| Document Title | Specification of Vehicle-2-X Geo Networking |
|---|---|
| **Document Owner** | AUTOSAR |
| **Document Responsibility** | AUTOSAR |
| **Document Identification No** | 793 |

| | |
|---|---|
| **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 | • Adaptation to new lower interface through L-SDU router |
| 2023-11-23 | R23-11 | AUTOSAR Release Management | • Removal of Tx confirmation and transaction ID |
| 2022-11-24 | R22-11 | AUTOSAR Release Management | • Editorial changes to reflect the introduction of V2xDM |
| 2021-11-25 | R21-11 | AUTOSAR Release Management | • Correction in V2xGn_TxConfirmation prototype<br>• Scrubbing and editorial Changes |
| 2020-11-30 | R20-11 | AUTOSAR Release Management | • Editorial Changes |
| 2019-11-28 | R19-11 | AUTOSAR Release Management | • Added support for C2CCC BSP 1.3<br>• Changed Document Status from Final to published |
| 2018-10-31 | 4.4.0 | AUTOSAR Release Management | • Update ETSI EN 302 636-4-1 v1.3.1<br>• Corrections on init and configuration<br>• Corrections in Tx and Rx flows<br>• 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**

# Contents

1 Introduction and functional overview 7
1.1 Architectural overview . . . 7
1.2 Functional overview . . . 8
2 Acronyms and Abbreviations 9
3 Related documentation 10
3.1 Input documents & related standards and norms . . . 10
3.2 Related specification . . . 11
4 Constraints and assumptions 12
4.1 Limitations . . . 12
4.2 Applicability to car domains . . . 12
4.3 Authorisation Tickets and Pseudonyms . . . 12
5 Dependencies to other modules 13
5.1 AUTOSAR DET (Default Error Tracer) . . . 13
5.2 AUTOSAR EcuM (Ecu State Manager) . . . 13
5.3 AUTOSAR Ethernet Interface (EthIf) . . . 13
5.4 AUTOSAR Linklayer Sdu Routing Module (LSduR) . . . 13
5.5 AUTOSAR Vehicle-2-X Basic Transport Protocol (V2xBtp) . . . 13
5.6 AUTOSAR Vehicle-2-X Management (V2xM) . . . 13
5.7 File structure . . . 14
5.7.1 Code file structure . . . 14
6 Requirements Tracing 15
7 Functional specification 16
7.1 General Functionality . . . 16
7.2 GeoNetworking Packet Structure and Format . . . 17
7.3 GeoNetworking Protocol Operations . . . 17
7.3.1 Network Management . . . 17
7.3.2 Security Mechanisms . . . 18
7.4 Message Forwarding . . . 19
7.5 Message Transmission . . . 19
7.6 Message Reception . . . 23
7.7 State handling of PDUs . . . 24
7.8 Error Classification . . . 25
7.8.1 Development Errors . . . 25
7.8.2 Runtime Errors . . . 26
7.8.3 Production Errors . . . 26
7.8.4 Extended Production Errors . . . 26
8 API specification 27
8.1 Imported types . . . 27

8.2 Type definitions . . . 28
8.2.1 V2xGn_TxParamsType . . . 28
8.3 Function definitions . . . 29
8.3.1 V2xGn_Init . . . 29
8.3.2 V2xGn_GetVersionInfo . . . 29
8.3.3 V2xGn_V2xM_PreparePseudonymChange . . . 30
8.3.4 V2xGn_V2xM_CommitPseudonymChange . . . 31
8.3.5 V2xGn_V2xM_AbortPseudonymChange . . . 32
8.3.6 V2xGn_Transmit . . . 33
8.4 Callback notifications . . . 34
8.4.1 V2xGn_V2xM_EncapConfirmation . . . 34
8.4.2 V2xGn_V2xM_DecapConfirmation . . . 35
8.4.3 V2xGn_RxIndication . . . 36
8.4.4 V2xGn_TxConfirmation . . . 37
8.5 Scheduled functions . . . 38
8.5.1 V2xGn_MainFunction . . . 38
8.6 Expected interfaces . . . 38
8.6.1 Mandatory Interfaces . . . 38
8.6.2 Optional Interfaces . . . 39

9 Sequence diagrams 41

9.1 V2xGn_RxIndication . . . 41
9.2 V2xGn_Transmit . . . 42
9.3 V2xGn_V2xM_UpdatePseudonym . . . 42
9.4 V2xGn_MainFunction . . . 43

10 Configuration specification 44

10.1 Containers and configuration parameters . . . 44
10.1.1 Variants . . . 44
10.1.2 V2xGn . . . 44
10.1.3 V2xGnGeneral . . . 45
10.1.4 V2xGnBeaconService . . . 52
10.1.5 V2xGnPacketForwarding . . . 54
10.1.6 V2xGnConfig . . . 62
10.1.7 V2xGnRxPdu . . . 63
10.1.8 V2xGnTxPdu . . . 64
10.2 Published Information . . . 65

A Not applicable requirements 66

B History of Specification Items 67

B.1 Specification Item History of this document compared to AUTOSAR R24-11. . . . 67
B.1.1 Added Specification Items in R24-11 . . . 67
B.1.2 Changed Specification Items in R24-11 . . . 68
B.1.3 Deleted Specification Items in R24-11 . . . 68
B.2 Constraint Item History of this document compared to AUTOSAR R24-11. 69

B.2.1 Added Constraints in R24-11 . . . 69
B.2.2 Changed Constraints in R24-11 . . . 69
B.2.3 Deleted Constraints in R24-11 . . . 69
B.3 Specification Item History of this document compared to AUTOSAR R23-11. . . . 69
B.3.1 Added Specification Items in R23-11 . . . 69
B.3.2 Changed Specification Items in R23-11 . . . 72
B.3.3 Deleted Specification Items in R23-11 . . . 72
B.4 Specification Item History of this document compared to AUTOSAR R22-11. . . . 72
B.4.1 Added Specification Items in R22-11 . . . 72
B.4.2 Changed Specification Items in R22-11 . . . 74
B.4.3 Deleted Specification Items in R22-11 . . . 75
B.5 Constraint Item History of this document compared to AUTOSAR R22-11. 75
B.5.1 Added Constraints in R22-11 . . . 75
B.5.2 Changed Constraints in R22-11 . . . 75
B.5.3 Deleted Constraints in R22-11 . . . 75

# 1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module Vehicle-2-X GeoNetworking (V2xGn).

V2xGn together with Vehicle-2-X Facilities (V2xFac) [1], Vehicle-2-X Basic Transport (V2xBtp) [2], Vehicle-2-X Management (V2xM) [3] , Vehicle-2-X Data Manager [4] and AUTOSAR BSW modules Ethernet Interface (EthIf) [5], Linklayer Sdu Routing Module (LSduR) [6], Wireless Ethernet Driver (WEth) [7] and Wireless Ethernet Transceiver Driver (WEthTrcv) [8] forms the V2X stack within the AUTOSAR architecture.

The base for this document is the GeoNetworking specification [9] [10]. It is assumed that the reader is familiar with this specification.

## 1.1 Architectural overview

V2xGn provides services to and is dependent on the upper V2xBtp module and uses the services of and gets services from the lower EthIf module to realize its functions explained in section 1.2 and chapter 7 of this document.

Positioning of the V2xGn module within the AUTOSAR BSW and the Layered Software architecture is shown in the figure below.

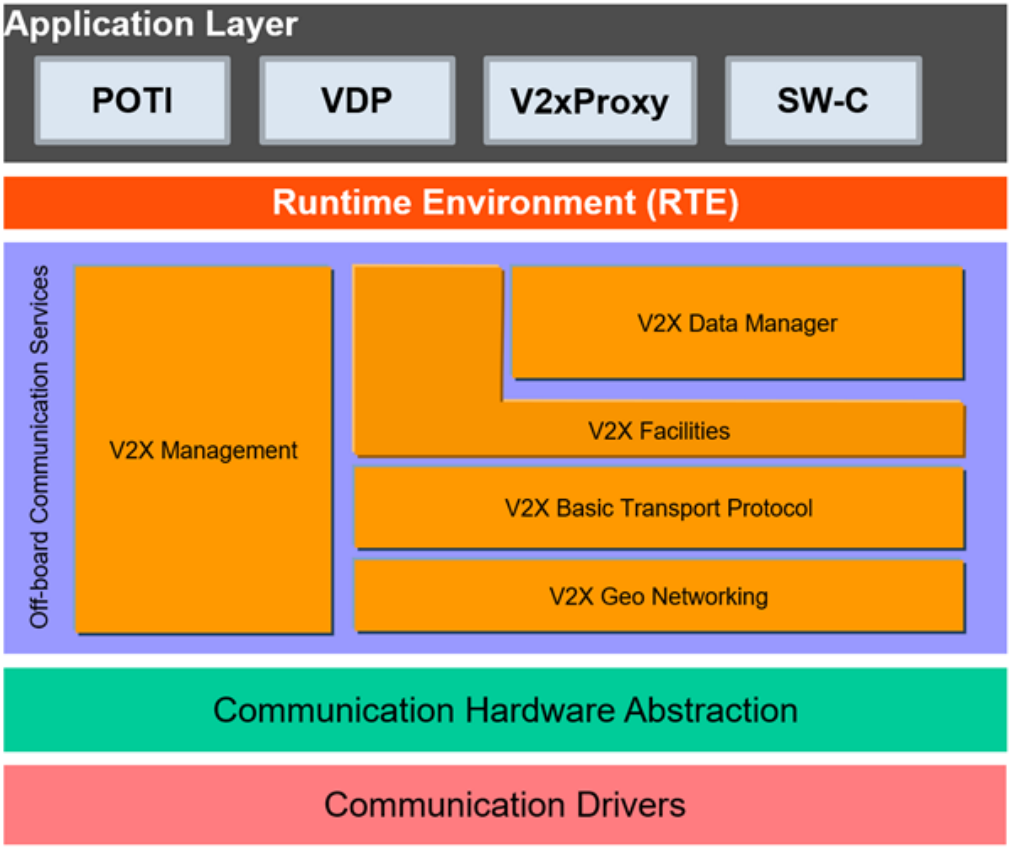


**Figure 1.1: AUTOSAR BSW software architecture - V2xGn module scope**

## 1.2 Functional overview

The internal functionality of the V2xGn module should comply to the medium independent specification of the GeoNetworking protocol [9] and the medium dependent specification of the GeoNetworking protocol [10], relying on ETSI ITS-G5 technology as medium. The module provides services to the upper V2xBtp module specified in [2] and in order to provide its packet transport services, it relies on the lower EthIf module [5]. Vehicle-2-X specific data is also exchanged with the V2xM module.

GeoNetworking protocol is a set of network layer functionalities that enables ad hoc communication without infrastructure support using geographical positions of the communicating entities. It supports communication among individual Intelligent Transport System (ITS) station and distribution of packets in geographical areas. As GeoNetworking can be executed over different ITS technologies such as ITS-G5 and infrared, GeoNetworking specification consists of a standard for media-independent functionality [9] which specifies all functions that are common to all ITS access technologies and one or more media-dependent specifications [10] which includes extensions for a specific ITS technology.

# 2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the V2xGeoNetworking module that are not included in the AUTOSAR glossary [11].

| Abbreviation / Acronym: | Description: |
|---|---|
| BTP | Basic Transport Protocol |
| CBF | Contention-Based Forwarding |
| DET | Default Error Tracer |
| GAC | GeoAnycast |
| GBC | GeoBroadcast |
| GN | GeoNetworking |
| GN-SDU | GeoNetworking Service Data Unit |
| ITS | Intelligent Transport System |
| MAC | Medium Access Control |
| SHB | Single Hop Broadcast |
| TC | Traffic Class |
| TSB | Topologically Scoped Broadcast |

**Table 2.1: Acronyms and abbreviations used in the scope of this Document**

# 3 Related documentation

## 3.1 Input documents & related standards and norms

[1] Specification of Vehicle-2-X Facilities
AUTOSAR_CP_SWS_V2XFacilities

[2] Specification of Vehicle-2-X Basic Transport
AUTOSAR_CP_SWS_V2XBasicTransport

[3] Specification of Vehicle-2-X Management
AUTOSAR_CP_SWS_V2XManagement

[4] Specification of Vehicle-2-X Data Manager
AUTOSAR_CP_SWS_V2XDataManager

[5] Specification of Ethernet Interface
AUTOSAR_CP_SWS_EthernetInterface

[6] Specification of Linklayer Sdu Routing Module
AUTOSAR_CP_SWS_LSduRouter

[7] Specification of Wireless Ethernet Driver
AUTOSAR_CP_SWS_WirelessEthernetDriver

[8] Specification of Wireless Ethernet Transceiver Driver
AUTOSAR_CP_SWS_WirelessEthernetTransceiverDriver

[9] EN 302 636-4-1 V1.3.1: Vehicular Communication; Geonetworking; Part 4 Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality

[10] TS 102 636-4-2 V1.1.1: Intelligent Transport Systems (ITS); GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 2: Media-dependent functionalities for ITS-G5

[11] Glossary
AUTOSAR_FO_TR_Glossary

[12] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[13] IEEE Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification. Amendment 6: Wireless Access in Vehicular Environments (IEEE STD 802.11p-2010).

[14] Car 2 Car Communication Consortium; Basic System Profile release 1.3

[15] Specification of Default Error Tracer
AUTOSAR_CP_SWS_DefaultErrorTracer

[16] Specification of ECU State Manager
AUTOSAR_CP_SWS_ECUStateManager

[17] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[18] Requirements on Vehicle-2-X Communication
AUTOSAR_CP_RS_V2XCommunication

[19] TS 102 724 V1.1.1: Intelligent Transport Systems (ITS); Harmonized Channel Specifications for Intelligent Transport Systems operating in the 5 GHz frequency band

[20] List of EtherTypes by IEEE
http://standards.ieee.org/develop/regauth/ethertype/eth.txt

## 3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [12], which is also valid for V2xGn.

Thus, the specification SWS BSW General shall be considered as additional and required specification for V2xGn.

