
	`  const uint8* SecuredDataPtr,`
	`  uint16* UnsecuredDataLength,`
	`  uint8* UnsecuredDataPtr,`
	`  V2x_SecReportType* SecReport,`
	`  uint64* CertificateId,`
	`  uint32* ItsAid,`
	`  uint8* SspLength,`
	`  uint8* SspBits`
	`)`	

Service ID[hex]:	0x07	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TransactionId32	Transaction Id of the received Packet
	SecuredDataLength	The length of the data to decrypt and verify
	SecuredDataPtr	The pointer to the data to decrypt and verify
Parameters (inout):	UnsecuredDataLength	The pointer to the data length of the unsecured data. Shall contain the maximum available length (incoming direction) and the actual used length (outgoing direction)
	UnsecuredDataPtr	The pointer where the decrypted /verified data shall be put

	SecReport	The security report.
	CertificateId	The identification of the used for verification (by certificate hash)
	ItsAid	The numerical value of the ITS-AID
	SspLength	The length (in octets, up to 31) of the SSP bits
	SspBits	The SSP bits
Parameters (out):	None	
Return value:	V2x_SecReturnType	V2X_E_OK: request successful V2X_E_NOT_OK: request failed V2X_E_UNVERIFIED: VOD is being used V2X_E_BUF_OVFL: UnsecuredDataLength is too small for security operation result data
Description:	This function is called by the V2xGn to decrypt and verify a message. An asynchronous V2xGn_V2xM_DecapConfirmation call will be used to notify V2xGn of the result.	
Available via:	`V2xM_V2xGn.h`	

] ()

[SWS_V2xM_00137] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00138] ⌈
If development error detection is enabled: the function shall check the parameter SecuredDataPtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00139] ⌈
If development error detection is enabled: the function shall check the parameter UnsecuredDataLength for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00140] ⌈
If development error detection is enabled: the function shall check the parameter UnsecuredDataPtr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00183] ⌈
If development error detection is enabled: the function shall check the parameter SecReport for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00184] ⌈
If development error detection is enabled: the function shall check the parameter CertificateId for being valid. If the check fails, the function shall raise the development

error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00185] ⌈
If development error detection is enabled: the function shall check the parameter ItsAid for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00186] ⌈
If development error detection is enabled: the function shall check the parameter SspLength for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

[SWS_V2xM_00187] ⌈
If development error detection is enabled: the function shall check the parameter Ssp for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return V2X_E_NOT_OK. ⌋ ( )

8.3.7 V2xM_TriggerPseudonymChange

[SWS_V2xM_00077] ⌈

Service name:	V2xM_TriggerPseudonymChange	
Syntax:	`Std_ReturnType V2xM_TriggerPseudonymChange( void )`	
Service ID[hex]:	0x08	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: request successful E_NOT_OK: request failed
Description:	This function is called by the V2xFac, V2xGn or another entity to change the Pseudonym used by the V2X-Stack, e.g. due to a GeoNetworking address conflict.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00142] ⌈
The function V2xM_TriggerPseudonymChange shall trigger the pseudonym change and update the identity of the V2X-Stack to the adjacent modules. ⌋ ( )

[SWS_V2xM_00143] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

[SWS_V2xM_00144][If the pseudonym change is locked E_NOT_OK shall be returned]()

8.3.8 V2xM_LockPseudonymChange

[SWS_V2xM_00078] [

Service name:	V2xM_LockPseudonymChange	

Syntax:	`Std_ReturnType V2xM_LockPseudonymChange(`
	`  uint16 Duration,`
	`  uint64* HandleId`
	`)`	

Service ID[hex]:	0x09	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Duration	Number of seconds to lock
Parameters (inout):	None	
Parameters (out):	HandleId	Handle to unlock manually
Return value:	Std_ReturnType	E_OK: request successful E_NOT_OK: request failed
Description:	This function is called by V2xGn or from the Application Service Interface to lock the pseudonym change.	
Available via:	`V2xM.h`	

] ()

[SWS_V2xM_00145] [
The function V2xM_LockPseudonymChange shall prevent the module from changing the pseudonym. The requirements from [20] shall apply.] ( )

[SWS_V2xM_00146] [
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.] ( )

[SWS_V2xM_00147] [
If development error detection is enabled: the function shall check the parameter HandleId for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.] ( )

8.3.9 V2xM_UnlockPseudonymChange

[SWS_V2xM_00079] [

Service name:	V2xM_UnlockPseudonymChange

Syntax:	`Std_ReturnType V2xM_UnlockPseudonymChange(`
	`  uint64 HandleId`
	`)`

Service ID[hex]:	0x0a

Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	HandleId	Handle to unlock manually, available from LockPseudonymChange function.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: request successful E_NOT_OK: request failed
Description:	This function is called by V2xGn or from the Application Service Interface to unlock the pseudonym change.	
Available via:	`V2xM.h`	

] ()

[SWS_V2xM_00149] ⌈
The function V2xM_UnlockPseudonymChange shall allow the module to change the pseudonym again.⌋ ( )

[SWS_V2xM_00150] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

[SWS_V2xM_00151] ⌈
If development error detection is enabled: the function shall check the parameter HandleId for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

8.3.10 V2xM_V2xGn_SetGlobalRxParams

[SWS_V2xM_00080] ⌈

Service name:	V2xM_V2xGn_SetGlobalRxParams	

Syntax:	`void V2xM_V2xGn_SetGlobalRxParams(`
	`  const uint16* Cbr_Gs,`
	`  const V2x_ChanType* Channel`
	`)`	

Service ID[hex]:	0x0b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	Cbr_Gs	List of current channel busy values
	Channel	List of channel types to that the busy values belong to
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	This function is called by V2xGn to set the current channel busy percentage for the specified channel	
Available via:	`V2xM_V2xGn.h`	