# 4 Constraints and assumptions

## 4.1 Limitations

- The GeoNetworking protocol and therefore the V2xGn module requires a broadcast capable access layer in order to provide transmit services.
- Wireless Communication supports IEEE 802.11p only [13]. Other 802.11 standards (e.g. for infrastructure networks and integration with TCP/IP) can be extended in future releases of the AUTOSAR standard.
- The V2X modules follow the guidance regarding the Day-1 scenarios defined by the Basic System Standards Profile from Car-2-Car-Consortium [14].

## 4.2 Applicability to car domains

This specification is applicable to all car domains.

## 4.3 Authorisation Tickets and Pseudonyms

The Authorisation Ticket (AT) is referred to as Pseudonym in this document.

# 5 Dependencies to other modules

This section describes the relations of the V2xGn module to other modules within the AUTOSAR basic software architecture. It outlines the modules that are required or optional for the realization of the V2xGn module and the V2xGn services that these modules use.

## 5.1 AUTOSAR DET (Default Error Tracer)

In development mode, the V2xGn module reports errors through DET [15].

## 5.2 AUTOSAR EcuM (Ecu State Manager)

The EcuM [16] is responsible for the initialization of V2xGn.

## 5.3 AUTOSAR Ethernet Interface (EthIf)

The Ethernet Interface is the lower layer module of the V2xGn module.

## 5.4 AUTOSAR Linklayer Sdu Routing Module (LSduR)

The Linklayer Sdu Routing Module is the lower layer module of the V2xGn module for the data flow.

## 5.5 AUTOSAR Vehicle-2-X Basic Transport Protocol (V2xBtp)

The V2xBtp is the upper layer module of the V2xGn module.

## 5.6 AUTOSAR Vehicle-2-X Management (V2xM)

V2xM is used for interchange of Data with other V2X-Modules. Security mechanisms are configured by the V2xM and are used by V2xGn.

## 5.7 File structure

### 5.7.1 Code file structure

For details refer to the chapter 5.1.6 "Code file structure" in "General Specification of Basic Software Modules" [12].

# 6 Requirements Tracing

The following tables reference the requirements specified in [17] and [18] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

| Requirement | Description | Satisfied by |
|---|---|---|
| **[SRS_BSW_00345]** | BSW Modules shall support pre-compile configuration | [SWS_V2xGn_00078] |
| **[SRS_V2X_00010]** | The implementation of the V2X system shall follow additional guidance given by C2C-CC requirements | [SWS_V2xGn_00269] [SWS_V2xGn_20169] [SWS_V2xGn_20181] [SWS_V2xGn_20260] [SWS_V2xGn_20262] [SWS_V2xGn_20263] [SWS_V2xGn_20264] [SWS_V2xGn_20265] [SWS_V2xGn_20266] [SWS_V2xGn_20267] [SWS_V2xGn_20268] [SWS_V2xGn_20270] [SWS_V2xGn_20401] |
| **[SRS_V2X_00160]** | The V2X system shall use end-to-end security for communication to external entities | [SWS_V2xGn_00026] [SWS_V2xGn_20251] |
| **[SRS_V2X_00161]** | The V2X system shall employ the security envelope on its Network layer | [SWS_V2xGn_00012] [SWS_V2xGn_20251] |
| **[SRS_V2X_00164]** | The V2X system shall only forward verified messages | [SWS_V2xGn_00026] |
| **[SRS_V2X_00176]** | The V2X system shall change pseudonyms | [SWS_V2xGn_00028] [SWS_V2xGn_00091] [SWS_V2xGn_00112] [SWS_V2xGn_00115] |
| **[SRS_V2X_00259]** | The V2X system shall manage the life time of all DENM packets | [SWS_V2xGn_20259] |
| **[SRS_V2X_00279]** | The V2X system shall support circular, rectangular and ellipsoidal geographical areas | [SWS_V2xGn_20266] |
| **[SRS_V2X_00391]** | The V2X system’s access layer shall be ITS-G5 compliant | [SWS_V2xGn_20414] |
| **[SRS_V2X_00531]** | The V2X system’s Networking Layer shall support addressing based on geographic coordinates | [SWS_V2xGn_20250] [SWS_V2xGn_20251] [SWS_V2xGn_20252] [SWS_V2xGn_20255] [SWS_V2xGn_20258] [SWS_V2xGn_20414] [SWS_V2xGn_20416] |
| **[SRS_V2X_26011]** | The V2X Geo Networking shall support PDU based communication | [SWS_V2xGn_00136] [SWS_V2xGn_00137] [SWS_V2xGn_00138] [SWS_V2xGn_00139] [SWS_V2xGn_00140] [SWS_V2xGn_00141] [SWS_V2xGn_00143] [SWS_V2xGn_00144] [SWS_V2xGn_00145] [SWS_V2xGn_00146] [SWS_V2xGn_00147] [SWS_V2xGn_00148] [SWS_V2xGn_00149] [SWS_V2xGn_00150] |

**Table 6.1: Requirements Tracing**

# 7 Functional specification

## 7.1 General Functionality

**[SWS_V2xGn_00012]**

*Upstream requirements:* SRS_V2X_00161

⌈The V2xGn Module shall implement the GeoNetworking Protocol as defined in [9], [10], and [14] unless specified otherwise in this document.⌋

**[SWS_V2xGn_00013]** ⌈The GeoNetworking Protocol shall support the GeoNetworking related requirements specified in [14].⌋

**[SWS_V2xGn_20250]**

*Upstream requirements:* SRS_V2X_00531

⌈All default constants and parameters of the V2xGn module not defined or overwritten in the current document shall be set as specified in Annex H of [9].⌋

**[SWS_V2xGn_20251]**

*Upstream requirements:* SRS_V2X_00531, SRS_V2X_00160, SRS_V2X_00161

⌈The V2xGn module shall be implemented assuming the ETSI parameter itsGnSecurity is constantly set to ENABLED.⌋

**[SWS_V2xGn_20252]**

*Upstream requirements:* SRS_V2X_00531

⌈The V2xGn module shall only support anonymous address configuration mode.⌋

**[SWS_V2xGn_20255]**

*Upstream requirements:* SRS_V2X_00531

⌈The V2xGn module shall support geo-areas areas of up to 80 km2. In consequence, the itsGnMaxGeoAreaSize shall have a value of 80. It is configurable by the configuration option `V2xGnItsGnMaxGeoAreaSize`.⌋

**[SWS_V2xGn_20414]**

*Upstream requirements:* SRS_V2X_00531, SRS_V2X_00391

⌈The V2xGn module shall be implemented with respect to the ETSI parameter itsGnIfType constantly set to ITS-G5.⌋

**[SWS_V2xGn_00130]** ⌈The V2xGn module shall get the pointer to the current time information via V2xM_GetRefTimePtr() within the `V2xGn_Init()`.⌋

**[SWS_V2xGn_20416]**

*Upstream requirements:* SRS_V2X_00531

⌈Packet repetition shall not be performed by V2xGn module and the corresponding steps in the packet handling procedures in [9] clause 10.3 shall not be executed.

The parameter 'Maximum repetition time' of the service primitive GN-DATA.request is not applicable. Also, the GN protocol constant itsGnMinPacketRepetitionInterval is not applicable.⌋

## 7.2 GeoNetworking Packet Structure and Format

**[SWS_V2xGn_00020]** ⌈The GeoNetworking protocol shall only support the packet header types Single Hop Broadcast packet header, GeoBroadcast packet headers and Beacon packet header.⌋

**[SWS_V2xGn_20258]**

*Upstream requirements:* SRS_V2X_00531

⌈The V2xGn module shall set the LifeTime field of all SHB packets to 1 second. Consequently, the multiplier bit of the LT field shall be set to 1 and the base bit of the LT field shall be set to 1.⌋

**[SWS_V2xGn_20259]**

*Upstream requirements:* SRS_V2X_00259

⌈The V2xGn module shall set the LifeTime field of all GBC packets to the value of the `maxPacketLifetime` from the transmit parameters `TxParams`.The value of the LifeTime field shall not exceed the itsGnMaxPacketLifetime, specified in [9], Annex H.⌋

## 7.3 GeoNetworking Protocol Operations

### 7.3.1 Network Management

**[SWS_V2xGn_00022]** ⌈The V2xGn module shall update the local position and time information. The minimum update frequency is configured by the configuration parameter `V2xGnItsGnMinUpdateFrequencyEPV`. The scheduled function `V2xGn_MainFunction()` shall be used for the cyclic update.⌋

**[SWS_V2xGn_00023]** ⌈The V2xGn module shall support GeoNetworking beaconing. The scheduled function `V2xGn_MainFunction()` shall be used for the cyclic beaconing.⌋

**[SWS_V2xGn_00269]**

*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module shall only send beacons if ego position is accurate enough to set the Position Accuracy Indicator (PAI) to 1.⌋

**[SWS_V2xGn_00081]** ⌈The V2xGn module shall support Location Table Maintainance. The scheduled function `V2xGn_MainFunction()` shall be used for the cyclic maintainance of the Location Table.⌋

**[SWS_V2xGn_00129]** ⌈The V2xGn module shall get the current position and time information via V2xM_GetPositionAndTime() within the MainFunction.⌋

### 7.3.2 Security Mechanisms

**[SWS_V2xGn_00026]**

*Upstream requirements:* SRS_V2X_00160, SRS_V2X_00164

⌈The V2xGn module shall use security services provided by V2xM V2xM_V2xGn_ReqEncap() and V2xM_V2xGn_ReqDecap().⌋

**[SWS_V2xGn_00028]**

*Upstream requirements:* SRS_V2X_00176

⌈The V2xGn shall suspend transmission of messages and clear transmit buffers when a pseudonym changes is in preparation.⌋

Note: The V2xM will notify the V2xGn about pseudonym changes via `V2xGn_V2xM_PreparePseudonymChange()`, `V2xGn_V2xM_CommitPseudonymChange()` and `V2xGn_V2xM_AbortPseudonymChange()`.

## 7.4 Message Forwarding

**[SWS_V2xGn_20266]**

*Upstream requirements:* SRS_V2X_00010, SRS_V2X_00279

⌈The V2xGn module shall only support Area forwarding algorithms specified in [9] Annex E.3.⌋

**[SWS_V2xGn_20267]**

*Upstream requirements:* SRS_V2X_00010

⌈When forwarding packets, the V2xGn module shall use the DCC profile DP3 as defined in [19].⌋

**[SWS_V2xGn_20169]**

*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module shall check the distance from the sender position - in the security envelope, if available - and forward only messages with a distance from the sender of equal or less than 6 km.⌋

## 7.5 Message Transmission

**[SWS_V2xGn_00034]** ⌈The V2xGn module shall provide the API `V2xGn_Transmit()` to enable transmit requests from the V2xBtp Module.⌋

**[SWS_V2xGn_00082]**

*Status:* OBSOLETE

⌈The V2xGn module shall use EthIf_ProvideTxBuffer() to aquire a buffer within the Wireless Ethernet Driver for a V2X Packet to be transmitted. This has to be done during the `V2xGn_Transmit()` context.⌋

**[SWS_V2xGn_00083]** ⌈The V2xGn module shall provide transmission parameters to the Wireless Ethernet Driver for a V2X Packet to be transmitted via an API call to EthIf_SetBufWTxParams(). This has to be done during the `V2xGn_Transmit()` context.⌋

**[SWS_V2xGn_00035]**
*Status:* OBSOLETE
*Use instead:* SWS_V2xGn_00136

⌈The V2xGn module shall transmits packets using the EthIf_Transmit() API provided by the EthIf Module. This has to be done during the `V2xGn_Transmit()` context.⌋

**[SWS_V2xGn_00136] Usage of `LSduR_V2xGnTransmit()` in context of `V2xGn_Transmit()`**
*Status:* DRAFT
*Replaces:* SWS_V2xGn_00082, SWS_V2xGn_00035
*Upstream requirements:* SRS_V2X_26011

⌈The V2xGn module shall transmit packets using the `LSduR_V2xGnTransmit()` API provided by the LSduR Module. This has to be done during the `V2xGn_Transmit()` context.⌋

**[SWS_V2xGn_00135]** ⌈The V2xGn module shall create a unique EncapReqId for each packet to be transmitted. This EncapReqId is used to track the result of each encapsulation request during which the packet to be transmitted is signed to authenticate the transmitting ITS station.⌋

**[SWS_V2xGn_20260]**
*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module shall buffer GBC packets when no neighbours are available (store-carry-forward) if the SCF bit of the TC (Traffic Class) field of GBC packets is set to 1.⌋

**[SWS_V2xGn_20262]**
*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module is not required to offload packets to another channel. Consequently, the channel offload bit of the TC (Traffic Class) field in the BTP frames to be sent should be set to 0. Value 1 will be ignored anyway.⌋

**[SWS_V2xGn_20263]**
*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module shall only use the DCC profiles specified in [SWS_WEth_20235]. Consequently, the DCC Profile ID bits of the TC (Traffic Class) field shall only use the DPID values defined in [SWS_WEth_20235].⌋

**[SWS_V2xGn_20264]**
*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn shall set the itsGnIsMobile bit of the Flags field to 1.⌋

**[SWS_V2xGn_20265]**

*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn shall set the Maximum Hop Limit (MHL) field to 10.⌋

**[SWS_V2xGn_20270]**

*Upstream requirements:* SRS_V2X_00010

⌈All GeoNetworking frames sent by the V2xGn module shall use the EtherType value 0x8947 as listed by the IEEE Registration Authority at [20].⌋

**[SWS_V2xGn_20401]**

*Upstream requirements:* SRS_V2X_00010

⌈The GN Source Address shall be constructed as follows:

- Set the field M (bit 0) to 0.
- Set the field ST (bits 1 to 5) to the station type of the ITS-S. The station type in the GN source address shall be identical to the station type in CAMs/DENMs.
- Set reserved bits 6 to 15 to 0.
- Set the field MID (bits 16 to 63) to the value of the MAC address.