] ()

[SWS_V2xM_00154] ⌈

If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT. ⌋ ( )

[SWS_V2xM_00155] ⌈
If development error detection is enabled: the function shall check the parameter Cbr_G for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER.⌋ ( )

[SWS_V2xM_00156] ⌈
If development error detection is enabled: the function shall check the parameter Channel for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER. ⌋ ( )

8.3.11 V2xM_V2xGn_GetGlobalTxParams

[SWS_V2xM_00081] ⌈

Service name:	V2xM_V2xGn_GetGlobalTxParams	

Syntax:	`void V2xM_V2xGn_GetGlobalTxParams(`
	`  const V2x_ChanType* channel,`
	`  uint16* Cbr`
	`)`	

Service ID[hex]:	0x0c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	channel	List of channels
Parameters (inout):	None	
Parameters (out):	Cbr	List of current channel busy values (in tenths of a percent) for the specified channel type
Return value:	None	
Description:	This function is called by V2xGn to get the current channel busy percentage for the specified channel	
Available via:	`V2xM_V2xGn.h`	

⌋ ()

[SWS_V2xM_00158] ⌈
The function V2xM_V2xGn_GetGlobalTxParams shall change provide a list with CBR values for the specific list of channels. ⌋ ( )

[SWS_V2xM_00159] ⌈
If development error detection is enabled: the function shall check that the service V2xM_Init was previously called. If the check fails, the function shall raise the development error V2XM_E_UNINIT. ⌋ ( )

[SWS_V2xM_00160] ⌈
If development error detection is enabled: the function shall check the parameter Cbr for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER. ⌋ ( )

[SWS_V2xM_00161] ⌈

If development error detection is enabled: the function shall check the parameter Channel for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER. ⌋ ( )

8.3.12 V2xM_CalcDistance

[SWS_V2xM_00176] ⌈

Service name:	V2xM_CalcDistance	

Syntax:	`Std_ReturnType V2xM_CalcDistance(`
	`  sint32 LatitudeA,`
	`  sint32 LongitudeA,`
	`  sint32 LatitudeB,`
	`  sint32 LongitudeB,`
	`  float32* Distance`
	`)`	

Service ID[hex]:	0x0e	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	LatitudeA	Latitude of geographical point A
	LongitudeA	Longitude of geographical point A
	LatitudeB	Latitude of geographical point B
	LongitudeB	Longitude of geographical point B
Parameters (inout):	None	
Parameters (out):	Distance	Distance between geographical points A and B [m]
Return value:	Std_ReturnType	E_OK: Calculation ok E_NOT_OK: Calculation failed, input parameters out of range
Description:	Calculates the distance between two geographical points on earth with the assumption that they are on elevation 0.	
Available via:	`V2xM.h`	

⌋ (SRS_V2X_00280)

[SWS_V2xM_00181] ⌈
If development error detection is enabled: the function shall check the parameter Distance for being valid. If the check fails, the function shall raise the development error V2XM_E_PARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK. ⌋ ( )

8.3.13 V2xM_CalcHeadingInTolerance

[SWS_V2xM_00178] ⌈

Service name:	V2xM_CalcHeadingInTolerance	

Syntax:	`boolean V2xM_CalcHeadingInTolerance(`
	`  float32 Heading1,`
	`  float32 Heading2,`
	`  float32 Tolerance`
	`)`	

Service ID[hex]:	0x0f	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Heading1	First heading value
	Heading2	Second heading value

	Tolerance	Allowed tolerance between heading values
Parameters (inout):	None	
Parameters (out):	None	
Return value:	boolean	TRUE: diff of headings is within tolerance FALSE: diff of headings is outside tolerance
Description:	Calculates if difference of heading values are within a tolerance value	
Available via:	`V2xM.h`	

⌋ ()

8.3.14 V2xM_SetTollingZoneInformation

[SWS_V2xM_00182] ⌈

Service name:	V2xM_SetTollingZoneInformation	

Syntax:	`void V2xM_SetTollingZoneInformation(`
	`  sint32 protectedZoneLatitude,`
	`  sint32 protectedZoneLongitude,`
	`  uint32 cenDsrcTollingZoneID`
	`)`	

Service ID[hex]:	0x10	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	protectedZoneLatitude	Latitude of the tolling zone
	protectedZoneLongitude	Longitude of the tolling zone
	cenDsrcTollingZoneID	ID of the tolling zone
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Set available tolling zone information. This is done from V2xFac that receives this information via CAM messages.	
Available via:	`V2xM.h`	

⌋ ()

[SWS_V2xM_00190] ⌈
The V2xM Module shall check the provided positional information. In case of a distance less than one kilometer to the Tolling Zone, tolling zone power reduction shall be enabled. ⌋()

[SWS_V2xM_00170]⌈
Used for automatic handling of tolling zone power reduction.
If no tolling zone power reduction is currently active, nothing needs to be done.
Otherwise, in case of a distance less than one kilometer to the Tolling Zone position given by V2xM_SetTollingZoneInformation, the module shall push that to the WEthTrcv via the API EthIf_SetRadioParams so that WEthTrcv is able to reduce output power of specific packets. If the position drops out of the one kilometer range, tolling zone power reduction shall be switched off. ⌋()

8.4 Call-back notifications

8.4.1 CSM callback interfaces

[SWS_V2xM_00163] ⌈
If the V2xM module uses the Csm module asynchronously to calculate or verify the signatures, V2xM shall provide callback functions according to Csm_CallbackType.
⌋ (SRS_BSW_00457)

8.5 Scheduled functions

8.5.1 V2xM_MainFunction

[SWS_V2xM_00164] ⌈

Service name:	V2xM_MainFunction

Syntax:	`void V2xM_MainFunction(`
	`  void`
	`)`

Service ID[hex]:	0x0D
Description:	Scheduled MainFunction of V2xM
Available via:	`SchM_V2xM.h`

⌋ ()

[SWS_V2xM_00165] ⌈
Used for polling DCC information via EthIf_GetChanRxParamsAPI call from Wireless Ethernet Transceiver Driver. ⌋ ( )

[SWS_V2xM_00166] ⌈
Used for cyclic pseudonym change. ⌋ ( )

[SWS_V2xM_00167] ⌈
Used for pushing DCC information to adjacent V2X modules. ⌋ ( )

[SWS_V2xM_00168] ⌈
Used for polling state of asynchronous security functions of CSM. ⌋ ( )

[SWS_V2xM_00169] ⌈
Used for automatic unlocking of pseudonym changes if locking interval is due. ⌋ ( )

8.6 Expected Interfaces

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

8.6.1 Mandatory Interfaces

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

[SWS_V2xM_00092] ⌈

API function	*Header File*	*Description*
Csm_CertificateParse	Csm.h	This function shall dispatch the certificate parse function to the CRYIF.
Csm_CertificateVerify	Csm.h	Verifies the certificate stored in the key referenced by verifyKeyId with the certificate stored in the key referenced by keyId. Note: Only certificates stored in the same Crypto Driver can be verified against each other. If the key element CRYPTO_KE_CERTIFICATE_CURRENT_TIME is used for the verification of the validity period of the certificate indentified by verifyKeyId, it shall have the same format as the timestamp in the certificate.
Csm_Hash	Csm.h	Uses the given data to perform the hash calculation and stores the hash.
Csm_KeyElementGet	Csm.h	Retrieves the key element bytes from a specific key element of the key identified by the keyId and stores the key element in the memory location pointed by the key pointer.
Csm_KeyElementSet	Csm.h	Sets the given key element bytes to the key identified by keyId.
Csm_RandomGenerate	Csm.h	Generate a random number and stores it in the memory location pointed by the result pointer.
Csm_SignatureGenerate	Csm.h	Uses the given data to perform the signature calculation and stores the signature in the memory location pointed by the result pointer.
Csm_SignatureVerify	Csm.h	Verifies the given MAC by comparing if the signature is generated with the given data.
EthIf_GetChanRxParams	EthIf.h	Read values related to the receive direction of the transceiver. For example, this could be a Channel Busy Ratio (CBR) or the average Channel Idle Time (CIT).
EthIf_SetChanRxParams	EthIf.h	Set values related to the receive direction of a transceiver's wireless channel. For example, this could be a channel parameter like the frequency.
EthIf_SetChanTxParams	EthIf.h	Set values related to the transmit direction of a transceiver's wireless channel. For example, this could be the bitrate of a channel.
EthIf_SetPhysAddr	EthIf.h	Sets the physical source address used by the indexed controller.
EthIf_SetRadioParams	EthIf.h	Set values related to a transceiver's wireless radio. For example, this could be the selection of the radio settings (channel, ...).
NvM_GetErrorStatus	NvM.h	Service to read the block dependent error/status information.
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.
V2xFac_V2xM_AbortPseudonymChange	V2xFac_V2xM.h	This function is called by the V2xM when not all modules are OK with the pseudonym change and the change is to be rolled back.
V2xFac_V2xM_CommitPseudonymChange	V2xFac_V2xM.h	This function is called by the V2xM when all modules are OK with the pseudonym change and the change is to be committed.
V2xFac_V2xM_PreparePseudonymChange	V2xFac_V2xM.h	By this API primitive the V2xFac module gets an indication that the given Pseudonym and hereby the StationId is about to be changed
V2xFac_V2xM_SetCaBsOperation	V2xFac_V2xM.h	By this API primitive the V2xFac module gets an indication of the current operation state of the CA Basic Service.
V2xFac_V2xM_SetTGenCamDcc	V2xFac_V2xM.h	By this API primitive the V2xFac module gets an indication of the current TGenCamDcc value.
V2xGn_V2xM_AbortPseudonymChange	V2xGn_V2xM.h	This function is called by the V2xM when not all modules are OK with the pseudonym change and the change is to be rolled back.
V2xGn_V2xM_CommitPseudonymChange	V2xGn_V2xM.h	This function is called by the V2xM when all modules are OK with the pseudonym change and the change is to be committed.
V2xGn_V2xM_DecapConfirmation	V2xGn_V2xM.h	This function is called by the V2xM when a decapsulation has been finished.
V2xGn_V2xM_EncapConfirmation	V2xGn_V2xM.h	This function is called by the V2xM when an encapsulation has been finished.
V2xGn_V2xM_PreparePseudonymChange	V2xGn_V2xM.h	This function is called by the V2xM when a Pseudonym Change occurs to prepare the change in every module using it.