⌋

**[SWS_V2xGn_00128]** ⌈The V2xGn module shall call V2xM_GetGlobalTxParams() that delivers own channel CBR information set in the GeoNetworking header to be transmitted.⌋

The transmission request towards the LSduR is performed with direct data provision. With direct data provision, the data for transmission is forwarded in one single call via the LSduR to the lower layer. The lower layer is responsible to transfer the data to a transmit buffer.

**[SWS_V2xGn_00137] Transmission request with direct data provision**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If a transmission request need to be performed, then the V2xGn module shall call `LSduR_V2XGnTransmit`() with the following arguments:

- `TxPduId` equal to the PDU id that is referenced by the used `V2xGnTxPdu`
- `PduInfoPtr.SduDataPtr` equal to the data pointer
- `PduInfoPtr.SduLength` equal to the data length

- PduInfoPtr.MetaDataPtr equal to the pointer of the created MetaDataItem configured at the V2xGnTxPdu that corresponds to the given TxPduId.

⌋

**[SWS_V2xGn_00138] Meta data handling while containing headers**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If the V2xGn module provides headers necessary for the transmission through PduInfoPtr.MetaDataPtr then MetaDataItem shall be set in the following order:

- ETHERNET_MAC_64 equal to the destination MAC address
- LISTELEM_PTR equal to the pointer of the created instance of type ListElemStructType in the following order:
  - create an instance of type ListElemStructType and set NextListElemPtr to NULL_PTR
  - set DataPtr to address of the created header and DataLength to the length of the created header
- PRIORITY_8 equal to the priority value used for this packet

⌋

**[SWS_V2xGn_00139] Meta data handling while not containing headers**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If the V2xGn module provides headers necessary for the transmission through PduInfoPtr.SduDataPtr then MetaDataItem shall be set in the following order:

- ETHERNET_MAC_64 equal to the Physical destination address (MAC address in network byte order)
- LISTELEM_PTR equal to NULL_PTR
- PRIORITY_8 equal to the priority value used for this packet

⌋

## 7.6 Message Reception

**[SWS_V2xGn_00140] Reception parameters derived from PDU**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If a packet is received, the V2xGn module shall derive the frame type (`EthIfFrameType`) and the EthIfCtrl (`EthIfController`) configured in EthIf via the PDU that is referenced by the `V2xGnRxPdu` which is identified by the given `RxPduId`.⌋

**[SWS_V2xGn_00141] Reception parameters derived from meta data items**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If a packet is received, the V2xGn module shall consume meta data items `PduInfoPtr.MetaDataPtr` configured at the `V2xGnRxPdu` that corresponds to the given `RxPduId` in the following order:

- `ETHERNET_MAC_64` indicating the Physical source address (MAC address in network byte order)
- `BROADCAST_8` indicating a broadcast frame

⌋

**[SWS_V2xGn_CONSTR_00142] Reception PDU constraint for keeping the local buffer**

*Status:* DRAFT

⌈Each `V2xGnRxPdu` shall refer to global PDU that has `KeepLocalPduBuffer` set to `FALSE`.⌋

**[SWS_V2xGn_00038]** ⌈The V2xGn module shall create a unique DecapReqId for each received packet. This DecapReqId is used to track the result of each decapsulation request during which the signature of the received packet is verified.⌋

**[SWS_V2xGn_00039]** ⌈The V2xGn module shall indicate received packets via the V2xBtp_RxIndication() callback to the BTP module.⌋

**[SWS_V2xGn_00084]** ⌈The V2xGn module shall get the reception status of a received packet during the `V2xGn_RxIndication()` from the EthIf module with a call to EthIf_GetBufWRxParams().⌋

**[SWS_V2xGn_20268]**

*Upstream requirements:* SRS_V2X_00010

⌈The V2xGn module shall only use duplicate packet detection as specified in [9] Annex A.2 and A.3.⌋

**[SWS_V2xGn_20181]**

*Upstream requirements:* SRS_V2X_00010

⌈If the V2xGn module detects a collision of the least significant 32 bit of the "Certificate digest" / "hashedId8" with the "Certificate digest" / "hashedId8" of another ITS station, it shall initiate a change of its authorization ticket (pseudonym) only if the certificate corresponding to the other "Certificate digest" / "hashedId8" is valid, and the current authorization ticket was selected according to rules defined in [SWS_V2xM_00201] (that is to say no such collision has already triggered the change to the current authorization ticket).⌋

**[SWS_V2xGn_00127]** ⌈The V2xGn module shall call V2xM_SetGlobalRxParams() with CBR information extracted from the GeoNetworking header.⌋

**[SWS_V2xGn_00131]** ⌈The V2xGn module shall use V2xM_CalcDistance() when calculations of geographical distances are necessary for the V2xGn protocol operations.⌋

## 7.7 State handling of PDUs

PDUs are used to transfer data across the layers in the AUTOSAR communication stack. The `V2xGnConfig` references one `V2xGnRxPdu` and one `V2xGnTxPdu` to interchange data with the lower layer. The V2xGn module request data transmission via the configured `V2xGnTxPdu`, and the module is indicated for data reception via the configured `V2xGnRxPdu`.

**[SWS_V2xGn_00143] Transmission PDU states**

*Status:* DRAFT

*Upstream requirements:* SRS_V2X_26011

⌈The V2xGn module shall maintain a separate state for each transmission PDU used by the V2xGn module (`V2xGnTxPdu`) and distinguish at least the following states:

- `PDU_AVAILABLE` : The PDU for a specific transmission is available and ready to be used (PDU resources are released)
- `PDU_IN_USE` : The PDU for a specific transmission is not available and is already used (PDU resources are valid)

⌋

**[SWS_V2xGn_00144] Starting transmission request**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈The V2xGn module shall request transmission only on PDU in state `PDU_AVAILABLE`, allocate PDU resources and necessary buffer depending on the transmission type (see [SWS_V2xGn_00137], [SWS_V2xGn_00138], [SWS_V2xGn_00139]), enter the state `PDU_IN_USE` and call `LSduR_V2XGnTransmit()`.⌋

**[SWS_V2xGn_00145] Finishing transmission request**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If the transmission confirmation `V2xGn_TxConfirmation()` is called on PDU in state `PDU_IN_USE`, the V2xGn module shall release all PDU resources and enter the state `PDU_AVAILABLE`.⌋

**[SWS_V2xGn_00146] Aborting transmission request**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If the V2xGn module requested to transmit data and the `LSduR_V2XGnTransmit()` returned `E_NOT_OK`, then the V2xGn module shall release all PDU resources and set the state of the affected PDU back to `PDU_AVAILABLE`.⌋

## 7.8 Error Classification

Section "Error Handling" of the document [12] "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

### 7.8.1 Development Errors

**[SWS_V2xGn_00134]** ⌈In case development error detection is enabled for the V2xGn module, the V2xGn module shall check API parameters for validity and report detected errors to the DET.⌋

**[SWS_V2xGn_00041] Definiton of development errors in module V2xGn** ⌈

| *Type of error* | *Related error code* | *Error value* |
|---|---|---|
| API service called with invalid parameter | V2XGN_E_PARAM | 0x01 |
| API service called with invalid pointer | V2XGN_E_PARAM_POINTER | 0x02 |
| API service used without module initialization | V2XGN_E_UNINIT | 0x03 |
| API service called with invalid configuration pointer | V2XGN_E_INIT_FAILED | 0x04 |

⌋

### 7.8.2 Runtime Errors

**[SWS_V2xGn_91000] Definiton of runtime errors in module V2xGn**

*Status:* DRAFT

⌈

| *Type of error* | *Related error code* | *Error value* |
|---|---|---|
| Internal transmission processing aborted | V2XGN_E_TX_INTERNAL_PROCESSING_FAILED | 0x01 |

⌋

**[SWS_V2xGn_00147] Error report for aborting the transmission request**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈The V2xGn shall report the runtime error by calling `Det_ReportRuntimeError(V2XGN_E_TX_INTERNAL_PROCESSING_FAILED)` if one of the following conditions applies:

- A transmission processing is requested from `LSduR_V2xGnTransmit()` and then aborted.

⌋

### 7.8.3 Production Errors

There are no production errors.

### 7.8.4 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 modules are listed:

**[SWS_V2xGn_00042] Definition of imported datatypes of module V2xGn** ⌈

| *Module* | *Header File* | *Imported Type* |
|---|---|---|
| Comtype | ComStack_Types.h | PduIdType |
| | ComStack_Types.h | PduInfoType |
| | ComStack_Types.h | PduLengthType |
| Std | Std_Types.h | Std_ReturnType |
| | Std_Types.h | Std_VersionInfoType |
| V2x_GeneralTypes | Rte_V2xM_Type.h | V2xM_PositionAndTimeType |
| | V2x_GeneralTypes.h | V2x_ChanType |
| | V2x_GeneralTypes.h | V2x_GnAddressType |
| | V2x_GeneralTypes.h | V2x_GnAreaShapeType |
| | V2x_GeneralTypes.h | V2x_GnDestinationAreaType |
| | V2x_GeneralTypes.h | V2x_GnDestinationType |
| | V2x_GeneralTypes.h | V2x_GnLongPositionVectorType |
| | V2x_GeneralTypes.h | V2x_GnPacketTransportType |
| | V2x_GeneralTypes.h | V2x_GnTxResultType |
| | V2x_GeneralTypes.h | V2x_GnUpperProtocolType |
| | V2x_GeneralTypes.h | V2x_PseudonymType |
| | V2x_GeneralTypes.h | V2x_SecProfileType |
| | V2x_GeneralTypes.h | V2x_SecReportType |
| | V2x_GeneralTypes.h | V2x_SecReturnType |
| | V2x_GeneralTypes.h | V2x_TrafficClassIdType |
| V2xBtp | V2xBtp.h | V2xBtp_RxParamsType |
| WEth | WEth_GeneralTypes.h | WEth_BufWRxParamIdType |
| | WEth_GeneralTypes.h | WEth_BufWTxParamIdType |

⌋

## 8.2 Type definitions

### 8.2.1 V2xGn_TxParamsType

**[SWS_V2xGn_00063] Definition of datatype V2xGn_TxParamsType** ⌈

| | | |
|---|---|---|
| ***Name*** | V2xGn_TxParamsType | |
| ***Kind*** | Structure | |
| ***Elements*** | upperProtocol | |
| | ***Type*** | V2x_GnUpperProtocolType |
| | ***Comment*** | The protocol which triggered the request. (e.g. BTP-A or BTP-B) |
| | transportType | |
| | ***Type*** | V2x_GnPacketTransportType |
| | ***Comment*** | Specifies the packet transportation type |
| | destinationAddress | |
| | ***Type*** | V2x_GnAddressType |
| | ***Comment*** | Destination address for GeoUnicast packet |
| | destinationArea | |
| | ***Type*** | V2x_GnDestinationAreaType |
| | ***Comment*** | Destination area for GeoBroadcast/GeoAnycast packet. |
| | destinationType | |
| | ***Type*** | V2x_GnDestinationType |
| | ***Comment*** | Select which destination type (destinationAddress or destinationArea is used for this packet). |
| | secProfile | |
| | ***Type*** | V2x_SecProfileType |
| | ***Comment*** | Parameters depending on the security service. |
| | maxPacketLifetime | |
| | ***Type*** | uint16 |
| | ***Comment*** | Time a packet can be buffered until it reaches the destination. |
| | trafficClassId | |
| | ***Type*** | V2x_TrafficClassIdType |
| | ***Comment*** | Set of parameter specifying the traffic class for the message. |
| ***Description*** | Structure containing parameters for the V2xGn_Transmit() API. | |
| ***Available via*** | V2xGn.h | |

⌋

## 8.3 Function definitions

### 8.3.1 V2xGn_Init

**[SWS_V2xGn_00068] Definition of API function V2xGn_Init** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_Init | |
| ***Syntax*** | `void V2xGn_Init (`<br>`  void* CfgPtr`<br>`)` | |
| ***Service ID [hex]*** | 0x01 | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | CfgPtr | Pointer to configuration set |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | None | |
| ***Description*** | Initializes the V2xGn module. | |
| ***Available via*** | V2xGn.h | |

⌋

**[SWS_V2xGn_00133]** ⌈If development error detection is enabled: The function shall check the parameter `CfgPtr` for containing a valid configuration. If the check fails, the function shall raise the development error `V2XGN_E_INIT_FAILED`.⌋

### 8.3.2 V2xGn_GetVersionInfo

**[SWS_V2xGn_00069] Definition of API function V2xGn_GetVersionInfo** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_GetVersionInfo | |
| ***Syntax*** | `void V2xGn_GetVersionInfo (`<br>`  Std_VersionInfoType* VersionInfoPtr`<br>`)` | |
| ***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. | |
| ***Available via*** | V2xGn.h | |

⌋

**[SWS_V2xGn_00090]** ⌈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 `V2XGN_E_PARAM_POINTER`.⌋

### 8.3.3 V2xGn_V2xM_PreparePseudonymChange

**[SWS_V2xGn_00072] Definition of API function V2xGn_V2xM_Prepare PseudonymChange** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_V2xM_PreparePseudonymChange | |
| ***Syntax*** | `Std_ReturnType V2xGn_V2xM_PreparePseudonymChange ( const V2x_PseudonymType* PseudonymPtr )` | |
| ***Service ID [hex]*** | 0x05 | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | PseudonymPtr | The Pseudonym provided by V2xM |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | Std_ReturnType | `E_OK`: operation successful<br>`E_NOT_OK`: operation failed |
| ***Description*** | This function is called by the V2xM when a Pseudonym Change occurs to prepare the change in every module using it. | |
| ***Available via*** | V2xGn_V2xM.h | |

⌋

**[SWS_V2xGn_00091]**

*Upstream requirements:* SRS_V2X_00176

⌈The function `V2xGn_V2xM_PreparePseudonymChange()` shall prepare the setting of the pseudonym specific part of the GeoNetworking Address being used for packet transmission.⌋

**[SWS_V2xGn_00092]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

**[SWS_V2xGn_00093]** ⌈If development error detection is enabled: the function shall check the parameter `PseudonymPtr` for being valid. If the check fails, the function shall raise the development error `V2XGN_E_PARAM_POINTER`.⌋

Note: This starts a module internal transaction for the pseudonym change. The actual pseudonym change becomes effective only after an API call to `V2xGn_V2xM_CommitPseudonymChange()`.