] ()

8.6.2 Optional Interfaces

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

[SWS_V2xM_00093] ⌈

API function	***Header File***	***Description***
Det_ReportError	Det.h	Service to report development errors.
Efx_ArcCos_s32_u32	Efx.h	This service computes the inverse cosine of a value.
Efx_ArcSin_s32_s32	Efx.h	This service computes the inverse sine of a value.
Efx_Cos_s32_s32	Efx.h	This service computes the cosine of an angle.
Efx_Sin_s32_s32	Efx.h	This service computes the sine of an angle.
Efx_Sqrt_u32_u32	Efx.h	This service computes the square root of a value
Gpt_GetPredefTimerValue	Gpt.h	Delivers the current value of the desired GPT Predef Timer.
Gpt_StartTimer	Gpt.h	Starts a timer channel.
Mfl_ArcCos_f32	Mfl.h	Returns the arc cosine of an angle, in the range of 0.0 through pi.
Mfl_ArcSin_f32	Mfl.h	Returns the arc sine of an angle, in the range of -pi/2 through pi/2.
Mfl_Cos_f32	Mfl.h	Calculates the cosine of the argument.
Mfl_Sin_f32	Mfl.h	Calculates the sine of the argument.
Mfl_Sqrt_f32	Mfl.h	Returns the square root of the operand (ValSqrt), determined according to the following equation
StbM_GetCurrentTime	StbM.h	Returns a time value (Local Time Base derived from Global

		Time Base) in standard format. Note: This API shall be called with locked interrupts / within an Exclusive Area to prevent interruption (i.e., the risk that the time stamp is outdated on return of the function call).
StbM_GetCurrentTimeExtended	StbM.h	Returns a time value (Local Time Base derived from Global Time Base) in extended format. Note: This API shall be called with locked interrupts / within an Exclusive Area to prevent interruption (i.e., the risk that the time stamp is outdated on return of the function call).

] ()

8.7 Service Interfaces

8.7.1 Client-Server-Interfaces

8.7.1.1 V2xM_Vdp

[SWS_V2xM_00095] [

Name	V2xM_Vdp	
Comment	Interfaces for Vehicle Data Provider (VDP) to get and set V2X related vehicle information in the BSW V2X-Stack	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetNextLongTermCertificateExpirationDate			
Comments	Service to get the certificate expiration date of the long term certificates that expires in the nearest future.		
Variation	--		
Parameters	ExpirationDate	Comment	Date is based on format Time32 that is specified in [19].
		Type	uint32
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

GetNextPseudonymCertificateExpirationDate			
Comments	Service to get the certificate expiration date of the pseudonym certificates that expires in the nearest future.		
Variation	--		
Parameters	ExpirationDate	Comment	Date is based on format Time32 that is specified in [19].
		Type	uint32
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

GetTime32			
Comments	Service to get the current reference time.		
Variation	--		
Parameters	Time32	Comment	Time is based on TAI mod 2^32, where TAI is the number of elapsed TAI milliseconds since 2004-01-01 00:00:00.000.
		Type	uint32
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

SetPositionAndTime			
Comments	Service for setting positional and time information relevant for the V2X-Stack		
Variation	--		
Parameters	positionAndTime	Comment	--
		Type	V2xM_PositionAndTimeType
		Variation	--
		Direction	IN

Possible Errors	E_OK	Operation successful
	E_NOT_OK	Operation failed

] (SRS_V2X_00412, SRS_V2X_00413, SRS_V2X_00190)

8.7.1.2 V2xM_PseudonymChange

[SWS_V2xM_00172] [

Name	V2xM_PseudonymChange	
Comment	Interfaces for Applications to lock and unlock pseudonym changes within the V2X-BSW-Stack.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

Lock			
Comments	Service for locking the pseudonym change. See [SWS_V2xM_00078] for more information about locking the pseudonym change.		
Variation	--		
Parameters	Duration	Comment	Duration to lock.
		Type	uint16
		Variation	--
		Direction	IN
	HandleId	Comment	HandleId for manual Unlock
		Type	uint64
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	
Unlock			
Comments	Service for unlocking the pseudonym change. See [SWS_V2xM_00079] for more information about locking the pseudonym change.		
Variation	--		

Parameters	HandleId	Comment	HandleId to unlock
		Type	uint64
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

] ()

8.7.1.3 V2xM_Sec

[SWS_V2xM_00173] [

Name	V2xM_Sec	
Comment	Security related interfaces for applications	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

Verify			
Comments	Interfaces for Applications to verify messages on demand instead of verify all incoming messages, to reduce ECU load.		
Variation	--		
Parameters	TransactionId32	Comment	TransactionId of the packet to be verified
		Type	uint32
		Variation	--
		Direction	IN
	SecReport	Comment	--
		Type	V2xM_SecReportType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

] ( )

8.7.1.4 V2xM_GeoMath

[SWS_V2xM_00180] [

Name	V2xM_GeoMath	
Comment	Interfaces for Applications to math functions	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

Distance			
Comments	Service for Calculating the distance between two geographical points		
Variation	--		
Parameters	latitudeA	Comment	Latitude of geographical point A
		Type	sint32
		Variation	--
		Direction	IN
	longitudeA	Comment	Longitude of geographical point A
		Type	sint32
		Variation	--
		Direction	IN
	latitudeB	Comment	Latitude of geographical point B
		Type	sint32
		Variation	--
		Direction	IN
	longitudeB	Comment	Longitude of geographical point B
		Type	sint32
		Variation	--
		Direction	IN
	distance	Comment	Distance between geographical points A and B in [m].

		Type	float32
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

HeadingInTolerance			
Comments	Service for Calculating if difference of heading values are within a tolerance value		
Variation	--		
Parameters	heading1	Comment	First heading value
		Type	float32
		Variation	--
		Direction	IN
	heading2	Comment	Next heading value
		Type	float32
		Variation	--
		Direction	IN
	toleranceValue	Comment	Tolerated difference between heading1 and heading2
		Type	float32
		Variation	--
		Direction	IN
	tolerated	Comment	Return value
		Type	boolean
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	Operation failed	

] ( )

8.7.2 Implementation Data Types

8.7.2.1 ImplementationDataType V2xM_SecReportType

[SWS_V2xM_91000] ⌈

Name	V2xM_SecReportType		
Kind	Type		
Derived from	uint8		
Description	Used to describe the security report after invocation of security services for Decapsulation (verify or decrypt)		
Range	V2X_SECREP_SUCCESS	0x00	Indicating security service has successfully executed
	V2X_SECREP_FALSE_SIGNATURE	0x01	Indicating false signature
	V2X_SECREP_INVALID_CERTIFICATE	0x02	Indicating invalid certificate
	V2X_SECREP_REVOKED_CERTIFICATE	0x03	Indicating revoked certificate
	V2X_SECREP_INCONSISTENT_CHAIN	0x04	Indicating inconsistent certificate chain
	V2X_SECREP_INVALID_TIMESTAMP	0x05	Indicating invalid timestamp
	V2X_SECREP_DUPLICATE_MESSAGE	0x06	Indicating duplicate message
	V2X_SECREP_INVALID_MOBILITY_DATA	0x07	Indicating invalid mobility data
	V2X_SECREP_UNSIGNED_MESSAGE	0x08	Indicating unsigned message
	V2X_SECREP_SIGNER_CERTIFICATE_NOT_FOUND	0x09	Indicating signer certificate not found

	V2X_SECREP_UNSUPPORTED_SIGNER_IDENTIFIER_TYPE	0x0a	Indicating unsupported signer identifier type
	V2X_SECREP_INCOMPATIBLE_PROTOCOL	0x0b	Indicating incompatible protocol
	V2X_SECREP_UNENCRYPTED_MESSAGE	0x0c	Indicating unencrypted message
	V2X_SECREP_DECRYPTION_ERROR	0x0d	Indicating decryption error
	V2X_SECREP_NONE	0xff	Indicating no security service has been executed.
Variation	--		
Available via	Rte_V2xM_Type.h		

] ()

8.7.2.2 ImplementationDataType V2xM_PositionAndTimeType

[SWS_V2xM_00047] [

Name	V2xM_PositionAndTimeType		
Kind	Structure		
Elements	latitude	sint32	Latitude [1/10 microdegree]
	longitude	sint32	Longitude [1/10 microdegree]
	altitude	sint32	Altitude [1/100 m]
	speed	sint16	Speed [1/100 m/s]
	heading	uint16	Heading [1/10 degrees]
	timestamp	uint32	Timestamp [ms]
	semiMajorConfidence	uint16	From position confidence ellipse
	semiMinorConfidence	uint16	From position confidence ellipse
	semiMajorOrientation	uint16	From position confidence ellipse
	pai	boolean	Positional accuracy indicator
	informationValid	boolean	Indicates that position information is valid

Description	Position and time related information as defined within [23] chapter 7.2.
Variation	--
Available via	Rte_V2xM_Type.h

] ( )

8.7.3 Ports

8.7.3.1 V2xM_V2xM_GeoMath

[SWS_V2xM_00192] [

Name	V2xM_GeoMath		
Kind	ProvidedPort	Interface	V2xM_GeoMath
Description	Service port for geographical calculation requests.		
Variation	--		

] ()

8.7.3.2 V2xM_V2xM_PseudonymChange

[SWS_V2xM_00193] [

Name	V2xM_PseudonymChange		
Kind	ProvidedPort	Interface	V2xM_PseudonymChange
Description	Service port for pseudonym lock and unlock requests.		
Variation	--		

] ()

8.7.3.3 V2xM_V2xM_Sec

[SWS_V2xM_00194] [

Name	V2xM_Sec		
Kind	ProvidedPort	Interface	V2xM_Sec
Description	Service port for security operations of V2X messages.		
Variation	--		

] ()

8.7.3.4 V2xM_V2xM_Vdp

[SWS_V2xM_00195] [

Name	V2xM_Vdp

Kind	ProvidedPort	Interface	V2xM_Vdp
Description	Service port for exchange of vehicle related data. This port is used by the Vehicle Data Provider SW-C.		
Variation	--		

] ()