### 8.3.4 V2xGn_V2xM_CommitPseudonymChange

**[SWS_V2xGn_00111] Definition of API function V2xGn_V2xM_CommitPseudonymChange** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_V2xM_CommitPseudonymChange | |
| ***Syntax*** | `Std_ReturnType V2xGn_V2xM_CommitPseudonymChange ( void )` | |
| ***Service ID [hex]*** | 0x09 | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | None | |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | Std_ReturnType | `E_OK`: operation successful<br>`E_NOT_OK`: operation failed |
| ***Description*** | This function is called by the V2xM when all modules are OK with the pseudonym change and the change is to be committed. | |
| ***Available via*** | V2xGn_V2xM.h | |

⌋

**[SWS_V2xGn_00112]**

*Upstream requirements:* SRS_V2X_00176

⌈The function `V2xGn_V2xM_CommitPseudonymChange()` shall update the pseudonym specific part of the module's GeoNetworking Address.⌋

**[SWS_V2xGn_00113]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

Note: The function requires previous preparation of the pseudonym via an API call to `V2xGn_V2xM_PreparePseudonymChange()`.

### 8.3.5 V2xGn_V2xM_AbortPseudonymChange

**[SWS_V2xGn_00126] Definition of API function V2xGn_V2xM_AbortPseudonym Change** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_V2xM_AbortPseudonymChange | |
| ***Syntax*** | `Std_ReturnType V2xGn_V2xM_AbortPseudonymChange ( void )` | |
| ***Service ID [hex]*** | 0x0a | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | None | |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | Std_ReturnType | `E_OK`: operation successful<br>`E_NOT_OK`: operation failed |
| ***Description*** | 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. | |
| ***Available via*** | V2xGn_V2xM.h | |

⌋

**[SWS_V2xGn_00115]**

*Upstream requirements:* SRS_V2X_00176

⌈The function `V2xGn_V2xM_AbortPseudonymChange()` shall set the state of the module to the state before the pseudonym change has been prepared.⌋

**[SWS_V2xGn_00116]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

Note: The function requires previous preparation of the pseudonym via an API call to `V2xGn_V2xM_PreparePseudonymChange()`.

### 8.3.6 V2xGn_Transmit

**[SWS_V2xGn_00070] Definition of API function V2xGn_Transmit** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_Transmit | |
| ***Syntax*** | `V2x_GnTxResultType V2xGn_Transmit ( const V2xGn_TxParamsType* TxParams, uint16 Length )` | |
| ***Service ID [hex]*** | 0x03 | |
| ***Sync/Async*** | Asynchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | TxParams | Additional transmission parameters |
| | Length | Length of the user data |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | V2x_GnTxResultType | Values specified in the Type could be returned.<br>`V2X_GNTX_ACCEPTED` if no error occurred.<br>`V2X_GNTX_ACCEPTED` transmit has been accepted<br>`V2X_GNTX_E_MAXSDUSIZEOVFL` transmit has been rejected due to maximum length exceedance<br>`V2X_GNTX_E_MAXPACKETLIFETIME` transmit has been rejected due to maximum lifetime exceedance<br>`V2X_GNTX_E_TCID` transmit has been rejected due to unsupported Traffic Class ID<br>`V2X_GNTX_E_MAXGEOAREASIZE` transmit has been rejected due to GeoArea exceeds max size<br>`V2X_GNTX_E_UNSPECIFIED` transmit has been rejected due to unspecified reasons |
| ***Description*** | Is called by V2x_Btp to send a message. | |
| ***Available via*** | V2xGn.h | |

⌋

**[SWS_V2xGn_00095]** ⌈The function `V2xGn_Transmit()` shall transmit a V2X Packet.⌋

**[SWS_V2xGn_00096]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

**[SWS_V2xGn_00098]**

*Status:* OBSOLETE

⌈The function shall return `V2X_GNTX_E_MAXSDUSIZEOVFL` if the call to EthIf_ProvideTxBuffer returns BUFREQ_E_OVFL.⌋

**[SWS_V2xGn_00148] Return value if `LSduR_V2xGnTransmit()` reports `E_NOT_OK`**

*Status:* DRAFT
*Replaces:* SWS_V2xGn_00098
*Upstream requirements:* SRS_V2X_26011

⌈The function shall return `V2X_GNTX_E_UNSPECIFIED` if the call to `LSduR_V2xGnTransmit()` returns `E_NOT_OK`.⌋

**[SWS_V2xGn_00099]** ⌈The function shall return `V2X_GNTX_E_MAXPACKETLIFETIME` if the parameter `TxParams.maxPacketLifetime` is invalid.⌋

**[SWS_V2xGn_00100]** ⌈The function shall return `V2X_GNTX_E_TCID` if the parameter `TxParams.trafficClassId` is invalid.⌋

**[SWS_V2xGn_00101]** ⌈The function shall return `V2X_GNTX_E_MAXGEOAREASIZE` if the parameter `TxParams.destinationType` is V2X_GNDESTINATION_AREA and the parameter `TxParams.destinationArea` is invalid.⌋

## 8.4 Callback notifications

### 8.4.1 V2xGn_V2xM_EncapConfirmation

**[SWS_V2xGn_00118] Definition of callback function V2xGn_V2xM_EncapConfirmation** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_V2xM_EncapConfirmation | |
| ***Syntax*** | `void V2xGn_V2xM_EncapConfirmation ( uint16 EncapReqId )` | |
| ***Service ID [hex]*** | 0x0b | |
| ***Sync/Async*** | Asynchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | EncapReqId | Unique Id of the packet which has been encapsulated with the signature of the transmitting ITS station |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | None | |
| ***Description*** | This function is called by the V2xM when an encapsulation has been finished. | |
| ***Available via*** | V2xGn_V2xM.h | |

⌋

**[SWS_V2xGn_00119]** ⌈The function `V2xGn_V2xM_EncapConfirmation()` shall finalize the packet transmission by transmitting the packet to the lower layer.⌋

**[SWS_V2xGn_00120]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

Note: The function requires previous successful transmission request via the API `V2xGn_Transmit()`.

### 8.4.2 V2xGn_V2xM_DecapConfirmation

**[SWS_V2xGn_00122] Definition of callback function V2xGn_V2xM_DecapConfirmation** ⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_V2xM_DecapConfirmation | |
| ***Syntax*** | `void V2xGn_V2xM_DecapConfirmation ( uint32 DecapReqId, V2x_SecReportType SecReport, uint64 CertificateId, uint32 ItsAid, uint8 SspLength, uint8* SspBits )` | |
| ***Service ID [hex]*** | 0x0c | |
| ***Sync/Async*** | Asynchronous | |
| ***Reentrancy*** | Non Reentrant | |
| ***Parameters (in)*** | DecapReqId | Unique Id of the received packet which has been decapsulated and which signature has been verified |
| | 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 (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | None | |
| ***Description*** | This function is called by the V2xM when a decapsulation has been finished. | |
| ***Available via*** | V2xGn_V2xM.h | |

⌋

**[SWS_V2xGn_00123]** ⌈The function `V2xGn_V2xM_DecapConfirmation()` shall continue the processing of a received packet by proceeding with V2xGn protocol operations.⌋

**[SWS_V2xGn_00124]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

Note: The function requires previous successful reception of a packet via the API `V2xGn_RxIndication()`.

### 8.4.3 V2xGn_RxIndication

**[SWS_V2xGn_91001] Definition of callback function V2xGn_RxIndication**

*Status:* DRAFT

⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_RxIndication (draft) | |
| ***Syntax*** | `void V2xGn_RxIndication (`<br>`  PduIdType RxPduId,`<br>`  const PduInfoType* PduInfoPtr`<br>`)` | |
| ***Service ID [hex]*** | 0x42 | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Reentrant for different PduIds. Non reentrant for the same PduId. | |
| ***Parameters (in)*** | RxPduId | ID of the received PDU. |
| | PduInfoPtr | Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU. |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | None | |
| ***Description*** | Indication of a received PDU from a lower layer communication interface module.<br>**Tags:** atp.Status=draft | |
| ***Available via*** | V2xGn.h | |

⌋

**[SWS_V2xGn_00103]** ⌈The function `V2xGn_RxIndication()` shall get reception parameters of the Wireless Ethernet Driver for a V2X Packet received via an API call to EthIf_GetBufWRxParams.⌋

This is done to get access to the wireless specific reception parameters of the packet that is not available through the `V2xGn_RxIndication()` API.

**[SWS_V2xGn_00104]** ⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init()` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

**[SWS_V2xGn_00105]**

*Status:* OBSOLETE

⌈If development error detection is enabled: the function shall check the parameter `DataPtr` for being valid. If the check fails, the function shall raise the development error `V2XGN_E_PARAM_POINTER`.⌋

**[SWS_V2xGn_00149] Development error handling for invalid `PduInfoPtr`**

*Status:* DRAFT
*Replaces:* SWS_V2xGn_00105
*Upstream requirements:* SRS_V2X_26011

⌈If development error detection is enabled: the function shall check the parameter `PduInfoPtr` for being valid. If the check fails, the function shall raise the development error `V2XGN_E_PARAM_POINTER`.⌋

### 8.4.4 V2xGn_TxConfirmation

**[SWS_V2xGn_91002] Definition of callback function V2xGn_TxConfirmation**

*Status:* DRAFT

⌈

| | | |
|---|---|---|
| ***Service Name*** | V2xGn_TxConfirmation (draft) | |
| ***Syntax*** | `void V2xGn_TxConfirmation ( PduIdType TxPduId, Std_ReturnType result )` | |
| ***Service ID [hex]*** | 0x40 | |
| ***Sync/Async*** | Synchronous | |
| ***Reentrancy*** | Reentrant for different PduIds. Non reentrant for the same PduId. | |
| ***Parameters (in)*** | TxPduId | ID of the PDU that has been transmitted. |
| | result | E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed. |
| ***Parameters (inout)*** | None | |
| ***Parameters (out)*** | None | |
| ***Return value*** | None | |
| ***Description*** | The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.<br>**Tags:** atp.Status=draft | |
| ***Available via*** | V2xGn.h | |

⌋

**[SWS_V2xGn_00150] Development Error handling if `V2xGn_TxConfirmation()` is indicated while V2XGn module is in uninitialized state**

*Status:* DRAFT
*Upstream requirements:* SRS_V2X_26011

⌈If development error detection is enabled: the function shall check that the service `V2xGn_Init` was previously called. If the check fails, the function shall raise the development error `V2XGN_E_UNINIT`.⌋

## 8.5 Scheduled functions

### 8.5.1 V2xGn_MainFunction

**[SWS_V2xGn_00075] Definition of scheduled function V2xGn_MainFunction** ⌈

| | |
|---|---|
| ***Service Name*** | V2xGn_MainFunction |
| ***Syntax*** | `void V2xGn_MainFunction ( void )` |
| ***Service ID [hex]*** | 0x08 |
| ***Description*** | Main function of the V2xGn module for periodical execution of protocol operations. |
| ***Available via*** | SchM_V2xGn.h |

⌋

## 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_V2xGn_00076] Definition of mandatory interfaces required by module V2x Gn** ⌈

| *API Function* | *Header File* | *Description* |
|---|---|---|
| EthIf_GetBufWRxParams | EthIf.h | Read out values related to the receive direction of the transceiver for a received packet. For example, this could be RSSI or Channel belonging to one single packet. |
| EthIf_GetBufWTxParams | EthIf.h | Read out values related to the transmit direction of the transceiver for a transmitted packet. |
| EthIf_SetBufWTxParams | EthIf.h | Set values related to the transmit direction of the transceiver for a specific buffer (packet to be sent). For example, this can be the desired transmit power or the channel belonging to one single packet. |
| LSduR_V2xGnTransmit (draft) | LSduR_V2xGn.h | Requests transmission of a PDU. |
| V2xBtp_CopyTxData | V2xBtp.h | This API is called by the V2xGn module to request the V2xBtp module to copy the transmission data to a specific location. |
| V2xBtp_RxIndication | V2xBtp.h | Via this API, the V2xBtp module gets the data (BTP-PDU) and the GeoNetworking parameters of a received GeoNetworking packet. |
| V2xM_CalcDistance | V2xM.h | Calculates the distance between two geographical points on earth with the assumption that they are on elevation 0. |
| V2xM_GetPositionAndTime | V2xM.h | Provides the instantaneous position information. |
| V2xM_GetRefTimePtr | V2xM.h | Provides a pointer to the time reference of the V2X-Stack. |
| V2xM_TriggerPseudonymChange | V2xM.h | 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. |
| V2xM_V2xGn_GetGlobalTxParams | V2xM_V2xGn.h | This function is called by V2xGn to get the current channel busy percentage for the specified channel |
| V2xM_V2xGn_ReqDecap | V2xM_V2xGn.h | 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. |
| V2xM_V2xGn_ReqEncap | V2xM_V2xGn.h | 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. |
| V2xM_V2xGn_SetGlobalRxParams | V2xM_V2xGn.h | This function is called by V2xGn to set the current channel busy percentage for the specified channel |

⌋

### 8.6.2 Optional Interfaces

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

**[SWS_V2xGn_00077] Definition of optional interfaces requested by module V2x Gn** ⌈

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

⌋

# 9 Sequence diagrams

The following sequence diagrams show the interactions between the V2xGn module and its adjacent modules.

Please note that the sequence diagrams are an extension for illustrational purposes to ease understanding of the specification and to support the functional specification described in chapter 7 and API specification described in chapter 8.

Note that all parameters and return types are left out to make the diagrams easier to read and understand.