9 Sequence diagrams

9.1 V2xM_Init – Time initialization

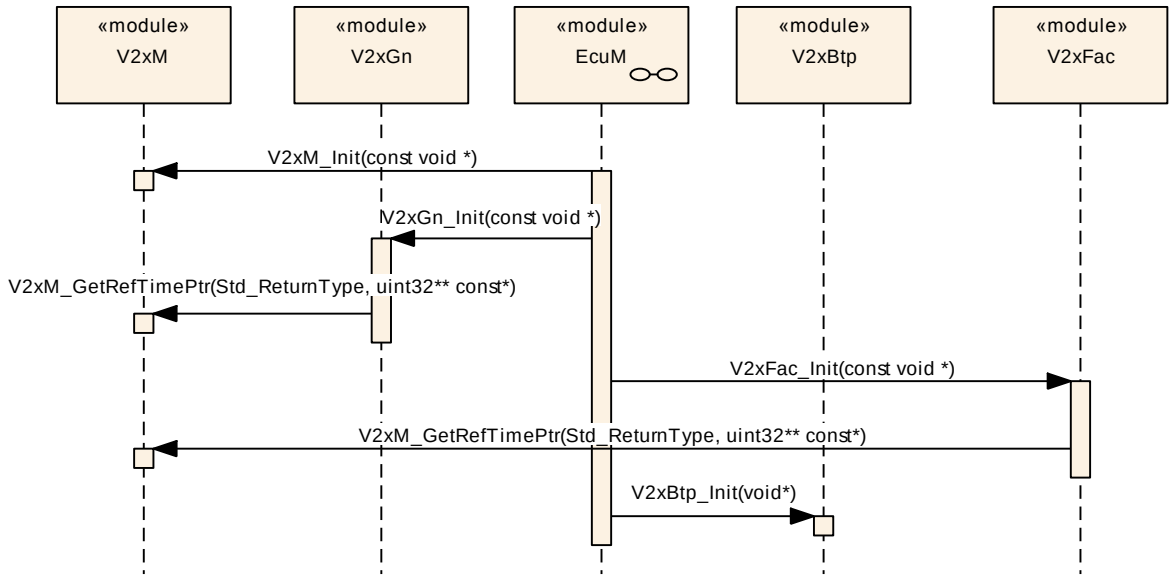

Figure 9-1: V2xM_Init - Time initialization

9.2 Position and time update V2xGn

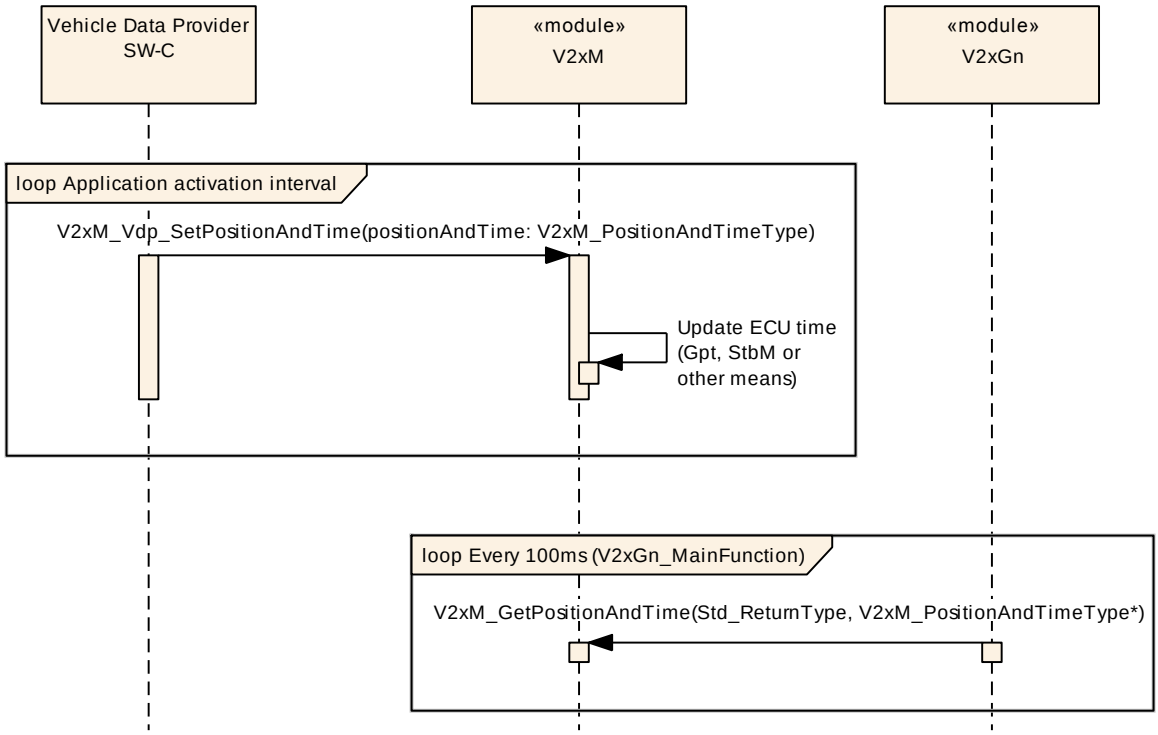

Figure 9-2: Position and time update V2xGn

9.3 Position and time update V2xFac

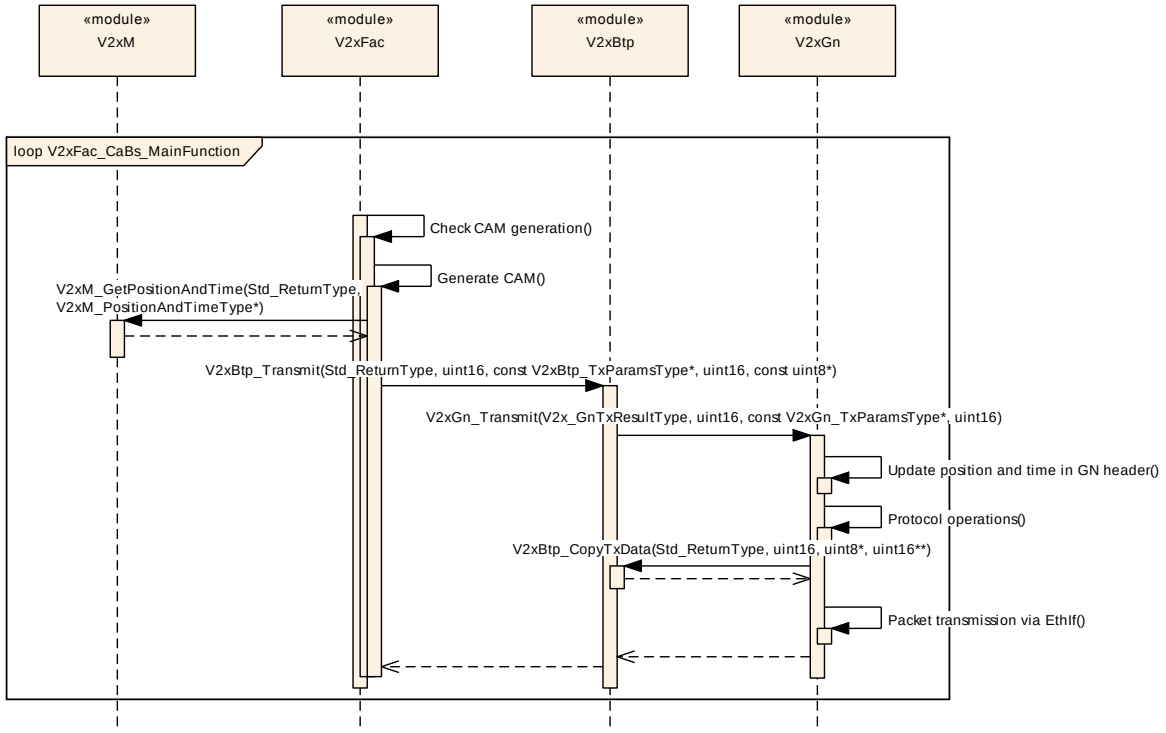

Figure 9-3: Position and time update V2xFac

9.4 Time handling at reception

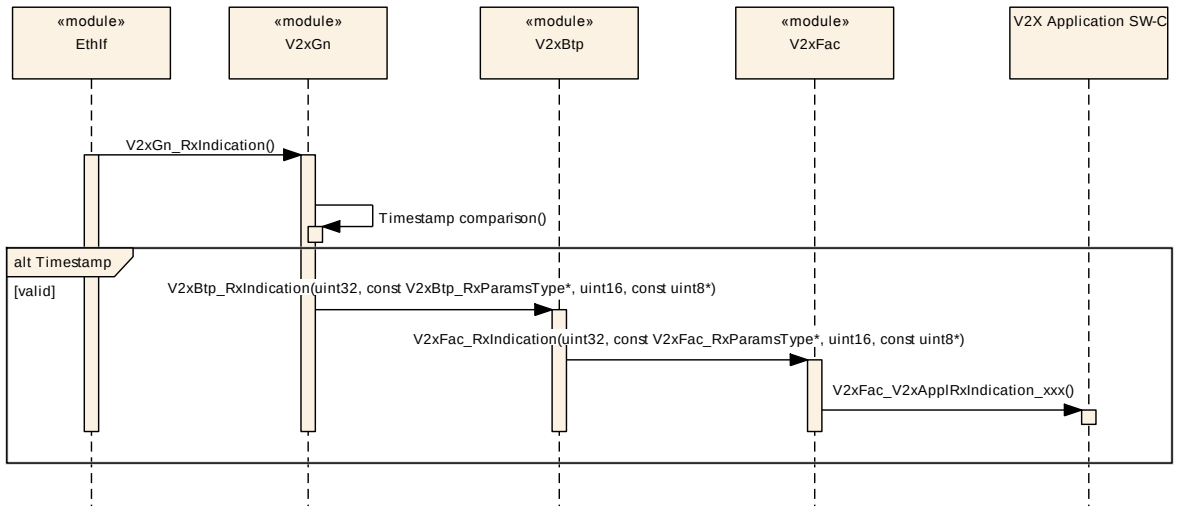

Figure 9-4 – Time handling at reception

9.5 Initialization of Wireless Drivers

The Initialization of the Wireless Ethernet Driver and the Wireless Ethernet Transceiver Driver shall be done as depicted in Figure 9-5.