## 9.1 V2xGn_RxIndication

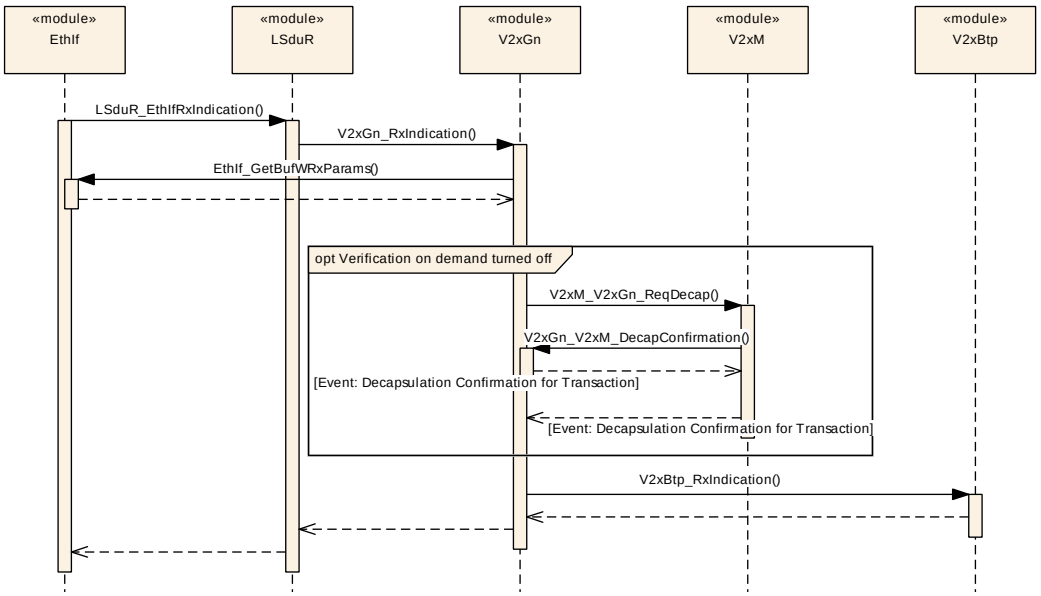


**Figure 9.1: GeoNetworking Packet Reception**

Note: Verification on demand is not anymore supported. The verification of each received packet is mandatory.

## 9.2 V2xGn_Transmit

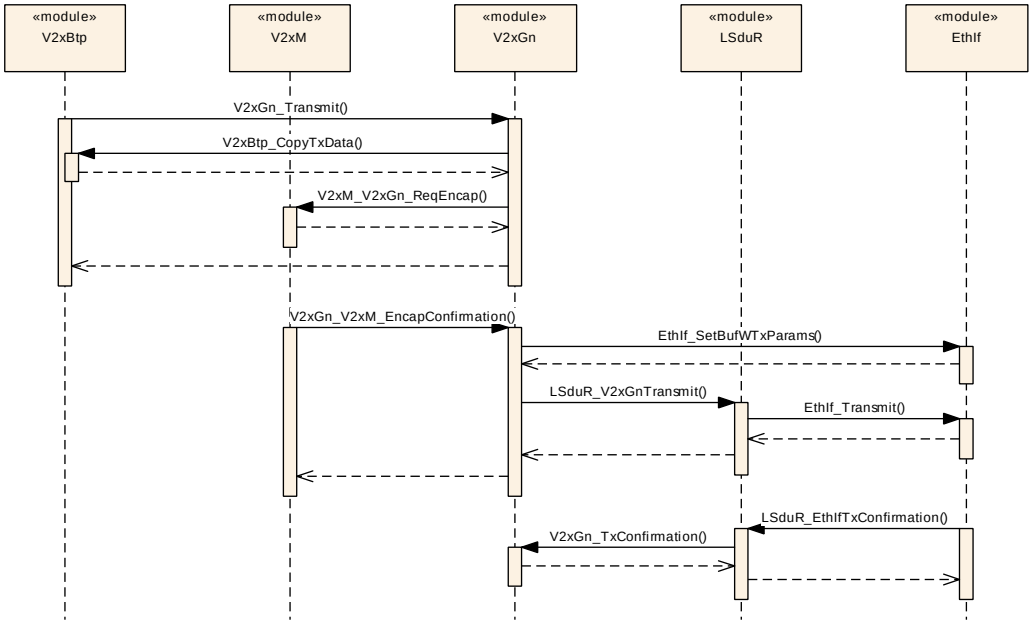


**Figure 9.2: GeoNetworking Packet Transmission**

## 9.3 V2xGn_V2xM_UpdatePseudonym

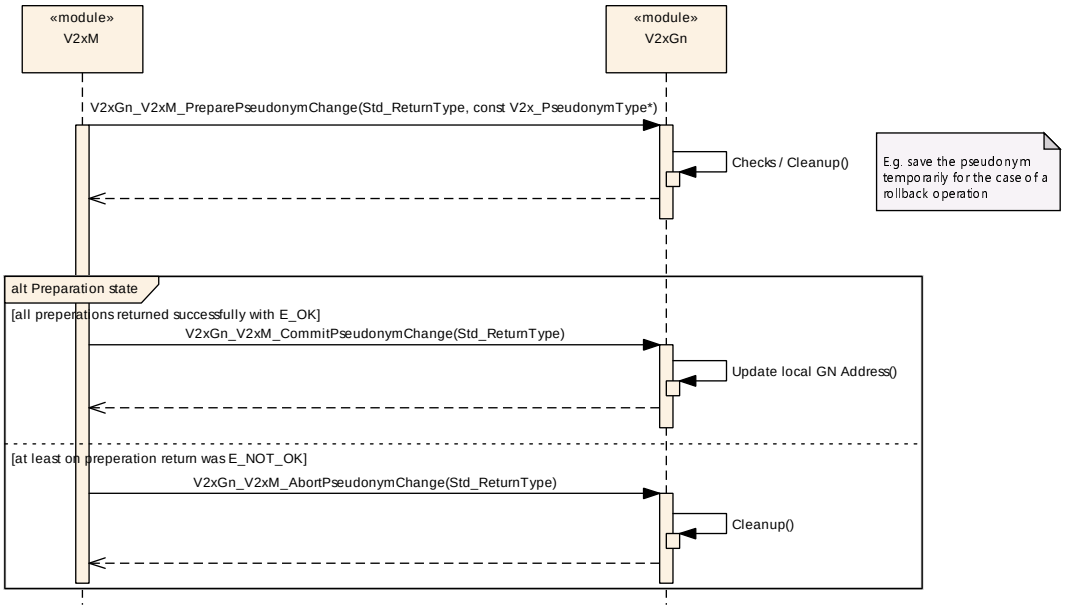


**Figure 9.3: V2x Pseudonym Update**

## 9.4 V2xGn_MainFunction

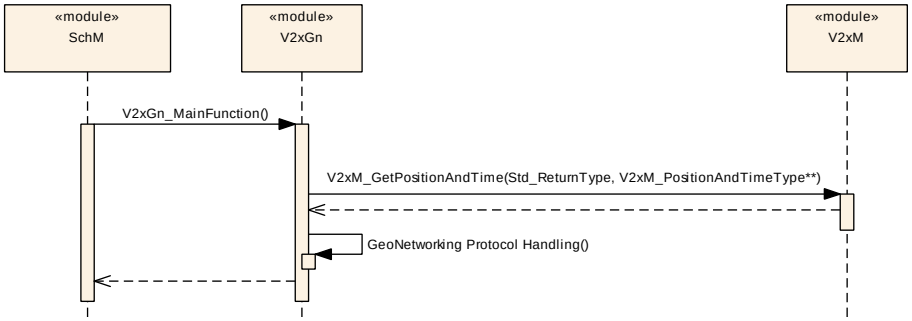


**Figure 9.4: V2xGn Main Function**

# 10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of the module V2xGn.

Chapter 10.2 specifies additionally published information of the module V2xGn.

## 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

**[SWS_V2xGn_00078]**

*Upstream requirements:* SRS_BSW_00345

⌈The V2xGn module only supports VARIANT-PRE-COMPILE.⌋

### 10.1.2 V2xGn

**[ECUC_V2xGn_00001] Definition of EcucModuleDef V2xGn** ⌈

| | |
|---|---|
| **Module Name** | V2xGn |
| **Description** | Configuration of the V2xGn (Vehicle-2-X Geo Networking) module. |
| **Post-Build Variant Support** | false |
| **Supported Config Variants** | VARIANT-PRE-COMPILE |

| **Included Containers** | | |
|---|---|---|
| **Container Name** | **Multiplicity** | **Scope / Dependency** |
| V2xGnBeaconService | 1 | This container contains the GeoNetworking configuration parameters related to the beacon service. |
| V2xGnConfig | 1 | This container contains the configuration parameters and sub containers of the V2xGn module.<br>**Tags:** atp.Status=draft |
| V2xGnGeneral | 1 | This container specifies the general configuration parameters of the V2xGn module. |
| V2xGnPacketForwarding | 1 | This container contains the GeoNetworking configuration parameters related to packet forwarding. |

⌋

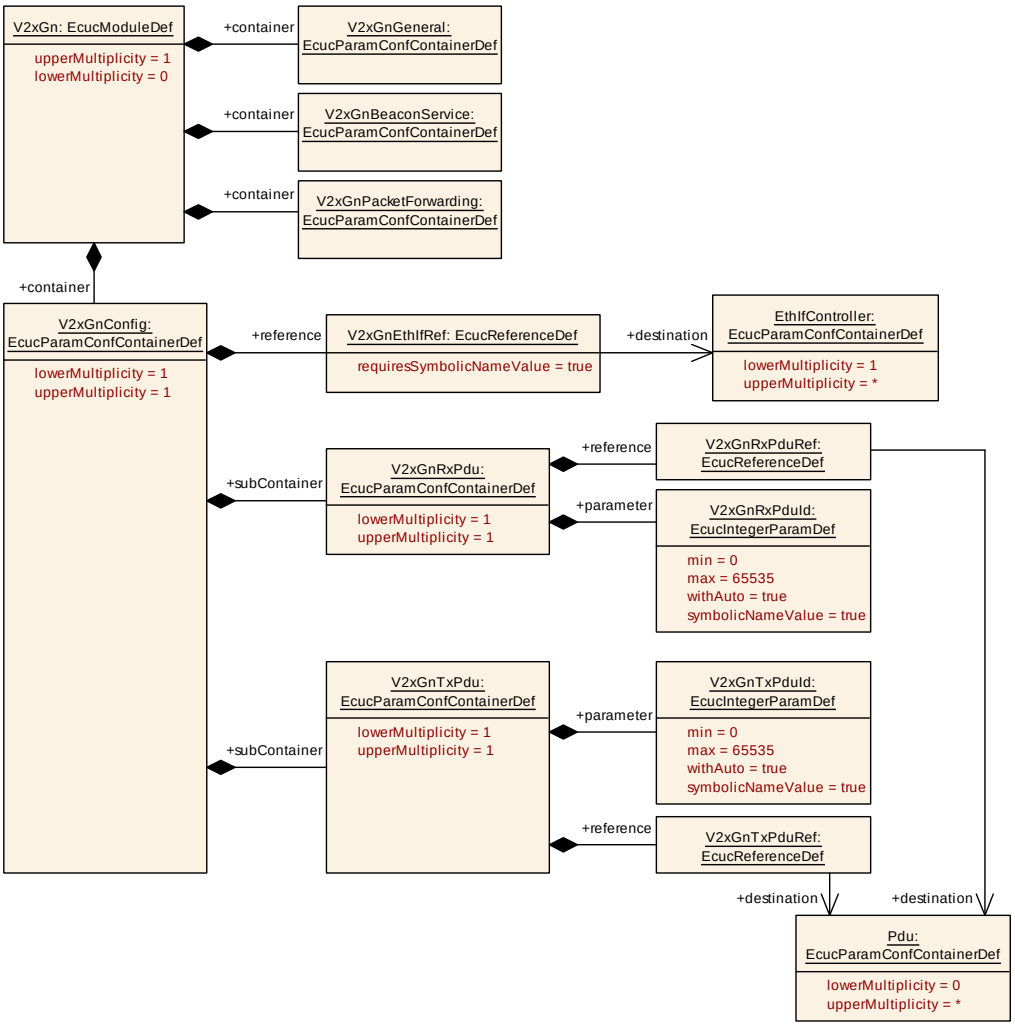


**Figure 10.1: V2xGeoNetworking**

### 10.1.3 V2xGnGeneral

### [ECUC_V2xGn_00002] Definition of EcucParamConfContainerDef V2xGnGeneral ⌈

| | |
|---|---|
| **Container Name** | V2xGnGeneral |
| **Parent Container** | V2xGn |
| **Description** | This container specifies the general configuration parameters of the V2xGn module. |
| **Configuration Parameters** | |

| Included Parameters | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnDevErrorDetect | 1 | [ECUC_V2xGn_00006] |
| V2xGnItsGnLifetimeLocTE | 1 | [ECUC_V2xGn_00016] |
| V2xGnItsGnLocalGnAddr | 1 | [ECUC_V2xGn_00009] |
| V2xGnItsGnMaxGeoNetworkingHeaderSize | 1 | [ECUC_V2xGn_00014] |
| V2xGnItsGnMaxSduSize | 1 | [ECUC_V2xGn_00013] |
| V2xGnItsGnMinUpdateFrequencyEPV | 1 | [ECUC_V2xGn_00011] |
| V2xGnItsGnPaiInterval | 1 | [ECUC_V2xGn_00012] |
| V2xGnItsGnProtocolVersion | 1 | [ECUC_V2xGn_00008] |
| V2xGnItsGnSnDecapResultHandling | 1 | [ECUC_V2xGn_00017] |
| V2xGnItsGnStationType | 1 | [ECUC_V2xGn_00015] |
| V2xGnMainFunctionPeriod | 1 | [ECUC_V2xGn_00018] |
| V2xGnVersionInfoApi | 1 | [ECUC_V2xGn_00005] |

| No Included Containers |
|---|

⌋

### [ECUC_V2xGn_00006] Definition of EcucBooleanParamDef V2xGnDevErrorDetect ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnDevErrorDetect | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Switches the Default Error Tracer (Det) detection and notification ON or OFF.<br>• true: enabled (ON)<br>• 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 | | |

⌋

### [ECUC_V2xGn_00016] Definition of EcucFloatParamDef V2xGnItsGnLifetimeLocTE ⌈

| | |
|---|---|
| **Parameter Name** | V2xGnItsGnLifetimeLocTE |
| **Parent Container** | V2xGnGeneral |
| **Description** | Location table maintenance: Lifetime of an entry in the location table in [s] |
| **Multiplicity** | 1 |
| **Type** | EcucFloatParamDef |

▽

△

| | | | |
|---|---|---|---|
| **Range** | [0 .. 65535] | | |
| **Default value** | 20 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local<br>dependency: shall be a multiple of the V2xGnMainFunctionPeriod | | |

⌋

### [ECUC_V2xGn_00009] Definition of EcucIntegerParamDef V2xGnItsGnLocalGnAddr ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnLocalGnAddr | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | 64bit GeoNetworking Address. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 18446744073709551615 | | |
| **Default value** | 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_V2xGn_00014] Definition of EcucIntegerParamDef V2xGnItsGnMaxGeoNetworkingHeaderSize ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMaxGeoNetworkingHeaderSize | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Maximum size of GeoNetworking header in [Byte]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | 88 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00013] Definition of EcucIntegerParamDef V2xGnItsGnMaxSduSize ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMaxSduSize | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Maximum size of GN-SDU in [Byte]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | 1398 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00011] Definition of EcucFloatParamDef V2xGnItsGnMinUpdateFrequencyEPV ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMinUpdateFrequencyEPV | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Minimum update frequency of ego position vector (EPV) in [s]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | [0 .. 65535] | | |
| **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<br>dependency: shall be a multiple of the V2xGnMainFunctionPeriod | | |

⌋

### [ECUC_V2xGn_00012] Definition of EcucIntegerParamDef V2xGnItsGnPaiInterval ⌈

| | | |
|---|---|---|
| **Parameter Name** | V2xGnItsGnPaiInterval | |
| **Parent Container** | V2xGnGeneral | |
| **Description** | Distance related to the confidence interval for latitude and longitude [m]. Used to determine the PAI. | |
| **Multiplicity** | 1 | |
| **Type** | EcucIntegerParamDef | |
| **Range** | 0 .. 100 | |

▽

△

| | | | |
|---|---|---|---|
| **Default value** | 80 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00008] Definition of EcucIntegerParamDef V2xGnItsGnProtocolVersion ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnProtocolVersion | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | GeoNetworking protocol version as defined in Annex H of [14] | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 255 | | |
| **Default value** | 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_V2xGn_00017] Definition of EcucEnumerationParamDef V2xGnItsGnSnDecapResultHandling ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnSnDecapResultHandling | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Indicates the handling of the V2xM_ReqDecap result code. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucEnumerationParamDef | | |
| **Range** | V2XGN_NON_STRICT_SEC_HANDLING | GN packets that are not correctly verified and decrypted can be passed to the upper protocol entity for further processing. | |
| | V2XGN_STRICT_SEC_HANDLING | Received GN packets that are not correctly verified and decrypted are always dropped. | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00015] Definition of EcucEnumerationParamDef V2xGnItsGnStationType ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnStationType | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Station Type used in GeoNetworking protocol, RoadSideUnit (15) not supported by AUTOSAR. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucEnumerationParamDef | | |
| **Range** | V2XFAC_ST_BUS | – | |
| | V2XFAC_ST_CYCLIST | – | |
| | V2XFAC_ST_HEAVYTRUCK | – | |
| | V2XFAC_ST_LIGHTTRUCK | – | |
| | V2XFAC_ST_MOPED | – | |
| | V2XFAC_ST_MOTORCYCLE | – | |
| | V2XFAC_ST_PASSENGERCAR | – | |
| | V2XFAC_ST_PEDESTRIAN | – | |
| | V2XFAC_ST_SPECIALVEHICLES | – | |
| | V2XFAC_ST_TRAILER | – | |
| | V2XFAC_ST_TRAM | – | |
| | V2XFAC_ST_UNKNOWN | – | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00018] Definition of EcucFloatParamDef V2xGnMainFunctionPeriod ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnMainFunctionPeriod | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Specifies the period of main function V2xGn_MainFunction in seconds. V2xGn does not require this information but the BSW scheduler. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | ]0 .. INF[ | | |
| **Default value** | 0.001 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

**[ECUC_V2xGn_00005] Definition of EcucBooleanParamDef V2xGnVersionInfo Api** ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnVersionInfoApi | | |
| **Parent Container** | V2xGnGeneral | | |
| **Description** | Enable/disables the API for reading the version information of the V2xGn Module.<br>• true: enabled (ON)<br>• 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 | | |

⌋

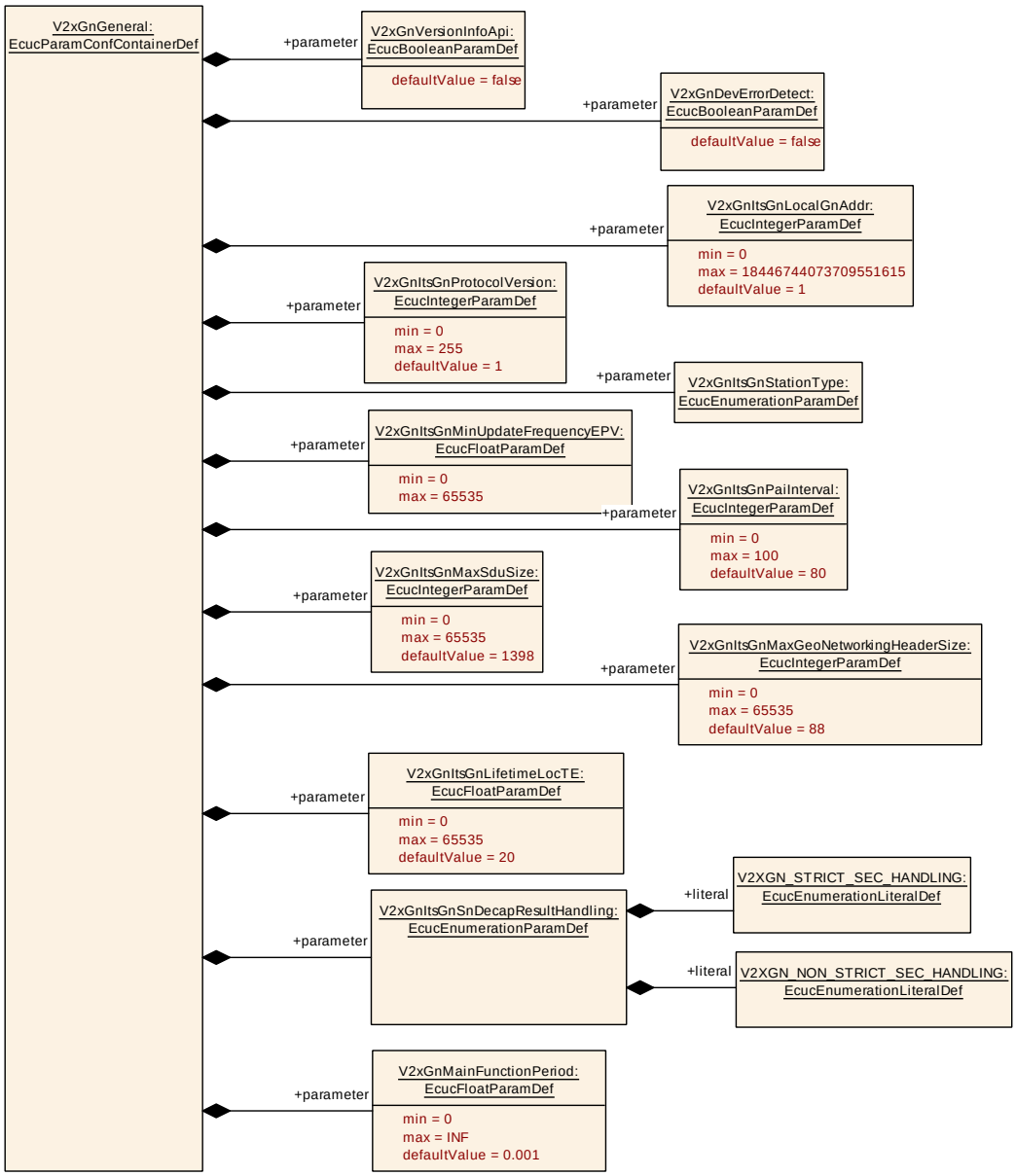


**Figure 10.2: V2XGn_General**

### 10.1.4 V2xGnBeaconService

### [ECUC_V2xGn_00003] Definition of EcucParamConfContainerDef V2xGnBeacon Service ⌈

| | |
|---|---|
| **Container Name** | V2xGnBeaconService |
| **Parent Container** | V2xGn |
| **Description** | This container contains the GeoNetworking configuration parameters related to the beacon service. |
| **Configuration Parameters** | |

| Included Parameters | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnItsGnBeaconServiceMaxJitter | 1 | [ECUC_V2xGn_00021] |
| V2xGnItsGnBeaconServiceRetransmitTimer | 1 | [ECUC_V2xGn_00020] |

| No Included Containers |
|---|

⌋

## [ECUC_V2xGn_00021] Definition of EcucFloatParamDef V2xGnItsGnBeaconServiceMaxJitter ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnBeaconServiceMaxJitter | | |
| **Parent Container** | V2xGnBeaconService | | |
| **Description** | Maximum beacon jitter [s]. The Jitter is used for the beacon retransmission. The actual jitter value is a random number between 0 and V2xGnItsGnBeaconServiceMaxJitter. The function introduces a random component for the timer to avoid synchronization issues among GeoAdhoc routers. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | [0.001 .. INF] | | |
| **Default value** | 0.75 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

## [ECUC_V2xGn_00020] Definition of EcucFloatParamDef V2xGnItsGnBeaconServiceRetransmitTimer ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnBeaconServiceRetransmitTimer | | |
| **Parent Container** | V2xGnBeaconService | | |
| **Description** | Duration of Beacon service retransmit timer [s]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | [0.001 .. INF] | | |
| **Default value** | 3 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local<br>dependency: shall be a multiple of the V2xGnMainFunctionPeriod. | | |

⌋

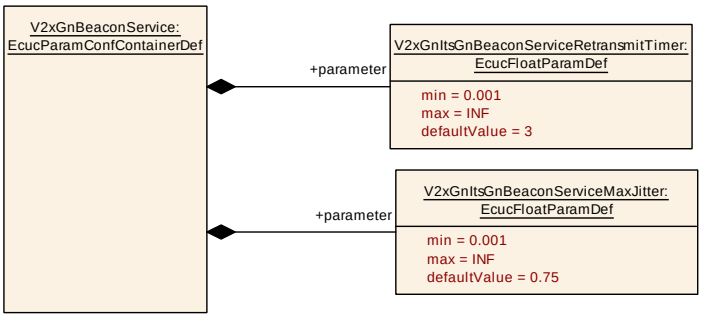


**Figure 10.3: V2xGn_BeaconService**

### 10.1.5 V2xGnPacketForwarding

## [ECUC_V2xGn_00004] Definition of EcucParamConfContainerDef V2xGnPacket Forwarding ⌈

| | |
|---|---|
| **Container Name** | V2xGnPacketForwarding |
| **Parent Container** | V2xGn |
| **Description** | This container contains the GeoNetworking configuration parameters related to packet forwarding. |
| **Configuration Parameters** | |

| **Included Parameters** | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnItsGnBcForwardingPacketBufferSize | 1 | [ECUC_V2xGn_00032] |
| V2xGnItsGnCbfMaxTime | 1 | [ECUC_V2xGn_00029] |
| V2xGnItsGnCbfMinTime | 1 | [ECUC_V2xGn_00028] |
| V2xGnItsGnCbfPacketBufferSize | 1 | [ECUC_V2xGn_00033] |
| V2xGnItsGnDefaultHopLimit | 1 | [ECUC_V2xGn_00022] |
| V2xGnItsGnDefaultMaxCommunicationRange | 1 | [ECUC_V2xGn_00030] |
| V2xGnItsGnDefaultPacketLifetime | 1 | [ECUC_V2xGn_00024] |
| V2xGnItsGnDefaultTrafficClass | 1 | [ECUC_V2xGn_00034] |
| V2xGnItsGnDplLength | 1 | [ECUC_V2xGn_00035] |
| V2xGnItsGnGeoAreaLineForwardingUsed | 1 | [ECUC_V2xGn_00031] |
| V2xGnItsGnMaxGeoAreaSize | 1 | [ECUC_V2xGn_00027] |
| V2xGnItsGnMaxPacketDataRate | 1 | [ECUC_V2xGn_00025] |
| V2xGnItsGnMaxPacketDataRateEmaBeta | 1 | [ECUC_V2xGn_00026] |
| V2xGnItsGnMaxPacketLifetime | 1 | [ECUC_V2xGn_00023] |

| **No Included Containers** |
|---|

⌋

### [ECUC_V2xGn_00032] Definition of EcucIntegerParamDef V2xGnItsGnBcForwardingPacketBufferSize ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnBcForwardingPacketBufferSize | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Size of BC forwarding packet buffer [Byte]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 18446744073709551615 | | |
| **Default value** | 1024000 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00029] Definition of EcucFloatParamDef V2xGnItsGnCbfMaxTime ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnCbfMaxTime | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Maximum duration a GeoNetworking packet shall be buffered in the CBF packet buffer [s] | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | ]0 .. INF[ | | |
| **Default value** | 0.001 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00028] Definition of EcucFloatParamDef V2xGnItsGnCbfMinTime ⌈

| | | |
|---|---|---|
| **Parameter Name** | V2xGnItsGnCbfMinTime | |
| **Parent Container** | V2xGnPacketForwarding | |
| **Description** | Minimum duration a GeoNetworking packet shall be buffered in the CBF packet buffer [s] | |
| **Multiplicity** | 1 | |
| **Type** | EcucFloatParamDef | |
| **Range** | ]0 .. INF[ | |