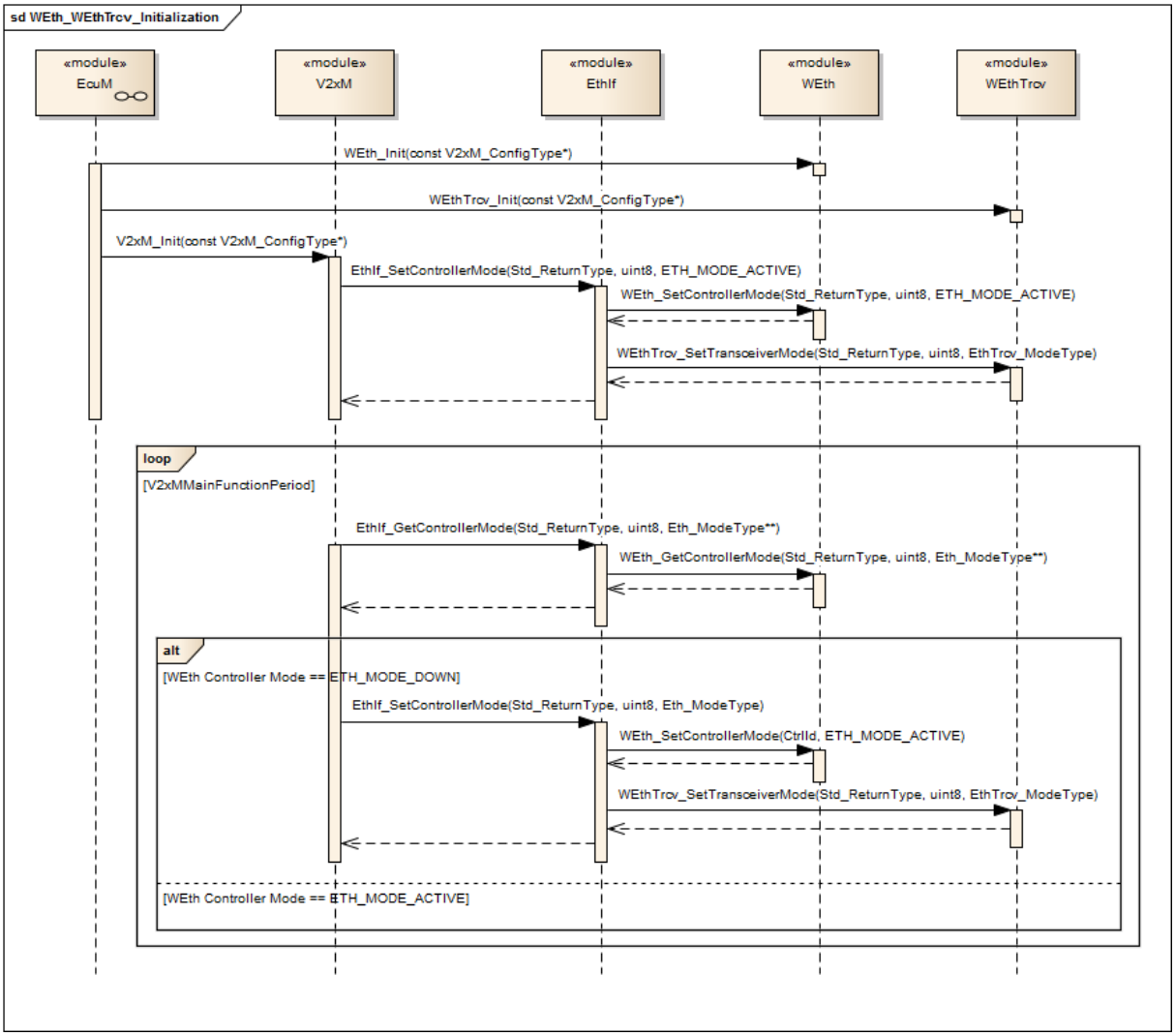

Figure 9-5: WEth and WEthTrcv initialization

10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of the module V2xM.

Chapter 10.2 specifies additionally published information of the module V2xM.

10.1 Containers and configuration parameters

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

10.1.1 Variants

[SWS_V2xM_00191] ⌈ The V2xM module only supports VARIANT-PRE-COMPILE ⌋ (SRS_BSW_00345)

10.1.2 V2xM

SWS Item	
Module Name	*V2xM*
Module Description	Configuration of the V2xM (V2XManagement) module.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
V2xMConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR V2xM module.
V2xMGeneral	1	General configuration of V2xM module.

10.1.3 V2xMConfig

SWS Item	**ECUC_V2xM_00001 :**
Container Name	V2xMConfig
Description	This container contains the configuration parameters and sub containers of the AUTOSAR V2xM module.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
V2xMSecurityConfig	1	Configuration of the security services of V2xM.

10.1.4 V2xMSecurityConfig

SWS Item	**ECUC_V2xM_00002 :**		
Container Name	V2xMSecurityConfig		
Description	Configuration of the security services of V2xM.		
Configuration Parameters			

SWS Item	**ECUC_V2xM_00005 :**		
Name	V2xMSecurityVerificationOnDemand		
Parent Container	V2xMSecurityConfig		
Description	Switches the Verification on Demand (VoD) ON or OFF. • true: enabled (ON) • false: disabled (OFF)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00004 :**		
Name	V2xMSecurityNvMBlockDescriptorLongTermCertificates		
Parent Container	V2xMSecurityConfig		
Description	Reference to NVRAM block containing the none volatile data of long term certificates.		
Multiplicity	1		
Type	Symbolic name reference to [NvMBlockDescriptor]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00003 :**		
Name	V2xMSecurityNvMBlockDescriptorPseudonymCertificates		
Parent Container	V2xMSecurityConfig		
Description	Reference to NVRAM block containing the none volatile data of pseudonym certificates.		
Multiplicity	1		
Type	Symbolic name reference to [NvMBlockDescriptor]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00007 :**
Name	V2xMSignatureGenerationConfigRef
Parent Container	V2xMSecurityConfig
Description	Select CSM service configuration that is used for authentication.
Multiplicity	1
Type	Reference to [CsmSignatureGenerateConfig]

Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00006 :**		
Name	V2xMSignatureVerifyConfigRef		
Parent Container	V2xMSecurityConfig		
Description	Select CSM service configuration that is used for authentication.		
Multiplicity	1		
Type	Reference to [CsmSignatureVerifyConfig]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.1.5 V2xMGeneral

SWS Item	**ECUC_V2xM_00008 :**
Container Name	V2xMGeneral
Description	General configuration of V2xM module.
Configuration Parameters	

SWS Item	**ECUC_V2xM_00009 :**		
Name	V2xMDevErrorDetect		
Parent Container	V2xMGeneral		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. ▪ true: enabled (ON) ▪ false: disabled (OFF)		
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		

SWS Item	**ECUC_V2xM_00015 :**	
Name	V2xMMainFunctionPeriod	
Parent Container	V2xMGeneral	
Description	This parameter defines the schedule period of V2xM_MainFunction.Unit: [s]	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	]0 .. 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		

SWS Item	**ECUC_V2xM_00010 :**		
Name	V2xMVersionInfoApi		
Parent Container	V2xMGeneral		
Description	Enable/disables the API for reading the version information of the V2xM Module. ▪ true: enabled (ON) ▪ false: disabled (OFF)		
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		

SWS Item	**ECUC_V2xM_00012 :**		
Name	V2xMEthIfCtrlRef		
Parent Container	V2xMGeneral		
Description	Reference to EthIf controller where the channel and radio parameters should be read and written to.		
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: local		

SWS Item	**ECUC_V2xM_00013 :**		
Name	V2xMGptChannelConfigurationRef		
Parent Container	V2xMGeneral		
Description	Reference to General Purpose Timer.		
Multiplicity	0..1		
Type	Symbolic name reference to [GptChannelConfiguration]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00011 :**
Name	V2xMNvMBlockDescriptor
Parent Container	V2xMGeneral
Description	Reference to NVRAM block containing the none volatile data.
Multiplicity	1
Type	Symbolic name reference to [NvMBlockDescriptor]
Post-Build Variant Value	false

Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_V2xM_00014 :**		
Name	V2xMStbMSynchronizedTimeBaseRef		
Parent Container	V2xMGeneral		
Description	Reference to synchronized time-base.		
Multiplicity	0..1		
Type	Symbolic name reference to [StbMSynchronizedTimeBase]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

11 Not applicable requirements