▽

△

| | | | |
|---|---|---|---|
| **Default value** | 0.001 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00033] Definition of EcucIntegerParamDef V2xGnItsGnCbfPacketBufferSize ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnCbfPacketBufferSize | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Size of CBF packet buffer [Byte] | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 18446744073709551615 | | |
| **Default value** | 256000 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00022] Definition of EcucIntegerParamDef V2xGnItsGnDefaultHopLimit ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnDefaultHopLimit | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Default hop limit indicating the maximum number of hops a packet travels. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 255 | | |
| **Default value** | 10 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00030] Definition of EcucIntegerParamDef V2xGnItsGnDefaultMaxCommunicationRange ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnDefaultMaxCommunicationRange | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Default theoretical maximum communication range [m] | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | 1000 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00024] Definition of EcucFloatParamDef V2xGnItsGnDefaultPacketLifetime ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnDefaultPacketLifetime | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Default packet lifetime [s]. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | [0 .. 6300] | | |
| **Default value** | 60 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00034] Definition of EcucIntegerParamDef V2xGnItsGnDefaultTrafficClass ⌈

| | | |
|---|---|---|
| **Parameter Name** | V2xGnItsGnDefaultTrafficClass | |
| **Parent Container** | V2xGnPacketForwarding | |
| **Description** | Forwarding: Default traffic class | |
| **Multiplicity** | 1 | |
| **Type** | EcucIntegerParamDef | |
| **Range** | 0 .. 255 | |
| **Default value** | 0 | |

▽

△

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

⌋

### [ECUC_V2xGn_00035] Definition of EcucIntegerParamDef V2xGnItsGnDplLength

⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnDplLength | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Length of Duplicate Packet List (DPL) per source (clause A.2 of [18]) | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | 8 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00031] Definition of EcucBooleanParamDef V2xGnItsGnGeoAreaLineForwardingUsed ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnGeoAreaLineForwardingUsed | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Forwarding of GBC/GAC packet if GeoAdhoc router is located outside the destination GeoArea.<br>• true: enabled (ON)<br>• 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 | | |

⌋

### [ECUC_V2xGn_00027] Definition of EcucIntegerParamDef V2xGnItsGnMaxGeoAreaSize ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMaxGeoAreaSize | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Maximum size of the geographical area for a GBC and GAC packet [km2]. If the geographical area size exceeds the maximum value, the GeoNetworking packet shall not be sent (source) and not be forwarded (forwarder). | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 18446744073709551615 | | |
| **Default value** | 80 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00025] Definition of EcucIntegerParamDef V2xGnItsGnMaxPacketDataRate ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMaxPacketDataRate | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Maximum packet data rate for a GeoAdhoc router [Byte/s]. If the mean (EMA) packet data rate a of a GeoAdhoc router exceeds the value, packets from this GeoAdhoc router (source or sender) are not forwarded. | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef | | |
| **Range** | 0 .. 4294967295 | | |
| **Default value** | 100000 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00026] Definition of EcucFloatParamDef V2xGnItsGnMaxPacketDataRateEmaBeta ⌈

| | |
|---|---|
| **Parameter Name** | V2xGnItsGnMaxPacketDataRateEmaBeta |
| **Parent Container** | V2xGnPacketForwarding |
| **Description** | Weight factor for the Exponential Moving Average of the packet data rate PDR in percent. |
| **Multiplicity** | 1 |

▽

△

| | | | |
|---|---|---|---|
| **Type** | EcucFloatParamDef | | |
| **Range** | ]0 .. 1] | | |
| **Default value** | 0.9 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

### [ECUC_V2xGn_00023] Definition of EcucFloatParamDef V2xGnItsGnMaxPacketLifetime ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnItsGnMaxPacketLifetime | | |
| **Parent Container** | V2xGnPacketForwarding | | |
| **Description** | Upper limit of the maximum lifetime [s] | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucFloatParamDef | | |
| **Range** | [0 .. 6300] | | |
| **Default value** | 600 | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: local | | |

⌋

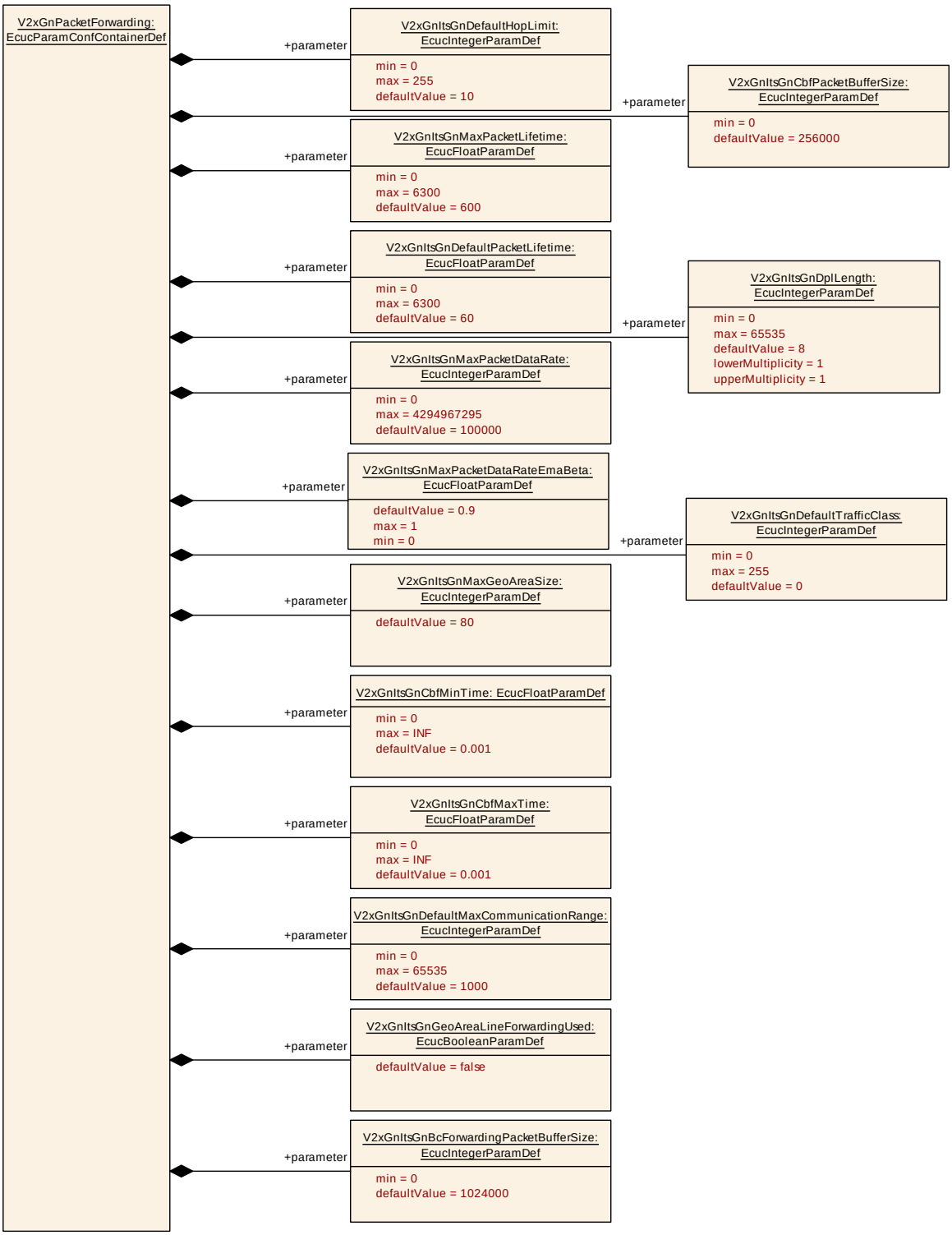


**Figure 10.4: V2xGn_PacketForwarding**

### 10.1.6 V2xGnConfig

## [ECUC_V2xGn_00036] Definition of EcucParamConfContainerDef V2xGnConfig

*Status:* DRAFT

⌈

| | |
|---|---|
| **Container Name** | V2xGnConfig |
| **Parent Container** | V2xGn |
| **Description** | This container contains the configuration parameters and sub containers of the V2xGn module.<br>**Tags:** atp.Status=draft |
| **Configuration Parameters** | |

| **Included Parameters** | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnEthIfRef | 1 | [ECUC_V2xGn_00019] |

| **Included Containers** | | |
|---|---|---|
| **Container Name** | **Multiplicity** | **Scope / Dependency** |
| V2xGnRxPdu | 1 | Represents the received PDU. This PDU is usually linked to the EthIf via LSduR. It consumes meta data items of the types BROADCAST_8 and ETHERNET_MAC_64.<br>**Tags:** atp.Status=draft |
| V2xGnTxPdu | 1 | Represents the transmitted PDU. This PDU is usually linked to the EthIf via LSduR. It produces meta data items of the type ETHERNET_MAC_64.<br>**Tags:** atp.Status=draft |

⌋

## [ECUC_V2xGn_00019] Definition of EcucReferenceDef V2xGnEthIfRef ⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnEthIfRef | | |
| **Parent Container** | V2xGnConfig | | |
| **Description** | This represents the reference to the Ethernet interface taken to transmit the V2X-Packets 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 | | |

⌋

### 10.1.7 V2xGnRxPdu

## [ECUC_V2xGn_00037] Definition of EcucParamConfContainerDef V2xGnRxPdu

*Status:* DRAFT

⌈

| | |
|---|---|
| **Container Name** | V2xGnRxPdu |
| **Parent Container** | V2xGnConfig |
| **Description** | Represents the received PDU. This PDU is usually linked to the EthIf via LSduR. It consumes meta data items of the types BROADCAST_8 and ETHERNET_MAC_64.<br>**Tags:** atp.Status=draft |
| **Configuration Parameters** | |

| **Included Parameters** | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnRxPduId | 1 | [ECUC_V2xGn_00039] |
| V2xGnRxPduRef | 1 | [ECUC_V2xGn_00038] |

| **No Included Containers** |
|---|

⌋

## [ECUC_V2xGn_00039] Definition of EcucIntegerParamDef V2xGnRxPduId

*Status:* DRAFT

⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnRxPduId | | |
| **Parent Container** | V2xGnRxPdu | | |
| **Description** | PDU identifier used for RxIndication from LSduR.<br>**Tags:** atp.Status=draft | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef (Symbolic Name generated for this parameter) | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | – | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: ECU<br>withAuto = true | | |

⌋

### [ECUC_V2xGn_00038] Definition of EcucReferenceDef V2xGnRxPduRef

*Status:* DRAFT

⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnRxPduRef | | |
| **Parent Container** | V2xGnRxPdu | | |
| **Description** | Reference to the global PDU.<br>**Tags:** atp.Status=draft | | |
| **Multiplicity** | 1 | | |
| **Type** | Reference to Pdu | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: ECU | | |

⌋

## 10.1.8 V2xGnTxPdu

### [ECUC_V2xGn_00040] Definition of EcucParamConfContainerDef V2xGnTxPdu

*Status:* DRAFT

⌈

| | |
|---|---|
| **Container Name** | V2xGnTxPdu |
| **Parent Container** | V2xGnConfig |
| **Description** | Represents the transmitted PDU. This PDU is usually linked to the EthIf via LSduR. It produces meta data items of the type ETHERNET_MAC_64.<br>**Tags:** atp.Status=draft |
| **Configuration Parameters** | |

| **Included Parameters** | | |
|---|---|---|
| **Parameter Name** | **Multiplicity** | **ECUC ID** |
| V2xGnTxPduId | 1 | [ECUC_V2xGn_00041] |
| V2xGnTxPduRef | 1 | [ECUC_V2xGn_00042] |

| **No Included Containers** |
|---|

⌋

### [ECUC_V2xGn_00041] Definition of EcucIntegerParamDef V2xGnTxPduId

*Status:* DRAFT

⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnTxPduId | | |
| **Parent Container** | V2xGnTxPdu | | |
| **Description** | PDU identifier used for TxConfirmation from LSduR.<br>**Tags:** atp.Status=draft | | |
| **Multiplicity** | 1 | | |
| **Type** | EcucIntegerParamDef (Symbolic Name generated for this parameter) | | |
| **Range** | 0 .. 65535 | | |
| **Default value** | – | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: ECU<br>withAuto = true | | |

⌋

### [ECUC_V2xGn_00042] Definition of EcucReferenceDef V2xGnTxPduRef

*Status:* DRAFT

⌈

| | | | |
|---|---|---|---|
| **Parameter Name** | V2xGnTxPduRef | | |
| **Parent Container** | V2xGnTxPdu | | |
| **Description** | Reference to the global PDU.<br>**Tags:** atp.Status=draft | | |
| **Multiplicity** | 1 | | |
| **Type** | Reference to Pdu | | |
| **Post-Build Variant Value** | false | | |
| **Value Configuration Class** | **Pre-compile time** | X | All Variants |
| | **Link time** | – | |
| | **Post-build time** | – | |
| **Scope / Dependency** | scope: ECU | | |

⌋

## 10.2 Published Information

For details refer to the chapter 10.3 “Published Information” in the General Specification on Basic Software modules [12].

# A Not applicable requirements

**[SWS_V2xGn_NA_00001]**

*Upstream requirements:* SRS_V2X_00451, SRS_V2X_00322, SRS_V2X_00242, SRS_V2X_00391, SRS_V2X_00232, SRS_V2X_00245

⌈This requirement references all not applicable access layer requirements⌋

**[SWS_V2xGn_NA_00002]**

*Upstream requirements:* SRS_V2X_00711, SRS_V2X_00291, SRS_V2X_00318, SRS_V2X_00741, SRS_V2X_00301

⌈This requirement references all not applicable facility layer requirements⌋

**[SWS_V2xGn_NA_00003]**

*Upstream requirements:* SRS_V2X_00405, SRS_V2X_00413, SRS_V2X_00163, SRS_V2X_00412, SRS_V2X_00407, SRS_V2X_00406, SRS_V2X_00184, SRS_V2X_00174

⌈This requirement references all not applicable security requirements⌋

**[SWS_V2xGn_NA_00004]**

*Upstream requirements:* SRS_V2X_00190, SRS_V2X_00193, SRS_V2X_00207, SRS_V2X_00214, SRS_V2X_00693, SRS_V2X_00189, SRS_V2X_00323, SRS_V2X_00511

⌈This requirement references all not applicable other requirements from SRS V2X⌋

# B History of Specification Items

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

## B.1 Specification Item History of this document compared to AUTOSAR R24-11.

### B.1.1 Added Specification Items in R24-11

| Number | Heading |
|---|---|
| [ECUC_V2xGn_00036] | Definition of EcucParamConfContainerDef V2xGnConfig |
| [ECUC_V2xGn_00037] | Definition of EcucParamConfContainerDef V2xGnRxPdu |
| [ECUC_V2xGn_00038] | Definition of EcucReferenceDef V2xGnRxPduRef |
| [ECUC_V2xGn_00039] | Definition of EcucIntegerParamDef V2xGnRxPduId |
| [ECUC_V2xGn_00040] | Definition of EcucParamConfContainerDef V2xGnTxPdu |
| [ECUC_V2xGn_00041] | Definition of EcucIntegerParamDef V2xGnTxPduId |
| [ECUC_V2xGn_00042] | Definition of EcucReferenceDef V2xGnTxPduRef |
| [SWS_V2xGn_00136] | Usage of `LSduR_V2xGnTransmit()` in context of `V2xGn_Transmit()` |
| [SWS_V2xGn_00137] | Transmission request with direct data provision |
| [SWS_V2xGn_00138] | Meta data handling while containing headers |
| [SWS_V2xGn_00139] | Meta data handling while not containing headers |
| [SWS_V2xGn_00140] | Reception parameters derived from PDU |
| [SWS_V2xGn_00141] | Reception parameters derived from meta data items |
| [SWS_V2xGn_00143] | Transmission PDU states |
| [SWS_V2xGn_00144] | Starting transmission request |
| [SWS_V2xGn_00145] | Finishing transmission request |
| [SWS_V2xGn_00146] | Aborting transmission request |
| [SWS_V2xGn_00147] | Error report for aborting the transmission request |
| [SWS_V2xGn_00148] | Return value if `LSduR_V2xGnTransmit()` reports `E_NOT_OK` |
| [SWS_V2xGn_00149] | Development error handling for invalid `PduInfoPtr` |
| [SWS_V2xGn_00150] | Development Error handling if `V2xGn_TxConfirmation()` is indicated while V2XGn module is in uninitialized state |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_91000] | Definiton of runtime errors in module V2xGn |
| [SWS_V2xGn_91001] | Definition of callback function V2xGn_RxIndication |
| [SWS_V2xGn_91002] | Definition of callback function V2xGn_TxConfirmation |

**Table B.1: Added Specification Items in R24-11**

### B.1.2 Changed Specification Items in R24-11

| Number | Heading |
|---|---|
| [ECUC_V2xGn_-00001] | Definition of EcucModuleDef V2xGn |
| [ECUC_V2xGn_-00002] | Definition of EcucParamConfContainerDef V2xGnGeneral |
| [ECUC_V2xGn_-00019] | Definition of EcucReferenceDef V2xGnEthIfRef |
| [SWS_V2xGn_00035] | |
| [SWS_V2xGn_00042] | Definition of imported datatypes of module V2xGn |
| [SWS_V2xGn_00076] | Definition of mandatory interfaces required by module V2xGn |
| [SWS_V2xGn_00082] | |
| [SWS_V2xGn_00098] | |
| [SWS_V2xGn_00105] | |

**Table B.2: Changed Specification Items in R24-11**

### B.1.3 Deleted Specification Items in R24-11

| Number | Heading |
|---|---|
| [SWS_V2xGn_00071] | Definition of callback function V2xGn_RxIndication |

**Table B.3: Deleted Specification Items in R24-11**

## B.2 Constraint Item History of this document compared to AUTOSAR R24-11.

### B.2.1 Added Constraints in R24-11

| Number | Heading |
|---|---|
| [SWS_V2xGn_CONSTR_00142] | Reception PDU constraint for keeping the local buffer |

**Table B.4: Added Constraints in R24-11**

### B.2.2 Changed Constraints in R24-11

none

### B.2.3 Deleted Constraints in R24-11

none

## B.3 Specification Item History of this document compared to AUTOSAR R23-11.

### B.3.1 Added Specification Items in R23-11

| Number | Heading |
|---|---|
| [SWS_V2xGn_00012] | |
| [SWS_V2xGn_00013] | |
| [SWS_V2xGn_00020] | |
| [SWS_V2xGn_00022] | |
| [SWS_V2xGn_00023] | |
| [SWS_V2xGn_00026] | |
| [SWS_V2xGn_00028] | |
| [SWS_V2xGn_00034] | |
| [SWS_V2xGn_00035] | |
| [SWS_V2xGn_00038] | |
| [SWS_V2xGn_00039] | |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_00041] | Definiton of development errors in module V2xGn |
| [SWS_V2xGn_00042] | Definition of imported datatypes of module V2xGn |
| [SWS_V2xGn_00063] | Definition of datatype V2xGn_TxParamsType |
| [SWS_V2xGn_00068] | Definition of API function V2xGn_Init |
| [SWS_V2xGn_00069] | Definition of API function V2xGn_GetVersionInfo |
| [SWS_V2xGn_00070] | Definition of API function V2xGn_Transmit |
| [SWS_V2xGn_00071] | Definition of callback function V2xGn_RxIndication |
| [SWS_V2xGn_00072] | Definition of API function V2xGn_V2xM_PreparePseudonymChange |
| [SWS_V2xGn_00075] | Definition of scheduled function V2xGn_MainFunction |
| [SWS_V2xGn_00076] | Definition of mandatory interfaces in module V2xGn |
| [SWS_V2xGn_00077] | Definition of optional interfaces in module V2xGn |
| [SWS_V2xGn_00078] | |
| [SWS_V2xGn_00081] | |
| [SWS_V2xGn_00082] | |
| [SWS_V2xGn_00083] | |
| [SWS_V2xGn_00084] | |
| [SWS_V2xGn_00090] | |
| [SWS_V2xGn_00091] | |
| [SWS_V2xGn_00092] | |
| [SWS_V2xGn_00093] | |
| [SWS_V2xGn_00095] | |
| [SWS_V2xGn_00096] | |
| [SWS_V2xGn_00098] | |
| [SWS_V2xGn_00099] | |
| [SWS_V2xGn_00100] | |
| [SWS_V2xGn_00101] | |
| [SWS_V2xGn_00103] | |
| [SWS_V2xGn_00104] | |
| [SWS_V2xGn_00105] | |
| [SWS_V2xGn_00111] | Definition of API function V2xGn_V2xM_CommitPseudonymChange |
| [SWS_V2xGn_00112] | |
| [SWS_V2xGn_00113] | |
| [SWS_V2xGn_00115] | |
| [SWS_V2xGn_00116] | |
| [SWS_V2xGn_00118] | Definition of callback function V2xGn_V2xM_EncapConfirmation |
| [SWS_V2xGn_00119] | |
| [SWS_V2xGn_00120] | |
| [SWS_V2xGn_00122] | Definition of callback function V2xGn_V2xM_DecapConfirmation |
| [SWS_V2xGn_00123] | |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_00124] | |
| [SWS_V2xGn_00126] | Definition of API function V2xGn_V2xM_AbortPseudonymChange |
| [SWS_V2xGn_00127] | |
| [SWS_V2xGn_00128] | |
| [SWS_V2xGn_00129] | |
| [SWS_V2xGn_00130] | |
| [SWS_V2xGn_00131] | |
| [SWS_V2xGn_00133] | |
| [SWS_V2xGn_00134] | |
| [SWS_V2xGn_00135] | |
| [SWS_V2xGn_00269] | |
| [SWS_V2xGn_20169] | |
| [SWS_V2xGn_20181] | |
| [SWS_V2xGn_20250] | |
| [SWS_V2xGn_20251] | |
| [SWS_V2xGn_20252] | |
| [SWS_V2xGn_20255] | |
| [SWS_V2xGn_20258] | |
| [SWS_V2xGn_20259] | |
| [SWS_V2xGn_20260] | |
| [SWS_V2xGn_20262] | |
| [SWS_V2xGn_20263] | |
| [SWS_V2xGn_20264] | |
| [SWS_V2xGn_20265] | |
| [SWS_V2xGn_20266] | |
| [SWS_V2xGn_20267] | |
| [SWS_V2xGn_20268] | |
| [SWS_V2xGn_20270] | |
| [SWS_V2xGn_20401] | |
| [SWS_V2xGn_20414] | |
| [SWS_V2xGn_20416] | |
| [SWS_V2xGn_NA_-00001] | |
| [SWS_V2xGn_NA_-00002] | |
| [SWS_V2xGn_NA_-00003] | |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_NA_-00004] | |

**Table B.5: Added Specification Items in R23-11**

### B.3.2 Changed Specification Items in R23-11

none

### B.3.3 Deleted Specification Items in R23-11

none

## B.4 Specification Item History of this document compared to AUTOSAR R22-11.

### B.4.1 Added Specification Items in R22-11

| Number | Heading |
|---|---|
| [SWS_V2xGn_00012] | |
| [SWS_V2xGn_00013] | |
| [SWS_V2xGn_00020] | |
| [SWS_V2xGn_00022] | |
| [SWS_V2xGn_00023] | |
| [SWS_V2xGn_00026] | |
| [SWS_V2xGn_00028] | |
| [SWS_V2xGn_00034] | |
| [SWS_V2xGn_00035] | |
| [SWS_V2xGn_00036] | |
| [SWS_V2xGn_00038] | |
| [SWS_V2xGn_00039] | |
| [SWS_V2xGn_00041] | |
| [SWS_V2xGn_00042] | |
| [SWS_V2xGn_00063] | |
| [SWS_V2xGn_00068] | |
| [SWS_V2xGn_00069] | |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_00070] | |
| [SWS_V2xGn_00071] | |
| [SWS_V2xGn_00072] | |
| [SWS_V2xGn_00074] | |
| [SWS_V2xGn_00075] | |
| [SWS_V2xGn_00076] | |
| [SWS_V2xGn_00077] | |
| [SWS_V2xGn_00078] | |
| [SWS_V2xGn_00081] | |
| [SWS_V2xGn_00082] | |
| [SWS_V2xGn_00083] | |
| [SWS_V2xGn_00084] | |
| [SWS_V2xGn_00090] | |
| [SWS_V2xGn_00091] | |
| [SWS_V2xGn_00092] | |
| [SWS_V2xGn_00093] | |
| [SWS_V2xGn_00095] | |
| [SWS_V2xGn_00096] | |
| [SWS_V2xGn_00098] | |
| [SWS_V2xGn_00099] | |
| [SWS_V2xGn_00100] | |
| [SWS_V2xGn_00101] | |
| [SWS_V2xGn_00103] | |
| [SWS_V2xGn_00104] | |
| [SWS_V2xGn_00105] | |
| [SWS_V2xGn_00107] | |
| [SWS_V2xGn_00111] | |
| [SWS_V2xGn_00112] | |
| [SWS_V2xGn_00113] | |
| [SWS_V2xGn_00115] | |
| [SWS_V2xGn_00116] | |
| [SWS_V2xGn_00118] | |
| [SWS_V2xGn_00119] | |
| [SWS_V2xGn_00120] | |
| [SWS_V2xGn_00122] | |
| [SWS_V2xGn_00123] | |
| [SWS_V2xGn_00124] | |
| [SWS_V2xGn_00126] | |
| [SWS_V2xGn_00127] | |

▽

△

| Number | Heading |
|---|---|
| [SWS_V2xGn_00128] | |
| [SWS_V2xGn_00129] | |
| [SWS_V2xGn_00130] | |
| [SWS_V2xGn_00131] | |
| [SWS_V2xGn_00133] | |
| [SWS_V2xGn_00134] | |
| [SWS_V2xGn_00269] | |
| [SWS_V2xGn_20169] | |
| [SWS_V2xGn_20181] | |
| [SWS_V2xGn_20250] | |
| [SWS_V2xGn_20251] | |
| [SWS_V2xGn_20252] | |
| [SWS_V2xGn_20255] | |
| [SWS_V2xGn_20258] | |
| [SWS_V2xGn_20259] | |
| [SWS_V2xGn_20260] | |
| [SWS_V2xGn_20262] | |
| [SWS_V2xGn_20263] | |
| [SWS_V2xGn_20264] | |
| [SWS_V2xGn_20265] | |
| [SWS_V2xGn_20266] | |
| [SWS_V2xGn_20267] | |
| [SWS_V2xGn_20268] | |
| [SWS_V2xGn_20270] | |
| [SWS_V2xGn_20401] | |
| [SWS_V2xGn_20414] | |
| [SWS_V2xGn_20416] | |
| [SWS_V2xGn_NA_00001] | |
| [SWS_V2xGn_NA_00002] | |
| [SWS_V2xGn_NA_00003] | |
| [SWS_V2xGn_NA_00004] | |

**Table B.6: Added Specification Items in R22-11**

### B.4.2 Changed Specification Items in R22-11

none

### B.4.3 Deleted Specification Items in R22-11

none

## B.5 Constraint Item History of this document compared to AUTOSAR R22-11.

### B.5.1 Added Constraints in R22-11

none

### B.5.2 Changed Constraints in R22-11

none

### B.5.3 Deleted Constraints in R22-11

none